

3



USER MANUAL

grandMA3



Table of contents

1.	About the Help	31
1.1.	Open the Help in the Console	32
1.2.	Open the Help in the Web	34
1.3.	Open the Help as PDF	35
1.4.	Navigate in the Help	36
2.	Device Overview	39
2.1.	grandMA3 consoles	40
2.1.1.	grandMA3 full-size	41
2.1.2.	grandMA3 full-size CRV	43
2.1.3.	grandMA3 light	45
2.1.4.	grandMA3 light CRV	47
2.1.5.	grandMA3 compact XT	49
2.1.6.	grandMA3 compact	50
2.2.	grandMA3 extension	51
2.3.	grandMA3 replay unit	53
2.4.	grandMA3 processing units	54
2.5.	grandMA3 nodes	55
2.6.	grandMA3 nodes DIN-Rail	57
2.7.	grandMA3 I/O Node	59
2.8.	grandMA3 I/O Node DIN-Rail	60
2.9.	grandMA3 onPC command wing XT	61
2.10.	grandMA3 onPC command wing	63
2.11.	grandMA3 onPC fader wing	65
2.12.	grandMA3 onPC rack-unit	66
2.13.	Screen Allocation	67
2.14.	Keyboard Shortcuts	73
2.14.1.	Framework Shortcuts	82
2.15.	Keys	83
2.15.1.	[Dot] Key	91
2.15.2.	<<< [GoFastBackward] Black Key	94
2.15.3.	>>> [GoFastForward] Flash Key	96
2.15.4.	- [Minus] Key	98



2.15.5. + [Plus] Key	100
2.15.6. / [Slash] * [Asterisk/Multiply] Key	102
2.15.7. At Key	104
2.15.8. Assign Key	107
2.15.9. Align Key	110
2.15.10. Blind Key	113
2.15.11. Clear Key	115
2.15.12. Channel Key	117
2.15.13. Copy Key	120
2.15.14. Ctrl	123
2.15.15. Cue Key	124
2.15.16. Down Key	127
2.15.17. Delete Key	129
2.15.18. Esc Key	132
2.15.19. Edit Key	134
2.15.20. Freeze Key	136
2.15.21. Full Key	138
2.15.22. Fixture Key	140
2.15.23. Group Key	142
2.15.24. Goto Key	145
2.15.25. Go+ [large] Key	147
2.15.26. Go- [large] Key	149
2.15.27. Go+ Temp Key	151
2.15.28. Go- Top Key	154
2.15.29. Help Key	156
2.15.30. Highlt [Highlight] Key	158
2.15.31. If Key	160
2.15.32. Learn Rate1 Key	163
2.15.33. List Key	165
2.15.34. MA Key	167
2.15.35. Menu Key	169
2.15.36. Move Key	171
2.15.37. Next Key	173
2.15.38. Numeric Keys Arrows	175



2.15.39. Oops Key	177
2.15.40. On Key	179
2.15.41. Off Key	181
2.15.42. Pause [large] Key	183
2.15.43. Pause Fix Key	185
2.15.44. Page+ Key	188
2.15.45. Page- Key	190
2.15.46. Please Key	192
2.15.47. Power Key	194
2.15.48. Preset Key	196
2.15.49. Prvw [Preview] Key	199
2.15.50. Prev [Previous] Key	201
2.15.51. Select Key	203
2.15.52. SelFix [SelectFixture] Key	205
2.15.53. Sequ [Sequence] Key	207
2.15.54. Set Key	209
2.15.55. Solo Key	211
2.15.56. Stomp Key	213
2.15.57. Store Key	215
2.15.58. Thru Key	217
2.15.59. Time Key	219
2.15.60. Up Key	222
2.15.61. Update Key	224
2.15.62. U1 Key	226
2.15.63. U2 Key	227
2.15.64. X1 Clone Key	228
2.15.65. X2 Link Key	230
2.15.66. X3 Grid Key	232
2.15.67. X4 Layout Key	234
2.15.68. X5 Step Key	237
2.15.69. X6 TC Key	239
2.15.70. X7 View Key	242
2.15.71. X8 DMX Key	245
2.15.72. X9 Key	247

2.15.73. X10 key	249
2.15.74. X11 Key	251
2.15.75. X12 Key	253
2.15.76. X13 Phaser Key	255
2.15.77. X14 Macro Key	257
2.15.78. X15 Page Key	260
2.15.79. X16 Exec Key	262
2.15.80. Xkeys Key	265
2.16. Control Elements	267
2.16.1. Command Area	268
2.16.2. Master Area	270
2.16.3. Custom Area	273
2.16.4. Dual Encoders	274
2.16.5. Level Wheel	277
2.16.6. Grand Master	279
2.16.7. Executor Elements	282
2.17. Connector Pin Assignment	284
2.18. UPS Battery	288
3. System Overview	291
3.1. Standalone Device	292
3.2. Locally Networked Devices	293
3.3. World Server	294
3.4. Parameters	295
3.4.1. Calculate Parameters	297
3.4.2. Expand the Amount of Parameters	299
4. First Steps	301
4.1. Unpack the Device	302
4.2. Check Scope of Delivery	303
4.3. Position the Device	306
4.4. Connect Power	307
4.5. Connect Desk Light	309
4.6. Connect External Screens	310
4.7. Connect USB Devices	316



4.8.	Connect DMX	318
4.9.	Connect Audio In	320
4.10.	Connect MIDI	321
4.11.	Connect LTC	323
4.12.	Connect Ethernet	325
4.13.	Connect DC Remote In	327
4.14.	Connect grandMA3 extension	330
4.15.	Connect grandMA3 fader wing	333
4.16.	Turn on the Device for the First Time	335
5.	grandMA3 onPC	339
5.1.	System Requirements grandMA3 onPC	340
5.2.	Windows™® Installation	342
5.3.	Optimize Windows™®	351
5.4.	macOS® Installation	355
5.5.	Optimize macOS®	361
5.6.	grandMA3 onPC Terminal App	368
5.7.	onPC Settings	370
6.	Show File Handling	372
6.1.	Load a Show File	376
6.2.	Save a Show File	378
6.3.	New Show File	383
6.4.	Backup, Demo and Template Show Files	385
6.5.	Organize Show Files	386
7.	Workspace	392
7.1.	User Interface	393
7.1.1.	Configuration of Displays	394
7.1.2.	Desk Lock	399
7.1.3.	User-Defined Area	401
7.1.4.	Command Line	403
7.1.5.	Control Bar	410
7.1.6.	View Bar	413
7.1.7.	Tables in General	414
7.1.8.	Trackpad Window	415



7.2.	Gestures	419
7.3.	Command Area	423
7.4.	Oops Overlay	425
7.5.	Master Controls	431
7.6.	Playback Controls	433
7.7.	Displays in grandMA3 onPC	434
7.8.	Encoder Bar	436
	7.8.1. Feature Group Control Bar	438
	7.8.2. Encoder Toolbar	440
7.9.	Calculator	444
7.10.	Playback Bar	455
7.11.	Command Wing Bar	457
7.12.	Colors	459
	7.12.1. System Colors	460
	7.12.2. Markers	469
	7.12.3. Color Theme	472
	7.12.4. Compare Tool	475
8.	Function of the Command Line	481
8.1.	Syntax Rules	482
8.2.	Parent Child Concept	484
8.3.	General Keywords	492
	8.3.1. ; [Semicolon] Character	493
	8.3.2. / [Slash] Character	494
	8.3.3. . [Dot] Character	495
	8.3.4. = [Equal] Character	496
	8.3.5. <<< [GoFastBackward] Keyword	497
	8.3.6. >>> [GoFastForward] Keyword	498
	8.3.7. - [Minus] Keyword	499
	8.3.8. * [Asterisk] Character	501
	8.3.9. % [Percent] Keyword	502
	8.3.10. + [Plus] Keyword	503
	8.3.11. Absolute Keyword	504
	8.3.12. Acceleration Keyword	505
	8.3.13. Action Keyword	506



8.3.14. ActivationGroup Keyword	507
8.3.15. Agenda Keyword	508
8.3.16. Align Keyword	509
8.3.17. AlignTransition Keyword	510
8.3.18. Appearance Keyword	511
8.3.19. Assign Keyword	512
8.3.20. At Keyword	514
8.3.21. Attribute Keyword	516
8.3.22. AutoCreate Keyword	518
8.3.23. Black Keyword	520
8.3.24. Blind Keyword	521
8.3.25. Block Keyword	522
8.3.26. BPM Keyword	523
8.3.27. Call Keyword	524
8.3.28. CancelSoftwareUpdate Keyword	525
8.3.29. Camera Keyword	526
8.3.30. Capture Keyword	527
8.3.31. ChangeDestination Keyword	528
8.3.32. Channel Keyword	530
8.3.33. ChannelSet Keyword	531
8.3.34. Chat Keyword	532
8.3.35. ChatJoin Keyword	533
8.3.36. ChatLeave Keyword	534
8.3.37. CleanUp Keyword	535
8.3.38. Clear Keyword	536
8.3.39. ClearActive Keyword	537
8.3.40. ClearAll Keyword	538
8.3.41. ClearSelection Keyword	539
8.3.42. Clone Keyword	540
8.3.43. CommandDelay Keyword	542
8.3.44. Collect Keyword	543
8.3.45. Collection Keyword	544
8.3.46. Color Keyword	545
8.3.47. ColorDefinition Keyword	546



8.3.48. ColorTheme Keyword	547
8.3.49. Configuration Keyword	548
8.3.50. Console Keyword	549
8.3.51. Cook Keyword	550
8.3.52. Copy Keyword	551
8.3.53. Cue Keyword	553
8.3.54. CueAbsolute Keyword	556
8.3.55. CueDelay Keyword	557
8.3.56. CueFade Keyword	559
8.3.57. CueInDelay Keyword	561
8.3.58. CueInFade Keyword	562
8.3.59. CueOutDelay Keyword	563
8.3.60. CueOutFade Keyword	564
8.3.61. CueRelative Keyword	565
8.3.62. CueUpdate Keyword	566
8.3.63. CurrentEnvironment Keyword	567
8.3.64. CurrentUser Keyword	568
8.3.65. CurrentUserProfile Keyword	569
8.3.66. Cut Keyword	570
8.3.67. DataPool Keyword	571
8.3.68. Deceleration Keyword	572
8.3.69. Decimal8 Keyword	573
8.3.70. Decimal16 Keyword	574
8.3.71. Decimal24 Keyword	575
8.3.72. Default Keyword	576
8.3.73. Delay Keyword	578
8.3.74. Delete Keyword	579
8.3.75. DeleteOtherVersions Keyword	580
8.3.76. DeleteGlobalVariable Keyword	581
8.3.77. DeleteUserVariable	582
8.3.78. Dismiss Keyword	583
8.3.79. Display Keyword	584
8.3.80. DMXAddress Keyword	585
8.3.81. DMXLayer Keyword	586



8.3.82. DMXReadout Keyword	587
8.3.83. DMXUniverse Keyword	588
8.3.84. DoubleSpeed Keyword	590
8.3.85. Down Keyword	591
8.3.86. DumpLog Keyword	592
8.3.87. Drive Keyword	593
8.3.88. DropOwnership Keyword	594
8.3.89. Echo Keyword	595
8.3.90. Edit Keyword	596
8.3.91. EditSetting Keyword	597
8.3.92. Eject Keyword	598
8.3.93. Effect Keyword	599
8.3.94. EncoderPage Keyword	600
8.3.95. EndIf Keyword	601
8.3.96. Environment Keyword	602
8.3.97. Exchange Keyword	603
8.3.98. Executor Keyword	604
8.3.99. Export Keyword	607
8.3.100. Extract Keyword	609
8.3.101. Extension Keyword	610
8.3.102. Fade Keyword	611
8.3.103. FaderCrossFade Keyword	613
8.3.104. FaderCrossFadeA Keyword	614
8.3.105. FaderCrossFadeB Keyword	615
8.3.106. FaderMaster Keyword	616
8.3.107. FaderRate Keyword	617
8.3.108. FaderSpeed Keyword	618
8.3.109. FaderTemp Keyword	619
8.3.110. FaderTime Keyword	620
8.3.111. FeatureGroup Keyword	621
8.3.112. Filter Keyword	622
8.3.113. Fix Keyword	623
8.3.114. FixtureClass Keyword	624
8.3.115. FixtureLayer Keyword	625



8.3.116. Fixture Keyword	626
8.3.117. FixtureType Keyword	627
8.3.118. Flip Keyword	629
8.3.119. Flash Keyword	630
8.3.120. Fog Keyword	631
8.3.121. Freeze Keyword	632
8.3.122. Full Keyword	633
8.3.123. Gel Keyword	634
8.3.124. GetGlobalVariable Keyword	635
8.3.125. GetUserVariable Keyword	636
8.3.126. Go+ Keyword	637
8.3.127. Go- Keyword	638
8.3.128. Goto Keyword	639
8.3.129. Grid Keyword	640
8.3.130. GridPosition Keyword	641
8.3.131. Group Keyword	642
8.3.132. HalfSpeed Keyword	643
8.3.133. Help Keyword	644
8.3.134. HelpLua Keyword	645
8.3.135. Hex8 Keyword	646
8.3.136. Hex16 Keyword	647
8.3.137. Hex24 Keyword	648
8.3.138. Highlight Keyword	649
8.3.139. Houselights Keyword	651
8.3.140. Hz Keyword	652
8.3.141. If Keyword	653
8.3.142. IfActive Keyword	655
8.3.143. IfOutput Keyword	656
8.3.144. IfProgrammer Keyword	657
8.3.145. Image Keyword	658
8.3.146. Import Keyword	659
8.3.147. Index Keyword	661
8.3.148. Insert Keyword	662
8.3.149. Integrate Keyword	663



8.3.150. Interface Keyword	665
8.3.151. Invert Keyword	666
8.3.152. Invite Keyword	667
8.3.153. IP Keyword	669
8.3.154. JoinSession Keyword	670
8.3.155. Key Keyword	672
8.3.156. Keyboard Keyword	673
8.3.157. KeyboardShortcut Keyword	674
8.3.158. Kill Keyword	675
8.3.159. KnockIn Keyword	676
8.3.160. KnockOut Keyword	677
8.3.161. Label Keyword	678
8.3.162. Layout Keyword	679
8.3.163. LearnSpeed Keyword	680
8.3.164. LeaveSession Keyword	681
8.3.165. Library Keyword	682
8.3.166. List Keyword	683
8.3.167. ListOwnership Keyword	687
8.3.168. ListReference Keyword	688
8.3.169. Load Keyword	690
8.3.170. Loaded Keyword	691
8.3.171. LoadShow Keyword	692
8.3.172. Lock Keyword	694
8.3.173. LogIn Keyword	695
8.3.174. LogOut Keyword	696
8.3.175. Lowlight Keyword	697
8.3.176. Lua Keyword	698
8.3.177. LuaFile Keyword	699
8.3.178. Macro Keyword	700
8.3.179. MArker Keyword	701
8.3.180. Master Keyword	702
8.3.181. MAricks Keyword	703
8.3.182. Measure Keyword	705
8.3.183. Media Keyword	707



8.3.184. MemoryInfo Keyword	708
8.3.185. Menu Keyword	709
8.3.186. Mesh Keyword	710
8.3.187. Move Keyword	711
8.3.188. Multipatch Keyword	712
8.3.189. MyRunningMacro Keyword	713
8.3.190. MyRunningPreset Keyword	714
8.3.191. MyRunningSequence Keyword	715
8.3.192. MyRunningTimer Keyword	716
8.3.193. Natural Keyword	717
8.3.194. NDI Keyword	718
8.3.195. NewShow Keyword	719
8.3.196. Next Keyword	720
8.3.197. NextY Keyword	722
8.3.198. NextZ Keyword	723
8.3.199. NetworkNode Keyword	724
8.3.200. NonDim Keyword	725
8.3.201. Normal Keyword	726
8.3.202. Note Keyword	727
8.3.203. Off Keyword	728
8.3.204. Offset Keyword	729
8.3.205. On Keyword	730
8.3.206. onPC Keyword	731
8.3.207. OSC Keyword	732
8.3.208. Oops Keyword	734
8.3.209. OutputLayer Keyword	735
8.3.210. Page Keyword	736
8.3.211. Part Keyword	738
8.3.212. Park Keyword	739
8.3.213. Paste Keyword	741
8.3.214. Patch Keyword	742
8.3.215. Pause Keyword	744
8.3.216. Percent Keyword	745
8.3.217. PercentFine Keyword	746



8.3.218. Phase Keyword	747
8.3.219. Physical Keyword	748
8.3.220. Plugin Keyword	749
8.3.221. Preset Keyword	751
8.3.222. PresetUpdate Keyword	754
8.3.223. Press Keyword	755
8.3.224. Preview Keyword	756
8.3.225. Previous Keyword	757
8.3.226. PreviousY Keyword	758
8.3.227. PreviousZ Keyword	759
8.3.228. Programmer Keyword	760
8.3.229. Property Keyword	761
8.3.230. ProcessingUnit Keyword	762
8.3.231. Pyro Keyword	763
8.3.232. Quickey Keyword	764
8.3.233. Rate1 Keyword	765
8.3.234. RDM Keyword	766
8.3.235. Readout Keyword	767
8.3.236. Reboot Keyword	768
8.3.237. Recast Keyword	770
8.3.238. Record Keyword	771
8.3.239. Relative Keyword	772
8.3.240. Release Keyword	773
8.3.241. ReloadAllPlugins Keyword	774
8.3.242. ReloadUI Keyword	775
8.3.243. Remote Keyword	776
8.3.244. RemoteHID Keyword	779
8.3.245. RemoteCommand Keyword	780
8.3.246. Remove Keyword	781
8.3.247. RenderQuality Keyword	782
8.3.248. Reset Keyword	783
8.3.249. Restart Keyword	784
8.3.250. Root Keyword	785
8.3.251. RealtimeChannel Keyword	786

8.3.252. RunningMacro Keyword	787
8.3.253. RunningPreset Keyword	788
8.3.254. RunningSequence Keyword	789
8.3.255. RunningTimecode Keyword	790
8.3.256. RunningTimer Keyword	791
8.3.257. SaveShow Keyword	792
8.3.258. ScreenConfiguration Keyword	793
8.3.259. ScreenContent Keyword	795
8.3.260. Scribble Keyword	796
8.3.261. Seconds Keyword	797
8.3.262. Select Keyword	798
8.3.263. Selection Keyword	800
8.3.264. SelectFixtures Keyword	801
8.3.265. Sequence Keyword	802
8.3.266. SendMIDI Keyword	803
8.3.267. SendOSC Keyword	805
8.3.268. Set Keyword	808
8.3.269. SetGlobalVariable Keyword	809
8.3.270. SetUserVariable Keyword	810
8.3.271. Shuffle Keyword	811
8.3.272. ShutDown Keyword	812
8.3.273. SnapDelay Keyword	814
8.3.274. SoftwareImport Keyword	815
8.3.275. SoftwareUpdate Keyword	816
8.3.276. SoundChannel Keyword	817
8.3.277. Solo Keyword	818
8.3.278. SpecialExecutor Keyword	819
8.3.279. Speed Keyword	820
8.3.280. Speed1 Keyword	821
8.3.281. SpeedMaster Keyword	822
8.3.282. Stage Keyword	823
8.3.283. Station Keyword	824
8.3.284. StationSettings Keyword	826
8.3.285. Step Keyword	827



8.3.286. Stomp Keyword	828
8.3.287. Store Keyword	829
8.3.288. SwitchTograndMA2Software Keyword	831
8.3.289. SwitchTograndMA3Software Keyword	832
8.3.290. Swap Keyword	833
8.3.291. Temp Keyword	834
8.3.292. Thru Keyword	835
8.3.293. Time Keyword	837
8.3.294. Timecode Keyword	838
8.3.295. TimecodeSlot Keyword	841
8.3.296. Timer Keyword	842
8.3.297. Toggle Keyword	844
8.3.298. TopUp Keyword	845
8.3.299. Top Keyword	846
8.3.300. Transition Keyword	847
8.3.301. Type Keyword	848
8.3.302. UIChannel Keyword	849
8.3.303. UIGridSelection Keyword	850
8.3.304. Unblock Keyword	851
8.3.305. Universal Keyword	852
8.3.306. Unlock Keyword	853
8.3.307. Unpark Keyword	854
8.3.308. Unpress Keyword	855
8.3.309. Up Keyword	856
8.3.310. Update Keyword	857
8.3.311. UpdateContent Keyword	858
8.3.312. User1 Keyword	859
8.3.313. User2 Keyword	860
8.3.314. User Keyword	861
8.3.315. UserProfile Keyword	863
8.3.316. Video Keyword	866
8.3.317. Version Keyword	867
8.3.318. View Keyword	869
8.3.319. ViewButton Keyword	871



8.3.320. Width Keyword	873
8.3.321. World Keyword	874
8.3.322. Zero Keyword	875
8.4. Option Keywords	876
8.4.1. /Active Option Keyword	877
8.4.2. /ActiveForSelected Option Keyword	878
8.4.3. /AddNewContent Option Keyword	879
8.4.4. /All Option Keyword	880
8.4.5. /AllForSelected Option Keyword	882
8.4.6. /Ask Option Keyword	883
8.4.7. /Auto Option Keyword	884
8.4.8. /CopyCueDestination Option Keyword	885
8.4.9. /CopyCueSource Option Keyword	886
8.4.10. /CreateReferenceObject Option Keyword	887
8.4.11. /CreateSecondCue Option Keyword	888
8.4.12. /CueOnly Option Keyword	889
8.4.13. /Date Option Keyword	890
8.4.14. /Default Option Keyword	891
8.4.15. /DiscardChanges Option Keyword	892
8.4.16. /DMX Option Keyword	893
8.4.17. /DMXProtocols Option Keyword	894
8.4.18. /Embed Option Keyword	895
8.4.19. /Enumerate Option Keyword	896
8.4.20. /File Option Keyword	897
8.4.21. /ForceGlobal Option Keyword	898
8.4.22. /GDTF Option Keyword	899
8.4.23. /Gaps Option Keyword	900
8.4.24. /Global Option Keyword	901
8.4.25. /GridMergeMode Option Keyword	902
8.4.26. /Indirect Option Keyword	904
8.4.27. /InputFilter Option Keyword	905
8.4.28. /KeepActivation Option Keyword	906
8.4.29. /LocalSettings Option Keyword	907
8.4.30. /Look Option Keyword	908



8.4.31. /MAtricks Option Keyword	910
8.4.32. /Merge Option Keyword	911
8.4.33. /MergeHighPriority Option Keyword	912
8.4.34. /MergeLowPriority Option Keyword	913
8.4.35. /MoveValues Option Keyword	914
8.4.36. /NoAutoClose Option Keyword	915
8.4.37. /NoConfirmation Option Keyword	916
8.4.38. /NoDependencies Option Keyword	917
8.4.39. /NoOops Option Keyword	918
8.4.40. /NoRefresh Option Keyword	919
8.4.41. /NoSave Option Keyword	920
8.4.42. /NoShowData Option Keyword	921
8.4.43. /NoSubfolders Option Keyword	922
8.4.44. /OriginalContentOnly Option Keyword	923
8.4.45. /Output Option Keyword	924
8.4.46. /OutputStations Option Keyword	925
8.4.47. /Overwrite Option Keyword	926
8.4.48. /PatchOffset Option Keyword	927
8.4.49. /Path Option Keyword	928
8.4.50. /PhaserData Option Keyword	929
8.4.51. /Programmer Option Keyword	930
8.4.52. /Release Option Keyword	931
8.4.53. /Remove Option Keyword	932
8.4.54. /Save Option Keyword	933
8.4.55. /Screen Option Keyword	934
8.4.56. /Selection Option Keyword	935
8.4.57. /Selective Option Keyword	936
8.4.58. /Single Option Keyword	937
8.4.59. /Tab Option Keyword	940
8.4.60. /Type Option Keyword	941
8.4.61. /Universal Option Keyword	943
8.4.62. /Wrap Option Keyword	944
8.6. Extended Command Line Syntax Options	945
9. Windows, Views, and Menus	949



9.1.	Add Window	950
9.2.	Rearrange Windows	956
9.3.	Store and Recall Views	957
9.4.	Remove Windows from a Screen	964
9.5.	Window Settings	965
9.6.	Title Bar Configuration	975
9.7.	Menus	977
9.8.	Change Menu Locations	981
9.9.	Pool Windows	983
9.9.1.	Create Pool Objects	986
9.9.2.	Label Pool Objects	987
9.9.3.	Move Pool Objects	989
9.9.4.	Insert Pool Objects	990
9.9.5.	Copy Pool Objects	991
9.9.6.	Lock and Unlock Pool Objects	992
9.9.7.	Delete Pool Objects	993
10.	Networking	994
10.1.	Interfaces and IP	996
10.2.	Session	999
10.2.1.	Create a Session	1006
10.2.2.	Join a Session	1007
10.2.3.	Leave a Session	1010
10.2.4.	Invite a Station into a Session	1011
10.2.5.	Dismiss Stations from a Session	1012
10.2.6.	Create a Custom Key	1013
10.2.7.	Session Master Selection	1016
10.3.	Web Remote	1017
10.4.	Network Design	1020
10.4.1.	Protocol Details	1024
10.4.2.	Wireless Networks WLAN - WiFi	1029
10.5.	Regulations and Standards	1030
11.	DMX In and Out	1031
11.1.	DMX Port Configuration	1032

11.2.	Ethernet DMX	1037
11.2.1.	Art-Net Menu	1039
11.2.2.	sACN (streaming ACN) Menu	1044
11.2.3.	Transmit DMX Using Art-Net	1048
12.	Single User and Multi User Systems	1050
12.1.	Create User	1051
12.2.	User Settings	1053
12.3.	Object Ownership	1062
13.	Patch and Fixture Setup	1065
13.1.	What are Fixtures	1068
13.2.	Add Fixtures to the Show	1070
13.3.	Add Multipatch Fixtures	1081
13.4.	My Virtual Rig	1083
13.5.	Live Patch	1084
13.6.	DMX Sheet	1085
13.7.	DMX Universes	1092
13.8.	Remove Fixtures from the Show	1095
13.9.	Position Fixtures in the 3D Space	1096
13.10.	3D Viewer	1104
13.11.	Render Quality	1114
13.12.	Camera Pool	1119
13.13.	Stages in grandMA3	1122
13.14.	Classes and Layers	1125
13.15.	Attribute Definitions	1126
13.15.1.	Activation Group	1129
13.15.2.	Feature Group	1131
13.15.3.	Deactivation Group	1133
13.16.	Parameter List	1135
13.17.	DMX Curves	1138
14.	Operate Fixtures	1142
14.1.	Select Fixtures	1144
14.2.	What Is the Programmer	1150
14.3.	Fixture Sheet	1155

14.3.1. Graphics	1159
14.4. Encoder Resolution	1161
14.5. Using the Color Picker	1165
14.6. Gel Pool	1172
14.7. Selection Bar	1174
14.8. Align	1175
14.9. Selection Grid	1182
14.10. SMArt View	1186
15. Scribbles	1187
15.1. Create Scribbles	1188
15.2. Edit Scribbles	1191
15.3. Assign Scribbles	1193
15.4. Delete Scribbles	1194
16. Images	1195
16.1. Screenshots	1199
17. Video	1200
18. Meshes	1205
19. Gobos	1206
20. Symbols	1207
20.1. Import Symbols	1208
20.2. Delete Symbols	1210
21. Appearances	1212
21.1. Create Appearances	1213
21.2. Use Appearances	1216
21.3. Delete Appearances	1217
22. Notes	1218
23. Groups	1224
23.1. Create Groups	1225
23.2. Edit Groups	1226
23.3. Delete Groups	1228



23.4.	Group Masters	1229
24.	Presets	1231
24.1.	Preset Pools	1240
24.2.	Create New Presets	1244
24.3.	Recipe Presets	1251
24.4.	Use Preset	1256
24.5.	Edit or Update Presets	1259
25.	Worlds and Filters	1263
25.1.	At Filter	1265
25.2.	Create a World	1267
25.3.	Create a Filter	1270
25.4.	Use a World or Filter	1272
25.5.	Delete a World	1273
25.6.	Delete a Filter	1274
26.	MATricks and Shuffle	1275
26.1.	MATricks Blocks	1281
26.2.	MATricks Groups	1283
26.3.	MATricks Wings	1285
26.4.	MATricks Width	1287
26.5.	MATricks Shuffle	1289
26.6.	MATricks Transform	1291
27.	Cues and Sequences	1296
27.1.	What is Tracking	1298
27.1.1.	Tracking Distance	1304
27.1.2.	Break	1315
27.2.	Sequence Sheet	1320
27.3.	Content Sheet	1329
27.4.	Sequence Settings	1332
27.5.	Store Cues	1339
27.6.	Update Cues	1350
27.7.	Copy Cues	1352
27.8.	Cue Recipes	1357
27.9.	Store Settings and Store Preferences	1359

27.10. Play Back Cues	1362
27.11. Move In Black	1367
27.12. Cue Timing	1371
27.13. Renumber Cues	1379
27.14. Delete Cues	1381
28. What are Executors	1382
28.1. Assign Object to an Executor	1388
28.2. Executor Configurations	1397
28.3. Running Playbacks	1402
28.4. Special Executors	1404
30. Masters	1407
30.1. Selected Masters	1409
30.2. Grand Masters	1410
30.2.1. Time Control	1412
30.3. Speed Masters	1413
30.4. Playback Masters	1414
31. Recipes	1415
32. Phasers	1420
32.1. Phaser Editor	1427
32.2. Create a Sinus Dimmer Phaser	1440
32.3. Create a Simple Circle Phaser	1445
32.4. Create a Circle Phaser Around a Position Preset	1449
32.5. Create Color Rainbow Phaser	1453
33. XYZ	1456
33.1. Activating XYZ for Fixture Types	1458
33.2. MArker Fixture	1459
34. Macros	1461
34.1. Create Macros	1463
34.2. Import Macros	1467
34.3. Edit Macros	1469
34.4. Assign Macros to Keys and Buttons	1471



34.5. Variables	1473
34.6. Example Macros	1481
35. Agenda	1482
35.1. View Modes	1483
35.2. Create an Agenda Entry	1484
35.3. Edit an Agenda Entry	1488
35.4. Agenda Toolbar	1489
36. Timers	1491
36.1. Stopwatch	1495
36.2. Countdown	1496
37. Timecode	1498
37.1. What are Timecode Slots	1499
37.2. Timecode Settings	1504
37.3. Track Groups	1509
37.4. Time Ranges and Events	1510
37.5. Toggle View Mode	1515
37.6. Record a Timecode Show	1520
37.7. Record an External Timecode Show	1522
37.8. Edit a Timecode Show	1525
37.9. External Sources	1527
38. Layouts	1529
38.1. Create a Layout	1530
38.2. Assign Multipatch Fixtures	1537
38.3. Edit Layout	1539
38.4. Layout View Settings	1541
38.5. Edit Layout View	1545
38.6. Edit Layout Elements	1548
38.7. Layout Encoder Bar	1553
39. Plugins	1556
39.1. What is Lua	1563
39.2. Handle - light_userdata	1564
39.3. Interface Functions	1565



39.3.1. Progress Bar	1566
39.4. Variable Functions	1567
39.5. Lua Functions - Object-Free API	1568
39.5.2. AddonVars(string)	1569
39.5.3. BuildDetails()	1570
39.5.6. ClassExists(string)	1572
39.5.7. CloseAllOverlays()	1573
39.5.8. CloseUndo(handle)	1574
39.5.9. Cmd(string[, handle])	1575
39.5.10. CmdIndirect(string[, handle[, handle]])	1576
39.5.11. CmdIndirectWait(string[, handle[, handle]])	1577
39.5.12. CmdObj()	1578
39.5.15. ConfigTable()	1579
39.5.16. Confirm(string[, string[, integer[, boolean]]])	1580
39.5.19. CreateUndo(string)	1581
39.5.20. CurrentEnvironment()	1582
39.5.21. CurrentExecPage()	1583
39.5.22. CurrentProfile()	1584
39.5.23. CurrentUser()	1585
39.5.24. DataPool()	1586
39.5.25. DefaultDisplayPositions()	1587
39.5.26. DelVar(handle, string)	1588
39.5.27. DeskLocked()	1589
39.5.28. DeviceConfiguration()	1590
39.5.31. DrawPointer(integer,table[,integer])	1591
39.5.33. Echo(string)	1592
39.5.34. ErrEcho(string)	1593
39.5.35. ErrPrintf(string)	1594
39.5.36. Export(filename, export_data)	1595
39.5.37. ExportCSV(filename, export_data)	1596
39.5.38. ExportJson(filename, export_data)	1597
39.5.45. FromAddr(string[, handle])	1598
39.5.46. GetApiDescriptor()	1599
39.5.47. GetAttributeByUIChannel(integer)	1600



39.5.48. GetAttributeCount()	1601
39.5.49. GetAttributeIndex(string)	1602
39.5.51. GetChannelFunction(integer, integer)	1603
39.5.52. GetChannelFunctionIndex()	1604
39.5.53. GetClassDerivationLevel(string)	1605
39.5.54. GetDisplayByIndex(integer)	1606
39.5.55. GetDisplayCollect()	1607
39.5.56. GetDMXUniverse(integer[, boolean])	1608
39.5.57. GetDMXValue(integer[, integer, boolean])	1609
39.5.59. GetExecutor(integer)	1610
39.5.60. GetFocus()	1611
39.5.61. GetFocusDisplay()	1612
39.5.62. GetObjApiDescriptor()	1613
39.5.63. GetPath(string[, boolean] integer)	1614
39.5.65. GetPathSeparator()	1615
39.5.71. GetRTChannel(integer)	1616
39.5.72. GetRTChannelCount()	1618
39.5.73. GetRTChannels(integer[,boolean] OR handle[,boolean])	1619
39.5.75. GetSample(string)	1621
39.5.76. GetSelectedAttribute()	1622
39.5.77. GetShowFileStatus()	1623
39.5.78. GetSubfixture(integer)	1624
39.5.79. GetSubfixtureCount()	1625
39.5.80. GetTokenName(string)	1626
39.5.81. GetTokenNameByIndex(int)	1627
39.5.85. GetUIChannelCount()	1628
39.5.86. GetUIChannelIndex(integer, integer)	1629
39.5.87. GetUIChannels(integer[,boolean] OR handle[,boolean])	1630
39.5.89. GetVar(handle, string)	1632
39.5.90. GlobalVars()	1633
39.5.91. HandleToInt(handle)	1634
39.5.92. HandleToStr(handle)	1635
39.5.93. HookObjectChange	1636
39.5.94. HostOS()	1637



39.5.95. HostSubType()	1638
39.5.96. HostType()	1639
39.5.97. Import	1640
39.5.98. IncProgress(handle, integer)	1642
39.5.99. IntToHandle(integer)	1643
39.5.101. IsObjectValid(handle)	1644
39.5.103. KeyboardObj()	1645
39.5.105. MasterPool()	1646
39.5.106. MessageBox(table)	1647
39.5.108. MouseObj()	1656
39.5.109. ObjectList	1657
39.5.111. Patch()	1658
39.5.113. PopupInput	1659
39.5.115. Printf(string)	1660
39.5.117. ProgrammerPart	1661
39.5.118. Pult()	1662
39.5.120. ReleaseType()	1663
39.5.122. Root()	1664
39.5.124. SelectedSequence()	1665
39.5.125. Selection()	1666
39.5.129. SelectionCount()	1667
39.5.130. SelectionFirst()	1668
39.5.131. SelectionNext()	1670
39.5.135. SerialNumber()	1672
39.5.136. SetBlockInput(boolean)	1673
39.5.141. SetProgress(handle, integer)	1674
39.5.142. SetProgressRange(handle, integer, integer)	1675
39.5.143. SetProgressText(handle, string)	1676
39.5.144. SetVar(handle, string, value)	1677
39.5.145. ShowData()	1678
39.5.146. ShowSettings()	1679
39.5.147. StartProgress(string)	1680
39.5.148. StopProgress(handle)	1681
39.5.149. StrToHandle(string)	1682

39.5.151. TextInput([string[, string[, integer[, integer]]]])	1683
39.5.152. Time()	1684
39.5.153. Timer	1685
39.5.156. TouchObj()	1686
39.5.159. UserVars()	1687
39.5.160. Version()	1688
39.6. Lua Functions - Object API	1689
39.6.1. Addr(handle[, handle[, boolean]])	1690
39.6.2. AddrNative(handle[, handle[, boolean]])	1691
39.6.3. Children(handle)	1692
39.6.4. Dump (handle)	1693
39.6.5. Export(handle, file_path, file_name)	1695
39.6.6. HasActivePlayback(handle)	1697
39.6.7. Import(handle, file_path, file_name)	1698
39.6.8. ToAddr(handle)	1700
40. What are Quickeys?	1701
40.1. Quickey Editor	1702
40.2. Use Quickey Pool Objects	1704
40.3. Example	1705
41. Data Pools	1707
42. System	1711
42.1. Date and Time	1712
42.2. Clock	1715
42.3. Desk Lights	1722
42.4. System Information	1732
42.5. System Monitor	1735
42.6. Info Window	1736
43. Sound	1739
44. Remote In and Out	1744
44.1. DC Remotes	1751
44.2. MIDI Remotes	1753
44.3. DMX Remotes	1757



44.4.	OSC (Open Sound Control)	1759
44.4.1.	Examples	1763
44.4.2.	TouchOSC	1764
44.4.3.	QLab	1766
44.4.4.	Open Stage Control	1768
44.4.5.	zactrack	1773
44.4.6.	Protocol Viewer	1777
44.5.	PSN (PosiStageNet)	1778
44.6.	MIDI	1779
45.	Remote Device Management (RDM)	1789
46.	Control other MA Devices	1791
46.1.	Control grandMA3 Nodes	1792
46.2.	Control an MA Network Switch	1796
46.3.	RemoteHID	1797
47.	Update the Software	1799
47.1.	Update grandMA3 Consoles	1806
47.2.	Update grandMA3 Nodes	1809
47.3.	Update grandMA3 onPC Windows Hardware	1811
47.4.	Update grandMA3 viz-key	1822
47.5.	Network Update	1825
47.6.	Delete Update Files	1828
47.7.	Troubleshooting Update Process	1830
48.	Fixture Types	1832
48.1.	Import Fixture Types	1833
48.1.1.	Import GDTF	1836
48.1.2.	Conflicts in Fixture Types	1839
48.2.	Build Fixture Types	1842
48.2.1.	Insert Fixture Types	1843
48.2.2.	Insert DMXModes	1845
48.2.3.	Insert Geometries	1855
48.2.4.	Insert Models	1858
48.2.5.	Link Models to Geometry	1860



48.2.6. Link DMX Modes to Geometries	1862
48.2.7. Environmental Fixture Types	1864
48.3. Export Fixture Types	1865
49. File Management	1867
49.1. SFTP Connection to a Console	1868
49.2. Import / Export Menu	1869
49.3. Folder Structure	1874
50. Shut down the System	1879
51. Troubleshooting	1882
51.1. Clean Start	1883
51.2. Update Does Not Work	1884
51.3. Station Does Not Connect	1886
51.4. Panic Macro	1887
52. Glossary	1890
52. Index	1898



1. About the Help

Use this guide to learn how grandMA3 can help you to make a MAzing show.

Start at the beginning, visit each section, use the context sensitive help, connect with the community to work your way through a show, or use our huge pdf document.

Support

Contact us for any questions about your MA product.

MA Lighting and its extensive distributor network offer unparalleled technical support. Call on our expertise for help with any problem, no matter if it is about the operation, software features, software installations, or troubleshooting.

Community

Share your knowledge and get help from other MA users.

A community can be stronger and better than the individual. Be part of the MA community! Go to <https://www.malighting.com/training-support/community>.

Central Support

If you are in need of further assistance or manufacturer support, please fill out an email request on www.malighting.com in Training and Support. Your request will automatically be sent to support@malighting.com (in English or German). This email service is monitored during MA Lighting's regular business hours in Germany from 8.30 am until 5 pm, Monday through Friday.

Emergency Hotline

In any case of a show critical emergency please contact the MA Lighting support hotline.

Phone +49 5251 68 88 65 99

Please note that this 24/7 hotline is strictly for emergency cases.

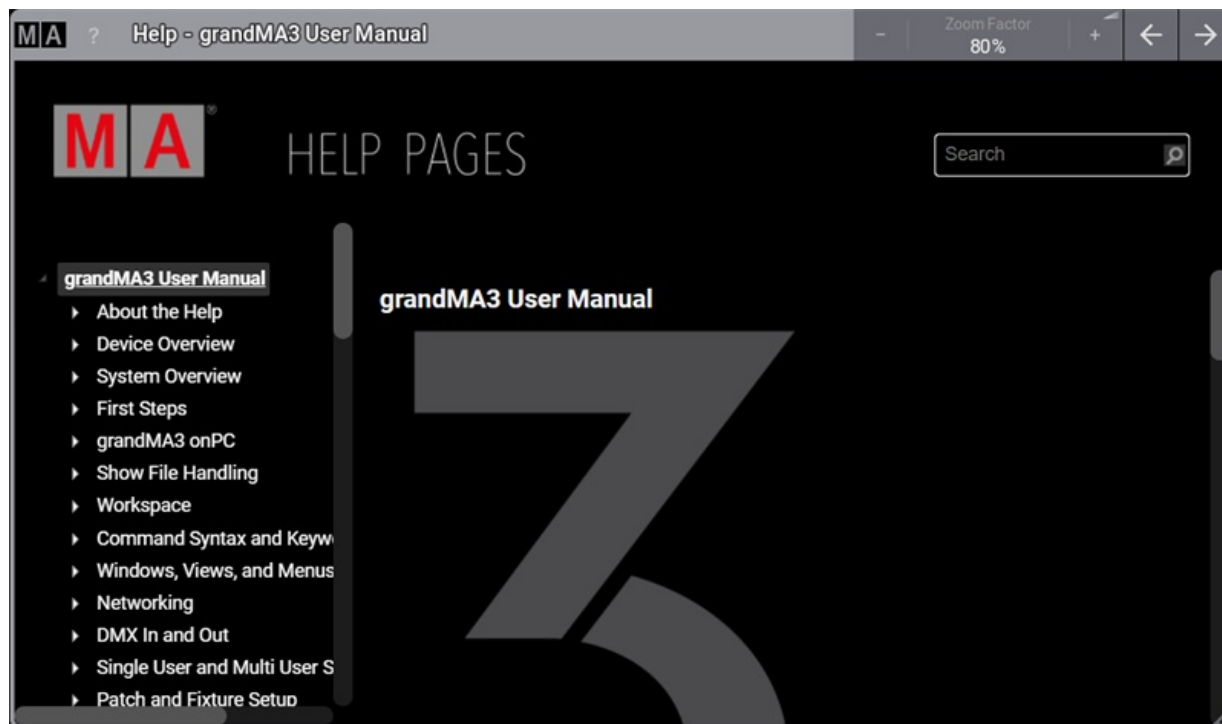
Intellectual Property

MA Lighting Technology has registered multiple patents, trademarks, design patents, and utility models for its products.

We will take legal action against the violation of this intellectual property.

For more information see: <https://www.malighting.com/intellectual-property>

1.1. Open the Help in the Console



Help pop-up

There are different ways to open the help on the console:

- To access and read the help pages, press **Help** + **Please**. The help pop-up opens on the main screen:
- To open the help pop-up, tap **?** in the **control bar** and **?** again.
- To open the help window, tap the Help view button in the **view bar**.
- To open another help window, open the **Add Window**, tap the **More** tab, then tap **Help**. For more information, see the **Add Windows** topic.

Open the Context Sensitive Help

To open the context sensitive help on the console:


- Press **Help** and tap the desired user interface element.
- Tap **?** in the control bar and tap at the user interface element.

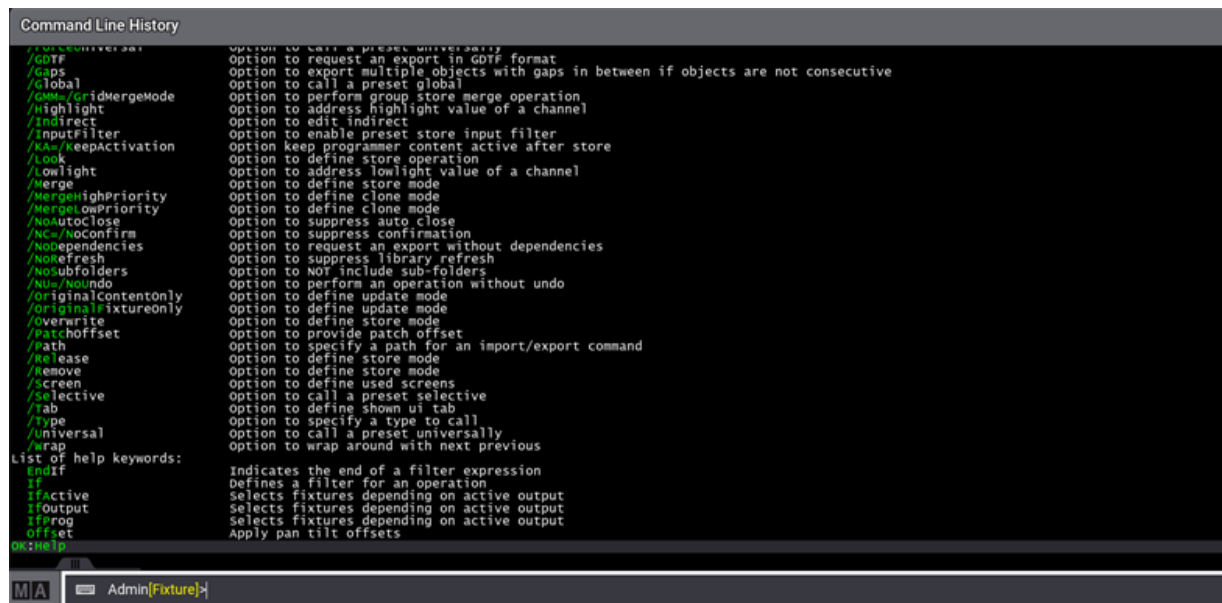
Open the Help via Command Line

To open the help window by using the command line:

- Press **Help**
 - Type **Help**
 - Type the shortcut **H**
-
- Press **Please** or click the command line with the mouse to confirm.

Open a Keyword Help

**Hint:**
To display all keywords in the command line history, type the command **Help + Enter** (keyboard) in the command line.



Keyword list in the command line history

For more information, see the [Help keyword topic](#).

To open a keyword topic immediately via the command line, use one of these options:

- Press **Help** and type the desired keyword.
- Type **Help** and type the desired keyword.

The desired help pages opens.



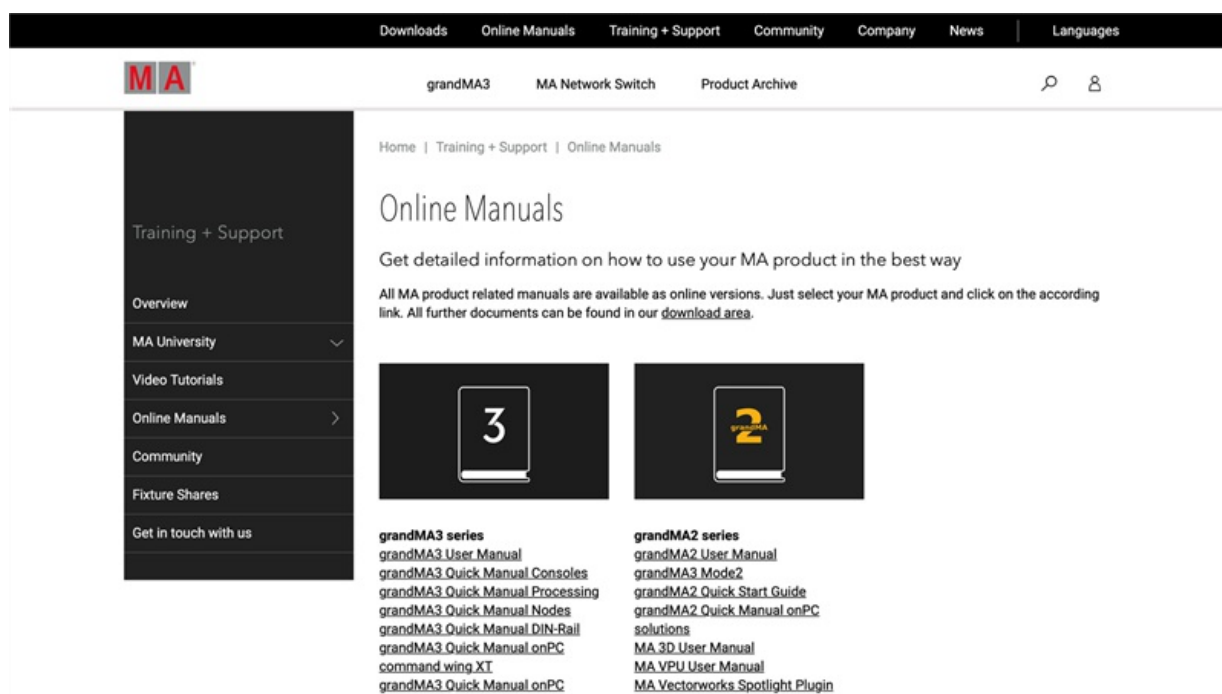
1.2. Open the Help in the Web

Please note that the help topics included in the software represent the status and volume of the help pages at the date of release.

The help system is constantly being updated and extended.

The most up-to-date and complete version of the manual is directly published on the following website:

<https://www.malighting.com/training-support/online-manuals>



Online manuals

The **Online Manuals section** is divided into four product families:

- grandMA3 series
- grandMA2 series
- dot2 series
- MA Network Switch



1.3. Open the Help as PDF

- Click the **Open topic as PDF** button in the upper-right corner of any topic viewed on the web to download the contents of that topic as a PDF file for offline viewing.

To download a PDF of the entire manual:

1. Navigate to the **Downloads** section of the MA Lighting website: <https://www.malighting.com/downloads>
2. Click the desired product family from the list on the left side.
3. Click **Offline User Manuals + QuickManual**.
4. Click the desired manual to download.

The download begins and will be saved at the computer.

1.4. Navigate in the Help

Buttons in the Title Bar

Tap **+** (plus) or **-** (minus) to increase or decrease the size of the content displayed within the help window.

To change the size of the content in a drop-down list, tap **Zoom Factor**.

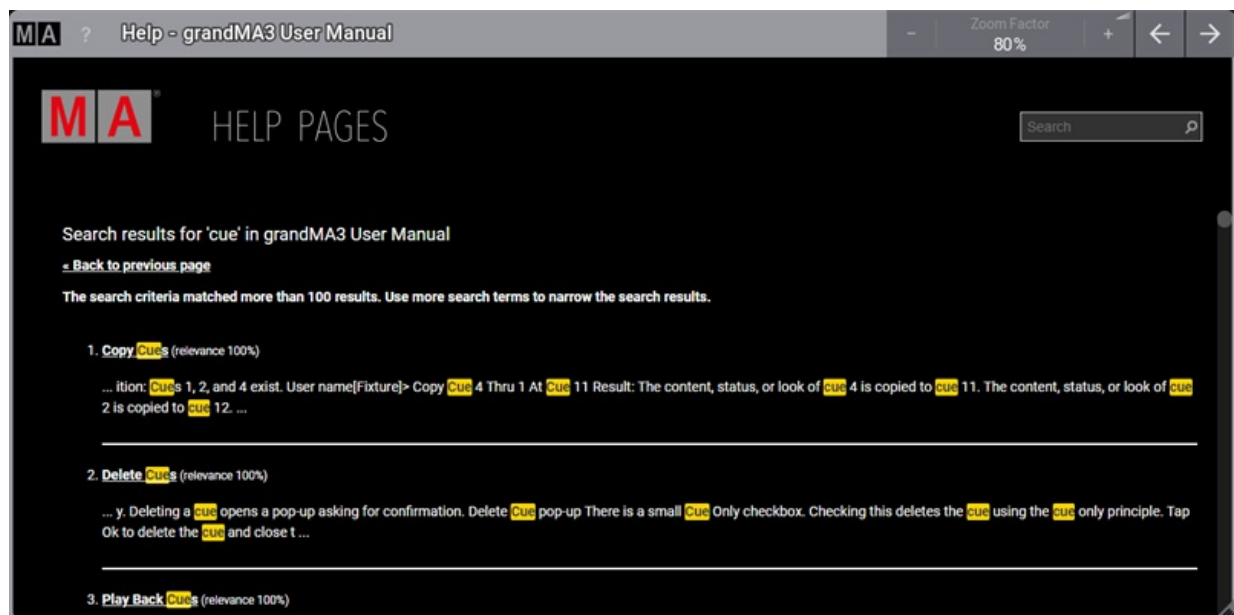
Tap **←** (left arrow) to reverse through the history of previously viewed help topics.

Tap **→** (right arrow) to advance through the history of previously viewed help topics.

Search Field

To find topics containing a specific search term, enter the term into the search field in the upper-right corner and press Please.

The **Help** window displays the search results with the search term highlighted.



Help pop-up with highlighted search results

To search for a specific product or search term, leave away the brand name or the series name (e.g. grandMA3).



Table of Contents

The left side of the **Help** window displays available topics in a tree structure. Tap the triangle next to any topic to reveal or hide any associated sub-topics. Tap the name of any topic to display its contents in the main area of the window.

The root topics in the tree structure are divided into the different products within the family. These main sections include:

- [grandMA3 User Manual](#)
- [grandMA3 Quick Manual consoles](#)
- [grandMA3 Quick Manual processing](#)
- [grandMA3 Quick Manual Nodes](#)
- [grandMA3 Quick Manual Nodes DIN-Rail](#)
- [grandMA3 Quick Manual onPC command wing XT](#)
- [grandMA3 Quick Manual onPC command wing](#)
- [grandMA3 Quick Manual onPC fader wing](#)
- [grandMA3 Quick Manual viz-key](#)
- [grandMA3 Quick Manual I/O Nodes](#)
- [Release Notes](#)



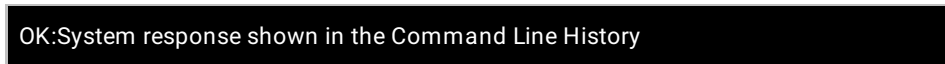
Important:

The complete grandMA3 documentation consists of the grandMA3 User Manual and the respective grandMA3 Quick Manual with technical specifications, safety instructions, and declaration of conformity.







Display of Content

Display	Definition
At	Keys found on the control surface of the hardware are displayed in a white font on a dark gray background.
Dimmer	Buttons found on the screen are displayed in a white font on a light gray background.
<ol style="list-style-type: none"> 1. Open... 2. Select... 	Step-by-step instructions are displayed in a numeric list.
<ul style="list-style-type: none"> • Tap 	One-step instructions are displayed with the help of bullet points.
Menu - Settings - User Configuration - User Profiles	Navigation to specific menus in the console is displayed in bold.
<u>Assign [Object] (At) [Object]</u>	The description of the syntax is underlined and displayed in bold.



The following symbols display possible danger, useful hints, and information.

For information on safety, harmonized standards, and conformity see [grandMA3 Quick Manual Consoles](#).

- 
Warning:
 The warning sign displays possible injury and hazard.
- 
Restriction:
 The restriction sign displays known limitations of functions.
- 
Important:
 The important sign displays essential information for the usage of the console.
- 
Hint:
 The hint sign displays tips useful for the usage of the console.



2. Device Overview

- All MA Lighting devices are built for touring and installation purposes and meet international standards for electronic safety and protection. The Declaration of Conformity and the Electromagnetic Compatibility regulation (EMC) can be found in the Quick Guide included with the product and available for download at <https://www.malighting.com>.
- All MA Lighting devices have a wide input voltage range and can operate from 90V to 230V.

The following chapters describe the different grandMA3 devices.



2.1. grandMA3 consoles

The grandMA3 consoles control all kinds of lighting genres such as conventional lights, moving lights, LEDs, video, and media via DMX signal or within a network environment.

All **grandMA3 components**, despite different hardware solutions, use the same software.

All components are fully integrable into the network environment.

This topic describes the different sections of the grandMA3 consoles.

These sections vary from model to model.

The flagship of the grandMA3 range is the **full-size** or **full-size CRV**, respectively.

The control room version (CRV) of full-size and light are models without the monitor wing, but come with additional external monitor ports.

The grandMA3 **light** console or the **light CRV** are the work-horses of the range.

The **compact** and **compact XT** models offer the full system benefits in a compact and lightweight format.

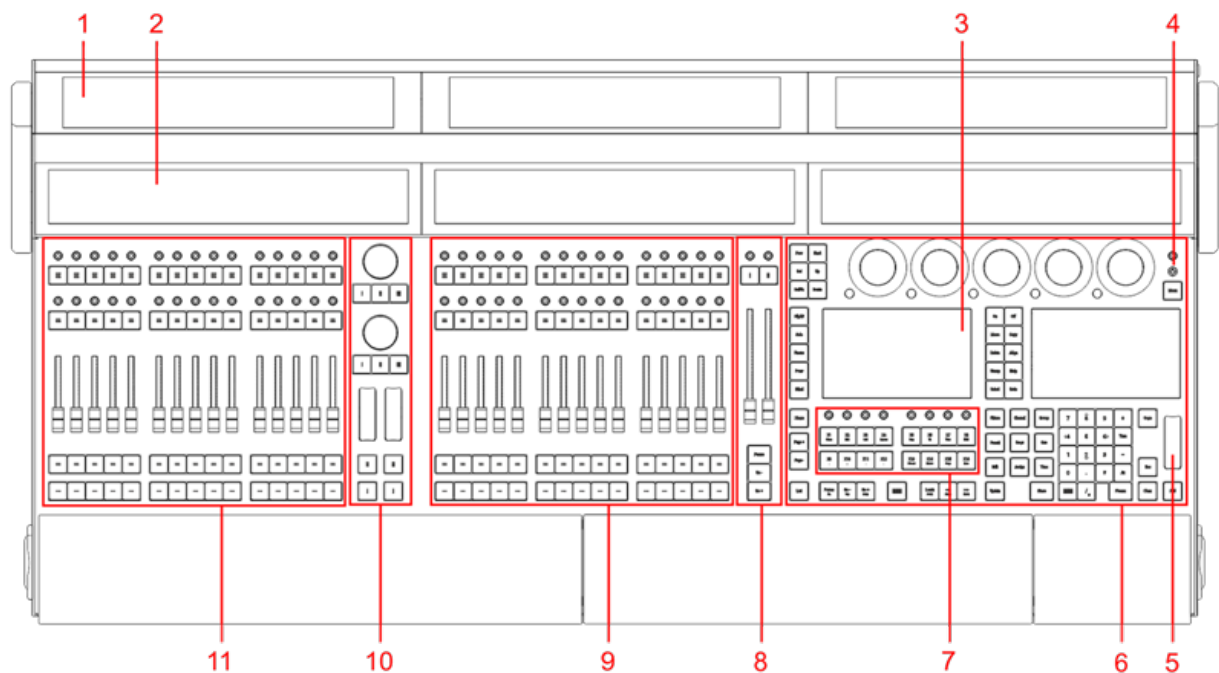
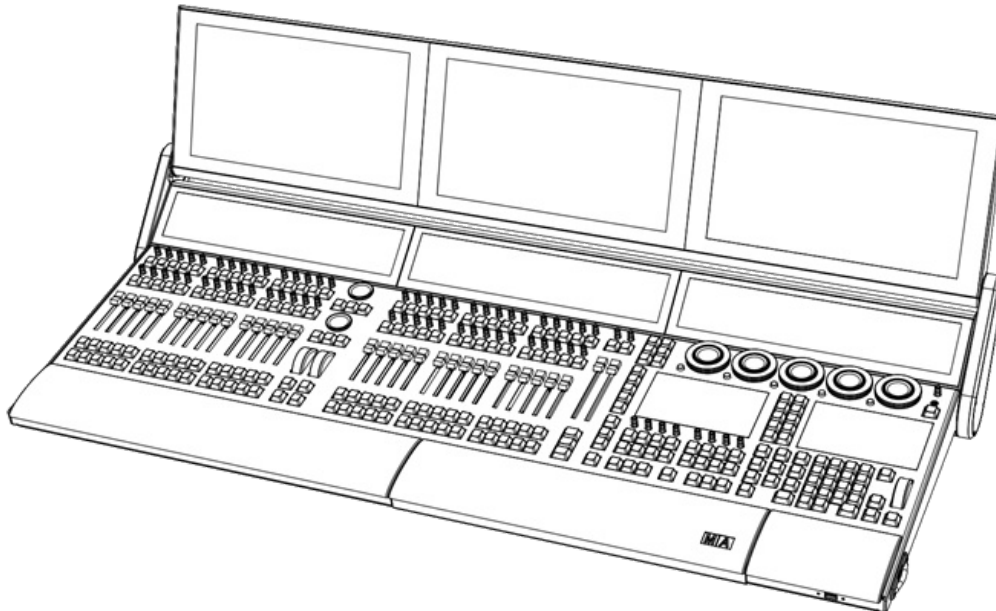
The grandMA3 consoles number their screens in a specific order, for more information see **Screen Allocation**.

You can connect several external touch screens with a console, see **Connect External Screens**.

The display elements of the screens can be modified, see **Configuration of Displays**.

For technical specifications, see **Technical Data** in the **grandMA3 Quick Manual consoles**.

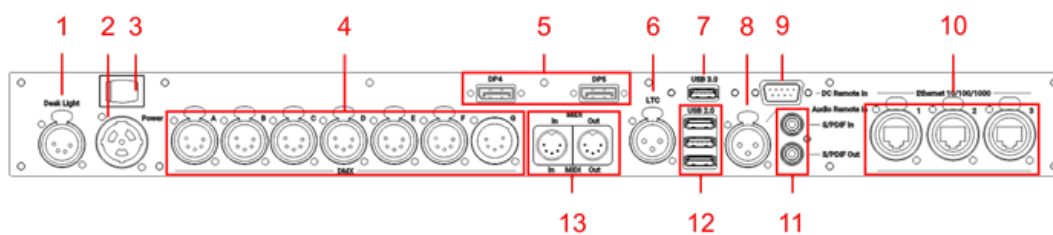
2.1.1. grandMA3 full-size



grandMA3 full-size front panel

1. **Screens 1-3**
2. **Letterbox screens 8-10**
3. **Command screens 6+7**
4. **Power key**
5. **Level wheel**

6. **Command area**
7. **Xkeys section**
8. **Master area**
9. **Right executor area with 3 executor sections**
10. **Custom area**
11. **Left executor area with 3 executor sections**



grandMA3 full-size rear panel

1. **Desk light**
2. **powerCON TRUE1**
3. Power switch
4. **DMX A-G**
5. **DisplayPort 4+5**
6. **LTC**
7. **USB 3.0**
8. **Audio Remote In**
9. **DC Remote Control**
10. **Ethernet 1-3**
11. **S/PDIF In+Out**
12. **USB 2.0**
13. **MIDI In+Out**

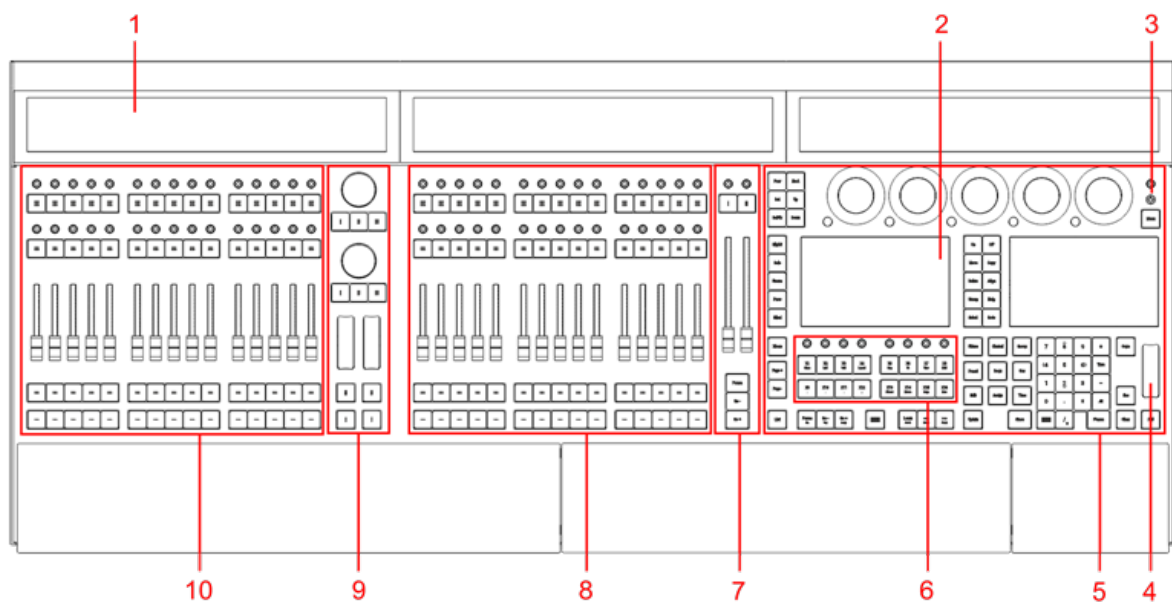
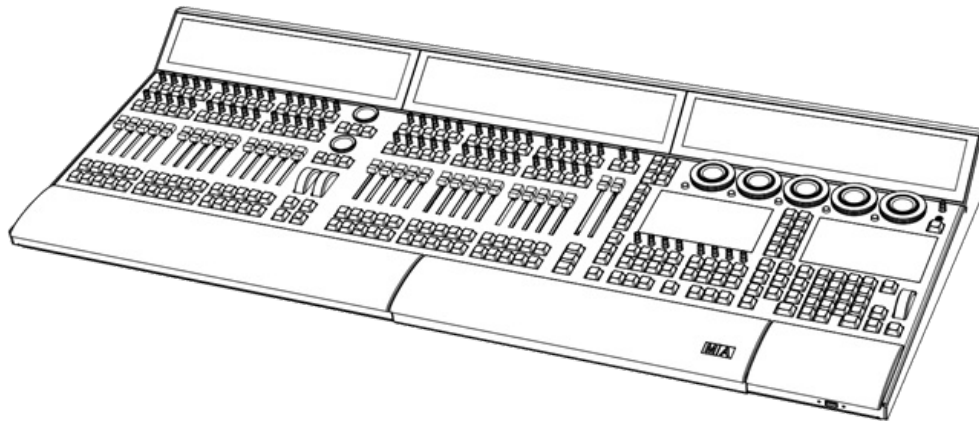


Hint:

For command line and graphical user interface access to an overlay showing the connectors on the back of the console, see the **Show Connectors section of the DMX Port Configuration** topic.

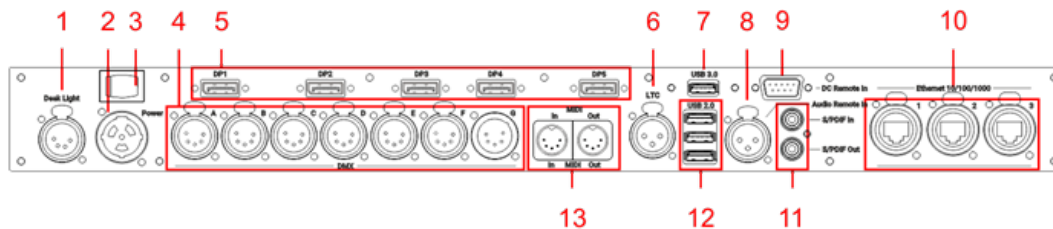
For technical specifications, see **Technical Data** in the **grandMA3 Quick Manual consoles**.

2.1.2. grandMA3 full-size CRV



grandMA3 full-size CRV front panel

1. **Letterbox screens 8-10**
2. **Command screens 6+7**
3. **Power key**
4. **Level wheel**
5. **Command area**
6. **Xkeys section**
7. **Master area**
8. **Right executor area with 3 executor sections**
9. **Custom area**
10. **Left executor area with 3 executor sections**



grandMA3 full-size CRV rear panel

1. **Desk light**
2. **powerCON TRUE1**
3. Power switch
4. **DMX A-G**
5. **DisplayPort 1-5**
6. **LTC**
7. **USB 3.0**
8. **Audio Remote In**
9. **DC Remote Control**
10. **Ethernet 1-3**
11. **S/PDIF In+Out**
12. **USB 2.0**
13. **MIDI In+Out**

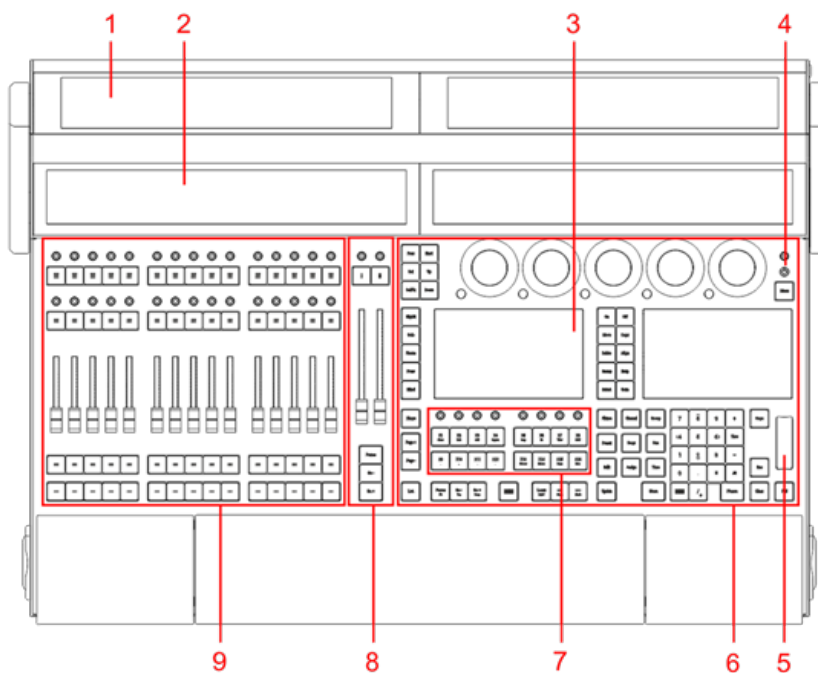
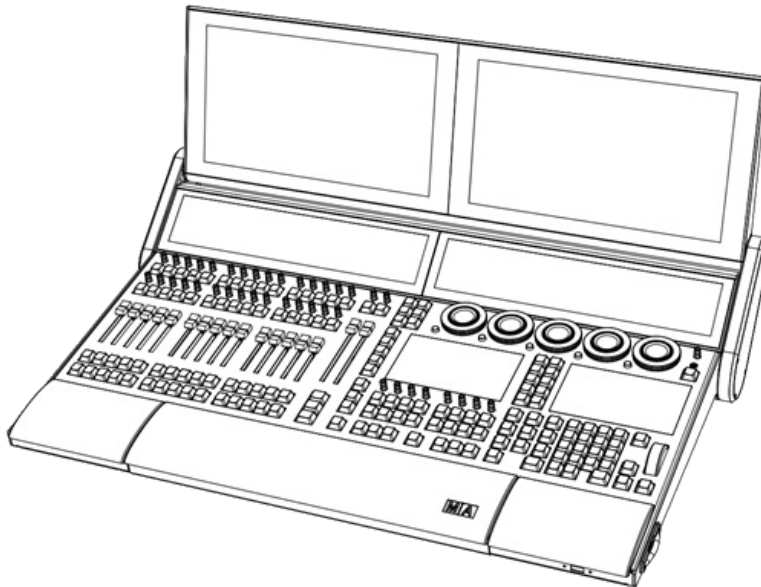


Hint:

For command line and graphical user interface access to an overlay showing the connectors on the back of the console, see the **Show Connectors section of the DMX Port Configuration** topic.

For technical specifications, see **Technical Data** in the **grandMA3 Quick Manual consoles**.

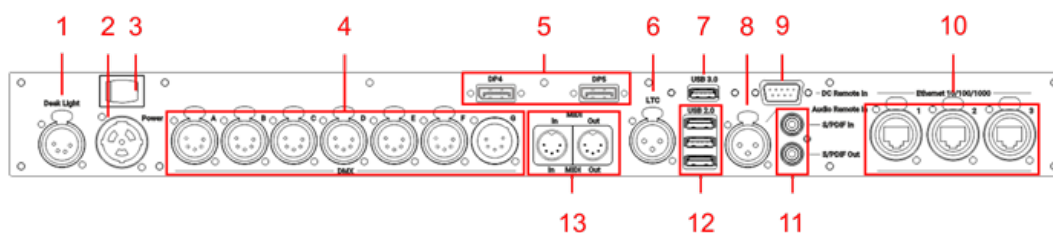
2.1.3. grandMA3 light



grandMA3 light front panel

1. **Screens 1+2**
2. **Letterbox screens 8+9**
3. **Command screens 6+7**
4. **Power key**

5. **Level wheel**
6. **Command area**
7. **Xkeys section**
8. **Master area**
9. **Executor area with 3 executor sections**



grandMA3 full-size rear panel

1. **Desk light**
2. **powerCON TRUE1**
3. Power switch
4. **DMX A-G**
5. **DisplayPort 4+5**
6. **LTC**
7. **USB 3.0**
8. **Audio Remote In**
9. **DC Remote Control**
10. **Ethernet 1-3**
11. **S/PDIF In+Out**
12. **USB 2.0**
13. **MIDI In+Out**

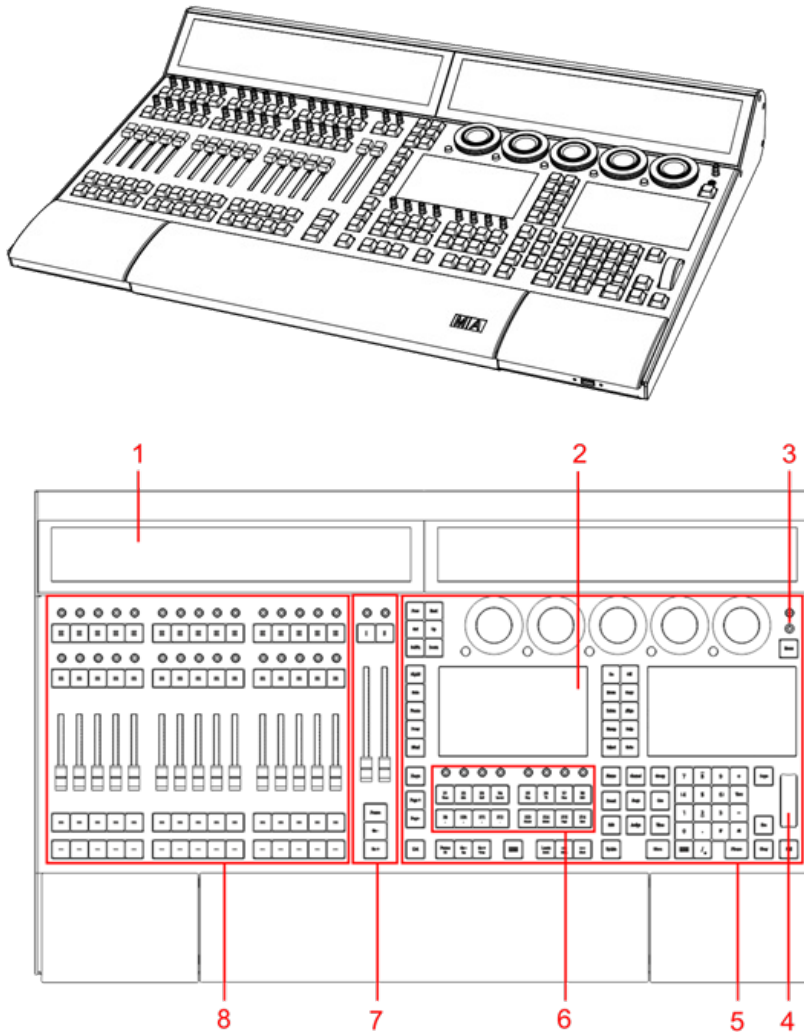


Hint:

For command line and graphical user interface access to an overlay showing the connectors on the back of the console, see the **Show Connectors section of the DMX Port Configuration** topic.

For technical specifications, see **Technical Data** in the **grandMA3 Quick Manual consoles**.

2.1.4. grandMA3 light CRV



grandMA3 light CRV front panel

1. **Letterbox screens 8+9**
2. **Command screens 6+7**
3. **Power key**
4. **Level wheel**
5. **Command area**
6. **Xkeys section**
7. **Master area**
8. **Executor area with 3 executor sections**



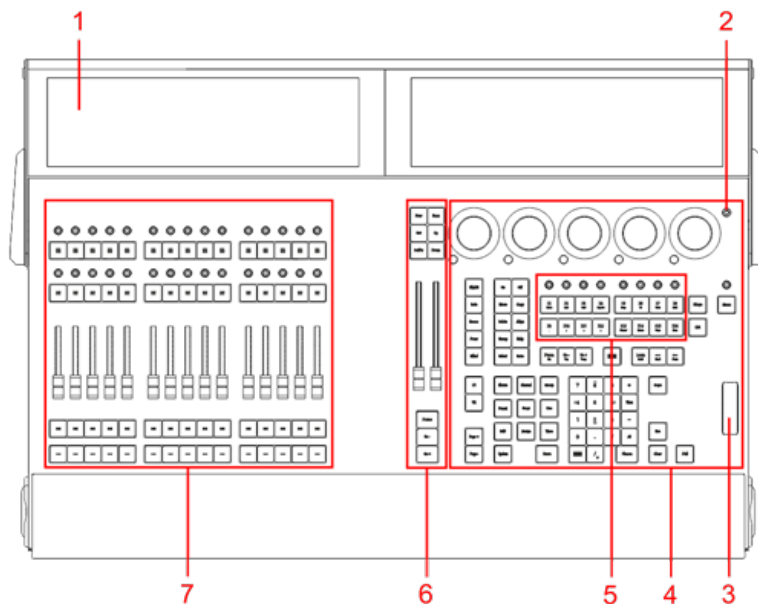
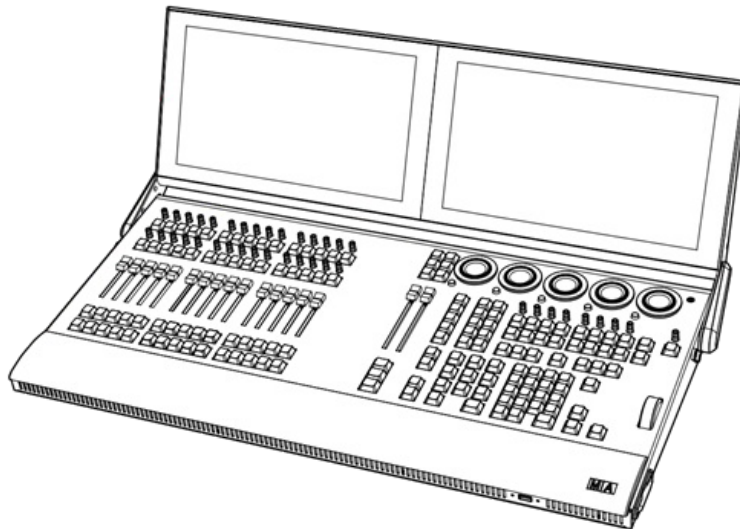
Hint:

For command line and graphical user interface access to an overlay showing the connectors on the back of the console, see the **Show Connectors section of the DMX Port Configuration** topic.



For technical specifications, see **Technical Data** in the **grandMA3 Quick Manual consoles**.

2.1.5. grandMA3 compact XT

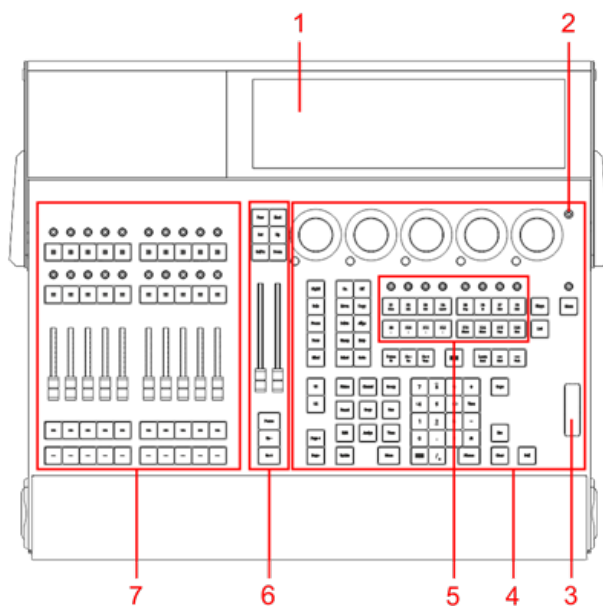
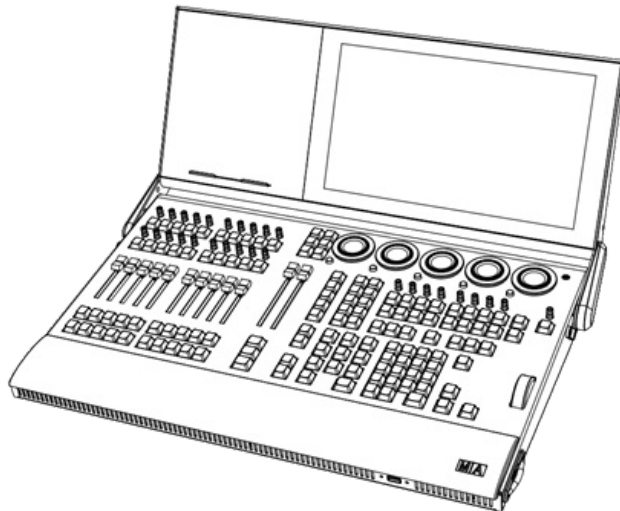


grandMA3 compact XT front panel

1. **Screens 1+2**
2. **Power key**
3. **Level wheel**
4. **Command section**
5. **Xkeys**
6. **Master section**
7. **Executor section**

For technical specifications, see **Technical Data** in the **grandMA3 Quick Manual consoles**.

2.1.6. grandMA3 compact

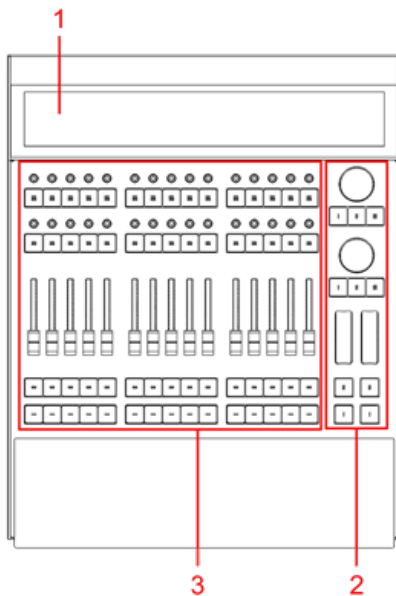
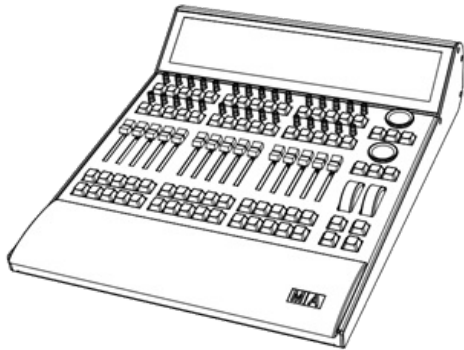


grandMA3 compact front panel

1. **Screen 1**
2. **Power key**
3. **Level wheel**
4. **Command section**
5. **Xkeys**
6. **Master section**
7. **Executor section**

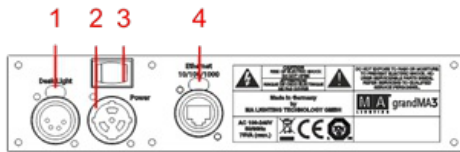
For technical specifications, see **Technical Data** in the **grandMA3 Quick Manual consoles**.

2.2. grandMA3 extension



grandMA3 extension front panel

1. **Letterbox screen 10**
2. **Custom section**
3. **Executor section**



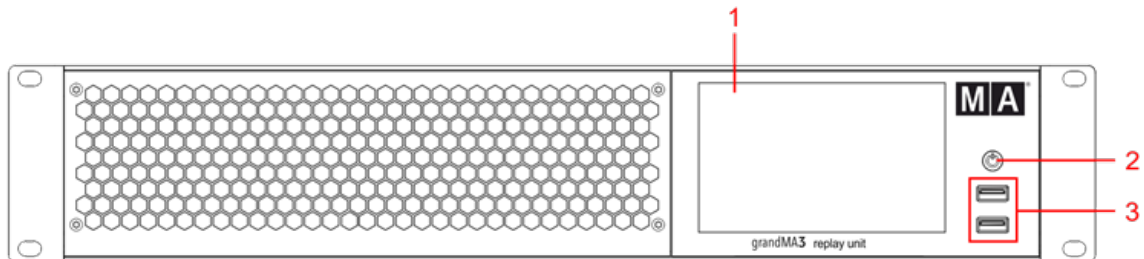
grandMA3 extension rear panel

1. **Desk light**
2. **powerCON TRUE1 connector**
3. Power switch
4. **Ethernet**

For more information, see the topics **First steps**, **Connect grandMA3 extension**.

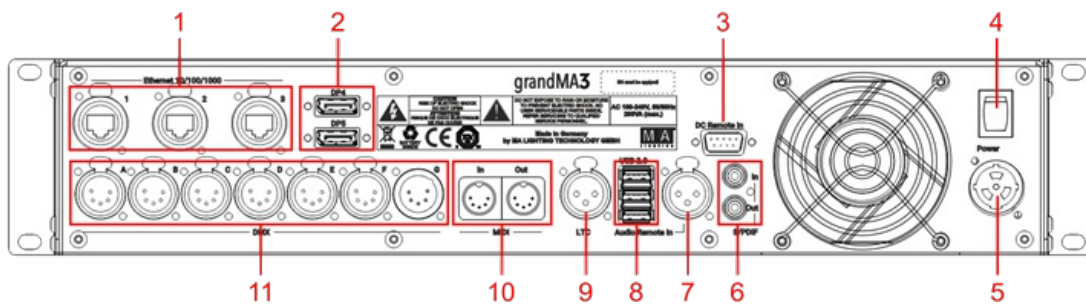
For technical specifications, see **Technical Data** in the **grandMA3 Quick Manual consoles**.

2.3. grandMA3 replay unit



grandMA3 replay unit front panel

1. Internal command multi-touch screen
2. **Power key**
3. **USB 2.0 (type A)**

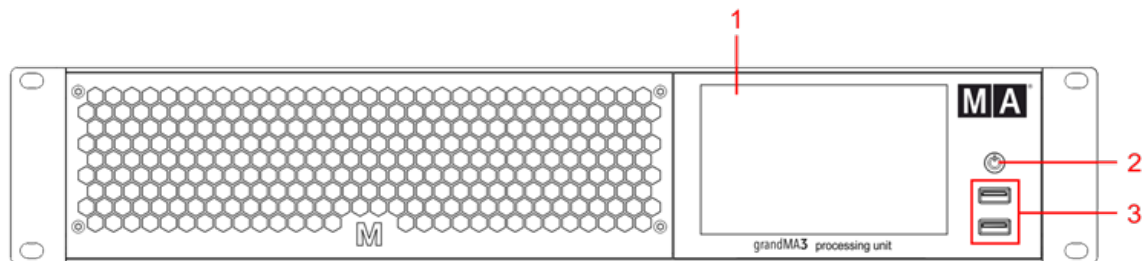


grandMA3 replay unit rear panel

1. **etherCON/RJ45**
2. **DisplayPort 1+2**
3. **DC Remote Control**
4. Power switch
5. **powerCON TRUE1**
6. **S/PDIF In+Out**
7. **Audio Remote In**
8. **USB 2.0 (type A)**
9. **LTC**
10. **MIDI In+Out**
11. **DMX-512-A OUT (5pin XLR female)**

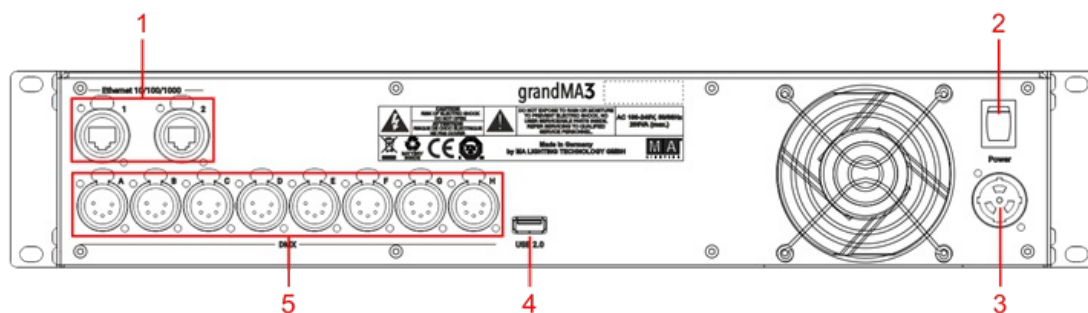
For technical specifications, see **Technical Data** in the **grandMA3 Quick Manual consoles**.

2.4. grandMA3 processing units



grandMA3 processing unit M, L, XL front panel

1. Internal command multi-touch screen
2. **Power key**
3. **USB 2.0 (type A)**



grandMA3 processing unit M, L, XL rear panel

1. **etherCON/RJ45**
2. Power switch
3. **powerCON TRUE1**
4. **USB 2.0 (type A)**
5. **DMX-512-A OUT (5pin XLR female)**

For technical specifications, see **Technical Data** in the **grandMA3 Quick Manual processing**.



2.5. grandMA3 nodes



grandMA3 8Port Node front panel

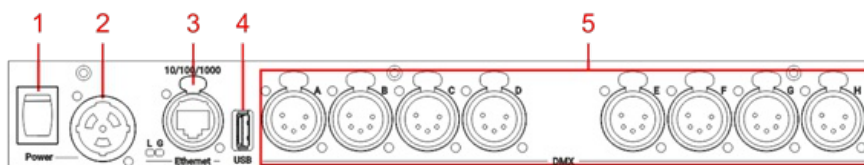


grandMA3 4Port Node front panel

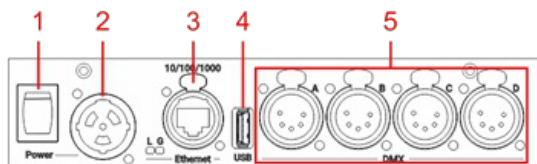


grandMA3 2Port Node front panel

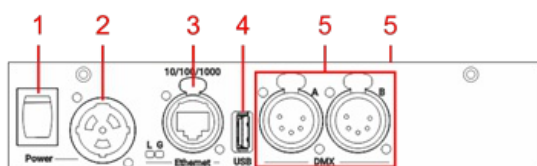
- 1. Display
- 2. Rotary knob



grandMA3 8Port Node rear panel



grandMA3 4Port Node rear panel

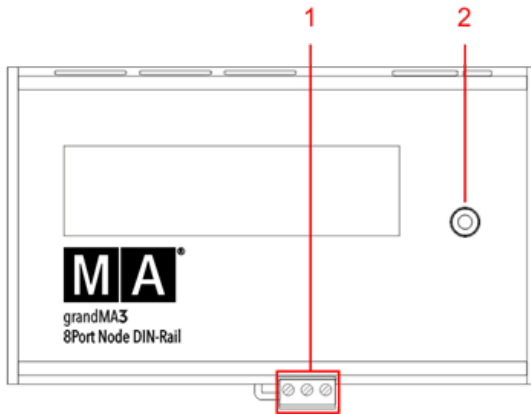




grandMA3 2Port Node rear panel

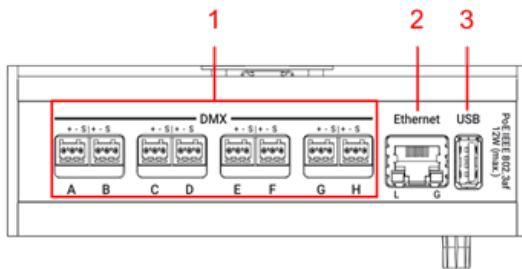
1. Power switch
2. **powerCON TRUE1**
3. **Ethernet** with **L (link) and G (gigabit) LEDs**
4. **USB port**
5. **DMX**

2.6. grandMA3 nodes DIN-Rail



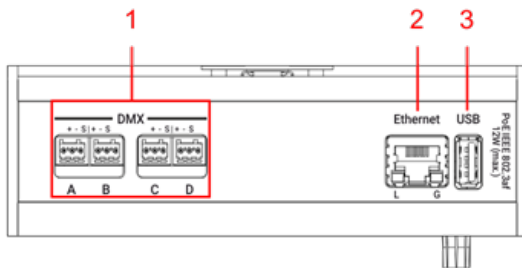
grandMA3 8Port Node DIN-Rail front panel

1. **Terminal block**
2. Rotary knob

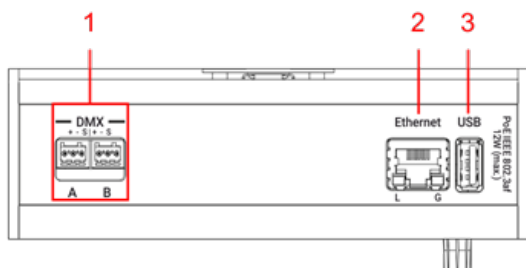


grandMA3 8Port Node DIN-Rail rear panel

1. **DMX**
2. **Ethernet** with **L (link) and G (gigabit) LEDs**
3. **USB 2.0**



grandMA3 4Port Node DIN-Rail rear panel



grandMA3 2Port Node DIN-Rail rear panel

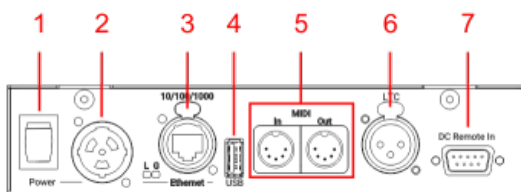
For technical specifications, see [Technical Data](#) in the [grandMA3 Quick Manual Nodes DIN-Rail](#).

2.7. grandMA3 I/O Node



grandMA3 I/O Node front panel

1. Display
2. Rotary knob

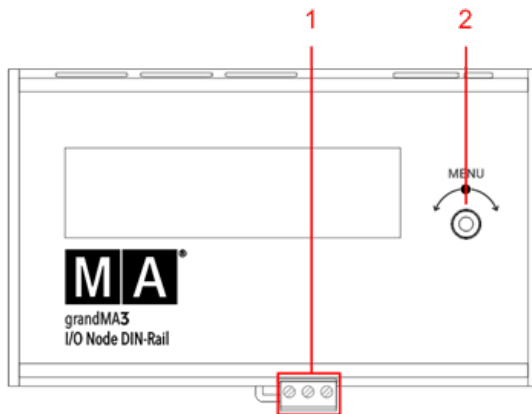


grandMA3 I/O Node rear panel

1. Power switch
2. **powerCON TRUE1**
3. **Ethernet** with **L (link)** and **G (gigabit) LEDs**
4. **USB 2.0**
5. **MIDI In+Out**
6. **LTC**
7. **DC Remote Control**

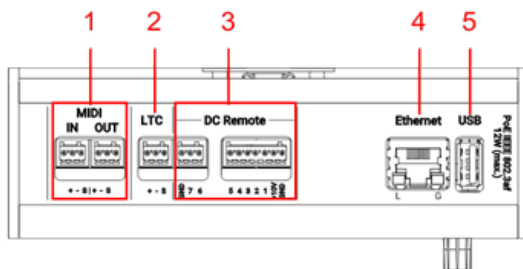
For technical specifications, see **Technical Data** in the **grandMA3 Quick Manual I/O Nodes**.

2.8. grandMA3 I/O Node DIN-Rail



grandMA3 I/O Node DIN Rail front panel

1. **Terminal block**
2. Rotary knob

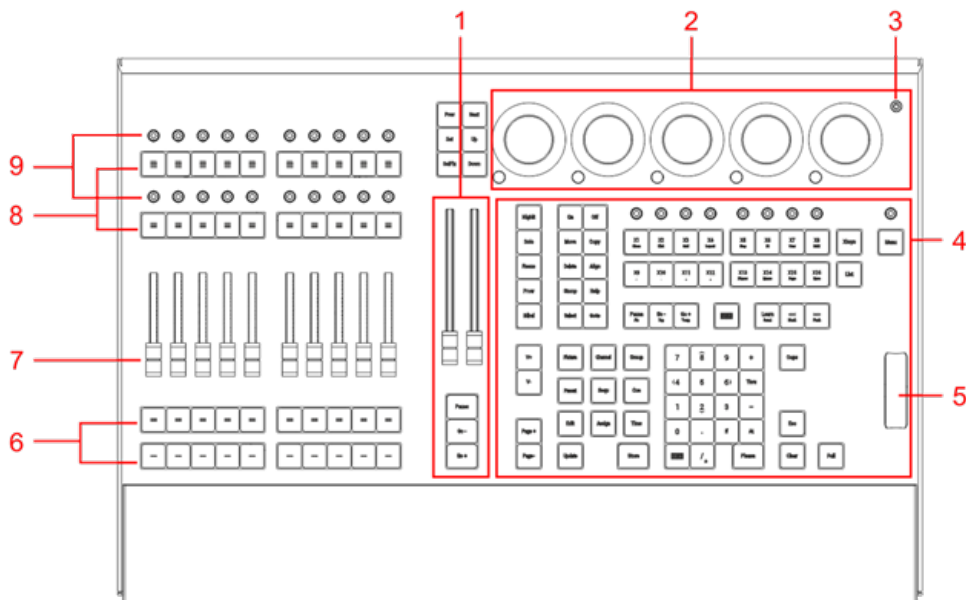
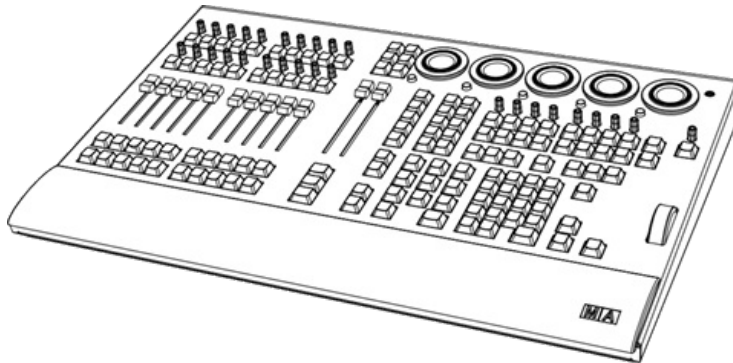


grandMA3 I/O Node DIN Rail rear panel

1. **MIDI In+Out**
2. **LTC**
3. **DC Remote Control**
4. **Ethernet** with **L (link)** and **G (gigabit)** LEDs
5. **USB 2.0**

For technical specifications, see **Technical Data** in the **grandMA3 Quick Manual I/O Nodes**.

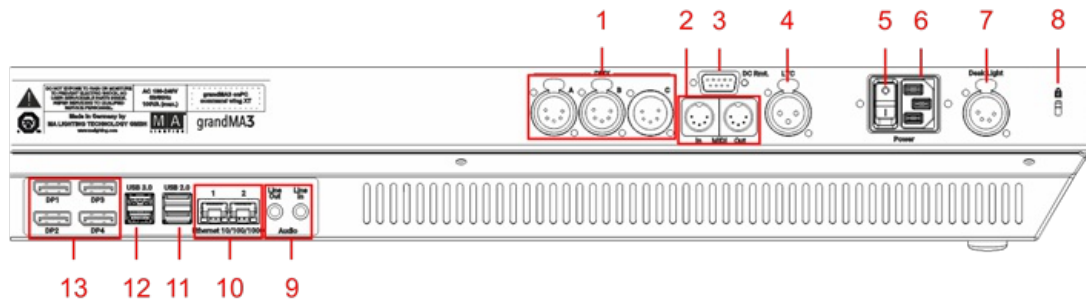
2.9. grandMA3 onPC command wing XT



grandMA3 onPC command wing XT front panel

1. **Master area**
2. **Dual encoder section**
3. **Power key**
4. **Command area**
5. **Level wheel**
6. Executor buttons 101-190 + 201-290
7. Executor faders 201-290
8. Executor buttons 301-390 + 401-490
9. Executor knobs 301-390 + 401-490

For more information about executors, see **Executor elements**.

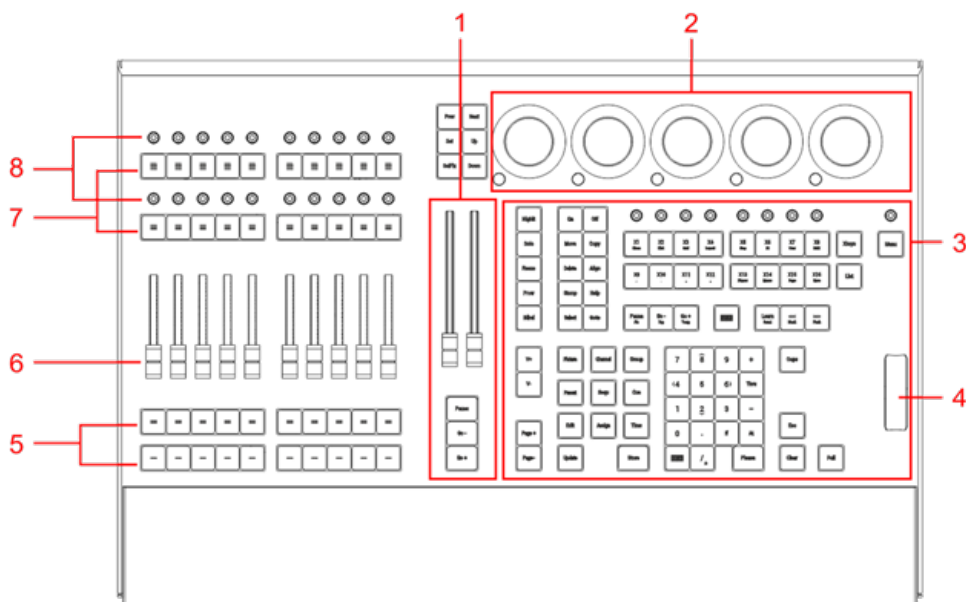
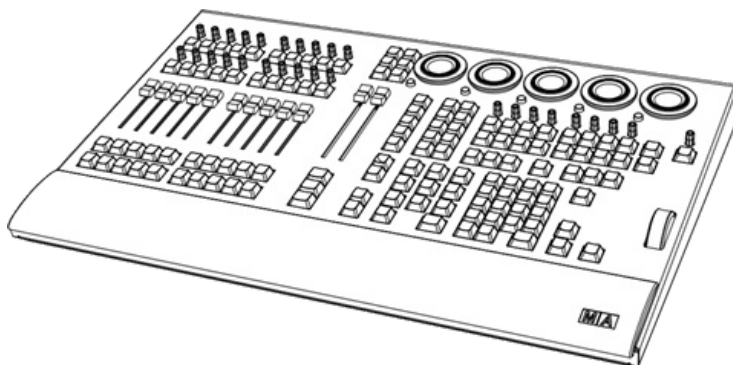


grandMA3 onPC command wing XT rear panel

1. **DMX A, B, C**
2. **MIDI In+Out**
3. **DC Remote Control**
4. **LTC**
5. Power switch
6. **IEC connector**
7. **Desk light**
8. Kensington lock
9. **Line In+Out**
10. **Ethernet 1+2**
11. **USB 2.0**
12. **USB 3.0**
13. **DisplayPort 1-4**

For technical specifications see **Technical Data** in the **grandMA3 Quick Manual onPC command wing XT**.

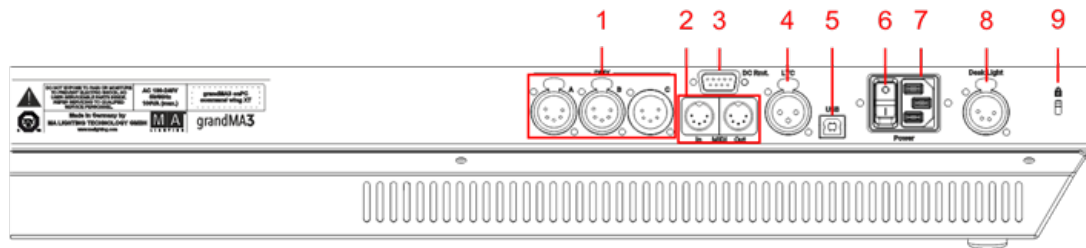
2.10. grandMA3 onPC command wing



grandMA3 onPC command wing front panel

1. **Master area**
2. **Dual encoder section**
3. **Command area**
4. **Level wheel**
5. Executor buttons 101-190 + 201-290
6. Executor faders 201-290
7. Executor buttons 301-390 + 401-490
8. Executor knobs 301-390 + 401-490

For more information about executors, see **Executor elements**.

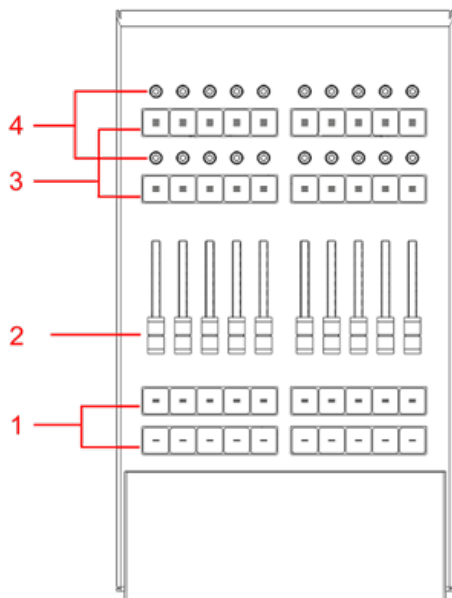


grandMA3 onPC command wing rear panel

1. **DMX A, B, C**
2. **MIDI In+Out**
3. **DC Remote Control**
4. **LTC**
5. USB
6. Power switch
7. **IEC connector**
8. **Desk light**
9. Kensington lock

For technical specifications see **Technical Data** in the **grandMA3 Quick Manual onPC command wing**.

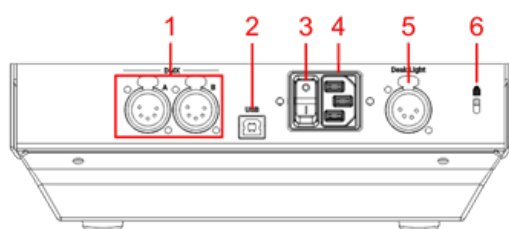
2.11. grandMA3 onPC fader wing



grandMA3 onPC fader wing front panel

1. Executor buttons 101-190 + 201-290
2. Executor faders 201-290
3. Executor buttons 301-390 + 401-490
4. Executor knobs 301-390 + 401-490

For more information about executors, see [Executor elements](#).

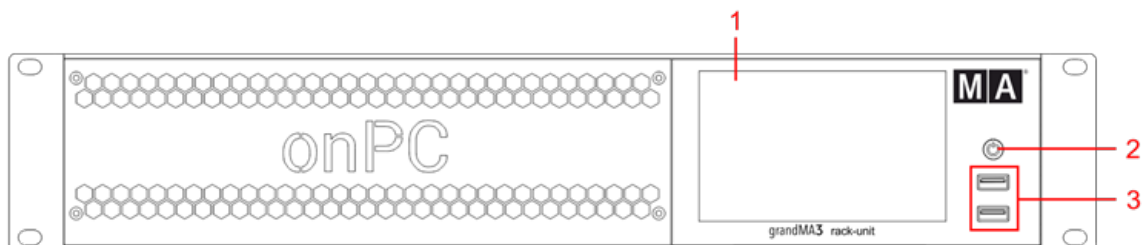


grandMA3 onPC fader wing rear panel

1. **DMX A+B**
2. USB
3. Power switch
4. **IEC connector**
5. **Desk light**
6. Kensington lock

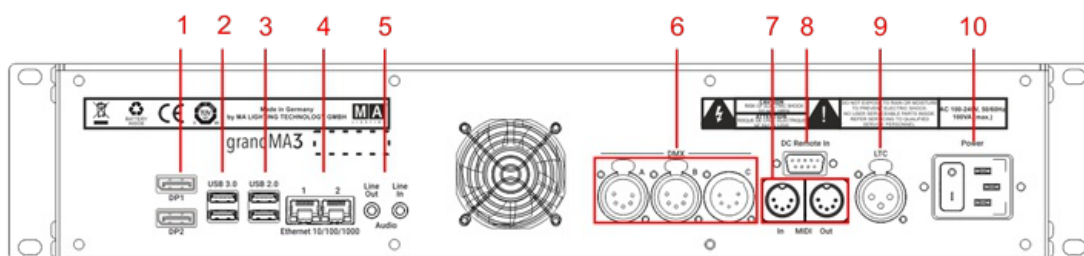
For technical specifications, see [Technical Data](#) in the [grandMA3 Quick Manual consoles](#).

2.12. grandMA3 onPC rack-unit



grandMA3 onPC rack-unit front panel

1. Screen
2. **Power key**
3. **USB 2.0**



grandMA3 onPC rack-unit rear panel

1. **DisplayPort 1+2**
2. **USB 3.0**
3. **USB 2.0**
4. **Ethernet 1+2**
5. **Line In+Out**
6. **DMX A-C**
7. **MIDI In+Out**
8. **DC Remote Control**
9. **LTC**
10. Power switch + **IEC connector**

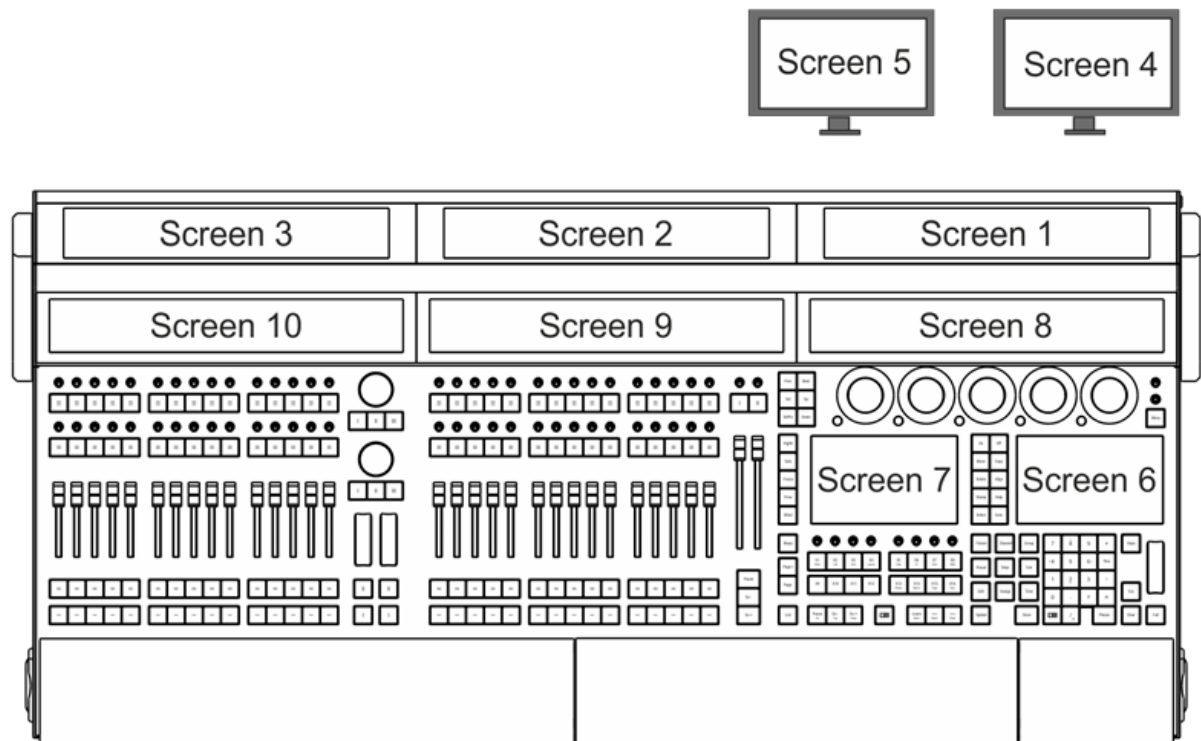
For technical specifications see **Technical Data** in the **grandMA3 Quick Manual onPC rack-unit**.

2.13. Screen Allocation

Numerical Order of Screens

grandMA3 consoles number their screens in a specific order. Different consoles within the range have different amounts of internal screens and connections for external monitors. This topic covers those physical differences as well as the numbering system applied to those screens.

grandMA3 full-size and light



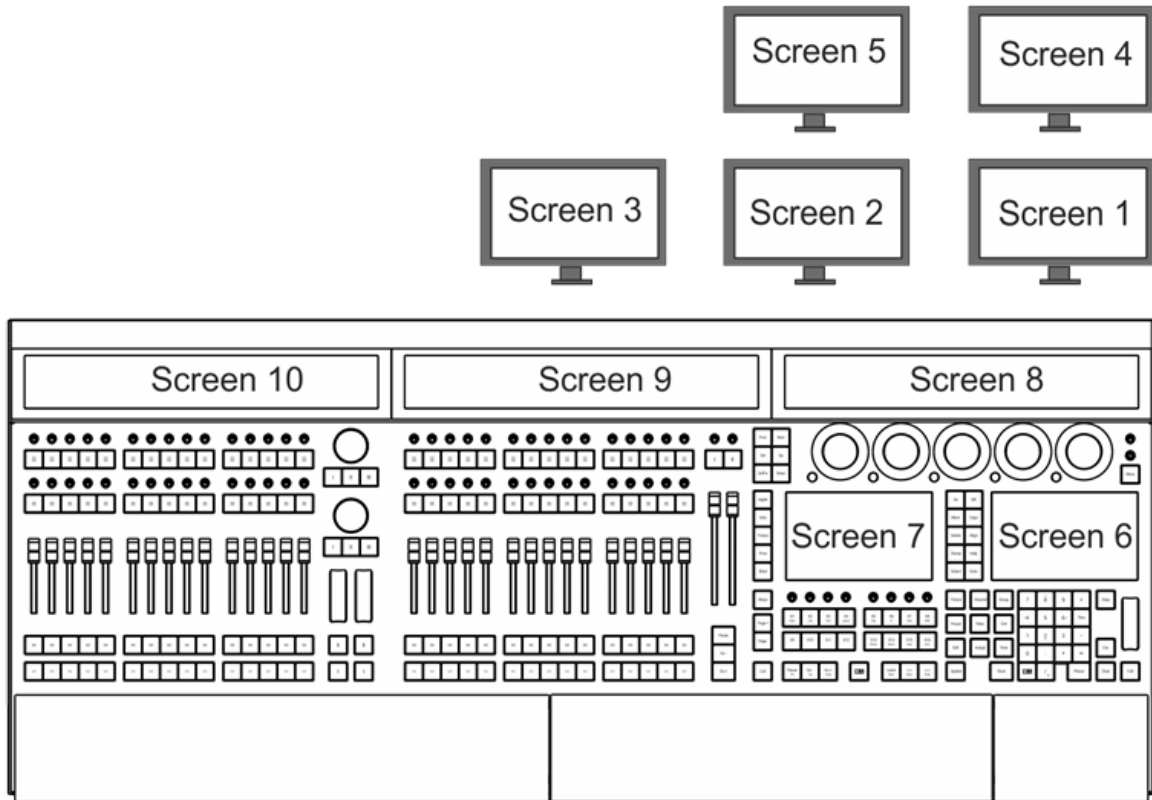
grandMA3 full-size with all screens numbered

The grandMA3 full-size includes three internal 15.6-inch screens mounted in a double-hinged monitor wing, three internal 14.9-inch letterbox screens, two internal 7-inch screens, and two connections for optional external monitors. The grandMA3 light includes two internal 15.6-inch screens mounted in a double-hinged monitor wing, two internal 14.9-inch letterbox screens, two internal 7-inch screens, and two connections for optional external monitors. The numbering of these screens is detailed in the following table:



Screen number	Size	Remark
Screen 1	15.6-inch	
Screen 2	15.6-inch	
Screen 3	15.6-inch	Only on grandMA3 full-size
Screen 4		External screen
Screen 5		External screen
Screen 6	7-inch	Right command screen
Screen 7	7-inch	Left command screen
Screen 8	14.9-inch	Letterbox encoder screen
Screen 9	14.9-inch	Right letterbox executor screen
Screen 10	14.9-inch	Left letterbox executor screen, on grandMA3 full-size or first grandMA3 extension connected to grandMA3 light
Screen 11	14.9-inch	Letterbox executor screen, on grandMA3 extension connected to grandMA3 full-size or second grandMA3 extension connected to grandMA3 light

grandMA3 full-size CRV and light CRV



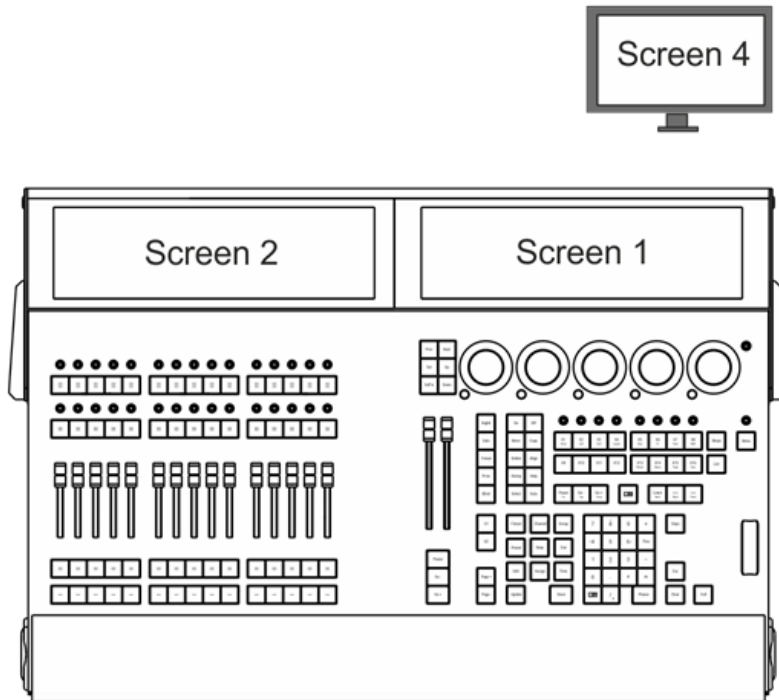
grandMA3 full-size CRV with all screens numbered



The grandMA3 full-size CRV includes three internal 14.9-inch letterbox screens, two internal 7-inch screens, and five connections for optional external monitors. The grandMA3 light CRV includes two internal 14.9-inch letterbox screens, two internal 7-inch screens, and four connections for optional external monitors. The numbering of these screens is detailed in the following table:

Screen number	Size	Remark
Screen 1		External screen
Screen 2		External screen
Screen 3		External screen (only grandMA3 full-size CRV)
Screen 4		External screen
Screen 5		External screen
Screen 6	7-inch	Right command screen
Screen 7	7-inch	Left command screen
Screen 8	14.9-inch	Letterbox encoder screen
Screen 9	14.9-inch	Right letterbox executor screen
Screen 10	14.9-inch	Left letterbox executor screen, on grandMA3 full-size CRV or grandMA3 extension connected to grandMA3 light CRV
Screen 11	14.9-inch	Letterbox executor screen, on grandMA3 extension connected to grandMA3 full-size CRV or second grandMA3 extension connected to grandMA3 light CRV

grandMA3 compact XT and compact

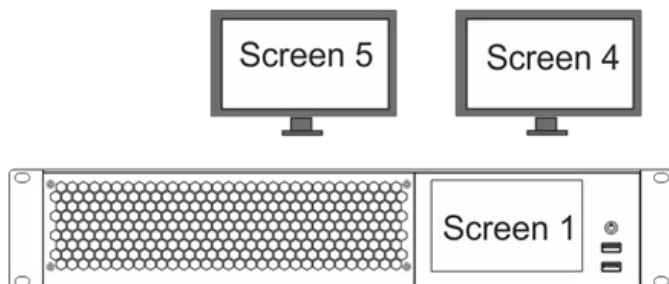


grandMA3 compact XT with all screens numbered

The grandMA3 compact XT includes two internal 15.6-inch screens mounted in a hinged monitor wing and one connection for an optional external monitor. The grandMA3 compact includes one internal 15.6-inch screen mounted in a hinged monitor wing and one connection for an optional external monitor. The numbering of these screens is detailed in the following table:

Screen number	Size	Remark
Screen 1	15.6-inch	
Screen 2	15.6-inch	Only on grandMA3 compact XT
Screen 4		External screen

grandMA3 replay unit



grandMA3 replay unit with all screens numbered

The grandMA3 replay unit includes two connections for optional external monitors. The numbering of these screens is detailed in the following table:

Screen number	Size	Remark
Screen 4		External screen
Screen 5		External screen
Screen 10	14.9-inch	Letterbox executor screen on first grandMA3 extension connected to grandMA3 replay unit
Screen 11	14.9-inch	Letterbox executor screen on second grandMA3 extension connected to grandMA3 replay unit
Screen 12	14.9-inch	Letterbox executor screen on third grandMA3 extension connected to grandMA3 replay unit

For more information on connecting external screens, see the [Connect External Screens](#) topic.

grandMA3 onPC rack-unit and grandMA3 onPC command wing XT

For onPC stations, the single windows for each display can be freely arranged. For more information, see the [Displays in grandMA3 onPC](#) topic.

2.14. Keyboard Shortcuts

Keyboard shortcuts allow fast operation of the console and onPC command wings via the (internal) keyboard.

Some of these user-editable shortcuts are for general use and override other commands. Others are related to specific windows and pop-ups.

The **KeyboardShortcut keyword** is used to edit and list the keyboard shortcuts. To modify the keyboard shortcuts, use the **Set keyword**.

Keyboard Shortcuts are part of the UserProfile. This makes it possible for every user/user profile to use their own shortcut definitions within a show file.

Turn the Keyboard Shortcuts on

- Tap **ShCuts** next to the command line or press **F10** on your keyboard.



Keyboard shortcuts enabled

The keyboard shortcuts are turned on (yellow text).

Turn the Keyboard Shortcuts off

- Tap **ShCuts** next to the command line or press **F10** on your keyboard.



Keyboard shortcuts disabled

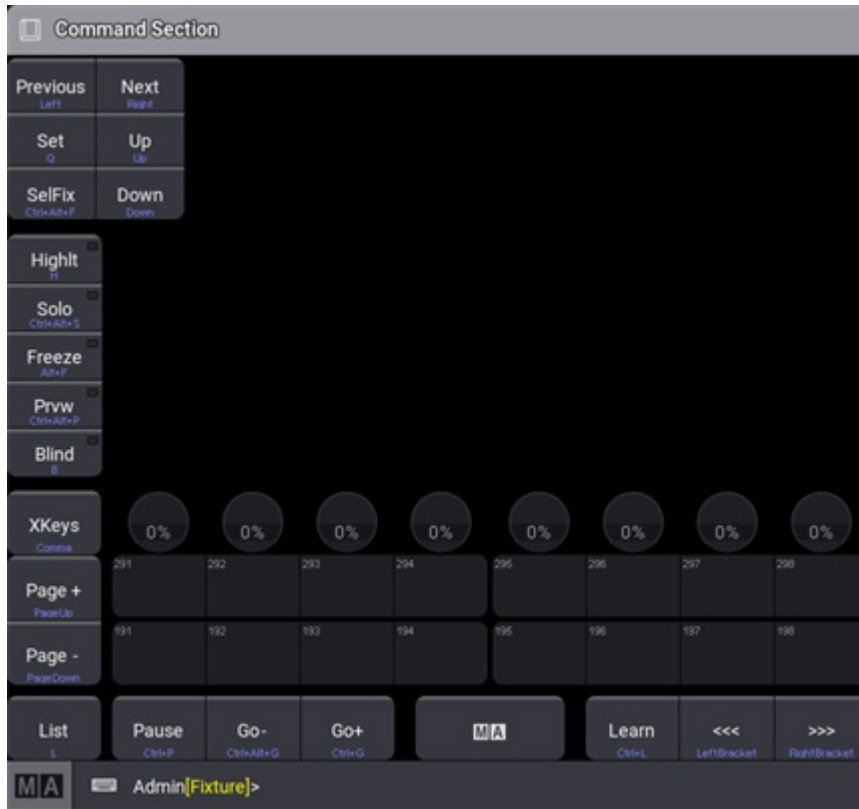
The keyboard shortcuts are turned off (white text).



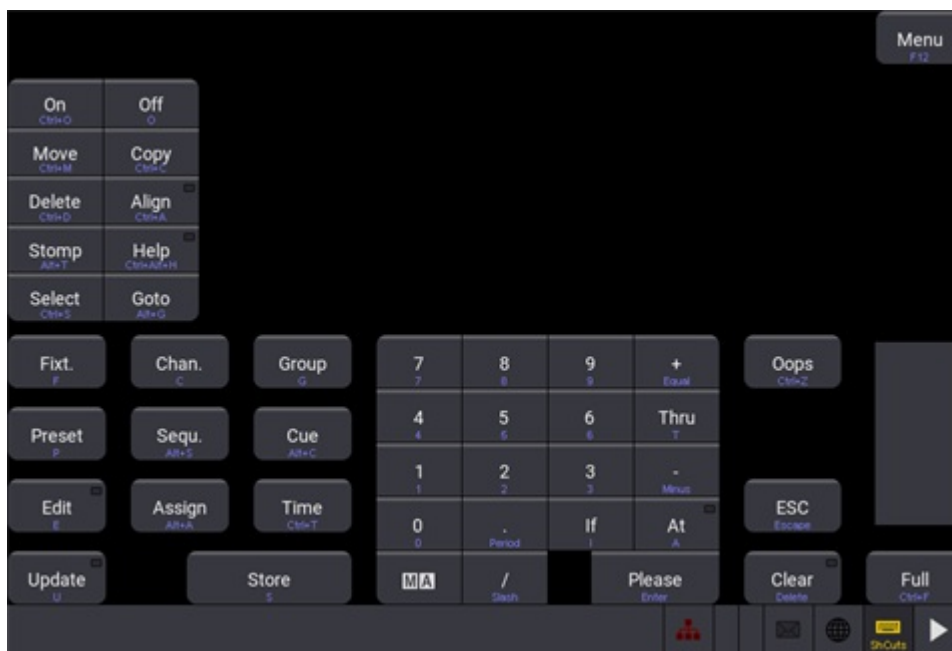
Hint:

In certain overlays and pop-ups where the shortcuts are not relevant, they are switched off and back on automatically. The shortcut symbol turns red.

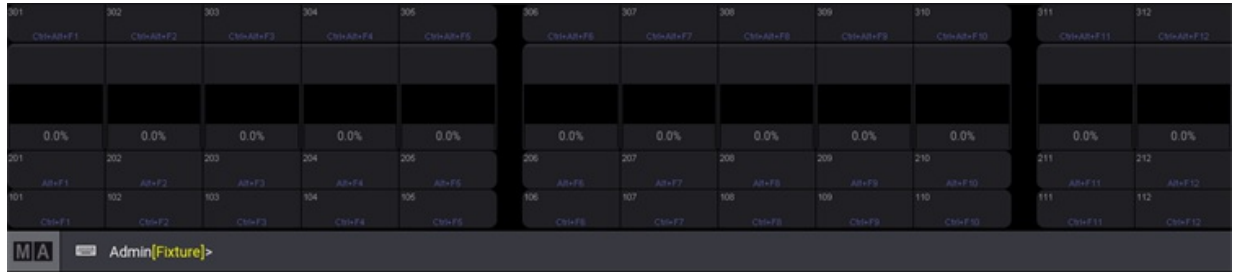
Shortcut Examples



Command section window left



Command section window right



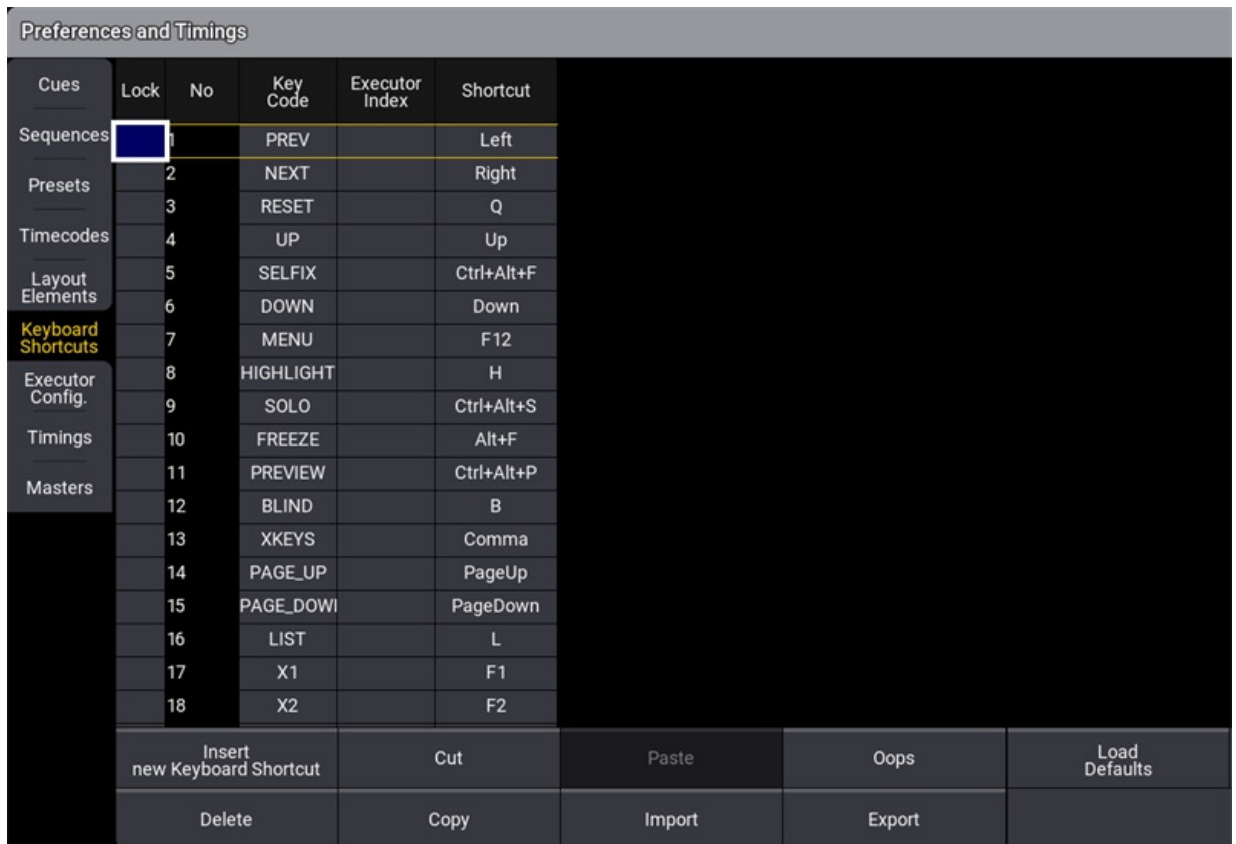
Playback controls window

The images above show the activated shortcut overlay. The overlay color can be changed in the **color theme**.

Keyboard Shortcuts Window

1. To edit the shortcuts, open the Preferences and Timings window.
2. Tap **Keyboard Shortcuts**.

The Keyboard Shortcuts window opens:



Edit Keyboard Shortcuts window



Add Keyboard Shortcuts

1. To add a new keyboard shortcut at the end of the list, scroll down and tap **New KeyboardShortcut** and then tap **Insert new Keyboard Shortcut**. A new empty cell is generated at the end of the list.

Preferences and Timings							
Cues	Lock	No	Key Code	Executor Index	Shortcut		
Sequences		141	PLEASE		Enter		
Presets		142	FULL		Ctrl+F		
		143	OOPS		Ctrl+Z		
Timecodes		144	CLEAR		Delete		
		145	ESC		Escape		
Layout Elements		146	MINUS		kpSubtract		
		147	SLASH		kpDivide		
Keyboard Shortcuts		148	PLUS		kpAdd		
		149	PLEASE		Enter		
Executor Config.		150	DOT		kpDecimal		
		151	OOPS		Backspace		
Timings		152	ONPC_SCRE		Ctrl+Alt+2		
		153	ONPC_SCRE		Ctrl+Alt+3		
Masters		154	ONPC_SCRE		Ctrl+Alt+4		
		155	ONPC_SCRE		Ctrl+Alt+5		
		156	ONPC_SCRE		Ctrl+Alt+6		
		157	ONPC_SCRE		Ctrl+Alt+7		
				New Keybo			
			Insert new Keyboard Shortcut	Cut	Paste	Oops	Load Defaults
			Delete	Copy	Import	Export	

Add new keyboard shortcut

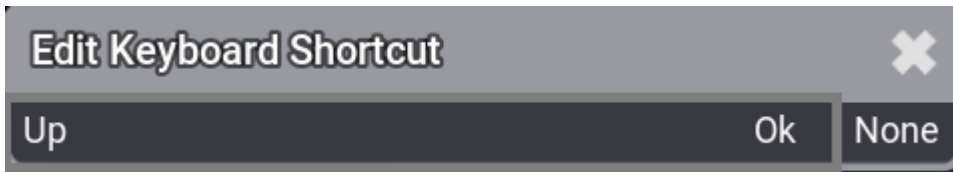
2. Tap and hold or right click on the empty cell. The Select Key Code pop-up opens.
3. Select the desired key code from the pop-up list. The pop-up closes.



Select Key Code	PAGE_UP	X11	GOBACK	SELECT	NUM0	DOT	FLASH	ONPC_SCREEN2	NEXT_STEP
PREV	PAGE_DOWN	X12	GO	GOTO	NUM1	IF	BLACK	ONPC_SCREEN3	STEP
NEXT	LIST	X13	LEARN	FIXTURE	NUM2	AT	KILL	ONPC_SCREEN4	TOGGLE_STEP
RESET	X1	X14	GOBACKFAST	CHANNEL	NUM3	DMX	RATE1	ONPC_SCREEN5	TOGGLE_MATRICKS
UP	X2	X15	GOFAST	GROUP	NUM4	SLASH	TEMP	ONPC_SCREEN6	RESET_MATRICKS
SELFIX	X3	X16	ON	SEQUENCE	NUM5	ASTERISK	TOGGLE	ONPC_SCREEN7	CLONE
DOWN	X4	EXEC	OFF	CUE	NUM6	PLEASE	TOP	PREV_X	GRID
MENU	X5	EXECUTOR	MOVE	PRESET	NUM7	FULL	LOAD	PREV_Y	LAYOUT
SOLO	X6	DEF_GO	COPY	EDIT	NUM8	OOPS	HIGHLIGHT	PREV_Z	TIMECODE
FREEZE	X7	DEF_PAUSE	DELETE	ASSIGN	NUM9	CLEAR	LOWLIGHT	PREV_STEP	VIEW
PREVIEW	X8	DEF_GOBACK	ALIGN	TIME	PLUS	ESC	SWAP	NEXT_X	PHASER
BLIND	X9	PAUSE	STOMP	UPDATE	THRU	USER1	HALF_SPEED	NEXT_Y	MACRO
XKEYS	X10	FIX	HELP	STORE	MINUS	USER2	DOUBLE_SPEED	NEXT_Z	PAGE

Select key code window

- To open the Edit Keyboard Shortcut pop-up, tap and hold or right click on the empty cell in the Shortcut row.
- To enter the desired shortcut, press the relevant keys. The pop-up closes.

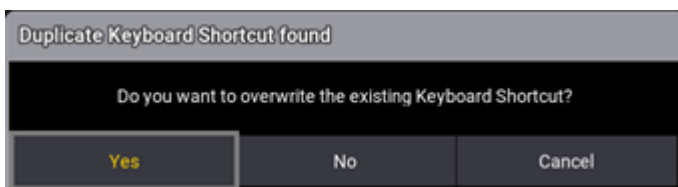


Edit Keyboard shortcut pop-up

As **Enter** can also be used as a shortcut, it is not possible to close the Edit keyboard shortcut pop-up with Enter.

- To confirm the shortcut tap **Ok**.
- To clear the shortcut, tap **None**.
- To cancel the edit, tap **X**.

- If a keyboard shortcut already exists, a warning pop-up appears. Tap **Yes** to save the new shortcut.



Warning pop-up



7. The new shortcut is added to the list.

Preferences and Timings						
Cues	Lock	No	Key Code	Executor Index	Shortcut	
Sequences		142	FULL		Ctrl+F	
Presets		143	OOPS		Ctrl+Z	
		144	CLEAR		Delete	
Timecodes		145	ESC		Escape	
	Layout Elements	146	MINUS		kpSubtract	
		147	SLASH		kpDivide	
Keyboard Shortcuts		148	PLUS		kpAdd	
	Executor Config.	149	PLEASE		Enter	
		150	DOT		kpDecimal	
Timings		151	OOPS		Backspace	
	Masters	152	ONPC_SCREEN2		Ctrl+Alt+2	
153		ONPC_SCREEN3		Ctrl+Alt+3		
154		ONPC_SCREEN4		Ctrl+Alt+4		
155		ONPC_SCREEN5		Ctrl+Alt+5		
156		ONPC_SCREEN6		Ctrl+Alt+6		
157		ONPC_SCREEN7		Ctrl+Alt+7		
158		UP		Up		
			New KeyboardShortcut			
		Insert new Keyboard Shortcut	Cut	Paste	Oops	Load Defaults
		Delete	Copy	Import	Export	

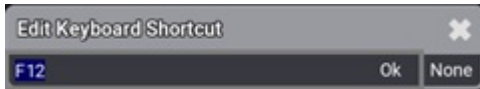
New shortcut is added



Edit Existing Keyboard Shortcut

1. To edit an existing shortcut definition, tap and hold the Keyboard Shortcut you want to edit.

The Edit keyboard shortcut pop-up opens



Edit keyboard shortcut pop-up

2. Enter the new shortcut.

Delete Keyboard Shortcuts

1. To select the keyboard shortcut to be deleted, tap the desired shortcut.

Preferences and Timings					
Cues	Lock	No	Key Code	Executor Index	Shortcut
Sequences		1	PREV		Left
Presets		2	NEXT		Right
		3	RESET		Q
Timecodes		4	UP		
Layout Elements		5	SELFIX		Ctrl+Alt+F
		6	DOWN		Down
Keyboard Shortcuts		7	MENU		F12
Executor Config.		8	HIGHLIGHT		H
		9	SOLO		Ctrl+Alt+S
Timings		10	FREEZE		Alt+F
Masters		11	PREVIEW		Ctrl+Alt+P
		12	BLIND		B
		13	XKEYS		Comma
		14	PAGE_UP		PageUp
		15	PAGE_DOWN		PageDown
		16	LIST		L
		17	X1		F1
		18	X2		F2
Insert new Keyboard Shortcut			Cut		Paste
Delete			Copy		Import
					Oops
					Export
					Load Defaults

Keyboard shortcuts window



2. Tap **Delete**.

Preferences and Timings							
Cues	Lock	No	Key Code	Executor Index	Shortcut		
Sequences		1	PREV		Left		
Presets		2	NEXT		Right		
		3	RESET		Q		
Timecodes		4	UP				
Layout Elements		5	SEFIX		Ctrl+Alt+F		
		6	DOWN		Down		
Keyboard Shortcuts		7	MENU		F12		
Executor Config.		8	HIGHLIGHT		H		
		9	SOLO		Ctrl+Alt+S		
Timings		10	FREEZE		Alt+F		
Masters		11	PREVIEW		Ctrl+Alt+P		
		12	BLIND		B		
		13	XKEYS		Comma		
		14	PAGE_DOWN		PageDown		
		15	LIST		L		
		16	X1		F1		
		17	X2		F2		
		18	X3		F3		
Insert new Keyboard Shortcut			Cut		Paste	Oops	Load Defaults
Delete			Copy		Import	Export	

Keyboard shortcut is deleted

3. The keyboard shortcut is deleted. To undo deletion, tap **Oops**.



Reset Keyboard Shortcuts to Defaults

- To reset the keyboard shortcuts to their defaults, tap **Load Defaults**. The keyboard shortcuts are reset to their defaults.



Hint:

The **Shift** keys on the keyboard correspond with the **MA** keys on the console.

Import / Export Keyboard Shortcuts

- To import or export the keyboard shortcuts, tap **Import** or **Export**.

By default, files will be exported to the relevant folder within the library folder structure, either on the local drive of the console or onPC station, or on a selected USB drive. For more information about this folder structure, see the **Folder Structure** topic.

For information about exporting and importing show data using command line syntax without the use of these menus, see the **Export keyword** and **Import keyword** topics.

2.14.1. Framework Shortcuts

The framework of the grandMA3 software uses the following keyboard shortcuts for user interface related grids, for example, in the Patch menu or the Fixture Builder.

Press key	Command	Description
Insert	Insert UIGridSelection	Executing a UI related insert function.
Ctrl + Insert	Store UIGridSelection	Executing a UI related store function.
Delete	Delete UIGridSelection	Executing a UI related delete function.
Ctrl + C	Copy UIGridSelection	Copying from a UI grid.
Ctrl + V	Paste UIGridSelection	Pasting into a UI grid.
Ctrl + X	Cut UIGridSelection	Cutting from a UI grid.
Ctrl + N	-	Toggles, for example, the "New Object Line" in a UI grid.
Ctrl + M	-	Toggles Merge children .
Ctrl + Enter	Edit UIGridSelection	Opens the Edit overlay for a selected cell.
Ctrl + Alt + D		Resets the color theme.



Hint:

On Apple® products, use **control** instead of Ctrl and **option** instead of Alt.



2.15. Keys

A shortcut is a series of one or several keys that invoke the software to perform a pre-programmed action.

In grandMA3, we differentiate between the term key and button.

A key is a (hard) key on the console or on the (integrated) keyboard.

A button is an element in the User Interface.

The keys **F9**, **F10**, **F11**, **Pause** and **Print Screen** provide system-wide shortcuts on all operating systems always available in all contexts as long as the program is running. These shortcuts cannot be edited.

The following shortcuts are available on all OS through several input methods:

Desk lock:

- Keyboard
 - **F9**
 - **Pause**
- Key
 - **MA** + **MA** + **Pause|Fix**
- Button
 - **Lock**

Toggle keyboard shortcuts:


- Keyboard
 - **F10**
- Button
 - **ShCuts**

Print screen:

- Keyboard
 - **F11**
 - **Print**

Reload UI (If the color theme is set to unreadable values):

- Key
 - **MA** + **MA** + **Clear**

 **Hint:**
The Shift keys on the keyboard correspond with the **MA** keys on the console.

All other shortcuts are user-editable.

This overview displays shortcuts for a quick execution of commands.

Press key	Command	Description
.	.	Sets a dot in the command line.
. .	<u>Zero</u>	Sets the selected fixtures dimmer value to 0.
MA .	<u>Default</u>	Sets the attributes to its default values.
<<<	<<<	Goes fast backward and ignores the cue timing on a given executor.
MA <<<	<u>Black</u>	Temporarily overrides the master level to zero on executing objects.
>>>	>>>	Goes fast forward and ignores the cue timing on a given executor.
MA >>>	<u>Flash</u>	Temporarily overrides the master level to full on a given executor.
-	=	Sets a minus in the command line.
- -	<u>At</u> -10	Decreases the intensity of the selected fixtures by 10.
+	<u>+</u>	Adds a plus to the command line.
+ +	<u>At</u> +10	Increases the intensity of the selected fixtures with +10.
/	/	Sets a slash in the command line.
MA /	<u>*</u>	Sets an asterisk in the command line.
At	<u>At</u>	Sets the values on the selected fixtures.
At At	<u>Normal</u>	Sets the intensity to Normal on the selected fixtures.
MA At	<u>Integrate</u>	Integrates preset values into another step of the phaser.
MA At At	<u>Extract</u>	Breaks any referenced links and applies the hard values.
Press and hold At		Opens the <u>At filter</u> (At filter overlay).
Store At	/Embed	With store in the command line, at adds /Embed to the command.
Assign	<u>Assign</u>	Assigns functions or objects to other objects.
Assign Assign	<u>Label</u>	Labels objects.
MA Assign	<u>Set</u>	Sets values to object properties.
Align		Cycles through the different align modes.

Press key	Command	Description
Press and hold Align		Sets the align mode to off.
MA Align		Cycles through the different align transitions.
Press and hold MA Align		Sets the align transition to linear.
Blind	<u>Blind</u>	Switches blind mode on and off.
Clear	<u>Clear</u>	Clears the selected fixtures.
Clear Clear	<u>Clear</u>	Clears the active values.
Clear Clear Clear	<u>Clear</u>	Clears the programmer.
Press and hold Clear	<u>ClearAll</u>	Clears the programmer + command filter. The same as 3 x Clear .
Channel	<u>Channel</u>	Selects fixtures by its Channel ID's.
Channel Channel		Cycles through the custom made ID types.
Copy	<u>Copy</u>	Creates a copy of an object.
Copy Copy	<u>Paste</u>	Pastes previously copied or cut objects.
Copy Copy Copy	<u>Insert</u>	Inserts previously copied or cut objects.
MA Copy	<u>Cut</u>	Cuts content to the clipboard.
Ctrl		Press and hold for multi-selection in sheets. Press and hold when the Mode Selector pop-up appears during the boot process for clean start.
Cue	<u>Cue</u>	Cue is an object type.
Cue Cue	<u>Part</u>	Part is an object type.
MA Cue	<u>Programmer</u>	Enters the Programmer keyword in the command line.
Down		Navigates down in the fixture/subfixture structure.
Delete	<u>Delete</u>	Use Delete to remove something from the show file.
Delete Delete	<u>Remove</u>	Loads Remove in the programmer.
Delete Delete Delete	<u>Release</u>	Loads Release in the programmer.
ESC		Closes pop-ups, returning keyboard focus to the command line. If there are no open pop-ups, ESC deletes unexecuted commands from the command line.
Edit	<u>Edit</u>	Opens an editor for objects like cues, presets, sequences, and macros.
Edit Edit	<u>EditSetting</u>	Adds EditSetting into the command line.
Freeze	<u>Freeze</u>	Toggles the Freeze function on and off.
Full	<u>Full</u>	Sets the intensity for the selected fixtures to 100%.
Fixture	<u>Fixture</u>	Selects the fixtures of a given ID.
Fixture Fixture	<u>Selection</u>	The actual programmer selection.



Press key	Command	Description
Group	<u>Group</u>	Selects a group of a given ID.
Group Group	<u>World</u>	Selects a world of a given ID.
Group Group Group	<u>Filter</u>	Selects a filter of a given ID.
Goto	<u>Goto</u>	Goes to a cue in the selected sequence.
Goto Goto	<u>Load</u>	Loads a cue from the selected sequence.
MA Goto Goto	<u>Loaded</u>	Adds Loaded to the command line.
Go+ Go+	<u>Unpark</u>	Releases a parked DMX channel or universe.
Go+ [Large]	<u>Go+ Executor</u>	Executes the next cue in the selected sequence.
MA Go+ [Large]	<u>Go+ Loaded</u>	Executes all the loaded cues at the same time.
Go- [Large]	<u>Go- Executor</u>	Executes the previous cue in the selected sequence.
Go+ Temp	<u>Go+</u>	Executes the next cue in the sequence.
MA Go+ Temp	<u>Temp</u>	Enables an executor only as long as that executor is held.
MA Go+ Temp Go+ Temp	<u>Toggle</u>	Toggles between on and off status.
Go- Top	<u>Go-</u>	Executes the previous cue in the sequence.
MA Go- Top	<u>Top</u>	Executes the first cue in the sequence.
MA Go- Top Go- Top	<u>Kill</u>	Executes the Go+ function on sequences and presets and disables all other played back sequences and presets.
Help	<u>Help</u>	Displays the help topic of a key or view.
Hight	<u>Highlight</u>	Toggles the highlight function on and off.
MA Hight	<u>Lowlight</u>	Toggles the lowlight function on and off.
If	<u>IfOutput</u>	Selects fixtures based on their current output.
If If	<u>IfActive</u>	Selects fixtures with active values in the programmer.
If If If	<u>IfProgrammer</u>	Selects fixtures that contain values in the programmer.
If If If If	<u>If</u>	Selects fixtures that exist in both groups.
Learn Rate1	<u>Learn</u>	Learn speed for a chaser or a phaser.
MA Learn Rate1	<u>Rate1</u>	Sets the rate to 1:1 for the executor or speed group.
List	<u>List</u>	Lists the sub-directories in the command line.
MA List	<u>ListReference</u>	Lists references and dependencies of an object.
MA		Pressing and holding MA in combination with other keys, gives shortcuts to other functions. Same as Shift on a keyboard.
MA Please		Forces the focus to the command line.
Menu		Opens the menu pop-up.
Menu Menu	<u>SaveShow</u>	Saves the show file using the current name.



Press key	Command	Description
Move	<u>Move</u>	Moves objects to another location.
Move Move	<u>Exchange</u>	Function keyword used to swap places.
Next	<u>Next</u>	Selects the next fixture.
MA Next	<u>Next Step</u>	Selects the next step.
Oops	<u>Oops</u>	Undoes the last operation or selection.
Press and hold Oops		Opens the Oops overlay.
On	<u>On</u>	Starts an executor or activates attributes.
Off	<u>Off</u>	Releases an executor or deactivates attributes.
Off Off		Opens the off menu.
MA MA Off		Closes any active RemoteHID connections.
Pause [Large]	<u>Pause Executor</u>	Toggles pause on and off for the selected sequence.
Pause Fix	<u>Pause</u>	Pauses a crossfade or a phaser.
Pause Fix Pause Fix	<u>Park</u>	Enters the park keyword in the command line.
MA Pause Fix	<u>Fix</u>	Fixes an executor for all executor pages.
MA MA Pause Fix		Toggles the desk lock.
Page+	<u>Next Page</u>	Calls the next page.
Page-	<u>Previous Page</u>	Calls the previous page.
Press and hold Page-	<u>Page</u>	Calls page 1.
Please		Executes the syntax written in the command line.
Please Please		Activates all attributes for the selected fixtures. Additional presses toggle this activation.
Power		Turns the console on and off.
Preset + 1 , 2 etc.		Changes the feature group in the encoder bar
Preset	<u>Preset</u>	Selects the preset of a given ID.
Preset Preset	<u>Attribute</u>	Selects the attribute of a given ID.
Preset Preset Preset	<u>Gel</u>	Selects the gel of a given ID.
MA Preset	<u>FeatureGroup</u>	Selects the feature group of a given ID.
MA Preset Preset	<u>DataPool</u>	Selects the data pool of a given ID.
Prev	<u>Previous</u>	Selects the previous fixture.
MA Prev	<u>Previous Step</u>	Selects the previous step.
Select	<u>Select</u>	Selects objects as defaults.
Select Select	<u>Collect</u>	Adds objects of the same type to a collection.
Selfix	<u>SelectFixtures</u>	Selects all fixtures from an executor or another object.
Selfix Selfix	<u>Shuffle</u>	Shuffles the order of the fixture selection.
Sequ	<u>Sequence</u>	Selects the sequence of a given ID.



Press key	Command	Description
Set	<u>Toggle Selection MATricks</u>	Restores the selection. Toggles MATricks on and off.
MA Set	<u>Step Toggle</u>	Toggles between the first step selected and all steps selected.
Solo	<u>Solo</u>	Toggles the solo function on and off.
Stomp	<u>Stomp</u>	For selected parameters with a running phaser, Stomp the absolute layer to the default value and sets the relative layer to 0 in the programmer. All other steps are deleted from the programmer.
Stomp Stomp	<u>Capture</u>	Translates the combined instantaneous output of selected fixtures to the absolute layer of a single step in the programmer. All other steps will be deleted from the programmer.
Store	<u>Store</u>	Stores objects in the show file.
Press and hold Store		Opens the Store Options pop-up.
Thru	<u>Thru</u>	Selects a range of items.
Time	<u>Time key</u>	Allows access to parameter and cue timing as well as relative and absolute value layers.
Up		Navigates up in the fixture/subfixture structure.
Update	<u>Update</u>	Opens the update pop-up.
MA Update	<u>Cook</u>	Converts recipe data into value, timing, and/or phaser data.
X1 Clone		At the moment all Xkeys behave like executors. X1 is executor 291.
MA X1 Clone	<u>Clone</u>	Copies data from one fixture to another.
MA X1 Clone X1 Clone	<u>Recast</u>	Updates dependent objects when attributes are added to or deleted from presets.
X2 Link		At the moment all Xkeys behave like executors. X2 is executor 292.
X3 Grid		At the moment all Xkeys behave like executors. X3 is executor 293.
MA X3 Grid	<u>Grid</u>	Sets the grid cursor in the Selection Grid window.
X4 Layout		At the moment all Xkeys behave like executors. X4 is executor 294.
MA X4 Layout	<u>Layout</u>	Selects the layout of a given ID.
MA X4 Layout X4 Layout	<u>Appearance</u>	Puts some colors on buttons backgrounds and borders of a given object.




Press key	Command	Description
MA X4 Layout X4 Layout X4 Layout	<u>Scribble</u>	Small drawings on the background of objects.
X5 Step		At the moment all Xkeys behave like executors. X5 is executor 295.
MA X5 Step	<u>Step</u>	Selects the step of a given ID.
X6 TC		At the moment all Xkeys behave like executors. X6 is executor 296.
MA X6 TC	<u>Timecode</u>	Selects the timecode of a given ID.
MA X6 TC X6 TC	<u>TimecodeSlot</u>	Addresses the timecode slots.
MA X6 TC X6 TC X6 TC	<u>Timer</u>	Access the timer objects.
X7 View		At the moment all Xkeys behave like executors. X7 is executor 297.
MA X7 View	<u>View</u>	Selects the view of a given ID.
MA X7 View X7 View	<u>ViewButton</u>	Selects the view button of a given ID.
MA X7 View X7 View X7 View	<u>ScreenContent</u>	Defines the size of a screen or has all the windows as content.
X8 DMX		At the moment all Xkeys behave like executors. X8 is executor 298.
MA X8 DMX	<u>DMXUniverse</u>	Selects the DMX universe of a given value.
MA X8 DMX X8 DMX	<u>DMXAddress</u>	Selects DMX addresses, using absolute numbering.
X9		At the moment all Xkeys behave like executors. X9 is executor 191.
X10		At the moment all Xkeys behave like executors. X10 is executor 192.
X11		At the moment all Xkeys behave like executors. X11 is executor 193.
X12		At the moment all Xkeys behave like executors. X12 is executor 194.
X13 Phaser		At the moment all Xkeys behave like executors. X13 is executor 195.
MA X13 Phaser	<u>Phaser Editor</u>	
X14 Macro		At the moment all Xkeys behave like executors. X14 is executor 196.
MA X14 Macro	<u>Macro</u>	Selects the macro of a given ID.
MA X14 Macro X14 Macro	<u>Plugin</u>	Access plugins.
MA X14 Macro X14 Macro X14 Macro	<u>Quickey</u>	Enters the Quickey keyword in the command line.
X15 Page		At the moment all Xkeys behave like executors. X15 is executor 197.




Press key	Command	Description
MA X15 Page	<u>Page</u>	Selects the page of a given ID.
X16 Exec		At the moment all Xkeys behave like executors. X16 is executor 198.
MA X16 Exec	<u>Executor</u>	Selects the executor of a given ID.
MA X16 Exec X16 Exec	<u>SpecialExecutor</u>	Enters the SpecialExecutor keyword in the command line.
MA X16 Exec X16 Exec X16 Exec	<u>FaderMaster</u>	Enters the FaderMaster keyword in the command line.





2.15.1.. [Dot] Key

Pressing  [Dot] enters a . into the command line.

```
 User name[fixture]> Store Cue 1.2
```



Zero

Pressing   enters the Zero keyword into the command line.

```
 OK : At Zero
```


For more information about Zero, see the [Zero keyword](#) topic.

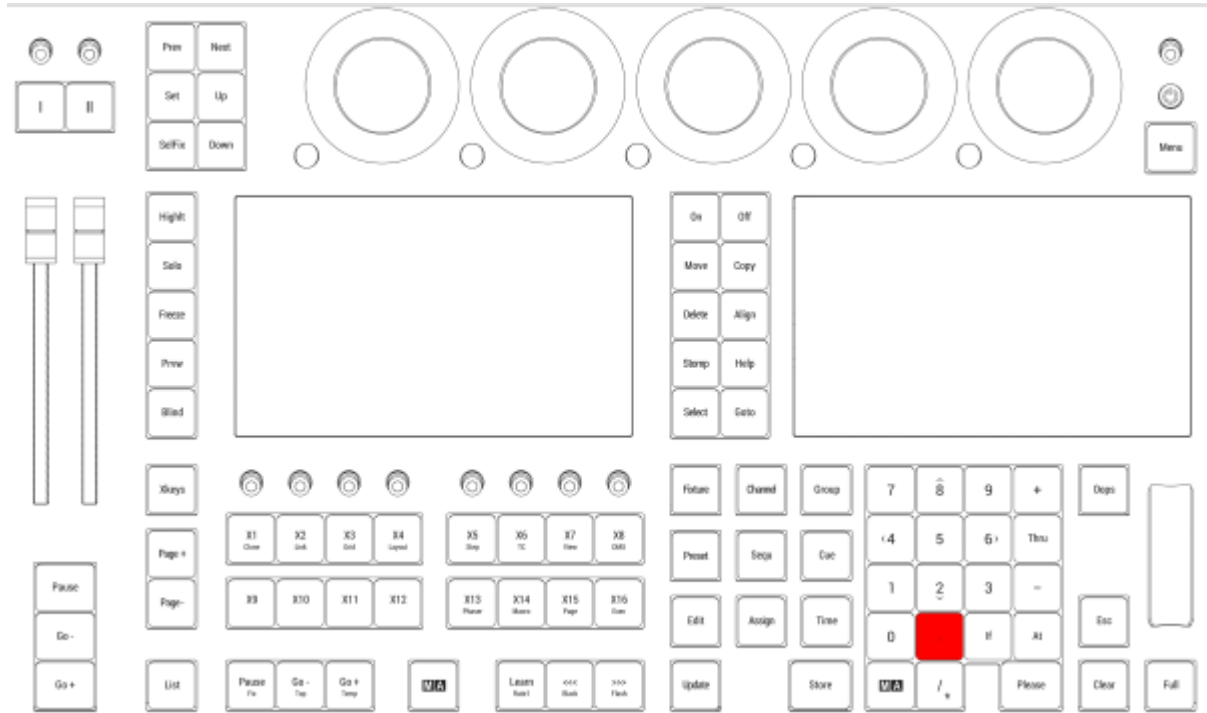
Default

Pressing  +  enters the Default keyword into the command line.

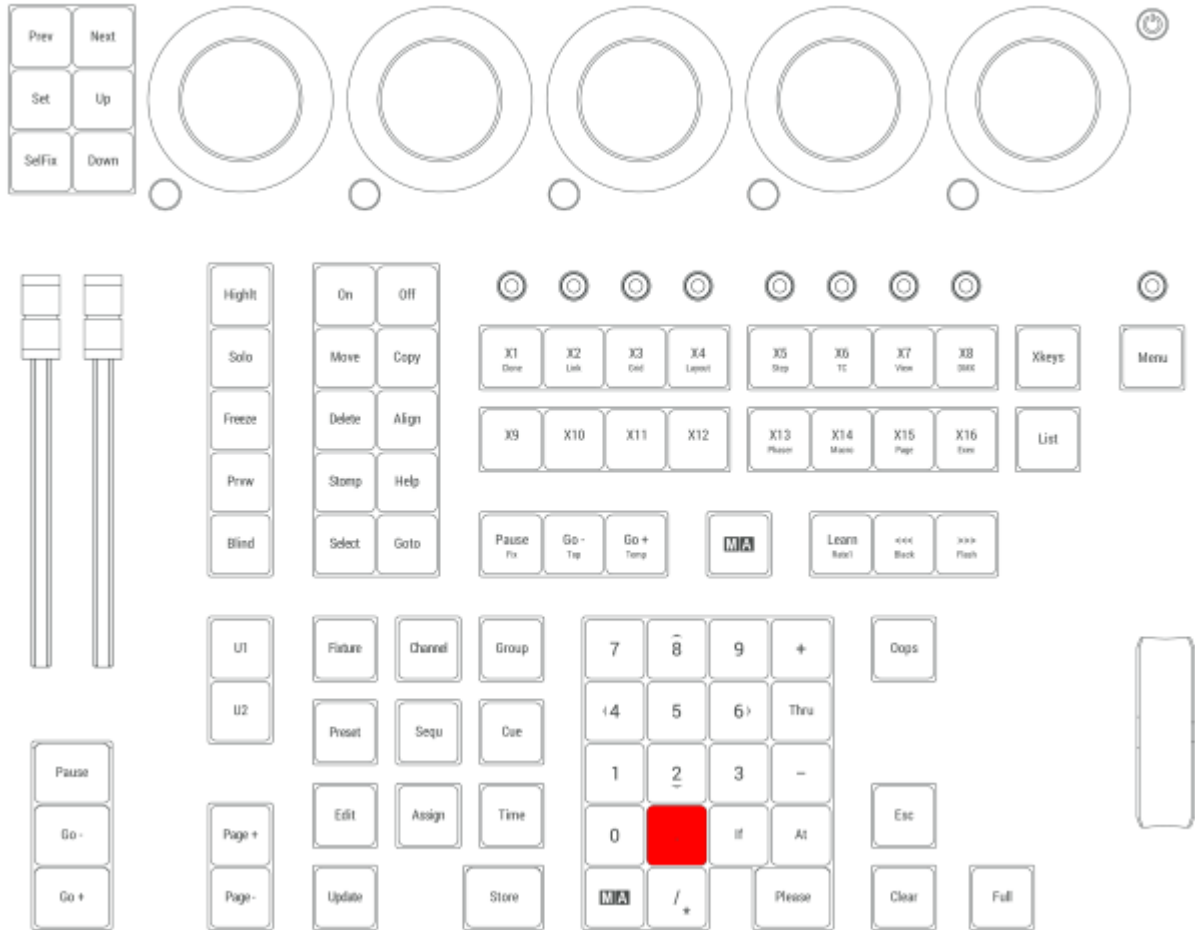
```
 User name[Fixture]> Default
```

For more information about Default, see the [Default keyword](#) topic.

The  [Dot] is located in the numeric keys section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.2. <<< [GoFastBackward] | Black Key

Pressing <<< enters the <<< keyword into the command line.



For more information about <<<, see the [GoFastBackward keyword](#) topic.

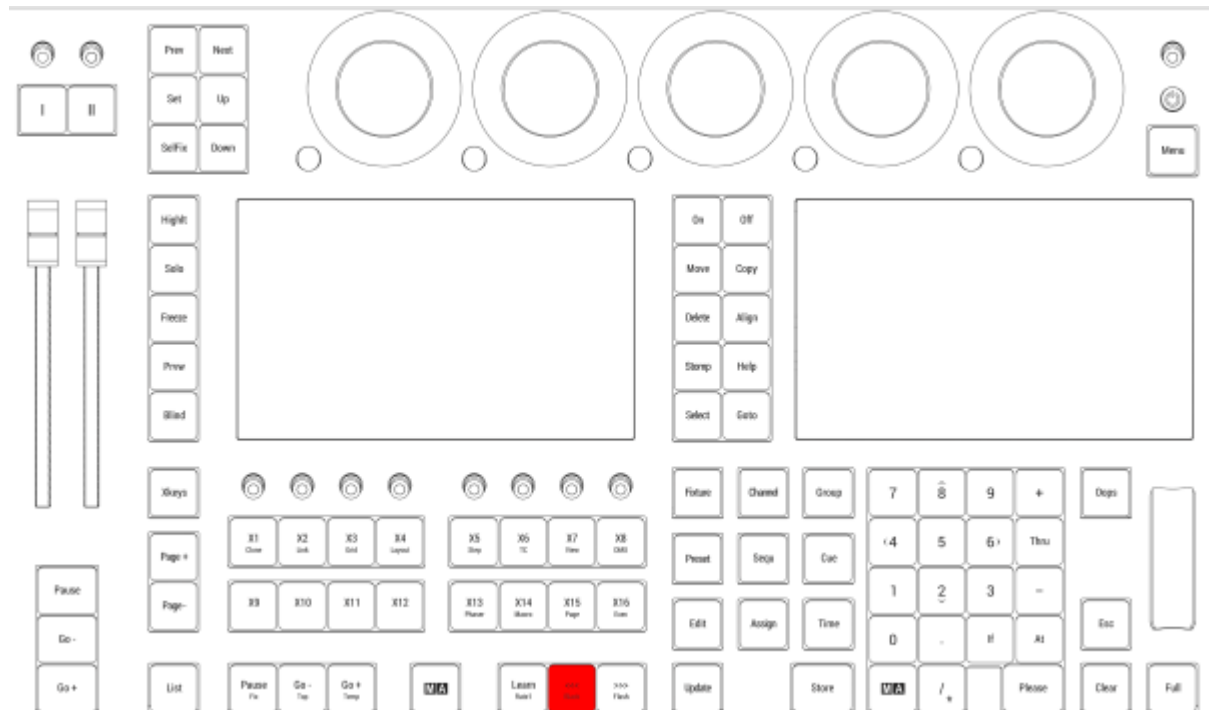
Black

Pressing and holding MA + <<< enters the Black keyword into the command line.

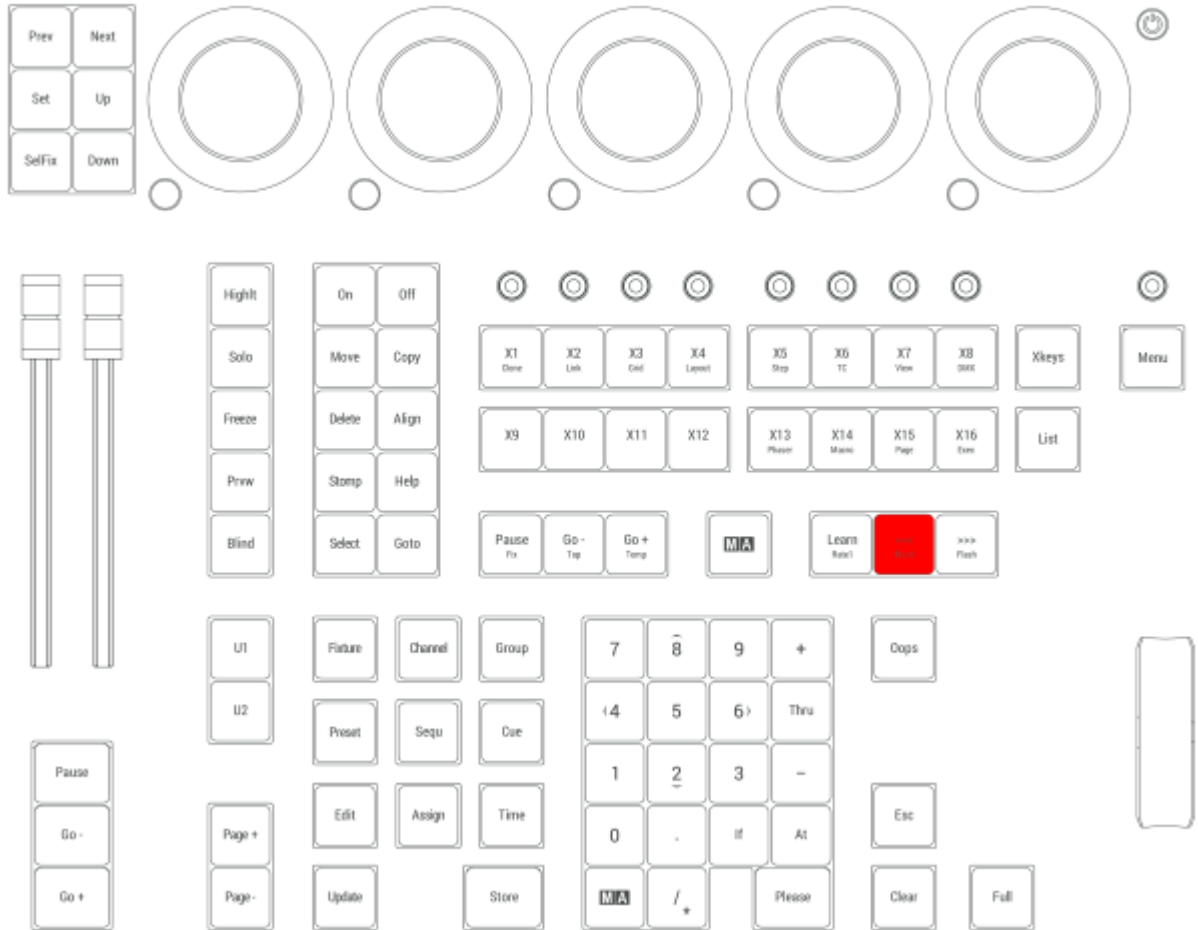


For more information about Black, see the [Black keyword](#) topic.

<<< is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.3. >>> [GoFastForward] | Flash Key

Pressing **>>>** enters the >>> keyword into the command line.



For more information about >>>, see the [GoFastForward keyword](#) topic.

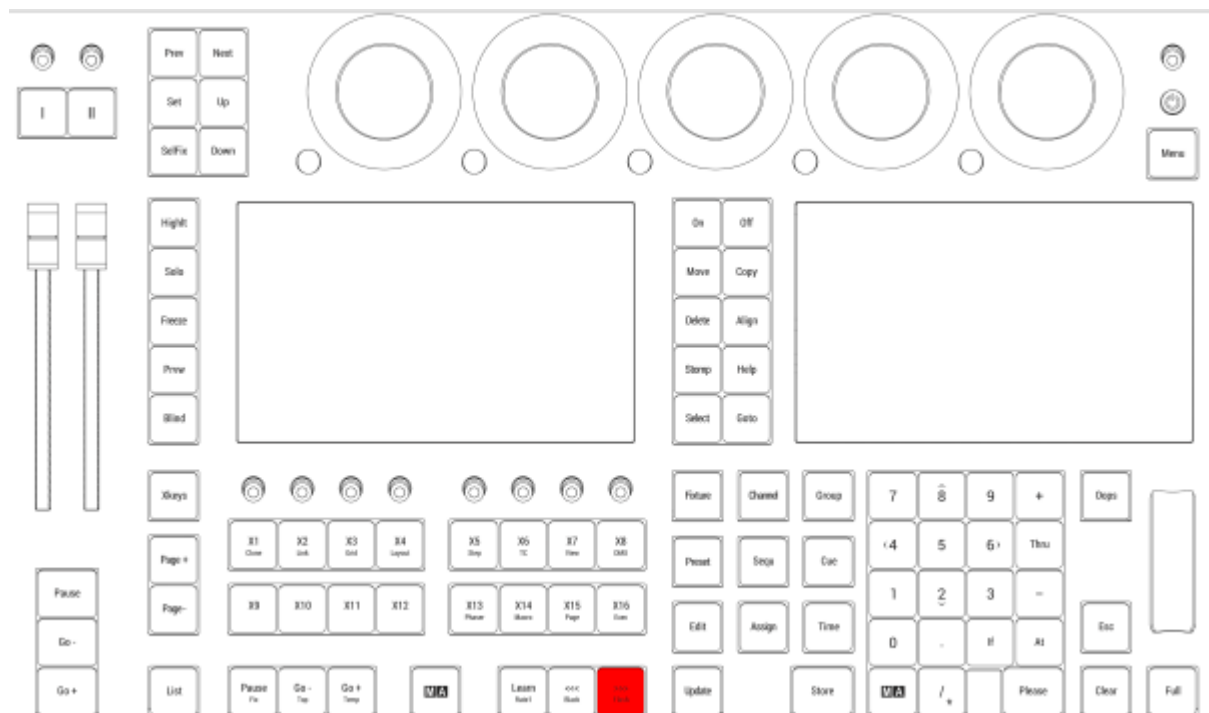
Flash

Pressing and holding **MA** + **>>>** enters the Flash keyword into the command line.

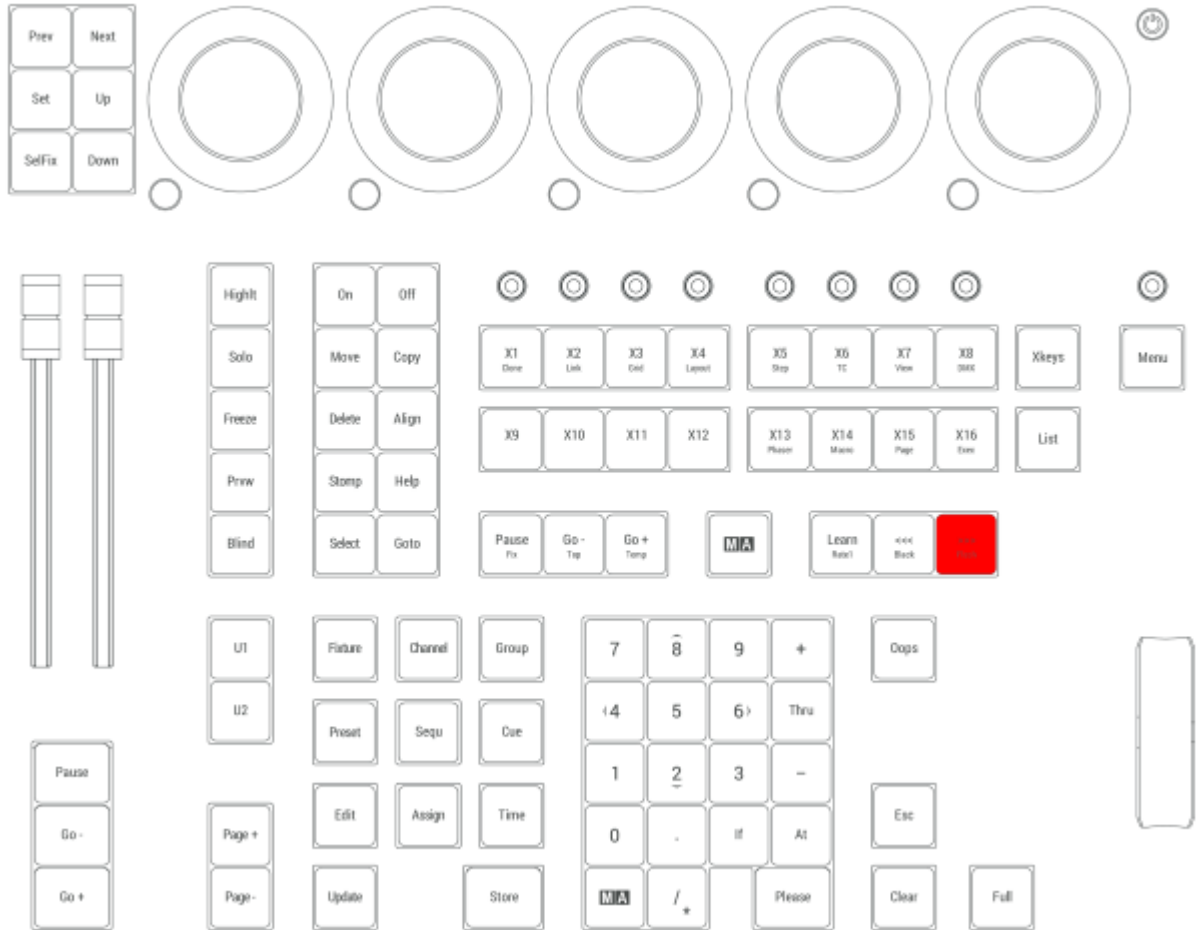


For more information about Flash, see the [Flash keyword](#) topic.

>>> is located in the command section.



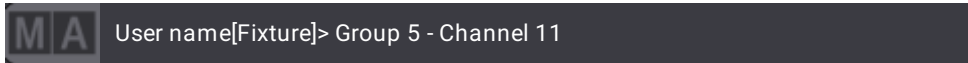
Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.4. - [Minus] Key

Pressing **-** enters a - into the command line.

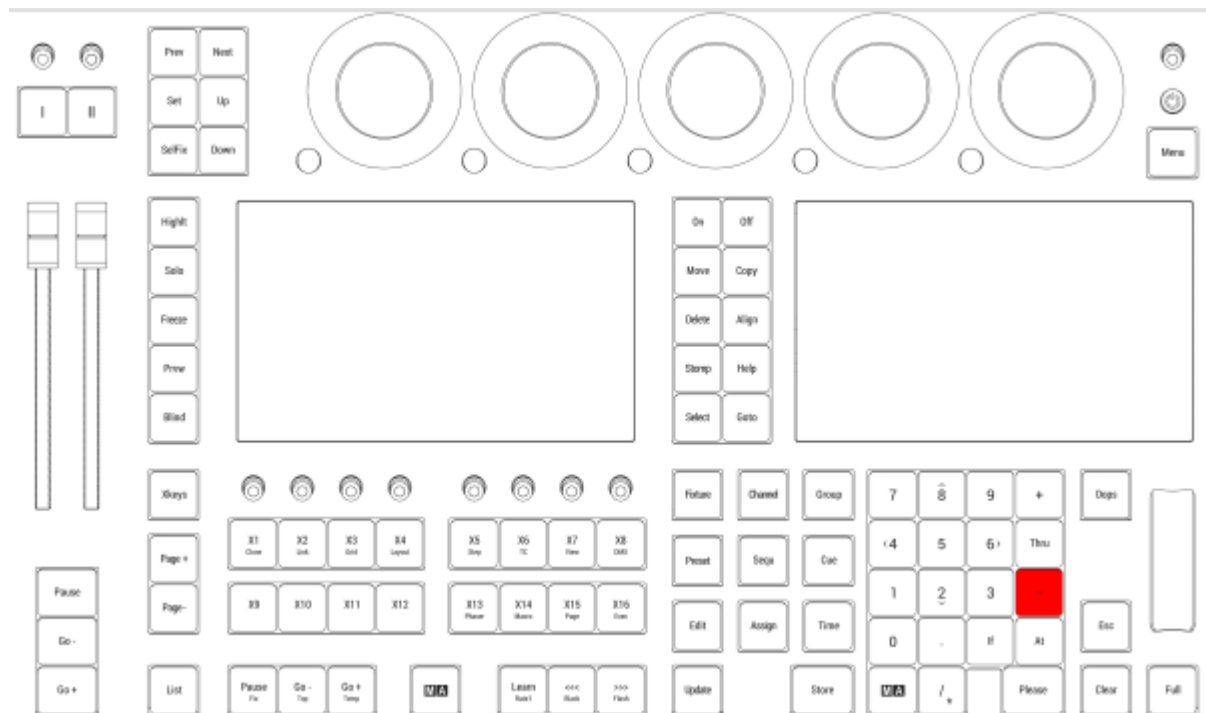


At - 10

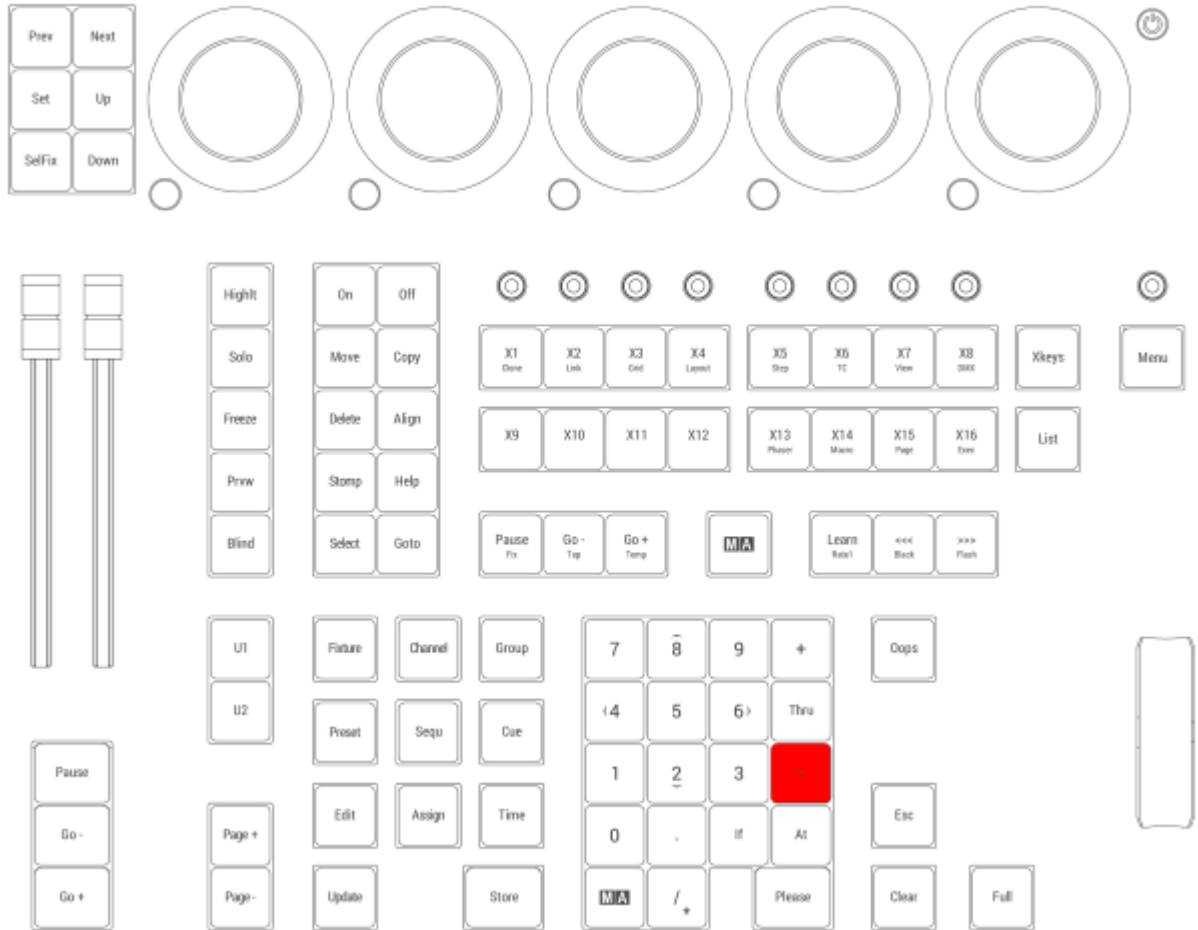
Pressing **-** **-** decrease the dimmer level of the selected fixtures with 10%.



- is located in the numeric keys section.



Location on grandMA3 full-size and grandMA3 light consoles

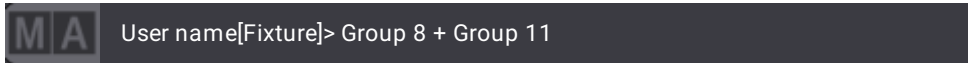


Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.5. + [Plus] Key

Pressing **+** enters the + into the command line.

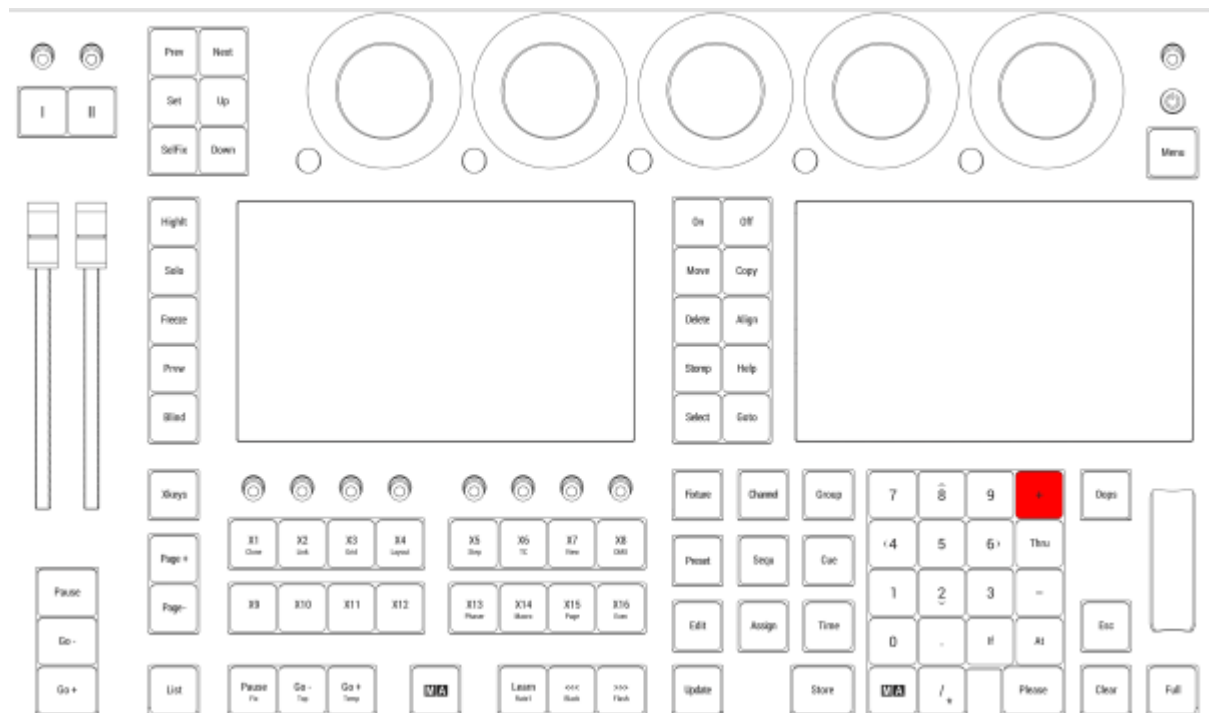


At + 10

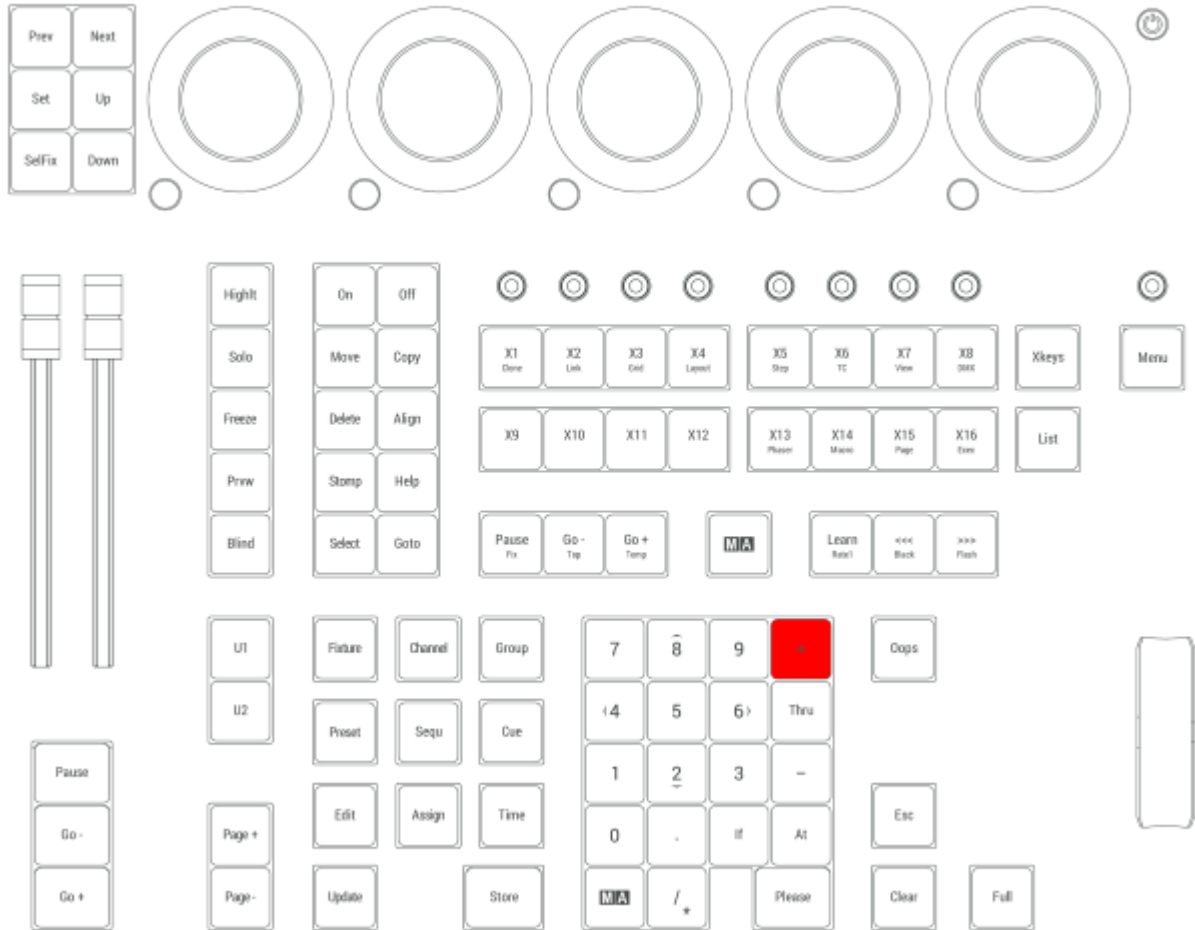
Pressing **+** **+** increase the dimmer level of the selected fixtures with 10%.



+ is located in the numeric keys section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.6. / [Slash] | * [Asterisk/Multiply] Key

Pressing **/** enters the / into the command line.



For more information about /, see the [/\[Slash\] keyword](#) topic.

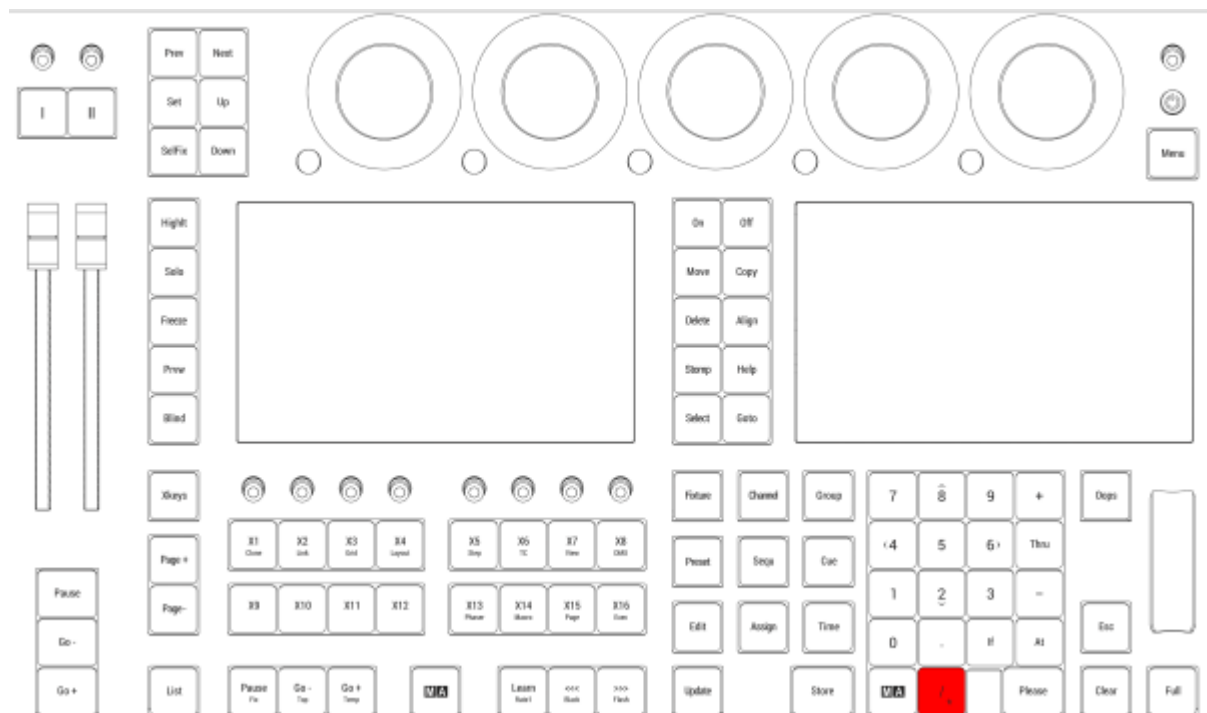
Asterisk

Pressing **MA + /** enters the * keyword into the command line.

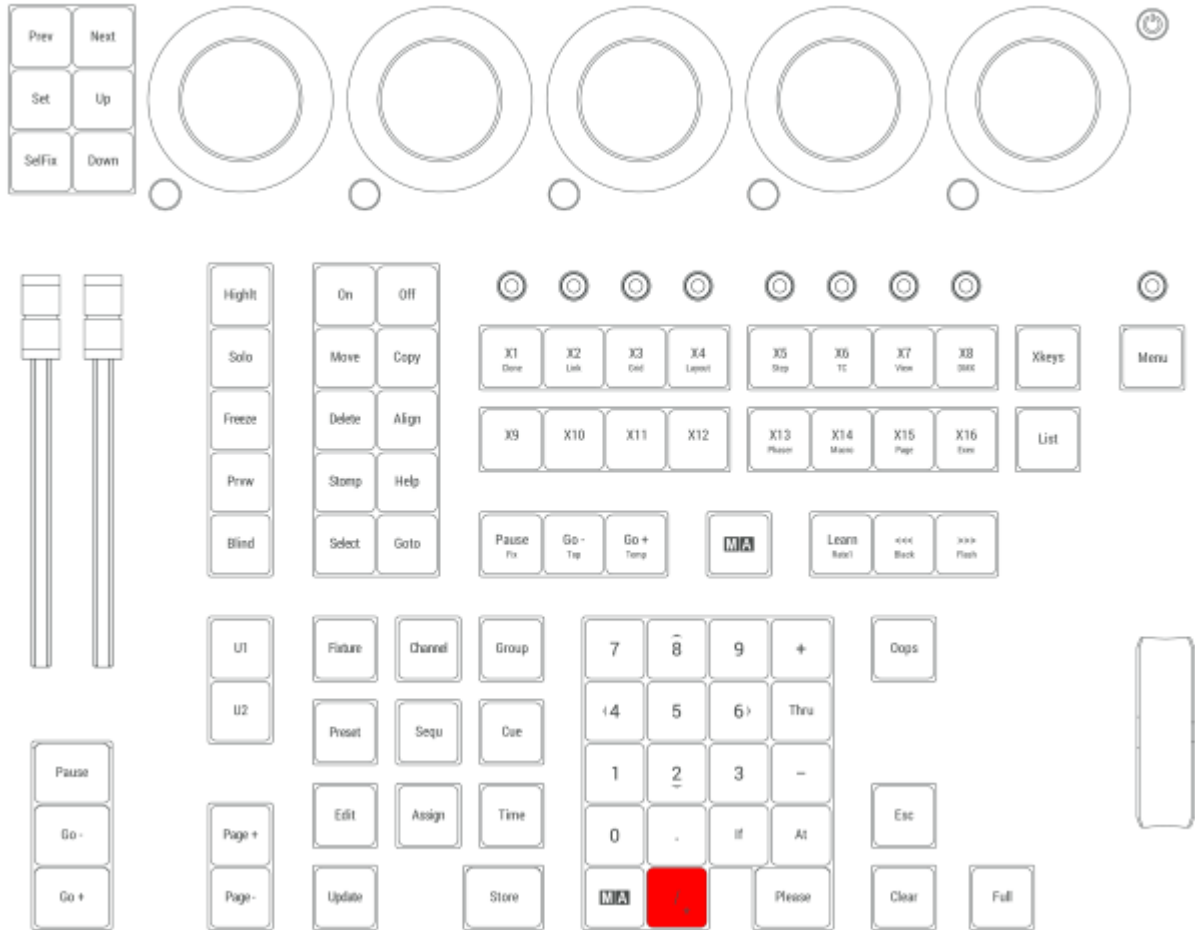


For more information about *, see the [* \[Asterisk\] keyword](#) topic.

/ is located in the numeric keys section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.7. At Key

Pressing **At** enters the At keyword into the command line.



For more information about At, see the [At keyword](#) topic.

Normal

Pressing **At** **At** executes the Normal keyword in the command line, for the selected fixtures.



For more information about Normal, see the [Normal keyword](#) topic.



Integrate

Pressing and holding **MA** + **At** enters the Integrate keyword into the command line.



For more information about Integrate, see the [Integrate keyword](#) topic.

Extract

Pressing and holding **MA** + **At** + **At** enters the Extract keyword into the command line.

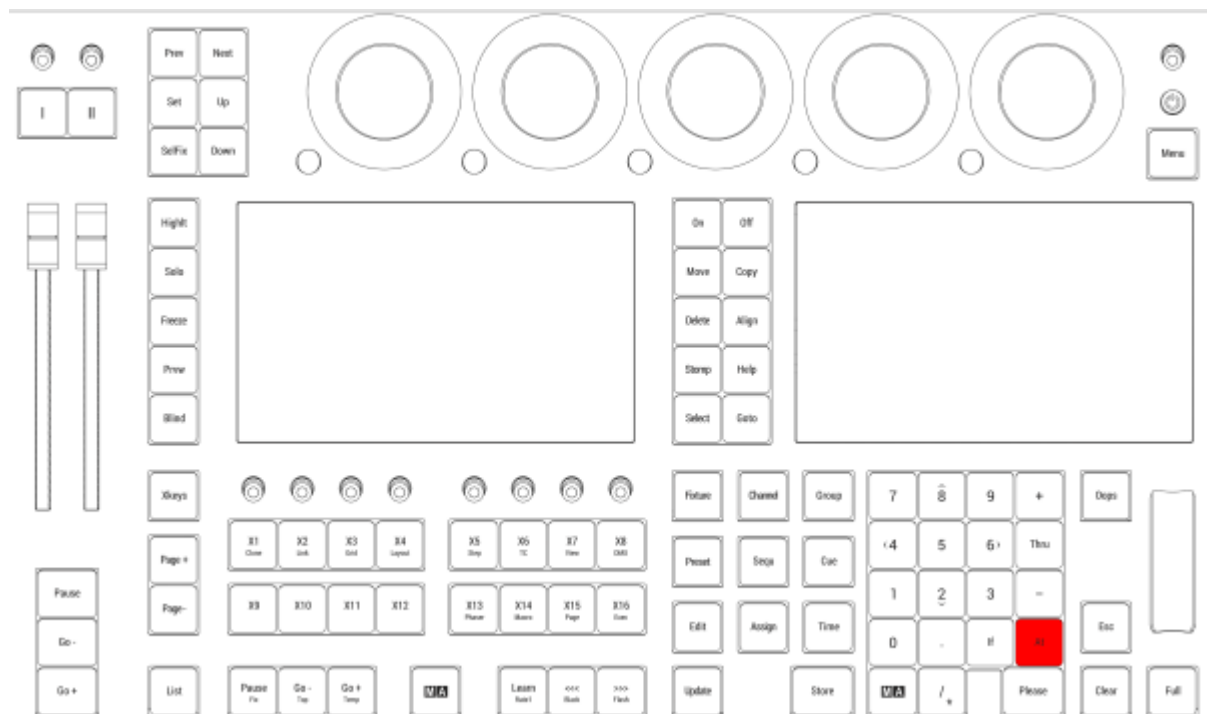


For more information about Extract, see the [Extract keyword](#) topic.

At Filter Menu

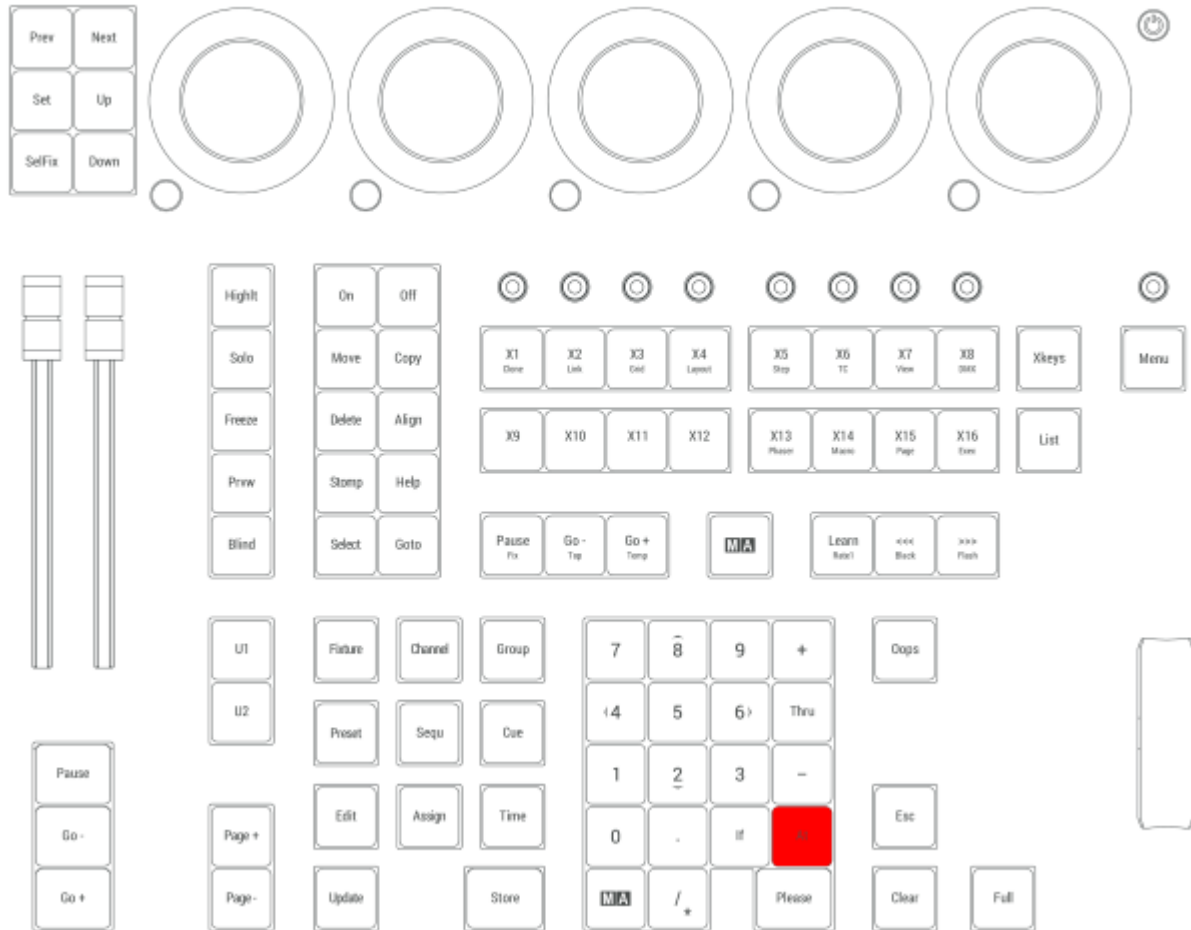
Pressing and holding **At** opens the At Filter menu.

At is located in the numeric keys section.





Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.8. Assign Key

Pressing **Assign** enters the Assign keyword into the command line.



For more information about Assign, see the [Assign keyword](#) topic.

Label

Pressing **Assign** **Assign** enters the Label keyword into the command line.



For more information about Label, see the [Label keyword](#) topic.

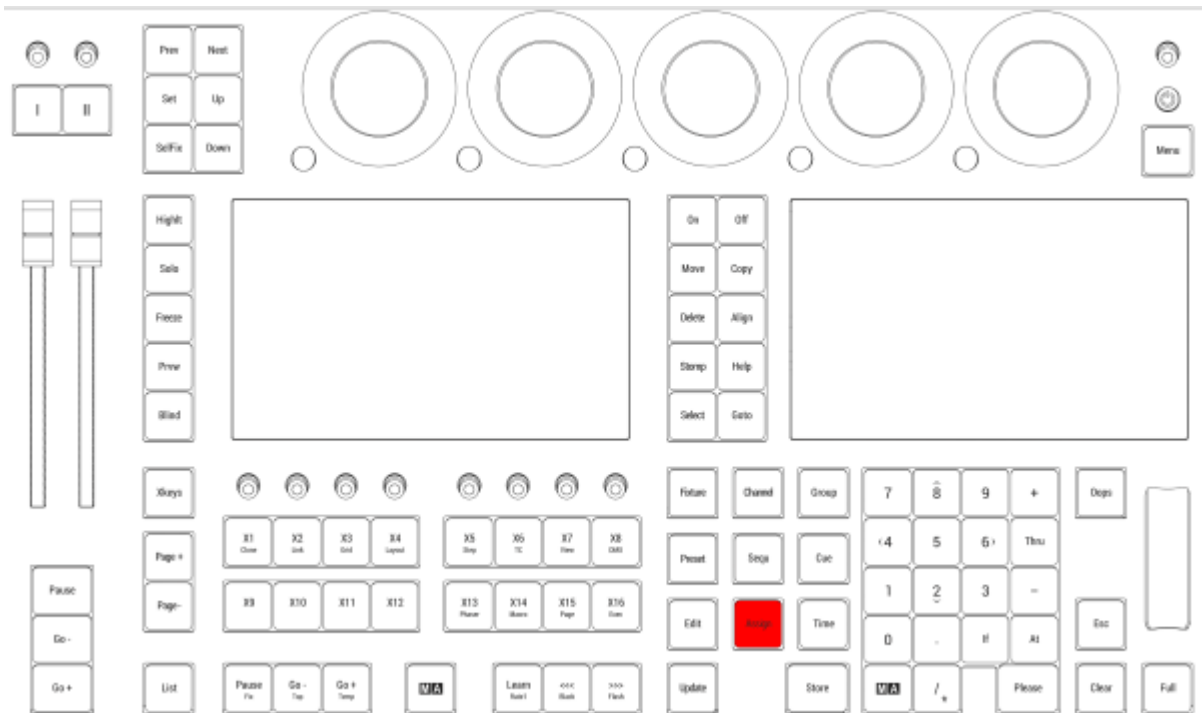
Set

Pressing **MA** + **Assign** enters the Set keyword into the command line.

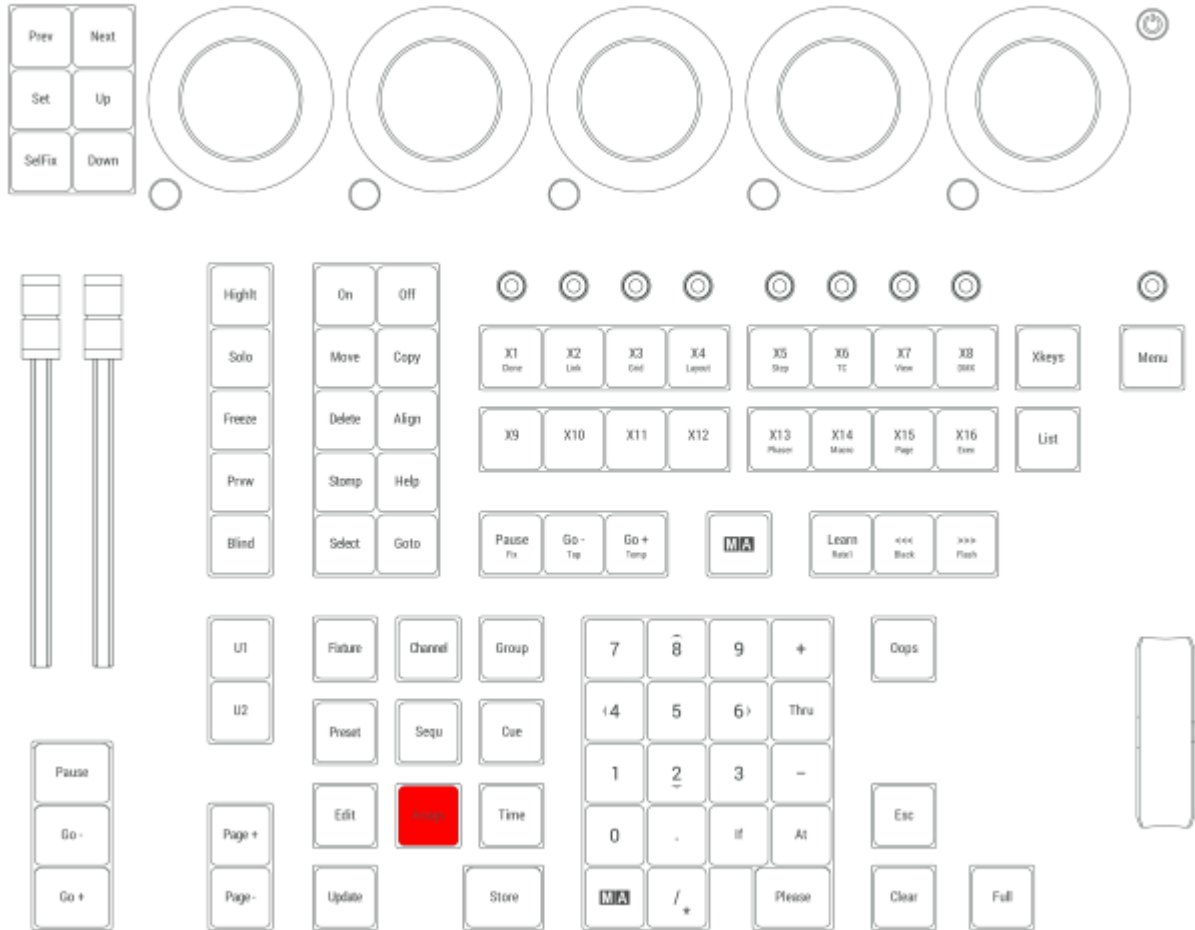


For more information about Set, see the [Set keyword](#) topic.

Assign is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.9. Align Key

Pressing **Align** toggles between the 6 different Align modes. By default, the Align mode is Off.

Pressing **MA** + **Align** toggles between the four different Transition modes. By default, the Transition mode is Linear.

The Align mode is displayed in the default encoder bar.



Default encoder bar

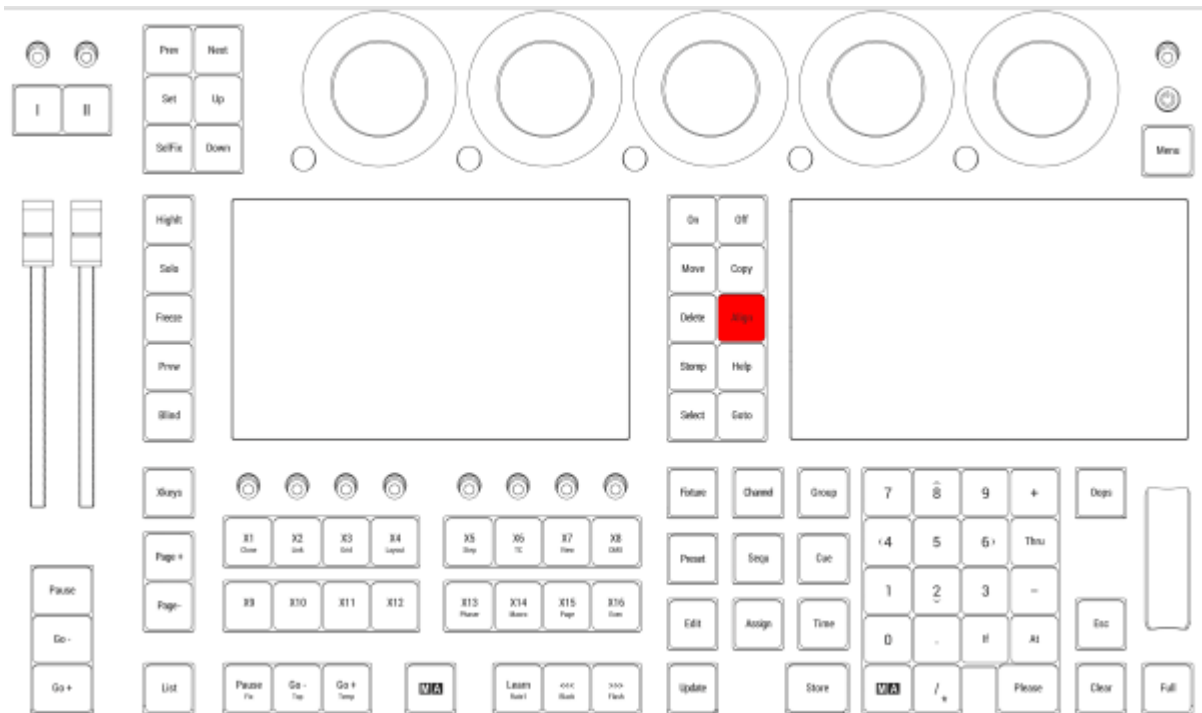
The align mode will be switched off when changing the value of a different attribute and the Transition mode becomes linear.

Pressing and holding **Align** resets the Align mode.

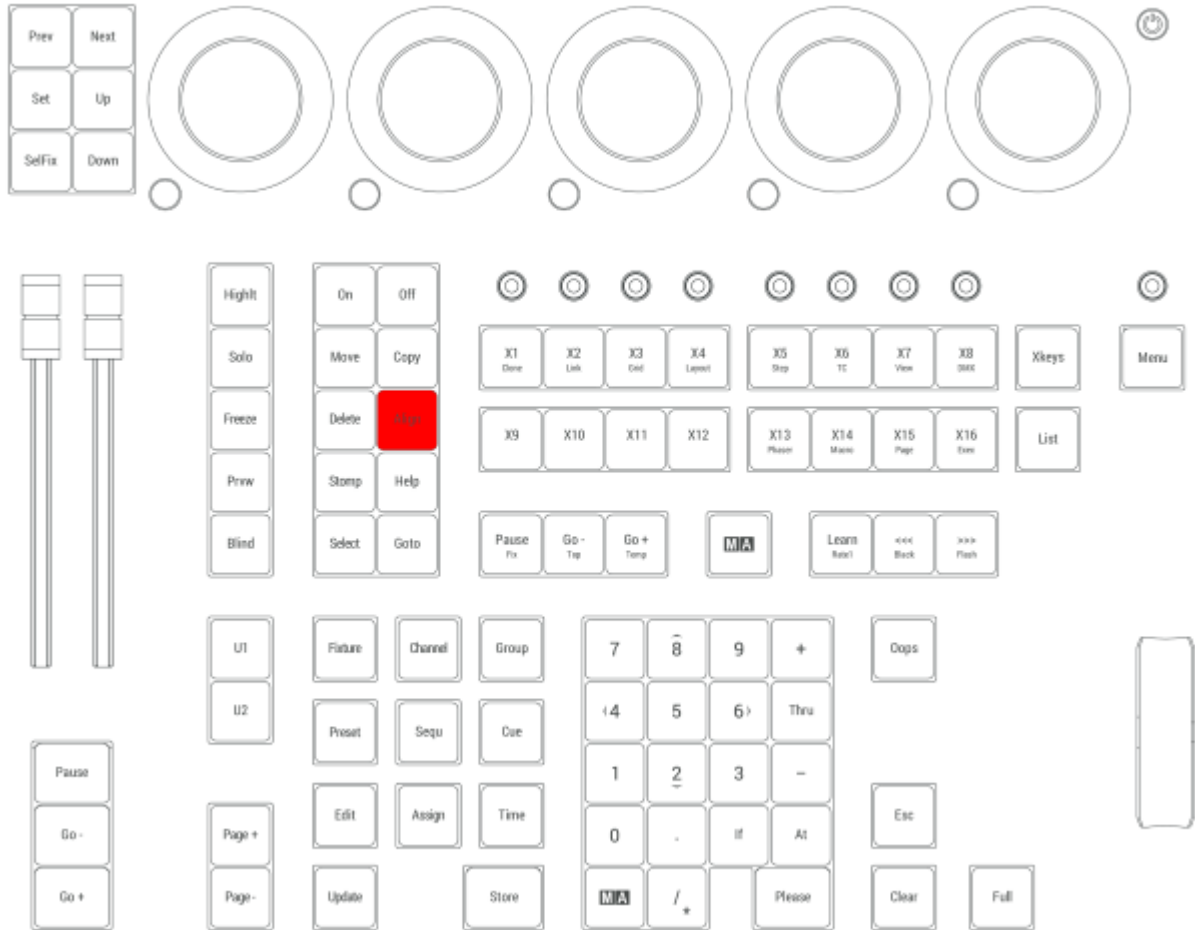
Pressing and holding **MA** + **Align** resets the Transition mode.

To read more about Align, please see the [Align topic](#).

Align is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

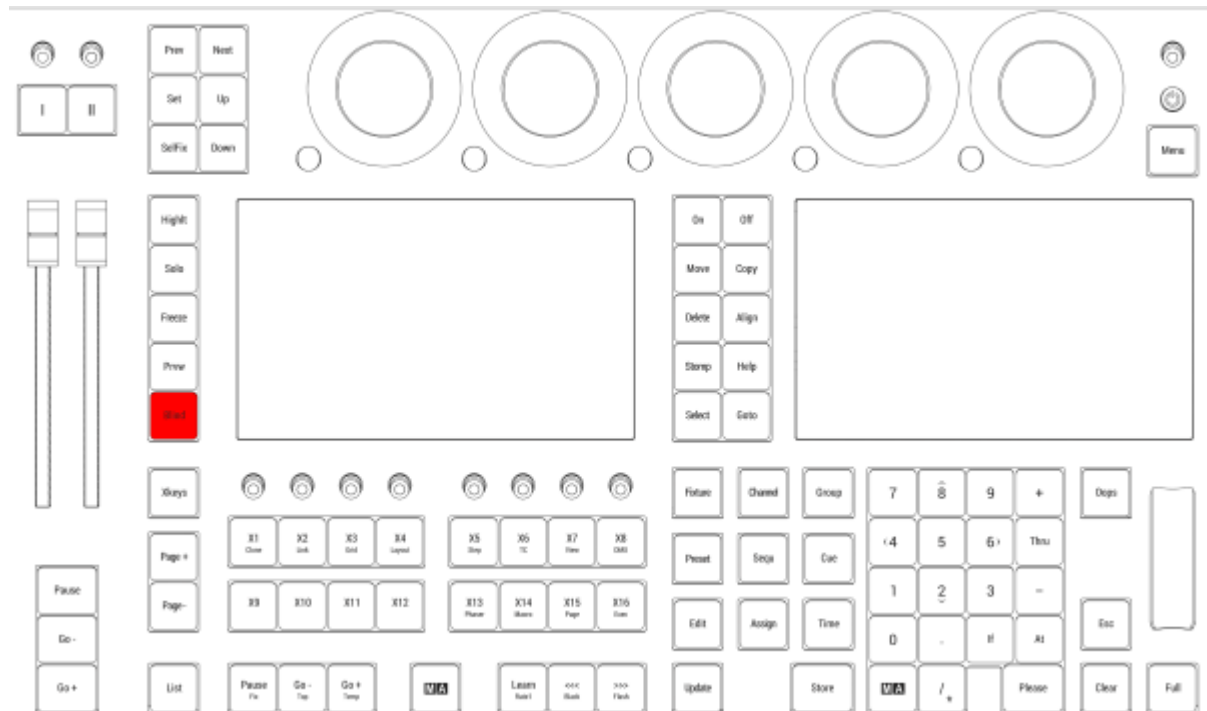
2.15.10. Blind Key

Pressing **Blind** toggles the Blind function.

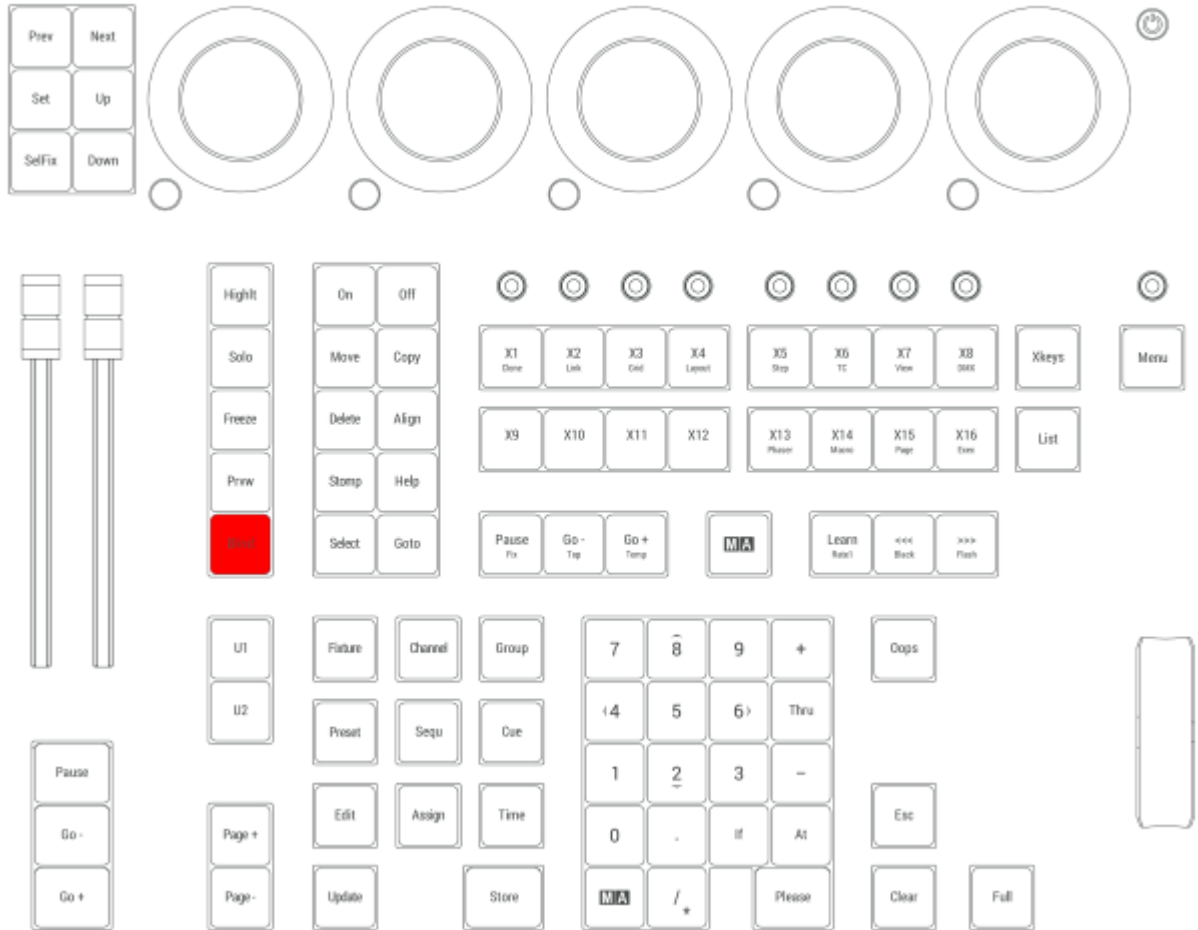
OK:Blind

For more information about Blind, see the **Blind keyword** topic.

Blind is located in the Command Section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.11. Clear Key

If there is something in the programmer, **Clear** behaves like this:

- Pressing **Clear** executes the ClearSelection keyword.
- Pressing **Clear** **Clear** executes the ClearActive keyword.
- Pressing **Clear** **Clear** **Clear** executes the ClearAll keyword.

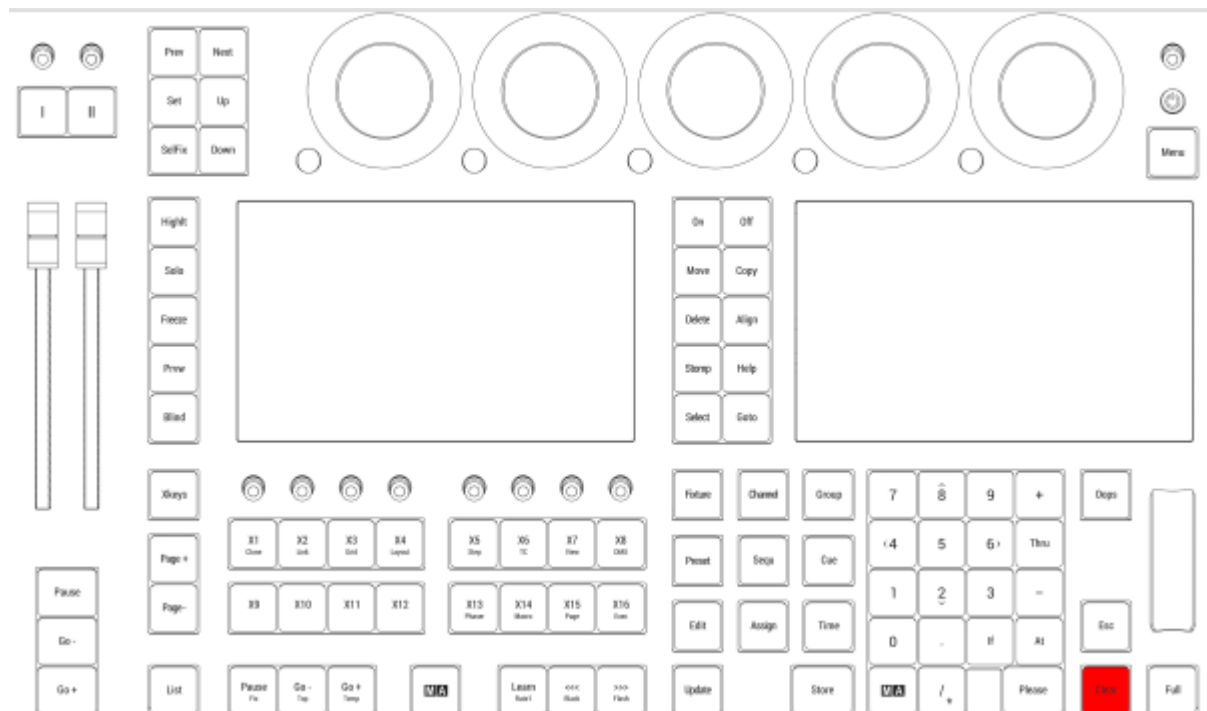
If there is nothing in the programmer, pressing **Clear** deletes unexecuted commands in the command line and executes the ClearActive keyword

Pressing and holding **Clear** executes the ClearAll keyword.

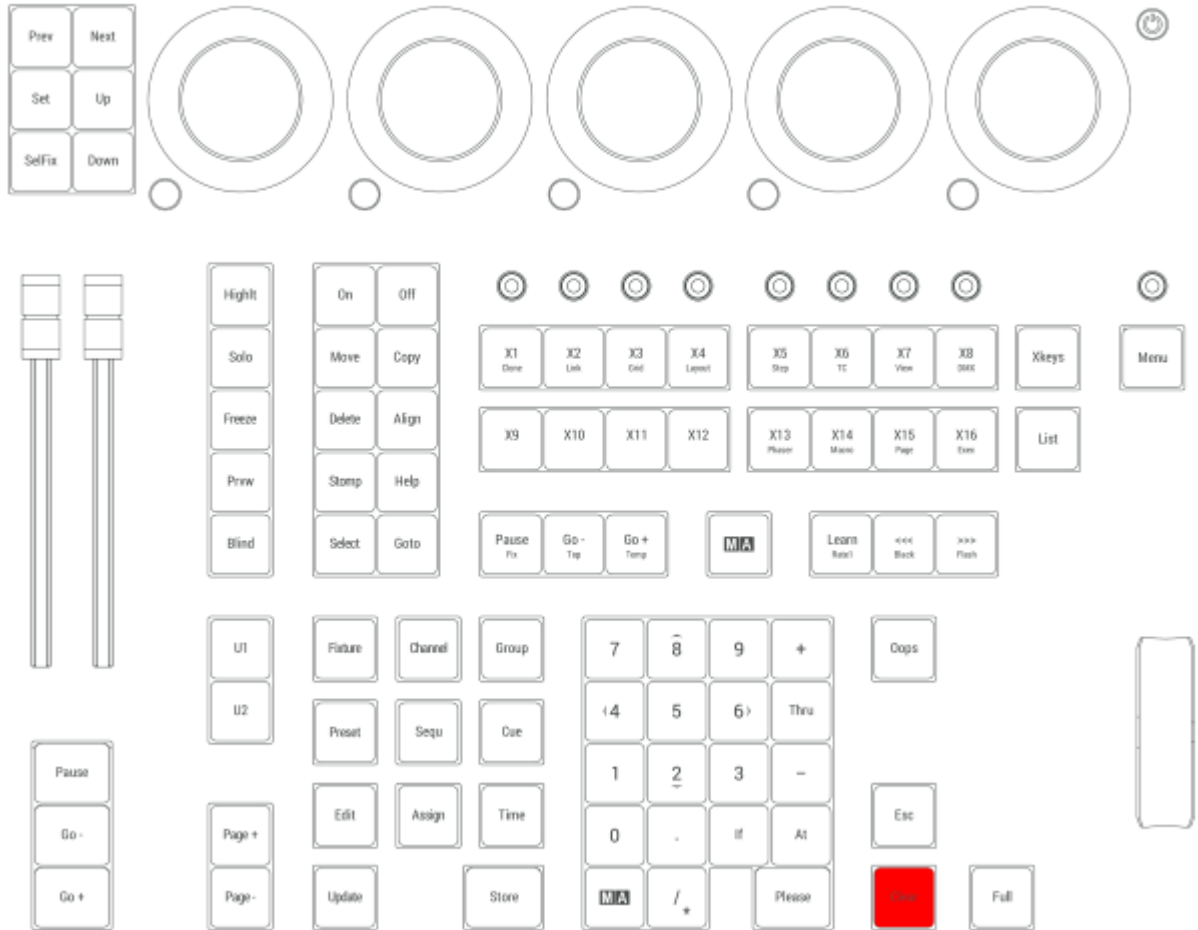
Pressing Clear, no matter how many times always gives a Clear in the Command Line Feedback

For more information about Clear, see the [Clear keyword](#) topic, the [ClearSelection keyword](#), the [ClearActive keyword](#) or the [ClearAll keyword](#) topic.

Clear is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.12. Channel Key

Pressing **Channel** enters the Channel keyword in the command line.

```
MA User name[Fixture]> Channel
```

For more information, see the [Channel keyword](#) topic.

Universal

Pressing **Channel** **Channel** enters the Universal keyword in the command line.

```
MA User name[Fixture]> Universal
```

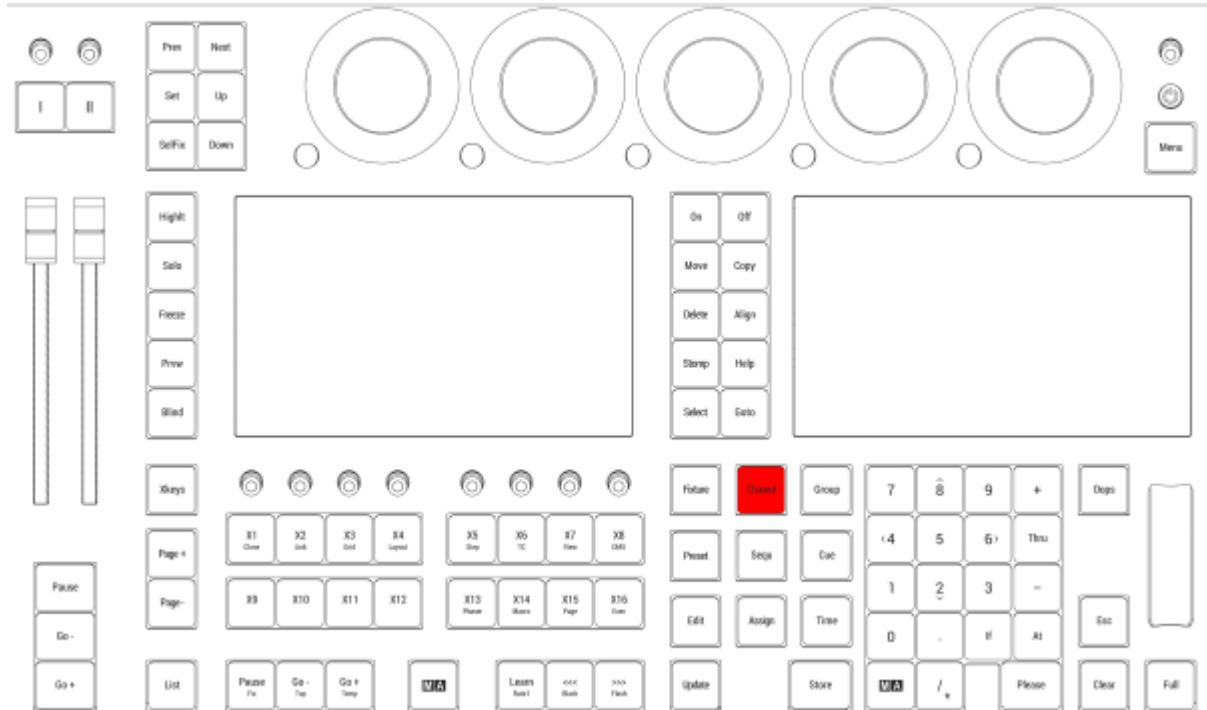
For more information, see the [Universal keyword](#) topic.

IDTypes

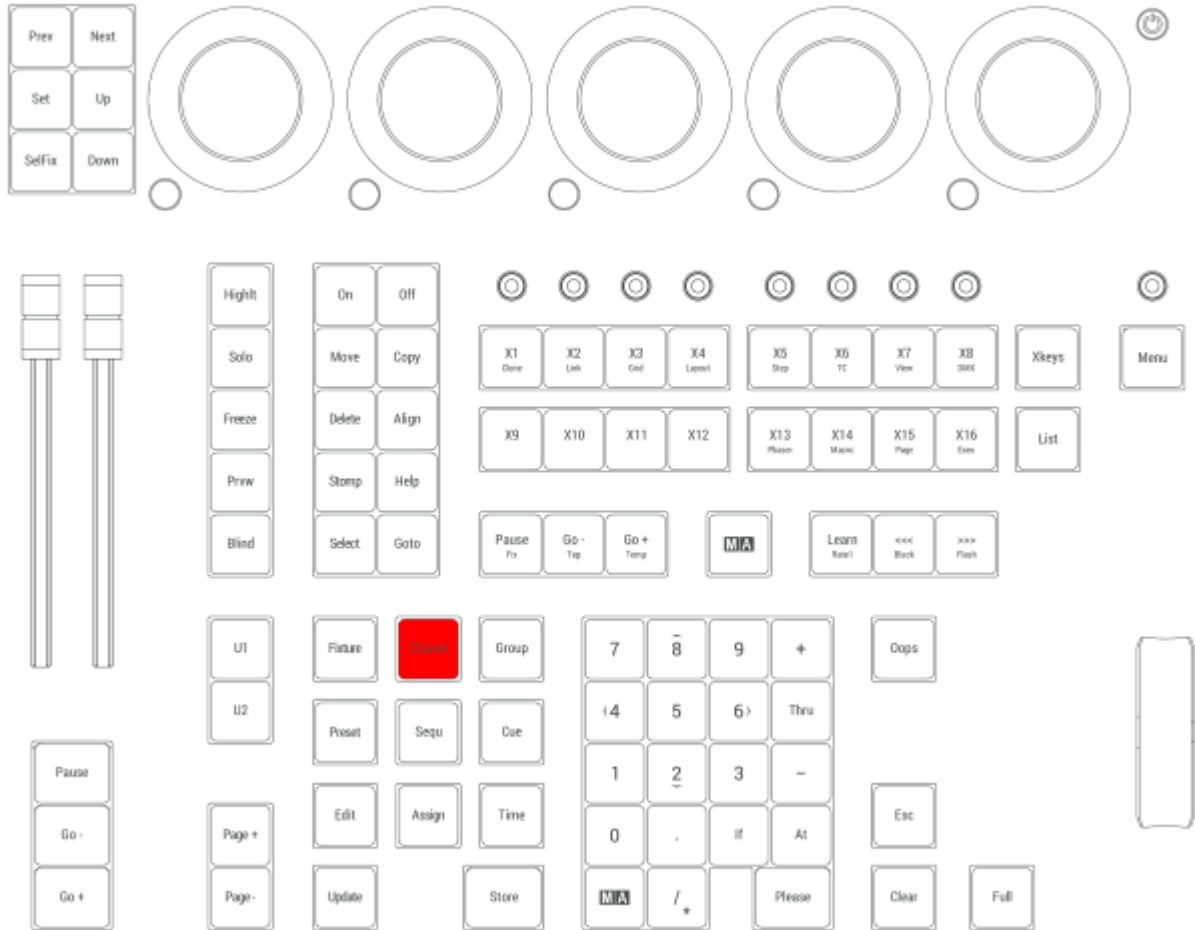
Pressing **Channel** repeatedly switches through Channel, Universal, and used IDTypes. IDTypes that have no patched fixtures assigned are not appearing in the command line.

For more information, see the [Add fixtures to the show](#) topic.

Channel is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.13. Copy Key

Pressing **Copy** enters the copy keyword into the command line.

A dark grey terminal window with a grey header bar containing the 'MA' logo. The text 'User name[Fixture]> Copy' is displayed in white on the terminal line.

For more information about Copy, see the [Copy keyword](#) topic.

Paste

Pressing **Copy** **Copy** enters the Paste keyword into the command line.

A dark grey terminal window with a grey header bar containing the 'MA' logo. The text 'User name[Fixture]> Paste' is displayed in white on the terminal line.

For more information about Paste, see the [Paste keyword](#) topic.

Insert

Pressing **Copy** **Copy** **Copy** enters the Insert keyword into the command line.

A dark grey terminal window with a grey header bar containing the 'MA' logo. The text 'User name[Fixture]> Insert' is displayed in white on the terminal line.

For more information about Insert, see the [Insert keyword](#) topic.

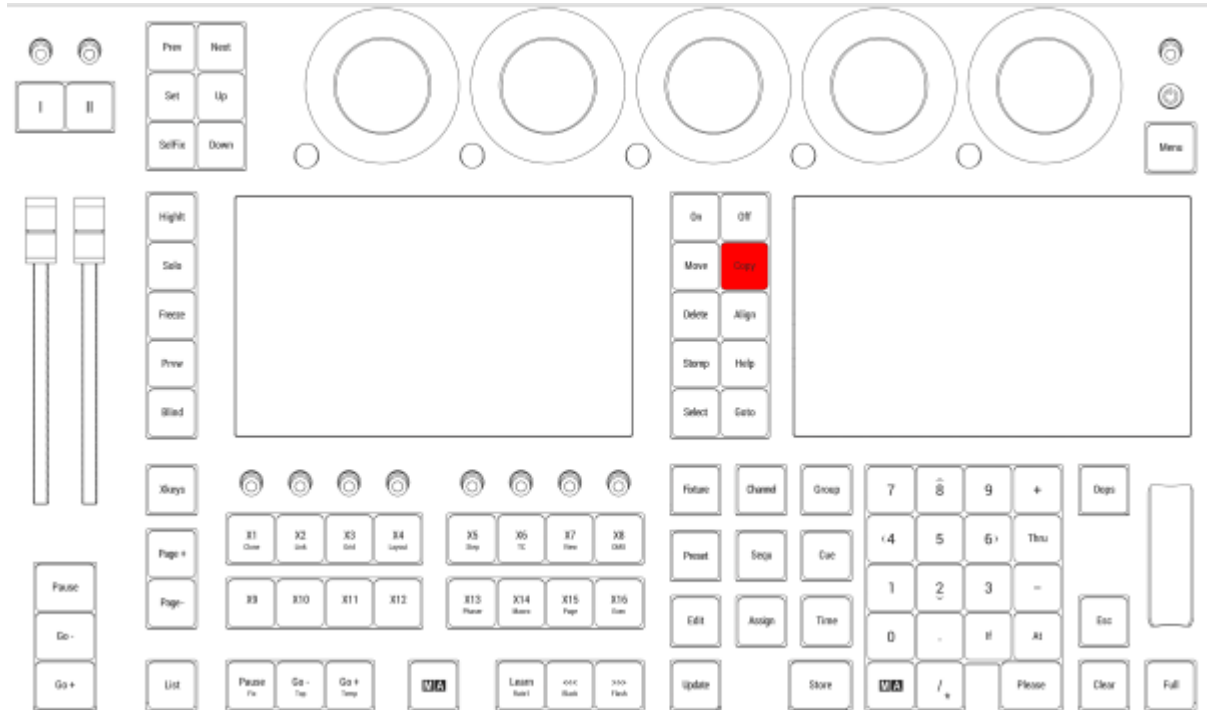
Cut

Pressing **MA** **Copy** enters the Cut keyword into the command line.

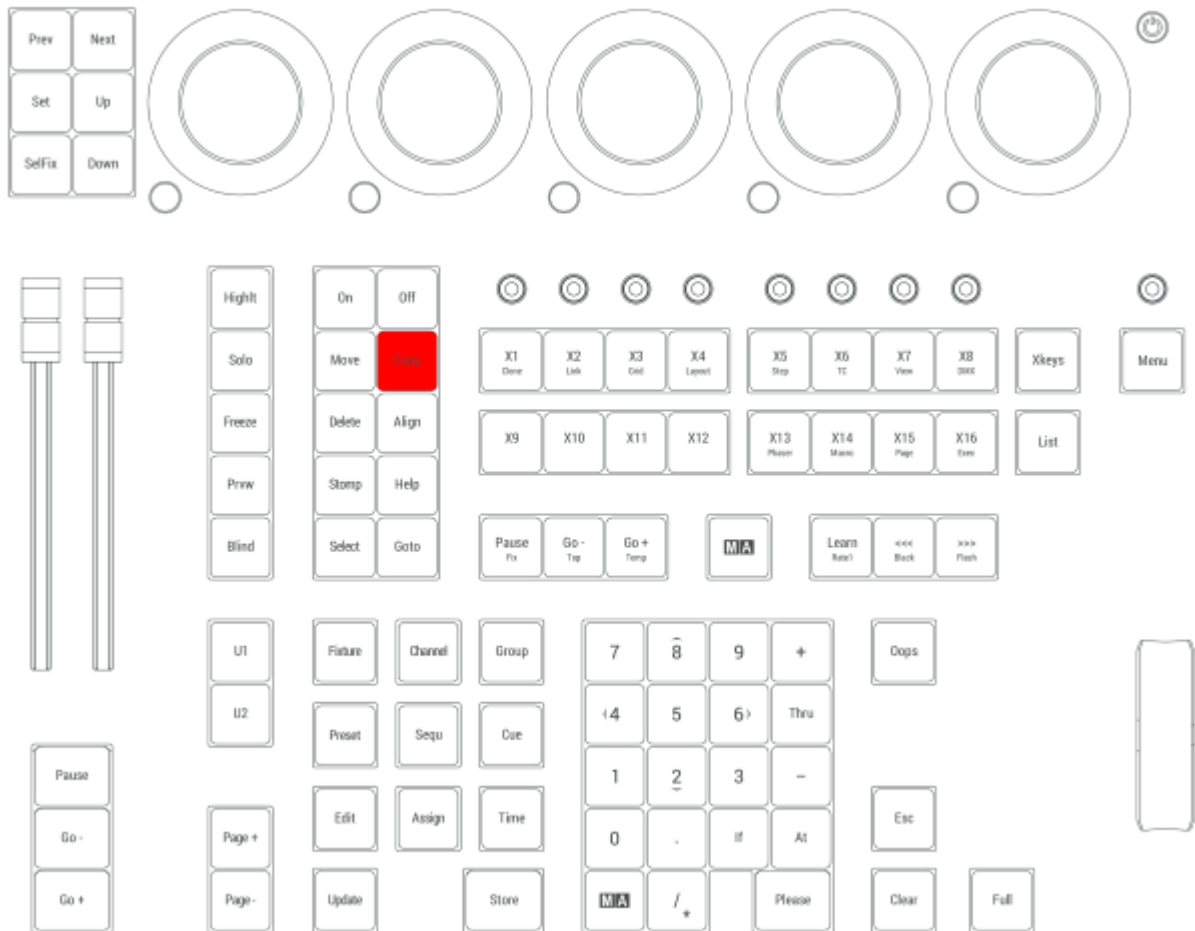
A dark grey terminal window with a grey header bar containing the 'MA' logo. The text 'User name[Fixture]> Cut' is displayed in white on the terminal line.

For more information about Cut, see the [Cut keyword](#) topic.

Copy is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.14. Ctrl

Multi-Select

To select multiple cells in a sheet, hold **Ctrl** and tap or click each desired cell.

Clean Start

To perform a clean start, hold **Ctrl** when the mode selection pop-up appears during the boot or reboot process and tap the desired mode. For more information about clean start, see the [Clean Start](#) topic.

Ctrl is located in two places on the bottom row of the alpha-numeric keyboard.



Location on grandMA3 full-size and grandMA3 light console



2.15.15. Cue Key

Pressing **Cue** enters the Cue keyword into the command line.

```
MA User name[Fixture]> Cue
```

For more information about Cue, see the [Cue keyword](#) topic.

Part

Pressing **Cue** **Cue** enters the Part keyword into the command line.

```
MA User name[Fixture]> Part
```

For more information about Part, see the [Part keyword](#) topic.

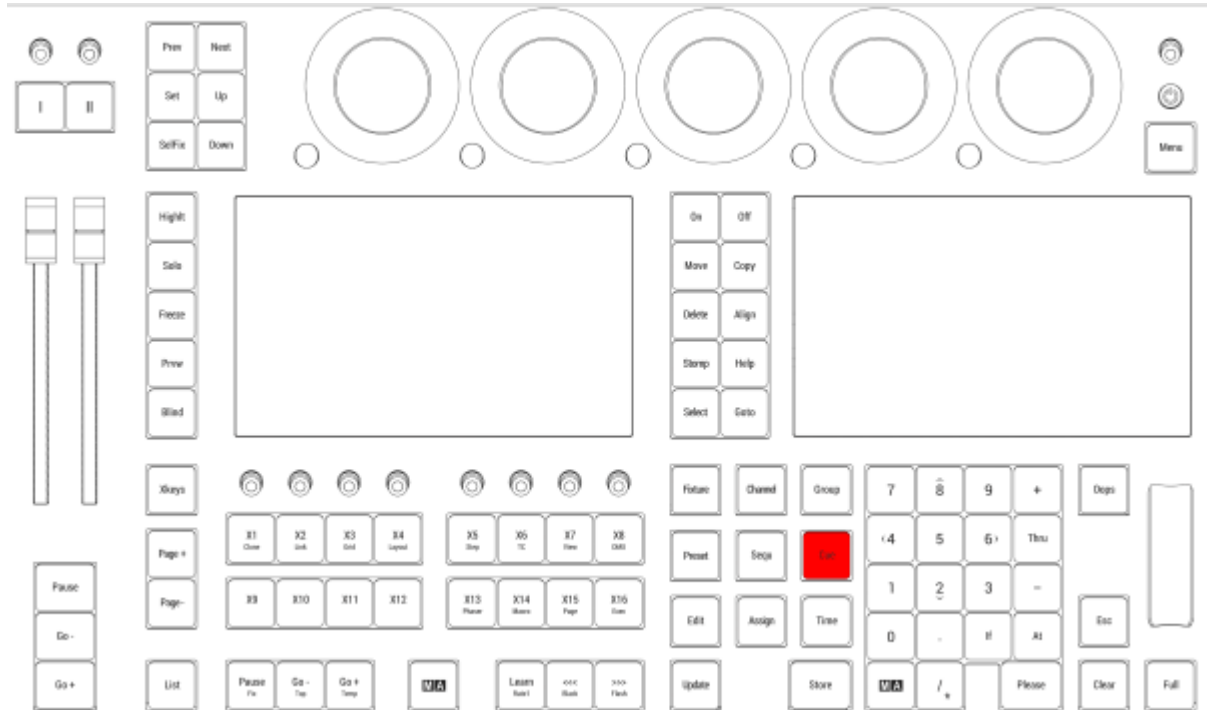
Programmer

Pressing **MA** + **Cue** enters the Programmer keyword into the command line.

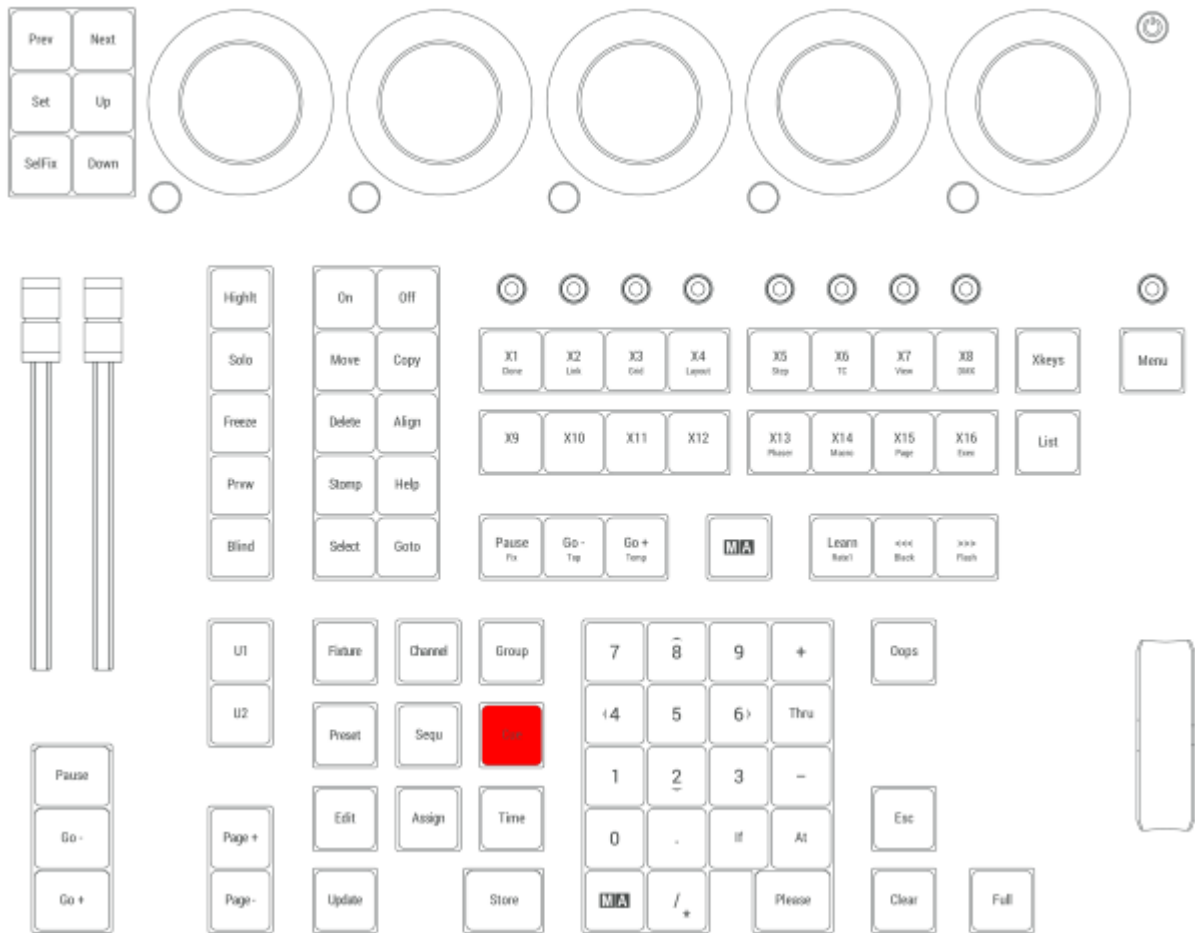
```
MA User name[Fixture]> Programmer
```

For more information about Programmer, see the [Programmer keyword](#) topic.

Cue is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles

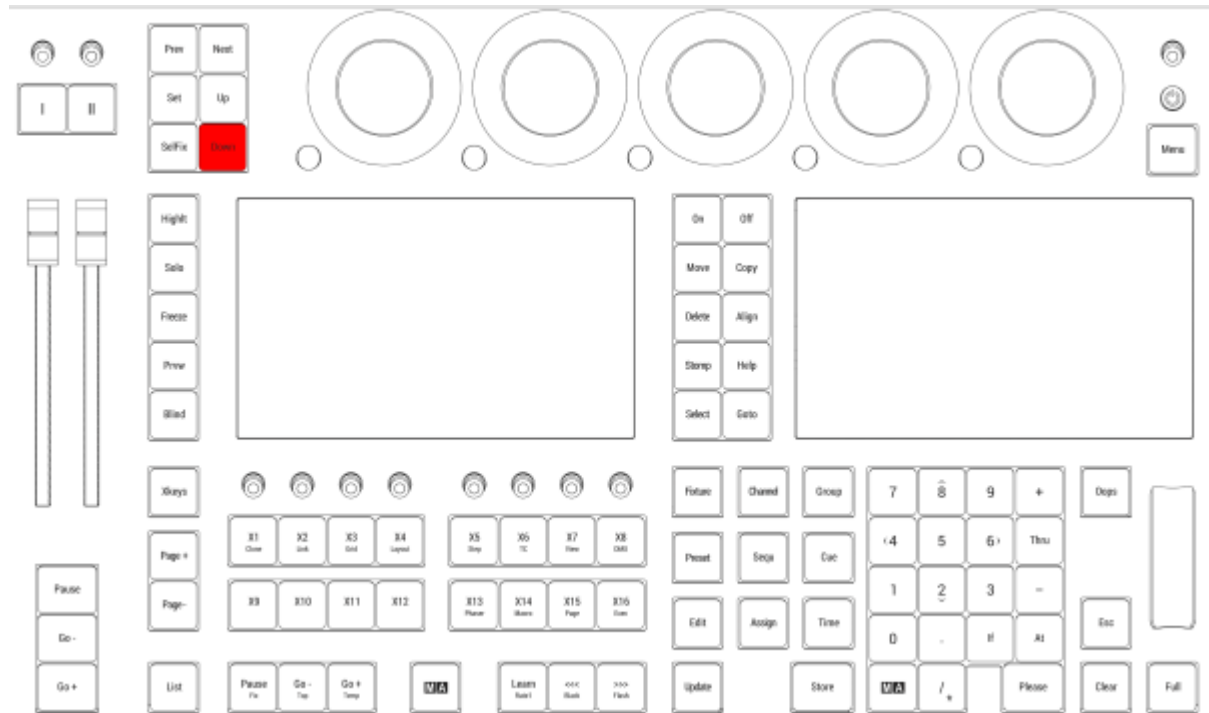


Location on grandMA3 compact consoles and grandMA3 onPC command wing

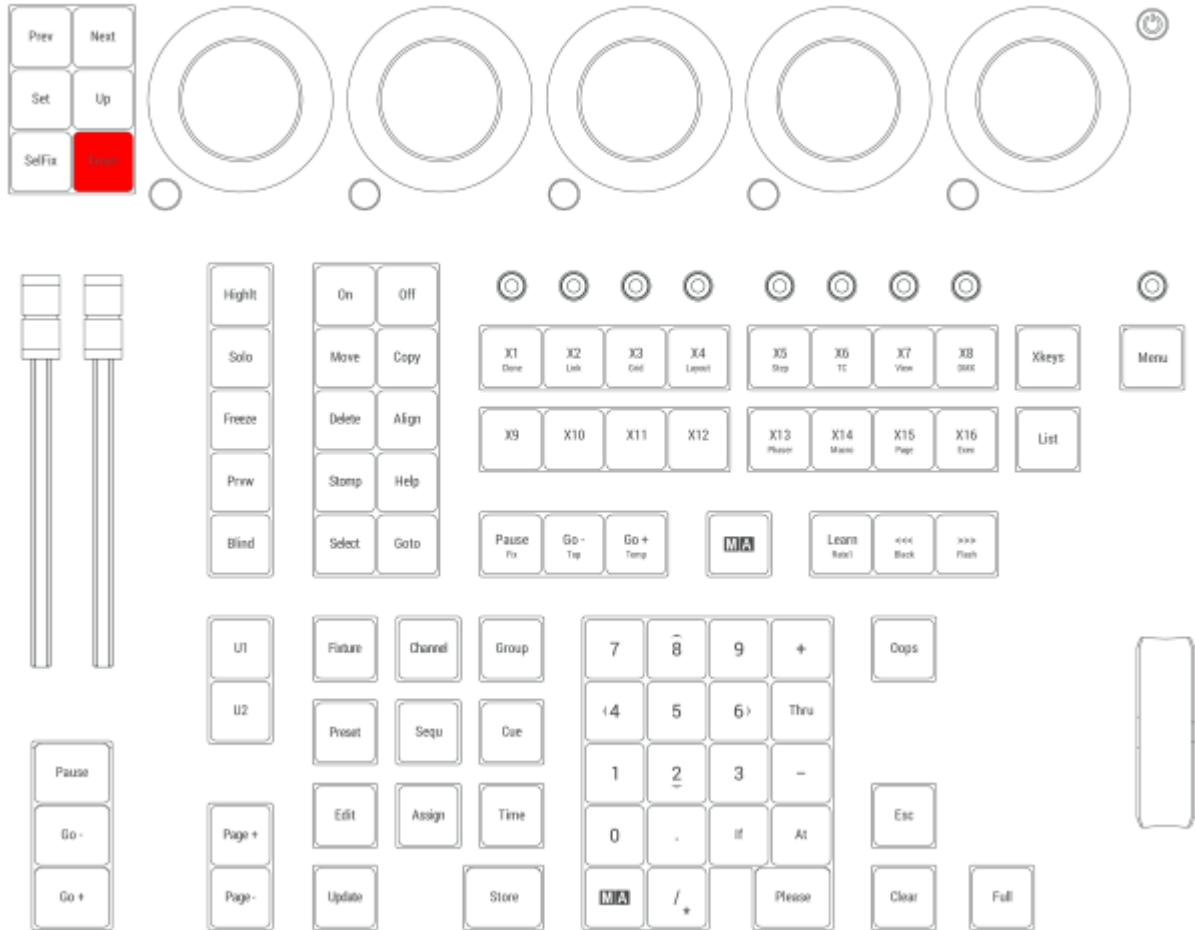
2.15.16. Down Key

Pressing **Down** goes one level down in the fixture structure.

Down is located in the command section on the left side of the five dual encoders.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.17. Delete Key

Pressing **Delete** enters the Delete keyword into the command line.

```
MA User name[Fixture]> Delete
```

For more information about Delete, see the [Delete keyword](#) topic.

Remove

Pressing **Delete** **Delete** enters the Remove keyword into the command line.

```
MA User name[Fixture]> Remove Selection
```

For more information about Remove, see the [Remove keyword](#) topic.

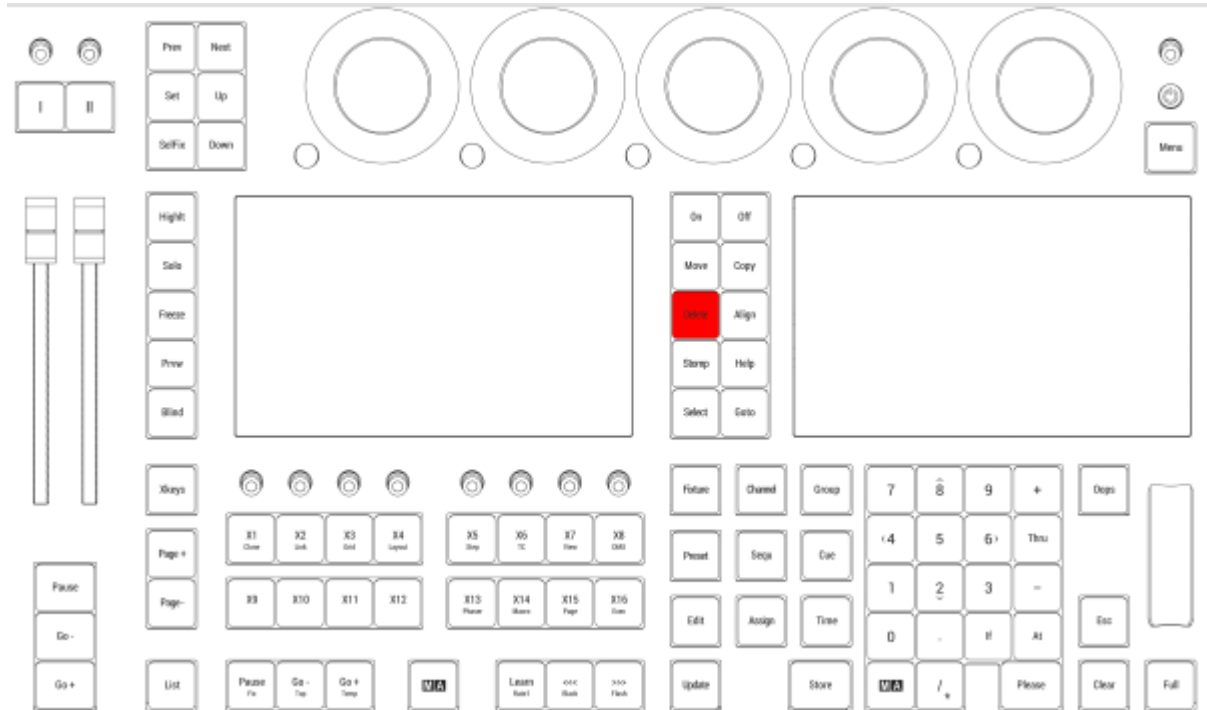
Release

Pressing **Delete** **Delete** **Delete** enters the Release keyword into the command line.

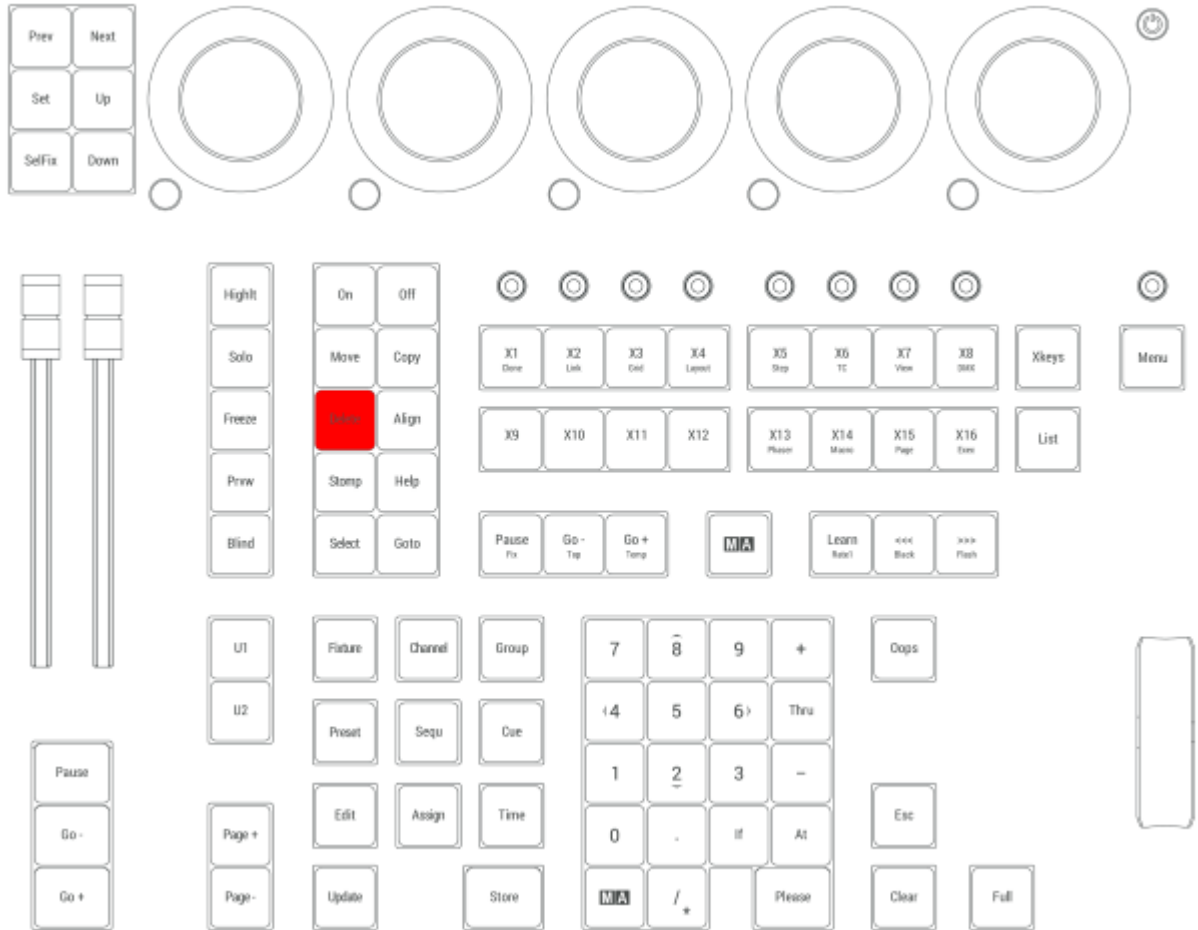
```
MA User name[Fixture]> Release Fixture 37
```

For more information about Release, see the [Release keyword](#) topic.

Delete is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles

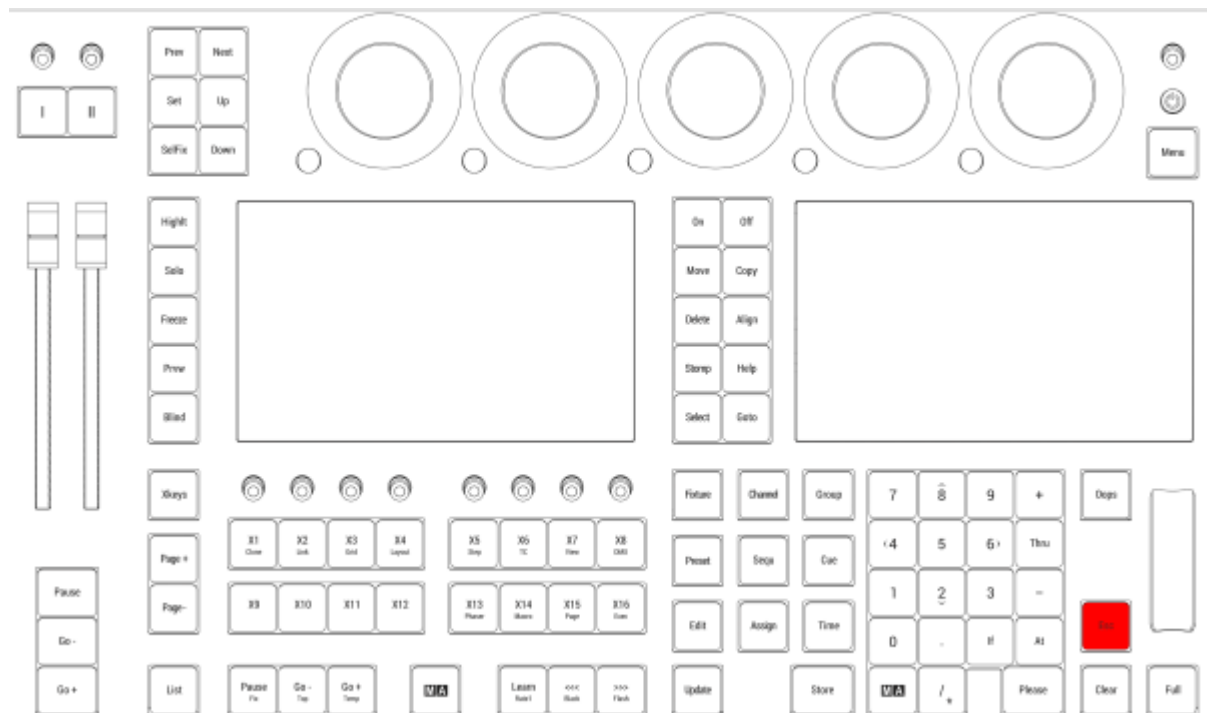


Location on grandMA3 compact consoles and grandMA3 onPC command wing

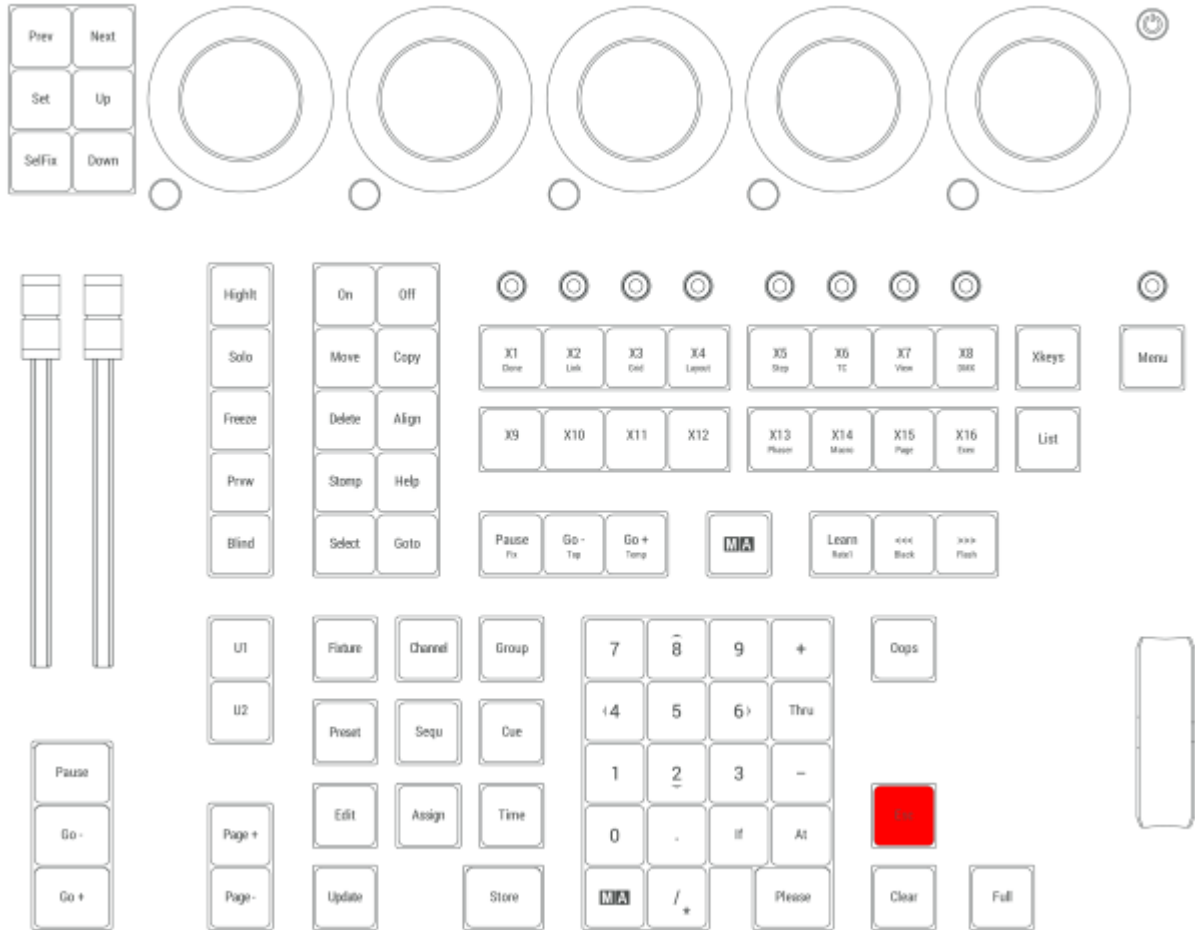
2.15.18. Esc Key

Pressing **Esc** deletes unexecuted commands in the command line and closes pop-ups.

Esc is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.19. Edit Key

Pressing **Edit** enters the Edit keyword into the command line.



For more information about Edit, see the [Edit keyword](#) topic.

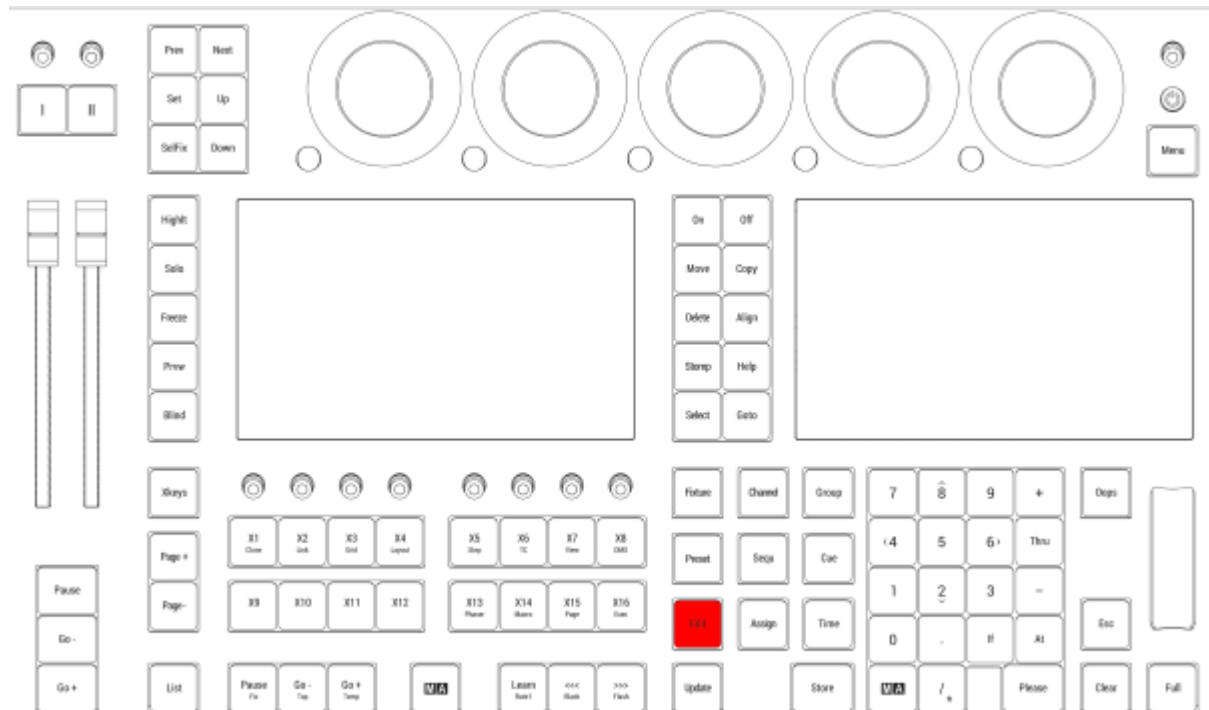
EditSetting

Pressing **Edit** **Edit** enters the EditSetting keyword into the command line.

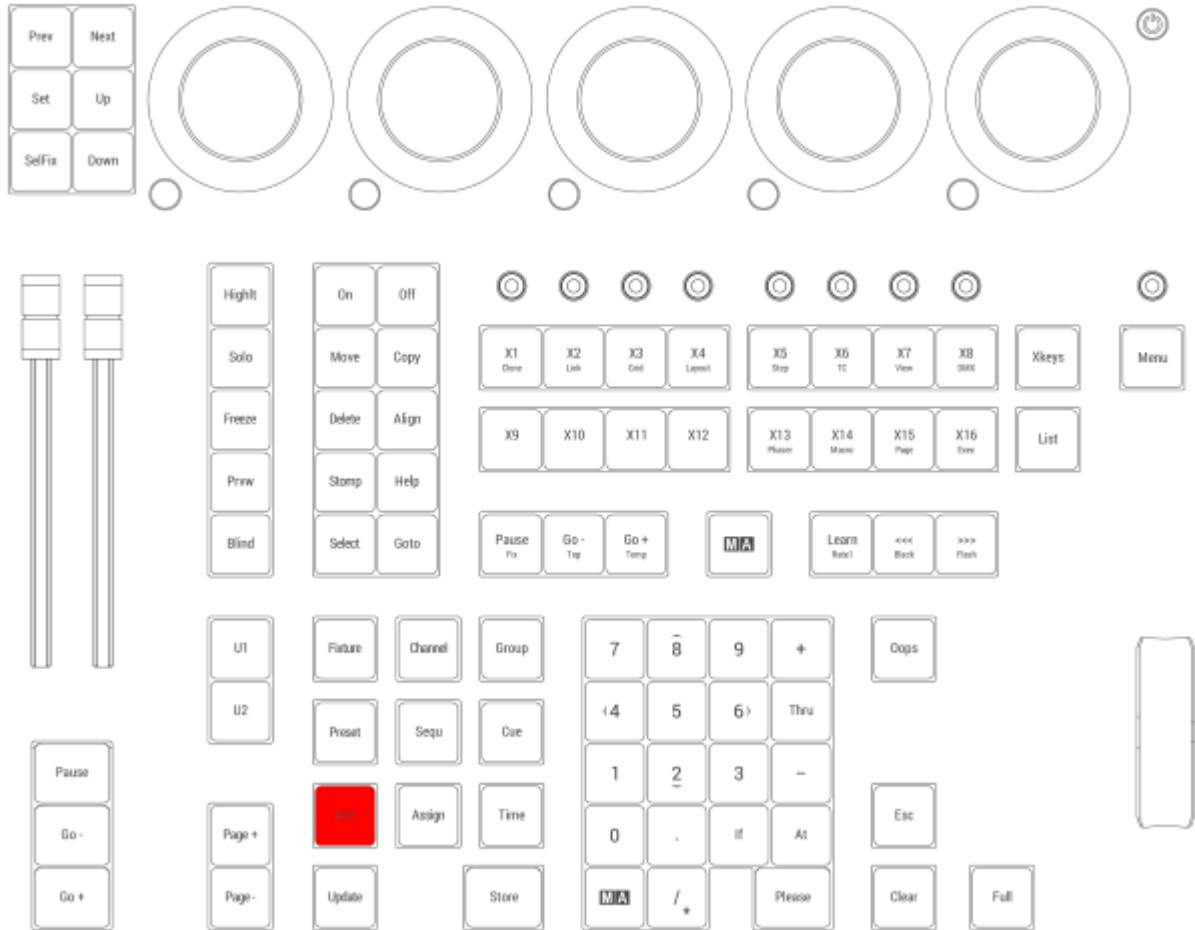


For more information about EditSetting, see the [EditSetting keyword](#) topic.

Edit is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

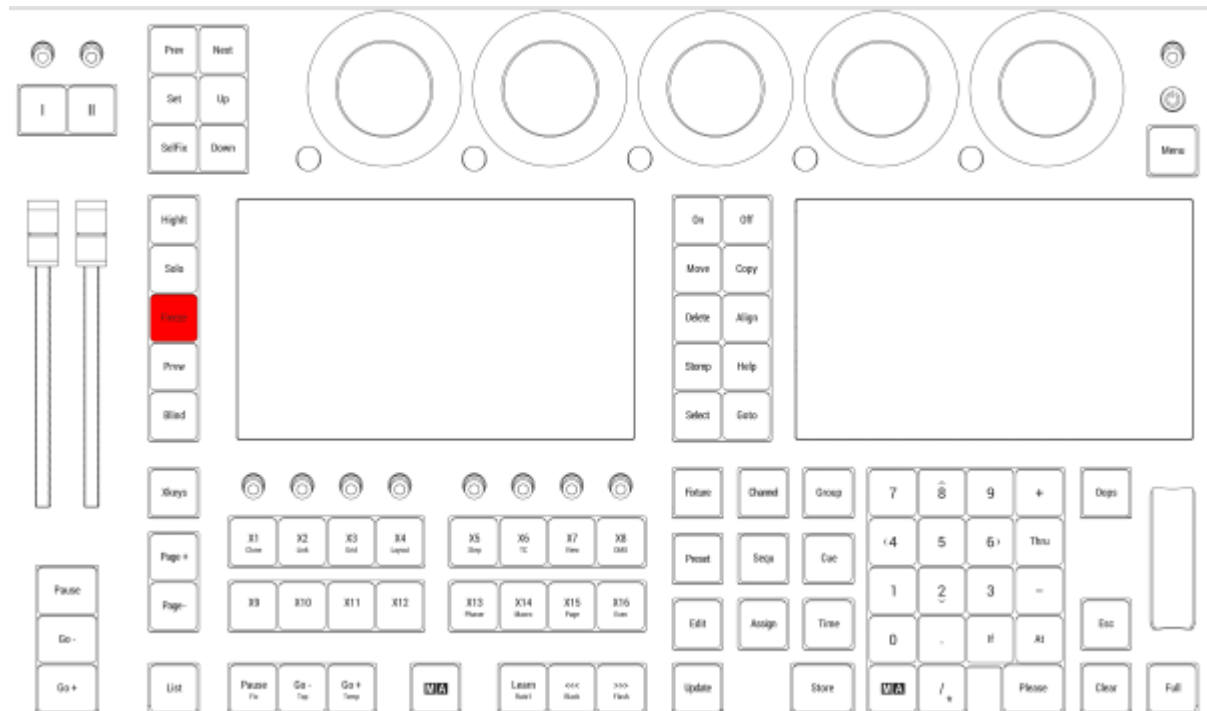
2.15.20. Freeze Key

Pressing **Freeze** toggles the Freeze function.

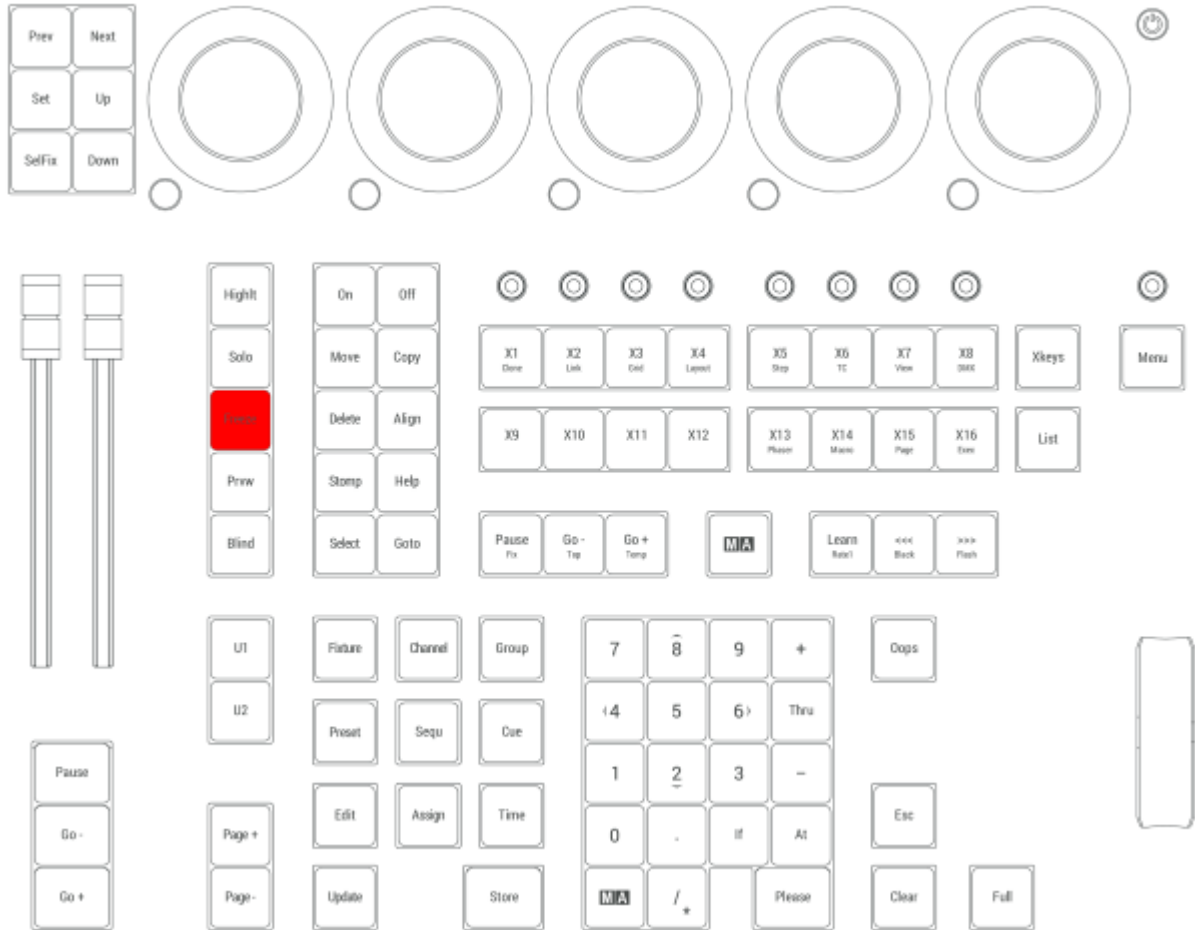
OK:Freeze

For more information about Freeze, see the [Freeze keyword](#) topic.

Freeze is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



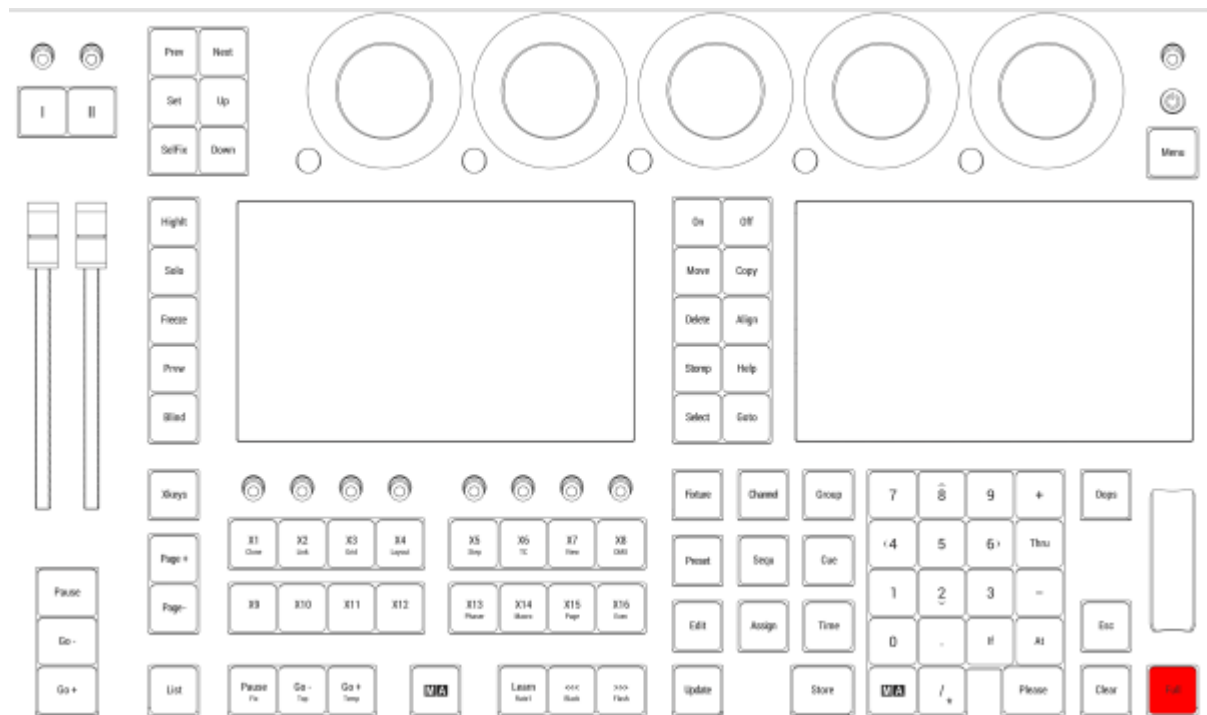
2.15.21. Full Key

Pressing **Full** enters the Full keyword into the command line. The dimmer values for the selected fixtures sets to 100%.

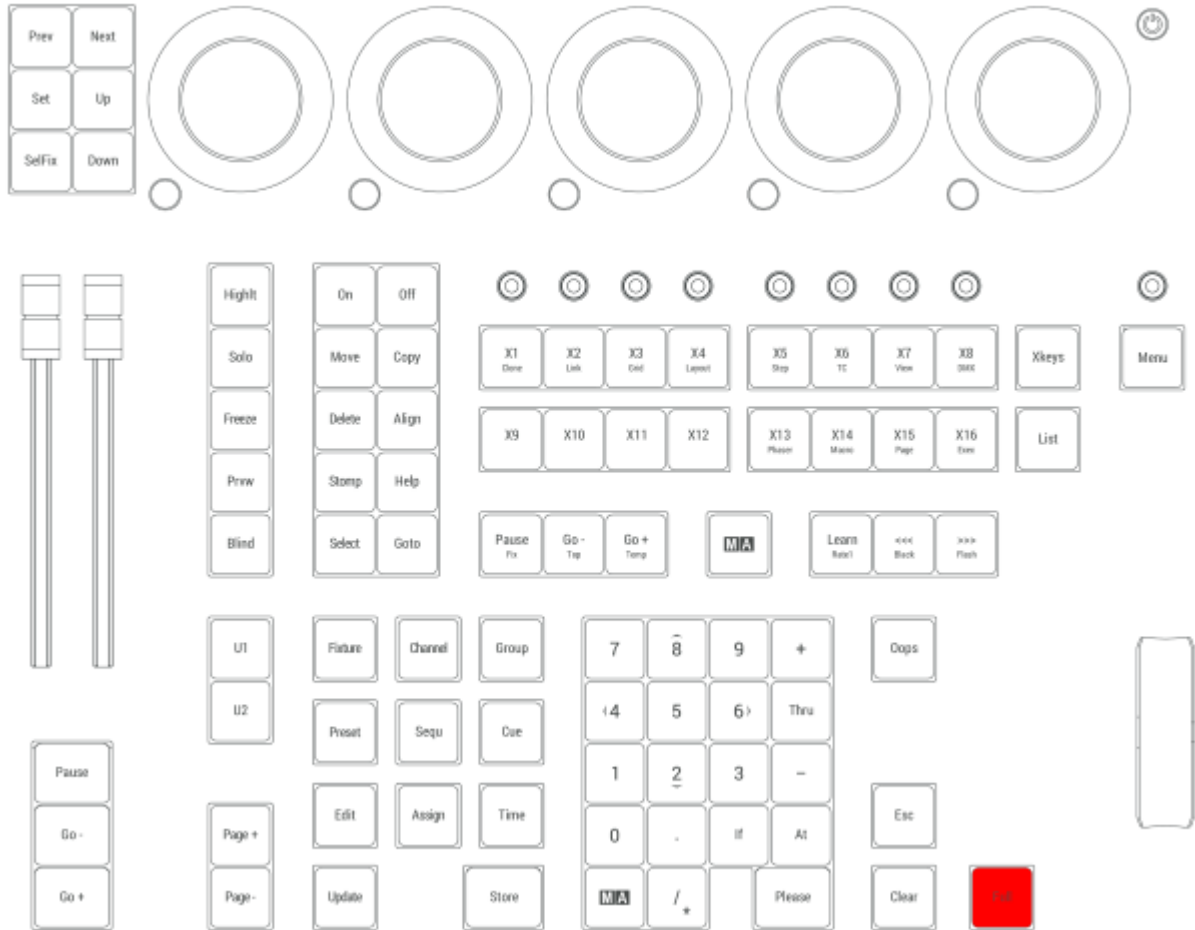


For more information about Full, see the [Full keyword](#) topic.

Full is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.22. Fixture Key

Pressing **Fixture** enters the Fixture keyword into the command line.



For more information about Fixture, see the [Fixture keyword](#) topic.

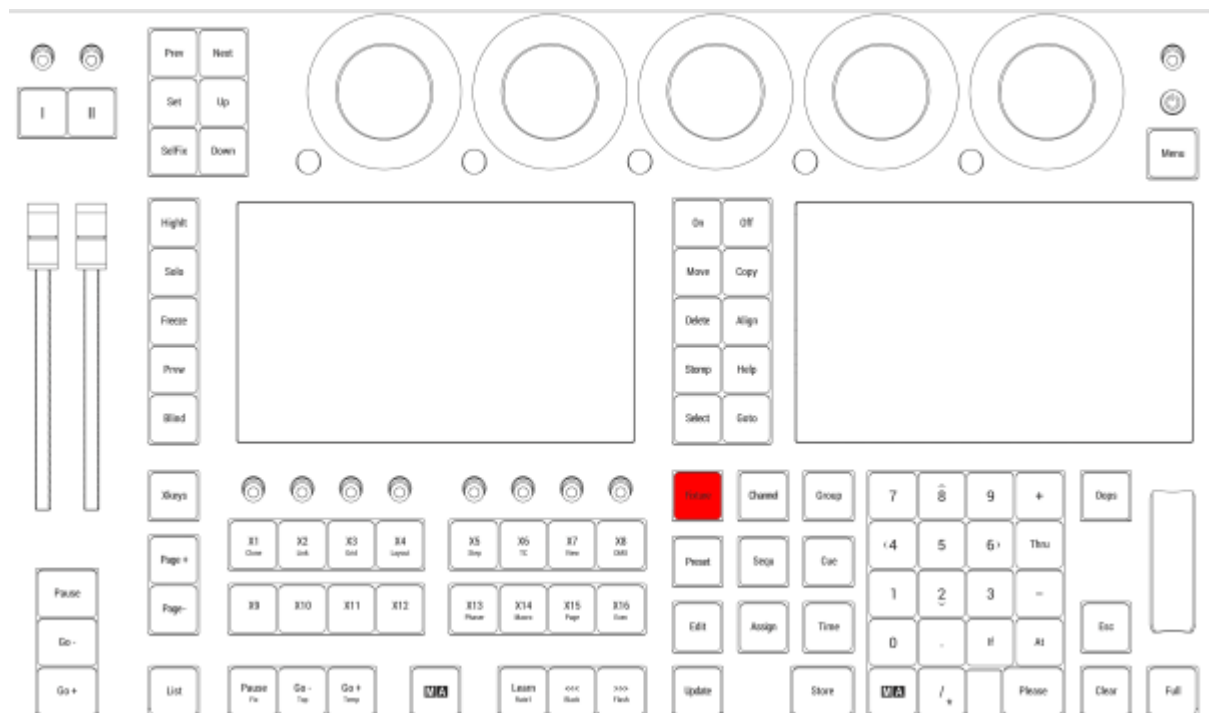
Selection

Pressing **Fixture** **Fixture** enters the Selection keyword into the command line.

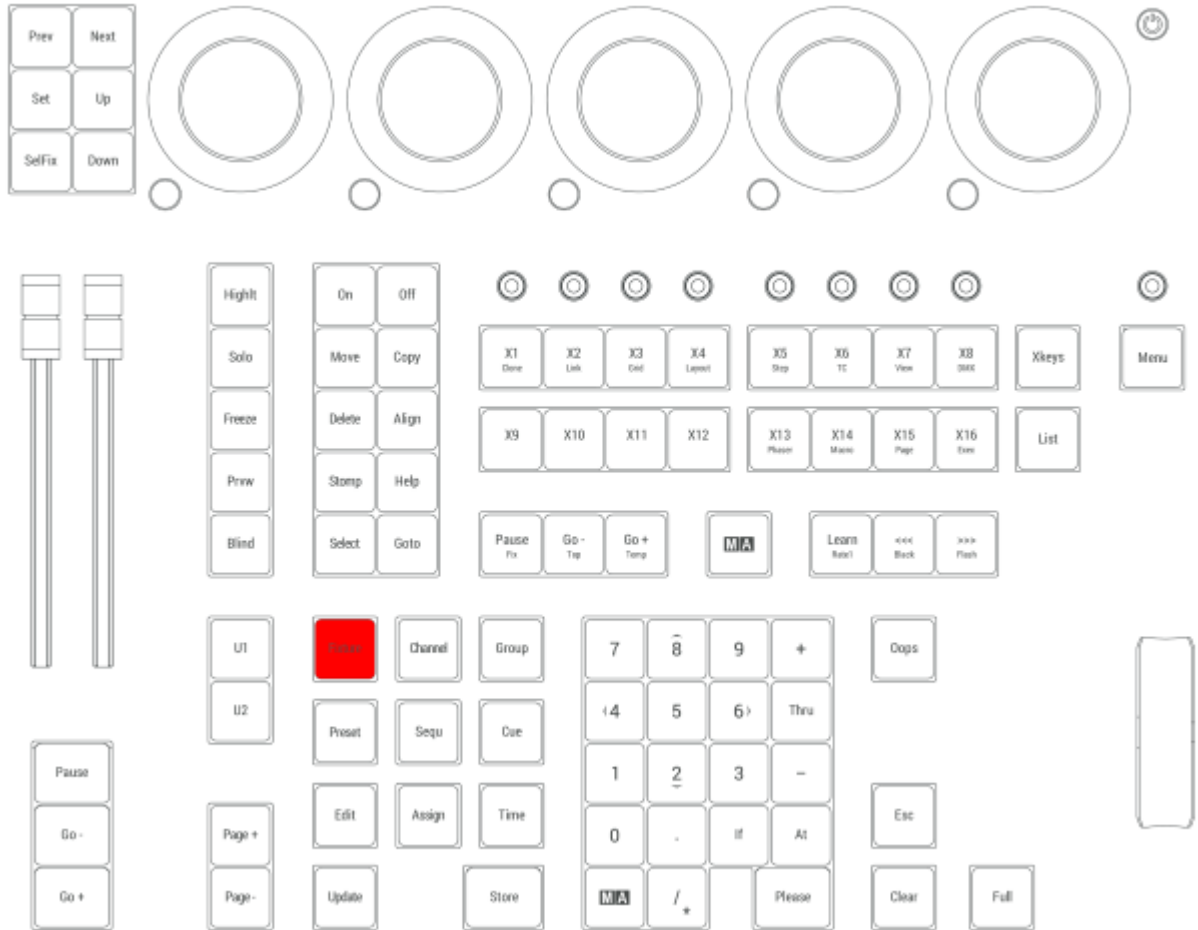


For more information about Selection, see the [Selection keyword](#) topic.

Fixture is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.23. Group Key

Pressing **Group** enters the Group keyword into the command line.



For more information about Group, see the [Group keyword](#) topic.

World

Pressing **Group** **Group** enters the World keyword into the command line.



For more information about World, see the [World keyword](#) topic.

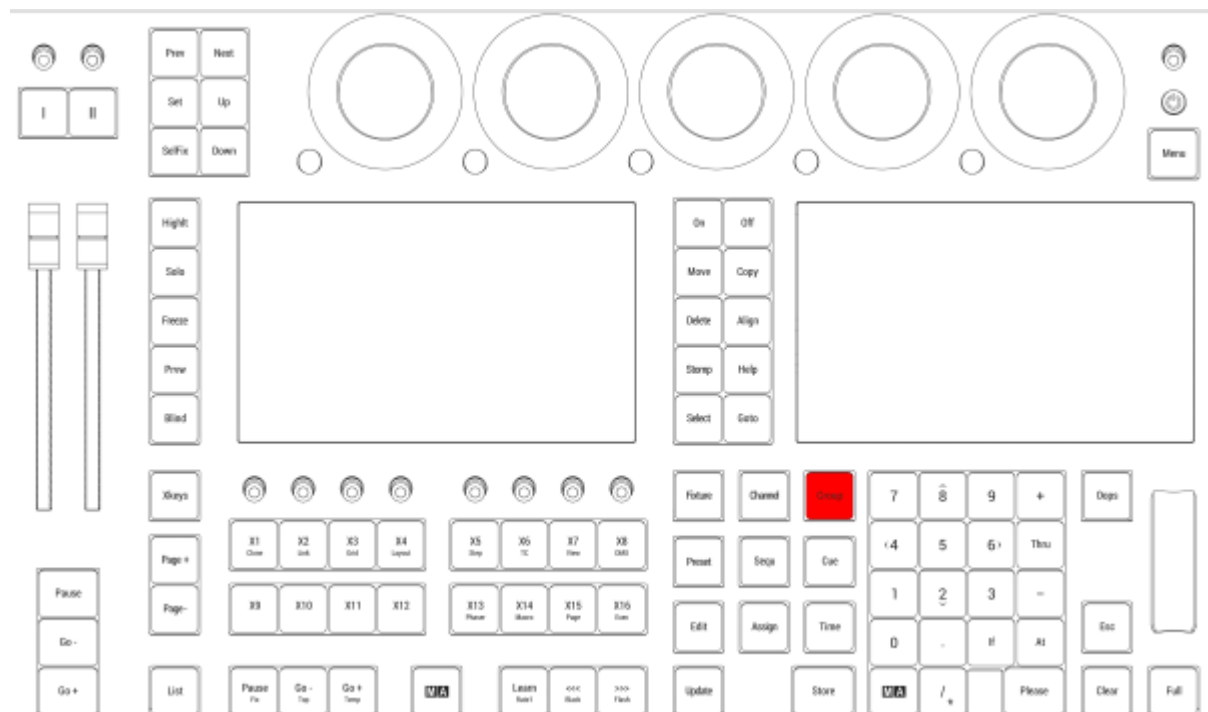
Filter

Pressing **Group** **Group** **Group** enters the Filter keyword into the command line.

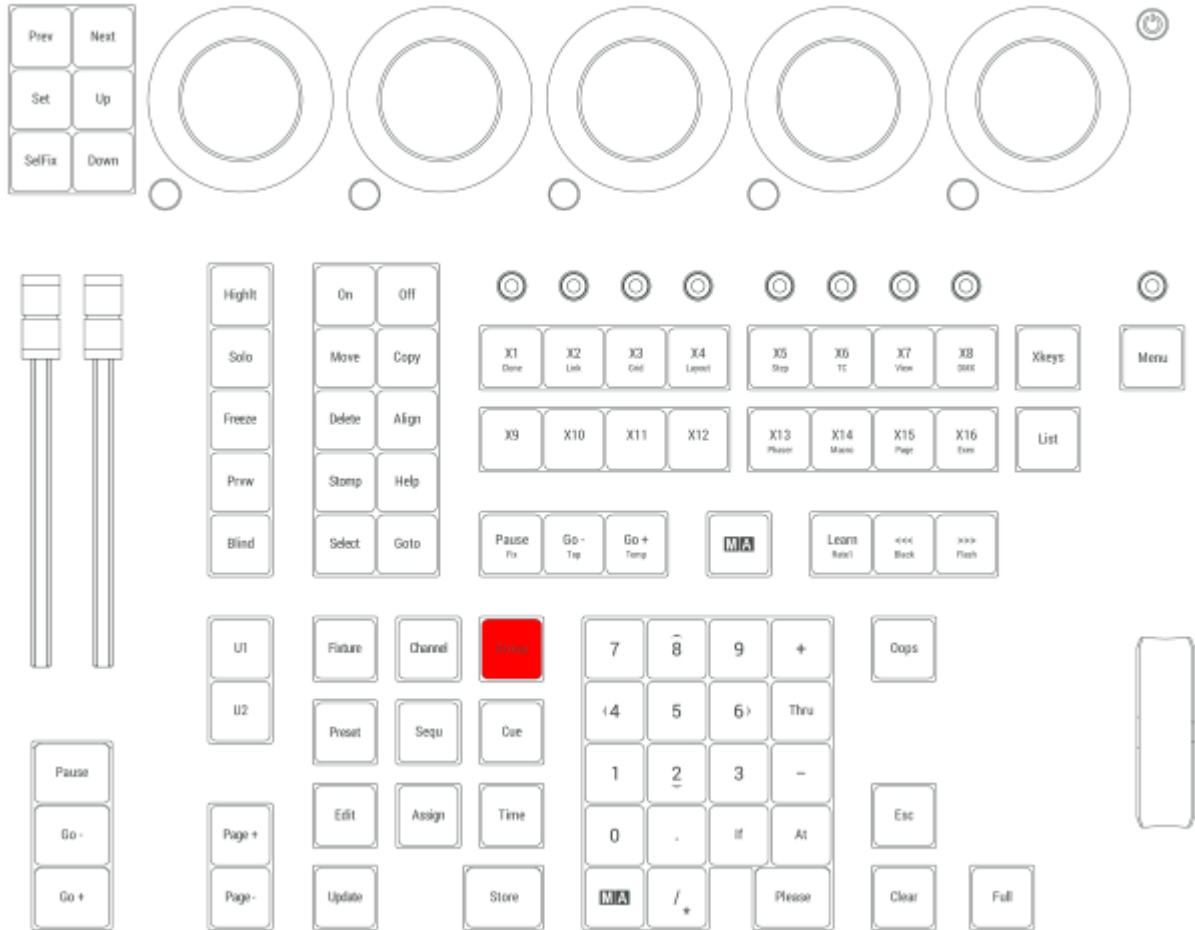


For more information about Filter, see the [Filter keyword](#) topic.

Group is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.24. Goto Key

Pressing **Goto** enters the Goto keyword into the command line.



For more information about Goto, see the [Goto keyword](#) topic.

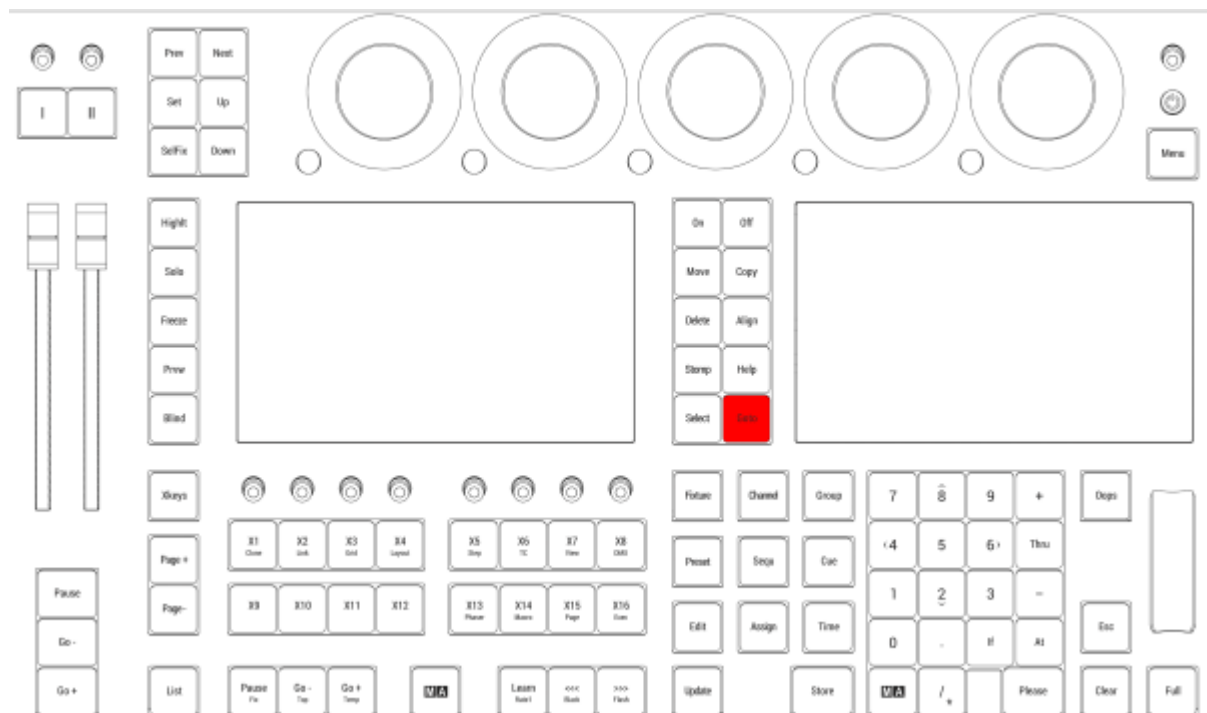
Load

Pressing **Goto** **Goto** enters the Load keyword into the command line.

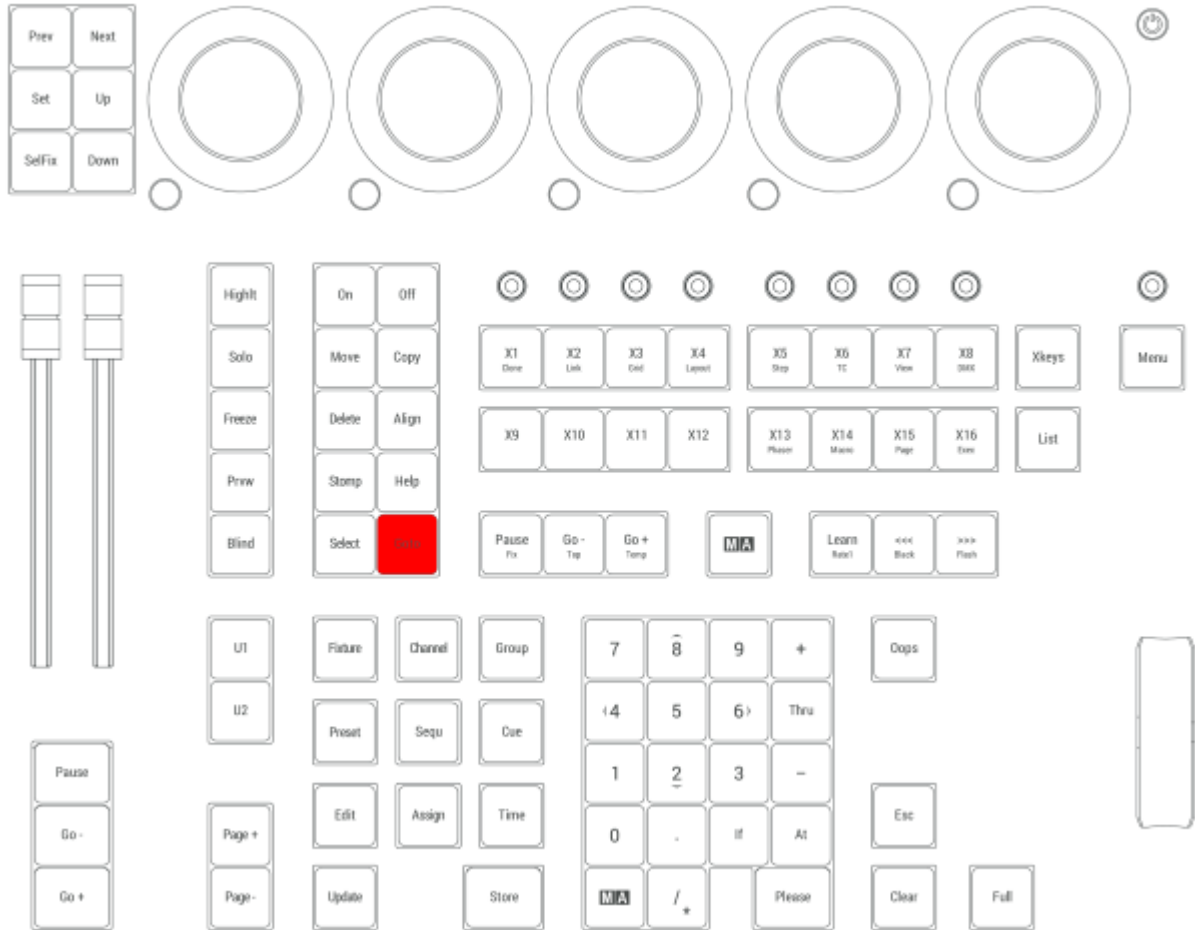


For more information about Load, see the [Load keyword](#) topic.

Goto is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.25. Go+ [large] Key

Go+ Executor is executed immediately in the command line when pressing **Go+** [large].

```
OK : Go+ Executor
```

For more information about Go+, see the [Go+ keyword](#) topic.

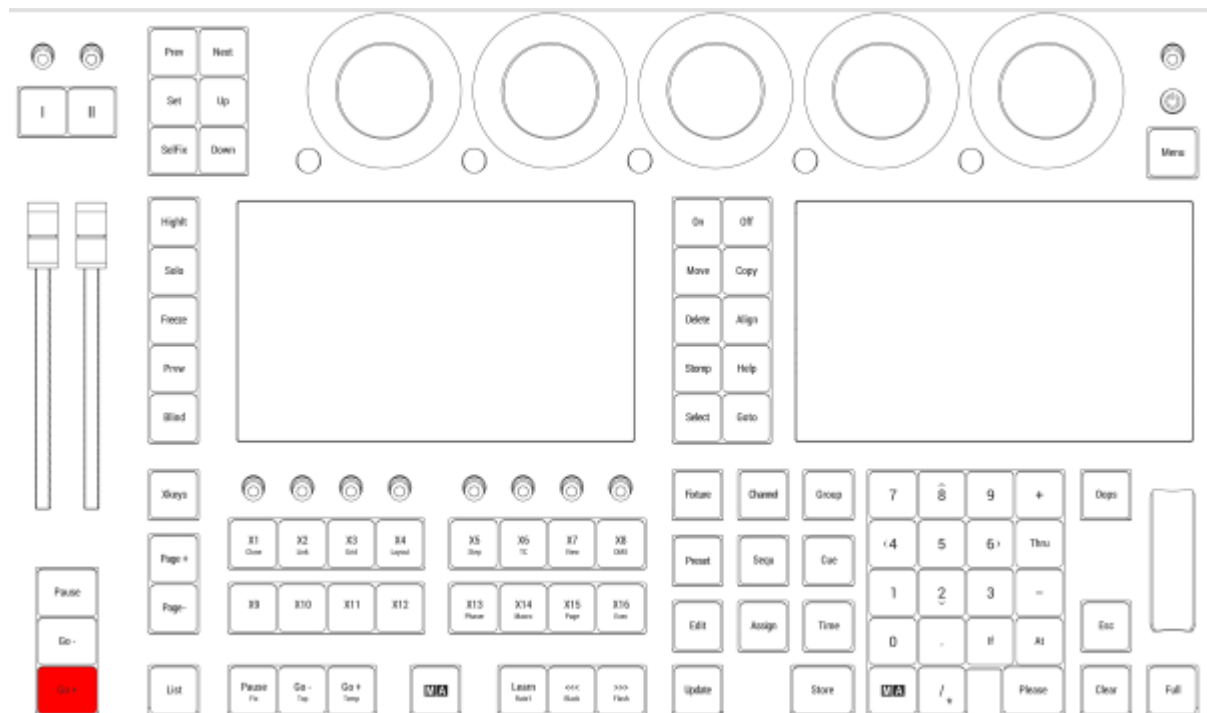
Go+ Loaded

Pressing **MA** + **Go+** [large] executes the Go+ Loaded keyword in the command line.

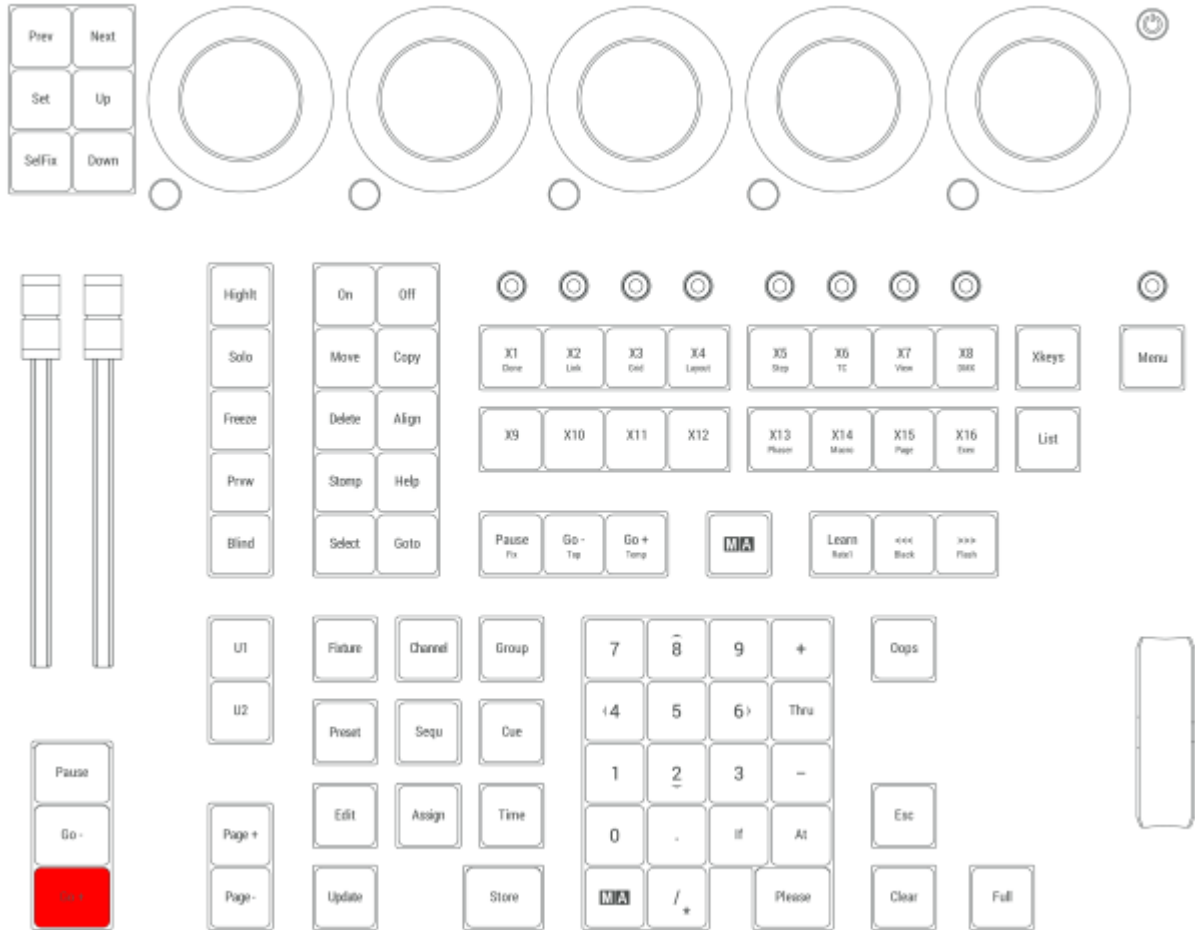
```
MA User name[Fixture]> Go+ Loaded
```

For more information about Go+, see the [Go+ keyword](#) topic.

Go+ [large] is located in the master section under the two master faders.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



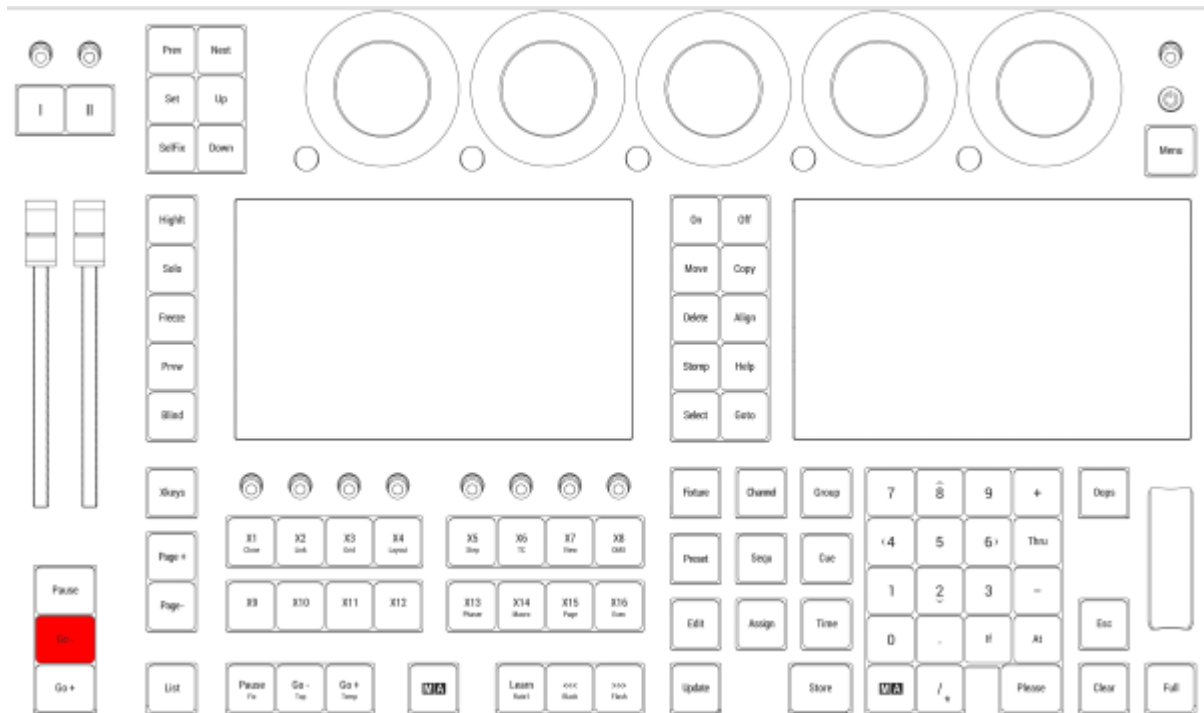
2.15.26. Go- [large] Key

Go- Executor is executed immediately in the command line when pressing **Go-** [large].

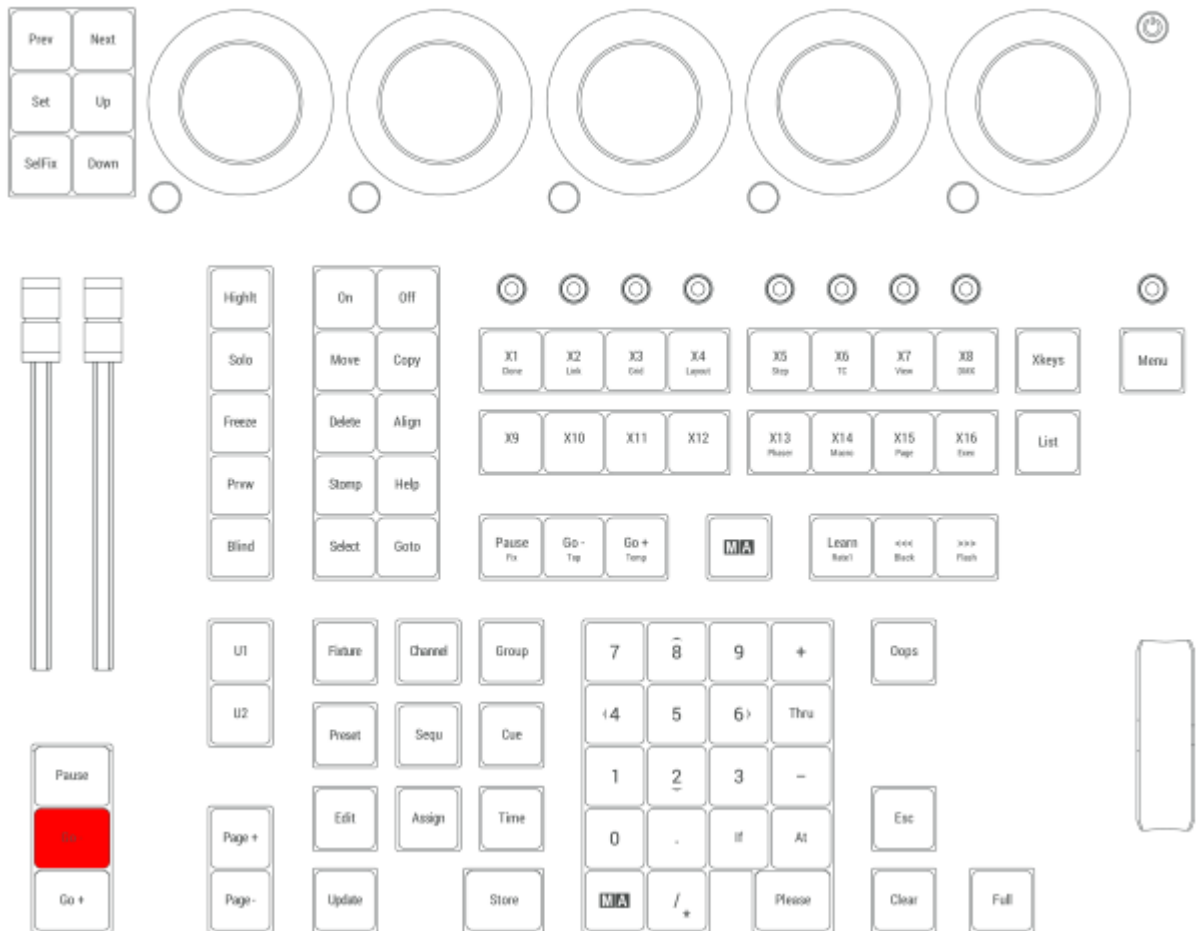
OK : Go- Executor

For more information about Go-, see the **Go- keyword** topic.

Go- [large] is located in the master section under the two master faders.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.27. Go+ | Temp Key

Pressing **Go+** enters the Go+ keyword into the command line.

```
MA User name[Fixture]> Go+
```

For more information about Go+, see the [Go+ keyword](#) topic.

Temp

Pressing and holding **MA** + **Go+** enters the Temp keyword into the command line.

```
MA User name[Fixture]> Temp
```

For more information about Temp, see the [Temp keyword](#) topic.

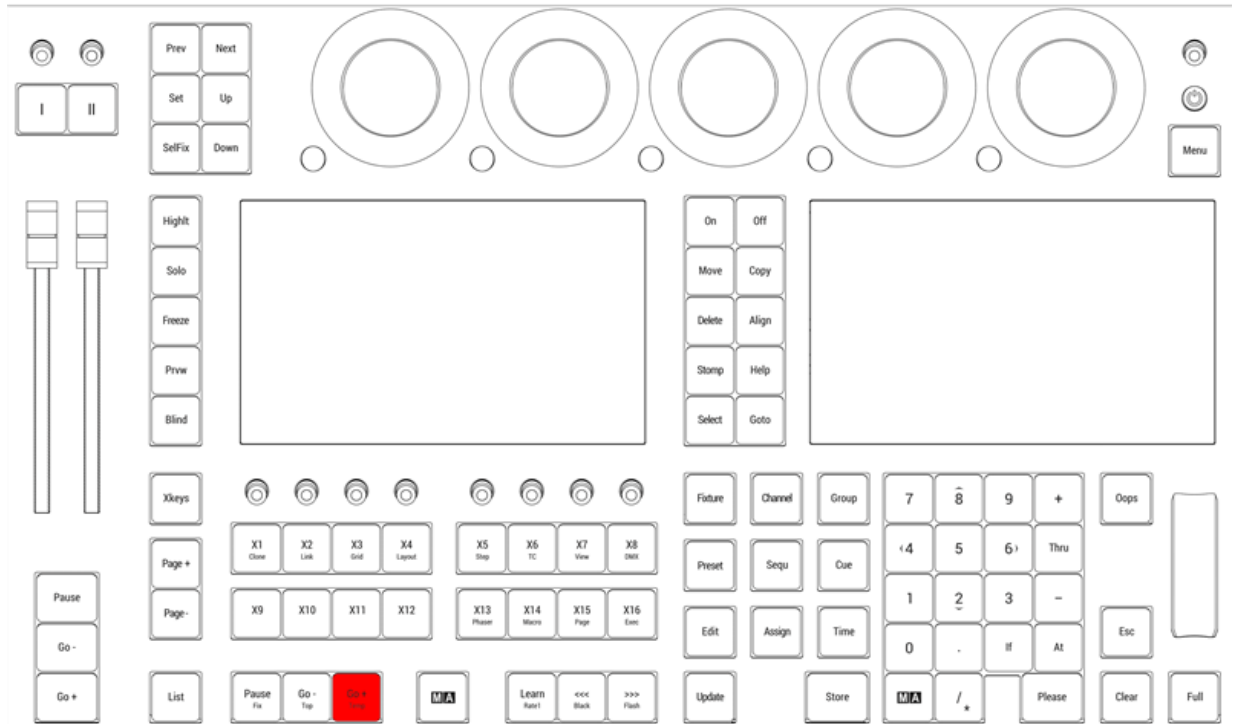
Toggle

Pressing and holding **MA** + **Go+** **Go+** enters the Toggle keyword into the command line.

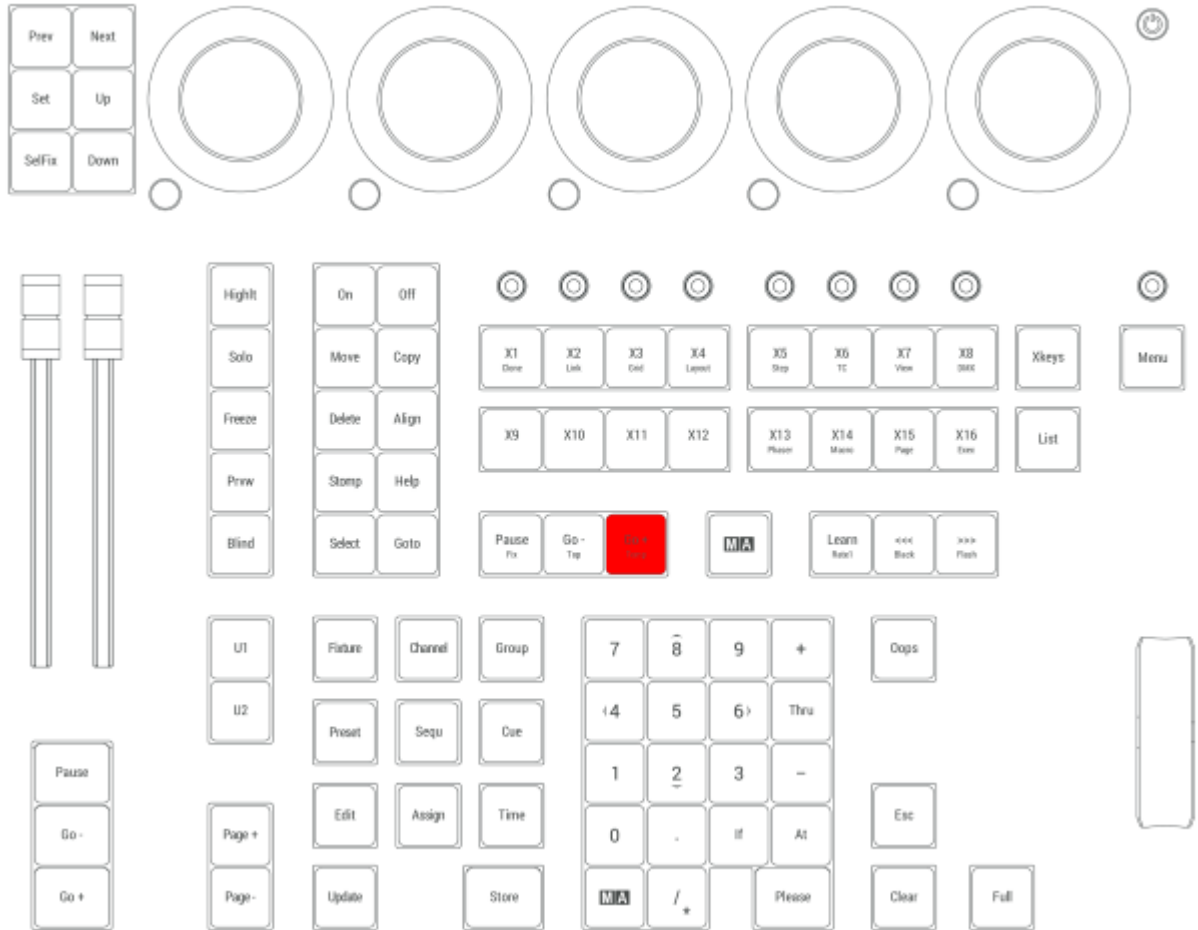
```
MA User name[Fixture]> Toggle
```

For more information about Toggle, see the [Toggle keyword](#) topic.

Go+ is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.28. Go- | Top Key

Pressing **Go-** enters the Go- keyword into the command line.



For more information about Go-, see the [Go- keyword](#) topic.

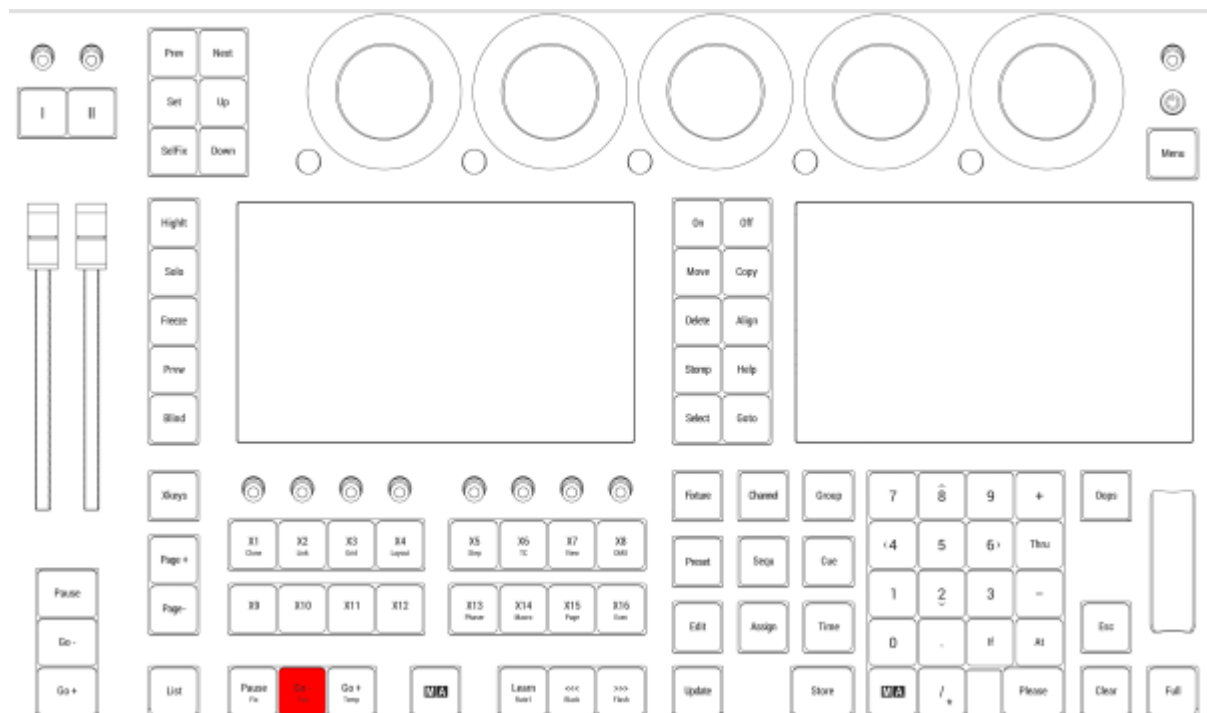
Top

Pressing and holding **MA** + **Go-** enters the Top keyword into the command line.

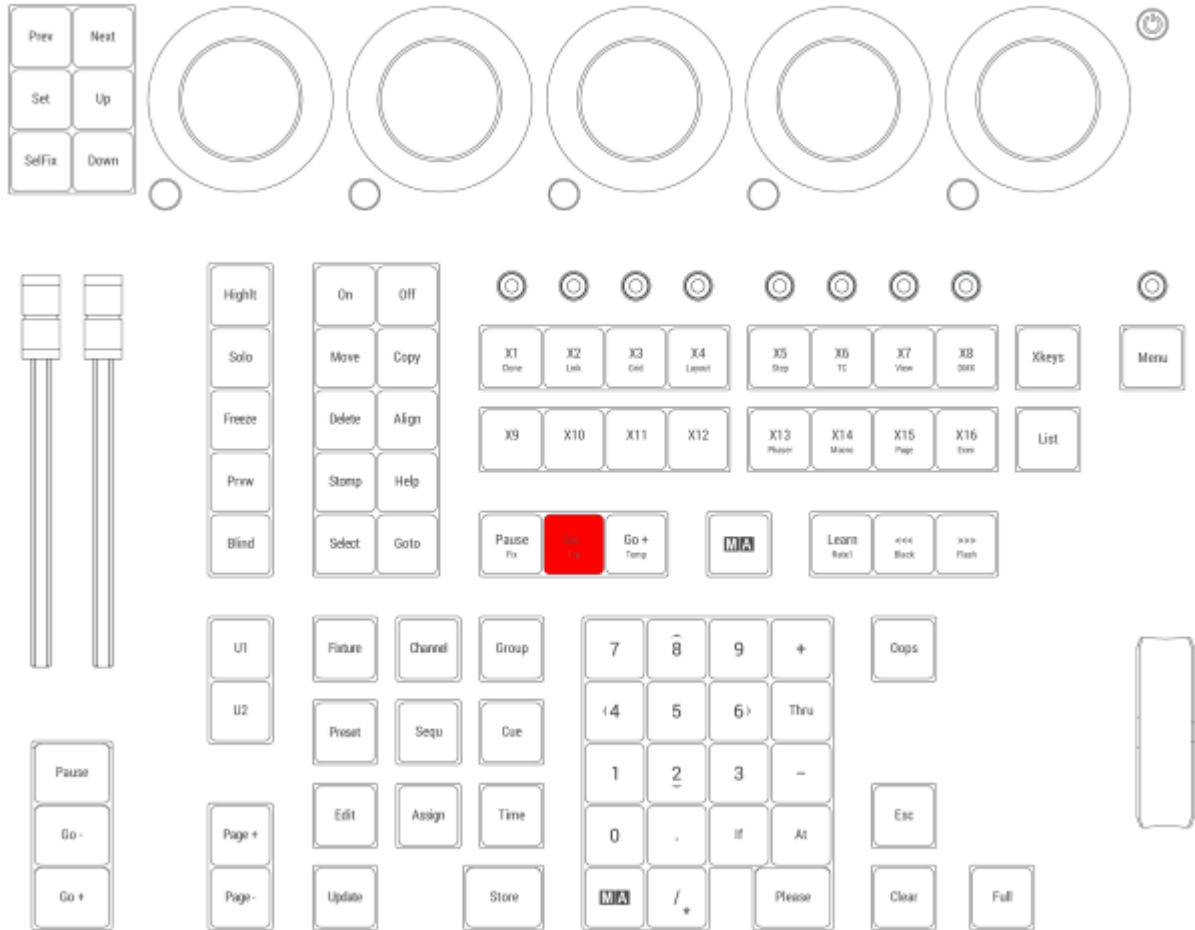


For more information about Top, see the [Top keyword](#) topic.

Go- is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.29. Help Key

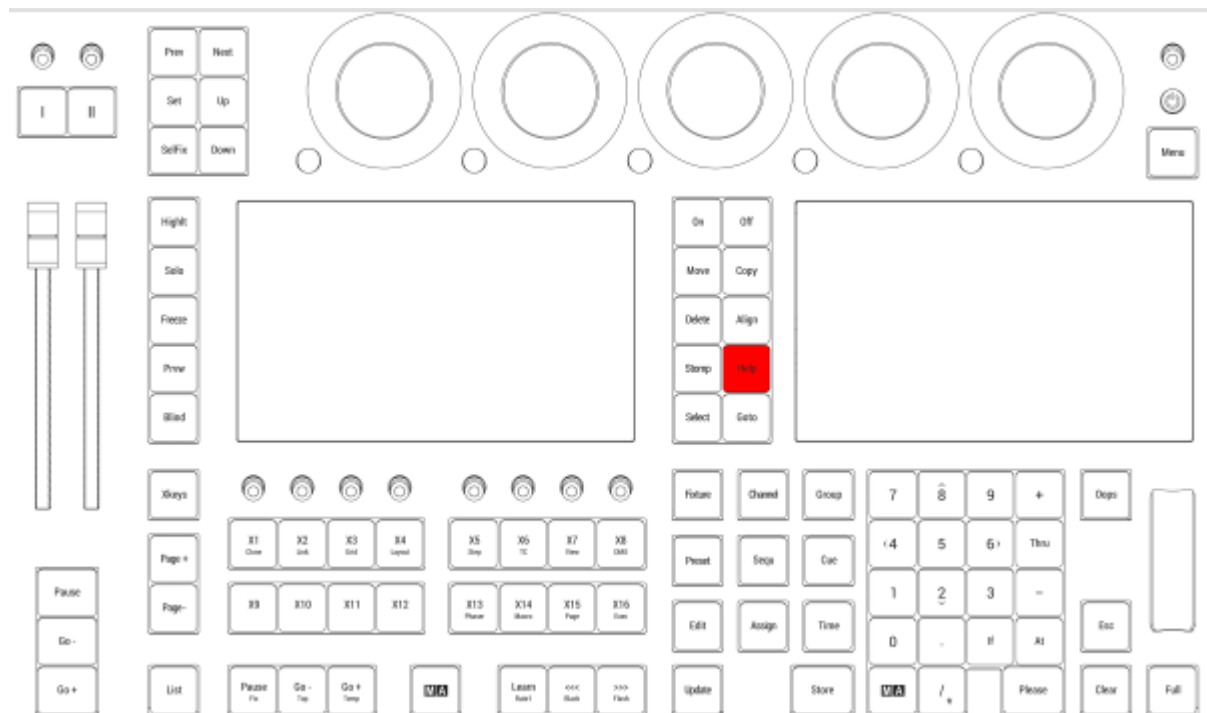
Pressing **Help** enters the Help keyword into the command line.



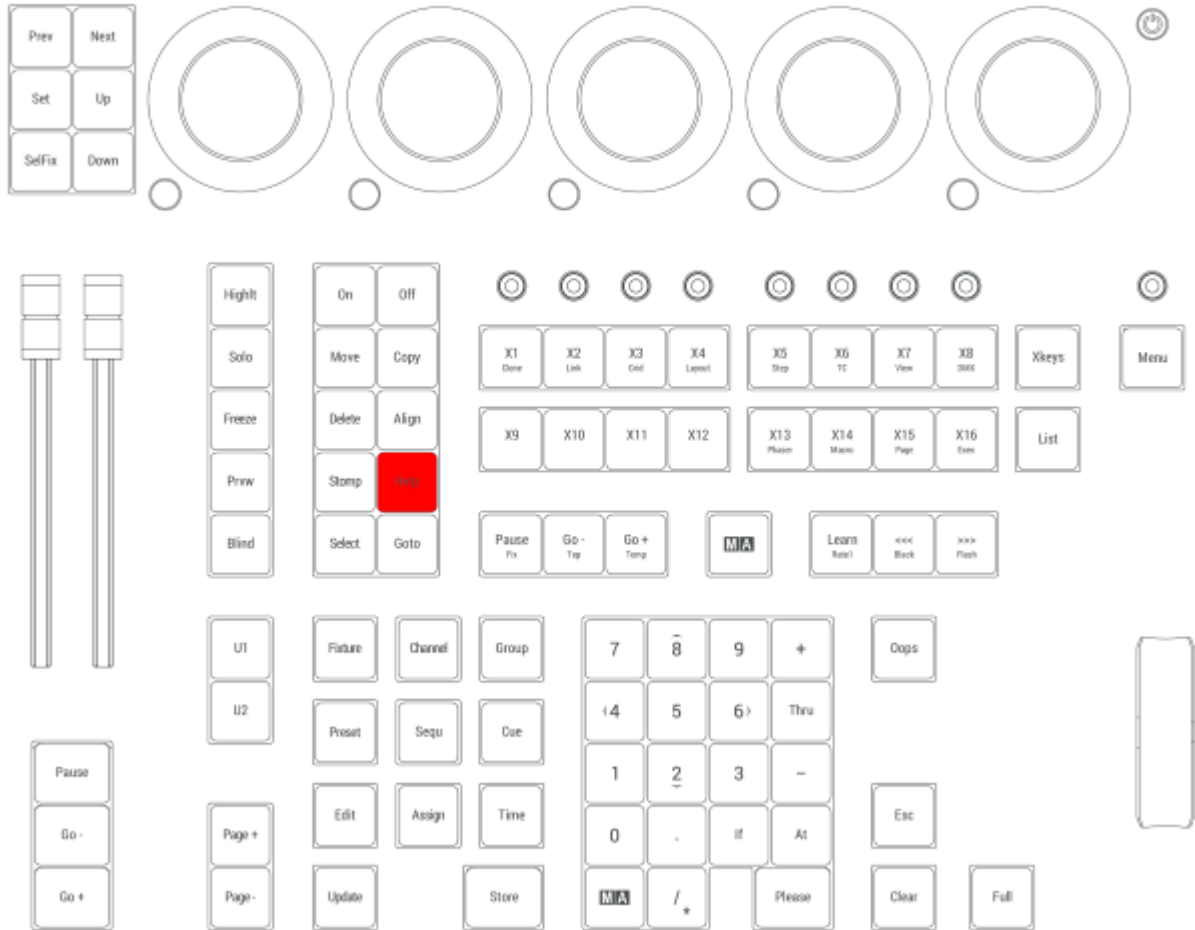
Pressing **Help** and an user interface element opens the corresponding help.

For more information about Help, see the [Help keyword](#) topic.

Help is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

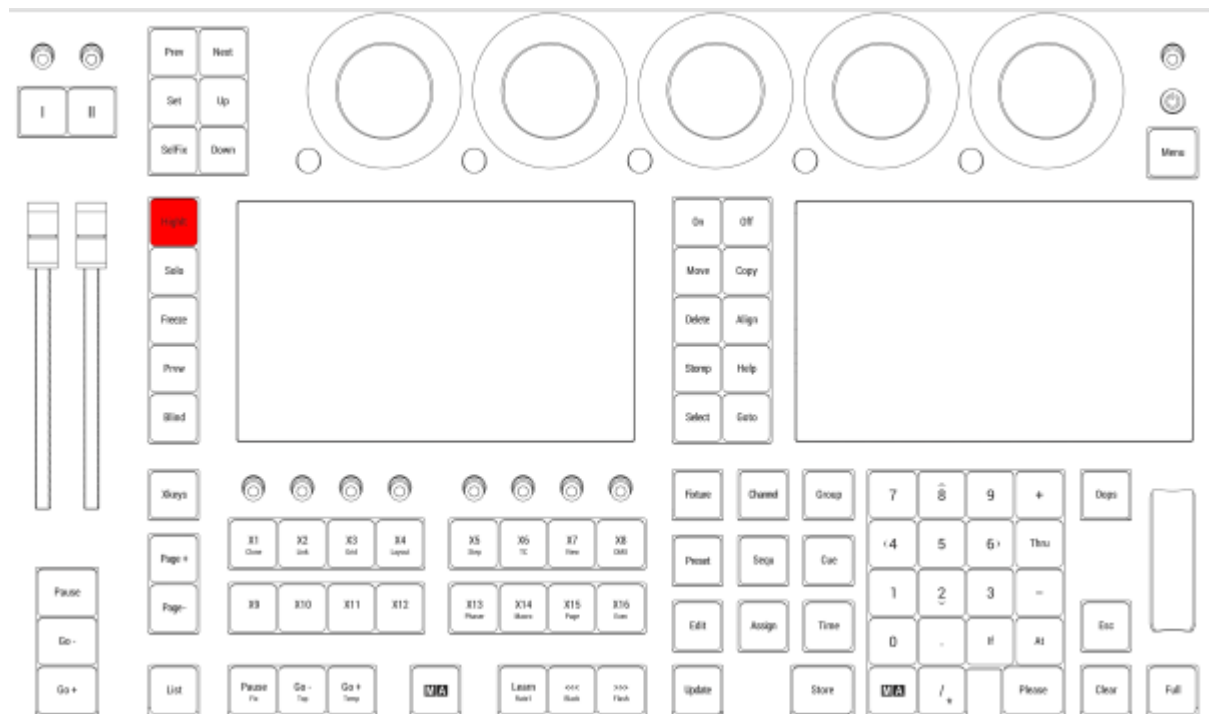


2.15.30. Hight [Highlight] Key

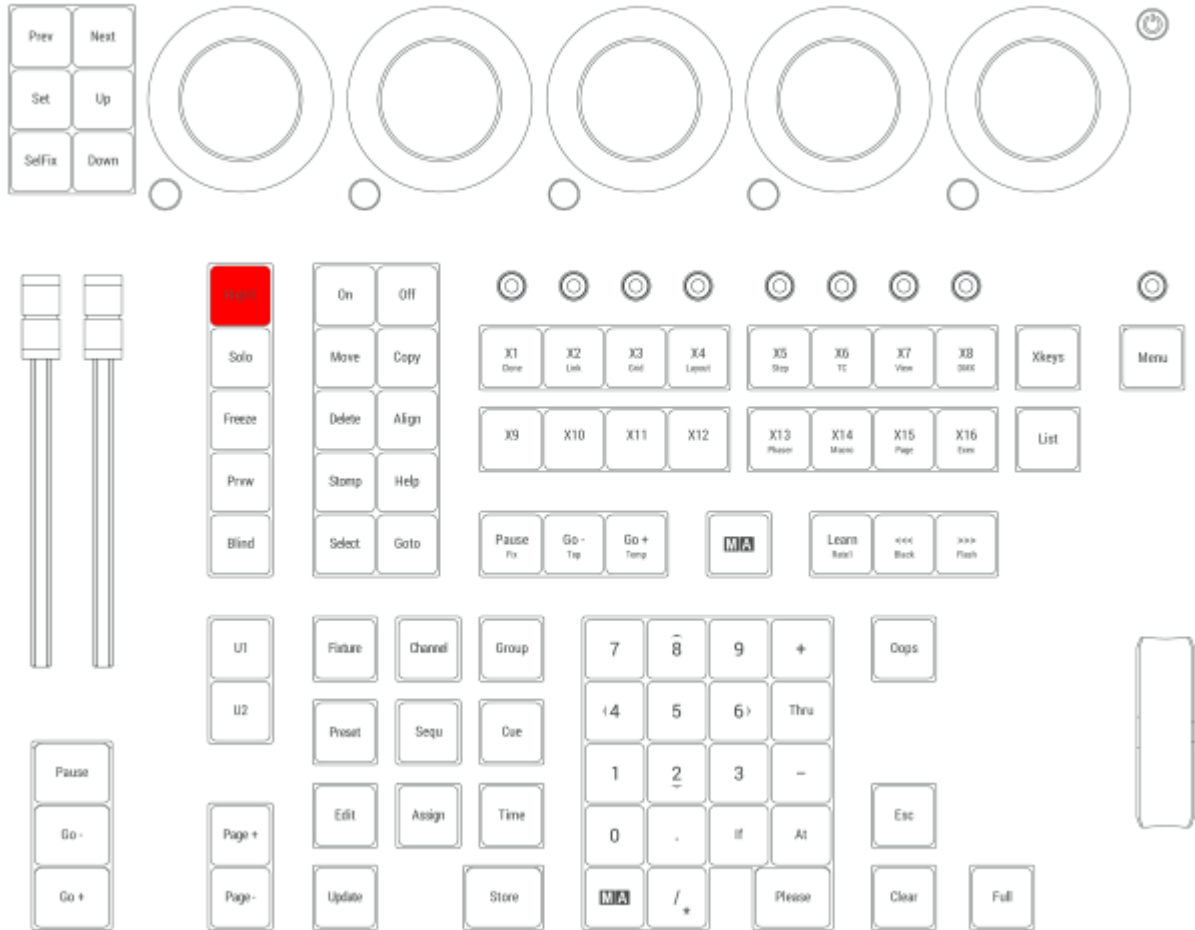
Pressing **Hight** toggles the highlight function. For more information about Highlight, see the [Highlight keyword](#) topic.

Pressing **MA** + **Hight** toggles the lowlight function. For more information about Lowligh, see the [Lowligh keyword](#) topic.

Hight is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.31. If Key

IfOutput

Pressing **If** enters the IfOutput keyword into the command line.



For more information about IfOutput, see the [IfOutput keyword](#) topic.

IfActive

Pressing **If If** enters the IfActive keyword into the command line.



For more information about IfActive, see the [IfActive keyword](#) topic.

IfProg

Pressing **If If If** enters the IfProg keyword into the command line.



For more information about IfProg, see the [IfProg keyword](#) topic.

If

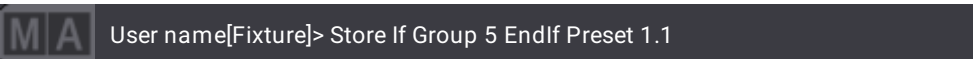
Pressing **If If If If** enters the If keyword into the command line.



For more information about If, see the [If keyword](#) topic.

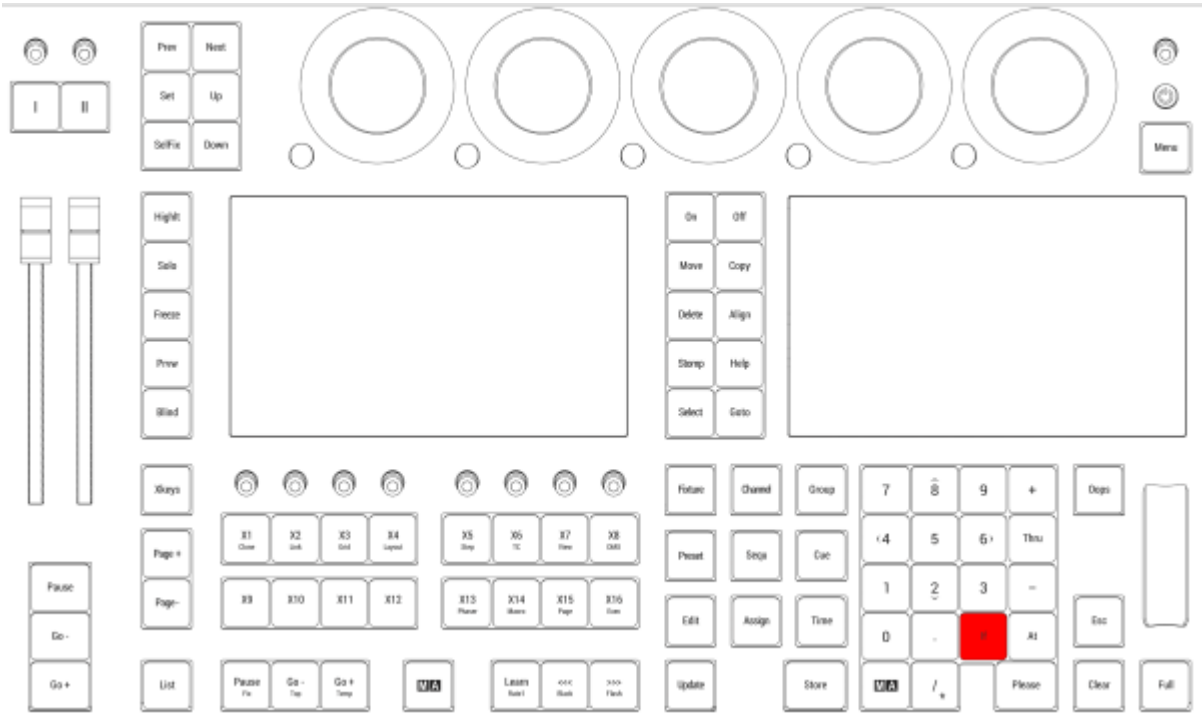
EndIf

Pressing **If** again once the If command is already in the command line enters the EndIf keyword into the command line.

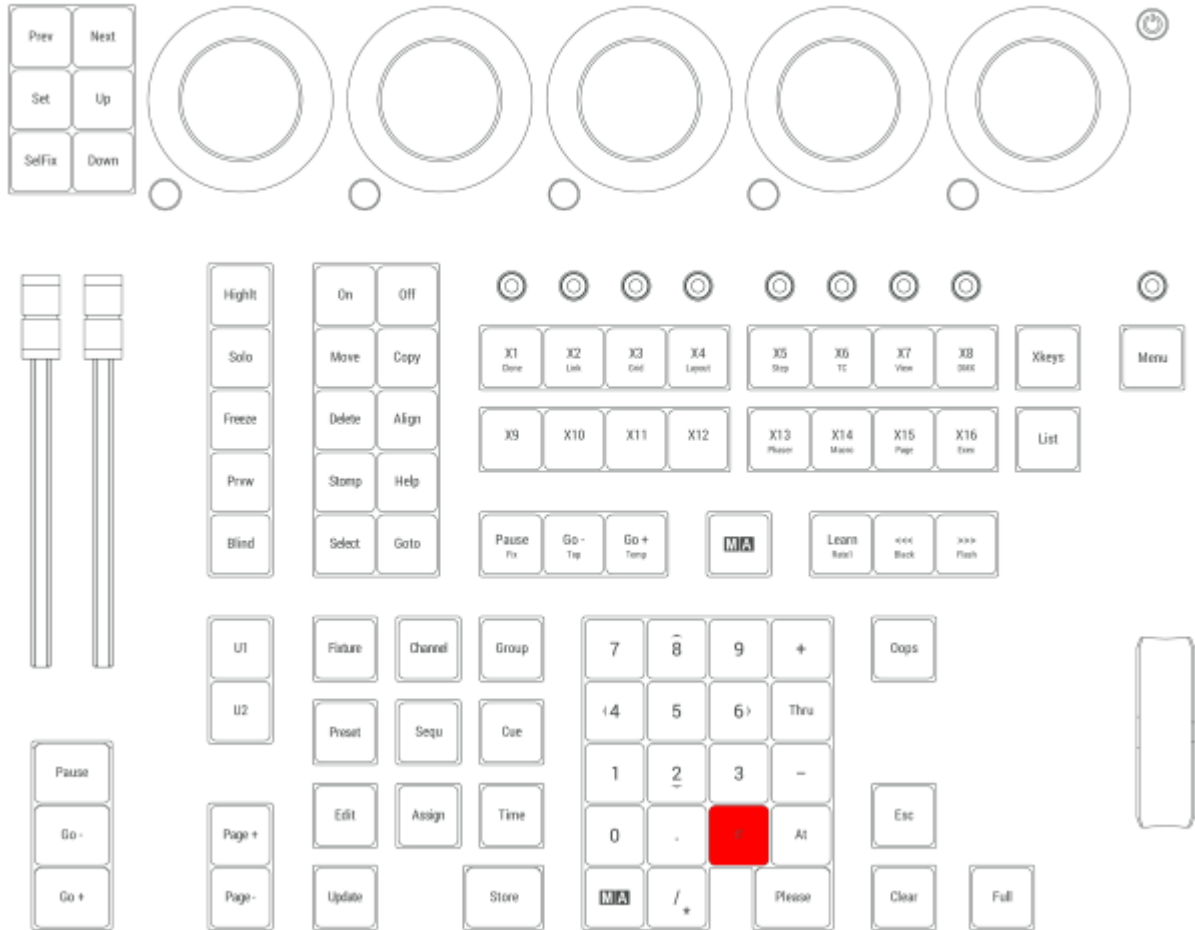


For more information about EndIf, see the [EndIf keyword](#) topic.

If is located in the numeric keys section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.32. Learn | Rate1 Key

Pressing **Learn** enters the Learn keyword into the command line.



For more information about Learn, see the [Learn keyword](#) topic.

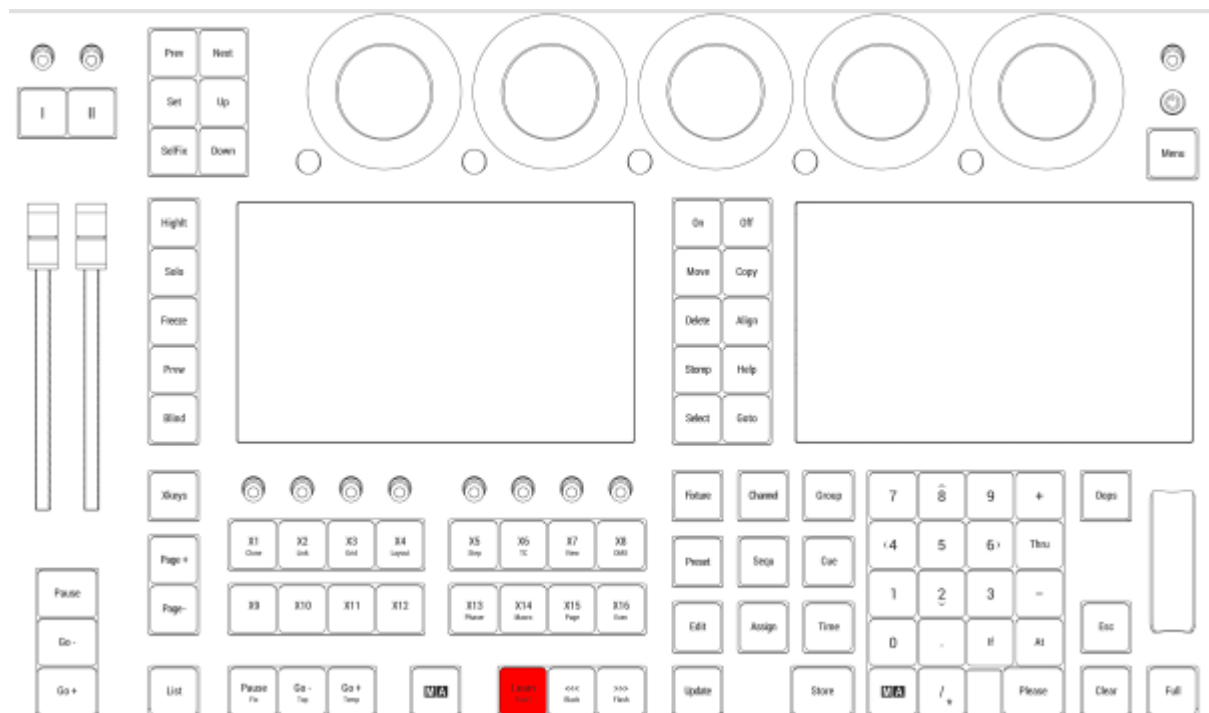
Rate1

Pressing and holding **MA** + **Learn** enters the Rate1 keyword into the command line.

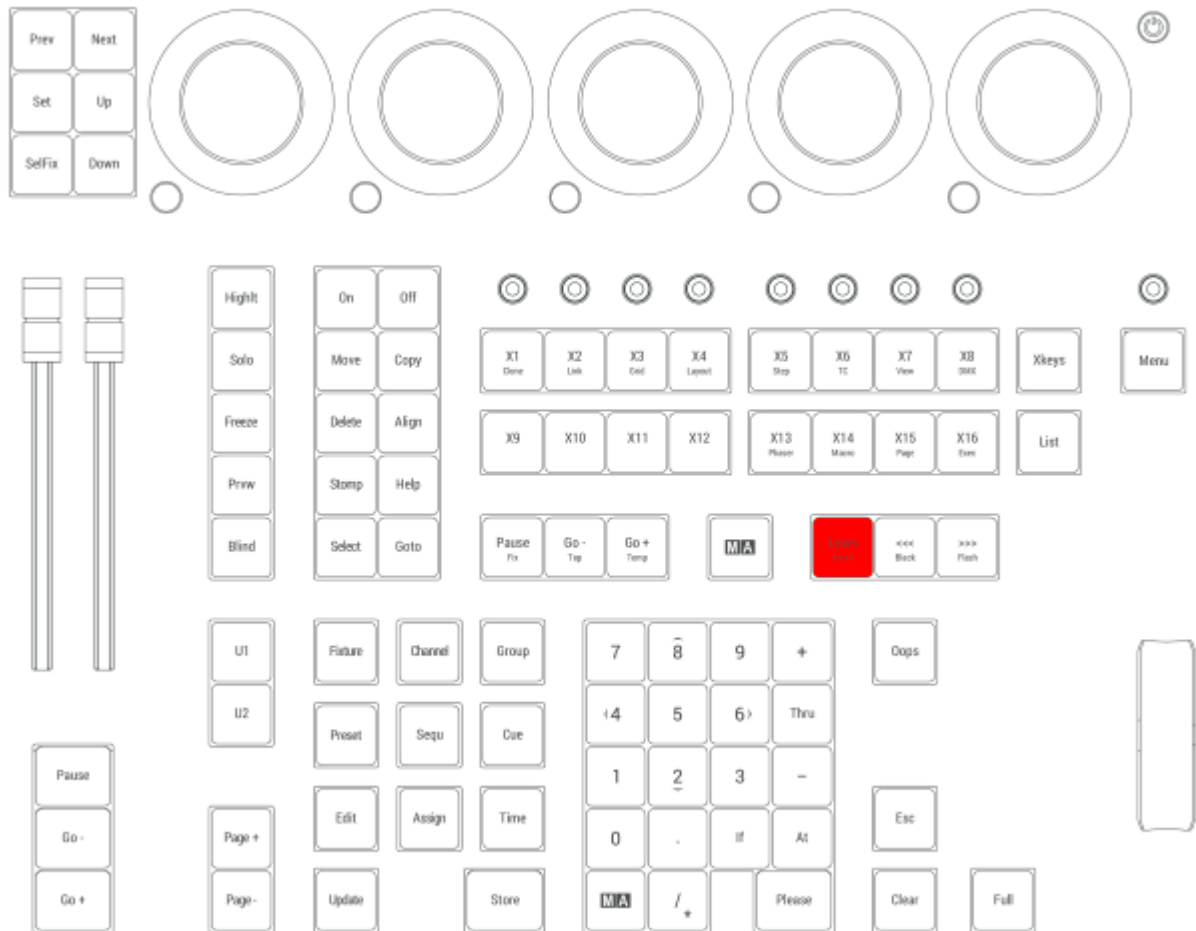


For more information about Rate1, see the [Rate1 keyword](#) topic.

Learn is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.33. List Key

Pressing **List** enters the List keyword into the command line.



For more information about List, see the [List keyword](#) topic.

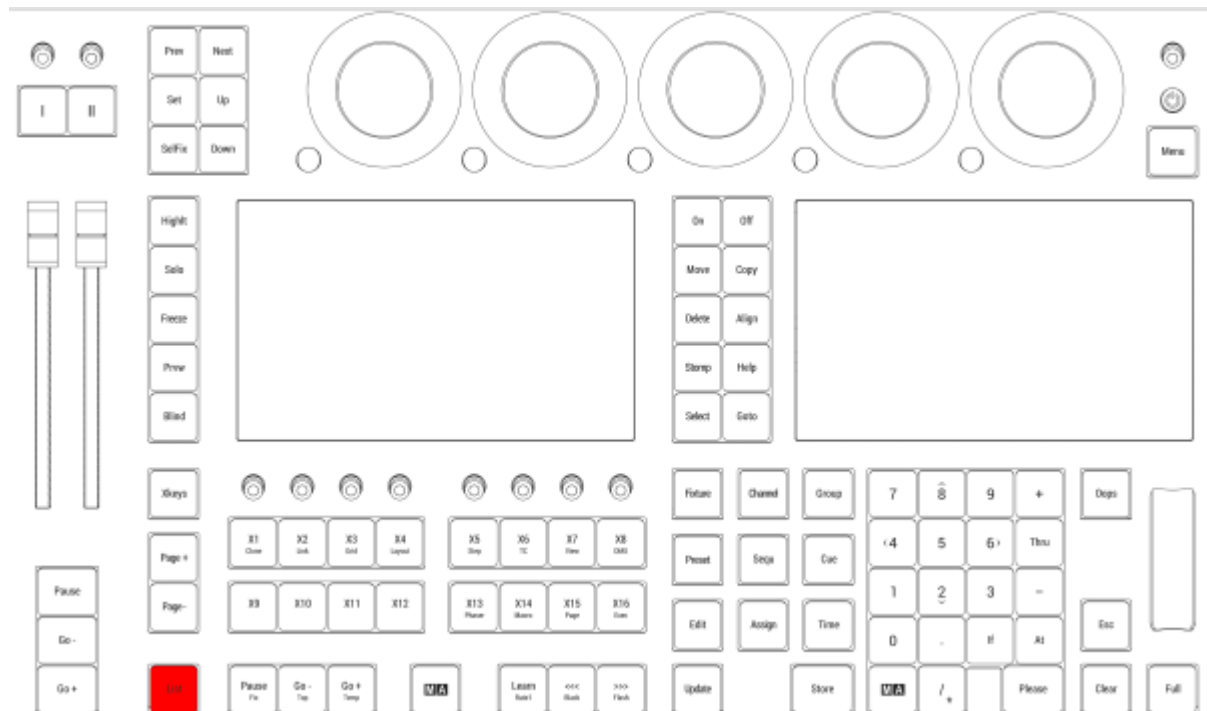
ListRef

Pressing and holding **MA** + **List** enters the ListRef keyword into the command line.

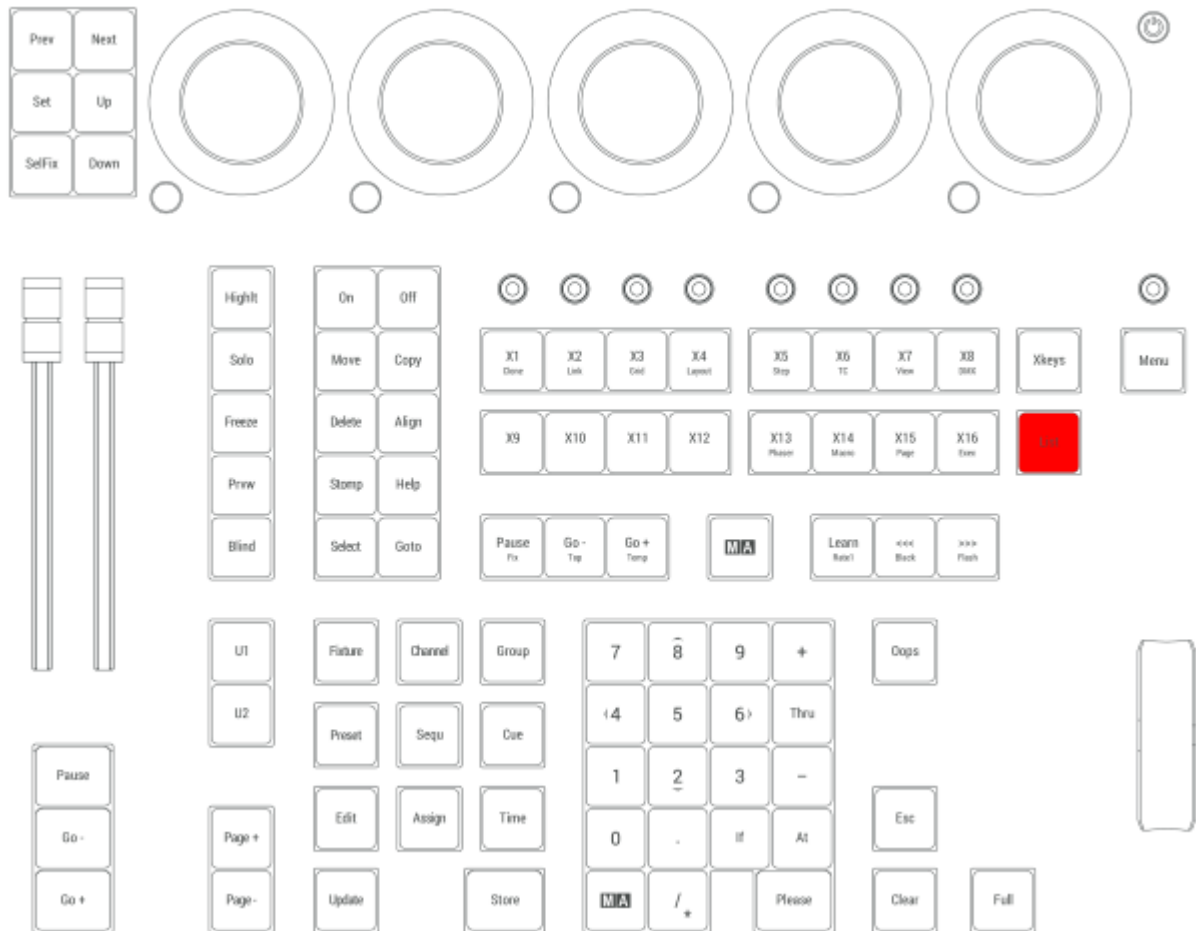


For more information about List, see the [ListRef keyword](#) topic.

List is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



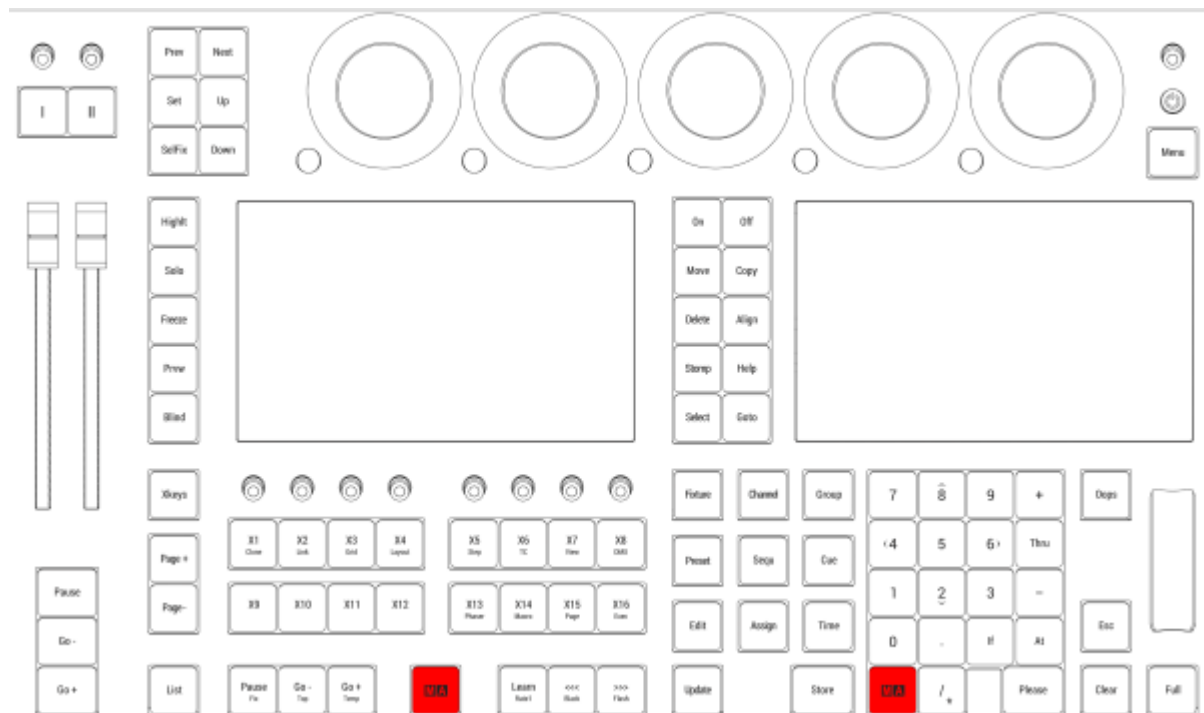
2.15.34. MA Key

Pressing and holding **MA** in combination with other keys, gives shortcuts to other functions. On both console layouts there are two MA keys. These are identical and have the same function.

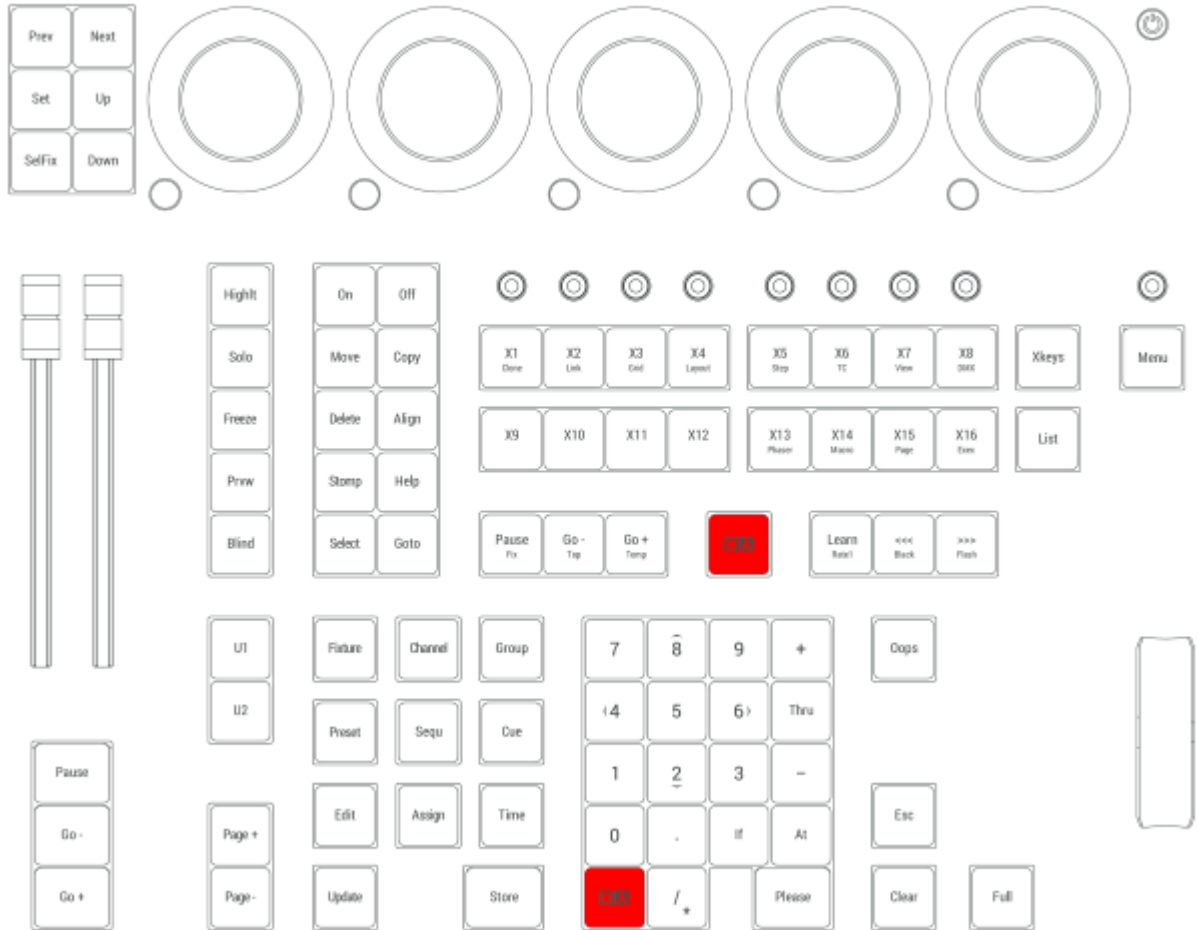
Pressing **MA** + **Please** forces the focus to the command line.

To get a complete list of functions, please see the **Keys topic**.

MA is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



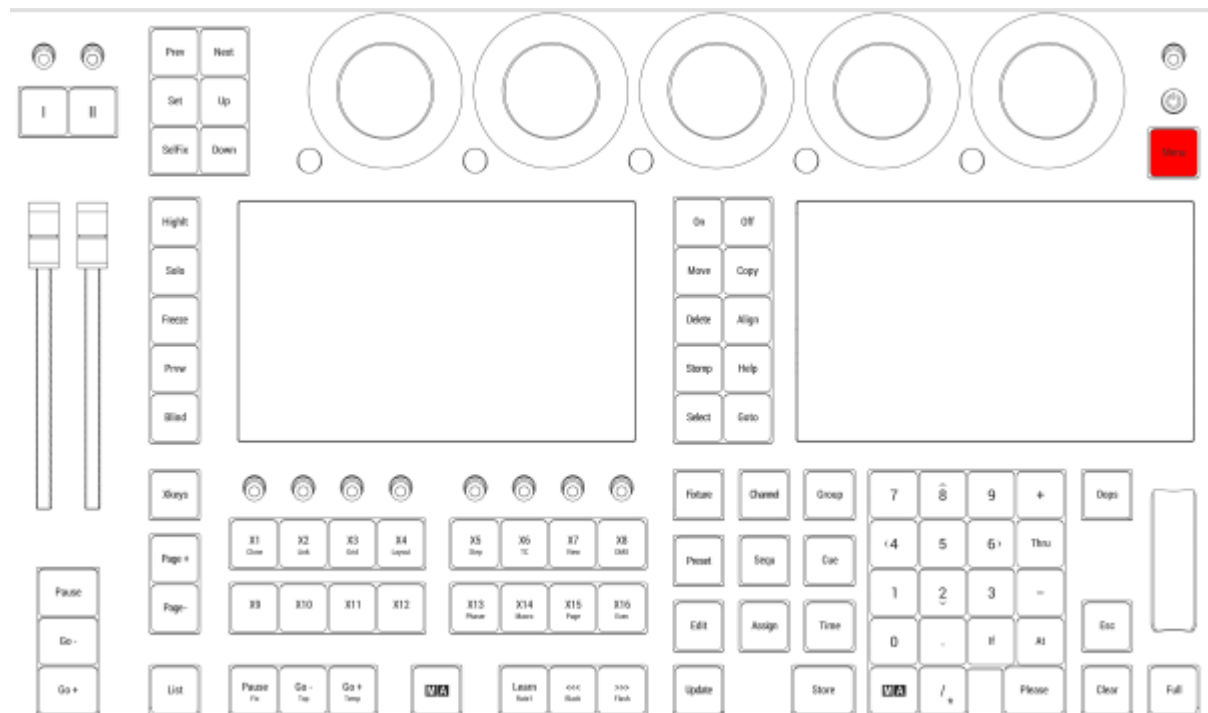
Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.35. Menu Key

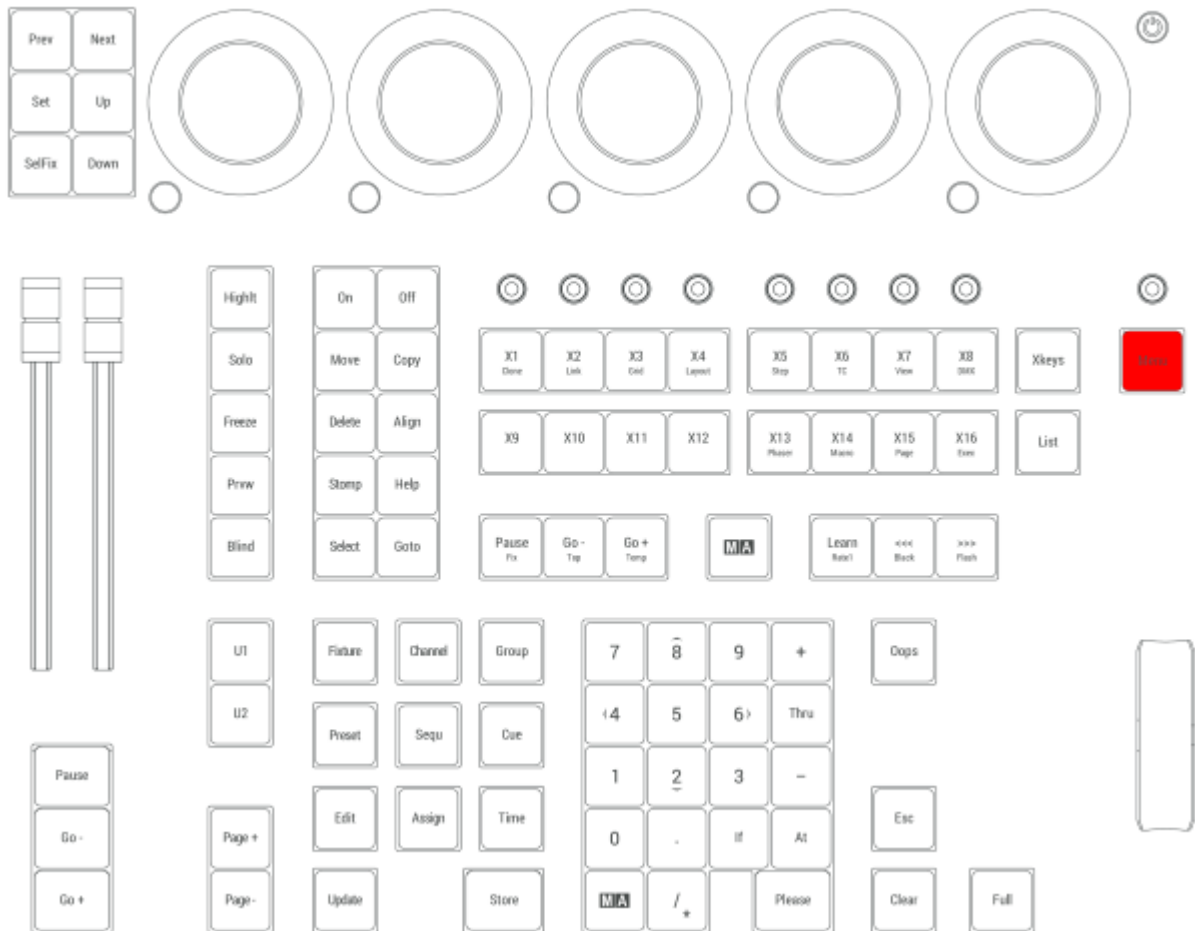
Pressing **Menu** opens the Menu pop-up.

For more information about Menu, see the **Menu keyword** topic.

Menu is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.36. Move Key

Pressing **Move** enters the Move keyword into the command line.



For more information about Move, see the [Move keyword](#) topic.

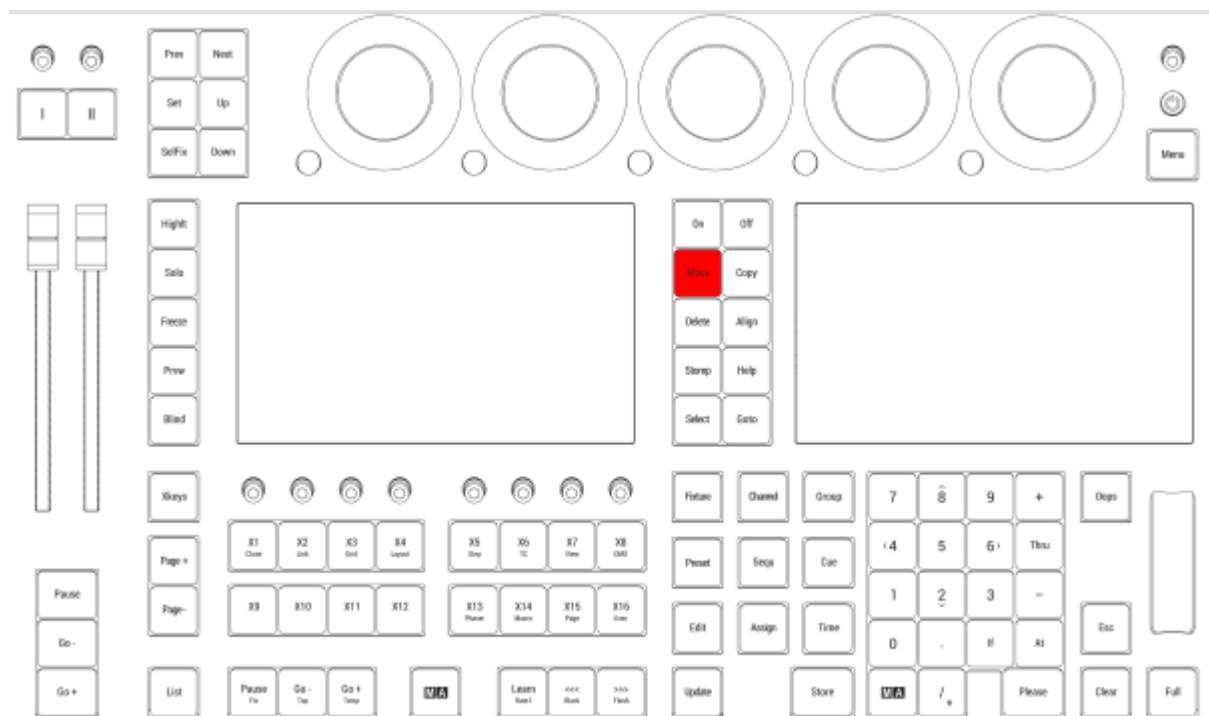
Exchange

Pressing **Move** **Move** enters the Exchange keyword into the command line.

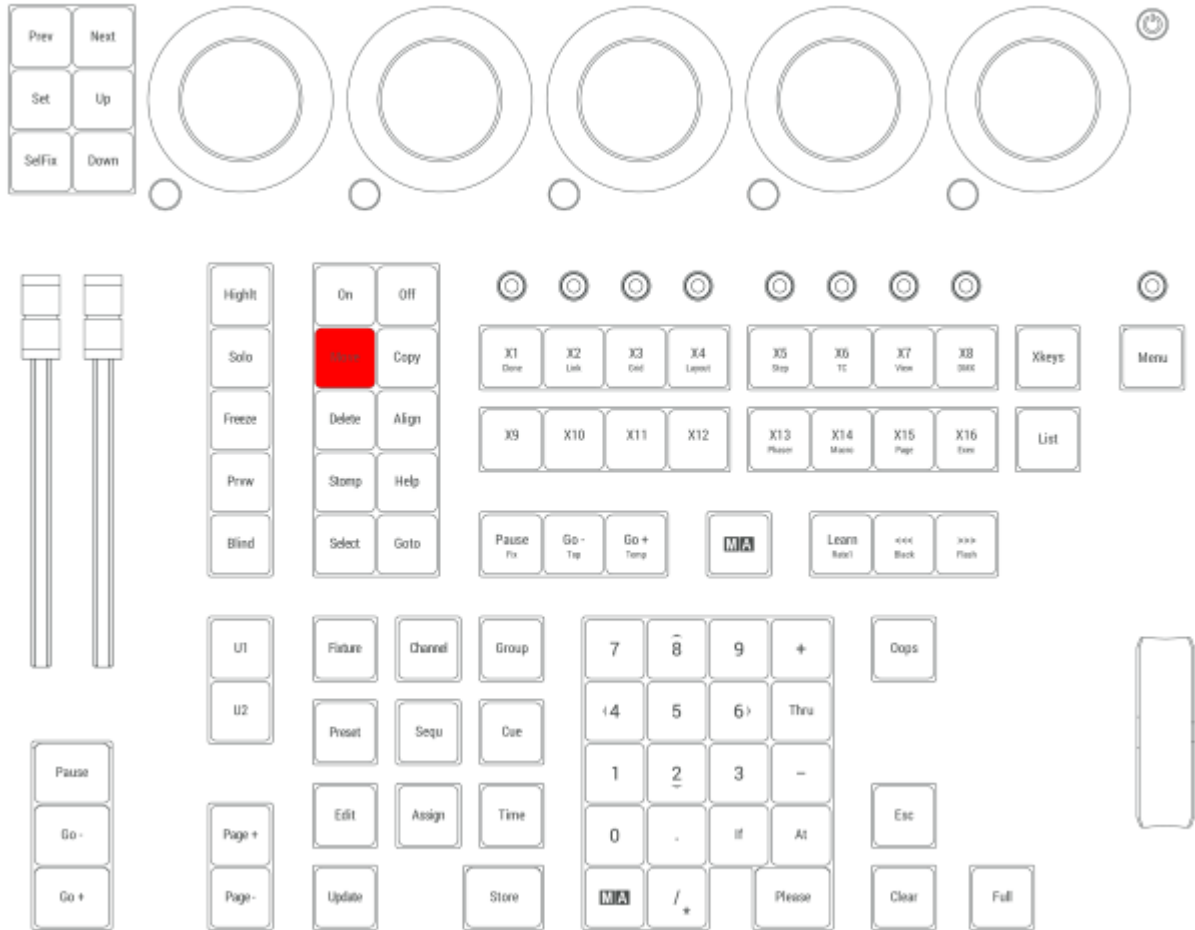


For more information about Exchange, see the [Exchange Keyword](#) topic.

Move is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.37. Next Key

Pressing **Next** executes the Next keyword in the command line.



For more information about Next, see the [Next keyword](#) topic.

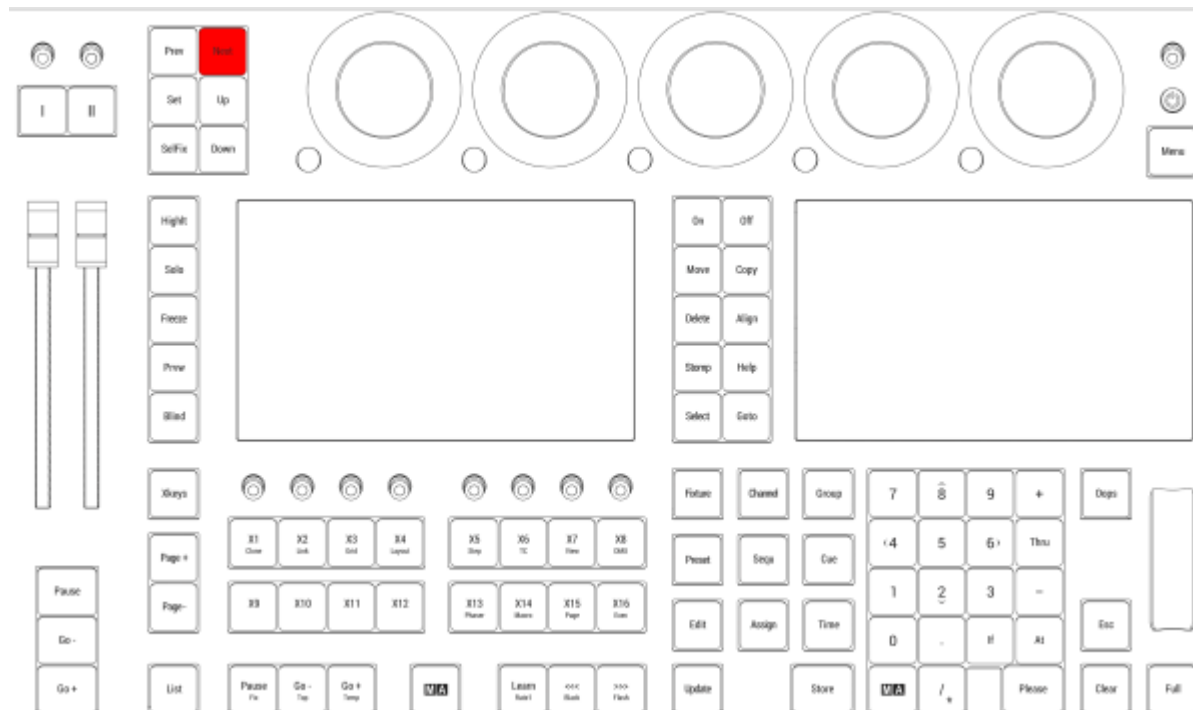
Next Step

Pressing **MA** + **Next** executes the Next Step command in the command line.

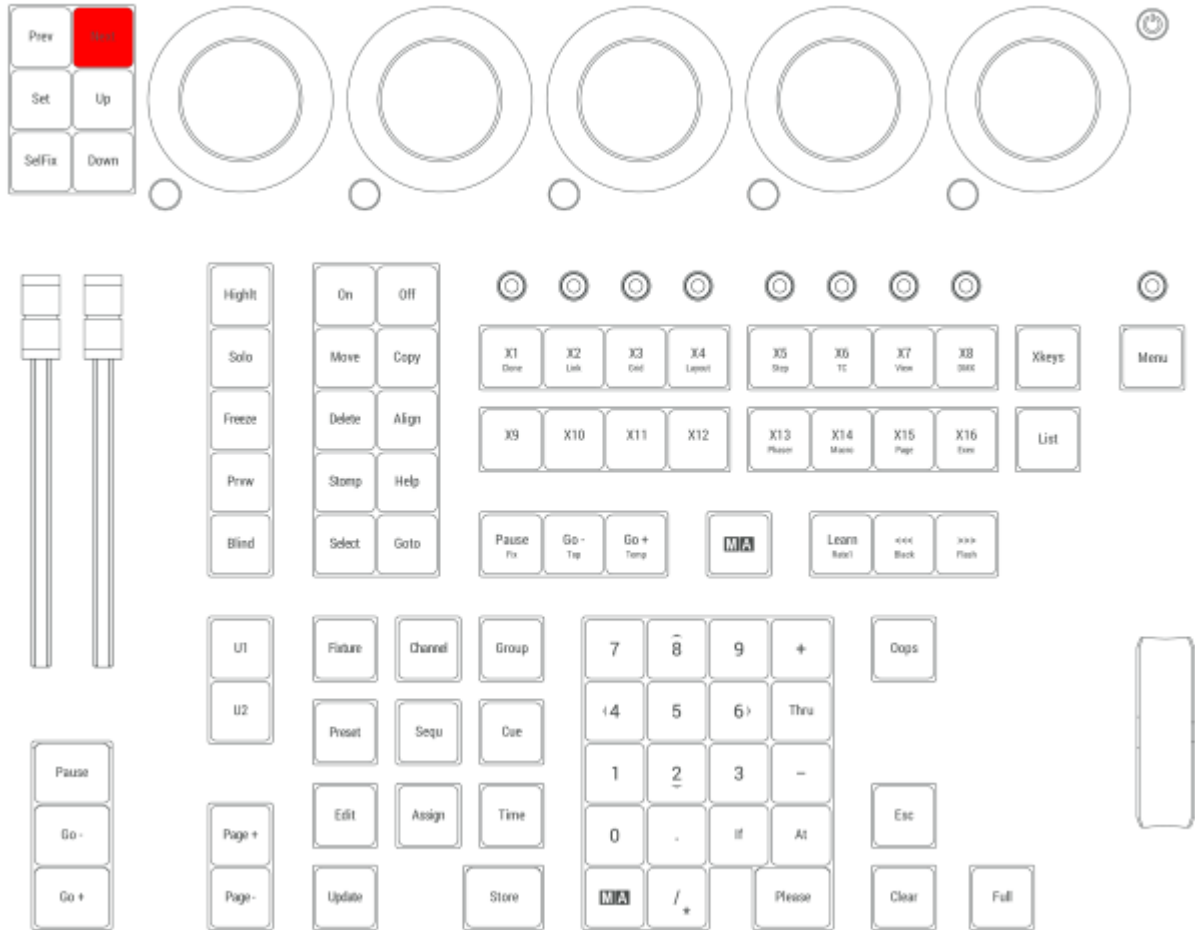


For more information about Next Step, see the [Next keyword](#) topic and [Step keyword](#) topic. Or see the [Phaser Topic](#) for more information about steps.

Next is located in the command section.



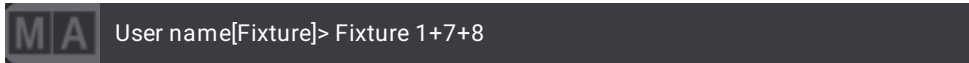
Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.38. Numeric Keys | Arrows

Pressing a numeric key enters the number into the command line.



Move Cursor

Pressing **MA** + **8** move the cursor 1 row up.

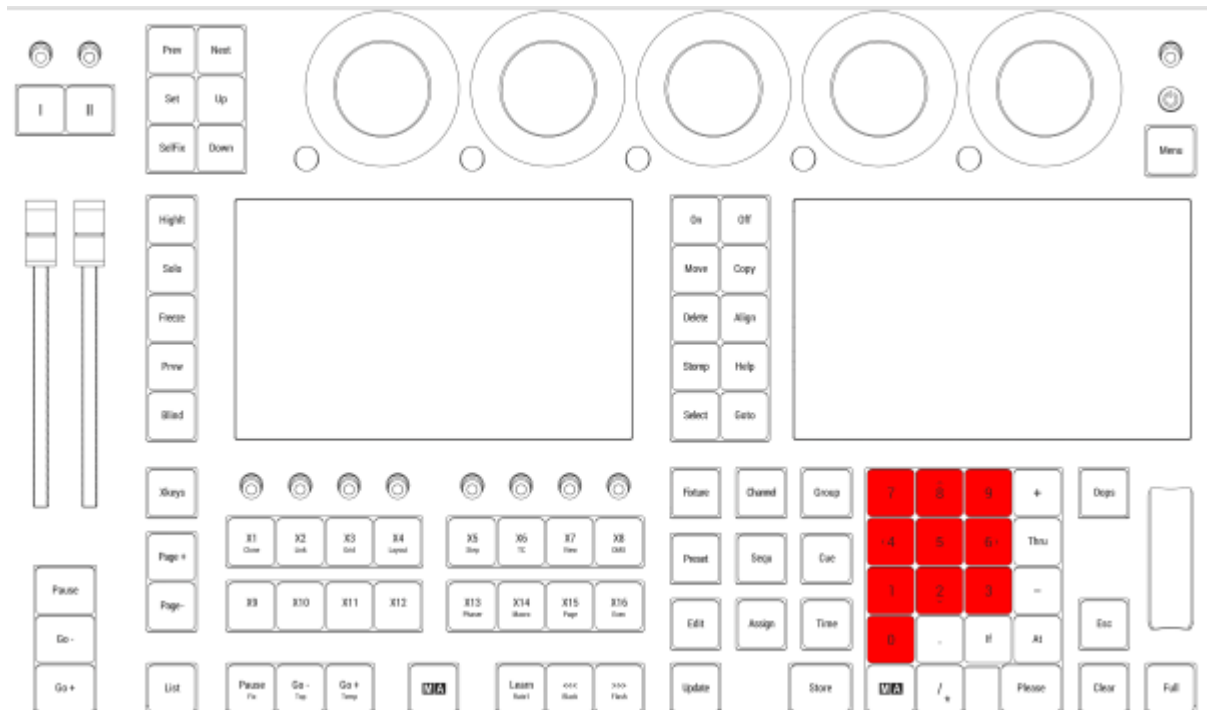
Pressing **MA** + **2** move the cursor 1 row down.

Pressing **MA** + **4** move the cursor 1 column left.

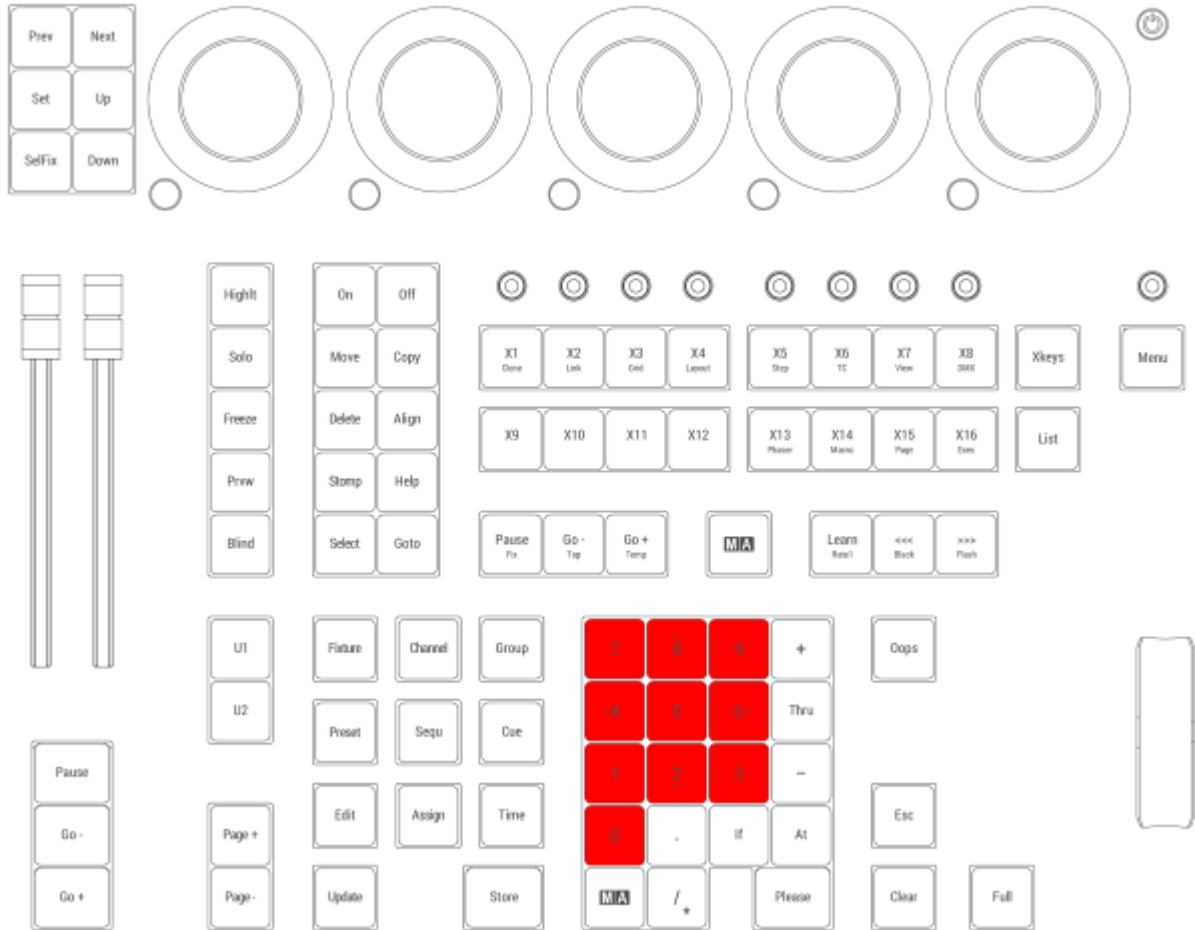
Pressing **MA** + **6** move the cursor 1 column right.

The arrow keys on a keyboard have the same functions.

The numeric keys is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

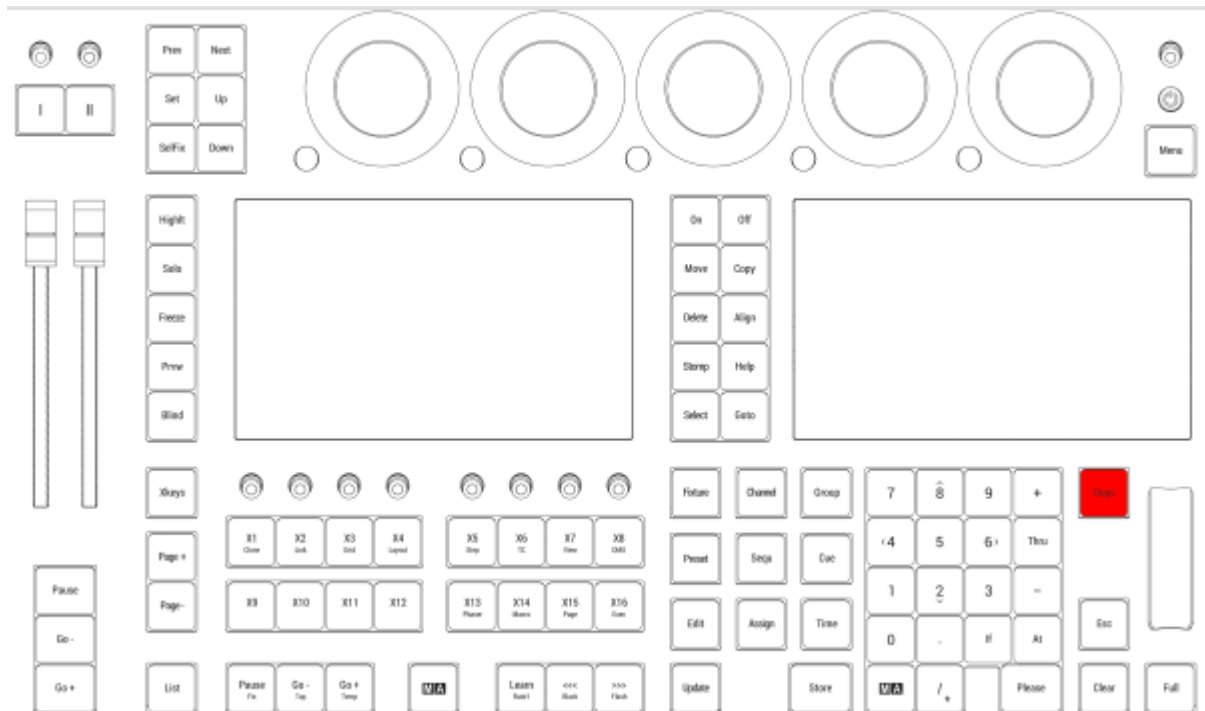
2.15.39. Oops Key

Pressing **Oops** executes the Oops command.

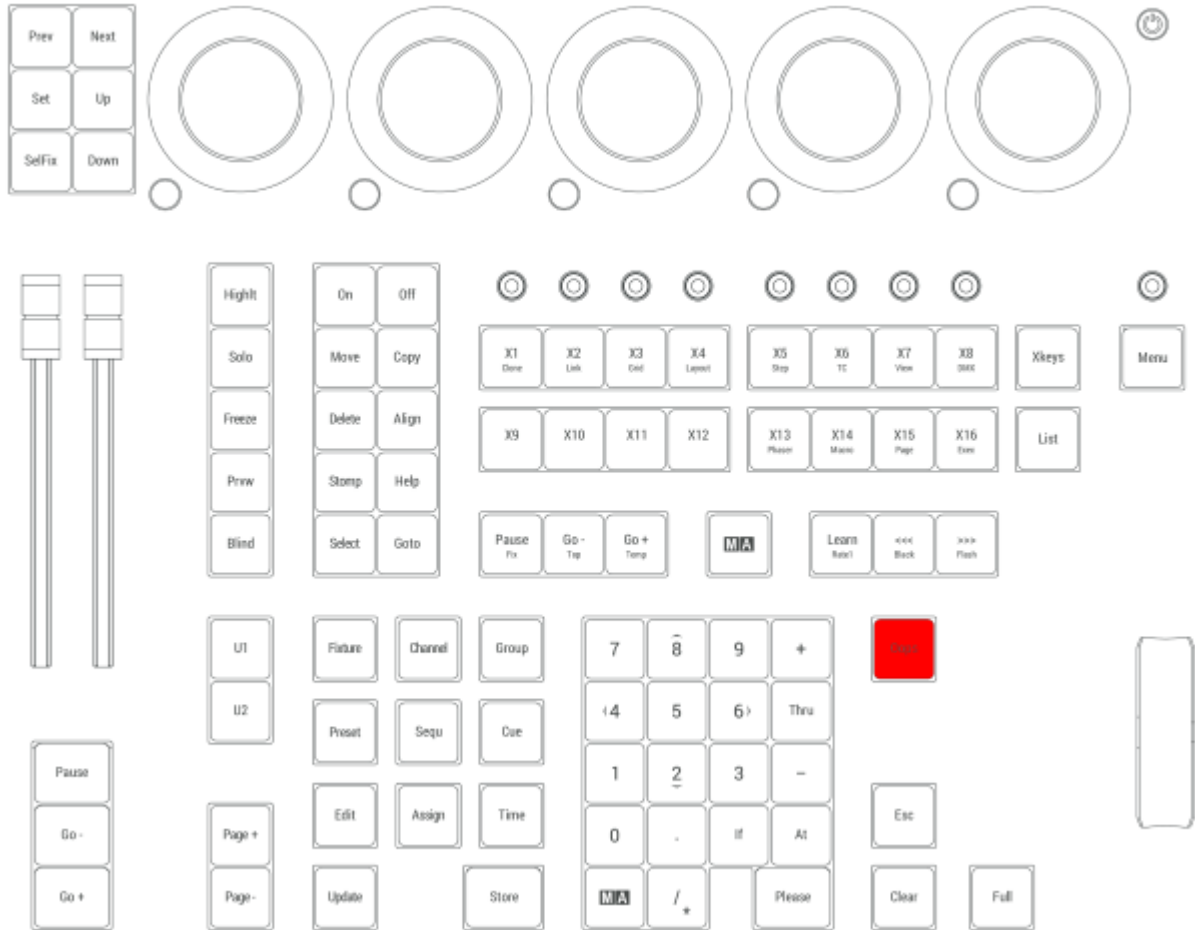
OK:Oops

For more information about Oops, see the [Oops keyword](#) topic.

Oops is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.40. On Key

Pressing **On** enters the On keyword into the command line.



For more information about On, see the **On keyword** topic.

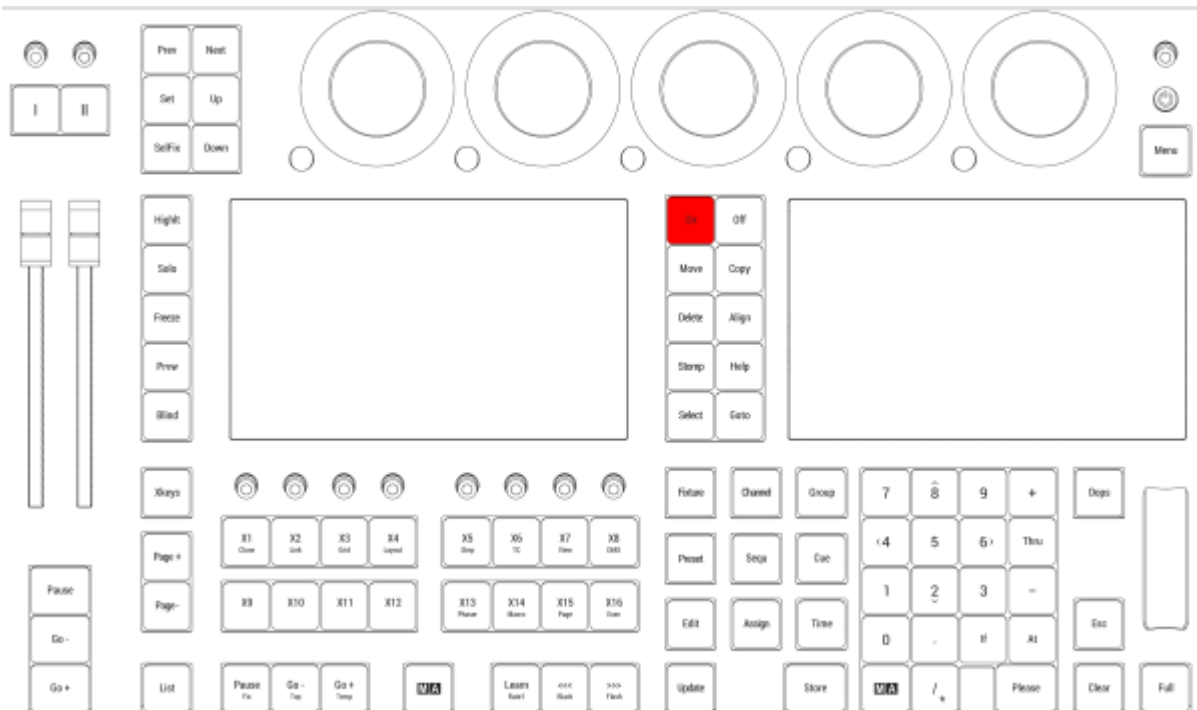
Call

Pressing **On** **On** enters the Call keyword into the command line.

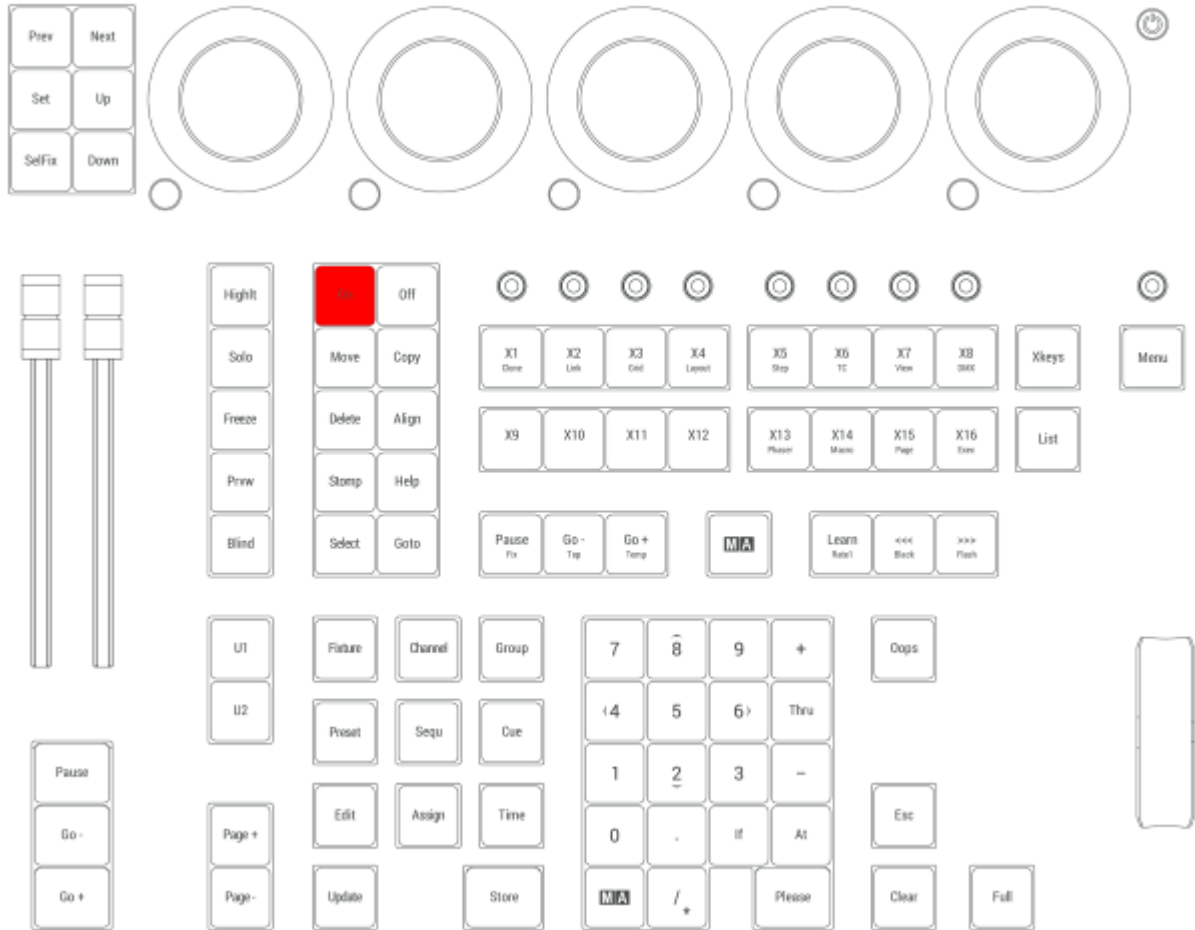


For more information about Call, see the **Call keyword** topic.

On is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.41. Off Key

Pressing **Off** enters the Off keyword into the command line.



For more information about Off, see the **Off keyword** topic.

Off Menu

Pressing **Off** **Off** opens the Off menu.

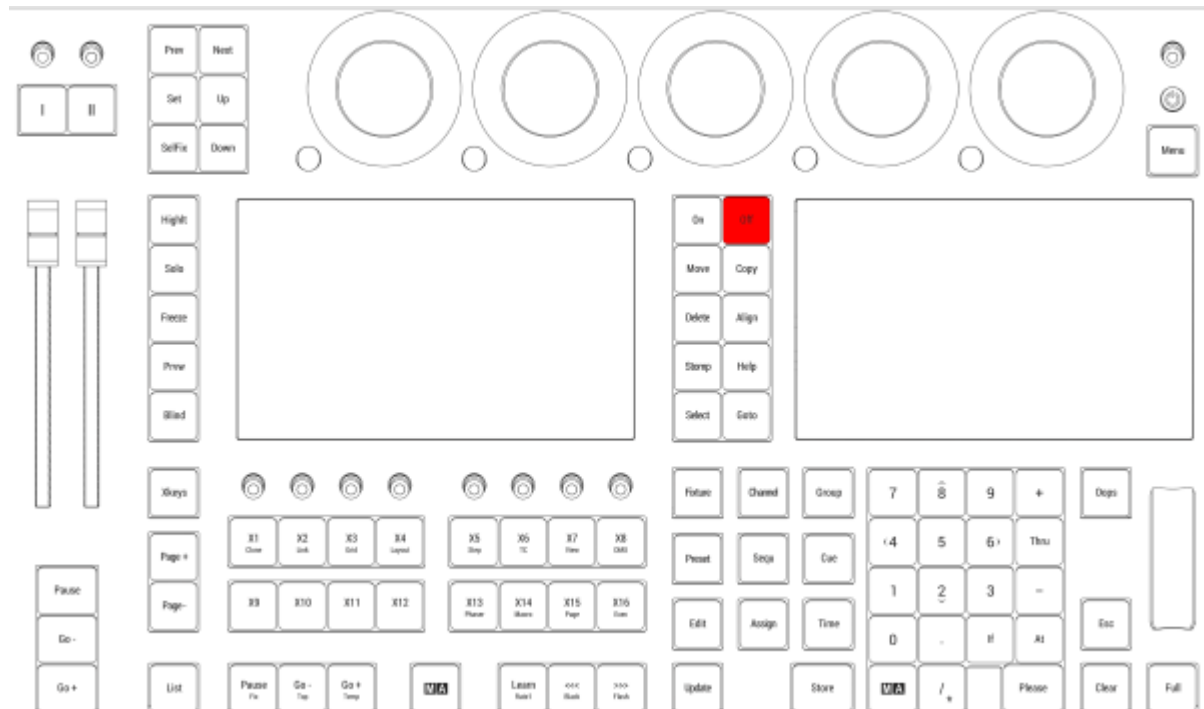
Read more about the Off menu [here](#).

Close RemoteHID Connection

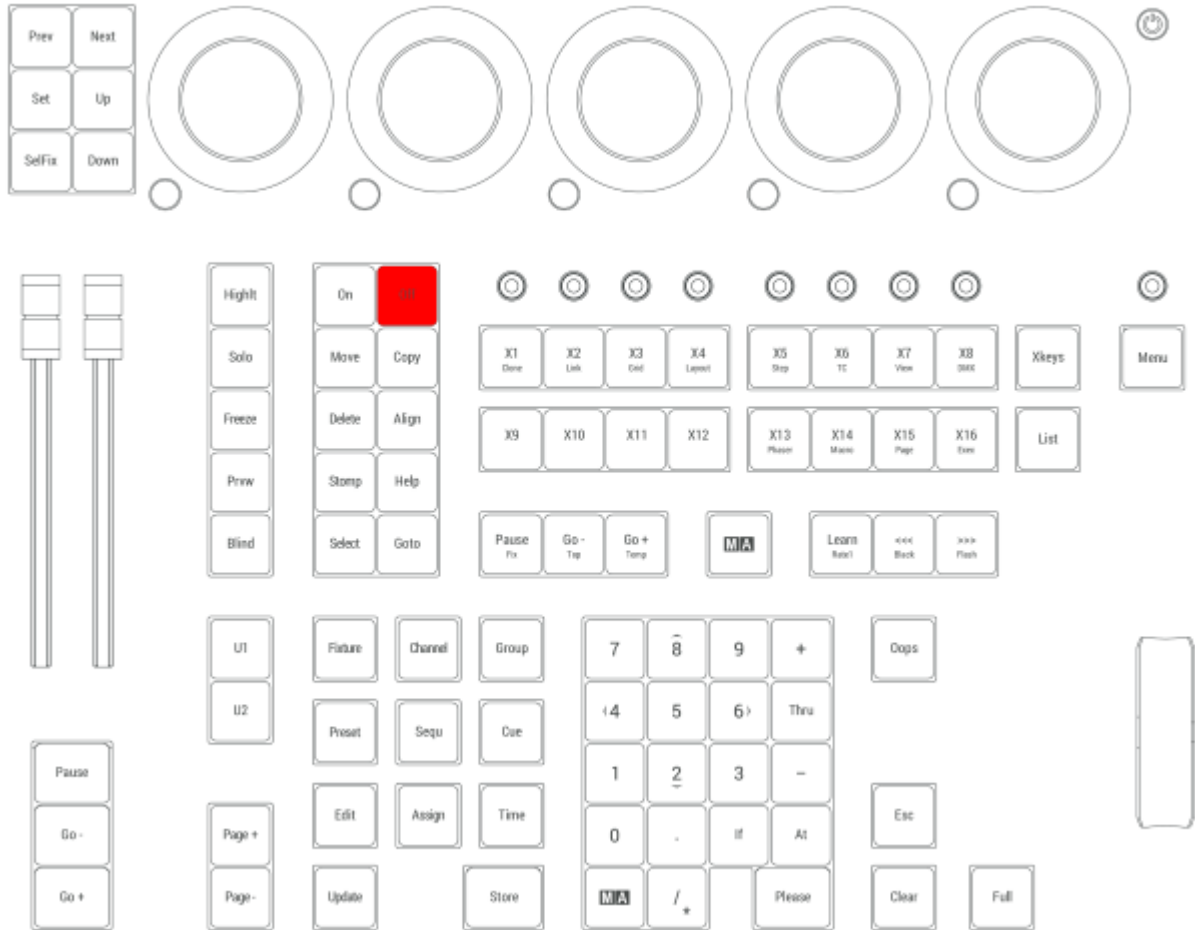
Pressing **MA** + **MA** + **Off** closes any active RemoteHID connections.

For more information about RemoteHID connections, see the **RemoteHID** topic.

Off is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

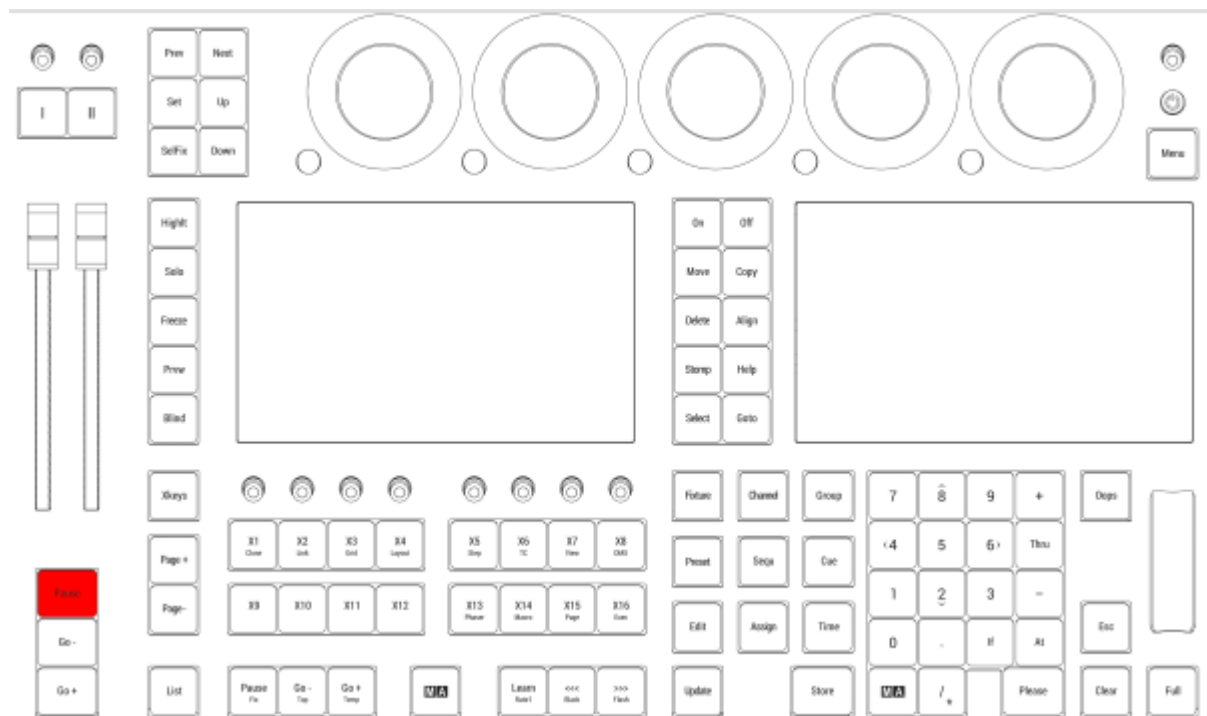
2.15.42. Pause [large] Key

Pressing **Pause** [large] executes the Pause Executor keyword in the command line

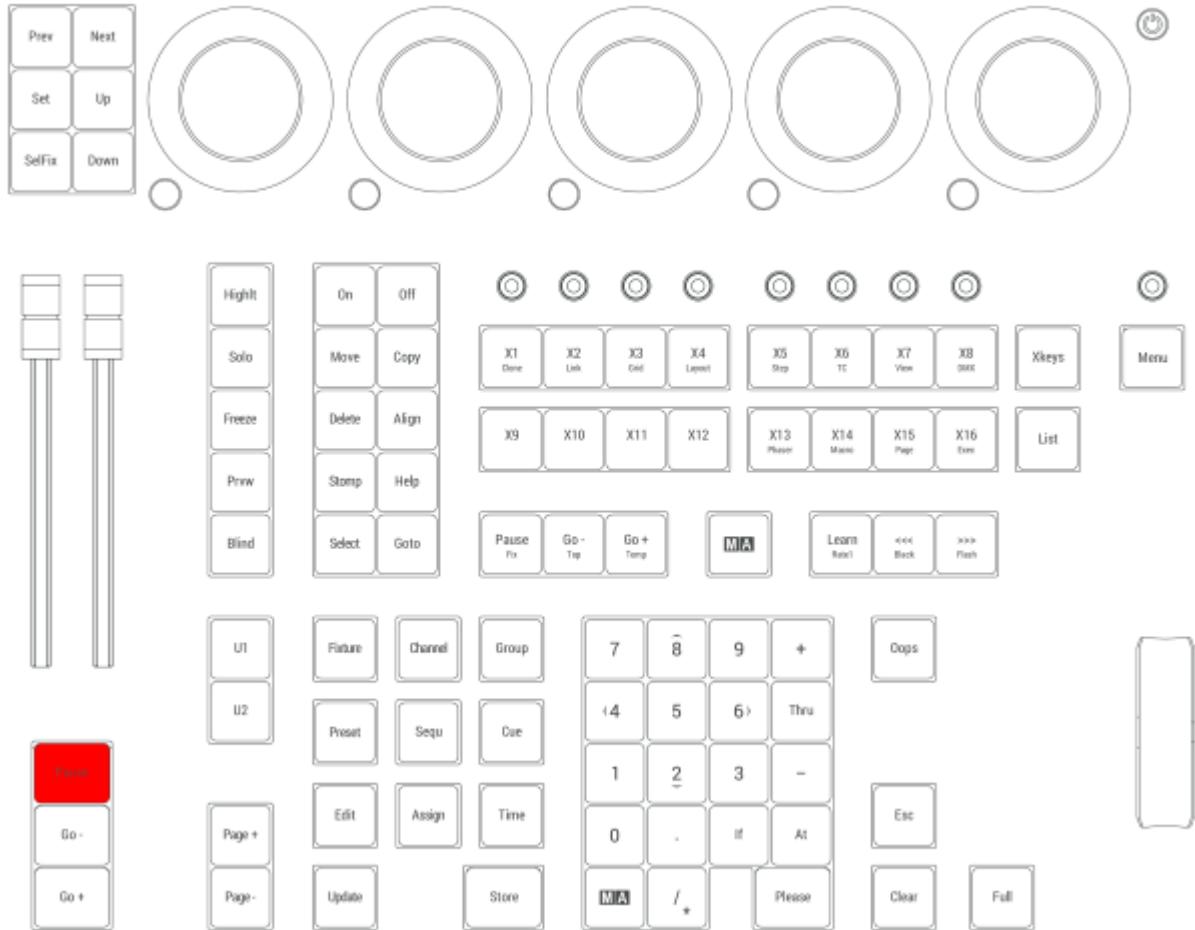


For more information about the Pause, see the **Pause keyword** topic.

Pause [large] is located in the master section under the two master faders.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.43. Pause | Fix Key

Pressing **Pause** enters the Pause keyword into the command line.

```
MA User name[Fixture]> Pause
```

For more information about Pause, see the [Pause keyword](#) topic.

Pause

Pressing **Pause** **Pause** enters the Park keyword into the command line.

```
MA User name[Fixture]> Park Fixture 17
```

For more information about Park, see the [Park keyword](#) topic.

Fix

Pressing and holding **MA** + **Pause** enters the Fix keyword into the command line.

```
MA User name[Fixture]> Fix
```

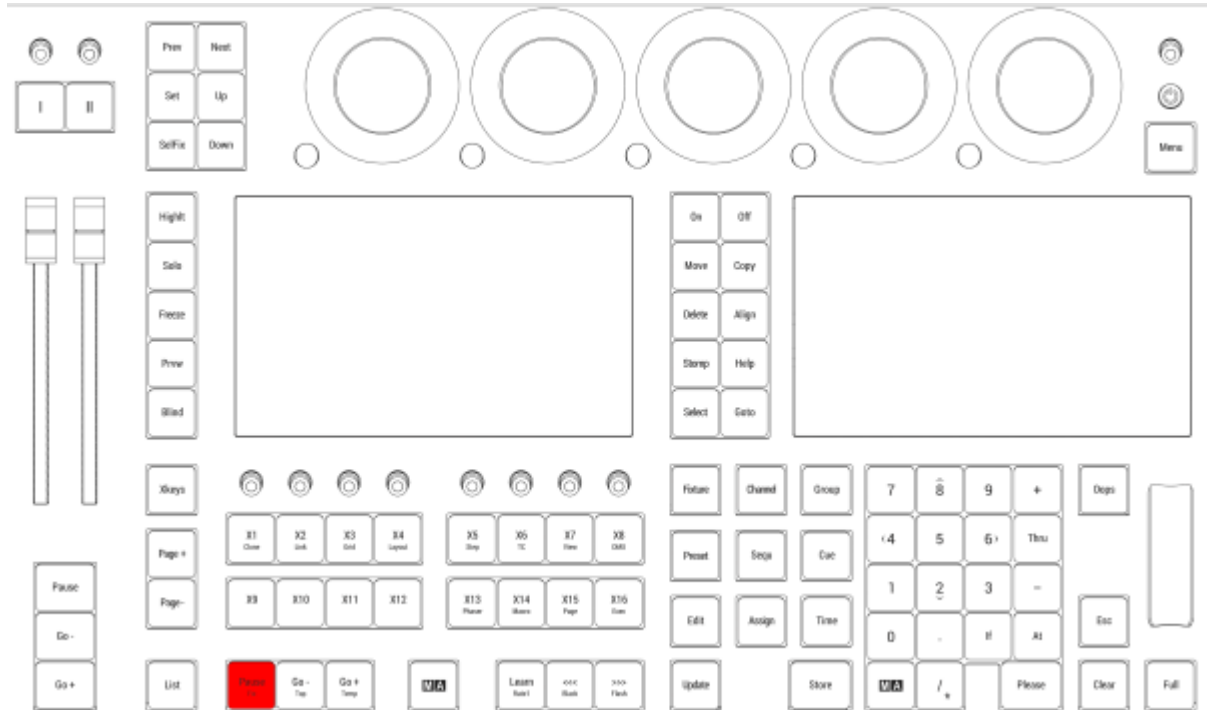
For more information about Fix, see the [Fix keyword](#) topic.

Desk Lock

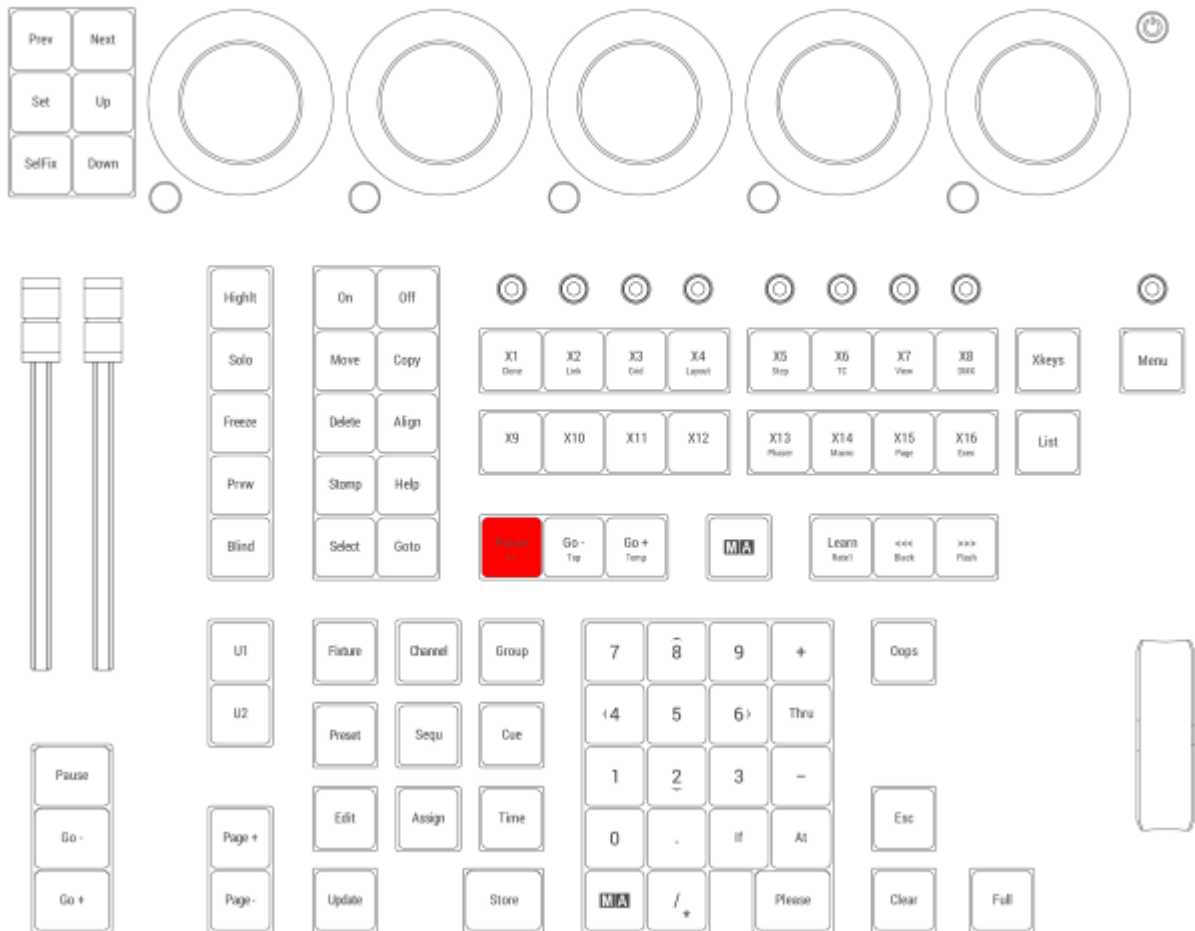
Pressing and holding **MA** + **MA** + **Pause** toggles the desk lock.

For more information about the desk lock, see the [Desk Lock](#) topic.

Pause is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



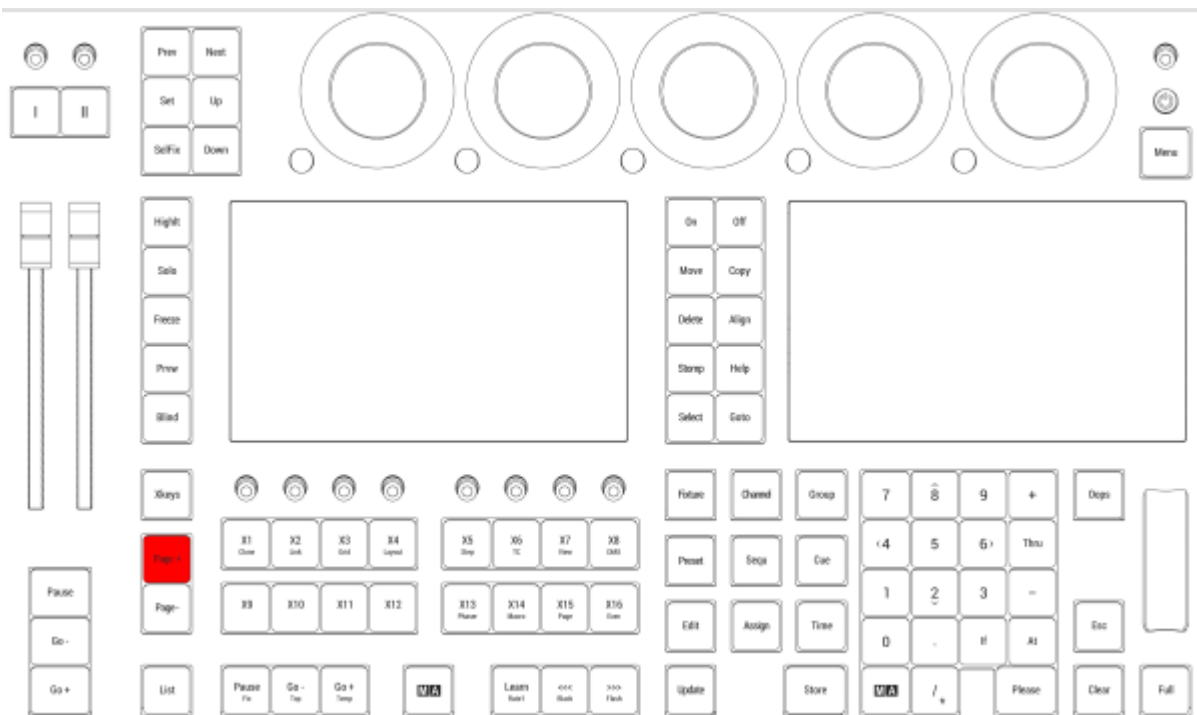
Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.44. Page+ Key

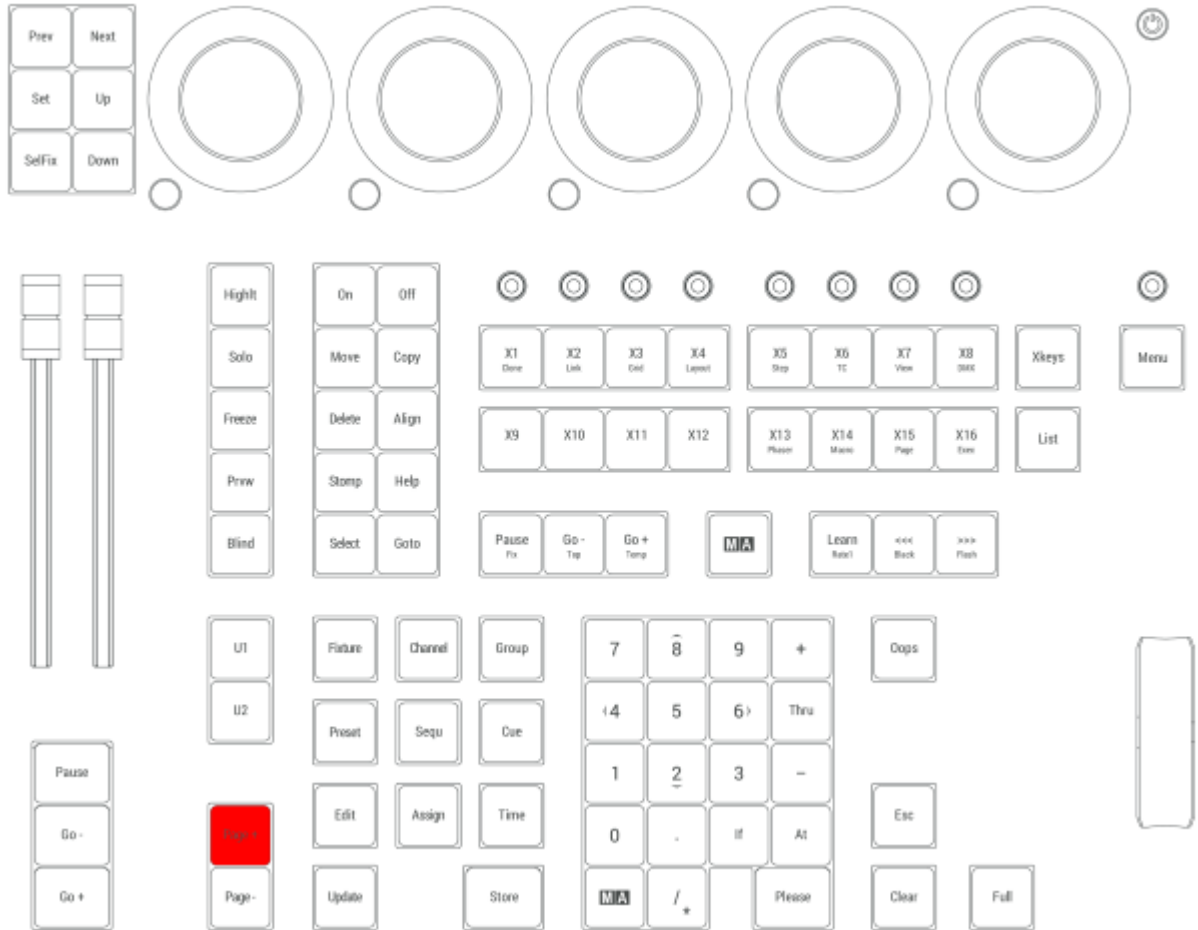
Pressing **Page+** executes the command Next Page in the command line.



Page+ is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



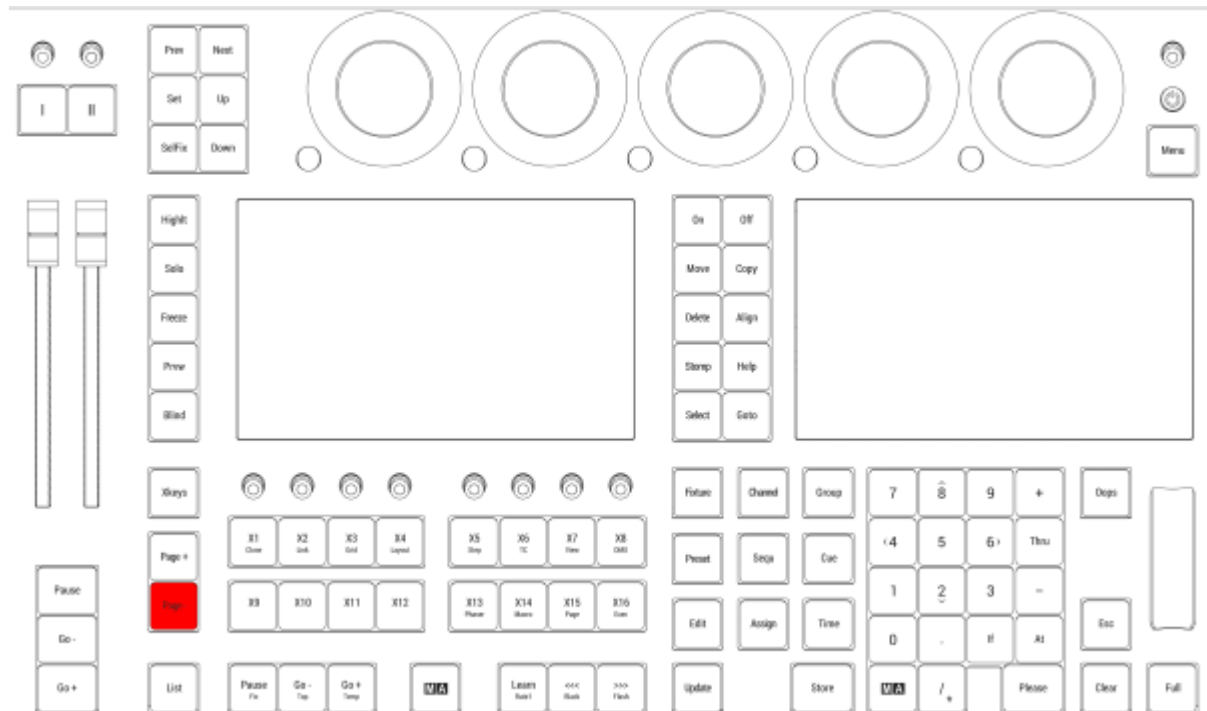
Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.45. Page- Key

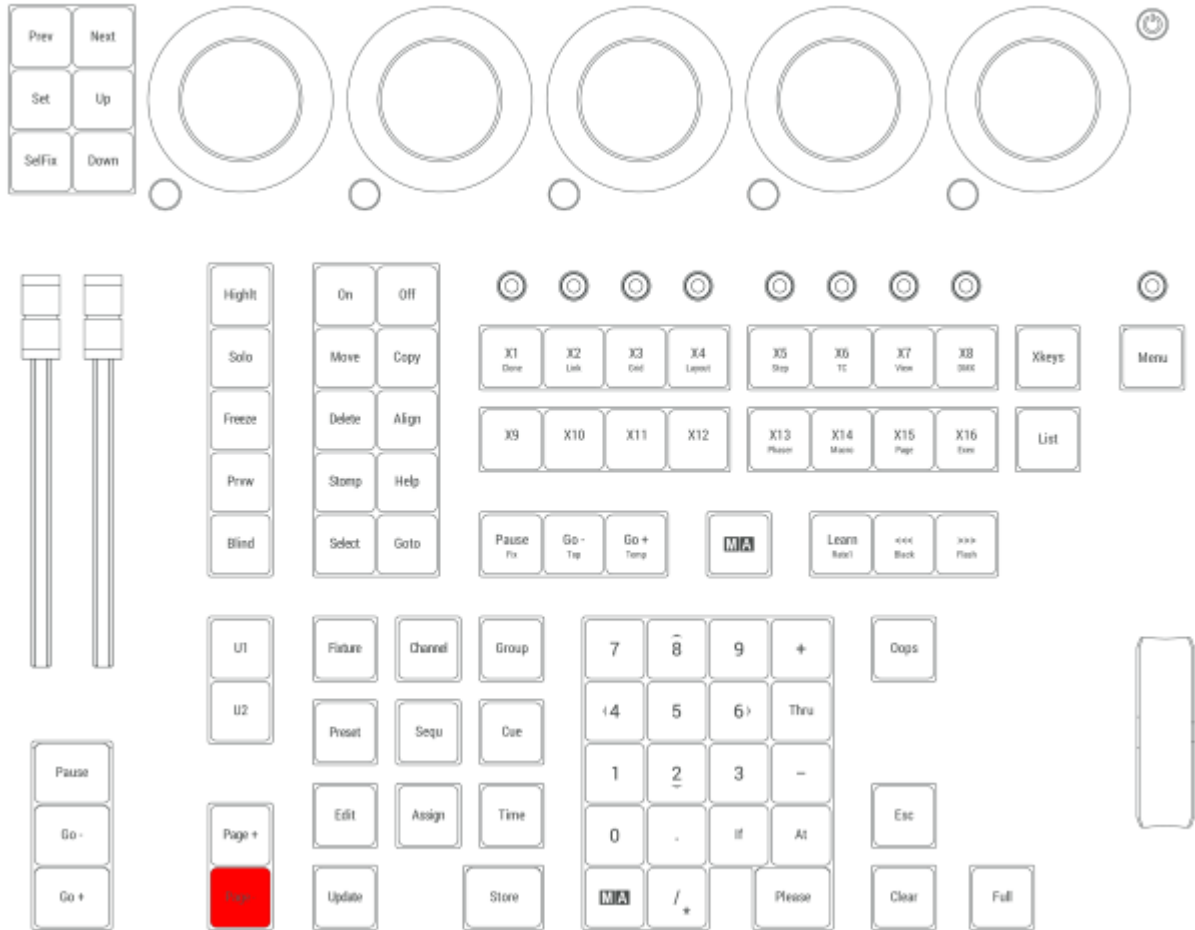
Pressing **Page-** executes the command Previous Page in the command line.

Pressing and holding **Page-** for more than 1 second sets the page to page 1.

Page- is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

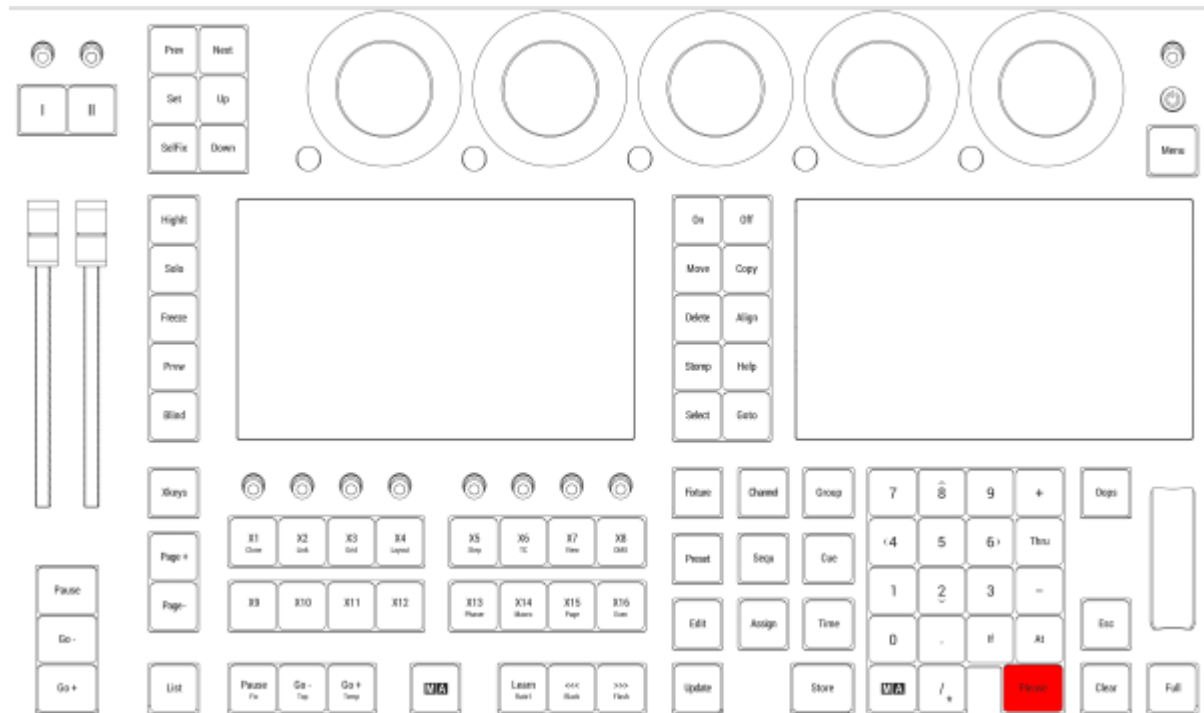
2.15.46. Please Key

Pressing **Please** executes the command in the command line. This is the same function as pressing **Enter** on an external keyboard.

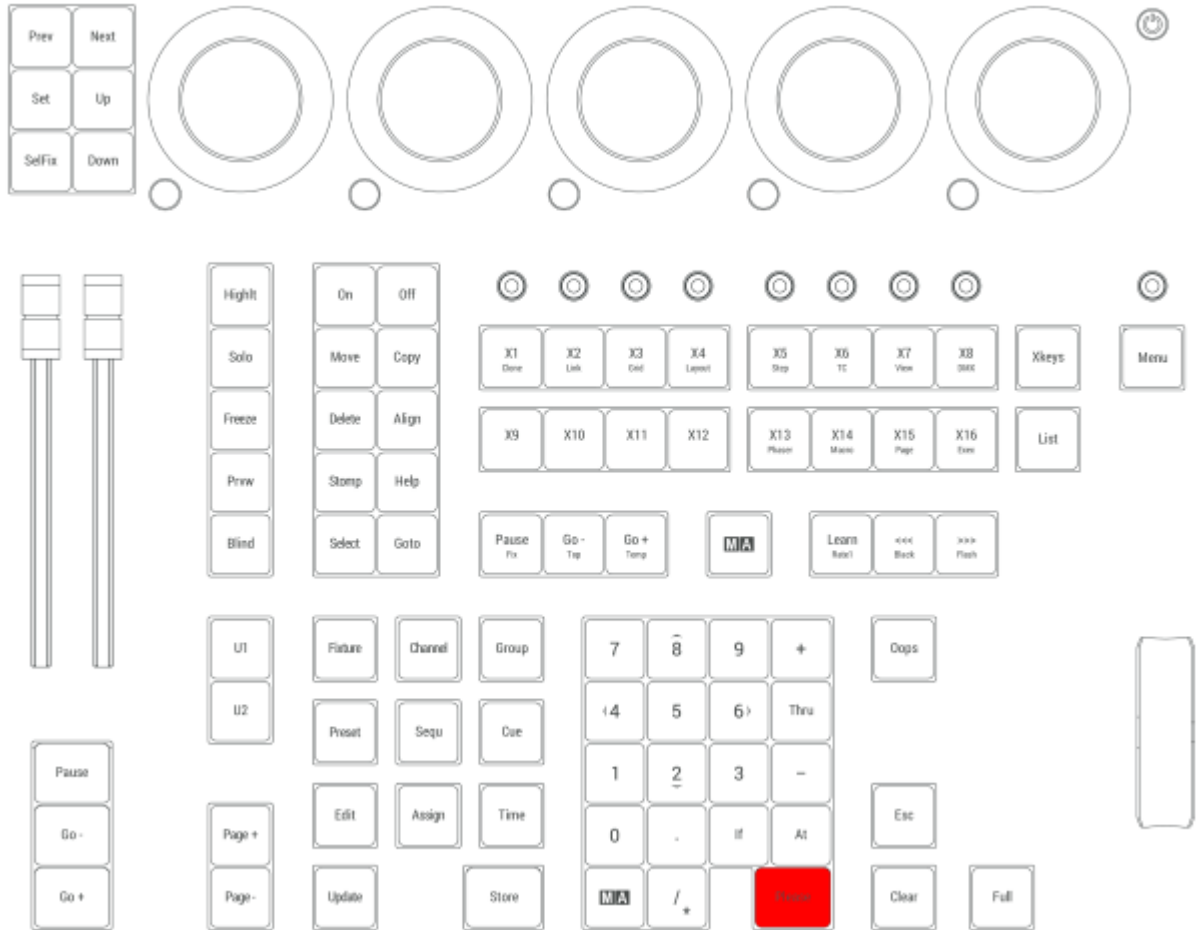
Activate all Attributes

- Pressing **Please Please** activates all attributes of the selected fixtures.
- Pressing **Please** once more deactivates all attributes of the selected fixtures.

Please is located in the numeric keys section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.47. Power Key

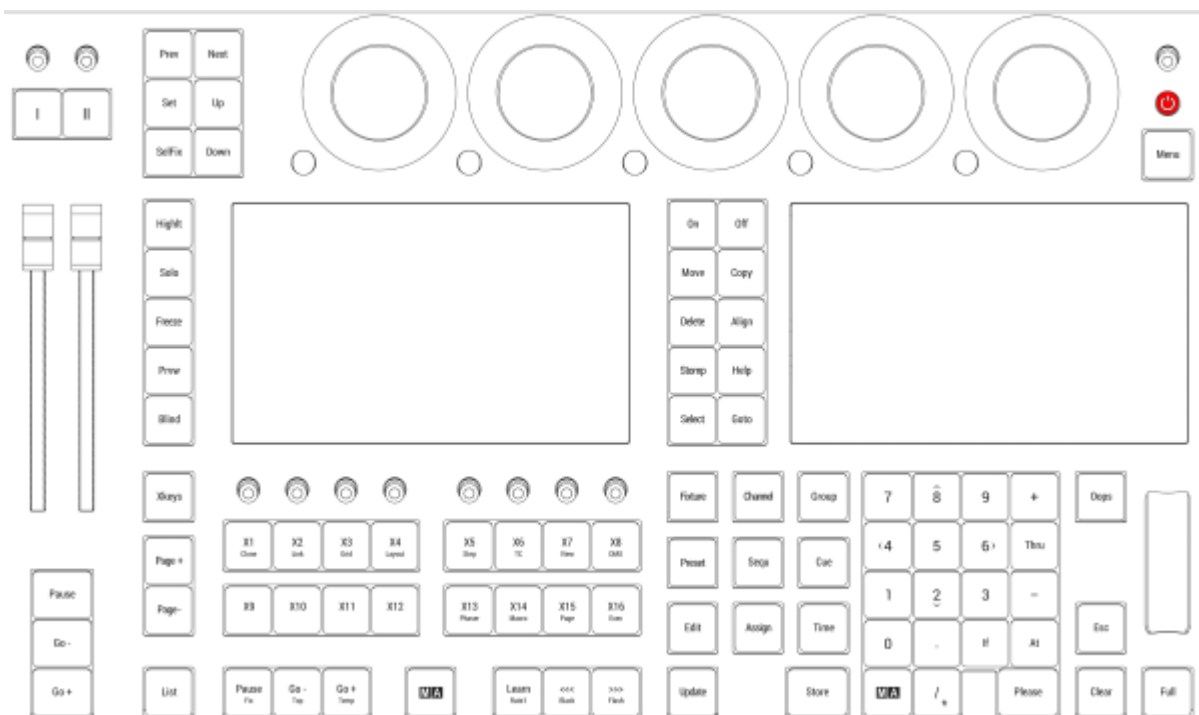
Pressing **Power** boots up the console or shuts it down.



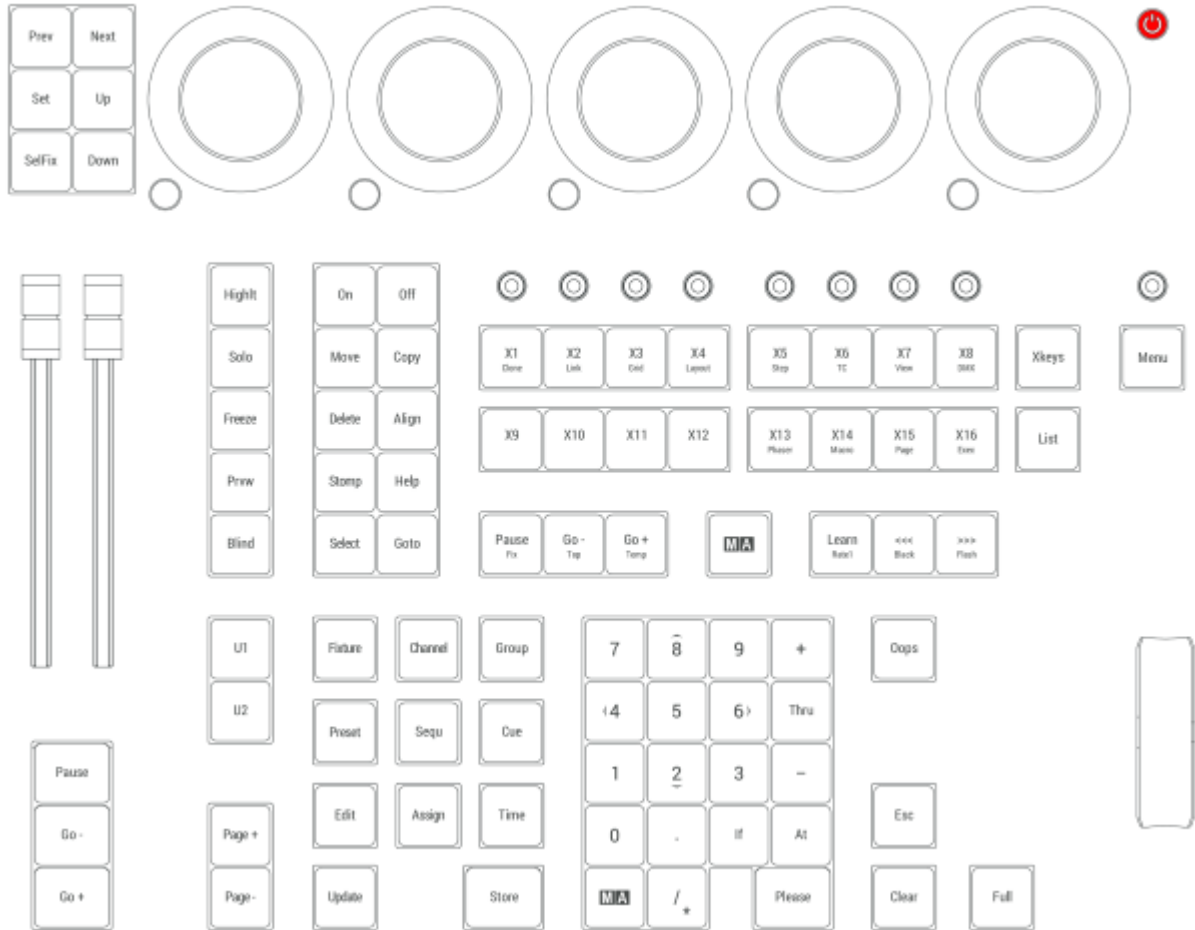
Important:

To switch the console on or off, switch the power switch on the rear panel.

Power is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.48. Preset Key

Pressing **Preset** enters the Preset keyword into the command line.



For more information about Preset, see the [Preset keyword](#) topic.

Attribute

Pressing **Preset** **Preset** enters the Attribute keyword into the command line.



For more information about Attribute, see the [Attribute keyword](#) topic.

Gel

Pressing **Preset** **Preset** **Preset** enters the Gel keyword into the command line.



For more information about Gel, see the [Gel keyword](#) topic.

FeatureGroup

Pressing **MA** + **Preset** enters the FeatureGroup keyword into the command line.



For more information about FeatureGroup, see the [FeatureGroup keyword](#) topic.

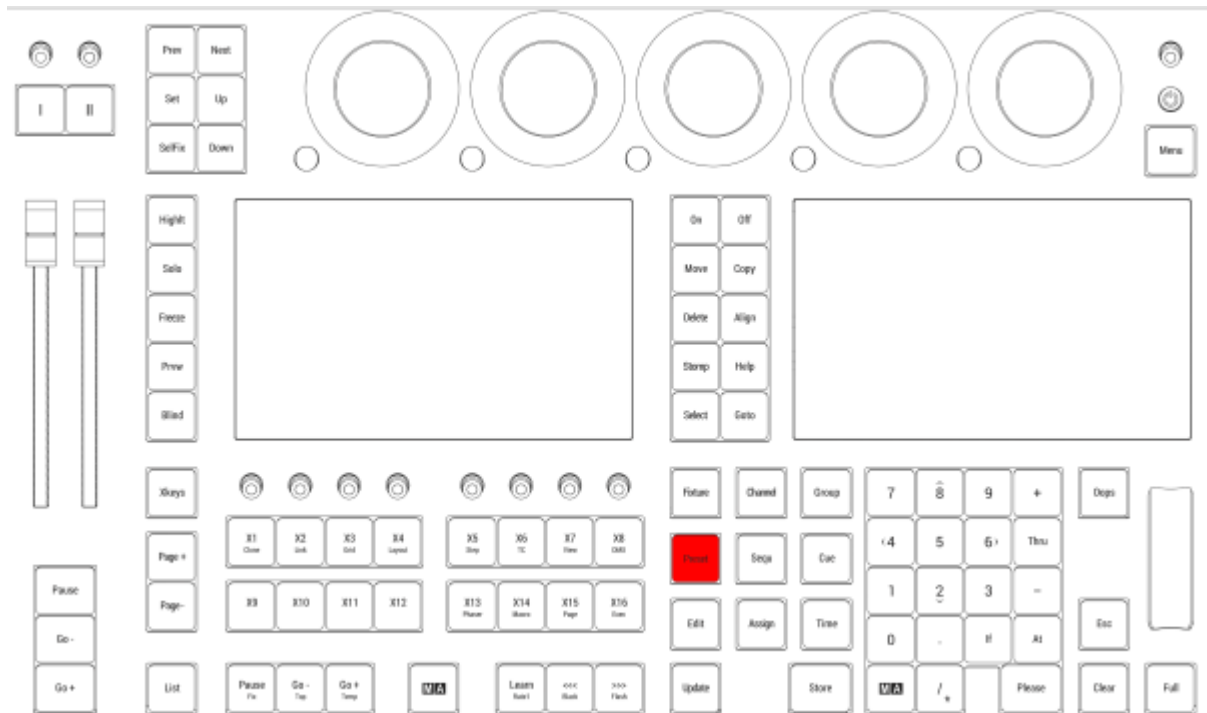
DataPool

Pressing **MA** + **Preset** + **Preset** enters the DataPool keyword into the command line.

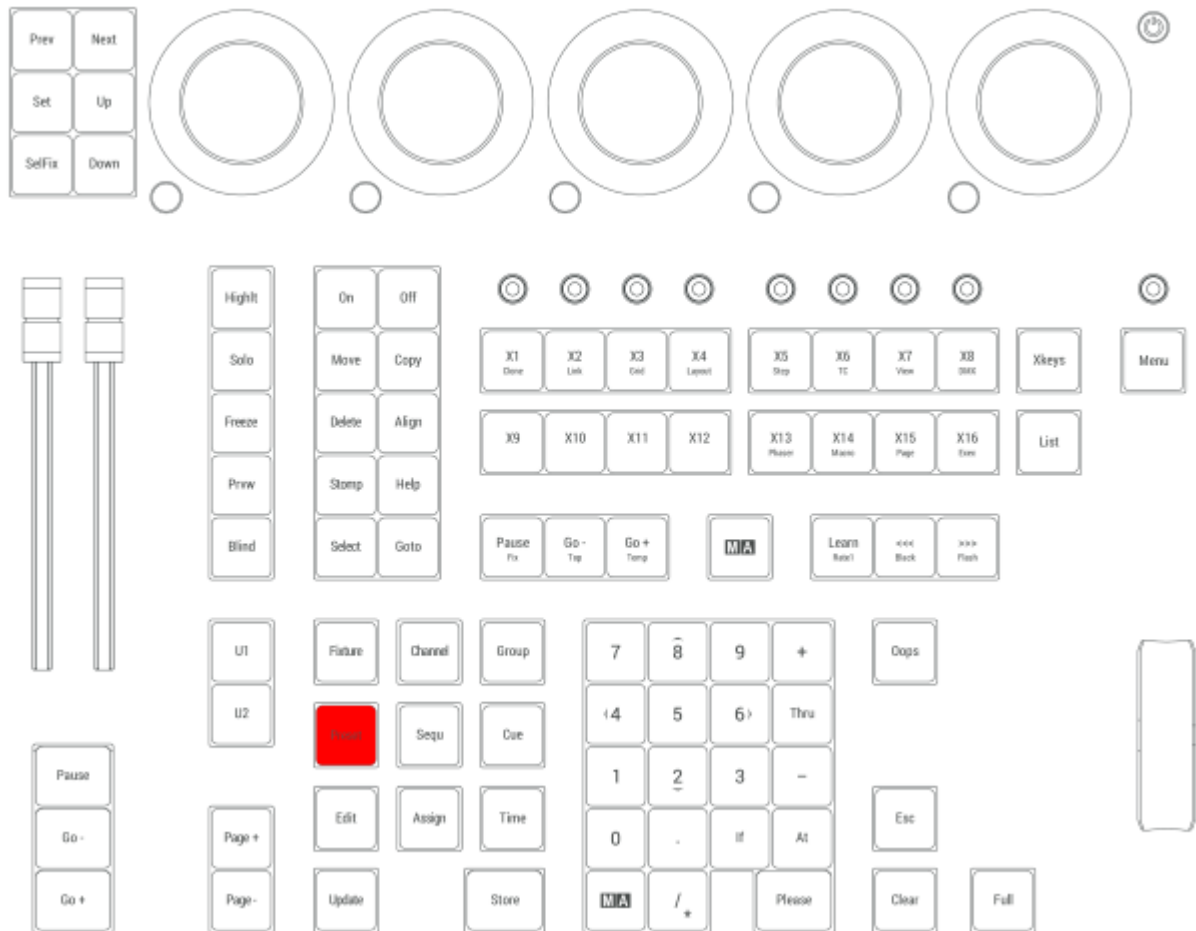


For more information about DataPool, see the [DataPool keyword](#) topic.

Preset is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



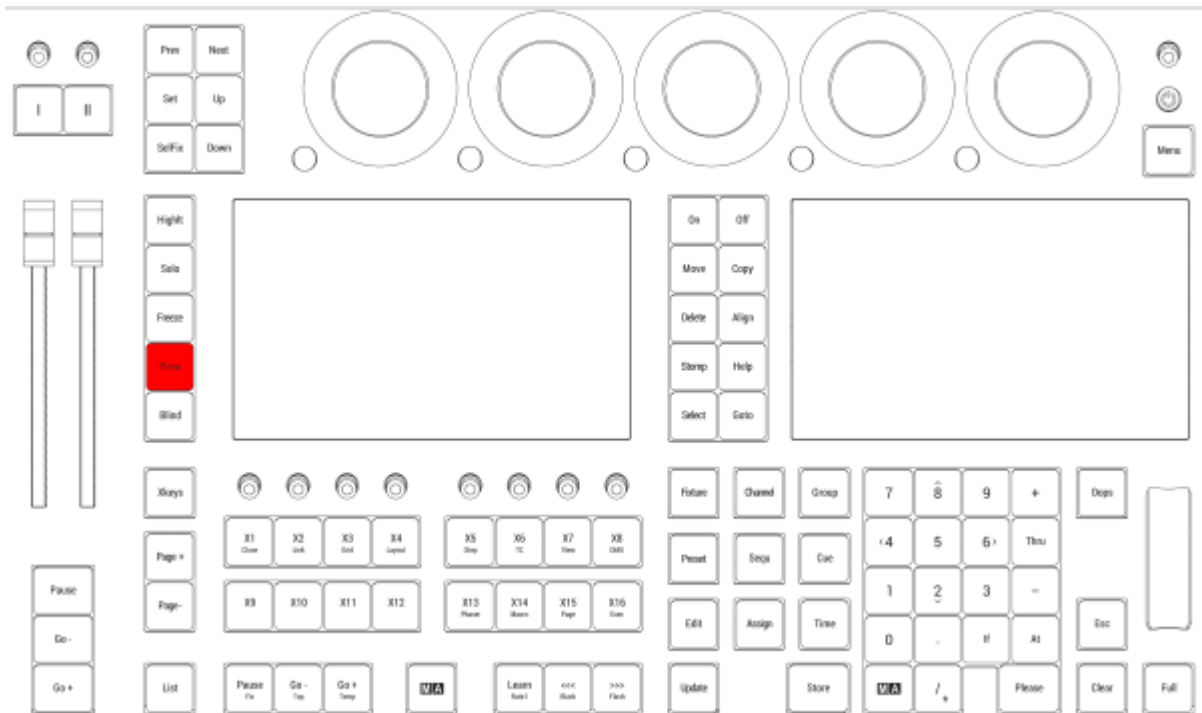
2.15.49. Prvw [Preview] Key

Pressing **Prvw** executes the the Preview keyword in the command line.

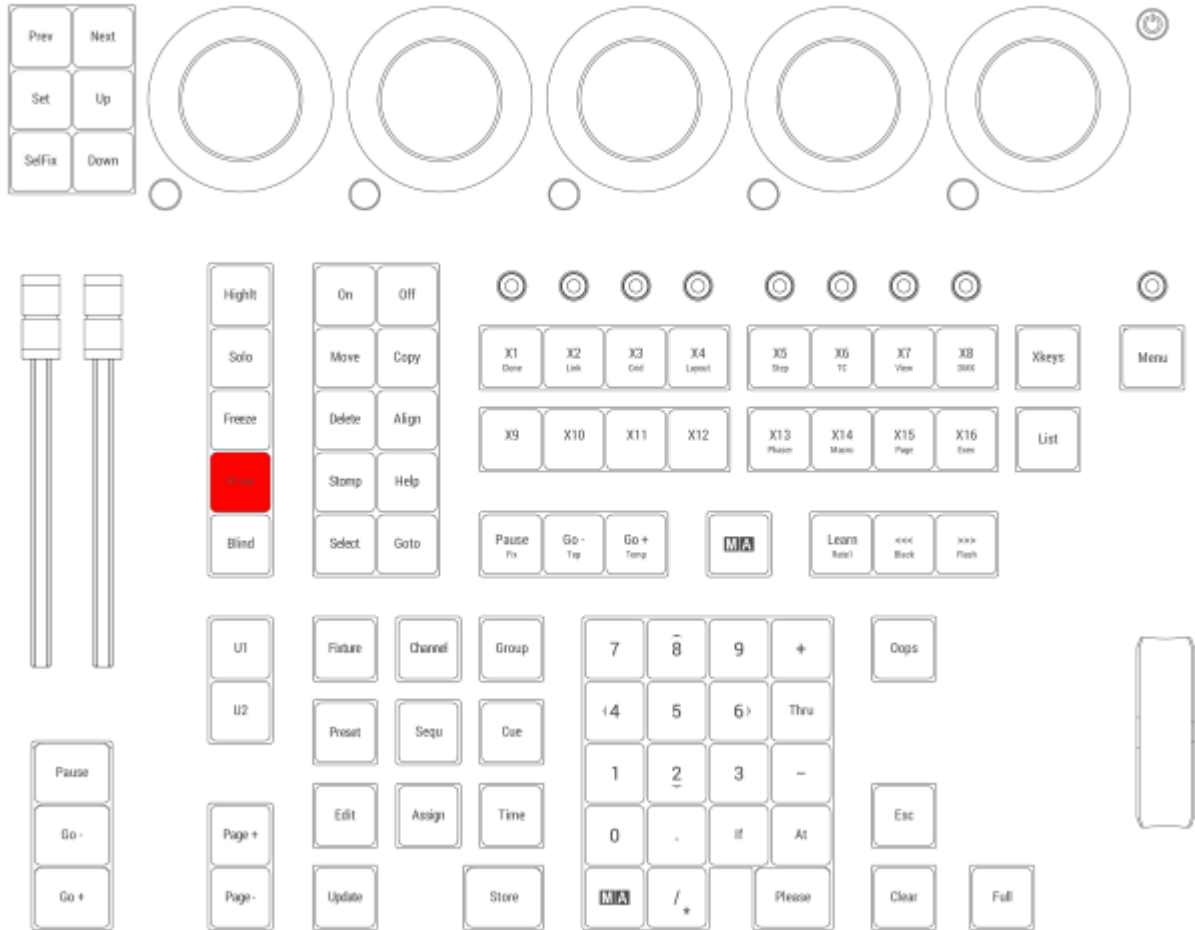


For more information about Prvw, see the [Preview keyword](#) topic.

Prvw is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.50. Prev [Previous] Key

Pressing **Prev** executes the Previous keyword in the command line.



For more information about Previous, see the [Previous keyword](#) topic.

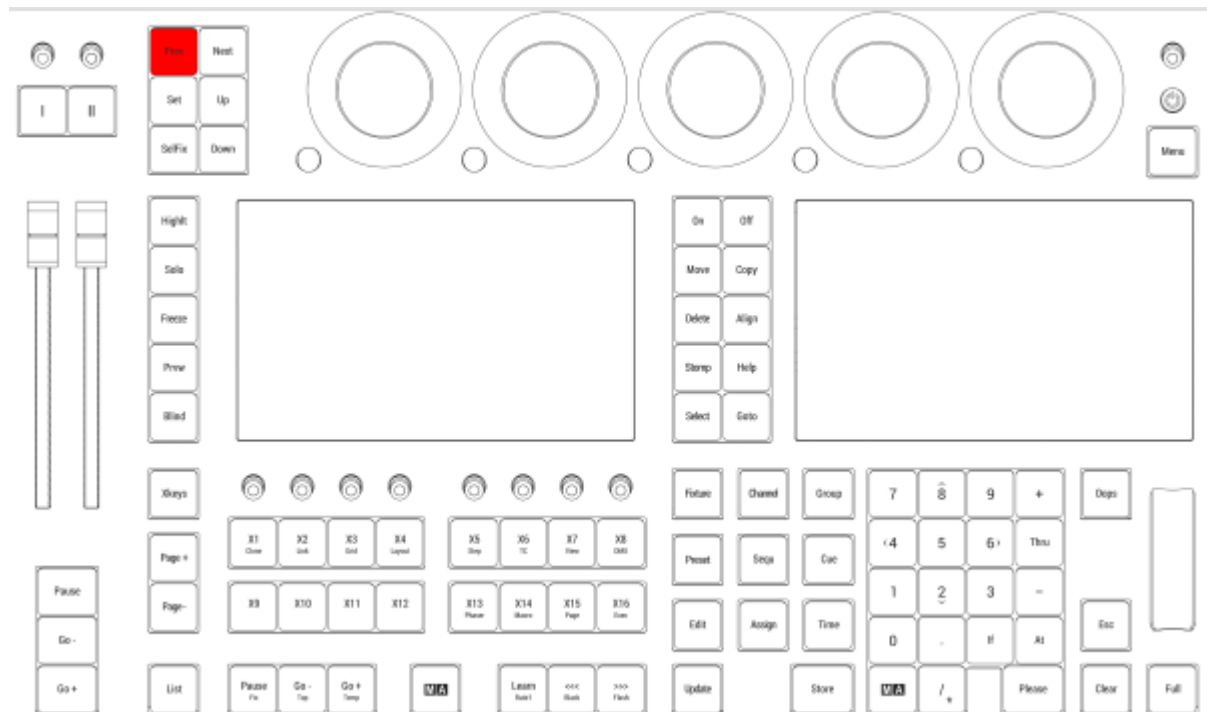
Previous Step

Pressing **MA** + **Prev** executes the Previous Step command in the command line.

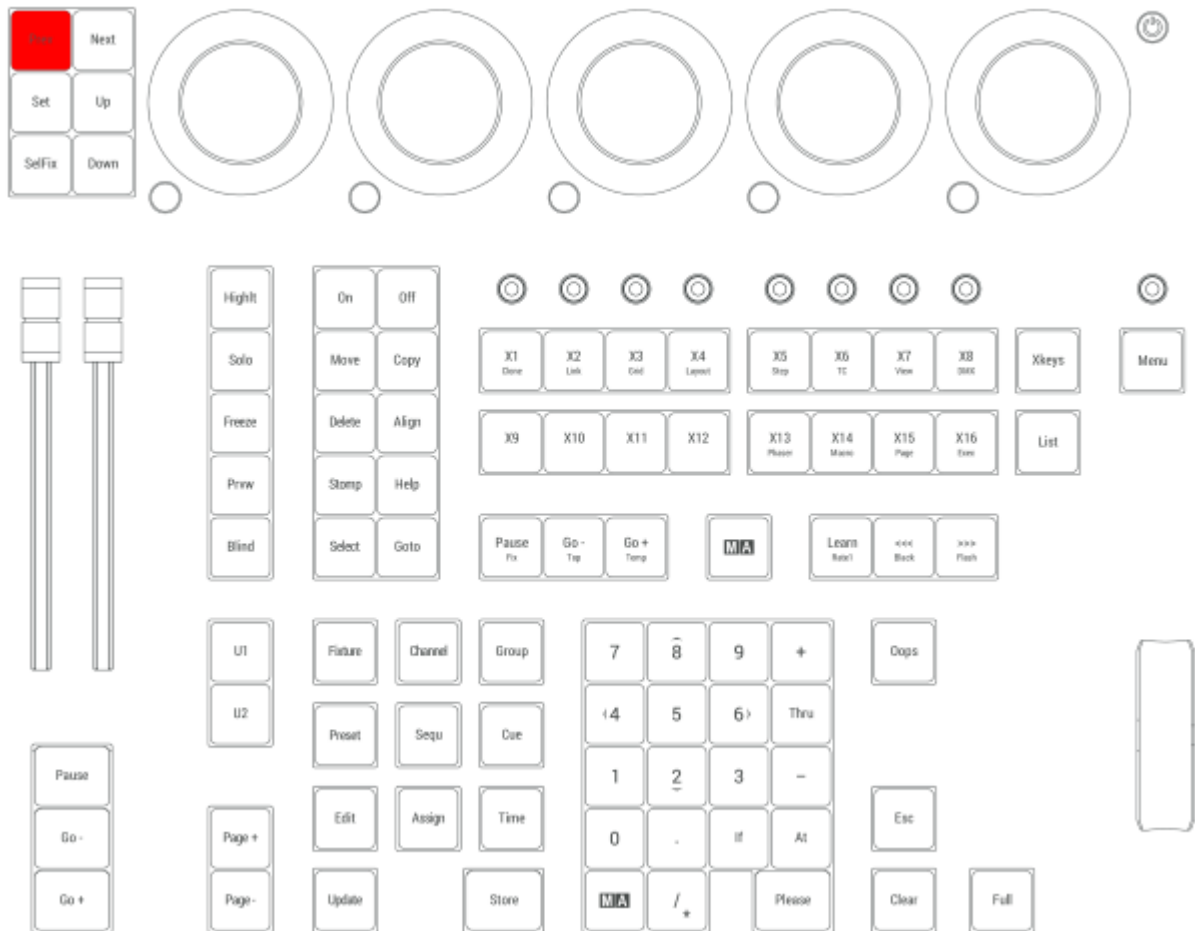


For more information about Previous Step, see the [Previous keyword](#) topic and [Step keyword](#) topic. Or see the [Phaser Topic](#) for more information about steps.

Prev is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



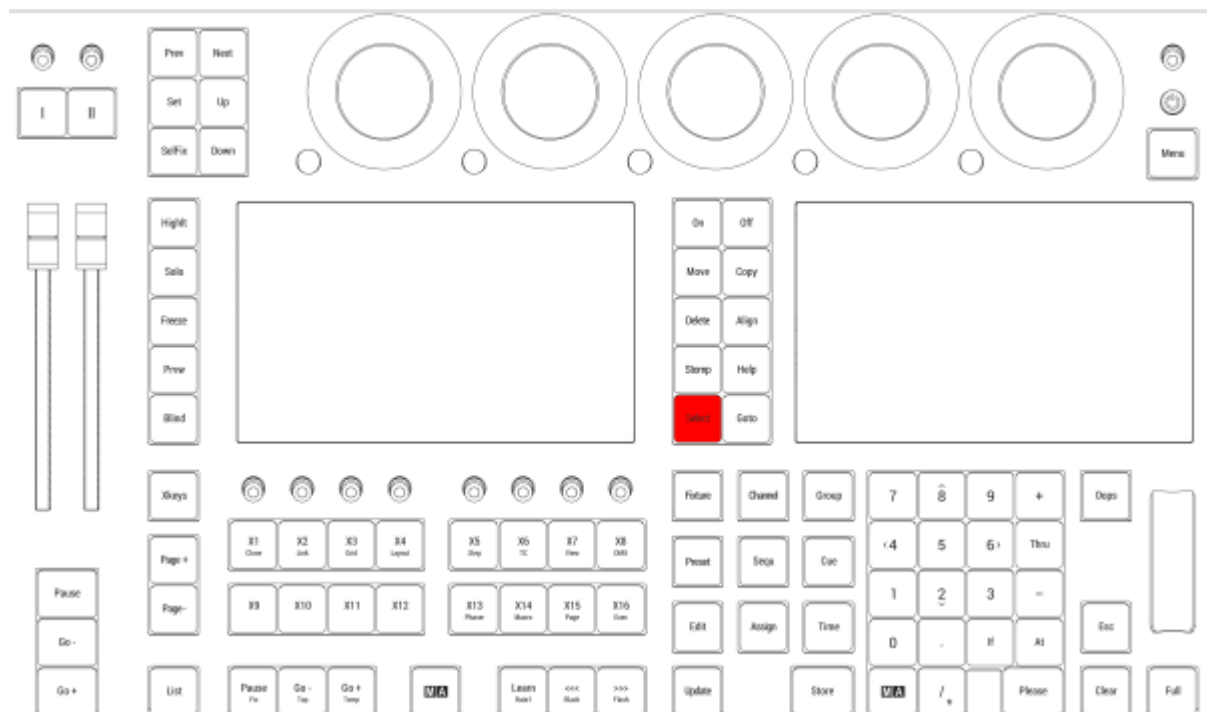
2.15.51. Select Key

Pressing **Select** enters the Select keyword into the command line.

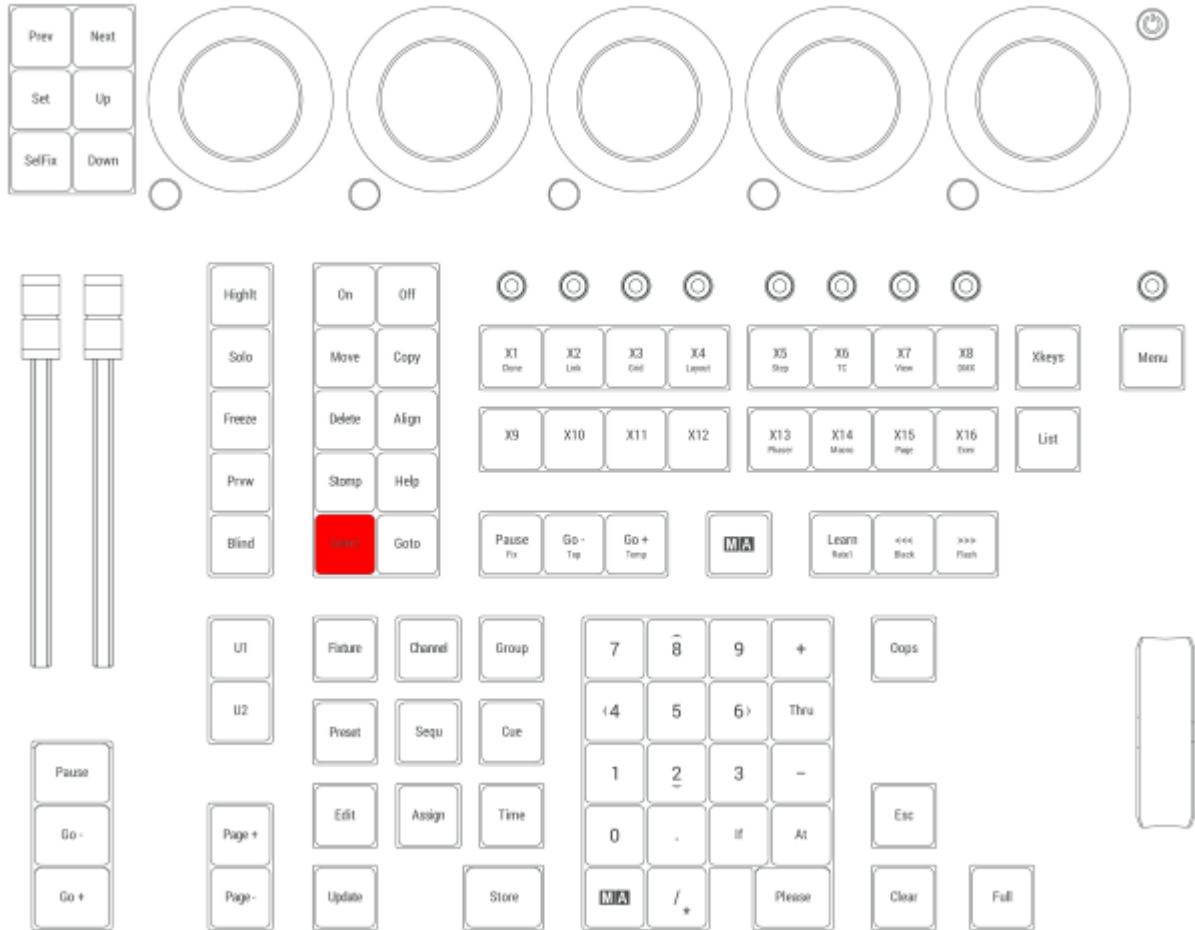


For more information about Select, see the [Select keyword](#) topic.

Select is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



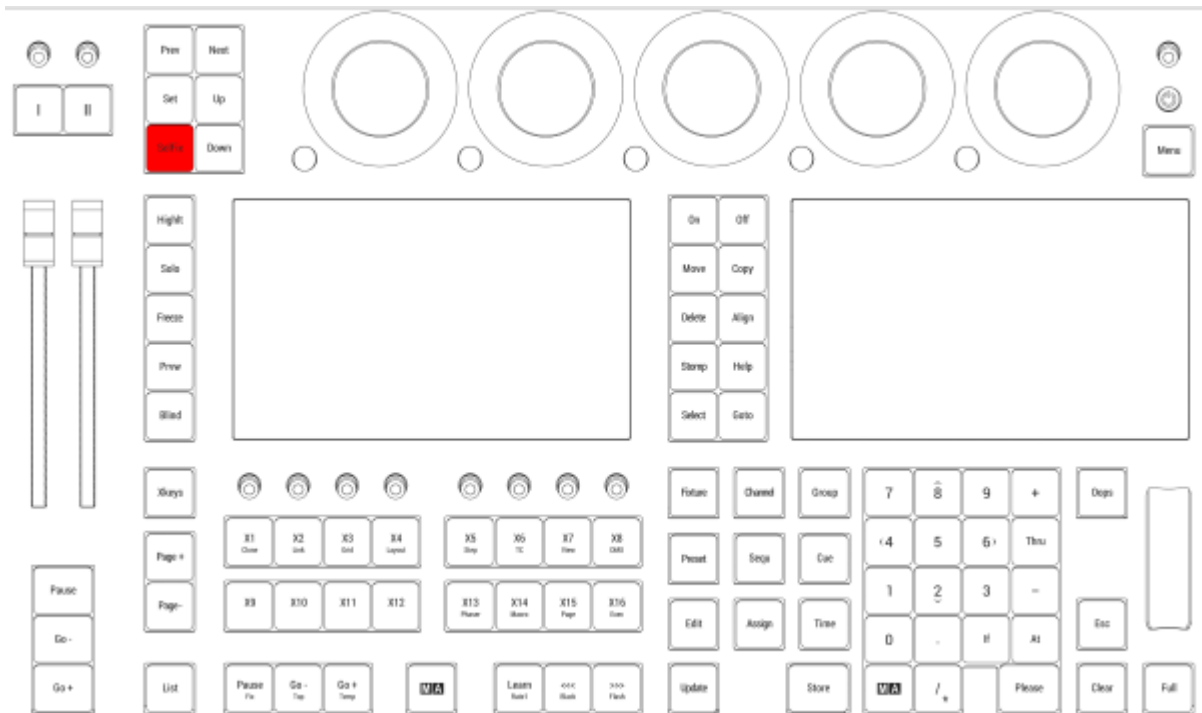
2.15.52. SelFix [SelectFixture] Key

Pressing **SelFix** enters the SelFix keyword into the command line.

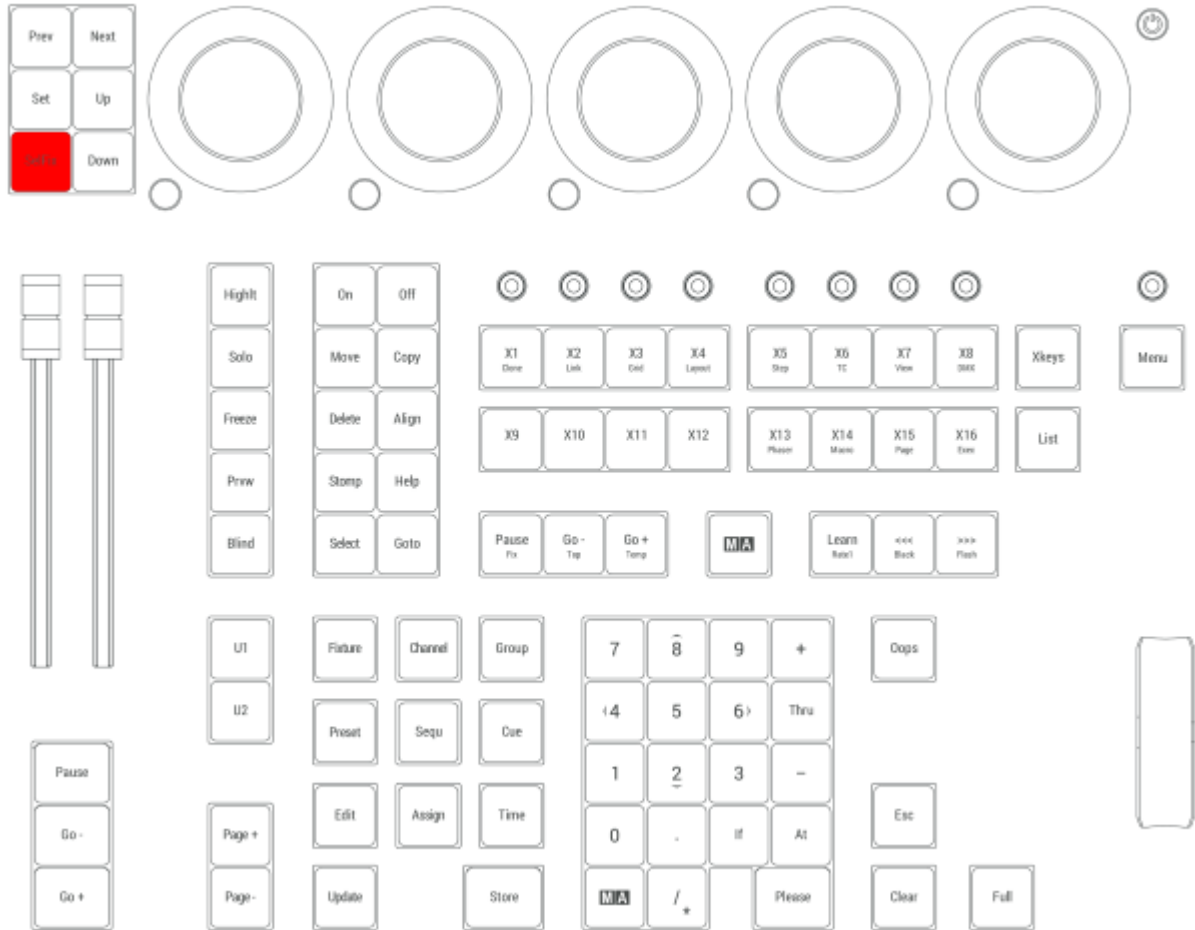


For more information about SelFix, see the **SelectFixture keyword** topic.

SelFix is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



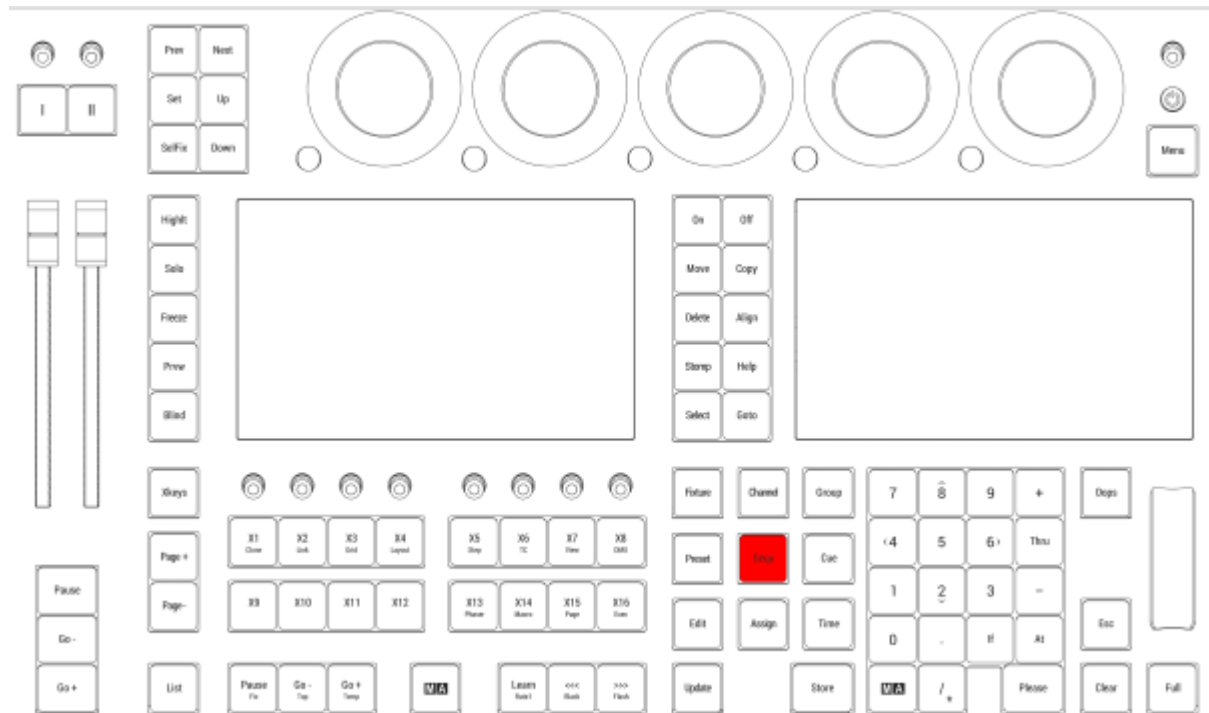
2.15.53. Sequ [Sequence] Key

Pressing **Sequ** enters the Sequence keyword into the command line.

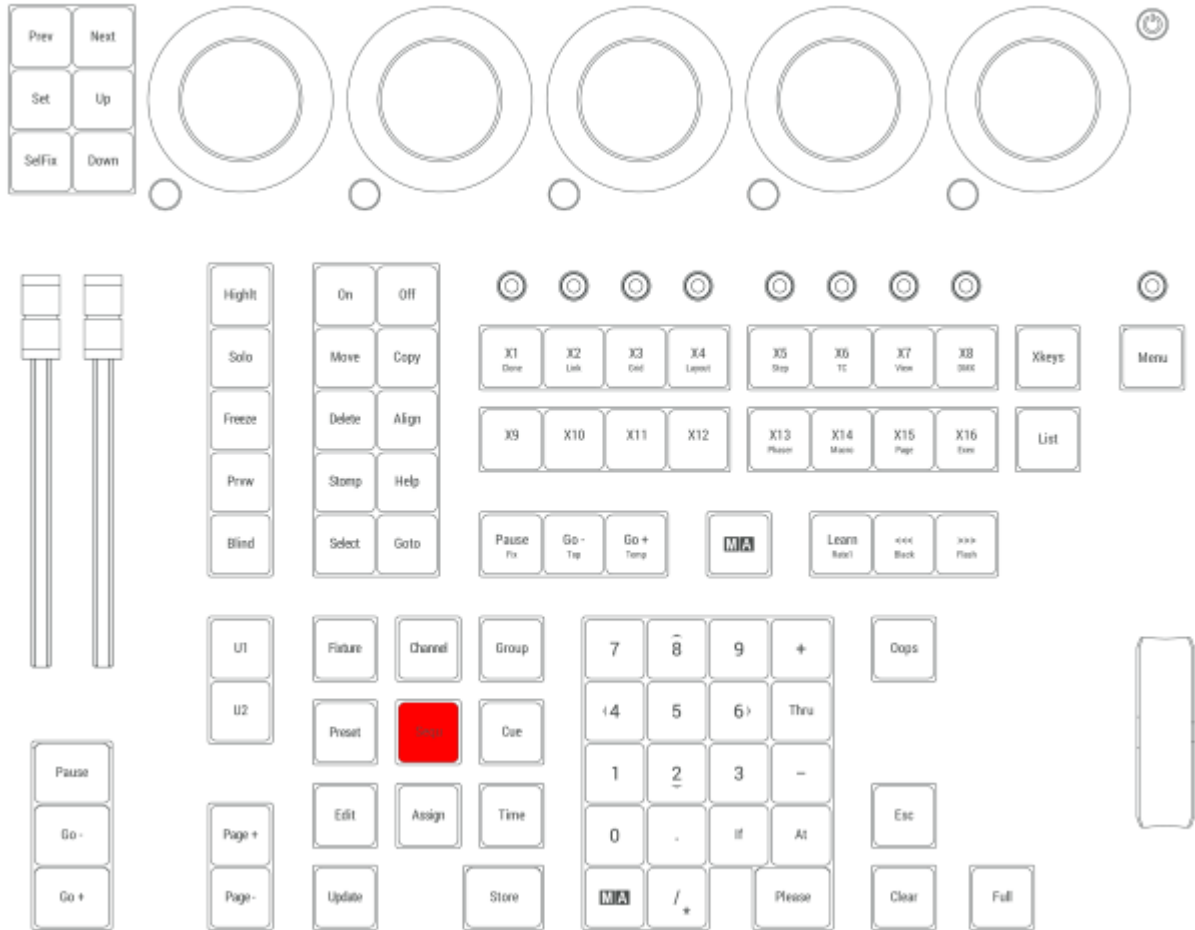


For more information about Sequence, see the [Sequence keyword](#) topic.

Sequ is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.54. Set Key

Pressing **Set** executes the Toggle MATricks command in the command line.

OK:Toggle MATricks

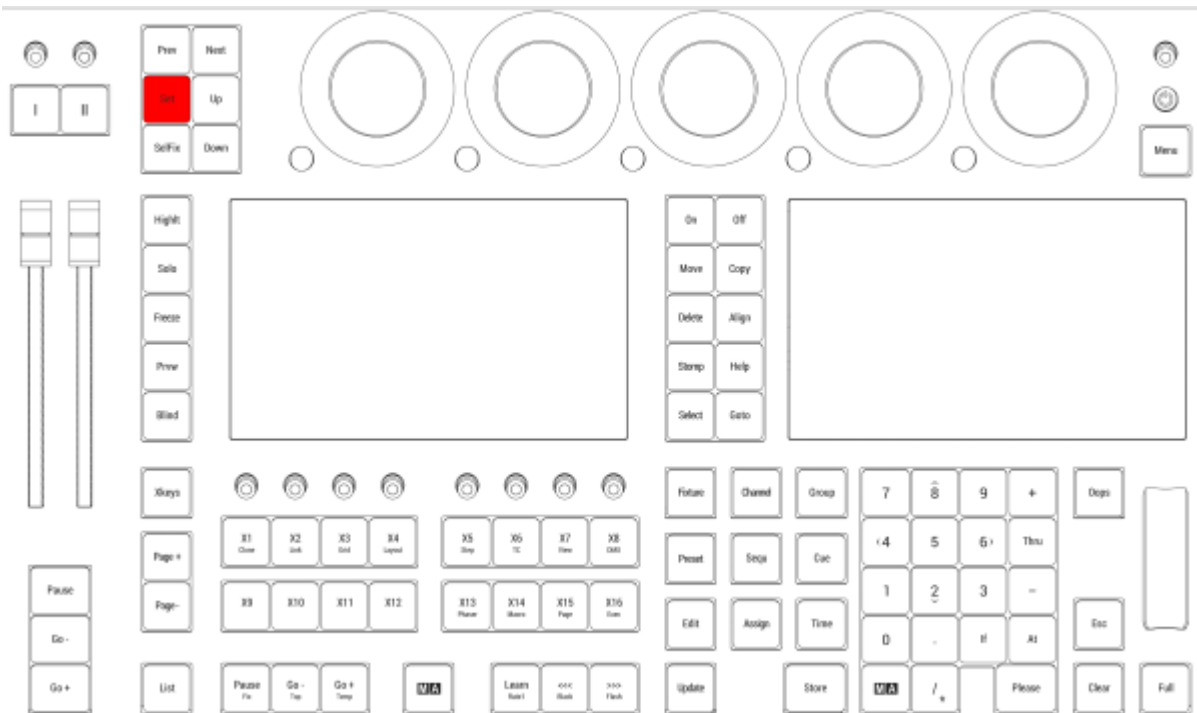
For more information about MATricks, see the [MATricks keyword](#) topic.

Step Toggle Executor

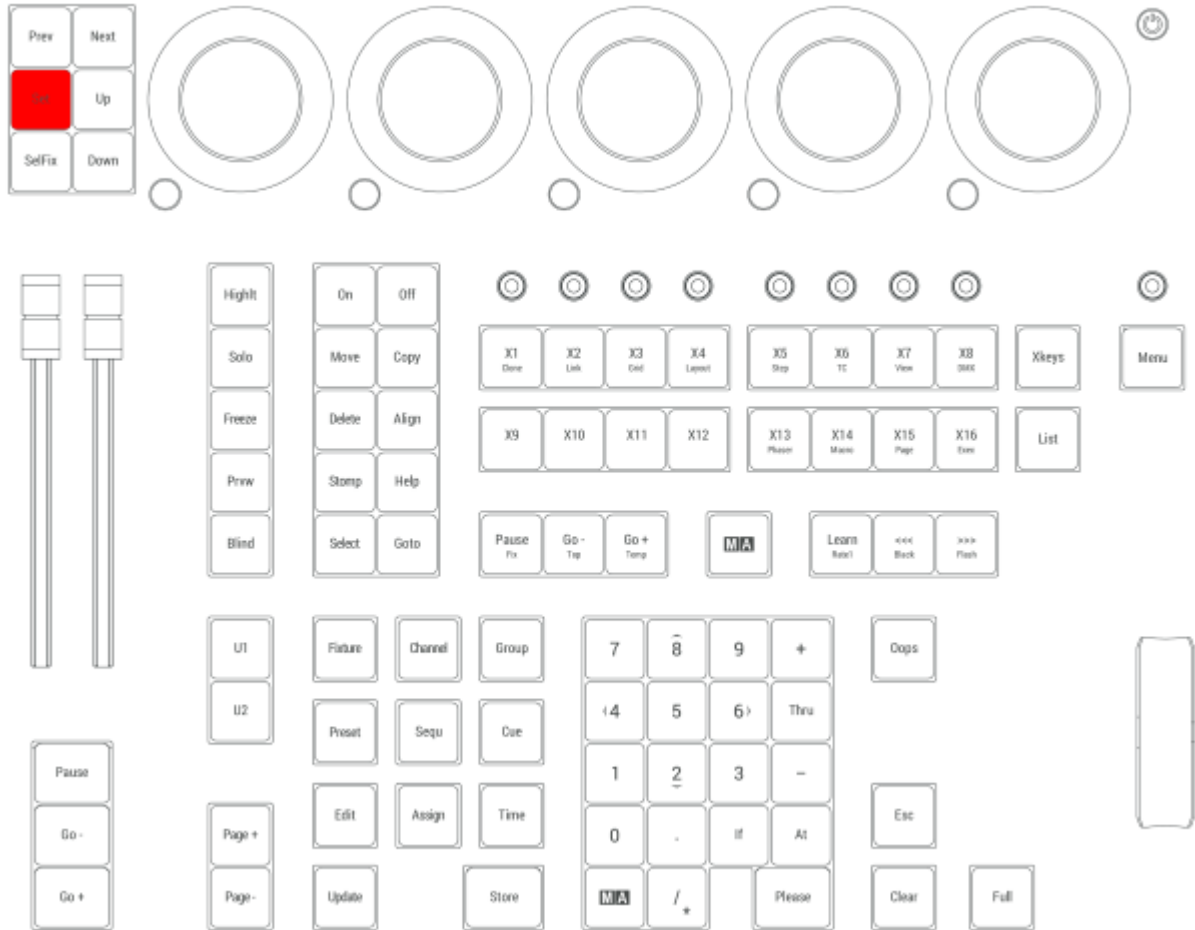
Pressing **MA** + **Set** executes the Step Toggle Executor command in the command line.

OK:Step Toggle Executor

Set is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



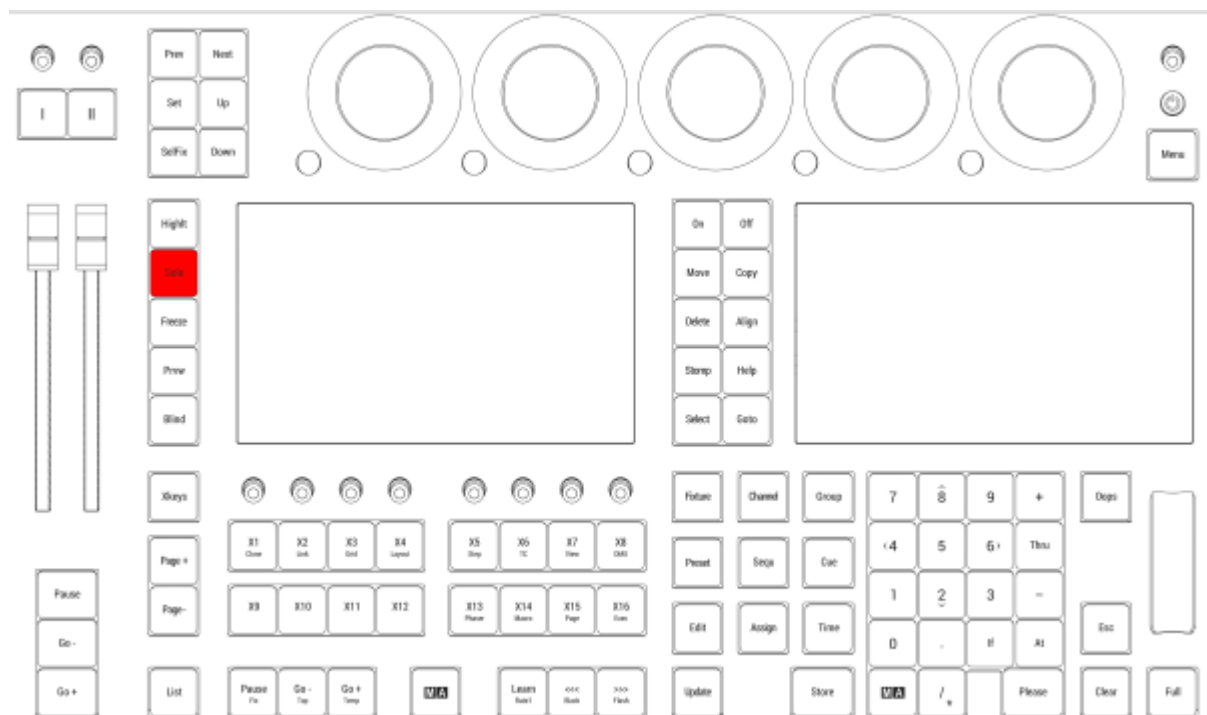
Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.55. Solo Key

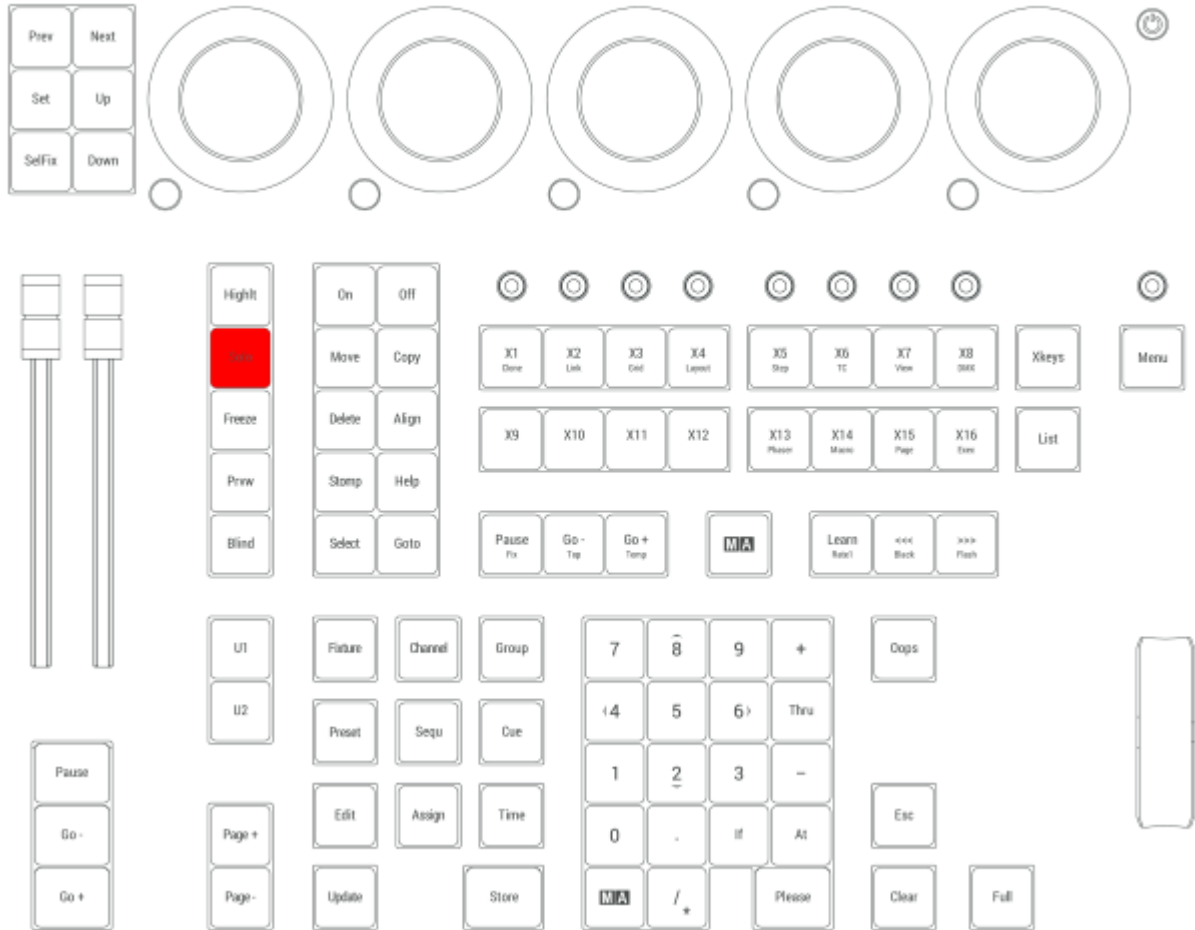
Pressing **Solo** executes directly solo selection and turns the Solo mode on or off.

For more information about Solo, see the **Solo keyword** topic.

Solo is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.56. Stomp Key

Pressing **Stomp** enters Stomp into the command line.



For more information on Stomp, see the [Stomp keyword](#) topic.

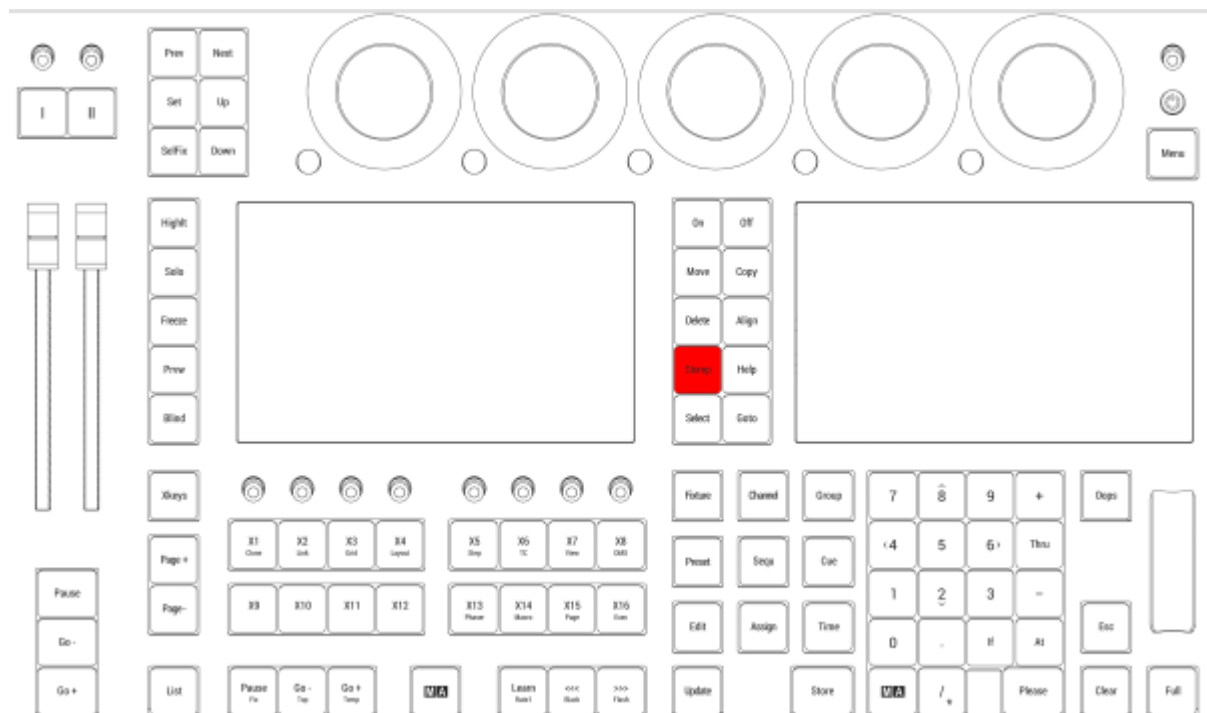
Capture

Pressing **Stomp** **Stomp** enters Capture into the command line.

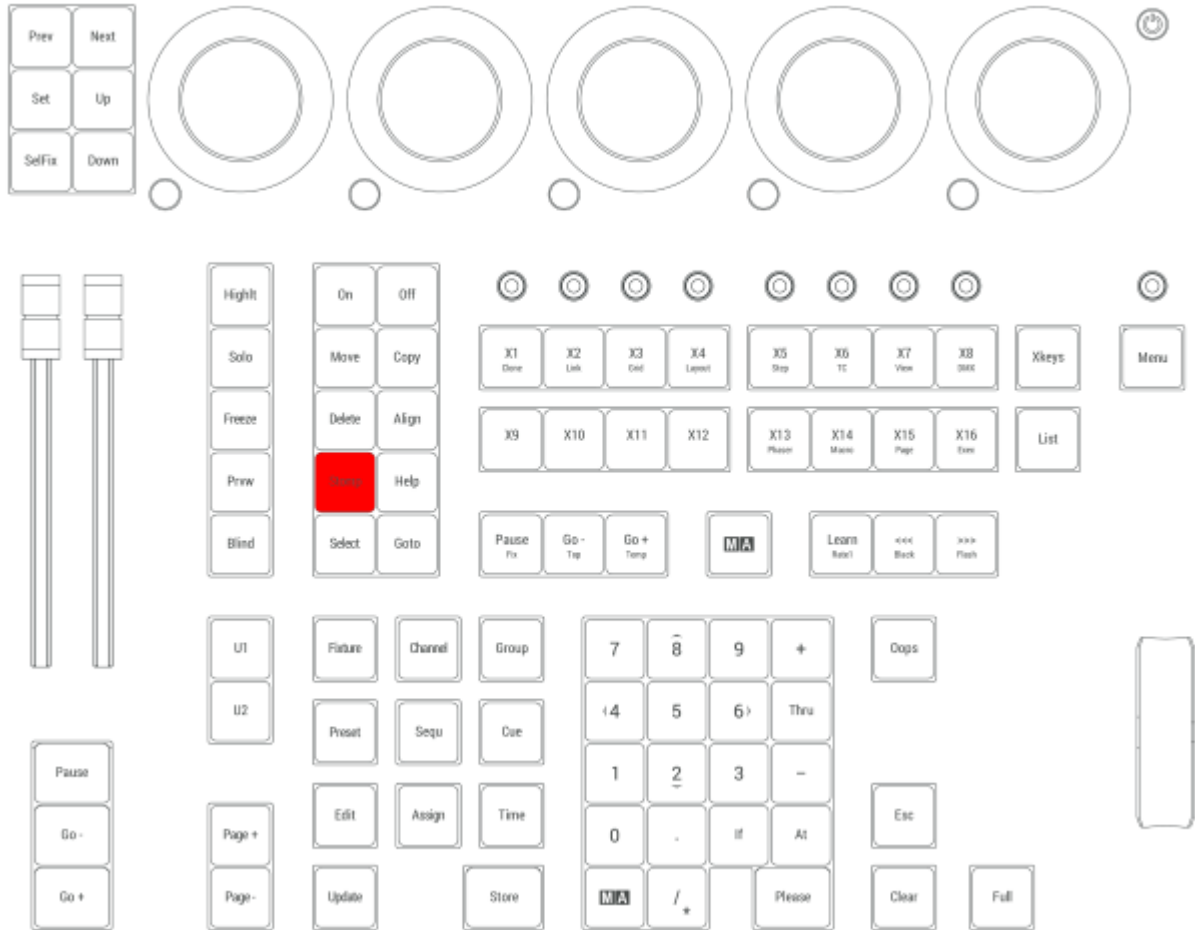


For more information on Capture, see the [Capture keyword](#) topic.

Stomp is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



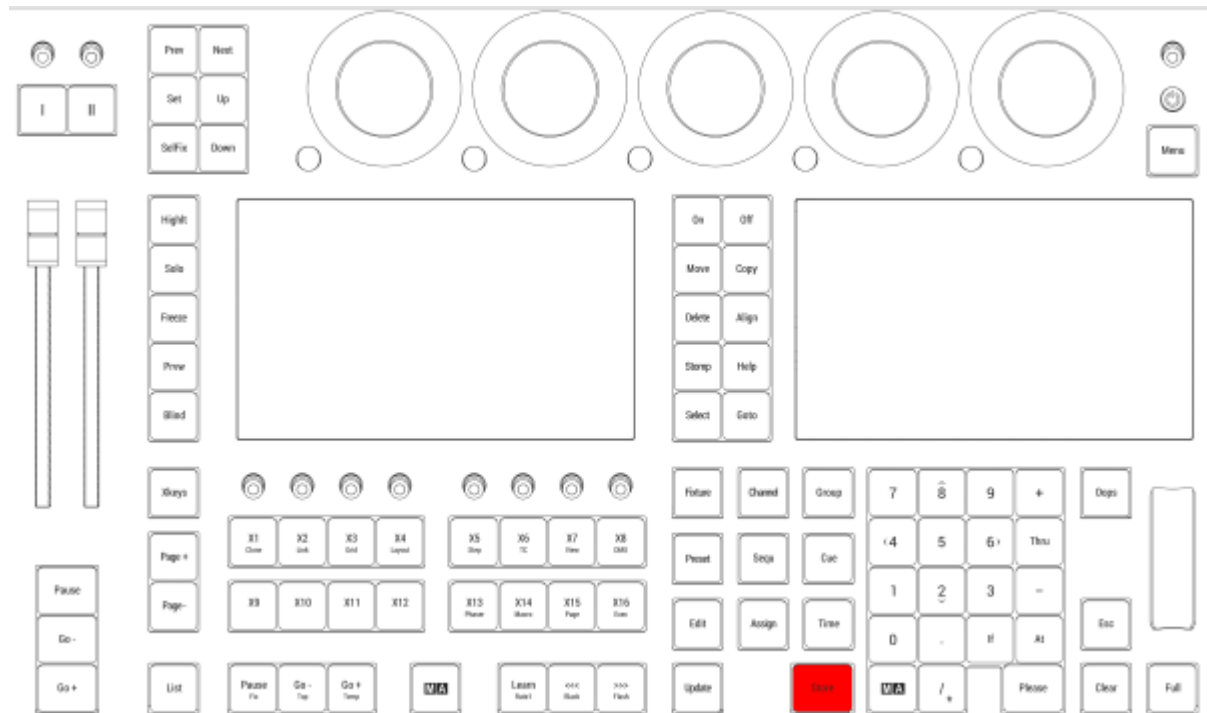
2.15.57. Store Key

Pressing **Store** enters the Store keyword into the command line.

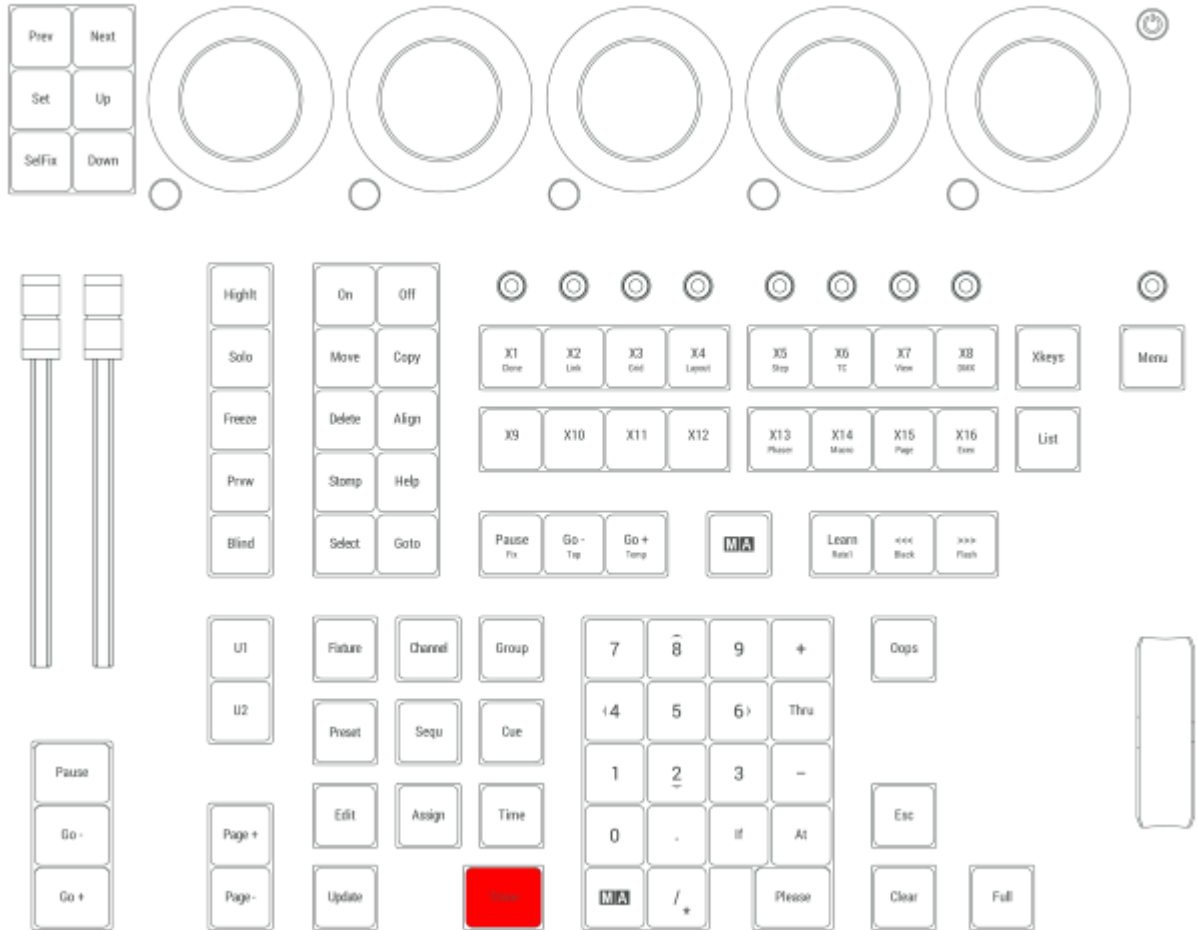


For more information about Store, see the **Store keyword** topic.

Store is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



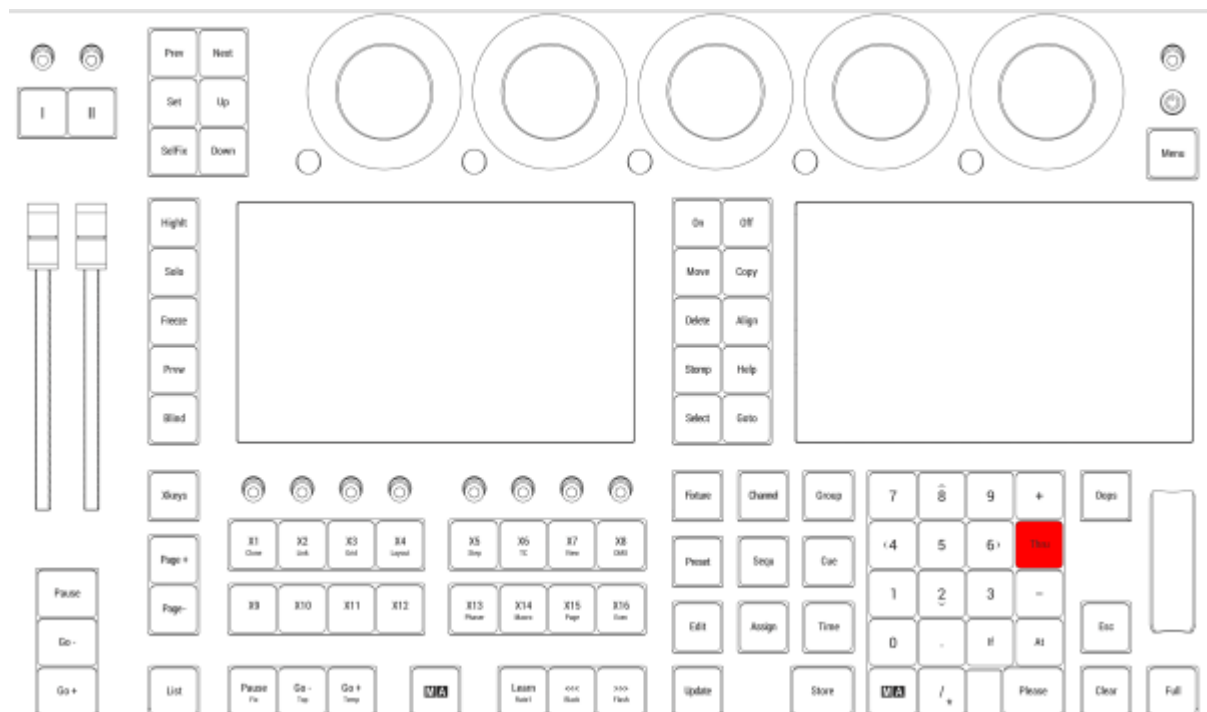
2.15.58. Thru Key

Pressing **Thru** enters the Thru keyword into the command line.

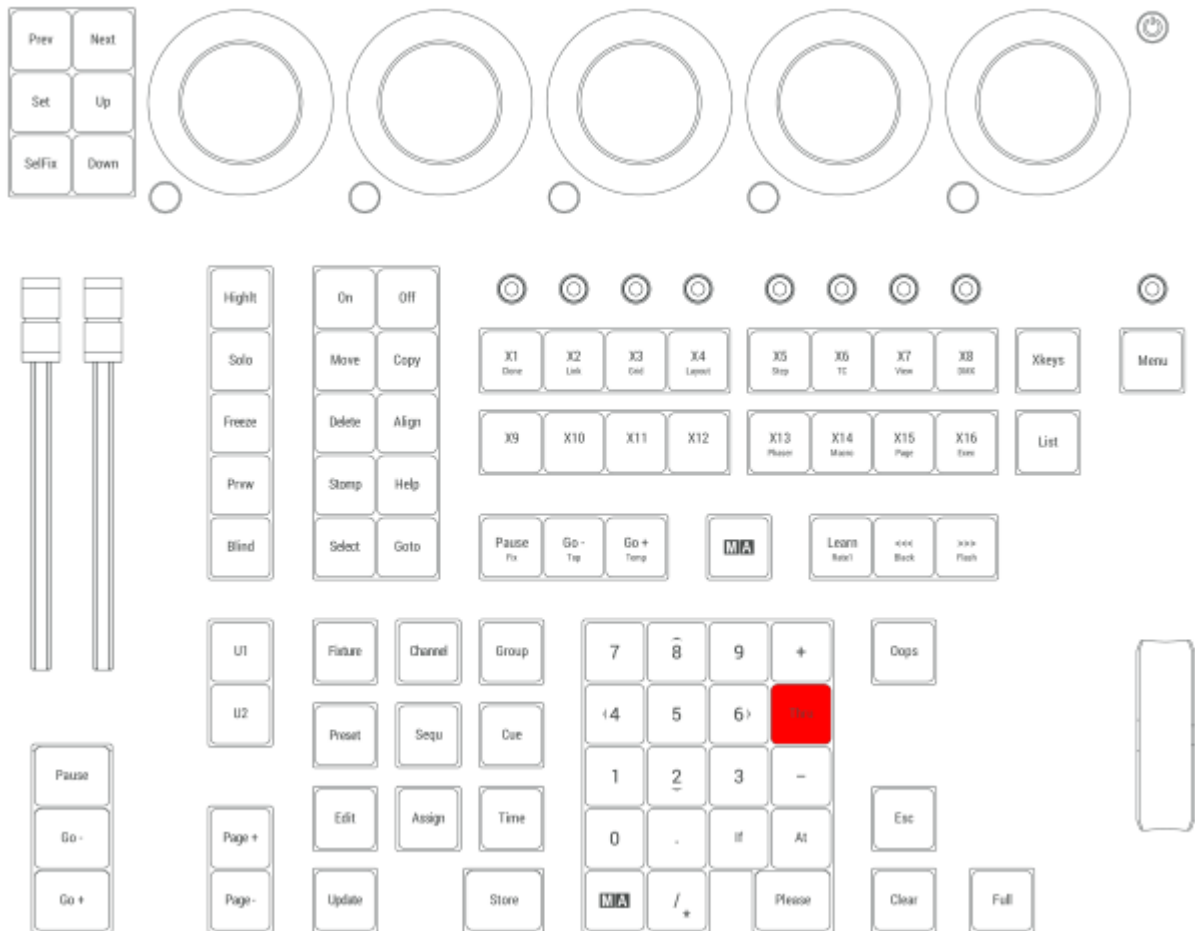


For more information about Thru, see the [Thru keyword](#) topic.

Thru is located in the numeric keys section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.59. Time Key

The default functionality of the Time key depends upon the "Time Key Target" setting for the current user profile.

If the "Time Key Target" is set to "Cue" and the command line is empty, pressing **Time** toggles between CueFade and CueDelay in the command line.

```
MA User name[Fixture]>CueFade
```

For more information about Time, see the [CueFade keyword](#) or [CueDelay keyword](#) topic.

If the "Time Key Target" is set to "Cue" and the command line contains a fixture selection, pressing **Time** toggles between Fade and Delay in the command line.

```
MA User name[Fixture]>Group 1 Delay
```

For more information about Time, see the [Fade keyword](#) or [Delay keyword](#) topic.

If the "Time Key Target" is set to "Fixture" and the command line is empty, pressing **Time** toggles between Fade and Delay in the command line.

```
MA User name[Fixture]>Delay
```

For more information about Time, see the [Fade keyword](#) or [Delay keyword](#) topic.

If the "Time Key Target" is set to "Fixture" and either Store or Cue is in the command line, pressing **Time** toggles between CueFade and CueDelay in the command line.

```
MA User name[Fixture]>Store CueFade
```

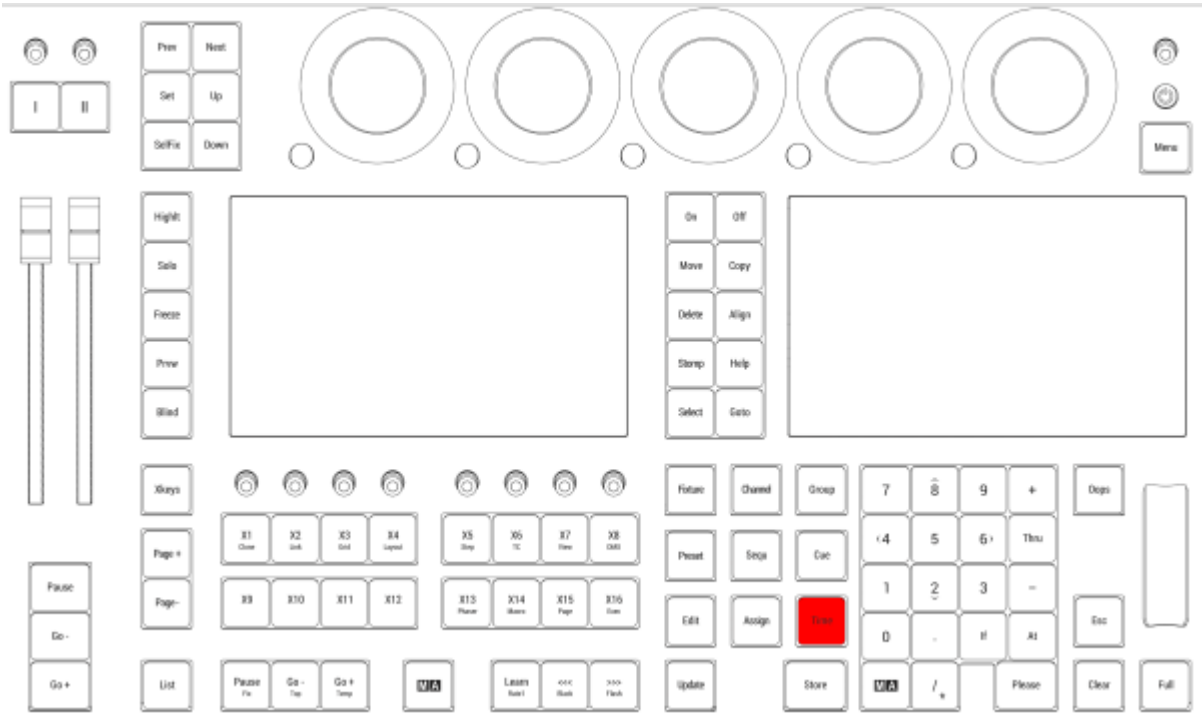
For more information about Time, see the [CueFade keyword](#) or [CueDelay keyword](#) topic.

Regardless of the "Time Key Target" setting, pressing **MA Time** toggles between Relative, Fade, Delay, and Absolute.

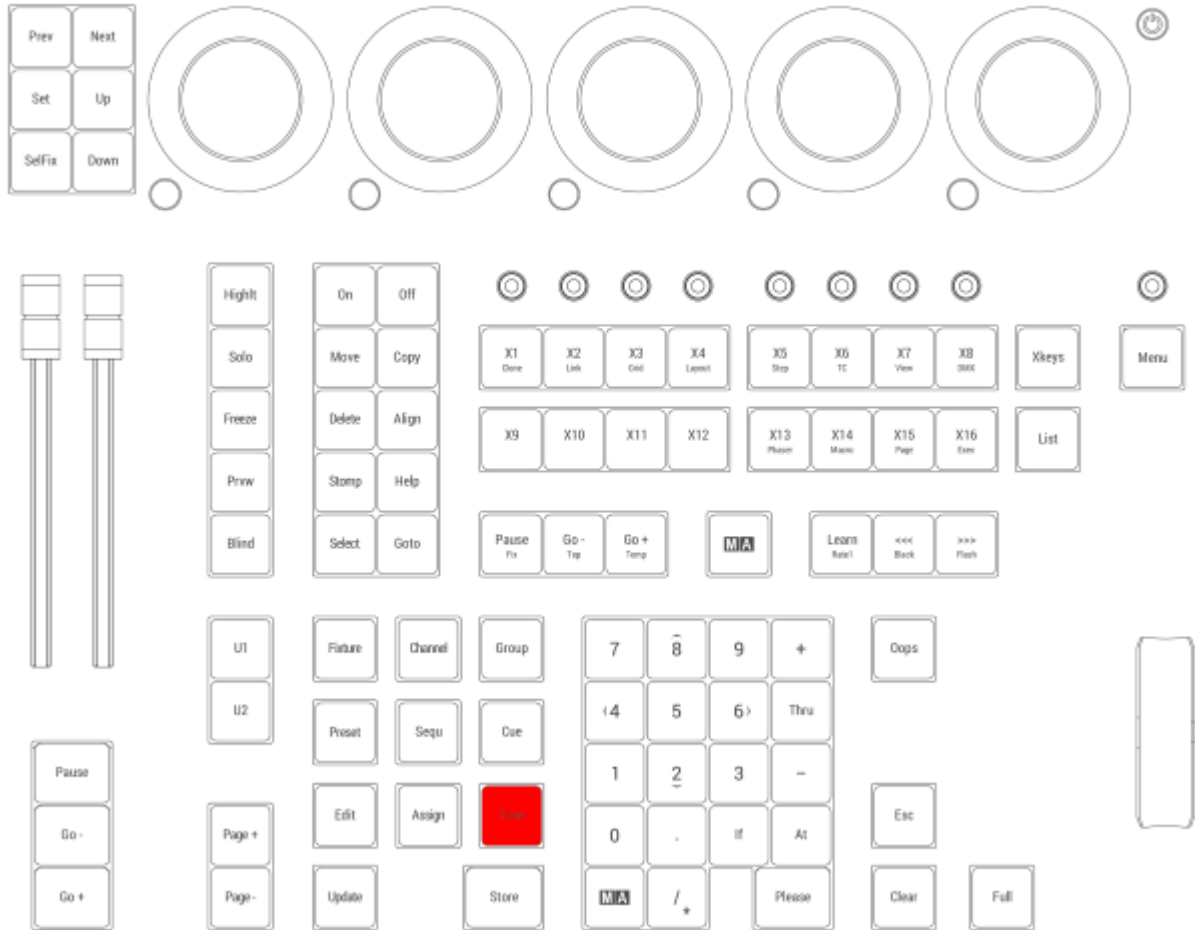
```
MA User name[Fixture]>Relative
```

For more information about Time, see the [Relative keyword](#), [Fade keyword](#), [Delay keyword](#), or [Absolute keyword](#) topic.

Time is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles

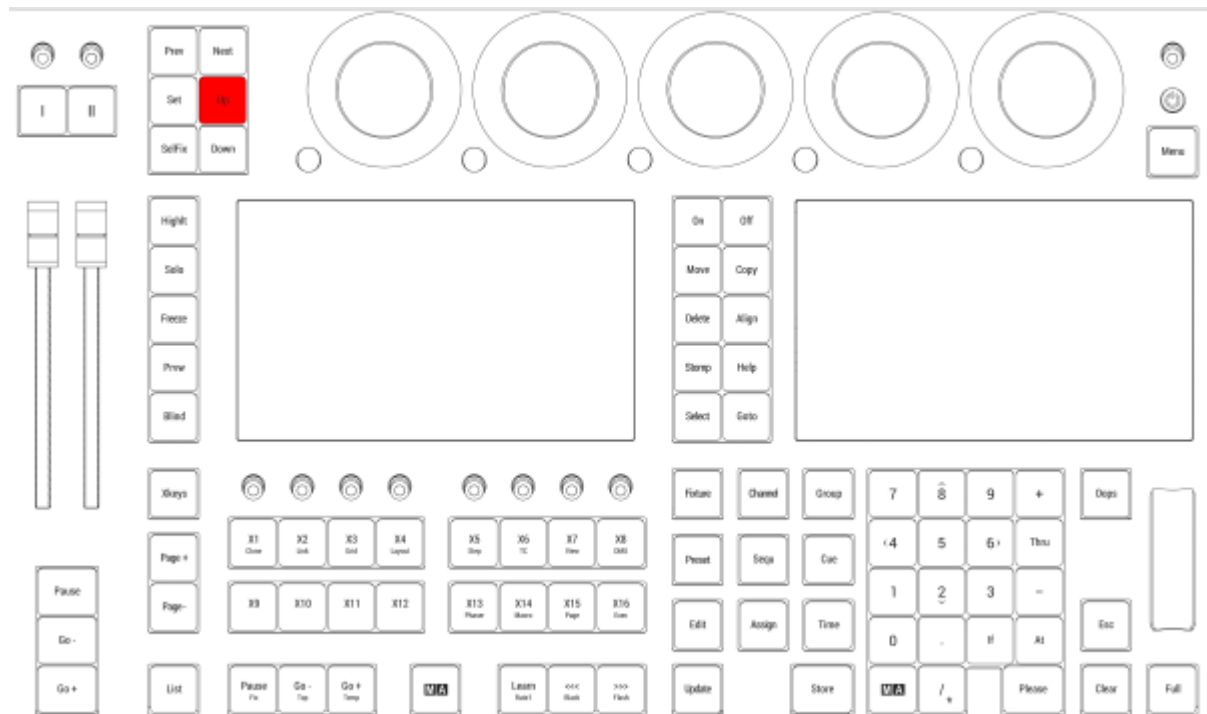


Location on grandMA3 compact consoles and grandMA3 onPC command wing

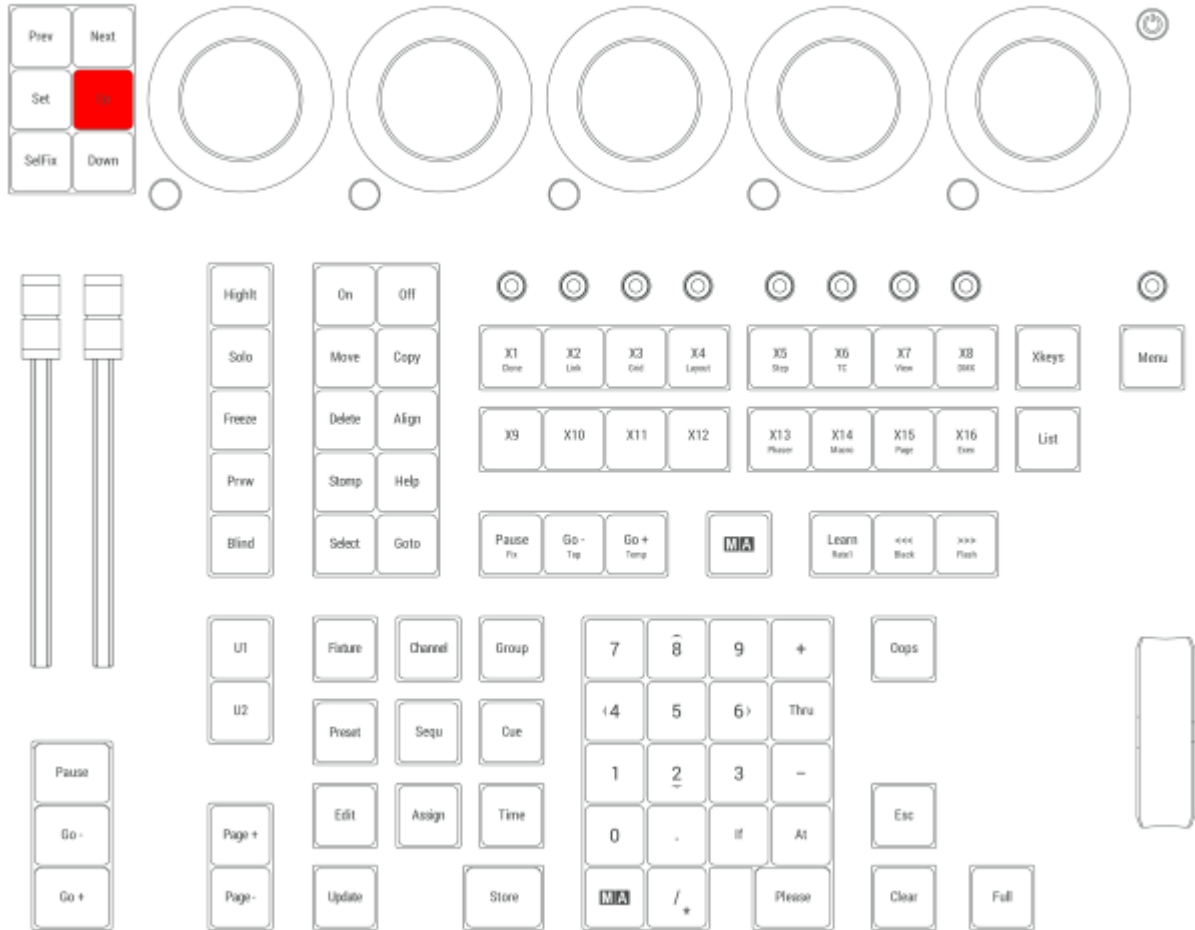
2.15.60. Up Key

Pressing **Up** navigates up in the fixture/subfixture structure.

Up is located in the command section on the left side of the five dual encoders.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.61. Update Key

Pressing **Update** enters the Update keyword into the command line.



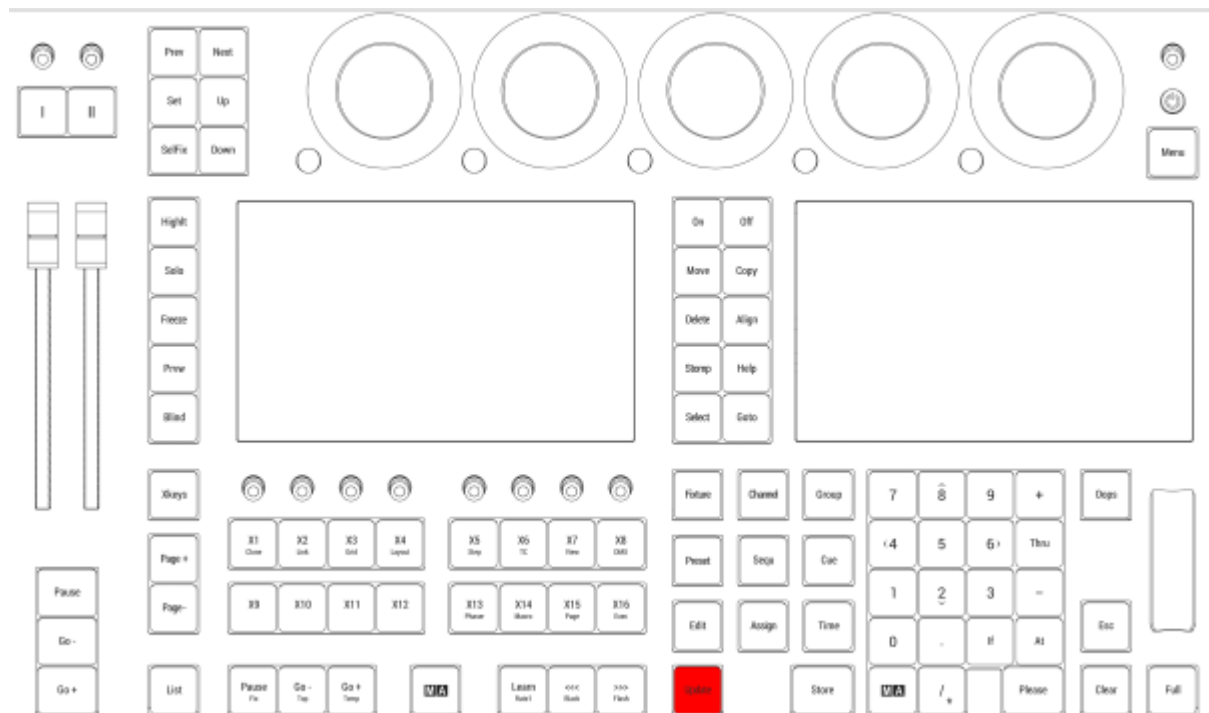
For more information about Update, see the [Update Keyword](#) topic.

Pressing **MA** + **Update** enters the Cook keyword into the command line.

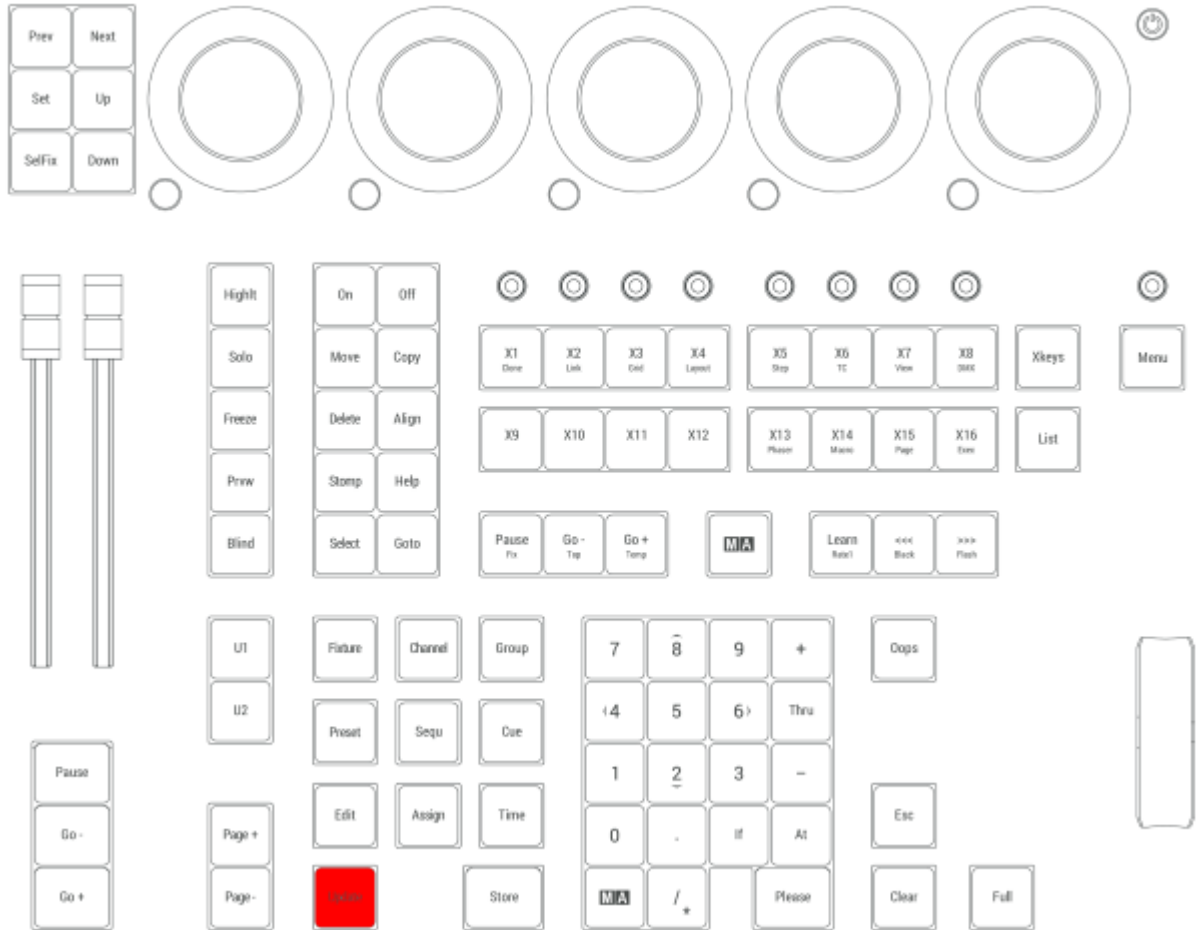


For more information about Cook, see the [Cook Keyword](#) topic.

Update is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.62. U1 Key

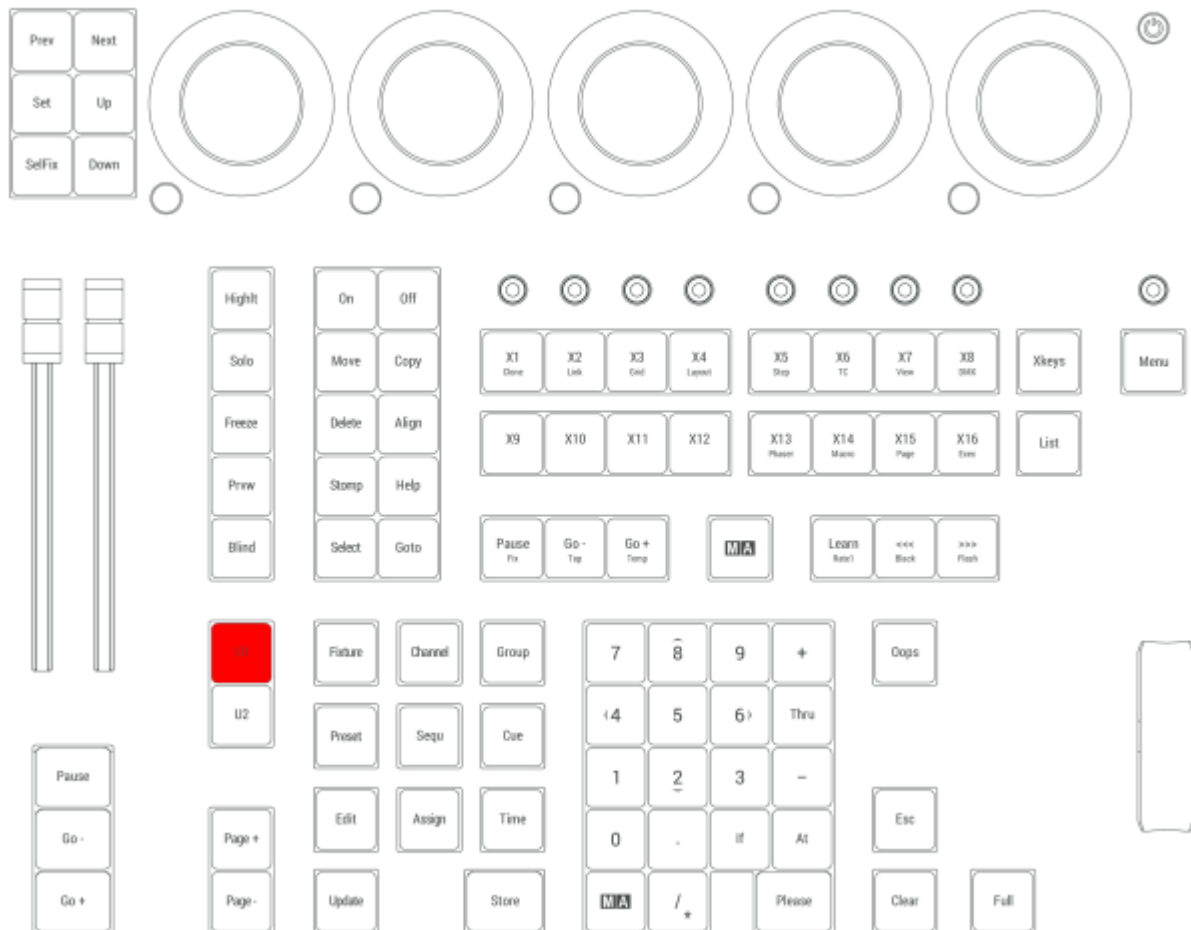
Pressing **U1** executes the User1 command in the command line.



This key is only present on grandMA3 compact, grandMA3 compact XT, grandMA3 onPC command wing and grandMA3 onPC command wing XT.

For more information about U1, see the [User1 keyword topic](#).

U1 is located in the command section on the left side of the numeric keys.



Location on grandMA3 compact consoles and grandMA3 onPC command wings



2.15.63. U2 Key

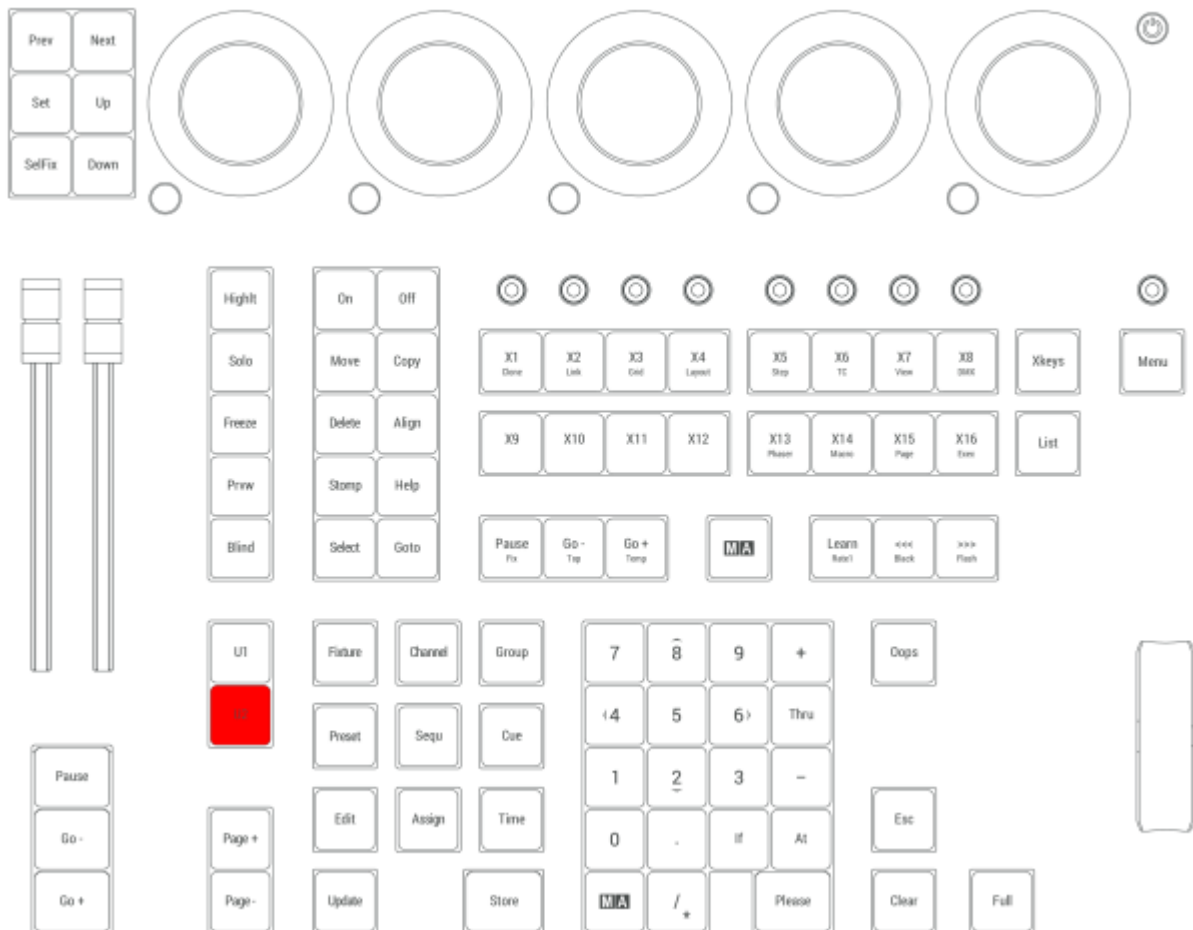
Pressing **U2** executes the User2 command in the command line.



This key is only present on grandMA3 compact, grandMA3 compact XT, grandMA3 onPC command wing and grandMA3 onPC command wing XT.

For more information about U2, see the [User2 keyword topic](#).

U2 is located in the command section on the left side of the numeric keys.



Location on grandMA3 compact consoles and grandMA3 onPC command wings



2.15.64. X1 | Clone Key

All Xkeys behave like executors. **X1** is executor 291.

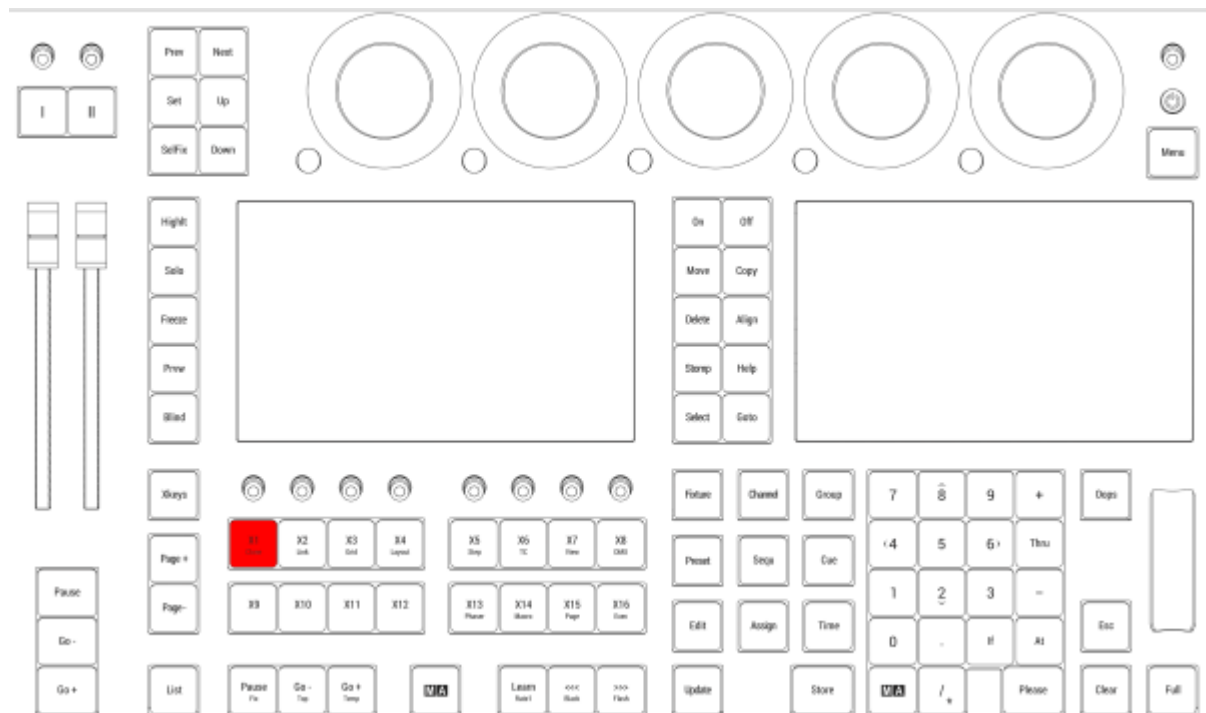
Clone

Pressing and holding **MA** + **X1** enters the Clone keyword into the command line.

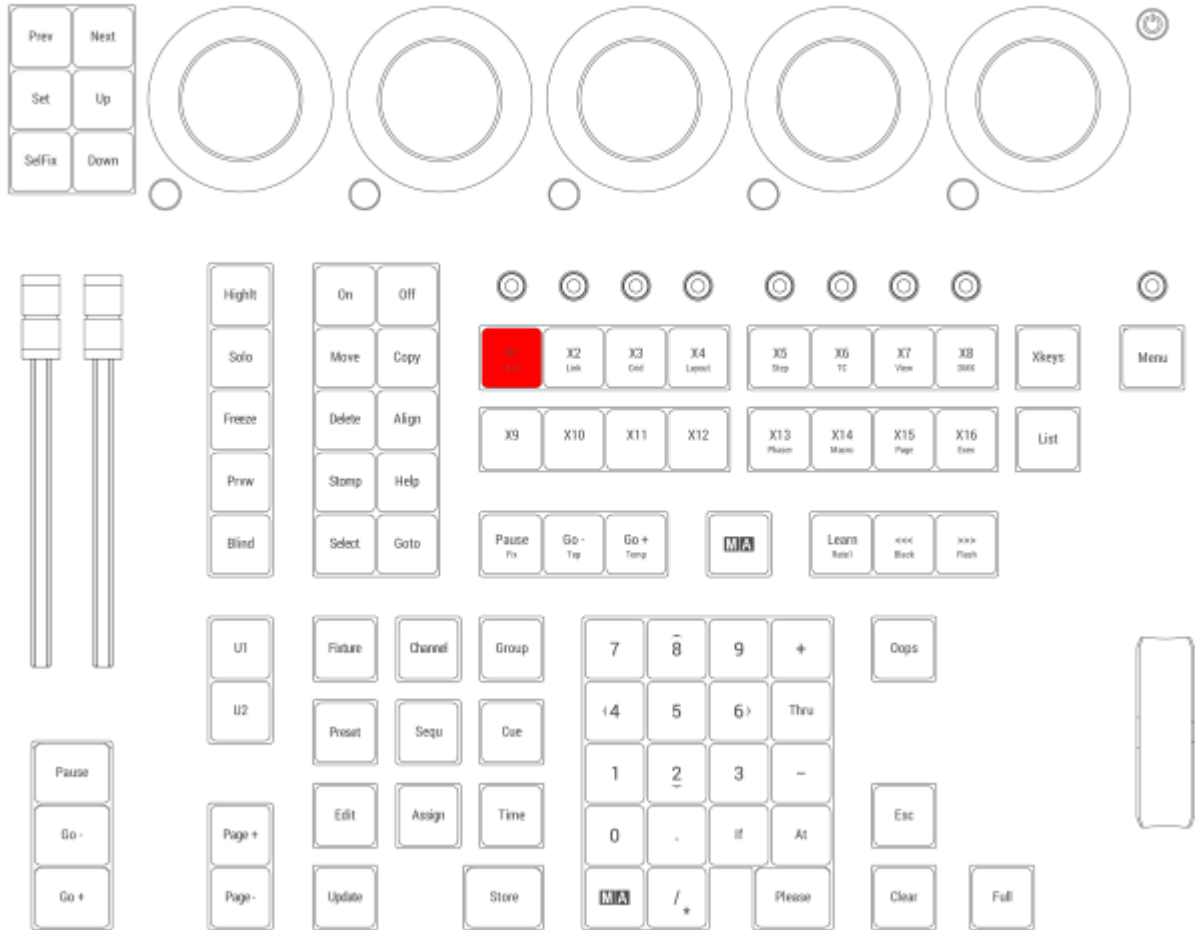


For more information about Clone, see the **Clone keyword** topic.

X1 is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



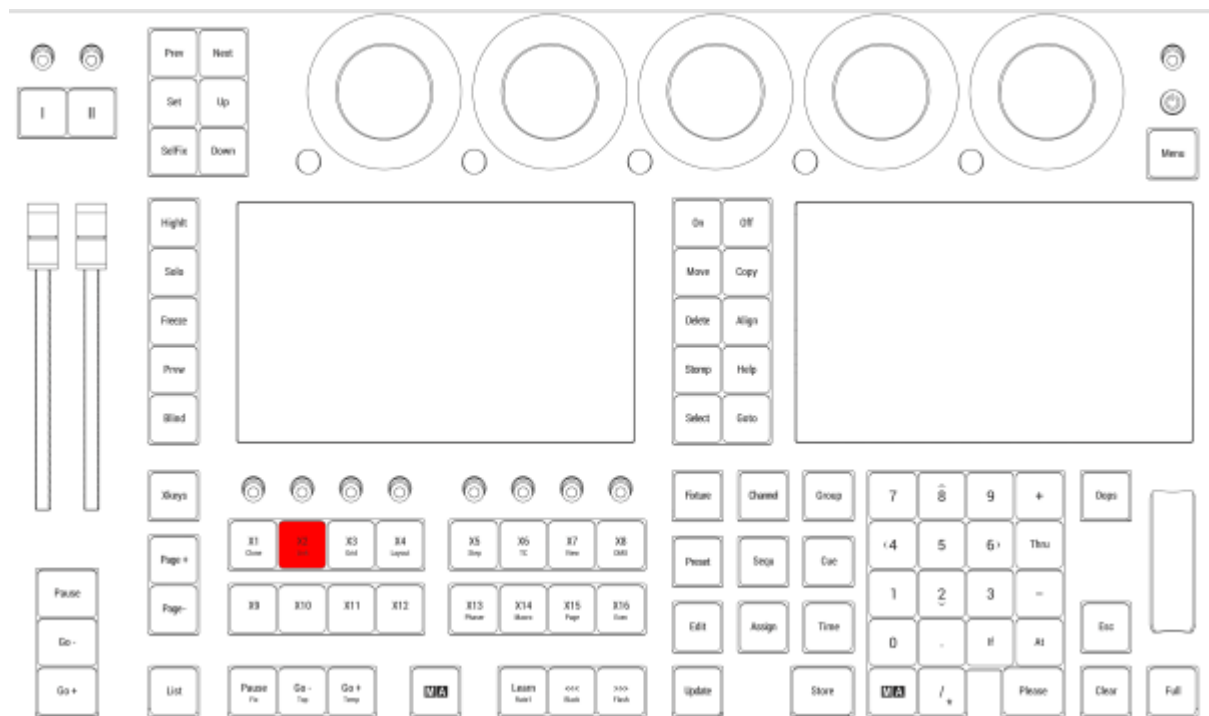
Location on grandMA3 compact consoles and grandMA3 onPC command wing



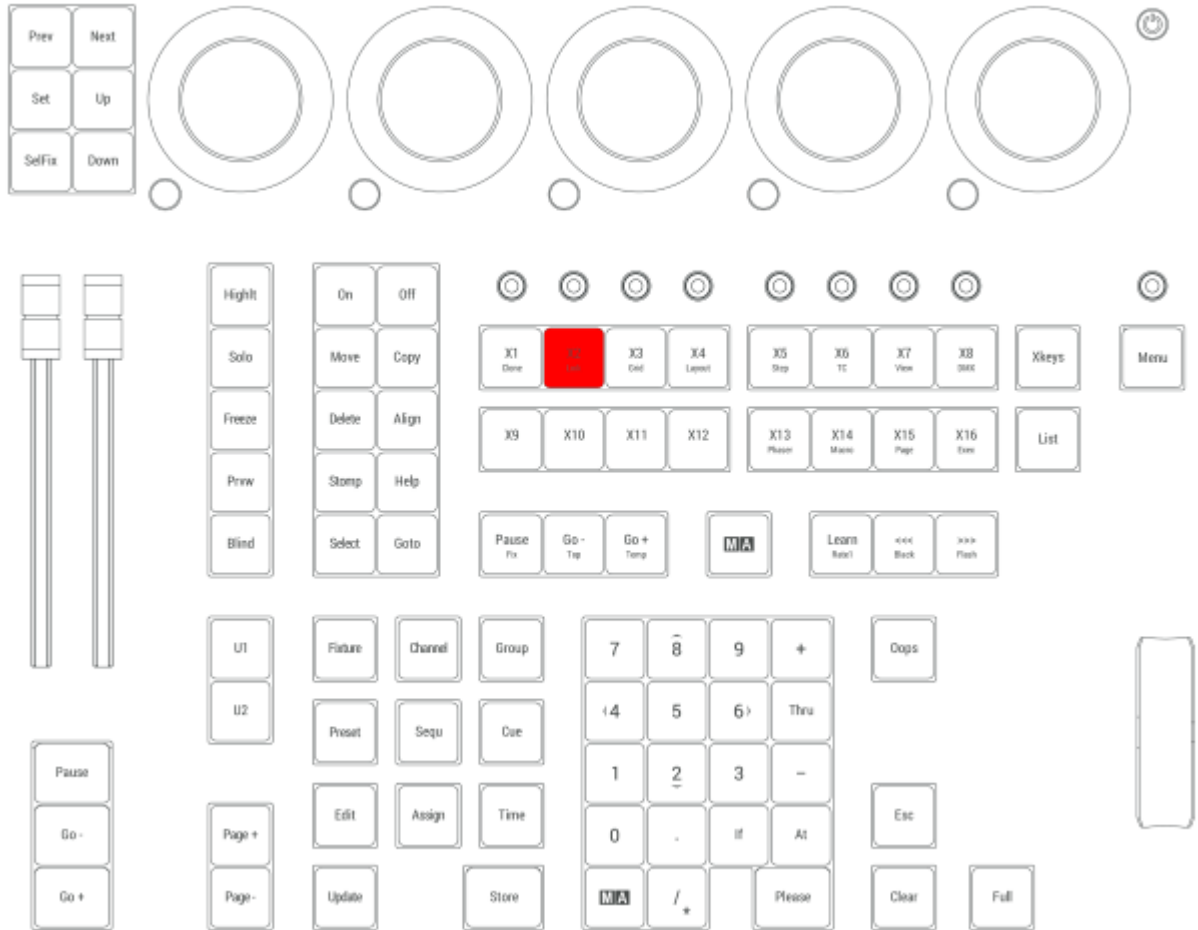
2.15.65.X2 | Link Key

All Xkeys behave like executors. **X2** is executor 292.

X2 is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.66. X3 | Grid Key

All Xkeys behave like executors. **X3** is executor 293.

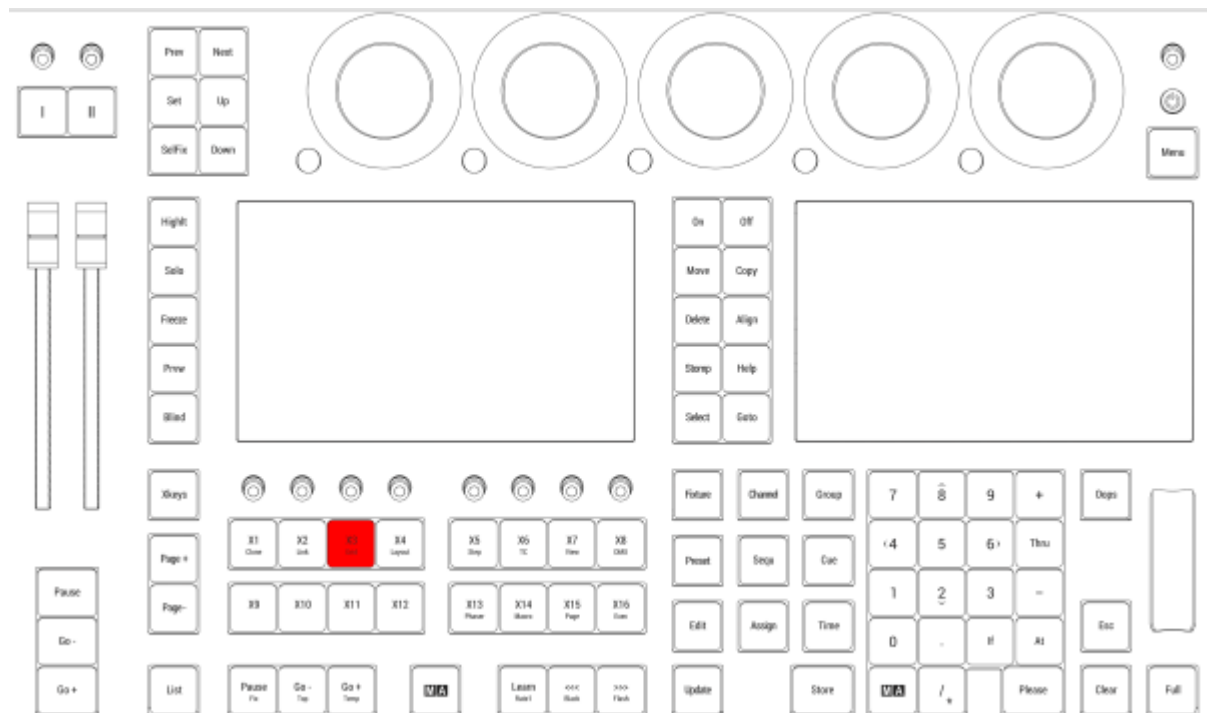
Grid

Pressing and holding **MA** + **X3** enters the Grid keyword into the command line.

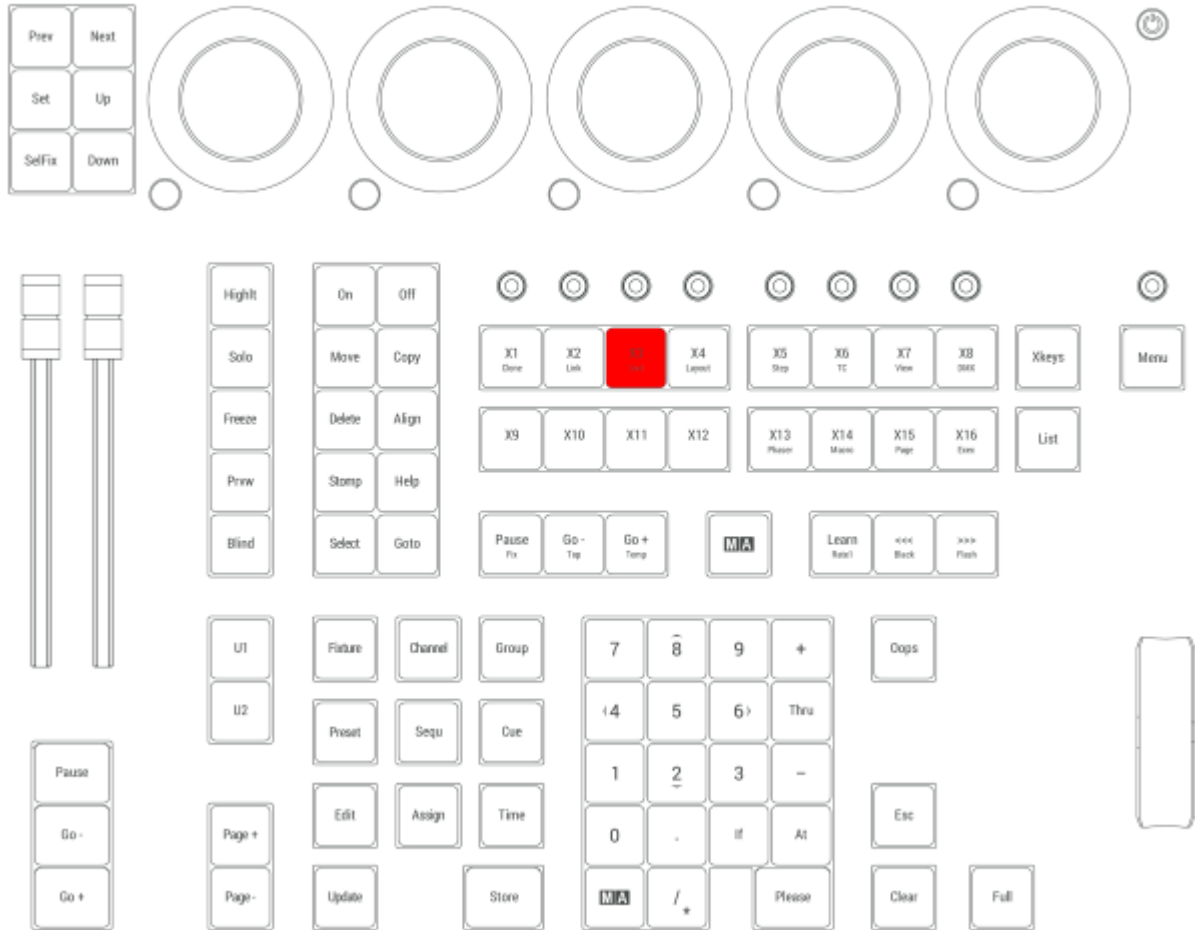


For more information about Grid, see the [Grid keyword](#) topic.

X3 is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

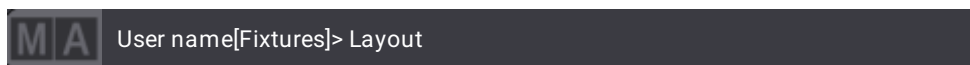


2.15.67. X4 | Layout Key

All Xkeys behave like executors. **X4** is executor 294.

Layout

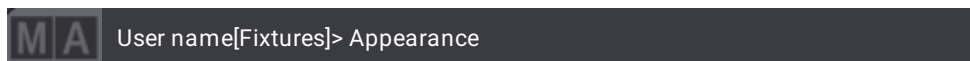
Pressing and holding **MA** + **X4** enters the Layout keyword into the command line.



For more information about Layout, see the [Layout keyword](#) topic.

Appearance

Pressing and holding **MA** + **X4** + **X4** enters the Appearance keyword into the command line.



For more information about Appearance, see the [Appearance keyword](#) topic.

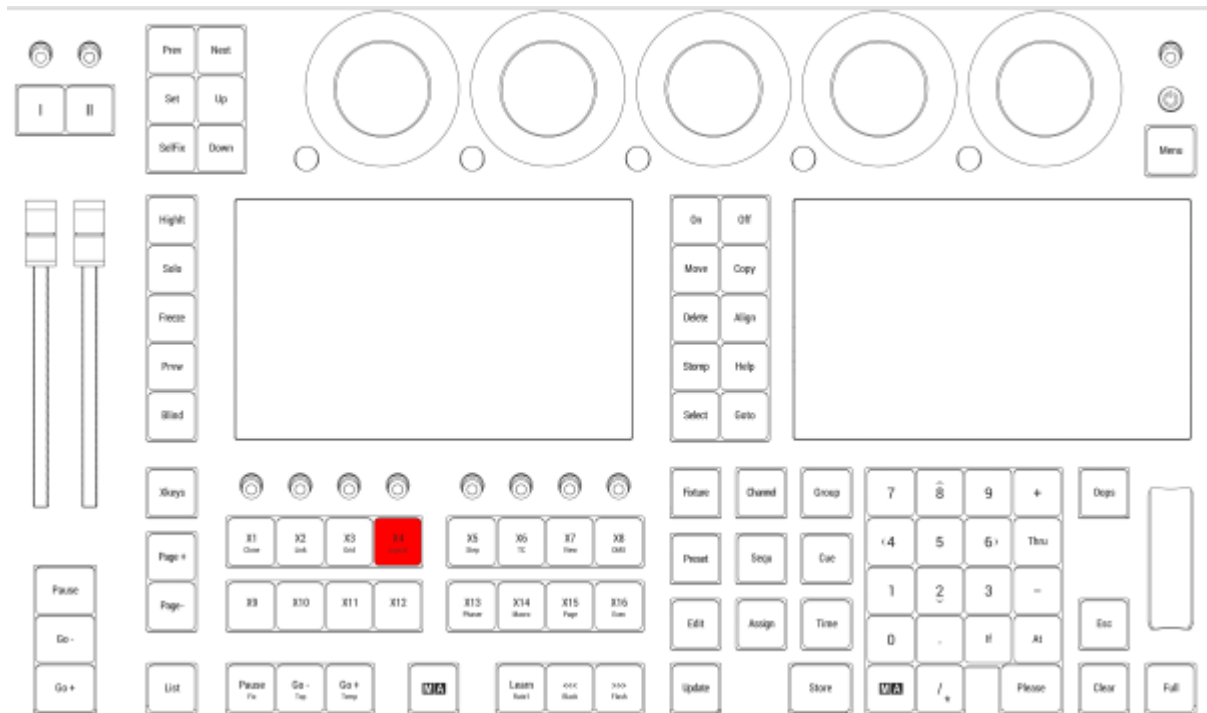
Scribble

Pressing and holding **MA** + **X4** + **X4** + **X4** enters the Scribble keyword into the command line.

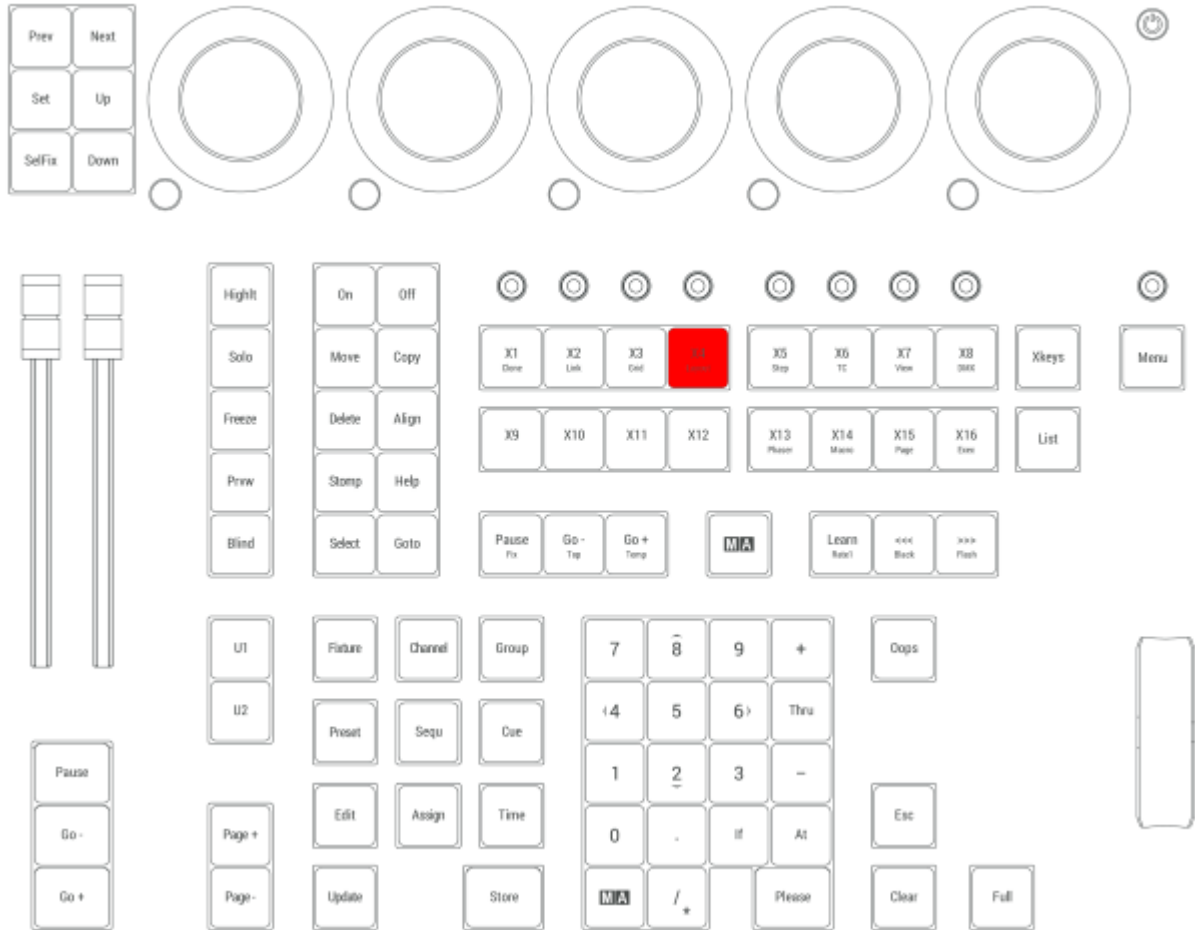


For more information about Scribble, see the [Scribble keyword](#) topic.

X4 is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.68.X5 | Step Key

All Xkeys behave like executors. **X5** is executor 295.

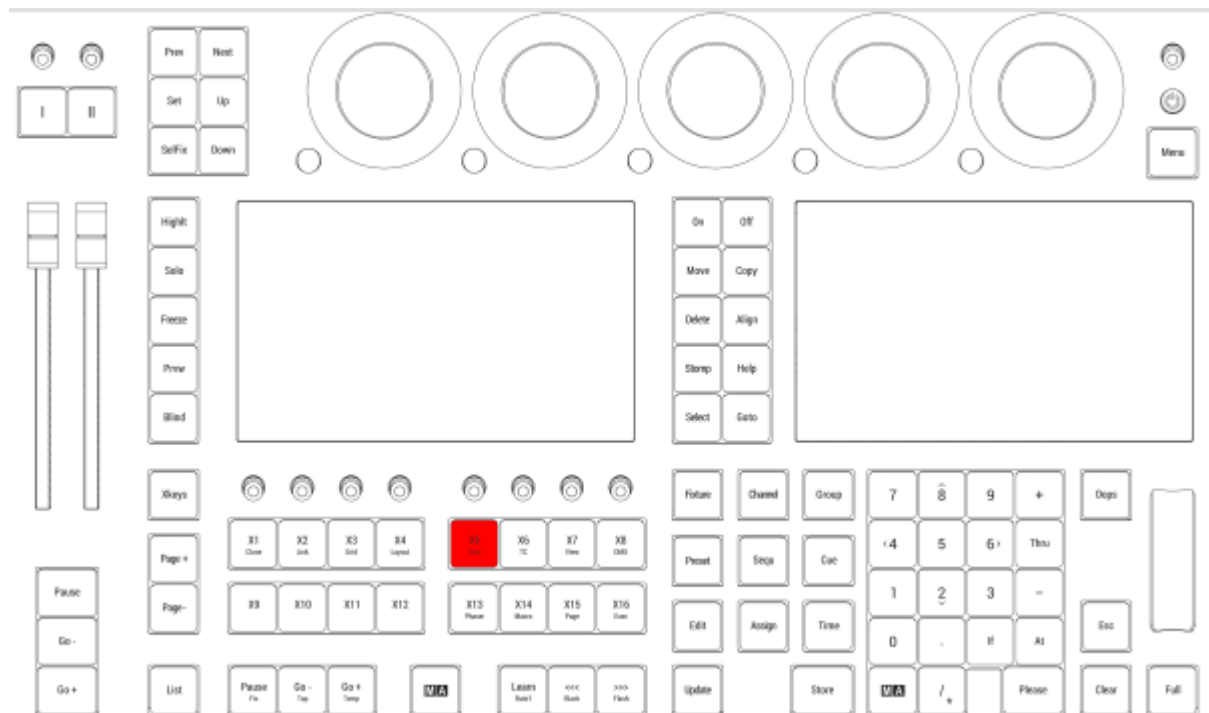
Step

Pressing and holding **MA** + **X5** enters the Step keyword into the command line.

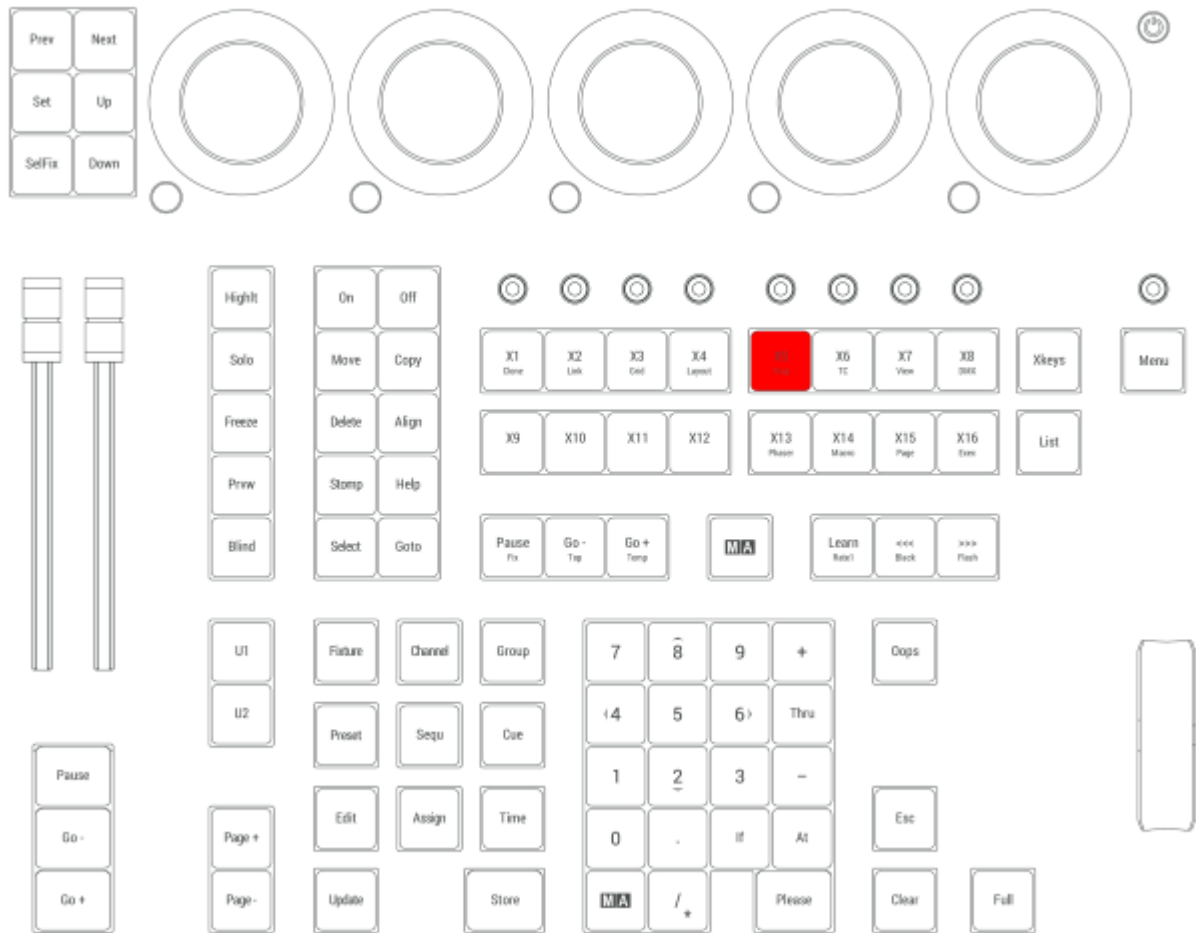


For more information about Step, see the [Step keyword](#) topic.

X5 is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.69. X6 | TC Key

All Xkeys behave like executors. **X6** is executor 296.

Timecode

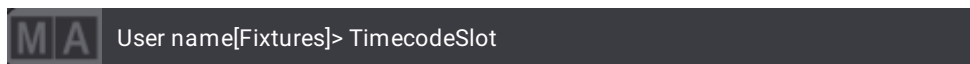
Pressing and holding **MA** + **X6** enters the Timecode keyword into the command line.



For more information about Timecode, see the [Timecode keyword](#) topic.

TimecodeSlot

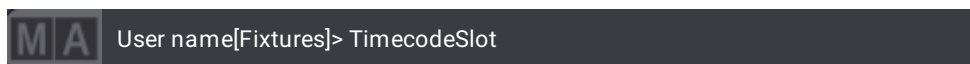
Pressing and holding **MA** + **X6** + **X6** enters the TimecodeSlot keyword into the command line.



For more information about TimecodeSlot, see the [TimecodeSlot keyword](#) topic.

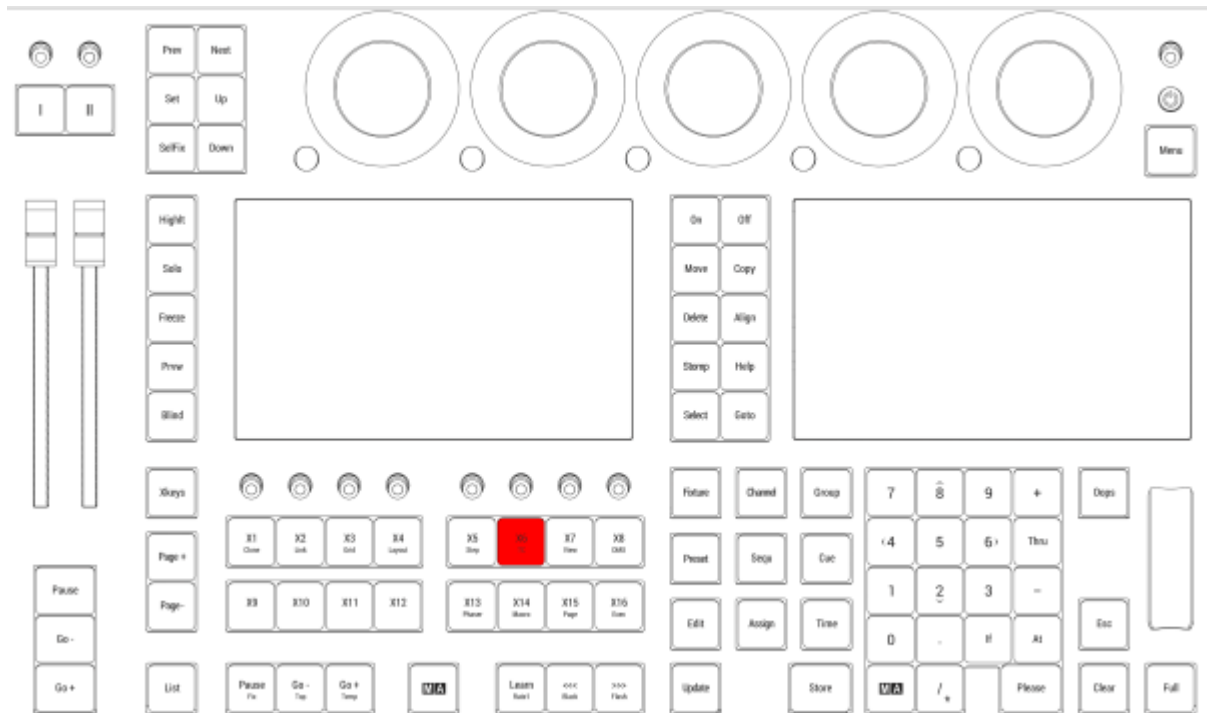
Timer

Pressing and holding **MA** + **X6** + **X6** + **X6** enters the Timer keyword into the command line.

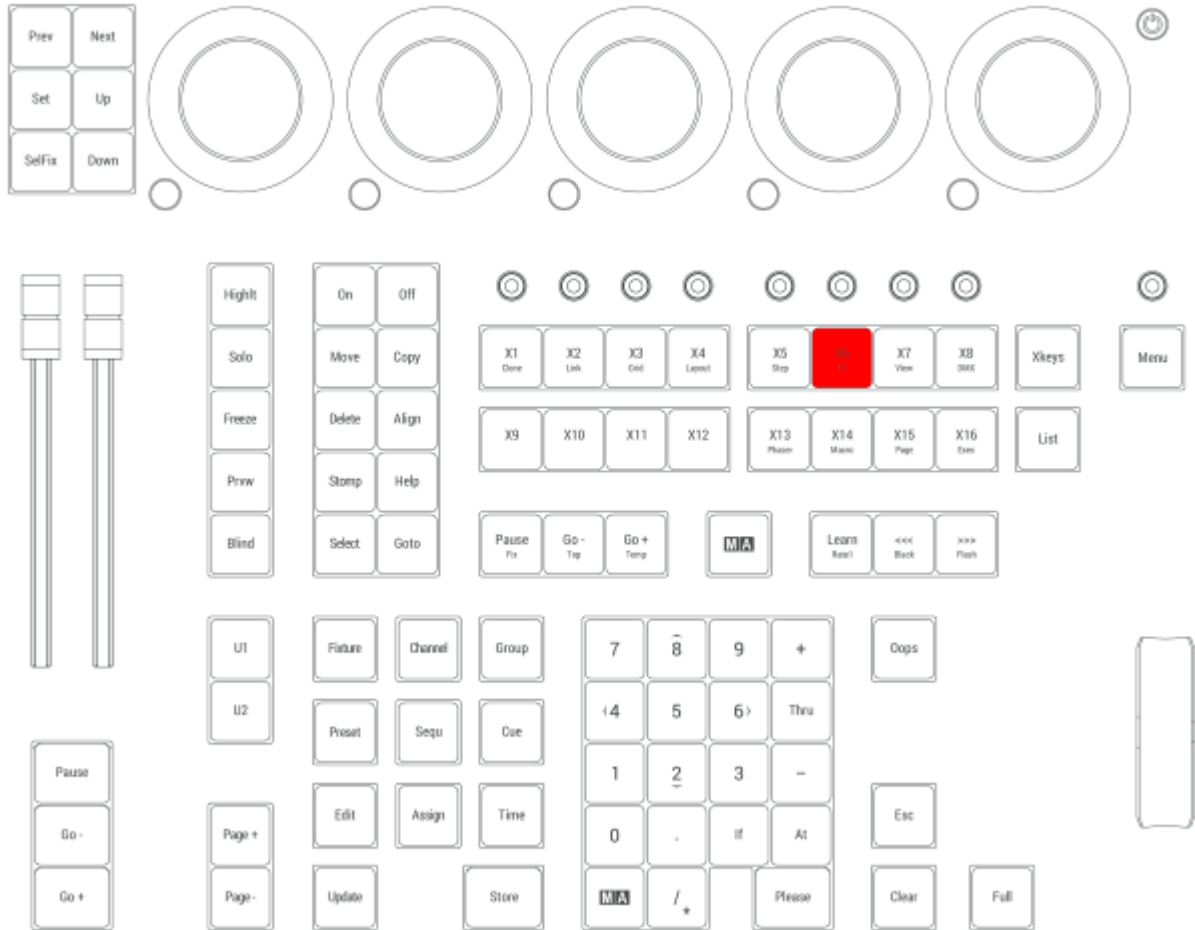


For more information about Timer, see the [Timer keyword](#) topic.

X6 is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.70.X7 | View Key

All Xkeys behave like executors. **X7** is executor 297.

View

Pressing and holding **MA** + **X7** enters the View keyword into the command line.

```
MA User name[Fixtures]> View
```

For more information about View, see the [View keyword](#) topic.

ViewButton

Pressing and holding **MA** + **X7** + **X7** enters the ViewButton keyword into the command line.

```
MA User name[Fixtures]> ViewButton
```

For more information about ViewButton, see the [ViewButton keyword](#) topic.

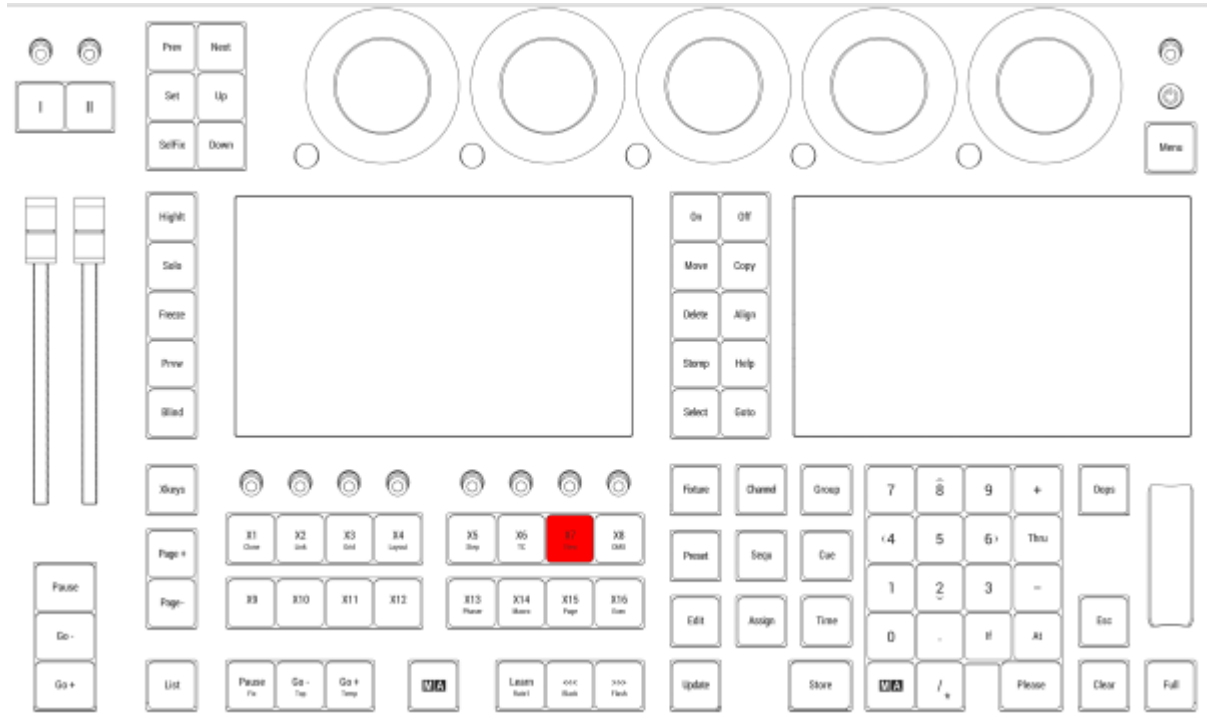
ScreenContent

Pressing and holding **MA** + **X7** + **X7** + **X7** enters the ScreenContent keyword into the command line.

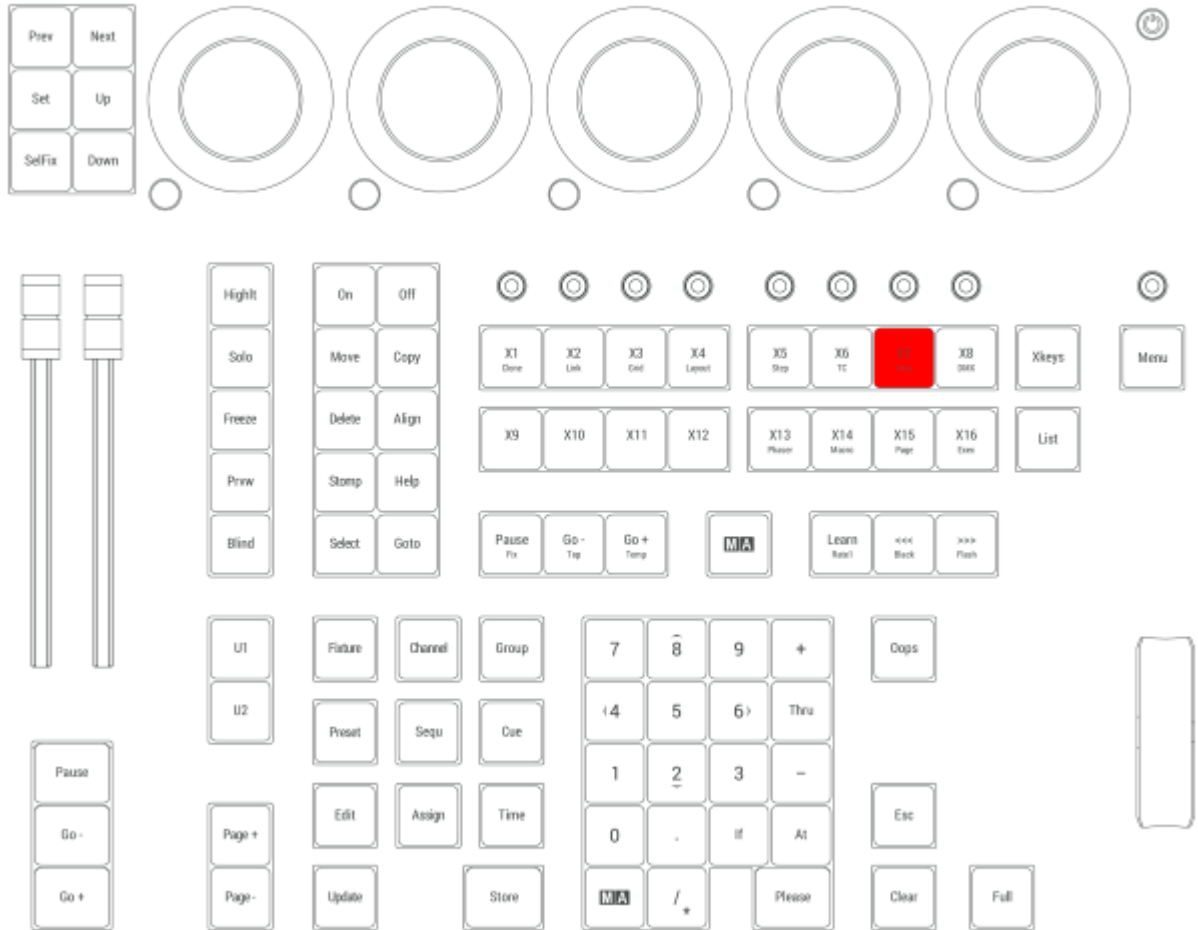
```
MA User name[Fixtures]> ScreenContent
```

For more information about ScreenContent, see the [ScreenContent keyword](#) topic.

X7 is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

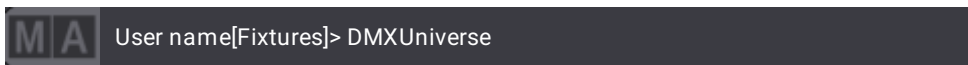


2.15.71. X8 | DMX Key

All Xkeys behave like executors. **X8** is executor 298.

DMXUniverse

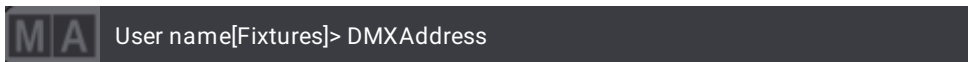
Pressing and holding **MA** + **X8** enters the DMXUniverse keyword into the command line.



For more information about DMXUniverse, see the [DMXUniverse keyword](#) topic.

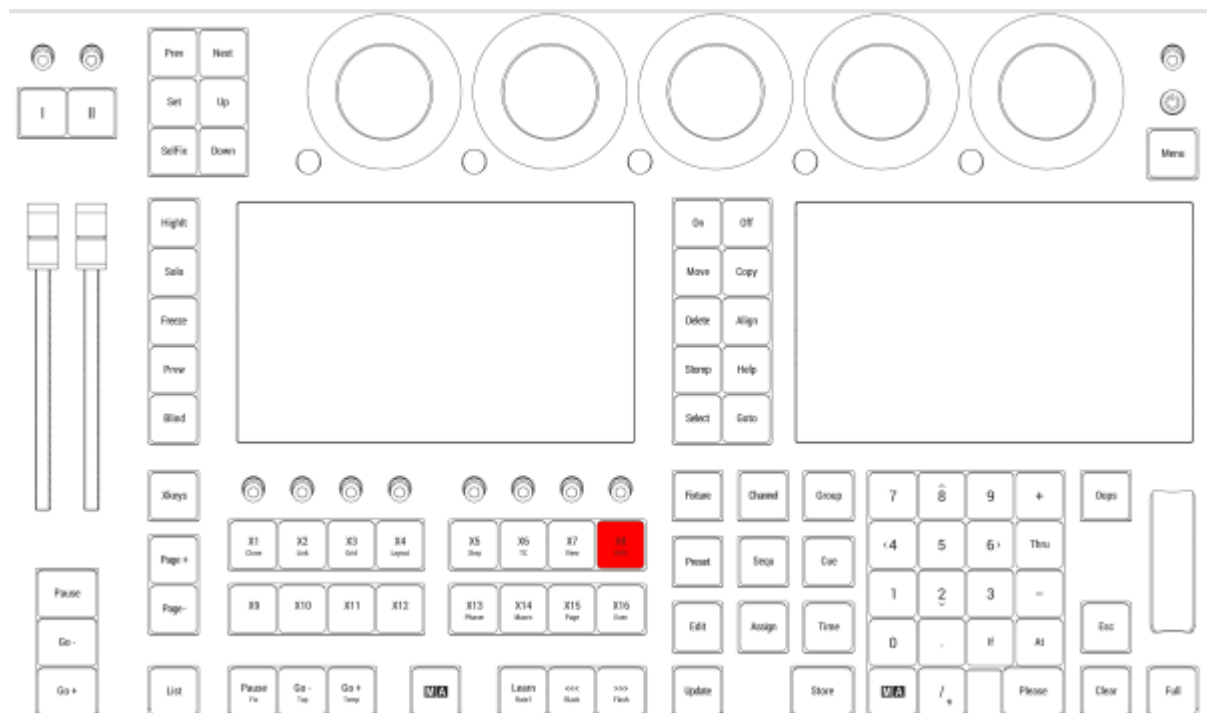
DMXAddress

Pressing and holding **MA** + **X8** + **X8** enters the DMXAddress keyword into the command line.

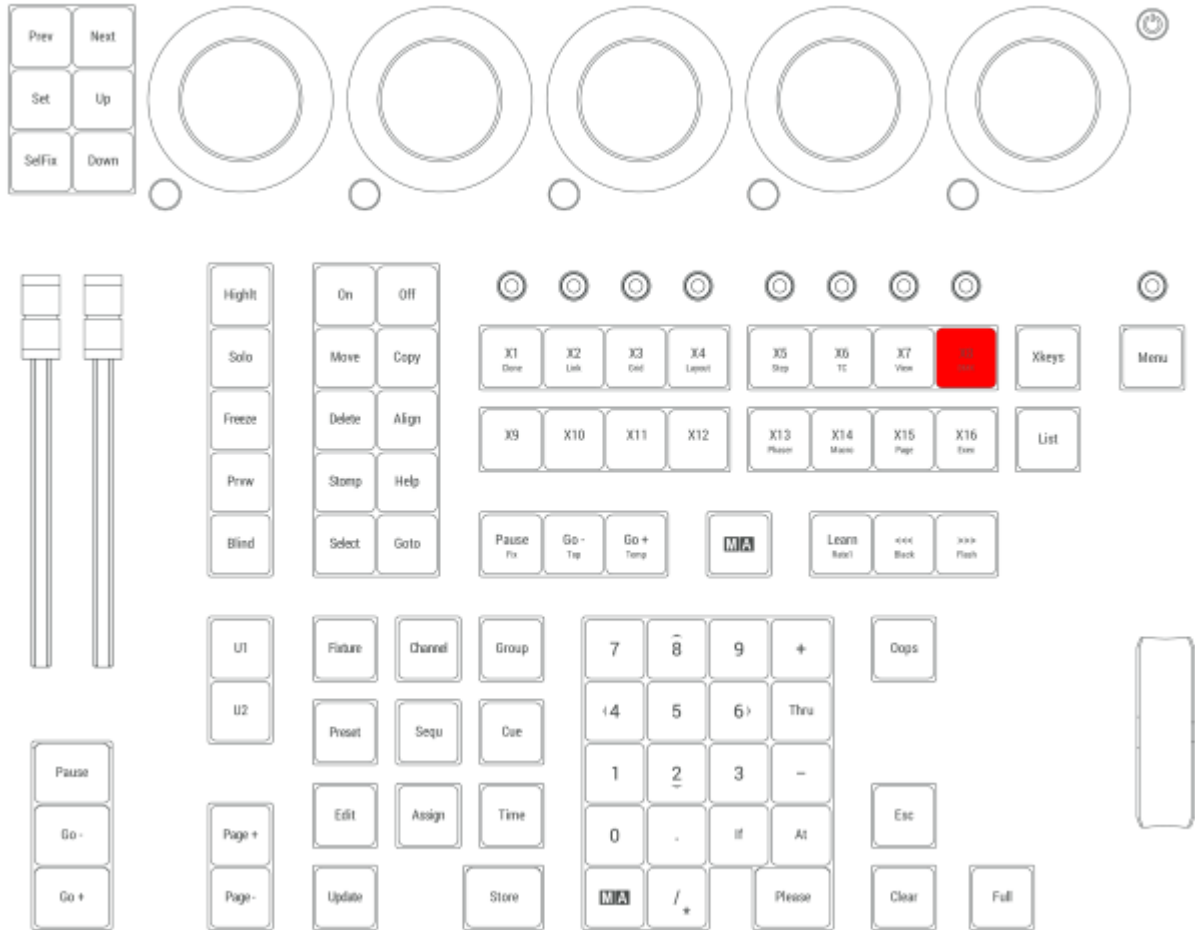


For more information about DMXAddress, see the [DMXAddress keyword](#) topic.

X8 is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles

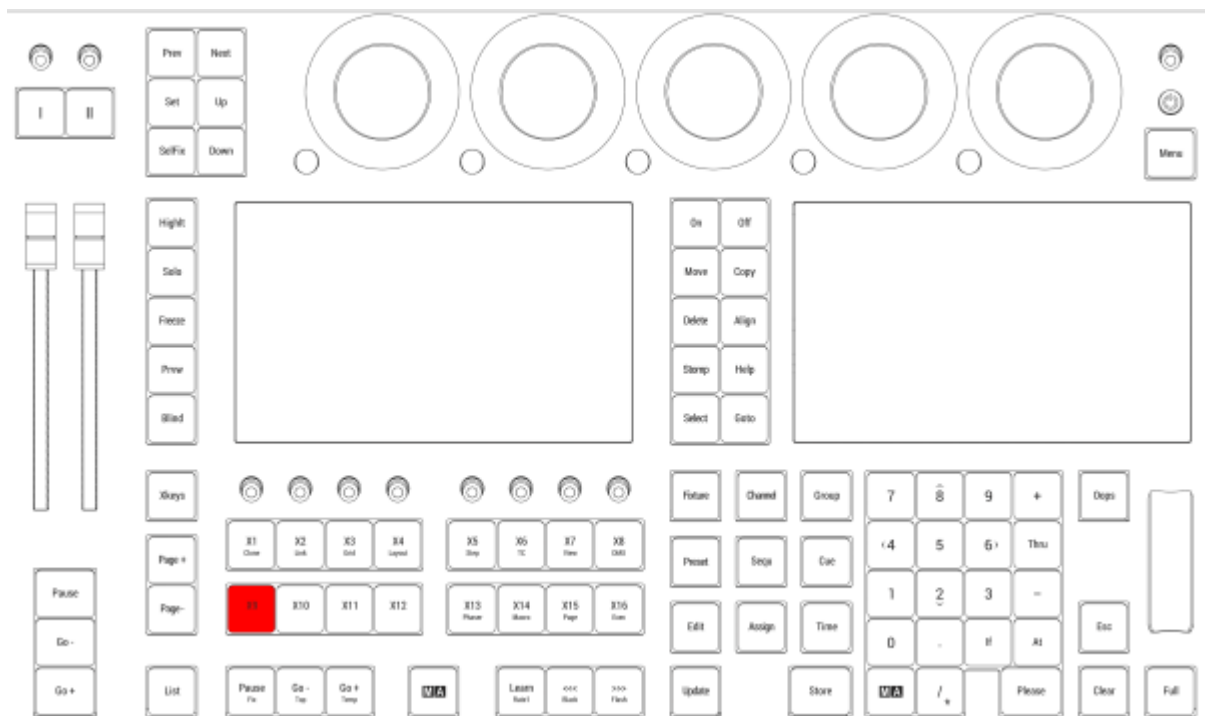


Location on grandMA3 compact consoles and grandMA3 onPC command wing

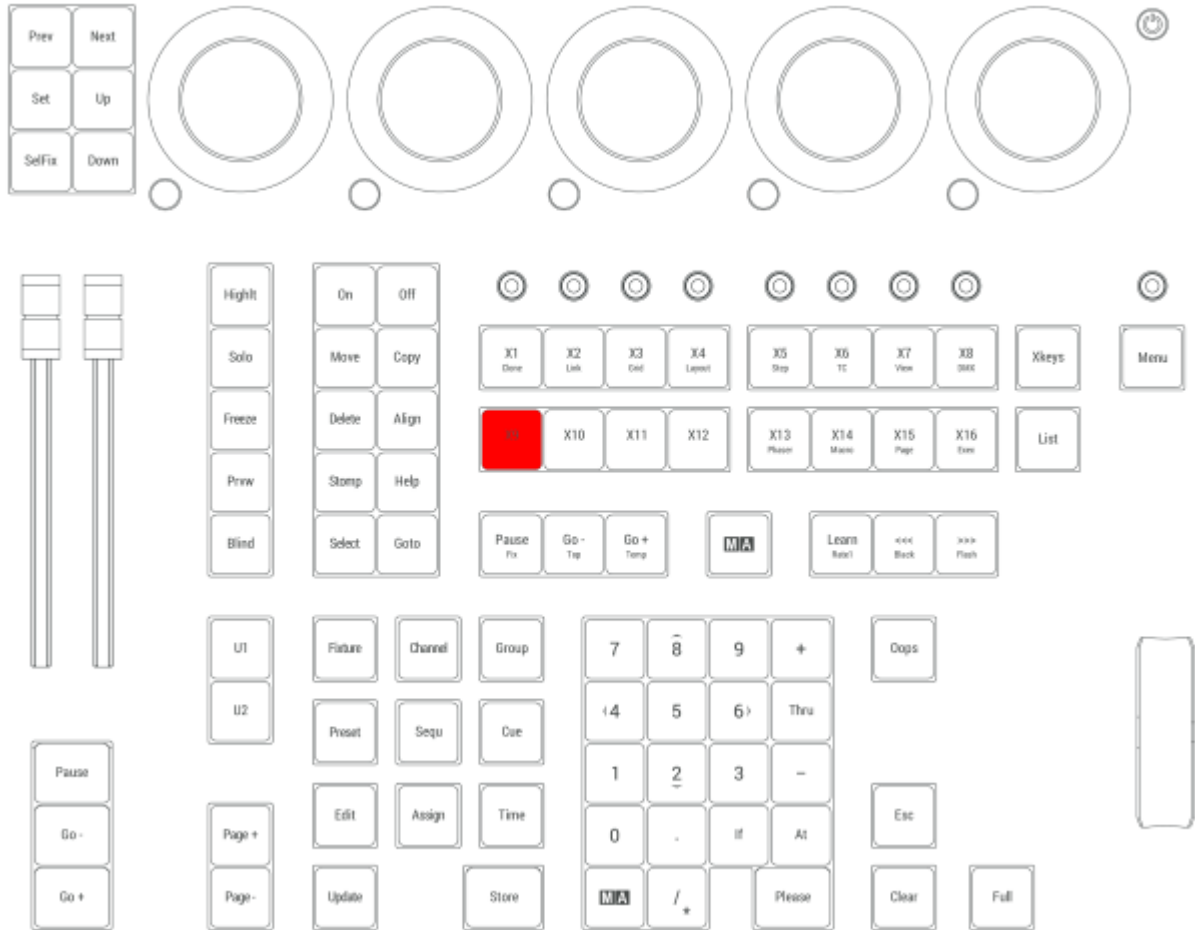
2.15.72. X9 Key

All Xkeys behave like executors. **X9** is executor 191.

X9 is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles

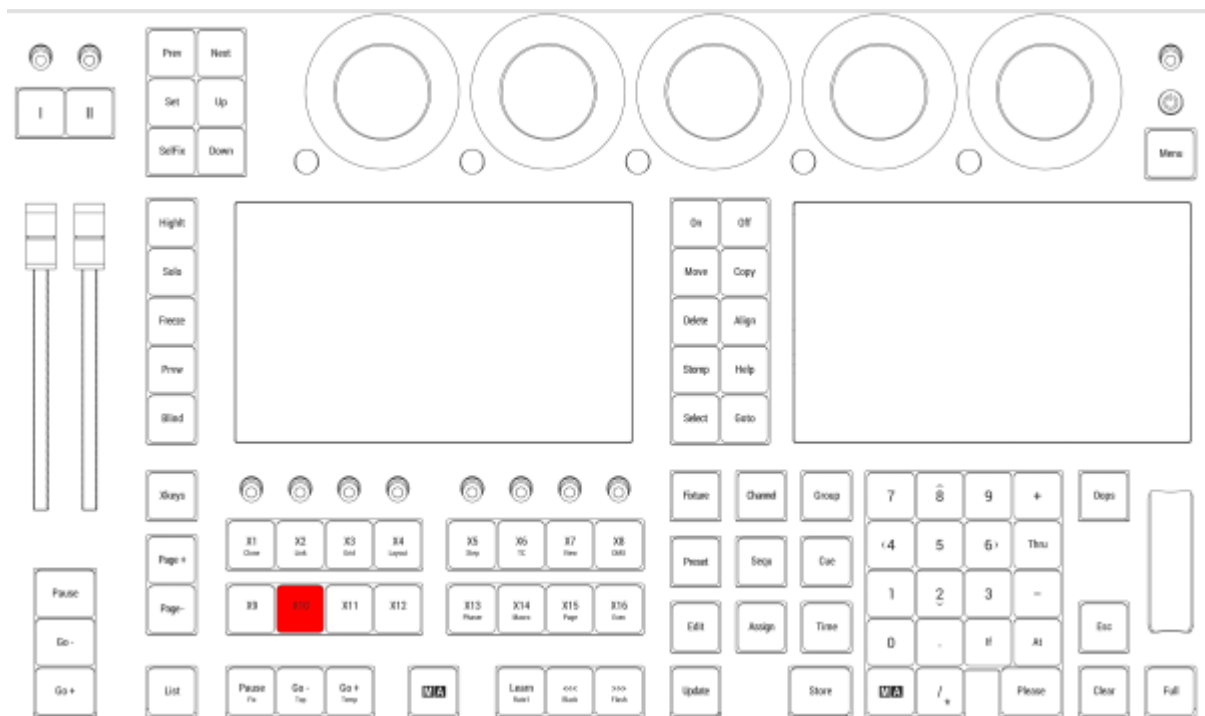


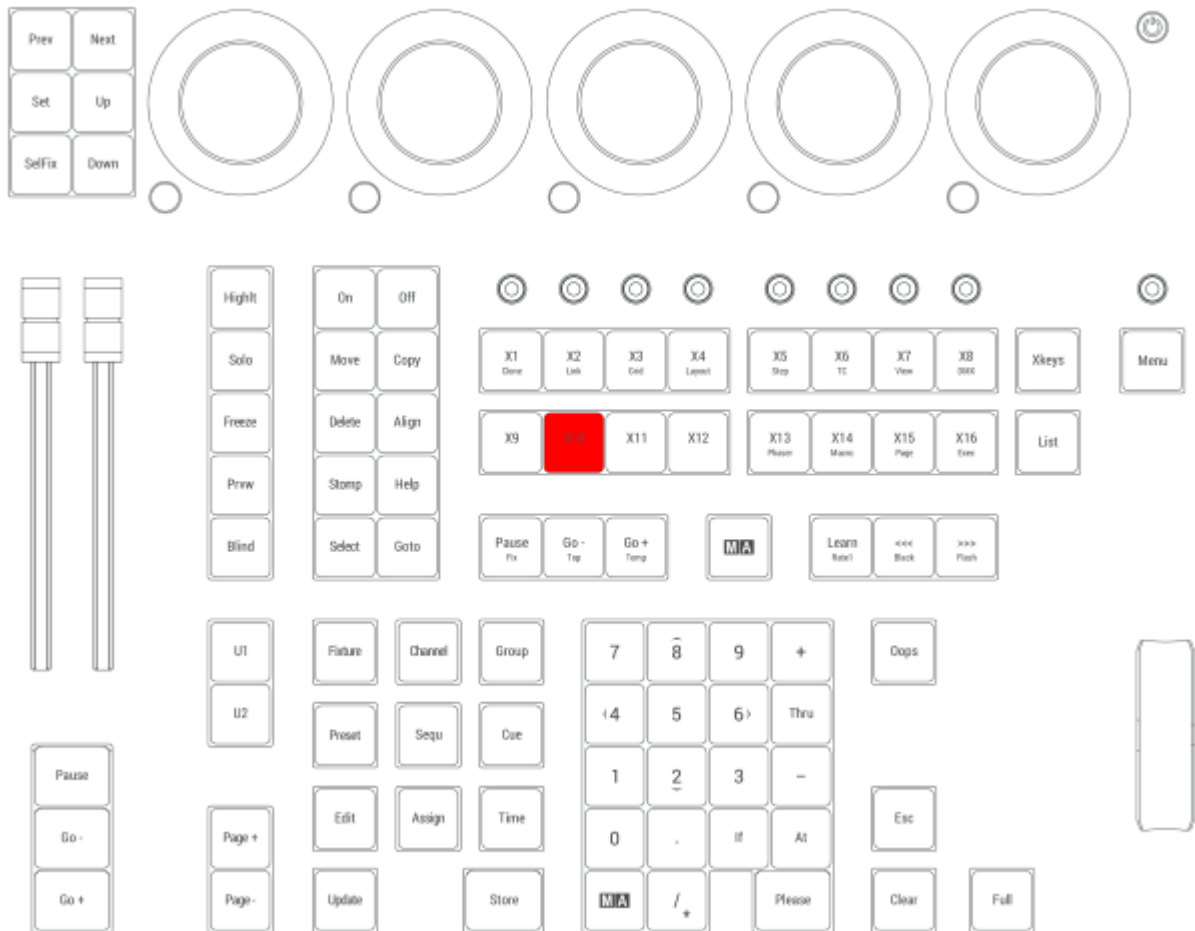
Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.73. X10 key

At the moment all Xkeys behave like executors. **X10** is executor 192.

X10 is located in the command section.



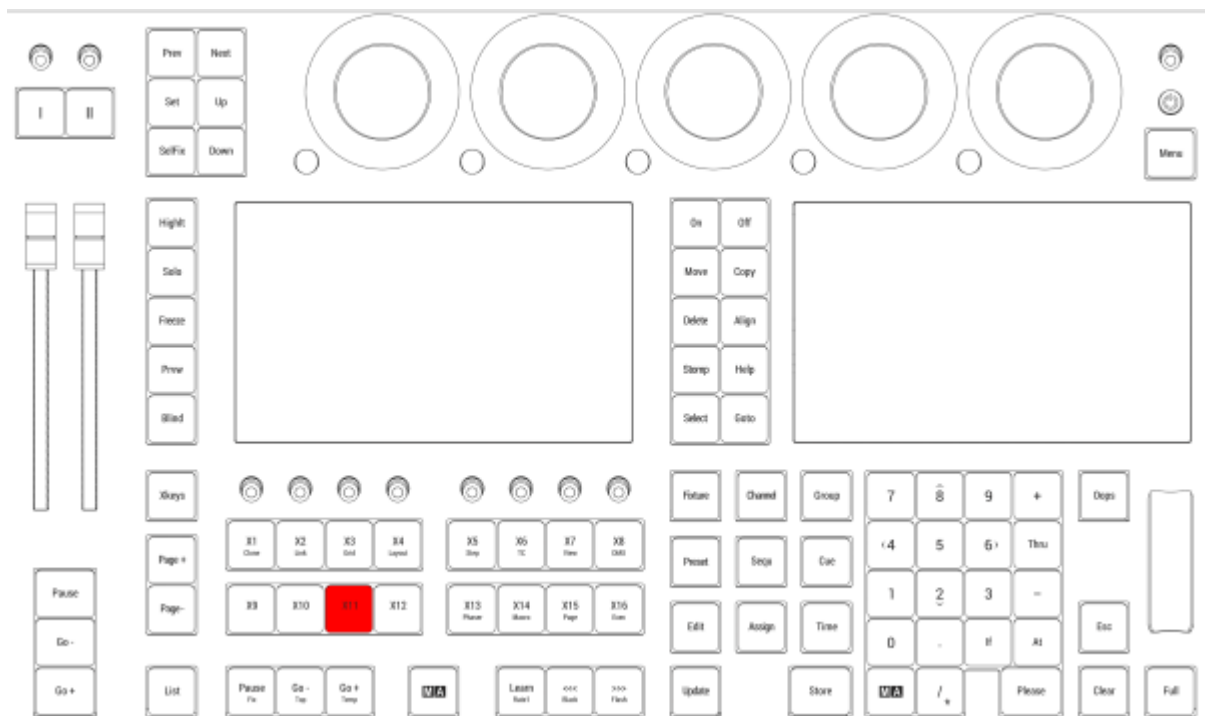


Location on grandMA3 compact consoles and grandMA3 onPC command wing

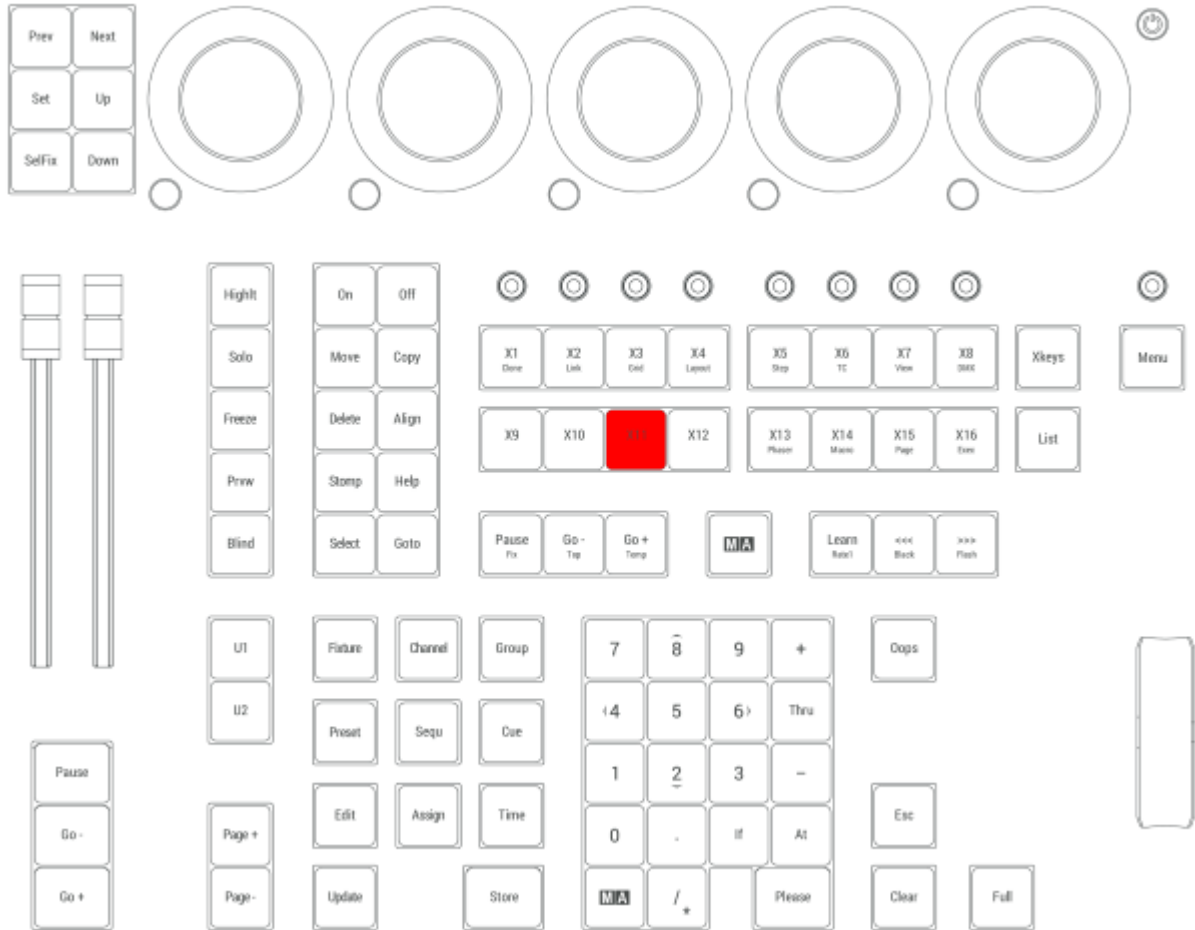
2.15.74.X11 Key

All Xkeys behave like executors. **X11** is executor 193.

X11 is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



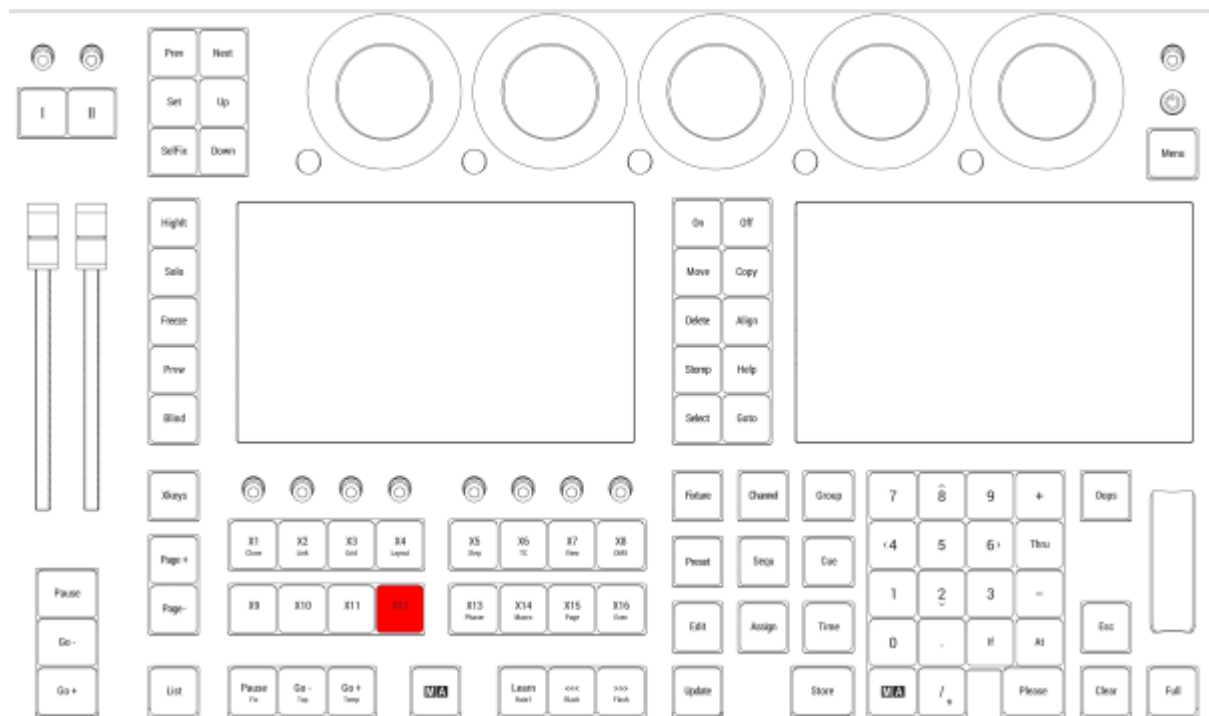
Location on grandMA3 compact consoles and grandMA3 onPC command wing



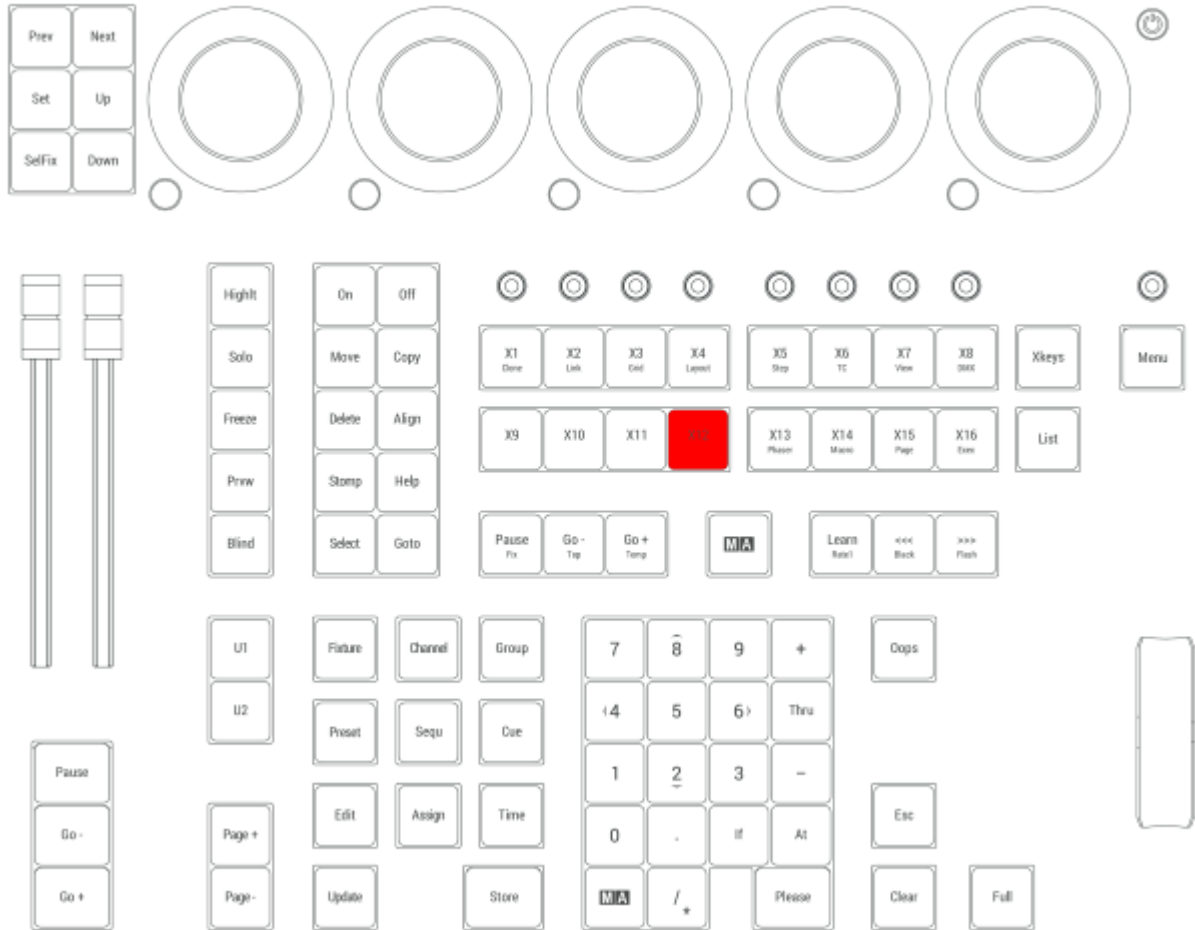
2.15.75. X12 Key

All Xkeys behave like executors. **X12** is executor 194.

X12 is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.76. X13 | Phaser Key

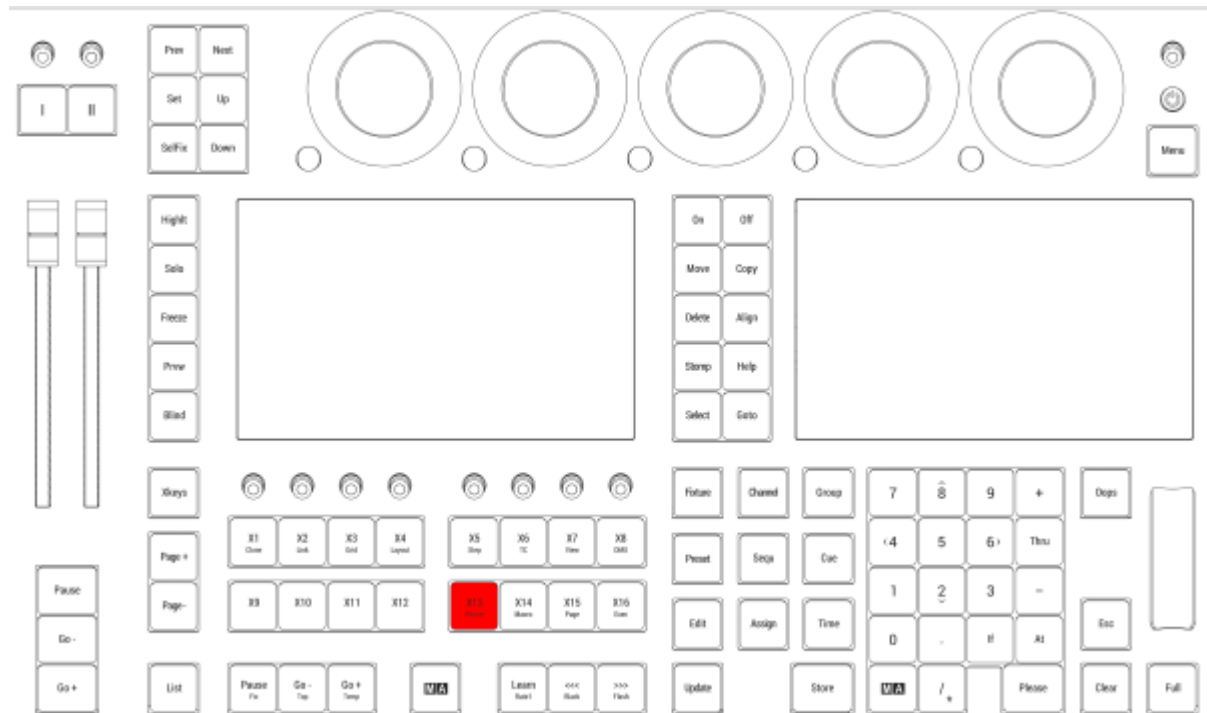
All Xkeys behave like executors. **X13** is executor 195.

Phaser

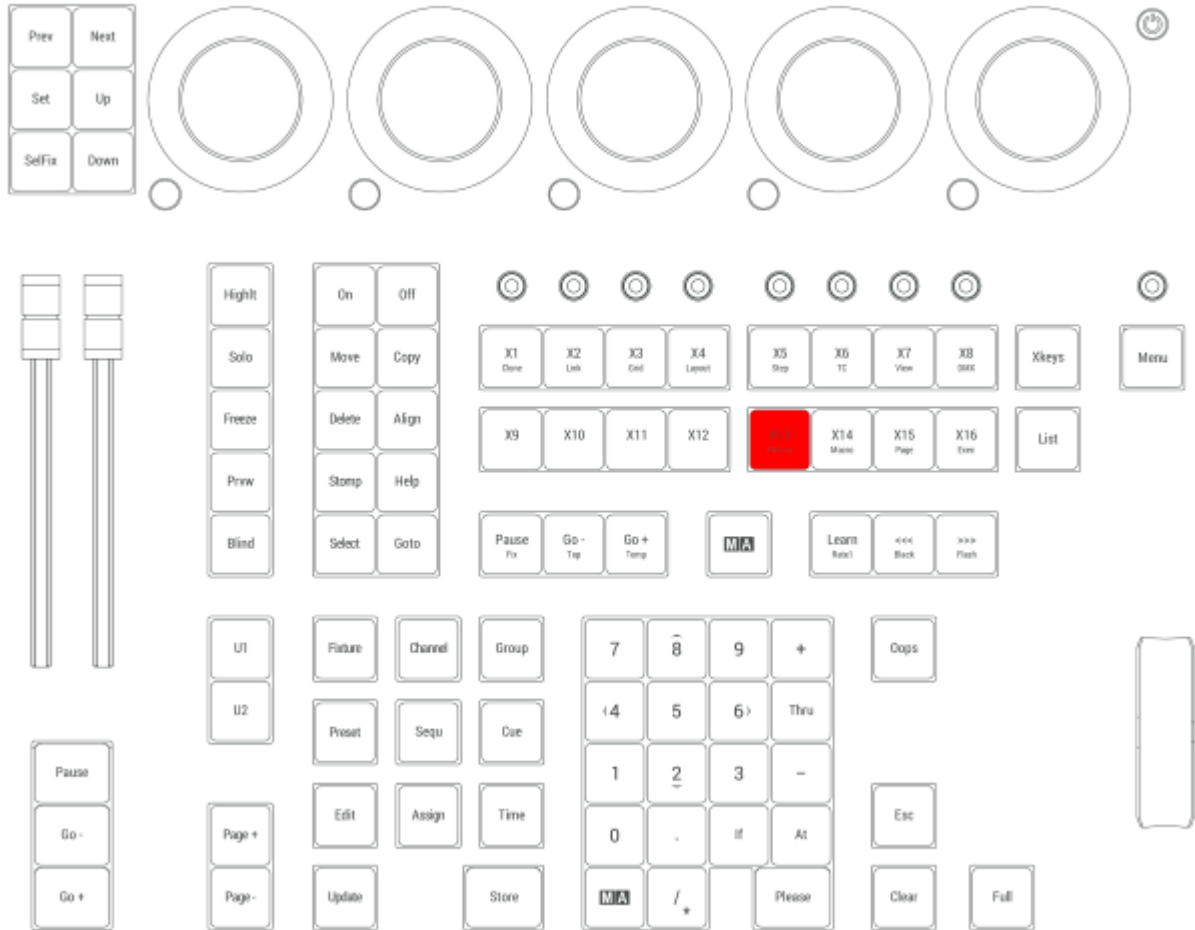
Pressing and holding **MA** + **X13** opens the **Phaser Editor**.

For more information about Phasers, see the topics **Phasers** and **Phaser Editor**.

X13 is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.77. X14 | Macro Key

All Xkeys behave like executors. **X14** is executor 196.

Macro

Pressing and holding **MA** + **X14** enters the Macro keyword into the command line.



For more information about Macro, see the [Macro keyword](#) topic.

Plugin

Pressing and holding **MA** + **X14** + **X14** enters the Plugin keyword into the command line.



For more information about Plugin, see the [Plugin keyword](#) topic.

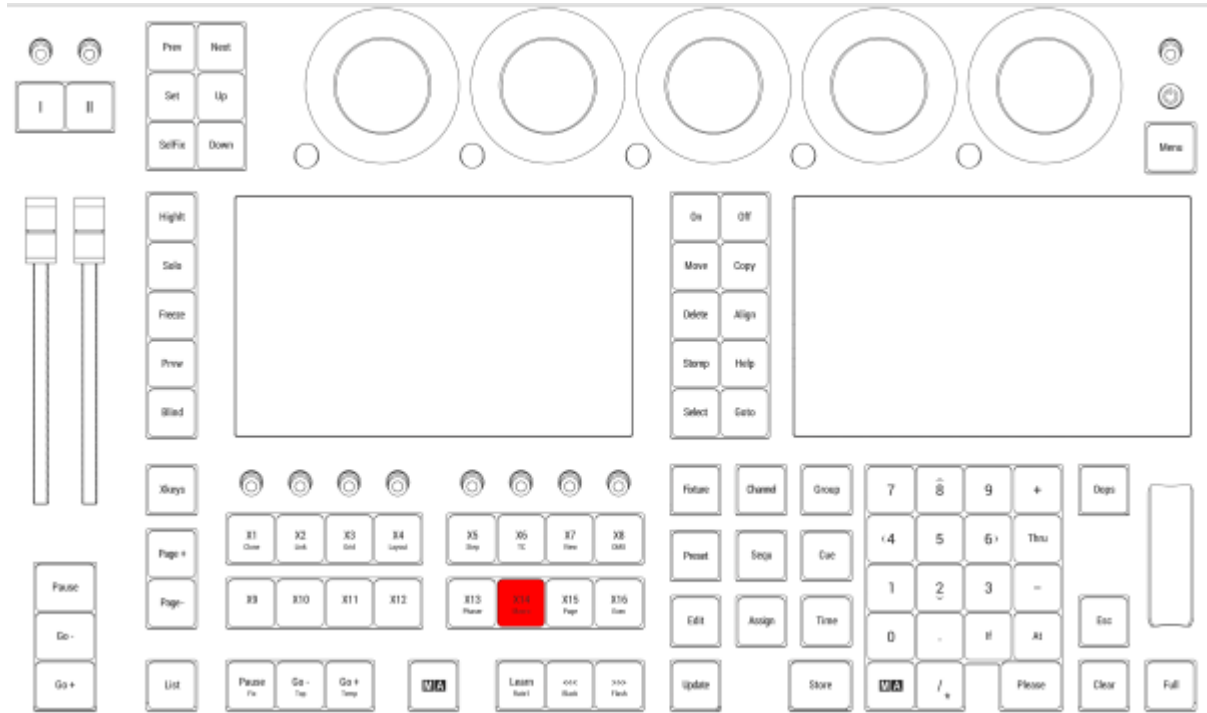
Quickey

Pressing and holding **MA** + **X14** + **X14** + **X14** enters the Quickey keyword into the command line.

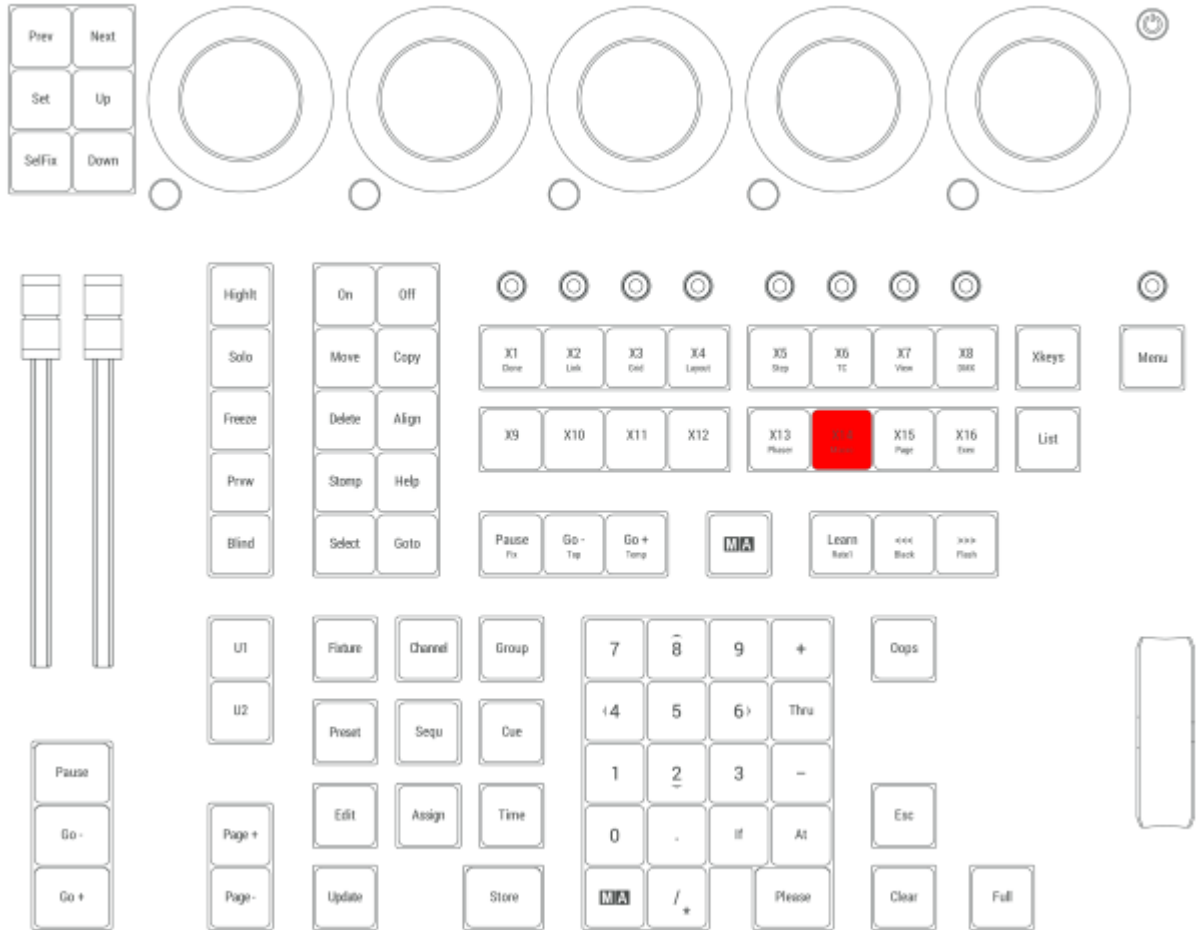


For more information about Quickey, see the [Quickey keyword](#) topic.

X14 is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.78.X15 | Page Key

All Xkeys behave like executors. **X15** is executor 197.

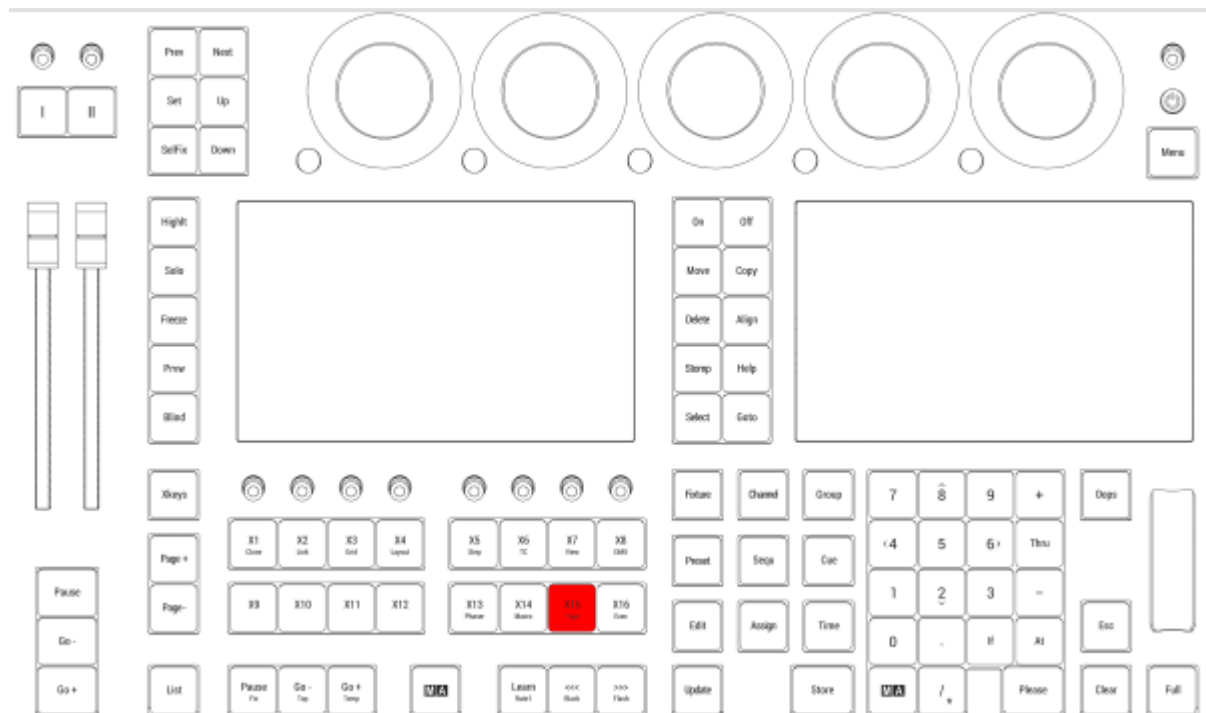
Page

Pressing and holding **MA** + **X15** enters the Page keyword into the command line.

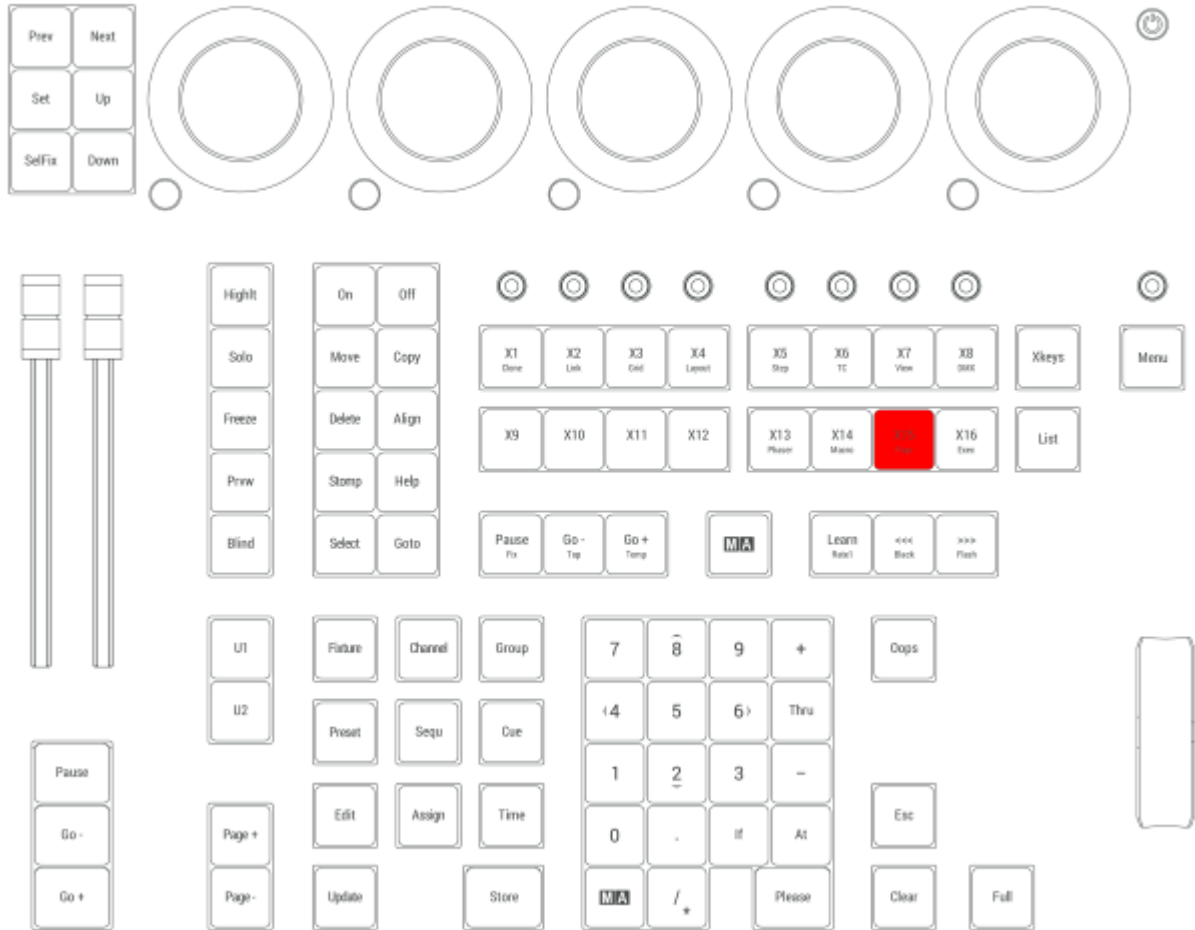


For more information about Page, see the **Page keyword** topic.

X15 is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing



2.15.79. X16 | Exec Key

All Xkeys behave like executors. **X16** is executor 198.

Executor

Pressing and holding **MA** + **X16** enters the Executor keyword into the command line.

```
MA User name[Fixtures]> Executor
```

For more information about Executor, see the [Executor keyword](#) topic.

SpecialExecutor

Pressing and holding **MA** + **X16** + **X16** enters the SpecialExecutor keyword into the command line.

```
MA User name[Fixtures]> SpecialExecutor
```

For more information about SpecialExecutor, see the [SpecialExecutor keyword](#) topic.

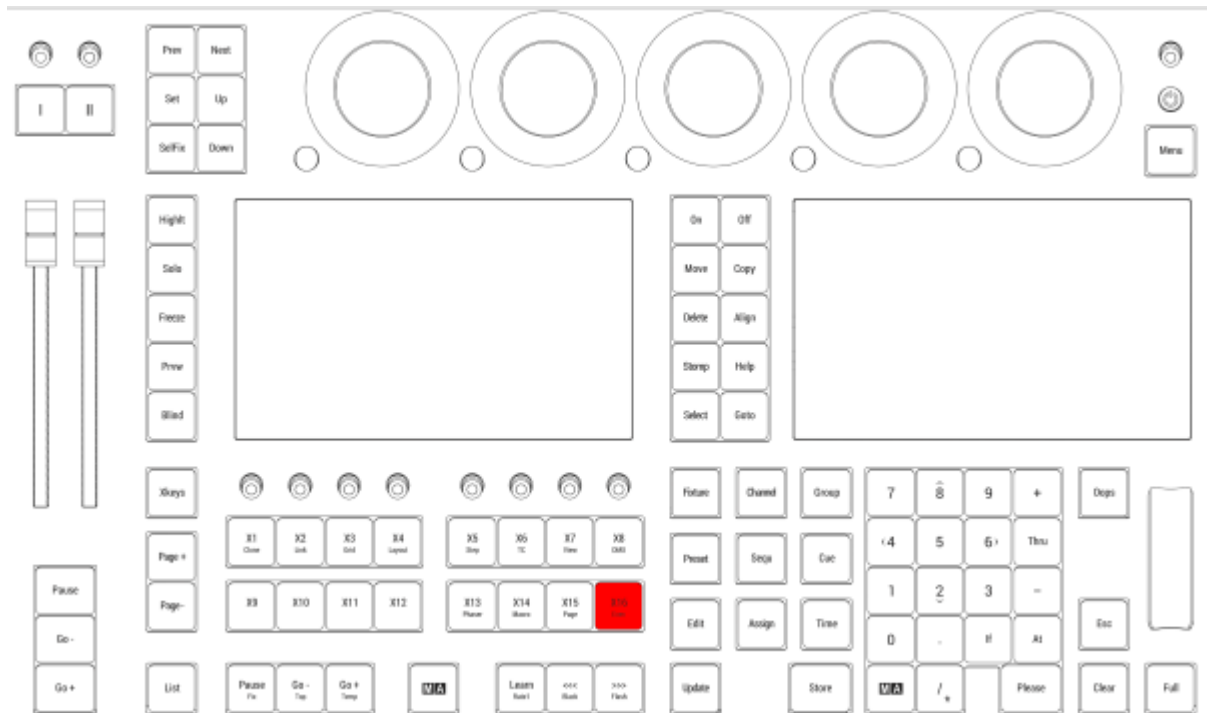
FaderMaster

Pressing and holding **MA** + **X16** + **X16** + **X16** enters the FaderMaster keyword into the command line.

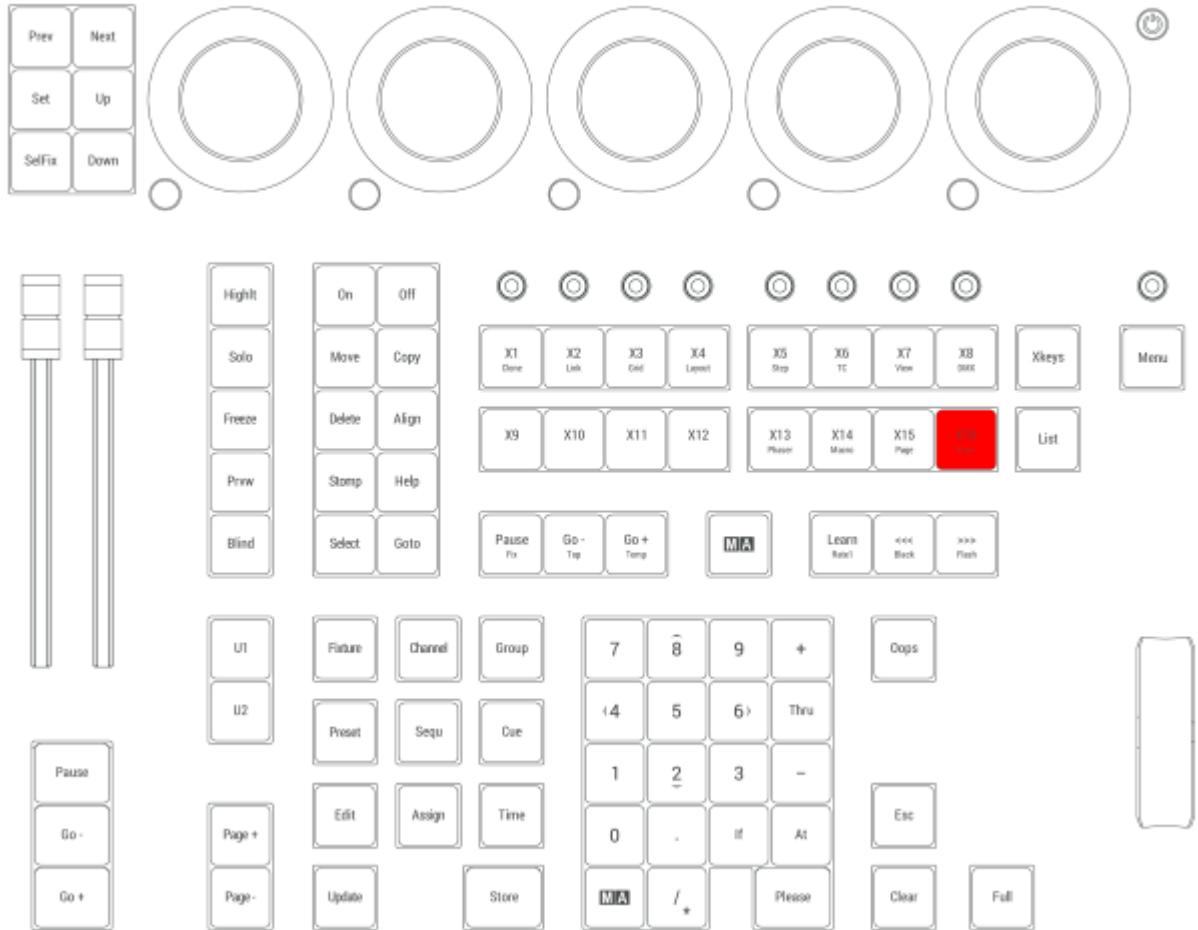
```
MA User name[Fixtures]> FaderMaster
```

For more information about FaderMaster, see the [FaderMaster keyword](#) topic.

X16 is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



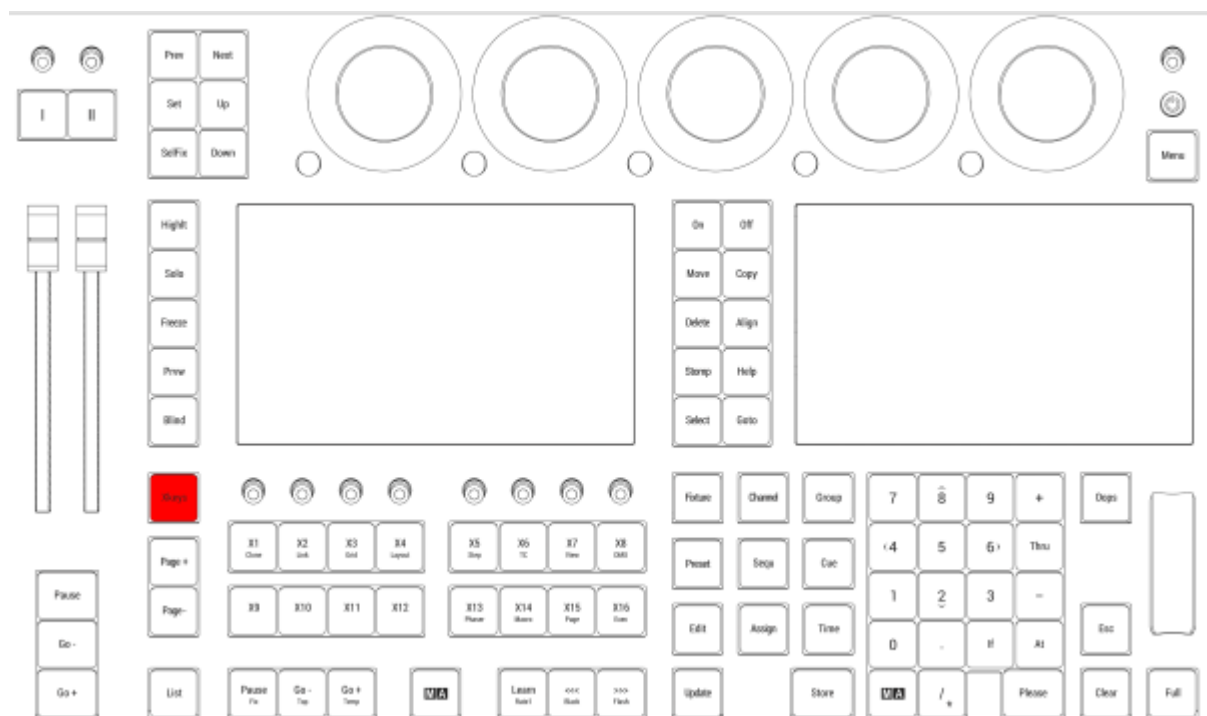
Location on grandMA3 compact consoles and grandMA3 onPC command wing

2.15.80. Xkeys Key

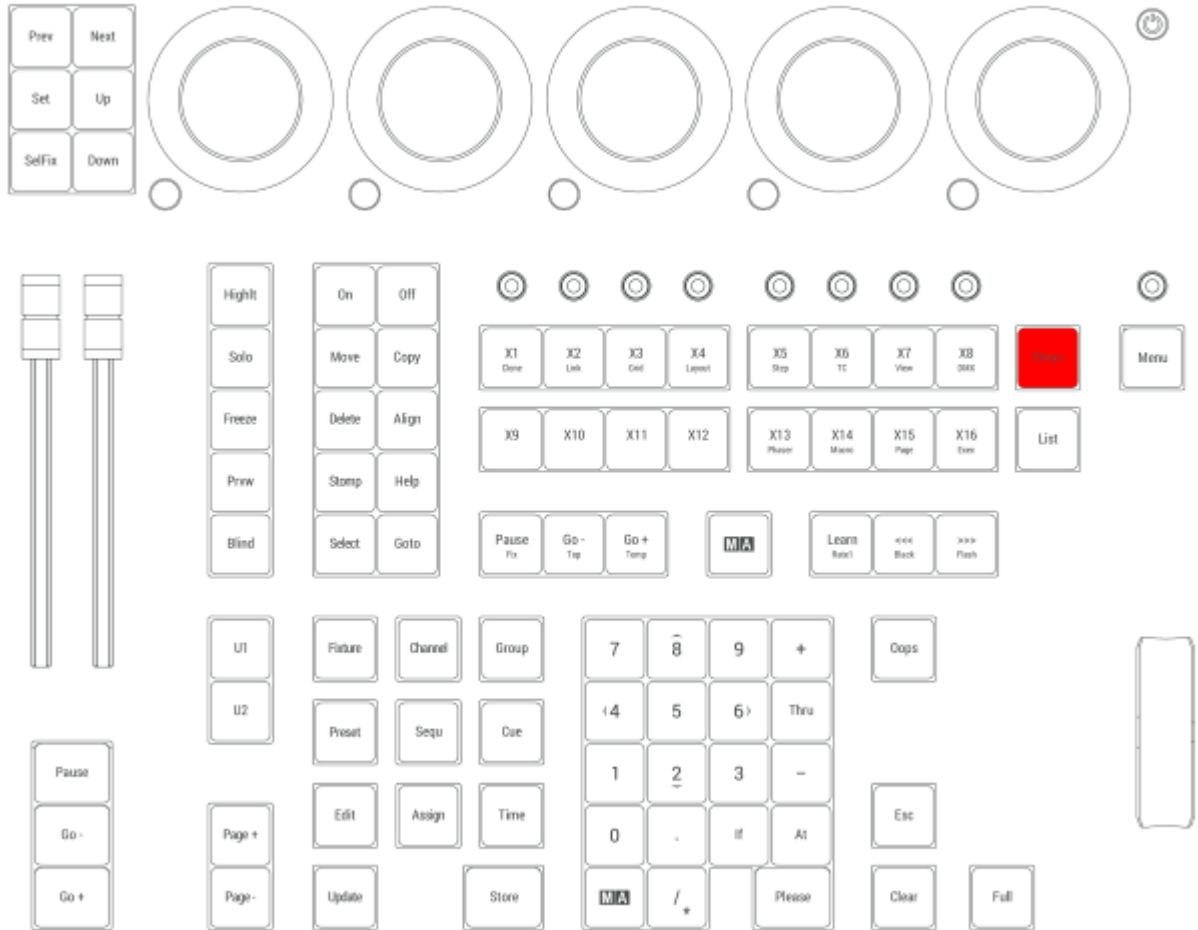
Pressing **Xkeys** executes the Xkeys keyword into the command line.



Xkeys is located in the command section.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing

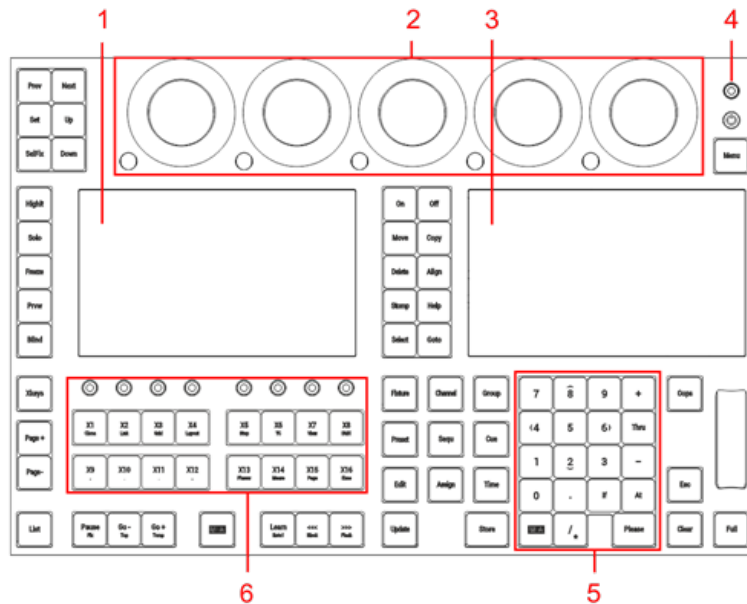


2.16. Control Elements

The following chapters describe the different control elements of the grandMA3 devices.

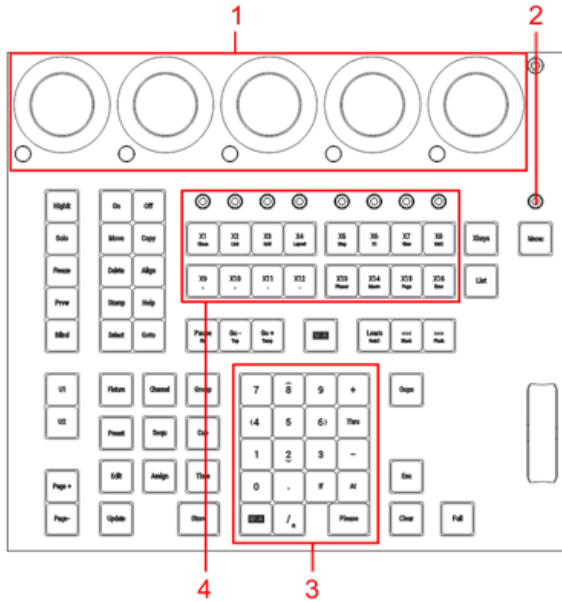
- Each chapter uses drawings of the grandMA3 product for visualization.
- Control elements described in the chapters are marked in red.

2.16.1. Command Area



Command area of grandMA3 full-size (CRV) and grandMA3 light (CRV)

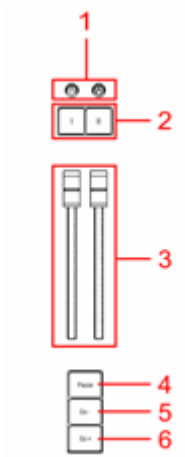
1. Left command screen
2. **Dual encoder section**
3. Right command screen
4. **Grand master**
5. Number pad
6. **Xkeys section**



Command area of grandMA3 compact (XT) and grandMA3 onPC command wing (XT)

1. **Dual encoder section**
2. **Grand master**
3. Number pad
4. **Xkeys section**

2.16.2. Master Area



Master area grandMA3 full-size (CRV)

1. Master knobs 1+2
2. Master keys 1+2
3. Master faders 1+2
4. Pause key
5. Go- key
6. Go+ key

Go+, Go- and Pause

The Go+ [Large] key, the Go- [Large] key, and the Pause [Large] key are located in the lower part of the master area. These keys execute the Go+, Go- or Pause command for the selected sequence.

For more information about Go+, Go- and Pause, please see the [Go+\[Large\] key](#), [Go-\[Large\] key](#), or the [Pause\[Large\] key](#) topic.

Master Fader 1 and Master Fader 2


The two faders in the master area are master fader 1 and master fader 2.

By default, master fader 1 is set to the master function for the selected sequence.

By default, master fader 2 is set to the crossfade function for the selected sequence.

- To change the function of the faders, please read below.
-

Master Key 1 and Master Key 2, Master Knob 1 and Master Knob 2

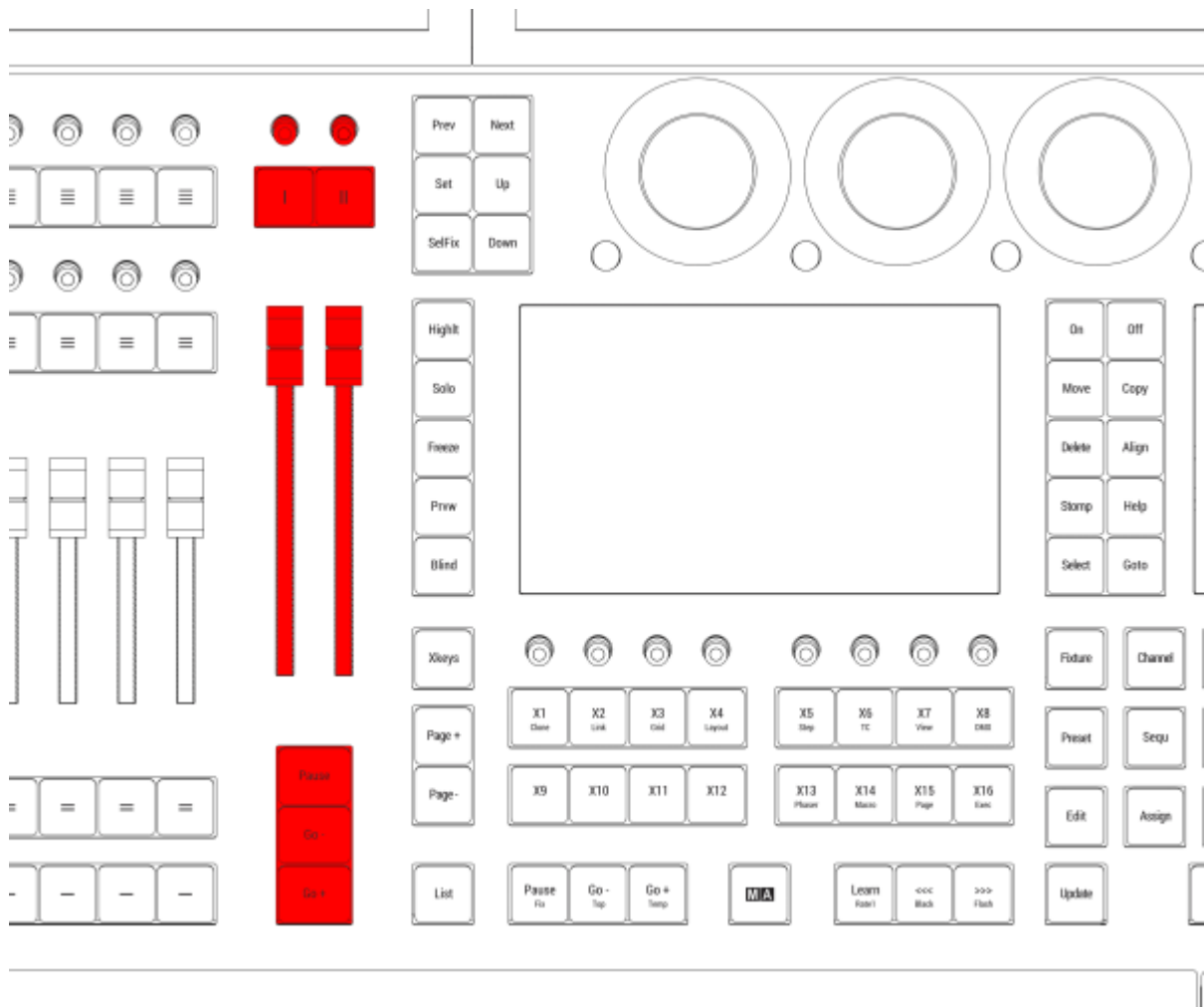
 **Restriction:**
 Master keys and master knobs are only available on grandMA3 full-size (CRV) and grandMA3 light (CRV) consoles.

The master keys 1+2 and the master knobs 1+2 are located in the upper part of the master area.

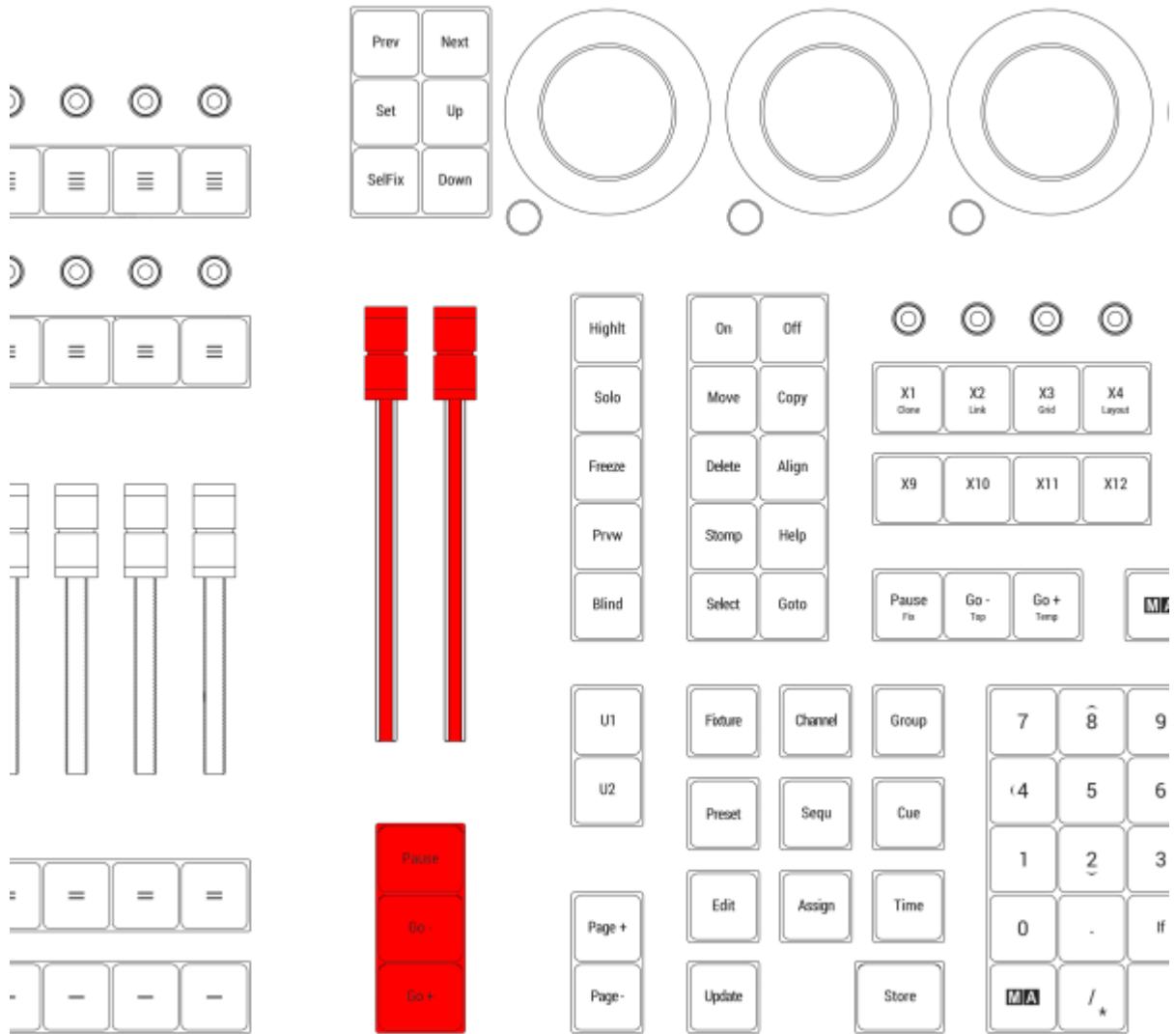
By default, these are set to toggle **Highlight** and **Solo** on and off and set the intensity of these.

- To change the function of these buttons and knobs, please read the topic **Special Executors**.

The master area is located between the **command area** and the **executor area**.




Location on grandMA3 full-size (CRV) and grandMA3 light (CRV) consoles

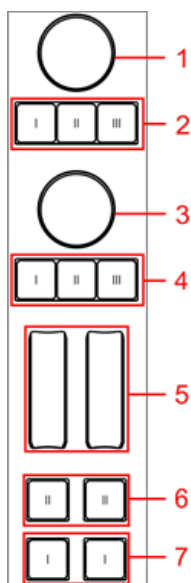


Location on grandMA3 compact consoles and grandMA3 onPC command wings

2.16.3. Custom Area

 **Restriction:**
The custom area is only available on grandMA3 full-size, grandMA3 full-size CRV and grandMA3 extension consoles.

Custom Area



Custom area

1. Upper encoder
2. Upper encoder keys 1-3
3. Lower encoder
4. Lower encoder keys 1-3
5. Wheels 1+2
6. Wheels 1+2 upper keys
7. Wheels 1+2 lower keys

To change the function of these control elements, please read the topic [Special Executors](#).

2.16.4. Dual Encoders

The five dual encoders are used to adjust the different attributes of the fixtures. There are four different actions on the encoders:

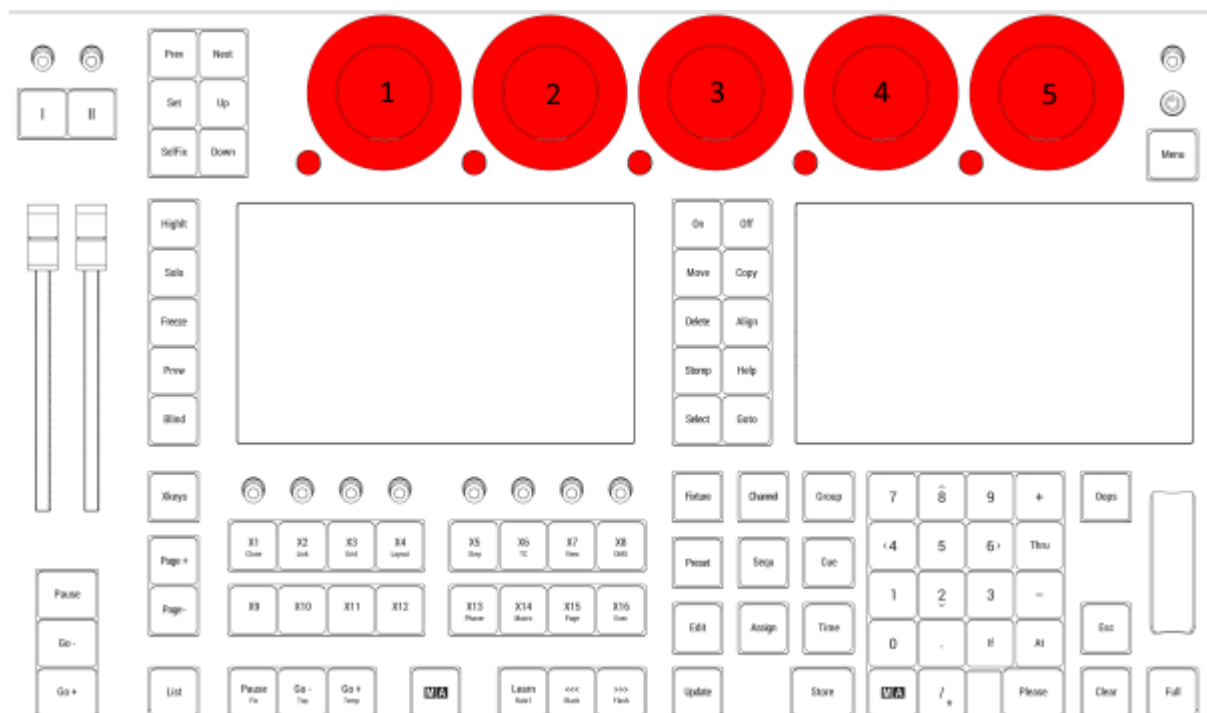
- Turn the inner ring to adjust the attribute "Coarse".
- Turn the outer ring to adjust the attribute "Fine".
- Press the inner ring to open a calculator to type in the value.
- Press and turn the inner ring to adjust the attribute "Coarse" with a higher speed.
- Press the dual encoder key to open a calculator to type in the value. The dual encoder key replaces the press function of the outer ring.



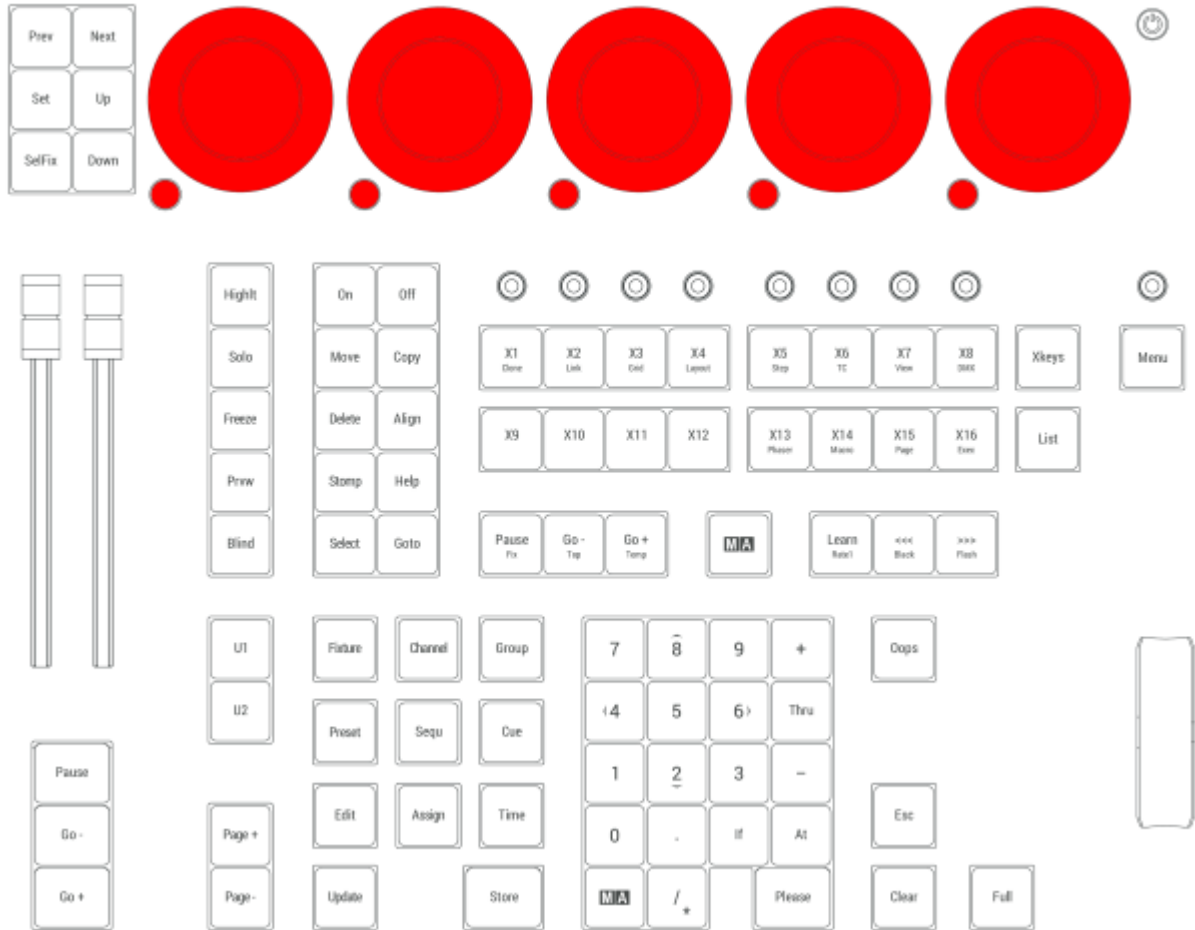
Hint:

For more information on the Readout and Resolution of the encoders, see [Readout Keyword](#) and [Encoder Resolution](#) topic.

These encoders are placed above the command section.



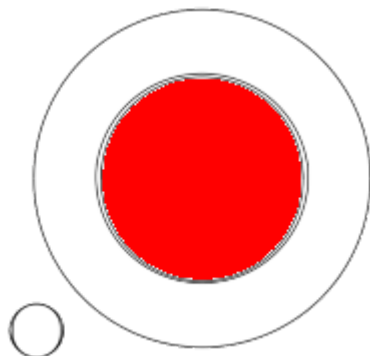
Location on grandMA3 full-size and grandMA3 light consoles



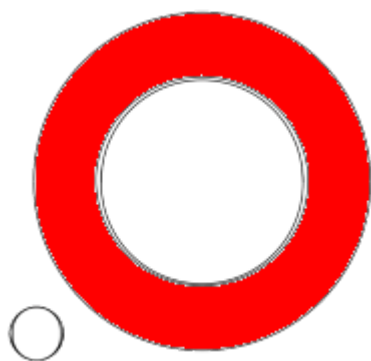
Location on grandMA3 compact consoles and grandMA3 onPC command wing and command wing XT

Each of the dual encoders is numbered 1 to 5 from left to right. There are three elements in each encoder:

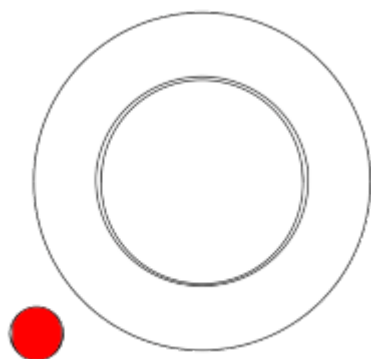
The inner ring



The outer ring



The dual encoder key



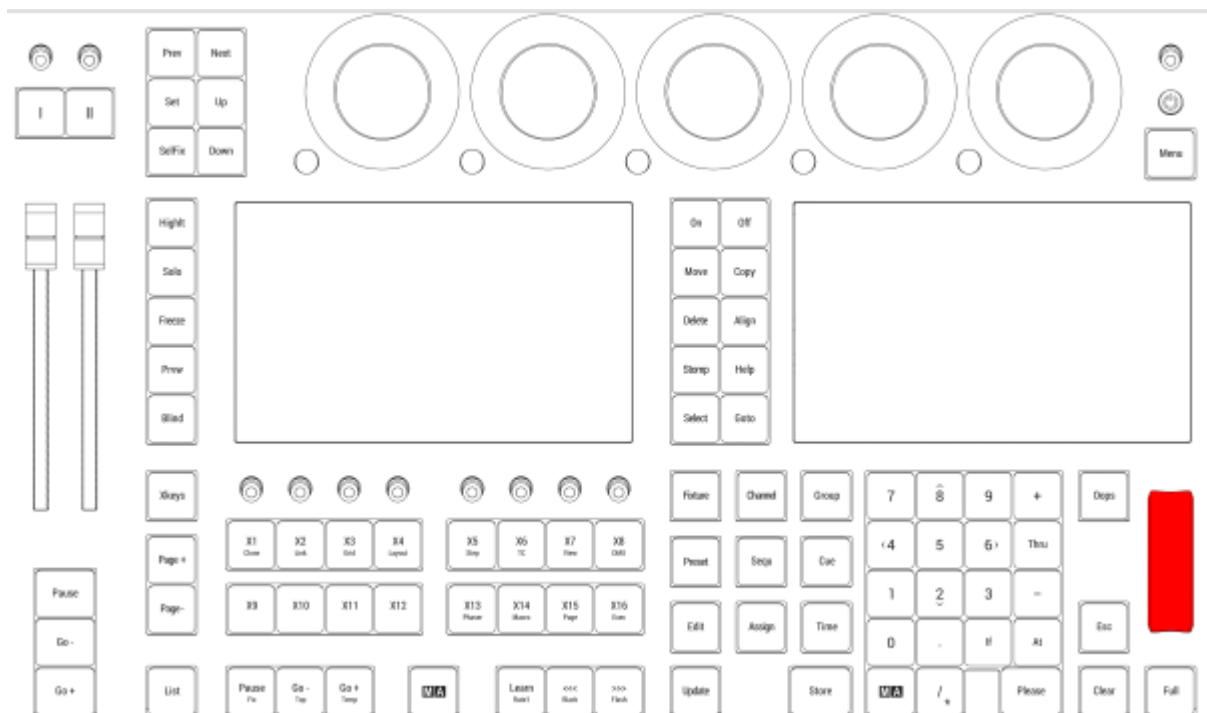


2.16.5. Level Wheel

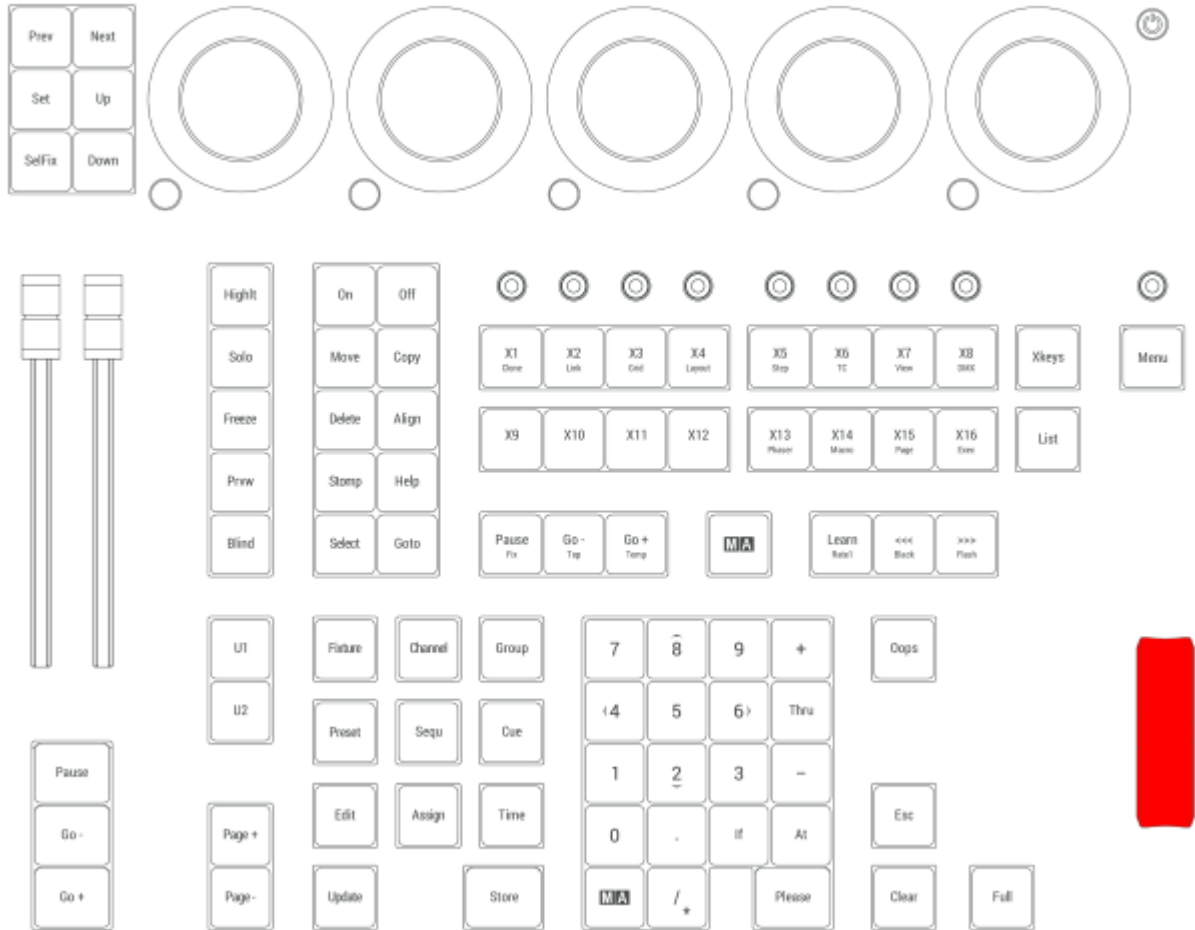
The level wheel is used to adjust the intensity of the selected fixtures.

The level wheel is placed on the right side of the numeric keys.

- To adjust the wheel, see Wheel Mode and Wheel Resolution in the **User settings**.



Location on grandMA3 full-size and grandMA3 light consoles



Location on grandMA3 compact consoles and grandMA3 onPC command wing/command wing XT

2.16.6. Grand Master

The Grand Master is used to limit the output of the intensity of all the fixtures patched in the show.

By default, turning the knob adjusts intensity. This can be changed in the Special Executor Configuration menu.

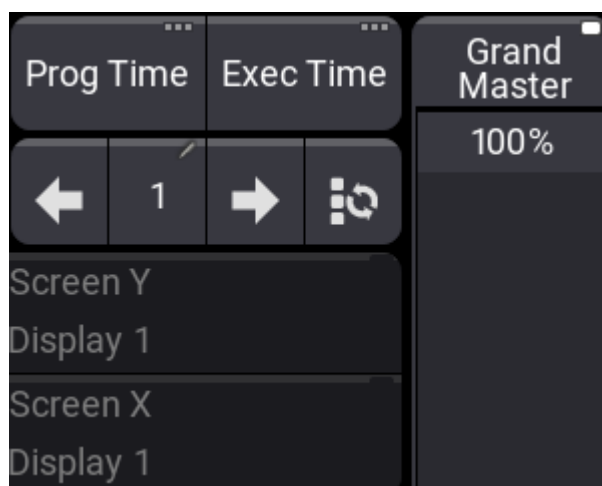
To access the Special Executor Configuration menu:

1. Press **Assign**.
2. Press Grand Master knob.

This menu can also be opened using the SpecialExecutor keyword in the command line. For more information about SpecialExecutor, see the [SpecialExecutor keyword](#) topic.

To read more about Executor Configuration, see the [Executor Configuration](#) topic.


The Grand Master is also displayed on the right side of the encoder bar on screen 1:



Grand Master in the encoder bar

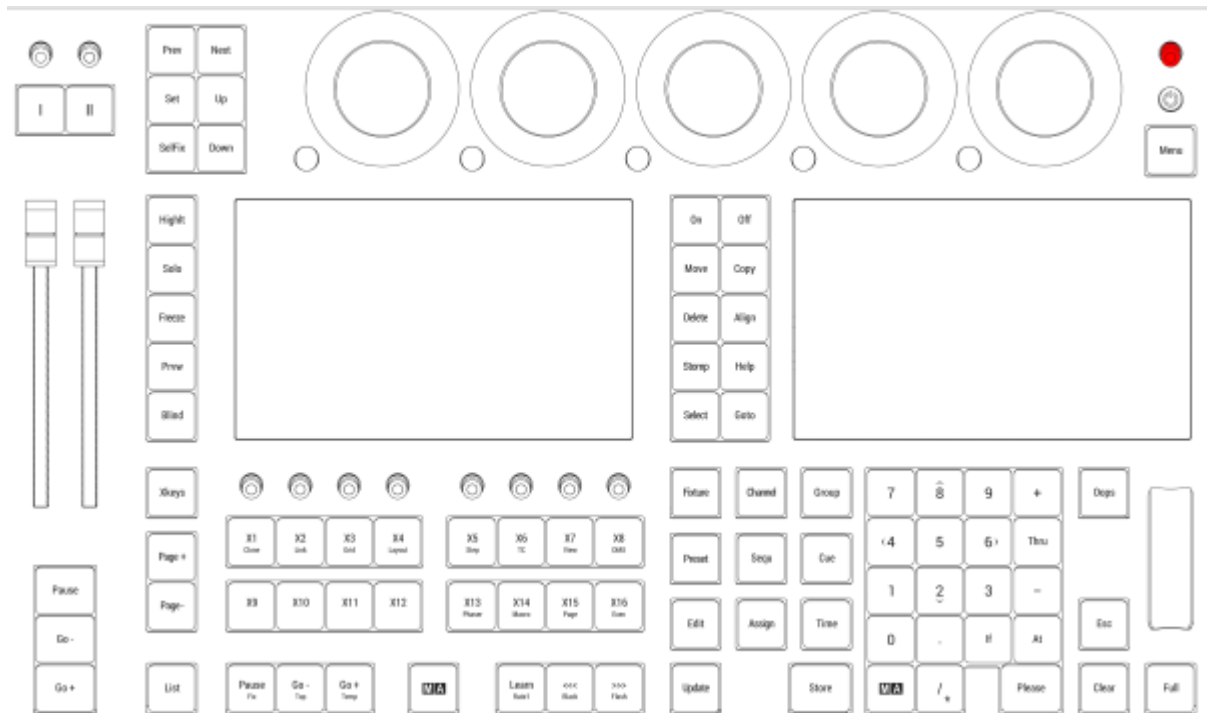
- The Grand Master level can be set by the fader.
- To enable or disable the Grand Master function, see [Master Modes](#).

The Grand Master can also be assigned to any executor. See the [Assign object to an Executor](#) topic.

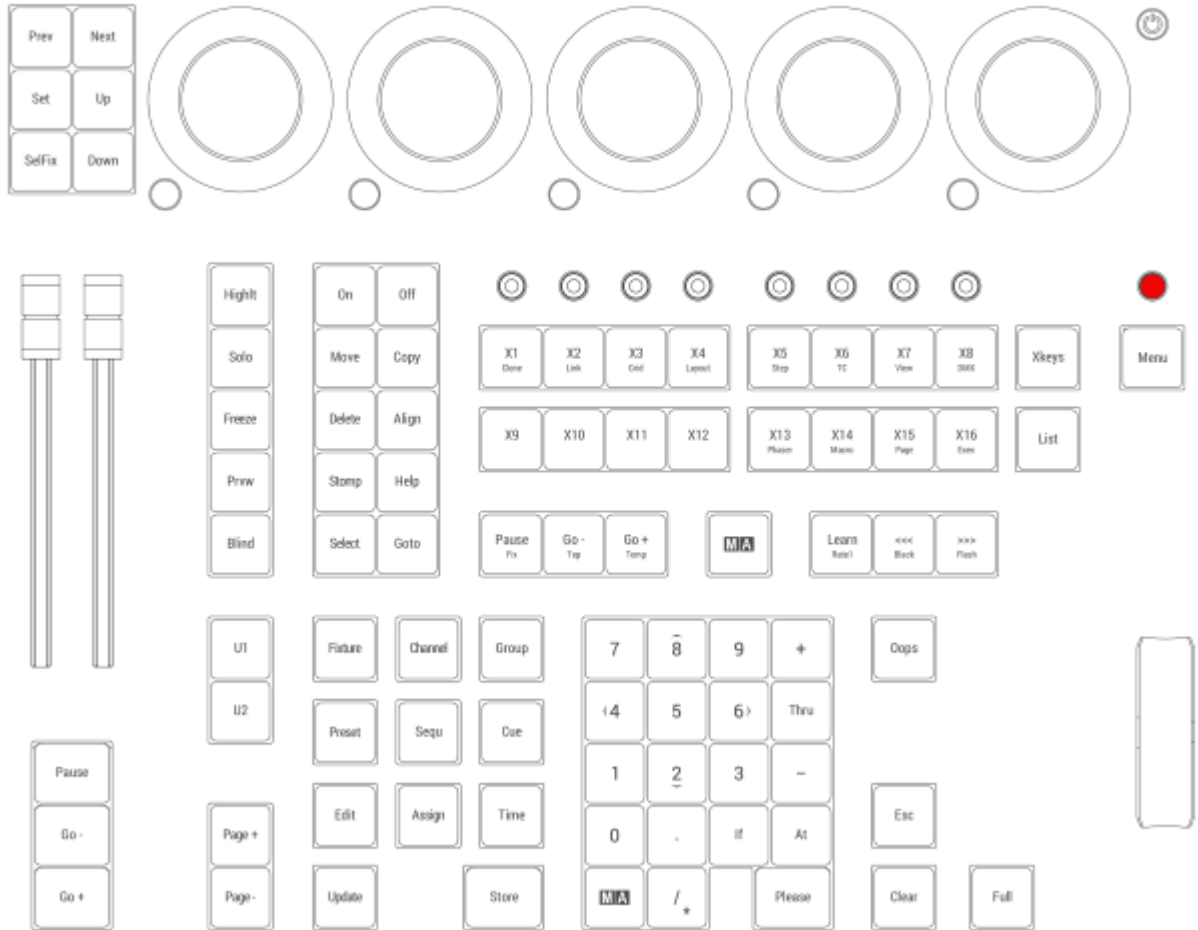


Hint:
The Output layer and the DMX layer in both the [Fixture sheet](#) and the [Layout view](#) as well as the [DMX sheet](#) and the [3D window](#) all display values as adjusted by the Grand Master.

The Grand master is located on the right side of the console.



Location on grandMA3 full-size and grandMA3 light consoles




Location on grandMA3 compact consoles and grandMA3 onPC command wing

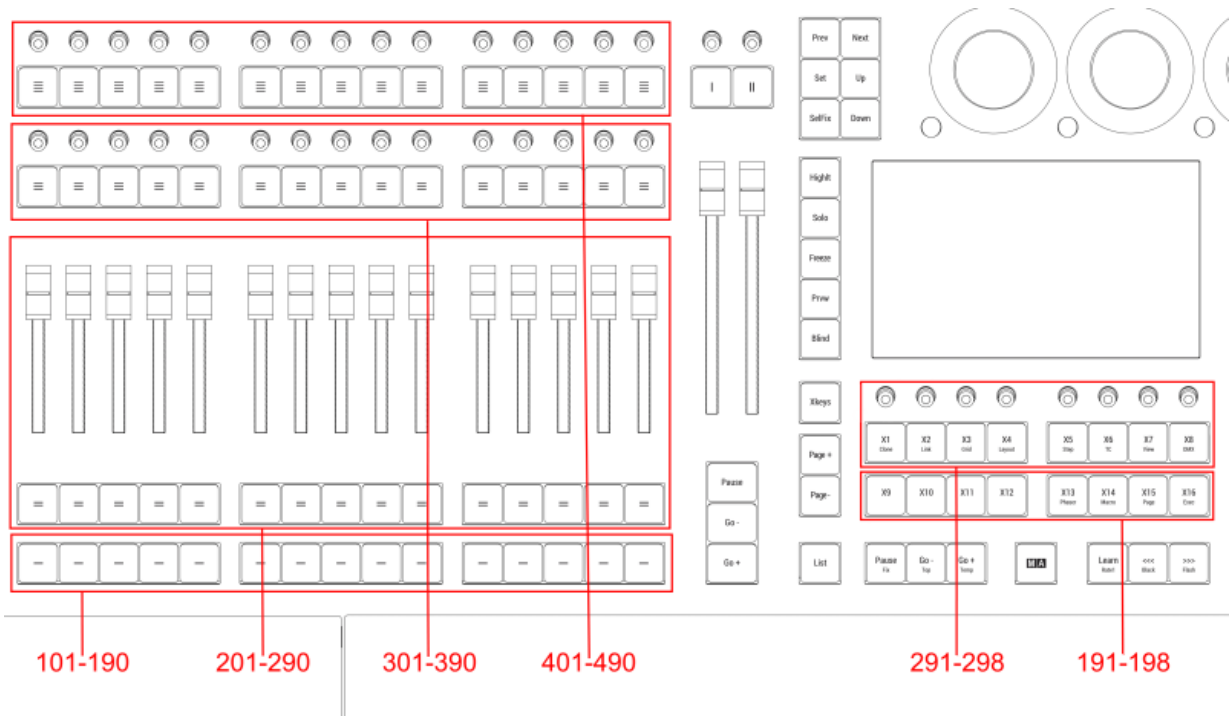


2.16.7. Executor Elements

Executors are used to playback and control elements of a show-file in an accessible way. For example they can be used to store a sequence.

For more informations about executors, see the [Executors topic](#).

- To open the Playback Control window, in the **Add Window pop-up** tap **Common** and **Playback**.
- To open the Playback Control overlay, press **F5** on the keyboard or tap  in the control bar.
- The number of physical executors depends on the console type.
- The executors are handles to control objects in the show file. The executors are located on the left part of the console.



Executor location on large consoles

Executor 101 thru 190

The executors 101 thru 190 are located as the lower buttons in the executor section. On the hardware keys, these have labeled one horizontal line on top of them. In the Playback Control overlay or the Playback window, they are named by their number.



Executor 191 thru 198

The executors 191 thru 198 are located as the lower row of Xkeys (X9 to X16). Please see the [Keys topic](#) for more information.

Executor 201 thru 290

The executors 201 thru 290 are located as the second lower buttons and the faders in the executor section. One button and the above fader is one executor. On the hardware keys, these have labeled two horizontal lines on top of them. In the Playback Control overlay or the Playback window, they are named by their number.

Executor 291 thru 298

The executors 291 thru 298 are located as the upper row of Xkeys (X1 to X8). Please see the [Keys topic](#) for more information.

Executor 301 thru 390

The executors 301 thru 390 are located as the second upper buttons and knobs in the executor section. One button and one knob is one executor. On the hardware keys, these have labeled three horizontal line on top of them. In the Playback Control overlay or the Playback window, they are named by their number.

Executor 401 thru 490

The executor 401 thru 490 are located as the upper buttons and knobs in the executor section. One button and one knob is one executor. On the hardware keys, these have labeled four horizontal line on top of them. In the Playback Control overlay or the Playback window, they are named by their number.


2.17. Connector Pin Assignment

This page gives an overview of all connectors and their respective pinouts.

- To receive more information, tap images below.


XLR Connectors

Desk Light

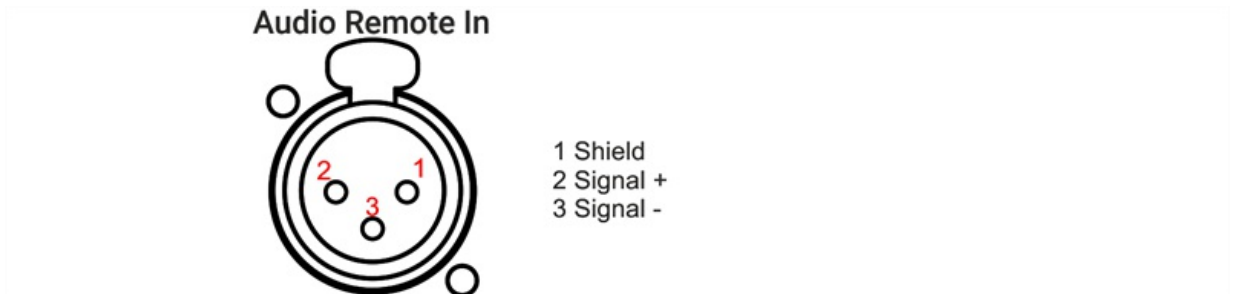
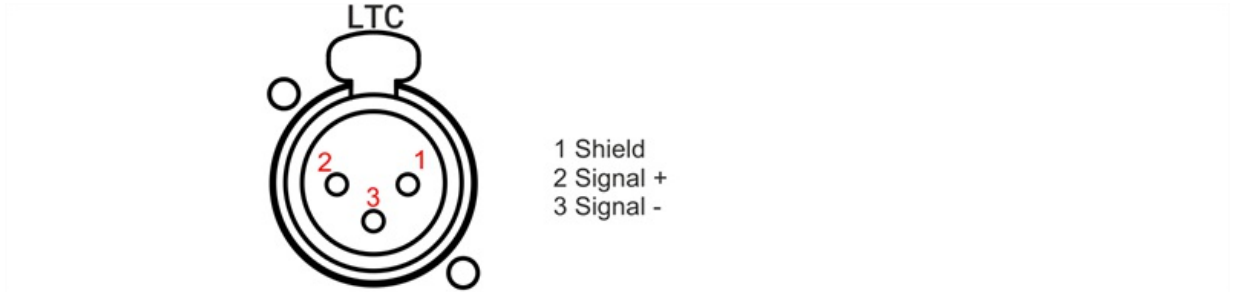


- 1 Shield
- 2 nc
- 3 12V, max. 300mA
- 4 GND (PWM)

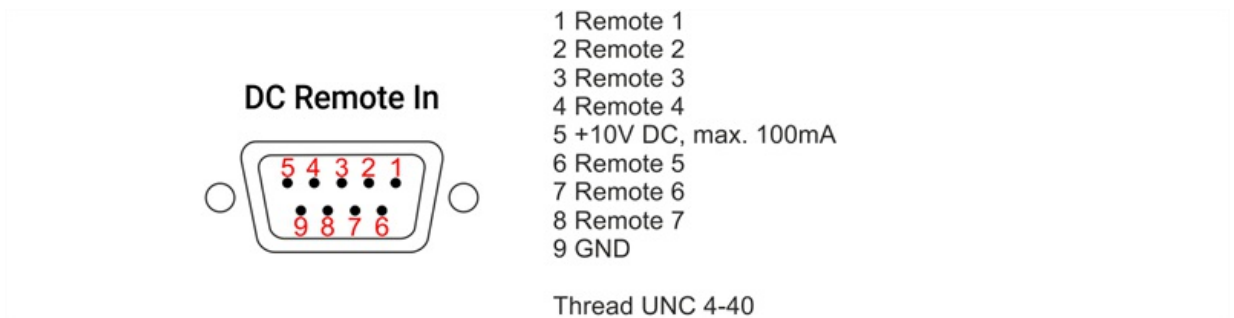
DMX



- 1 Shield
- 2 DMX -
- 3 DMX +
- 4 nc
- 5 nc



D-Sub Connectors



GPI for grandMA3 consoles, grandMA3 onPC command wing, command wing XT

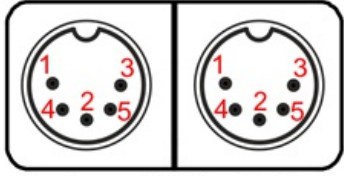
RCA Connectors



1/8" (3.5mm) Jack Sockets

<p>Line In/Out</p> 	<p>1 Shield 2 Signal R 3 Signal L</p>
---	---

DIN Connectors

	<p>1 nc 2 Shield 3 nc 4 Signal+ (Voltage Reference Line) 5 Signal- (Data Line)</p>
---	--

Phoenix Terminals

<p>Power</p> 	<p>1 PE 2 N 3 L</p>
---	---

Power connector: MC 1.5/ 3-ST1-5.08

<p>LTC</p> 	<p>1 Shield 2 Signal - 3 Signal +</p>
---	---

LTC connector: FK-MC 0.5/ 3-ST-2.5

<p>DMX</p> 	<p>1 Shield 2 Signal - 3 Signal +</p>
---	---

DMX connector: FK-MC 0.5/ 3-ST-2.5

MIDI




1 2 3

- 1 Shield
- 2 Signal- (Data Line)
- 3 Signal+ (Voltage Reference Line)

MIDI connector: FK-MC 0.5/ 3-ST-2.5

Remote



1 2 3

- 1 Remote 6
- 2 Remote 7
- 3 GND

Remote connector: FK-MC 0.5/ 3-ST-2.5

Remote



1 2 3 4 5 6 7

- 1 GND
- 2 +10V
- 3 Remote 1
- 4 Remote 2
- 5 Remote 3
- 6 Remote 4
- 7 Remote 5

Remote connector: FK-MC 0.5/ 7-ST-2.5

To show the rear panel connectors on the letterbox screens on grandMA3 light and full-size:

1. Press **Menu** and then tap **Output Configuration**.
2. To show the rear panel connector, enable **Show Connectors** button.

or use the command line:

```
MA User name[Fixture]> Menu "ConnectorView"
```

2.18. UPS Battery

The grandMA3 full-size, grandMA3 light, and their CRV versions have a rechargeable lithium-ion battery for an uninterrupted power supply (UPS).

The purpose of the battery is to enable to save the show file and shut down the console in case of a power loss.



Hint:

To receive general information on the power status and on the battery status, see **Command Line** topic.



Restriction:

Do not use the battery to bypass the console for a longer period of time! Do not move the console around while in UPS mode!

In case of a power loss:

- A Power Lost pop-up appears on every screen and command screen, depending on the console type.
- The power status icon in the command line turns red.

Power Lost Pop-up

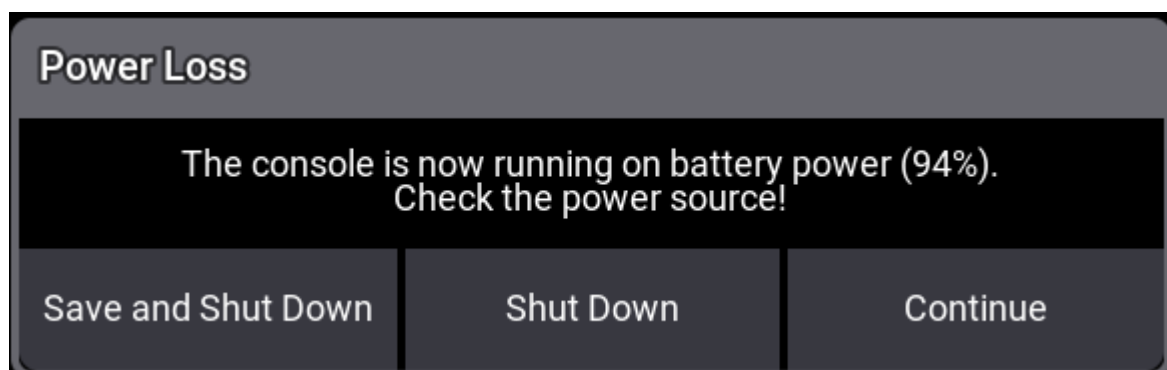
While this pop-up is open, the percentage in the parentheses (X%) continuously decreases. The percentage in the parentheses is linked to the power status of the battery.

In the pop-up window, three commands are available to tap:

Save and Shut Down: Saves the current show file and shuts down the console properly.

Shut Down: Properly shuts down the console without saving the show file.

Continue: The console continues running on battery power. Further Power Lost pop-ups appear during battery discharge process. When continued, the console will automatically shut down reaching (0%). The console reboots automatically when power supply is back.



After a power loss, the Power Lost pop-up appears on the console.



Hint:

To reboot the console after a proper shutdown, press **Power** when power supply is back.

Battery Status Information

To avoid deep discharge, recharge the battery to 80% capacity once a year.

- To check the battery status, switch on the console, boot it up, and mind the status bar of the UPS battery.

Battery Transportation

- Follow the transportation regulations for each country. If it is necessary to discharge the battery at a level of 30% for air cargo, run the console in UPS mode until the battery voltage reaches 17V. For more information about transportation, see **Transport** in the grandMA3 Quick Manual.



Software Check

The battery function of the grandMA3 console should be checked before every major show programming or at least every two months.

To check the battery:

1. Power up the grandMA3 console and wait until the battery is fully charged.
2. Load a new show file.
3. To disconnect the console from the power, flick the power switch on the back of the console to off (0). The console now runs on battery.
4. The Power Loss pop-up appears. To proceed the battery check, tap **Continue**.
5. Let the console run on battery for four minutes. After four minutes the battery status should not be lower than approximately 50%.
6. To connect the console back to power, flick the power switch to on (I). Let the battery charge to 100%.



Important:

The discharge speed depends on several conditions. For example, the type of console, show size or 3D rendering.



Important:

Battery degradation can vary depending on time and charge-discharge cycles.



3. System Overview

This section takes a look at the different possibilities with a standalone console but also the options when expanding the system.



3.1. Standalone Device

If a console is not connected to anything, then it is a **Standalone** system. It is also a standalone system using a grandMA3 onPC with a grandMA3 onPC command wing connected.

Any grandMA3 device that can create and run a **session** is called **Station**.

Stations with network disabled are in **Standalone** mode.

A station is in **IdleMaster** mode when the network is On and it is ready to be in a session with other stations, but currently, it is alone. So a console that is connected to a network, but not in a session with other stations is considered as a master that is ready to connect to other stations.

The current status can be seen in the Network menu. The title bar has an area that displays the status.

If the station is in a session and it needs to be set to Standalone mode, then turn off the network. This can be done using the **GUI** or the command line (**LeaveSession keyword**).

In the **Standalone** or **IdleMaster** mode, the station is limited to only controlling the number of **parameters** the console/onPC unlocks.

You can only use the DMX ports on the console/wings.

If DMX is output via an Ethernet connection, then the console needs to be in a session. Even though not connected to other stations, it still needs to run an active session as IdleMaster.



3.2. Locally Networked Devices

When more stations are connected with each other in a **session**, then it is a networked system. Local networked systems are sessions running in the same network. This might be a big network spanning a big area, but if it is a closed system without contact with the rest of the world (for instance through the internet), then it is considered a local networked system.

The smallest networked system is two stations running in a session together where one is the master of the session and the other is connected running as a backup. A single station that outputs DMX locally is considered a **standalone system**.

Most systems have one console with one backup console or onPC and some networked devices to output the DMX.

If the devices that translate Ethernet data to DMX are MA nodes, then the DMX output is in sync, no matter where the DMX is output. So a blackout cue on LED fixtures is in sync when some LEDs got signal from the console and some LEDs from a node somewhere in the system.

Besides the stations, there are a lot of different devices that can be added to the system. Read more about the different devices in the **Device Overview topics**.

In all networked systems, there is one station that is the session master - this is the **GlobalMaster**. This is the one that is in charge of the communication and the one that executes commands triggered by remote inputs or from a command in a sequence.

Other stations in the system are in a **Connected** status. They are connected to the master station.

Traditionally we often recommend creating local networks with good quality switches, cables, and equipment. Most systems are local systems without any connection to the outside world.

3.3. World Server

Each station can be connected to the internet and connected to a World Server.

If an internet connection is detected, the station automatically tries connecting to a world server.

The connection status of the world server can be seen in the **Command Line**. A globe icon indicates the connection status.



If the globe is green, then the station is connected to the server.

Learn more about networking in the **Networking topic**.

The address of the official world server, provided by MA Lighting is: worldserver.malighting.de

The server address can be changed in the Network Menu. Learn more about this in the **Session topic**.

The server offers two functions: Fixture type files and Crash Log upload.

Fixture Type Files

Fixture type files from the GDTF-share and grandMA3 Share are provided as a direct import to the show file.

Learn more about manually importing GDTF in the **Import GDTF topic** and patching fixtures in the **Add Fixtures to the Show topic**.

Crash Logs

A station creates a crash log if it crashes. These log files are automatically sent to the server when there is a connection. This means that the file is sent the next time the station is connected. This is also true if the station returns from a job and is connected to the world server when it returns to the workshop.

If Tech Support is needed, please make sure the station has been online and provide the Tech Support date and time the crash happened and the station's serial number (grandMA3 onPC stations also have a generated serial number). The serial number can be found using the **Version Keyword**.



Important:

If these functions are not desired, then change the server address in the **Network menu** to "0".



3.4. Parameters

Most people are used to think in DMX channels when considering how many fixtures can be controlled by a system.

MA Lighting cares more about **Parameters**.

What are Parameters

Parameters are also called **Attributes** in the software. A dimmer function is a parameter and a pan function is also a parameter.

The software is calculating the different attributes using a higher precision than what is output via DMX. So the software is calculating the parameter or attribute once and it is then scaled to the amount of DMX channels a fixture uses - typically one or two per attribute.

Why Counting in Parameters and Not DMX

It is to your advantage. In the MA world, you do not pay extra for using fixtures that are running 16 or 24 bit instead of the 8 bit used by one DMX channel.

There might be a need to add more networked nodes or devices to output all the parameters that are allowed to be controlled.

Example:

A simple moving head with a dimmer might use 5 DMX channels. The channels can be defined like this:

Definition:	DMX Channel:
Dimmer	1
Pan	2
Pan Fine	3
Tilt	4
Tilt Fine	5

The pan and tilt is one attribute each. Even though both pan and tilt each use two DMX channels it is only counted as one each in the parameter count. This means that the fixture only costs 3 parameters.

Definition:	DMX Channel:	Parameter cost:
Dimmer	1	1
Pan	2	2
Pan Fine	3	free
Tilt	4	3
Tilt Fine	5	free

This can be a big advantage when there are many fixtures with many 16-bit (or fine) channels.



What about Preprogramming and Parameters?

The show can be preprogrammed and visualized in the **3D window** without any parameter unlocking hardware. The lights are still visualized.

If third-party visualizers are used, then grandMA3 hardware is needed to give access to the parameters.



3.4.1. Calculate Parameters

Calculating the needed amount of parameters can be a big task depending on the size of the setup.

It is generally true that most modern fixtures use fewer parameters than DMX channels.

The best way to see how many parameters a show needs is to patch all the fixtures and then open the **Details** in the **System Info Window**.

Here the amount of needed (used) parameters can be seen, and the amount of currently available parameters.

Read the **Expand the amount of parameters** topic to learn how to get more.

Parameter Count

Some devices provide parameters:

- **grandMA3 full-size and grandMA3 full-size CRV:**
20 480 parameters.
- **grandMA3 light and grandMA3 light CRV:**
16 384 parameters.
- **grandMA3 extension:**
none parameters.
- **grandMA3 replay unit:**
8 192 parameters.
- **grandMA3 compact XT:**
8 192 parameters.
- **grandMA3 compact:**
8 192 parameters.
- **grandMA3 processing unit XL:**
16 384 parameters.
- **grandMA3 processing unit L:**
8 192 parameters.
- **grandMA3 processing unit M:**
4 096 parameters.
- **grandMA3 8Port Node:**
none parameters.
- **grandMA3 4Port Node:**
none parameters.
- **grandMA3 2Port Node:**
none parameters.
- **grandMA3 8Port Node DIN-Rail:**
none parameters.
- **grandMA3 4Port Node DIN-Rail:**
none parameters.



- **grandMA3 2Port Node DIN-Rail:**
none parameters.
- **grandMA3 onPC rack-unit:**
4 096 parameters.
- **grandMA3 onPC command wing XT:**
4 096 parameters.
- **grandMA3 onPC command wing:**
4 096 parameters.
- **grandMA3 onPC 8Port Node 4k:**
4 096 parameters (when connected to grandMA2 onPC this node provides only 2 048 parameters).
- **grandMA3 onPC 4Port Node 4k:**
4 096 parameters (when connected to grandMA2 onPC this node provides only 2 048 parameters).
- **grandMA3 onPC 2Port Node 2k:**
4 096 parameters (when connected to grandMA2 onPC this node provides only 1 024 parameters).
- **grandMA3 onPC 8Port Node DIN-Rail 4k:**
4 096 parameters (when connected to grandMA2 onPC this node provides only 2 048 parameters).
- **grandMA3 onPC 4Port Node DIN-Rail 4k:**
4 096 parameters (when connected to grandMA2 onPC this node provides only 2 048 parameters).
- **grandMA3 onPC 2Port Node DIN-Rail 2k:**
4 096 parameters (when connected to grandMA2 onPC this node provides only 1 024 parameters).
- **grandMA3 onPC Software:**
none parameters.



3.4.2. Expand the Amount of Parameters

The grandMA3 processing units are the **only units** that expand the parameter count when using grandMA3 consoles.

Every grandMA3 processing unit added to the network also adds a number of parameters depending on the model!

There is a **maximum limit of 250 000 parameters** in a grandMA3 session.

The grandMA3 processing units help with parameter calculations.

Examples:

1 grandMA3 full-size (20 480) + 1 grandMA3 processing unit XL (16 384) = 36 864 parameters

1 grandMA3 full-size (20 480) + 15 grandMA3 processing unit XL (16 384) = 250 000 parameters (the calculation is 266 240, but the limit is 250 000)

1 grandMA3 light (16 384) + 1 grandMA3 processing unit M (4 096) = 20 480 parameters

1 grandMA3 light (16 384) + 15 grandMA3 processing unit XL (16 384) = 250 000 parameters (the calculation is 262 144, but the limit is 250 000)

1 grandMA3 full-size (20 480) + 1 grandMA3 light (16 384) = 20 480 parameters (consoles cannot expand the parameter count, so the parameters from the console with the highest number are unlocked)

1 grandMA3 light (16 384) + 15 grandMA3 onPC 2Port Node 2k (4 096) = 16 384 parameters (nodes cannot expand the parameter count with consoles, so the parameters from the console are used)

1 grandMA3 light (16 384) + 1 onPC computer with a grandMA3 onPC command wing (4 096) = 16 384 parameters (onPC command wings cannot expand the parameter count with consoles, so the parameters from the console are used)

Using a grandMA3 onPC

When a grandMA3 onPC is used as the primary station (the system does not include any grandMA3 consoles), some grandMA3 **onPC** hardware is needed to unlock parameters.

The grandMA3 processing units also unlock parameters in an onPC system.



Important:

The maximum number of parameters allowed in a grandMA3 onPC system is 4 096.

Any amount of onPC hardware and processing units can be used with a computer or the grandMA3 onPC rack-unit. Every piece of hardware will add its parameters until the limit of 4 096 parameters is reached.



grandMA3 xPort Nodes need to be onPC versions to unlock parameters.

These are the only two rules for parameters with grandMA3 onPC.

Remember, more units can be added to get more DMX ports even after reaching the parameter limit.

Examples:

grandMA3 onPC rack-unit (4 096) + grandMA3 onPC 2Port Node 2k (4 096) = 4 096 parameters (the limit)

grandMA3 onPC rack-unit (4 096) + grandMA3 onPC 4Port Node 4k (4 096) = 4 096 parameters (the limit)

grandMA3 onPC + grandMA3 onPC command wing (4 096) = 4 096 parameters (the limit)

grandMA3 onPC rack-unit (4 096) + grandMA3 onPC command wing (4 096) = 4 096 parameters (the limit)

grandMA3 onPC command wing XT (4 096) = 4 096 parameters (the limit)

grandMA3 onPC + grandMA3 onPC command wing (4 096) + grandMA3 onPC 2Port Node 2k (4 096) = 4 096 parameters (the limit)

grandMA3 onPC + grandMA3 onPC command wing (4 096) + grandMA3 onPC fader wing (4 096) = 4 096 parameters (the limit)

grandMA3 onPC + grandMA3 onPC 2Port Node DIN-Rail 2k (4 096) = 4 096 parameters

grandMA3 onPC + grandMA3 onPC 2Port Node 2k (4 096) + grandMA3 8Port Node (none) = 4 096 parameters (the 8Port Node is not an onPC node)

grandMA3 onPC + grandMA3 onPC command wing (4 096) + grandMA3 onPC 8Port node 4k (4 096) = 4 096 parameters (the limit is reached and cannot be exceeded)

grandMA3 onPC + 1 grandMA3 processing unit M (4 096) = 4 096 parameters (the limit)

grandMA3 onPC + 1 grandMA3 processing unit XL (16 384) = 4 096 parameters (the limit is reached and cannot be exceeded)



4. First Steps

This chapter describes how to prepare the grandMA3 consoles and the grandMA3 onPC for operation, from unpacking the device to turning it on for the first time.



4.1. Unpack the Device

- Unpack the device. Remove all packing material, strips and protection films.
- Keep the packing material for transport.



Information:

The device was tested for proper function at the factory.



4.2. Check Scope of Delivery

The list below shows the scope of delivery. If anything is missing, contact your local distributor.

grandMA3 full-size

- 2 x LED desk light grandMA3
- 1 x Dust cover grandMA3 full-size
- 2 x Magnetic plate for easy labeling of 15 faders
- 1 x grandMA3 Quick Manual consoles

grandMA3 full-size CRV

- 2 x LED desk light grandMA3
- 1 x Dust cover grandMA3 full-size CRV
- 2 x Magnetic plate for easy labeling of 15 faders
- 1 x grandMA3 Quick Manual consoles

grandMA3 light

- 2 x LED desk light grandMA3
- 1 x Dust cover grandMA3 light
- 1 x Magnetic plate for easy labeling of 15 faders
- 1 x grandMA3 Quick Manual consoles

grandMA3 light CRV

- 2 x LED desk light grandMA3
- 1 x Dust cover grandMA3 light CRV
- 1 x Magnetic plate for easy labeling of 15 faders
- 1 x grandMA3 Quick Manual consoles

grandMA3 compact XT

- 1 x LED desk light grandMA3
- 1 x Dust cover grandMA3 compact XT
- 1 x Magnetic plate for easy labeling of 15 faders
- 1 x grandMA3 Quick Manual consoles

grandMA3 compact

- 1 x LED desk light grandMA3
- 1 x Dust cover grandMA3 compact
- 1 x Magnetic plate for easy labeling of 10 faders
- 1 x grandMA3 Quick Manual consoles

grandMA3 extension



- 1 x LED desk light grandMA3
- 1 x Dust cover
- 1x Magnetic plate for easy labeling of 15 faders
- 1 x grandMA3 Quick Manual consoles

grandMA3 replay unit

- 1x grandMA3 Quick Manual consoles

grandMA3 processing units

- 1x grandMA3 Quick Manual processing unit

grandMA3 xPort Nodes

- 1x grandMA3 Quick Manual Nodes

grandMA3 Nodes DIN-Rail

- 1x grandMA3 Quick Manual Nodes DIN-Rail

grandMA3 I/O Nodes

- 1x grandMA3 Quick Manual I/O Nodes

grandMA3 I/O Nodes DIN-Rail

- 1x grandMA3 Quick Manual I/O Nodes

grandMA3 onPC command wing

- 1 x Dust cover
- 1 x Magnetic plate for easy labeling of 10 faders
- 1 x USB cable
- 1 x grandMA3 Quick Manual onPC command wing

grandMA3 onPC command wing XT

- 1 x Dust cover
- 1 x Magnetic plate for easy labeling of 10 faders
- 1 x grandMA3 Quick Manual onPC command wing XT

grandMA3 onPC fader wing

- 1x Dust cover
- 1x USB cable
- 1x Magnetic plate for easy labeling of 10 faders
- 1x grandMA3 Quick Manual onPC fader wing

grandMA3 rack-unit

- 1x grandMA3 Quick Manual onPC rack-unit



grandMA3 viz-key

- 1x USB cable (C/C)
- 1x USB cable (C/A)
- 1x grandMA3 Quick Manual viz-key



4.3. Position the Device

Follow the instructions below:

- Do not place the device in a humid area.
- Place the device on a stable, flat, dry surface.
- Do not cover the ventilation holes.
- Do not expose the device to direct sunlight.
- Maintain a minimum distance of 15 cm (5.91 inches) between the multi-touch screen surface and radio intercom systems. Going below the minimum distance may cause unexpected behavior, such as unwanted multi-touch actions or mouse movements.

4.4. Connect Power

Devices with powerCON Connector or powerCON TRUE1 TOP Connector



Warning:

If only the powerCON TRUE1 connector or powerCON TRUE1 TOP connector is provided, attach a suitable cable to the connector.

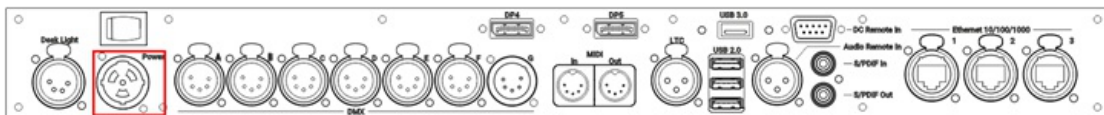
To attach a suitable cable to the powerCON TRUE1 connector or powerCON TRUE1 TOP connector, contact the local dealer.

1. Insert the powerCON TRUE1 connector or powerCON TRUE1 TOP connector in the powerCON TRUE1 plug or powerCON TRUE1 TOP plug and twist it to lock clockwise.
An audible click is heard.
2. Connect the other end of the cable to a suitable power source.

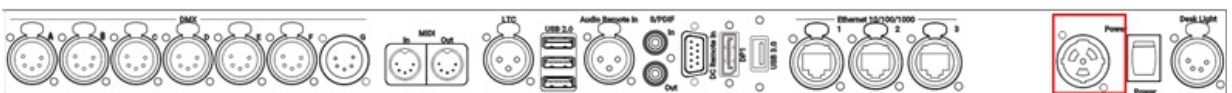
Devices with IEC Connector

1. Insert the IEC connector into the corresponding jack.
2. Connect the power plug.

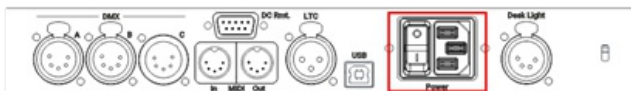
The device is connected to power.



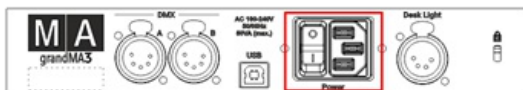
grandMA3 full-size (CRV) and light (CRV) rear panel – power



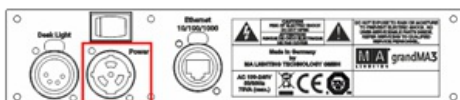
grandMA3 compact (XT) rear panel – power



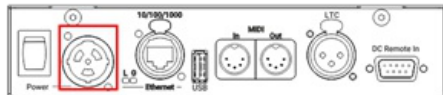
grandMA3 onPC command wing (XT) rear panel – power



grandMA3 fader wing rear panel – power



grandMA3 extension rear panel – power



grandMA3 I/O Node rear panel – power

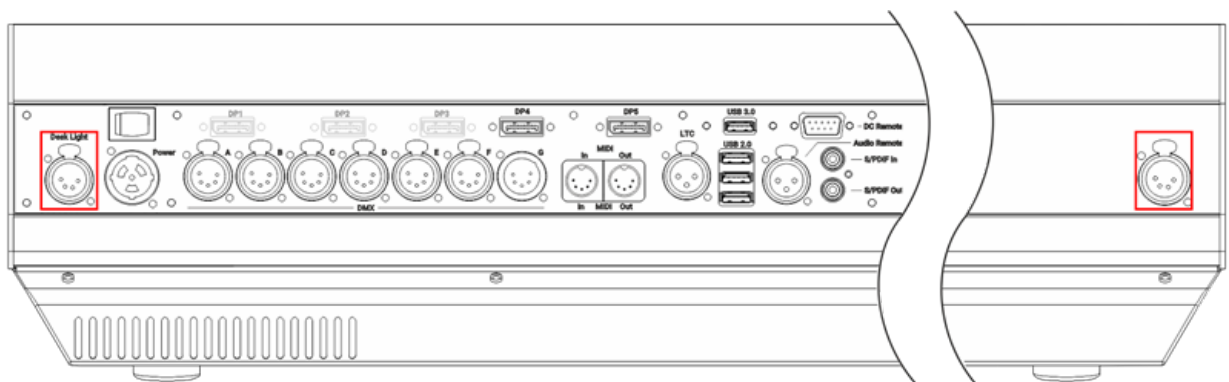


4.5. Connect Desk Light

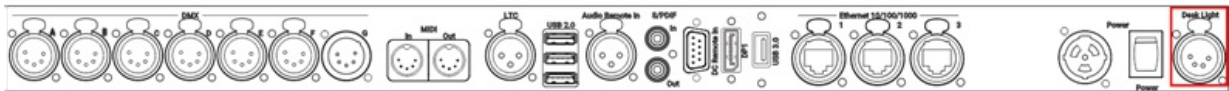
The grandMA3 comes with one or two desk lights, depending on the model. For more information see, [delivery contents](#).

- Connect the 4 pin XLR connector to the desk light connectors on the rear panel.

The desk lights are connected.



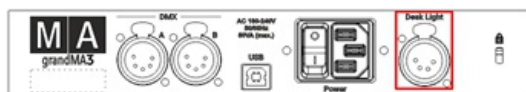
grandMA3 full-size (CRV) and light (CRV) rear panel – desk light



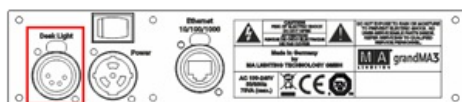
grandMA3 compact (XT) rear panel – desk light



grandMA3 onPC command wing (XT) rear panel – desk light



grandMA3 fader wing rear panel – desk light



grandMA3 extension rear panel – desk light

To view the pinout of the XLR connector, refer to the topic [Connector Pin Assignment](#).

4.6. Connect External Screens

You can connect up to five external touch screens with a grandMA3 full-size CRV (= Control Room Version). For more information about the amount of DisplayPort connectors of each model, see [Quick Manual Consoles - Technical Data](#).



Important:

The grandMA3 series supports only native DisplayPort connectors on external screens.



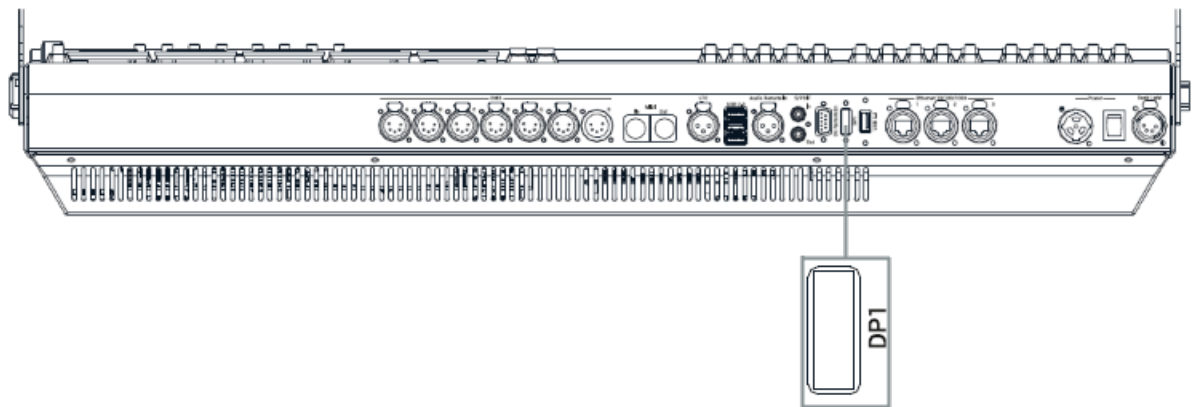
Important:

In grandMA3 CRV versions, use at least one Full HD display and a touch display or a mouse.



Hint:

The grandMA3 compact and compact XT models only have one DisplayPort called DP1.



Example of a DisplayPort connector on a grandMA3 compact XT

Screen Arrangement

Place the screens in the following order:



grandMA3 full-size screen order

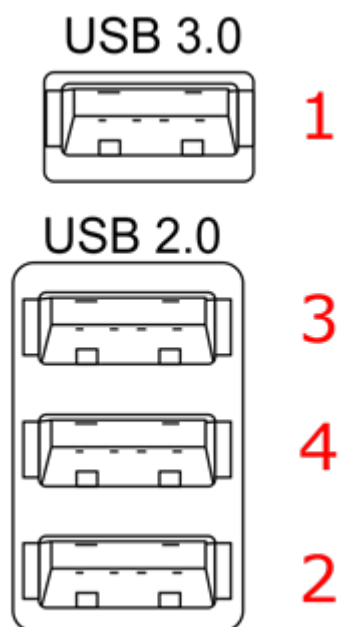
- Screens 5 and 4 are always external screens.
- Screens 3, 2, and 1 can be internal or external screens, depending on the product.
- The screen order is important for the mouse behavior from screen to screen.

Requirements for external monitors:

- Native DisplayPort connection
- Resolution of 1920 x 1080 (full HD)
- Microsoft Windows® 10 multi-touch compliance
- Separated USB connection for touch functionality

Connect external touch screens:


1. Make sure the console is turned off.
2. Place the first external screen right beside or above screen 1 and the second external screen left beside or above screen 3.
The screen number of the external screens will be numbered consecutively.
3. Connect the DisplayPort cable of screen 4 with the DisplayPort connector DP4.
4. Connect the USB cable with the corresponding USB port.
5. Connect the DisplayPort cable of screen 5 with the DisplayPort connector DP5.
6. Connect the touch screen USB cable to the appropriate USB port in the correct order as shown below:

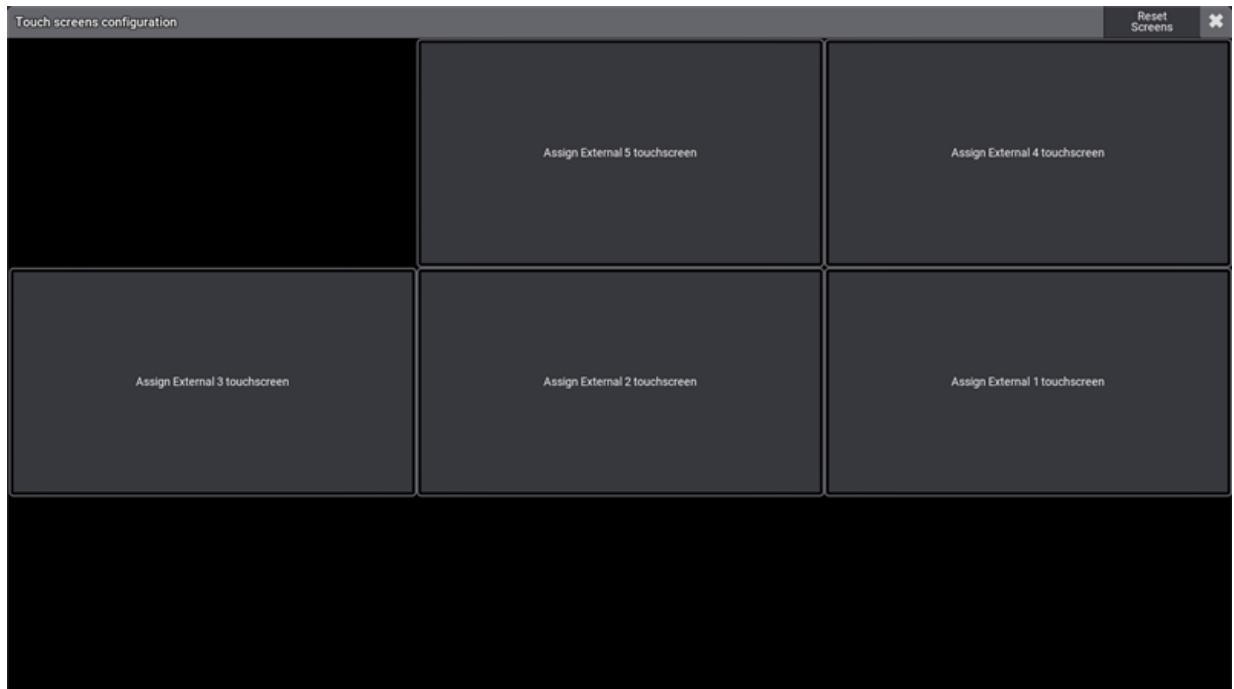


Screen allocation for a grandMA3 light CRV.

7. Press **Power** to boot up the console.

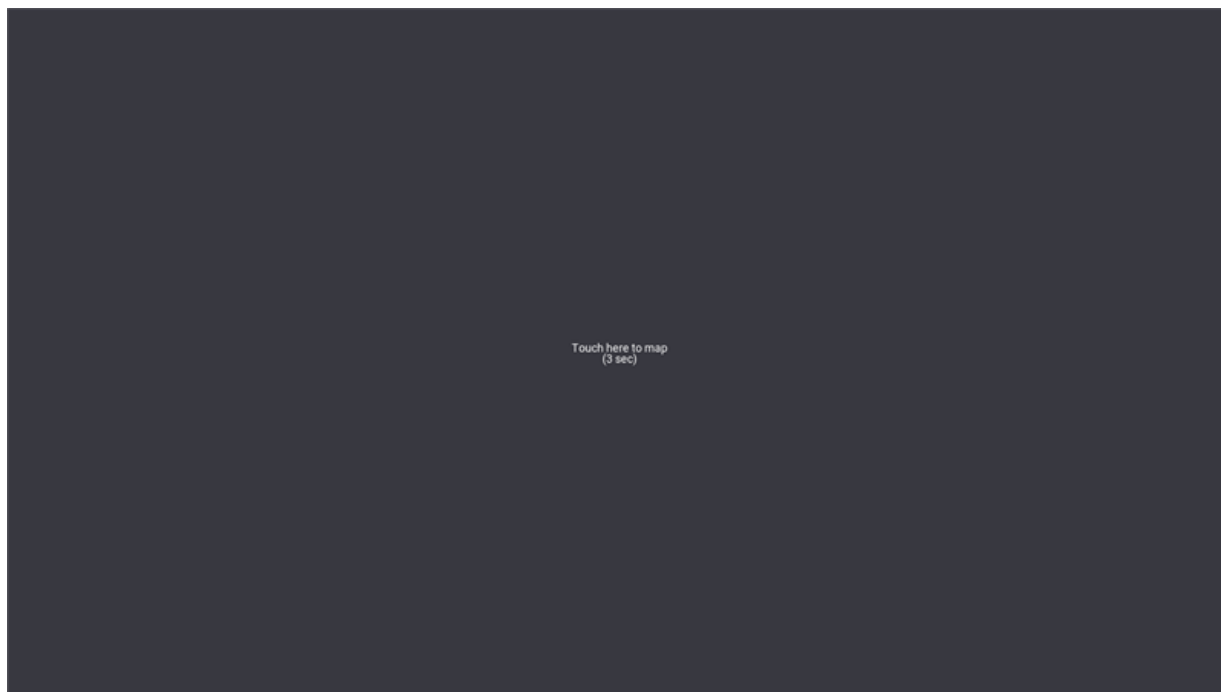
Touch Configuration

1. To adjust the touch configuration, tap , **Settings** and then tap **Touch Configuration**. The Touch Configuration overlay opens:



grandMA3 full-size CRV touch configuration


2. To assign a touch input to a monitor, tap a monitor button in the touch configuration overlay. For example, **Assign External 1 touchscreen**.
3. Touch the appropriate monitor screen to assign the touch input to that device. An overlay appears on the currently selected screen:

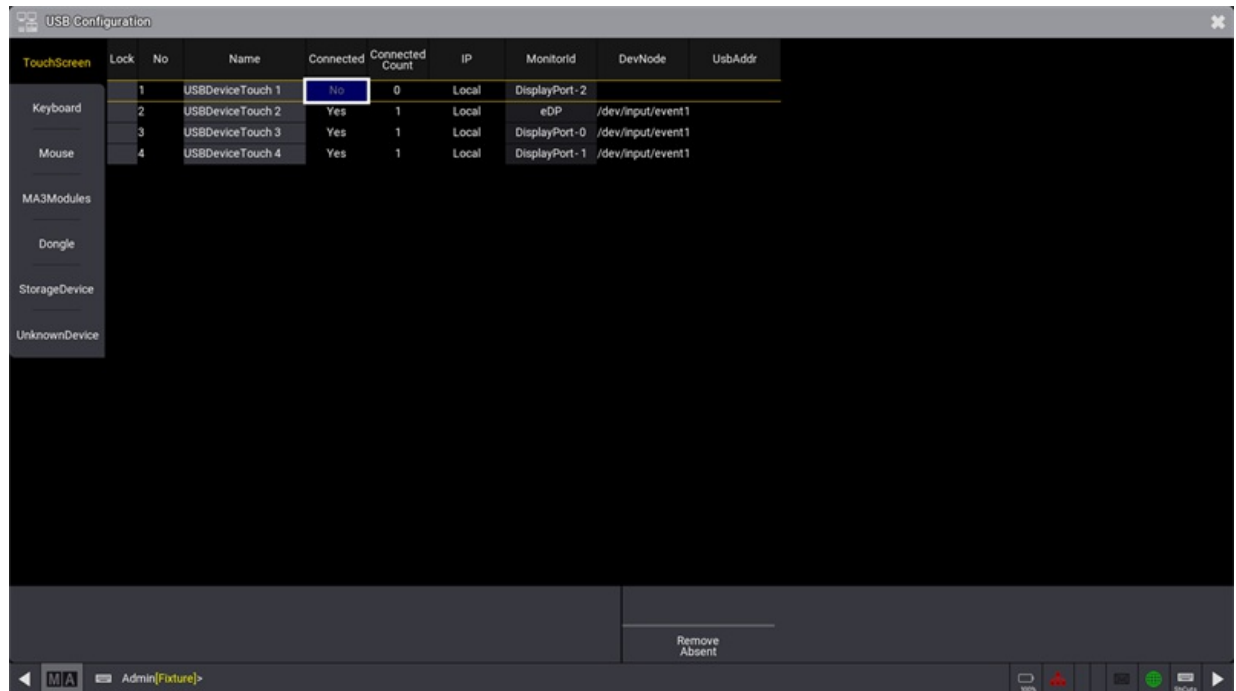


Touch mapping overlay

4. The touch function is assigned to the monitor.
- **Reset Screens** : Resets the screen allocation.

USB Configuration

- To check the USB settings, tap , **Settings** and then tap **USB configuration**. The USB Configuration overlay opens:



USB configuration window with display information

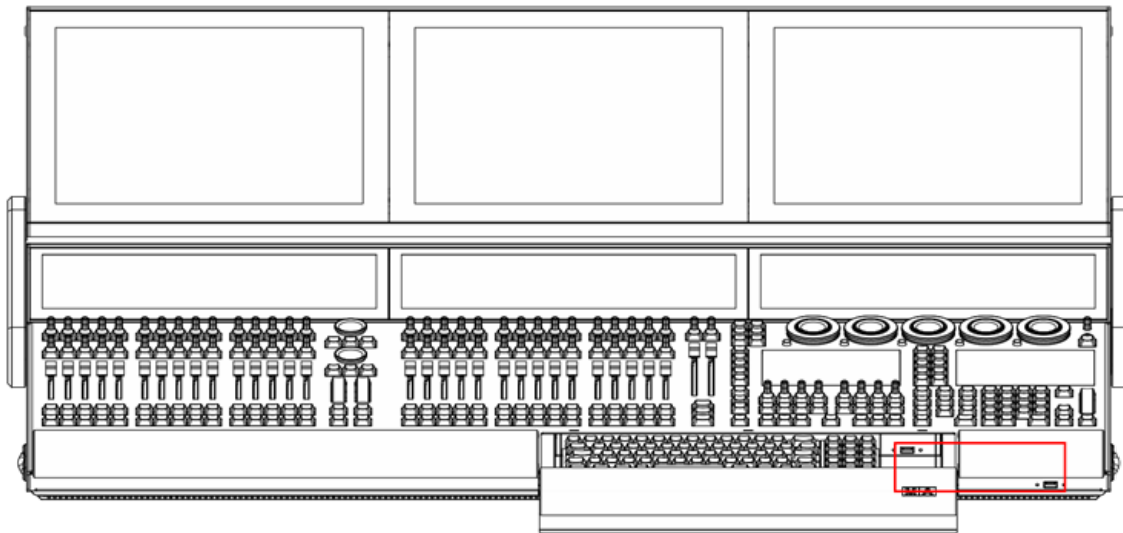
- **Remove Absent**: Removes disconnected monitors from the list above.

4.7. Connect USB Devices

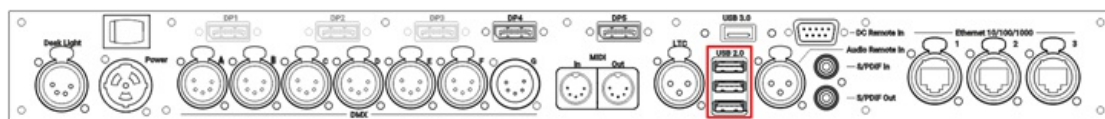
You can connect an external USB mouse, an external USB keyboard, or an external touch screen to the grandMA3 device using the USB ports.

Depending on the type of the grandMA3 device there are up to 6 USB connectors:

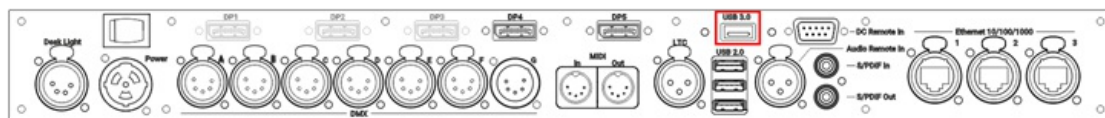
- 1x USB 3.0 at the front on the right of the console
- 1x USB 3.0 inside the keyboard drawer
- 3x USB 2.0 and 1x USB 3.0 on the rear panel



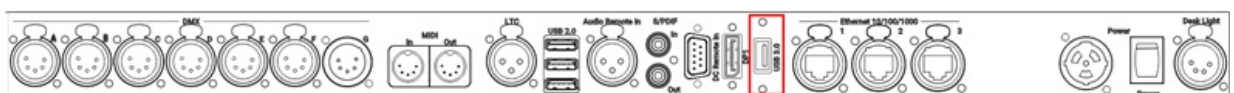
grandMA3 full-size USB 3.0 (type A) inside keyboard drawer and on the right



grandMA3 full-size (CRV) rear panel USB 2.0 (type A)

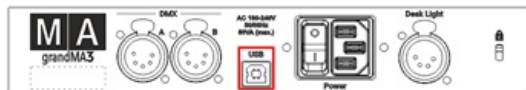


grandMA3 full-size rear panel USB 3.0 (type A)



grandMA3 compact (XT) rear panel USB 3.0 (type A)

To connect a grandMA3 onPC device with the computer or laptop, use the USB connector (type B):



grandMA3 onPC fader wing rear panel USB 2.0 (type B)



grandMA3 onPC command wing rear panel USB 2.0 (type B)

4.8. Connect DMX

You can connect DMX devices to the grandMA3 devices.

Depending on the type of the grandMA3 device there are up to six DMX Out ports (e.g. A-F) and one DMX In port (e.g. G).

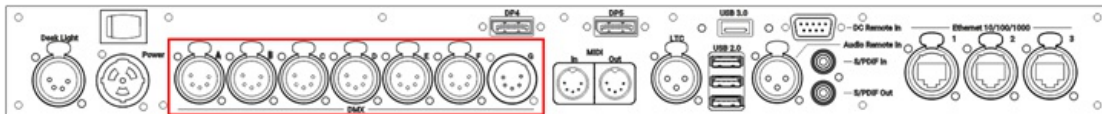
No matter which physical direction the connector has, it can be configured in the software and is therefore capable for both directions: in and out.

To adjust the DMX Remote settings, read the [In and Out topic](#).

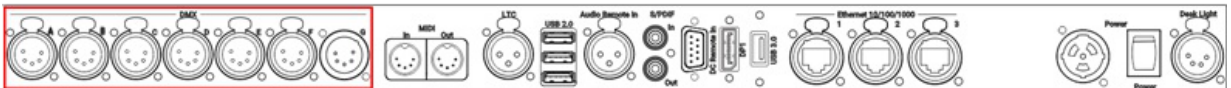
To configure the DMX ports (Off, Out, RDM, In), read the [DMX Port Configuration topic](#).

- Connect the 5pin XLR DMX cable to a DMX connector on the console and a DMX device.

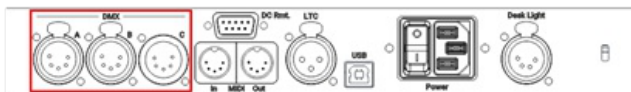
The DMX device is connected to the [XLR connector](#).



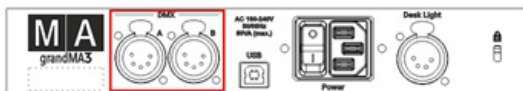
grandMA3 full-size (CRV) and light (CRV) rear panel – DMX



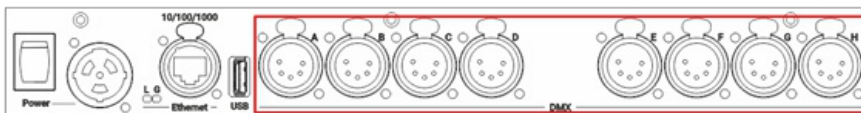
grandMA3 compact (XT) rear panel – DMX



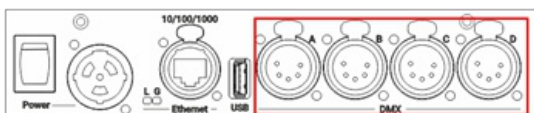
grandMA3 onPC command wing (XT) rear panel – DMX



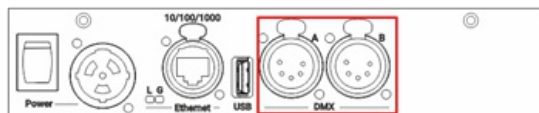
grandMA3 onPC fader wing rear panel – DMX



grandMA3 8Port Node rear panel – DMX



grandMA3 4Port Node rear panel – DMX



grandMA3 2Port Node rear panel – DMX

To view the pinout of the XLR connector, refer to the topic [Connector Pin Assignment](#).

4.9. Connect Audio In

To use a sound trigger or BPM, connect a sound source to the console.

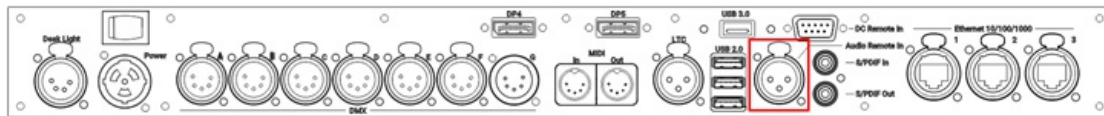


Information:

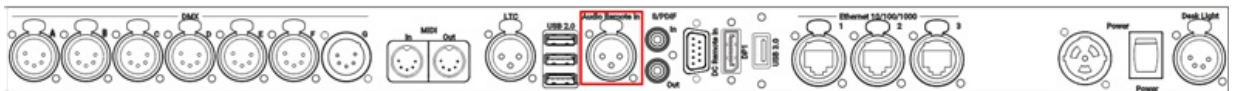
The signal strength should be a minimum of 50 mV.

- Connect the 3pin XLR sound source cable to the **Audio Remote In connector** on the rear panel.

The sound source is connected to the **Audio Remote In connector**.



grandMA3 full-size rear panel – Audio Remote In



grandMA3 compact (XT) rear panel – Audio Remote In

To view the pinout of the DIN connector, refer to the topic **Connector Pin Assignment**.



4.10. Connect MIDI

MIDI connectors can be used for MIDI input or output. For example to send the MIDI timecode signal to the sound engineer.

- To adjust the MIDI Remotes settings, read the [Remote In and Out topic](#).
- To configure the MIDI Port settings per grandMA3 device, read [Output Configuration menu](#).

1. To access the Output configuration window, press **Menu** and tap **Connector Configuration**.

-OR-

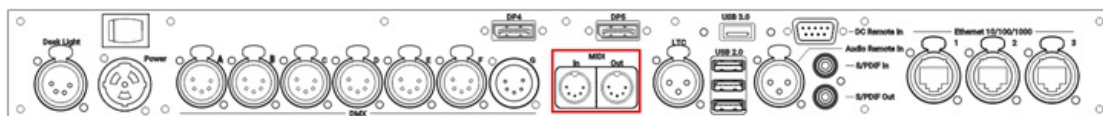
1. Use the command line to open the menu:



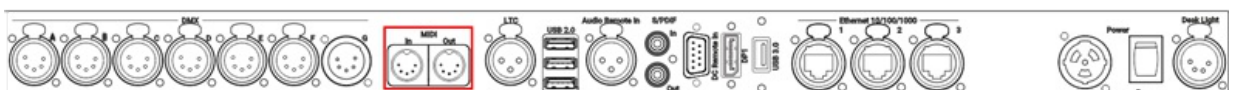
2. To select the MIDI Mode, tap and hold or right-click MIDI Mode. A pop-up window opens.
3. To set the MIDI direction, select **In**, **Out**, or **Through**.
4. Connect the MIDI source to the **DIN connector** on the rear panel of the console.

Hint:
 MIDI Mode 'Through' receives and forwards MIDI Notes, MIDI Control Changes, and MIDI Program Changes.

Location of MIDI Connectors on Different Products



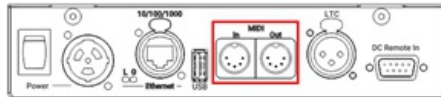
grandMA3 full-size (CRV) and light (CRV) rear panel – MIDI



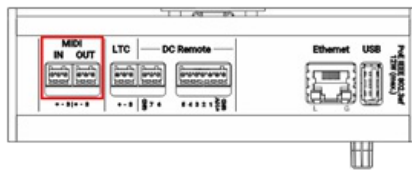
grandMA3 compact (XT) rear panel – MIDI




grandMA3 onPC command wing (XT) rear panel – MIDI



grandMA3 I/O Node rear panel – MIDI



grandMA3 I/O Node DIN-Rail top panel – MIDI

 **Hint:**
For more information on DIN connector pinout, see [Connector Pin Assignment](#).

4.11. Connect LTC

The LTC connector can be used for timecode input or for timecode output, e.g. to send the timecode signal to the sound engineer.

You can configure the direction of the LTC port in the **Output Configuration**.

To synchronize the console with an SMPTE timecode source, connect an SMPTE source to the LTC port.



Information:

The supported time formats are:

- 24 fps (frames per second)
- 25 fps
- 30 fps

If you send 29.97 or 30 drop frame, it will be interpreted as 30 fps.

Sound and timecode signal levels

Min. level	Max. level	Recom. level
-11 dBu	+15 dBu	0 dBu
0.2 Veff	4.4 Veff	0.8 Veff



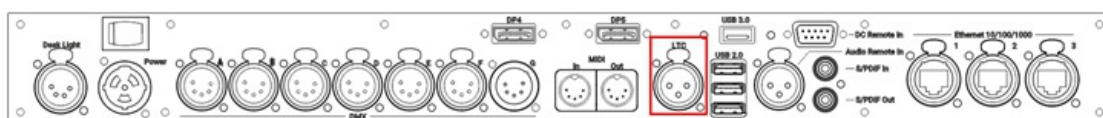
Information:

The signal strength should be a minimum of 200 millivolts.

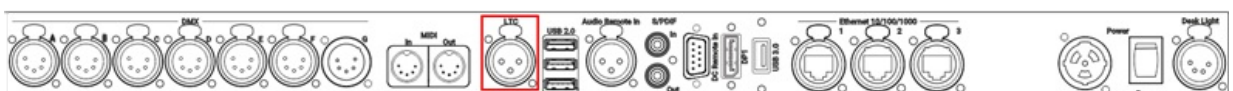
- Pin 1: Ground
- Pin 2: - (minus)
- Pin 3: + (plus)

- Connect the SMPTE source to the LTC connector on the rear panel of the console.

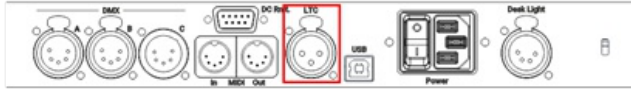
The SMPTE source is connected to the LTC connector.



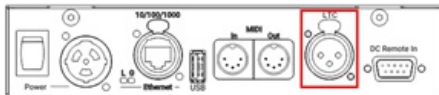
grandMA3 full-size (CRV) and light (CRV) rear panel – LTC



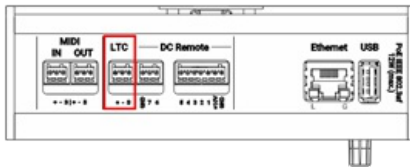
grandMA3 compact (XT) rear panel – LTC



grandMA3 onPC command wing (XT) rear panel – LTC



grandMA3 I/O Node rear panel – LTC

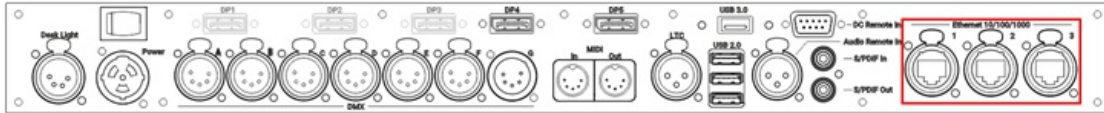


grandMA3 I/O Node DIN-Rail rear panel – LTC

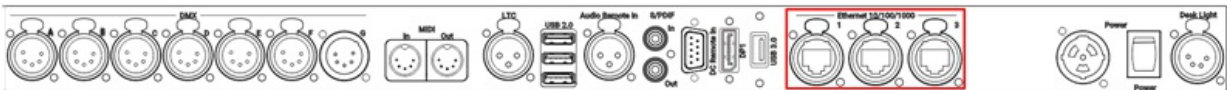
To view the pinout of the DIN connector, refer to the topic [Connector Pin Assignment](#).

4.12. Connect Ethernet

Depending on the type of grandMA3 device, there are up to three Ethernet ports on the rear panel.



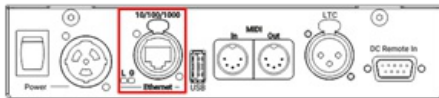
grandMA3 full-size (CRV) and light (CRV) rear panel – Ethernet



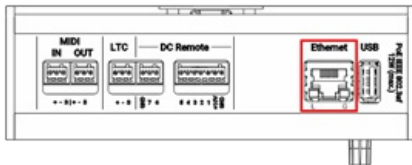
grandMA3 compact (XT) rear panel – Ethernet




grandMA3 extension rear panel – Ethernet




grandMA3 I/O Node rear panel – Ethernet



grandMA3 I/O Node DIN-Rail rear panel – Ethernet

 **Hint:**
 The naming of the Ethernet ports is individual, for example CON1, CON2, CON3 in the console.
 In the onPC software, the Ethernet ports are named by the operating system, for example Ethernet, Ethernet2 etc.

 **Information:**
 All Ethernet ports can use these DMX protocols:
 - Art-Net
 - sACN



Important:

To avoid damage, the Ethernet ports must not be connected with Power over Ethernet (PoE) except for the grandMA3 devices qualified for PoE (e.g. xPort Nodes PoE, onPC xPort Nodes PoE, or I/O Node PoE).

Ethernet Ports 1-3

Requirement: Use STP (shielded twisted pair) cable with an RJ45 connector, at least CAT-5e.

To connect one of the three Ethernet ports:

1. Connect one end of an Ethernet cable to the **Ethernet 1, 2, or 3**.
2. Connect the other end to a suitable switch.
3. Connect other grandMA3 equipment to the switch and the switch to the power supply.

Ethernet is connected to a port.

Ethernet LEDs

The L (link) LED flashes when the Ethernet connection is active.

The G (gigabit) LED flashes when a gigabit connection is available.

4.13. Connect DC Remote In

To use the DC Remote In with grandMA3 consoles, onPC command wings, and I/O Nodes connect a contact closure switch, for example, a light barrier or a push button.

For further information see the topics [Remote keyword](#), [Remote In and Out](#), and [Output configuration](#).



Hint:

You can use up to 64 input channels within a session.



Hint:

It is possible to analogously move the Master Fader using a grandma3 console or I/O Node connected with a potentiometer (0 to +10V DC).

The onPC products can switch on and off, but do not fade.

- Generate a switch or connect an external source that sends up to +10V DC to pin 1 for the console to react to analog input number 1.
- The recommended resistance is 5 kohms to 10 kohms.
- To use the DC Remote, feed a voltage signal (max. +10V DC into the corresponding input pin. For more information see the pinout image below.

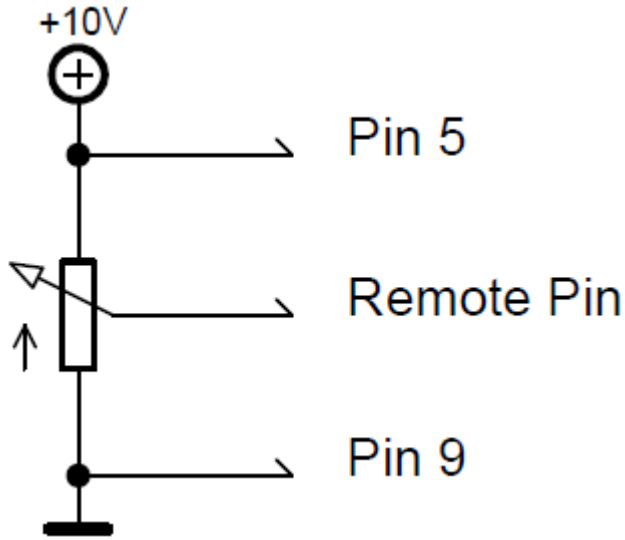
To connect a switch:

1. Take +10V DC voltage in pin 5 for the grandMA3 console or I/O Node
-or-
+5V DC voltage for grandMA3 onPC command wing and command wing XT
2. Take an external voltage source (+10V DC in grandMA3 consoles or I/O Nodes and +5V DC in grandMA3 onPC command wing and command wing XT), connect its ground to the common ground pin of the device.

Connect the +10V DC voltage source to one input pin 1-4 or 6-8 with a potential-free contact (switch, buzzer, motion detector, or any other switching device) in between.

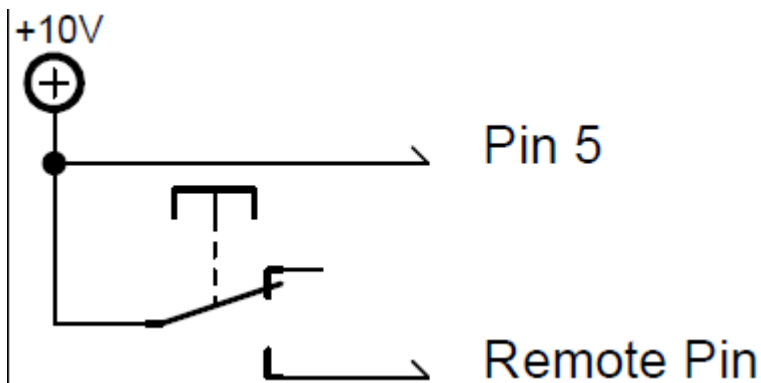
Circuit examples:

- Potentiometer +10V DC in grandMA3 consoles or I/O Nodes



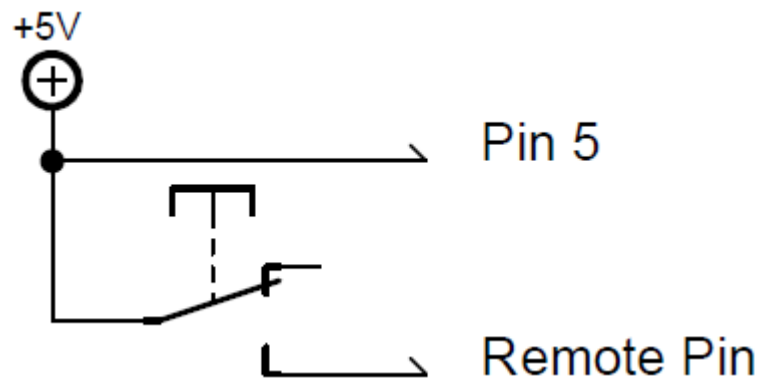
Potentiometer +10V DC

- Switch +10 DC in grandMA3 consoles or I/O Nodes



Switch +10V DC

- Switch +5V DC in grandMA3 onPC command wing and command wing XT



Switch +5V DC

Important:

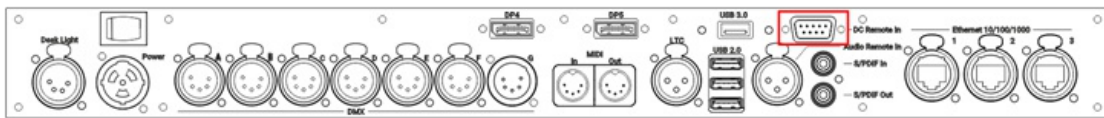
Pin layout in grandMA3 consoles or I/O Nodes:

The grandMA3 consoles or I/O Nodes have a 9-pin D-sub, enabling 7 remote inputs:

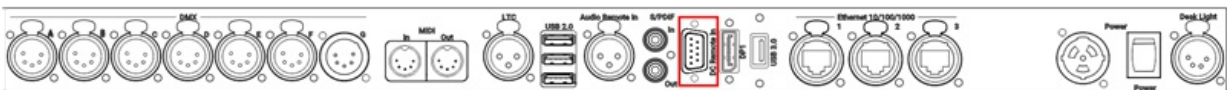
- Pin 1-4 = input channels 1, 2, 3, 4
- Pin 5 = +10V DC grandMA3 consoles or I/O Nodes/+5V DC grandMA3 onPC command wing and command wing XT
- Pin 6-8 = input channels 5, 6, 7
- Pin 9 = common ground

- Connect a D-sub plug to the DC Remote In connector on the rear panel.

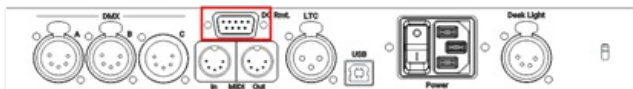
DC Remote In control is connected.



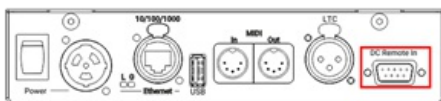
grandMA3 full-size (CRV) and light (CRV) rear panel – DC Remote In



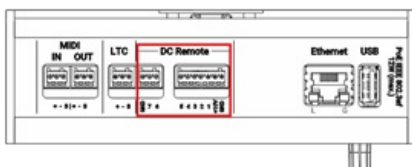
grandMA3 compact (XT) rear panel – DC Remote In



grandMA3 onPC command wing rear panel – DC Remote In



grandMA3 I/O Node rear panel – DC Remote In



grandMA3 I/O Node DIN-Rail rear panel – DC Remote In

To view the Pinout of the Sub-D connector, refer to the topic [Connector Pin Assignment](#).

4.14. Connect grandMA3 extension

The grandMA3 extension allows to extend the amount of physical executor handles for a grandMA3 full-size, grandMA3 light or grandMA3 replay unit.

For more information about this device, please refer to [grandMA3 extension](#) in the Device Overview section.

The number of grandMA3 extensions connected to a grandMA3 console depends on the console type:

Device	Number of grandMA3 extensions
grandMA3 full-size / full-size CRV	1
grandMA3 light / light CRV	2
grandMA3 replay unit	3


Requirements:

- The grandMA3 extension is similar to network / network interface as MA-Net communication. For more information about the network, see [Protocol Details](#).
- The grandMA3 extension software version must match the console software version.

To display the software version on an extension:

- Boot the extension. The software version is displayed in the lower-left corner.

To display the extension software version on a console:

1. Tap **Menu** and **Network**. The Network window opens.
2. Tap on the arrow  next to Extension. The flip-menu opens.
3. The software version is shown under Version Big in the grandMA3 extension column.



Hint:

An extension only connects to a specific console. It is not a standalone device.




Important:

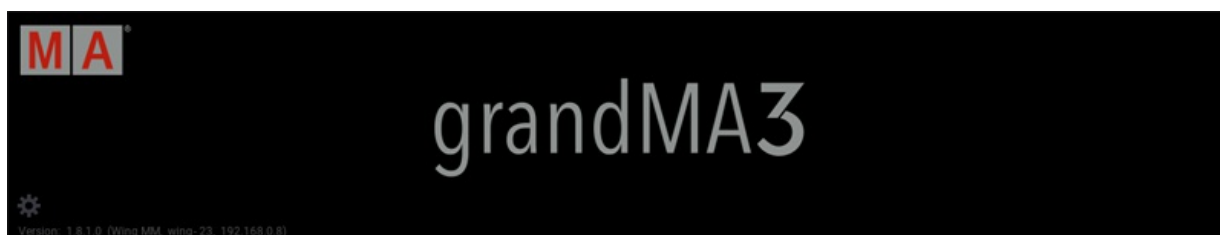
To establish a connection with an extension, the IP address of the extension has to be in the same IP range as the console.

To change the extensions IP address remotely from a console or onPC station:

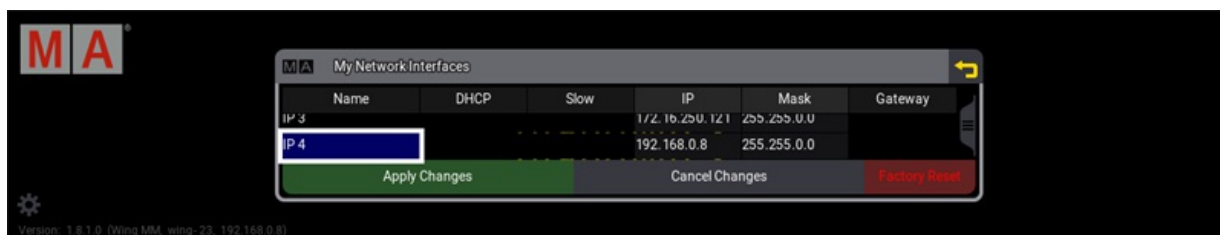
1. Tap **Menu** and **Network**. The network window opens.
2. Select the grandMA3 extension entry and click IP address / tap and hold IP address.
3. Use the pop-up window to change the IP settings.

To change the IP address on the extension:

1. Turn on the extension.
2. Tap the **gear icon**  to open the Network Interface. A network interface pop-up opens.
3. Tap on the IP address. The IP editor opens.



Gear icon on the grandMA3 extension



Network Interfaces on the grandMA3 extension

To establish a connection between a console and an extension:

1. Select the extension in the network menu on the console,
2. Tap **Invite Station**. The connection will be initiated.



Important:

It is not possible to invite the extension to a second console, when it is already connected to a different console.

To disconnect the extension from a console, cancel the connection on the console the extension is connected to at the moment.

- Select the console in the network menu and tap **Dismiss Station**.

It can be connected to a different console now.

The column Remote IP in the network menu displays to which console the extensions are connected at the moment.

When an extension is successfully connected with a console, it will display the first wing of executors by default.



To display a different executor wing:

- Tap **Menu** - **Settings** - **Extension Configuration** . Within the Extension Configuration menu, it is only possible to change the WingID for the connected extension.

There are 2 column modes: Condensed (default) and Full.

1. The condensed mode only displays information about the connection state, IP, and WingID.
2. The full mode also displays the columns Connected Count and Device Type.

-
- **Connection State:** Can be Yes or No. This cell cannot be edited by the user and displays if the extension is connected with the console or not.
 - **IP:** Displays the IP address of the extension. This cell also cannot be edited by the user.
 - **Wing ID:** Displays the ID of the executor wing the extension controls. This property can be changed by the user.
 - **Connected Count:** The higher the number the more often the extension tried to connect to the console. In a faulty network environment, the number can increase fast due to reconnection. This value cannot be edited by the user.
 - **Device Type:** This column displays the device type of the extension. Typically it is grandMA3 Fader Module Encoder (MFE). This cell also cannot be edited.

The desk light of an extension is controlled together with the desk lights of the console.

The custom section of an extension can mirror the **custom section of the console** or can be independent of the console, depending on the user settings.

To learn more about user settings, read the **User settings topic**.

4.15. Connect grandMA3 fader wing

The grandMA3 onPC fader wing expands the onPC system with additional playback capabilities.

For more information about this device, please refer to **grandMA3 onPC fader wing** in the Device Overview section.



Important:

It is only possible to connect fader wings to grandMA3 onPC systems.
A maximum of two fader wings and one command wing can be connected.



Important:

Independent of the devices connected to the grandMA3 onPC software, the number of parameters can only reach a maximum of 4 096.

For more information, see **Expand the Amount of Parameters topic**.

Procedure:

1. Connect the grandMA3 onPC fader wing with:

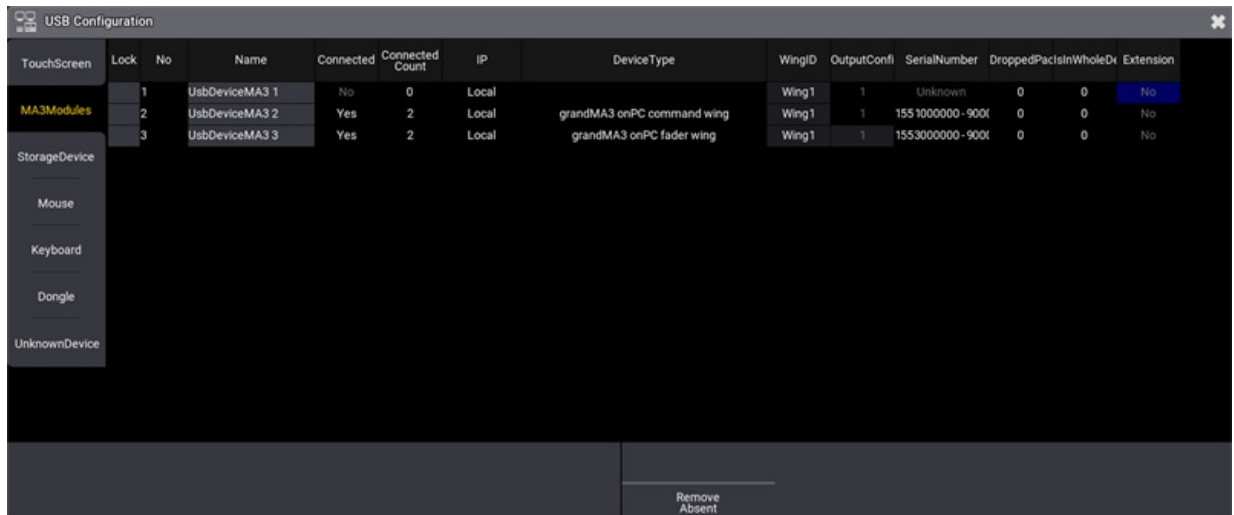
- command wing XT
- PC
- Laptop

2. Start the grandMA3 command wing XT, PC or laptop.

3. Switch on the grandMA3 onPC fader wing.

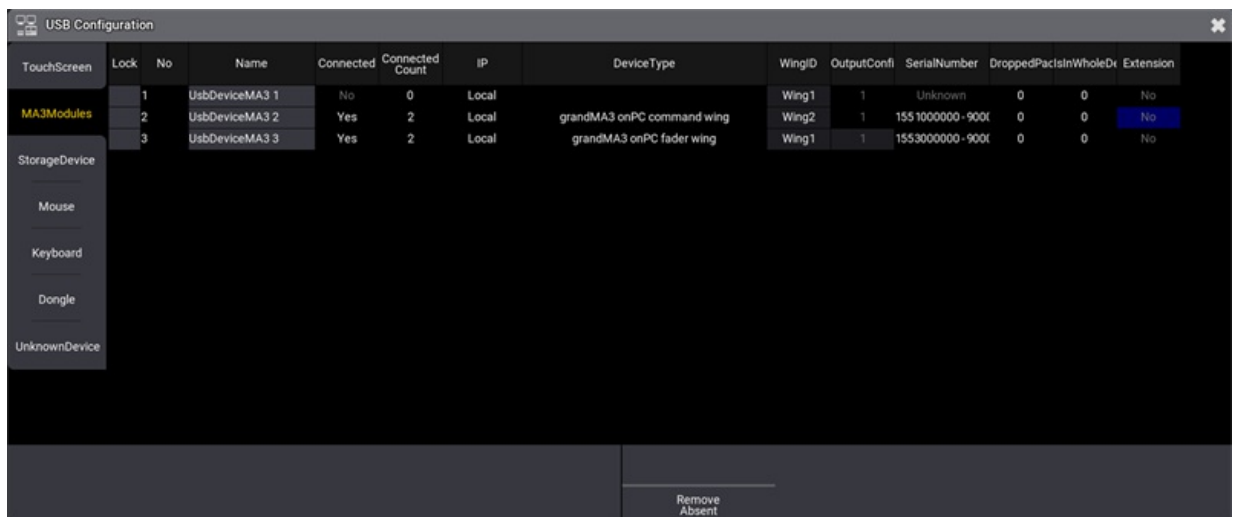
To connect a grandMA3 onPC fader wing in combination with a grandMA3 onPC command wing to expand the number of executors, the wing configuration has to be changed:

1. To check the USB setting, tap **Settings** , **USB Configuration** , **MA3Modules** .



USB Configuration setting

1. To adjust the wing ID of the command wing, tap or right-click **WingID**. A pop-up window opens.
2. To change the WingID to 2, tap **Wing2**.



USB configuration set to Wing2

4.16. Turn on the Device for the First Time



Hint:

The safety instructions and technical specifications can be found in the Quick Manual of the respective product, e.g. consoles:

For information on safety instructions, see [grandMA3 Quick Manual Consoles – Safety](#).

For technical specifications, see [Technical Data](#) in the [grandMA3 Quick Manual consoles](#).

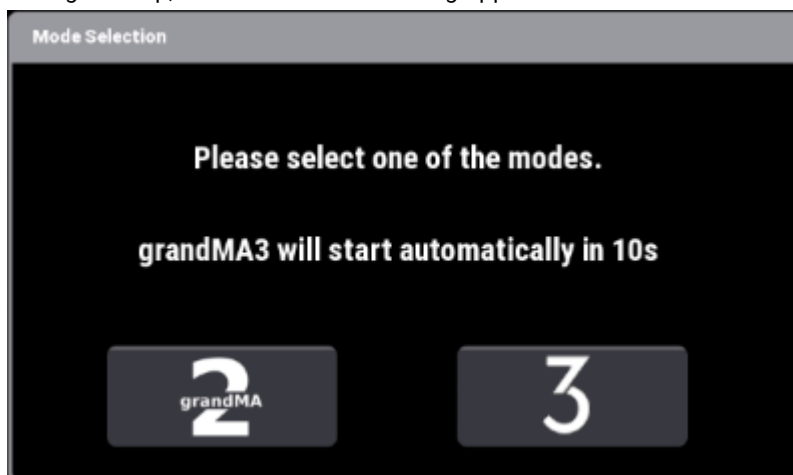
Turn on the grandMA3 Device

1. Unpack the grandMA3 device.
2. Remove packaging and cushioning material.
3. Place the device indoors on a stable surface.
4. Connect external monitors using the native DisplayPort connectors (optional).
5. Connect an external mouse or keyboard using the USB ports (optional).
6. Insert the power connector into the corresponding jack.
7. Connect the power plug with the mains.
8. Turn on the power switch on the rear panel.
9. Press the power key on the front panel.

The device starts booting.

Select Mode

During start-up, the **Mode Selection** dialog appears:



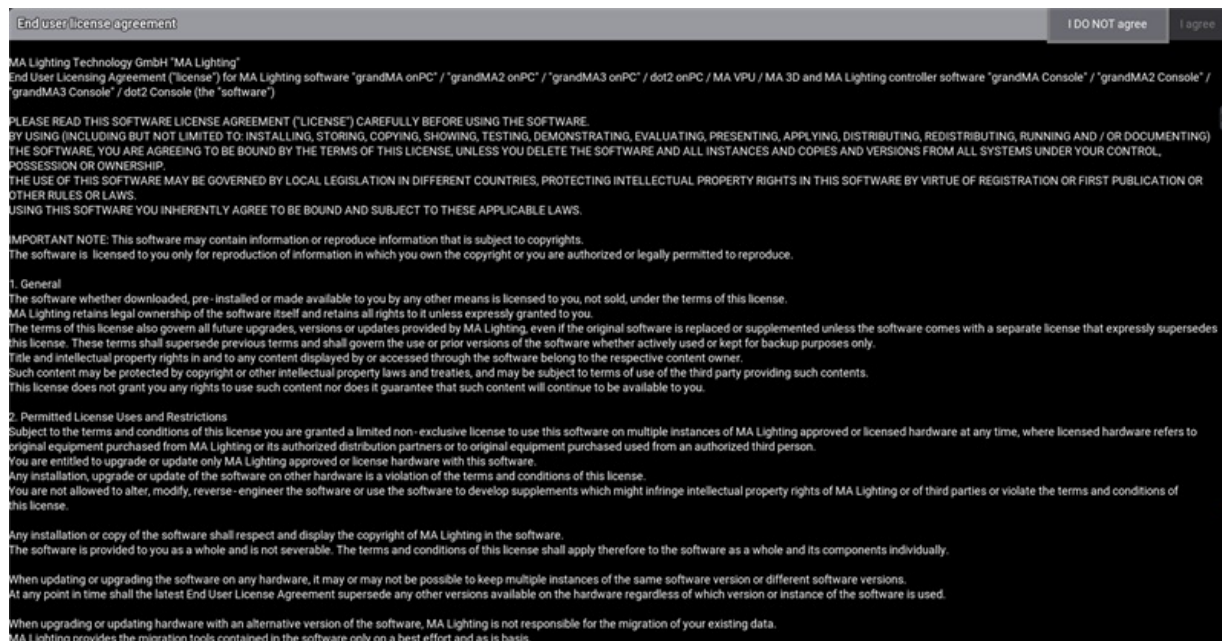
- Tap to select a mode.

For more information, read the [Mode2 topic](#) in the section grandMA3 Mode2 of the [grandMA2 User Manual](#).



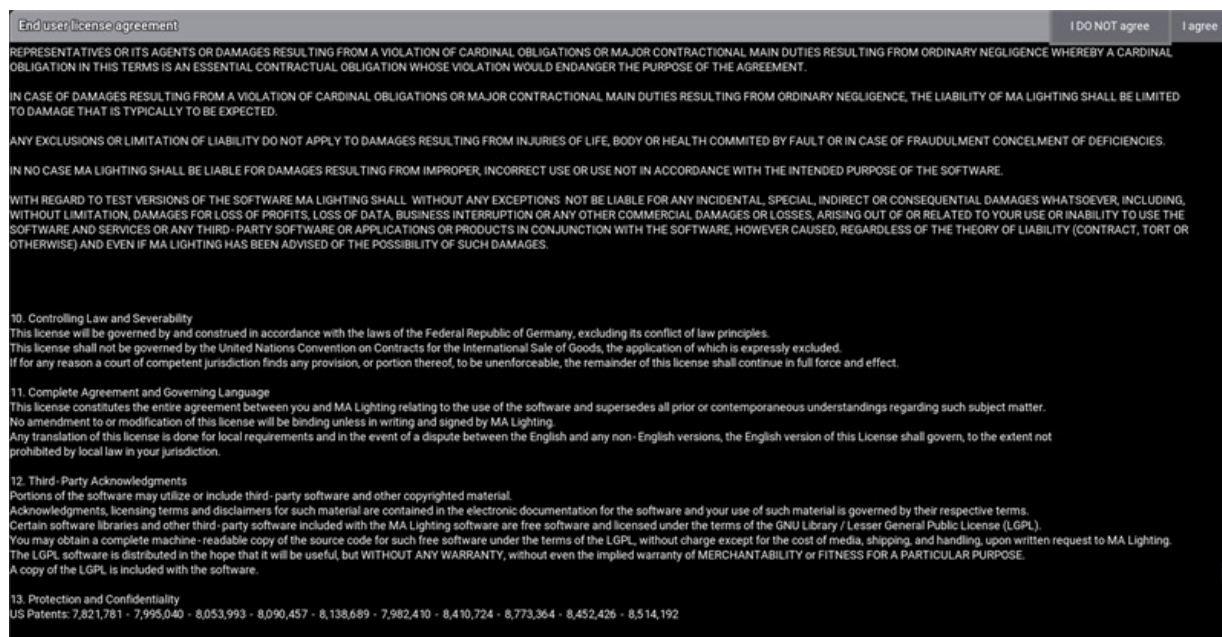
End-User License Agreement (EULA)

After selecting grandMA3 mode, the console starts to boot and the pop-up **End-User License Agreement (EULA)** opens.



End-user License Agreement (EULA)


- Scroll down to read the complete EULA. The button **I agree** in the upper right corner of the pop-up turns white.





End-user License Agreement (EULA) I agree button

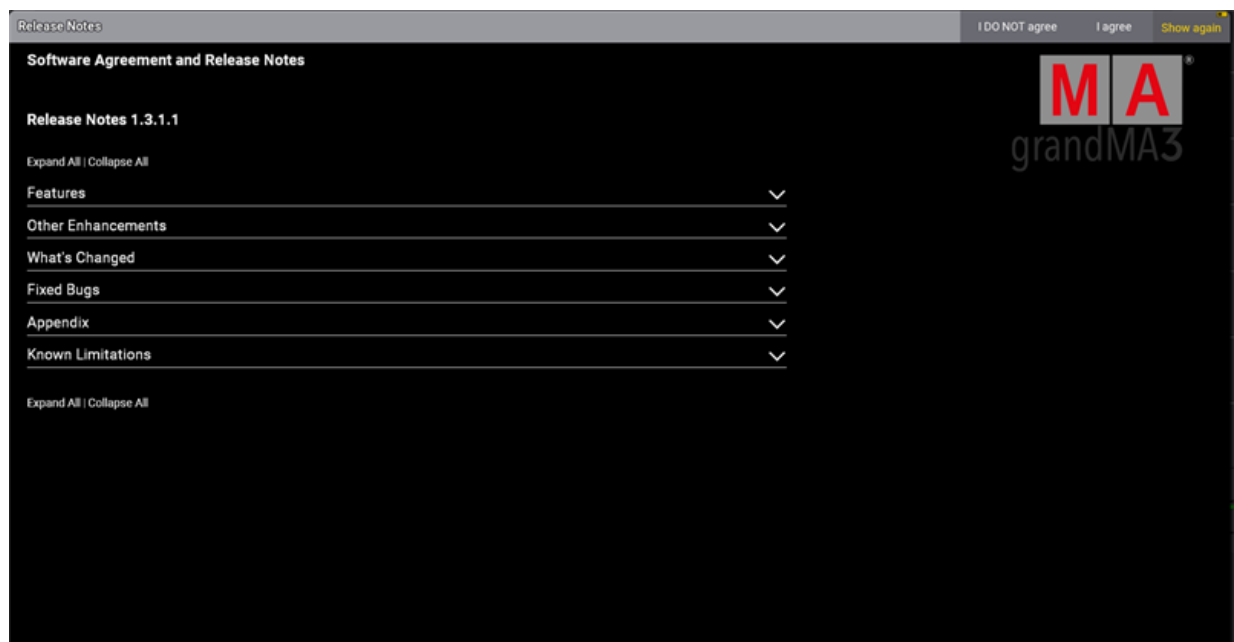
- To confirm the EULA, tap **I agree**.

**Hint:**

The EULA will only be displayed the first time of booting the console and every first boot after updating the software.

Release Notes

The Release Notes of the current version appear on the screen.



Release Notes

- To confirm the Release Notes, tap **I agree**.
- If you do not want the software to show the Release Notes again after booting, tap **Show again** to disable.

You can now use your grandMA3 device.

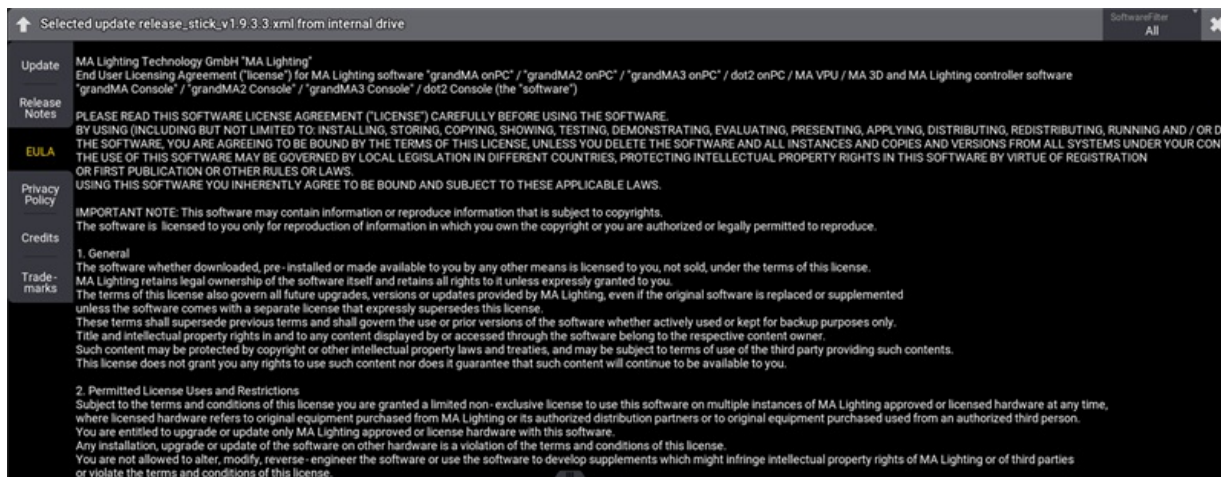
Third-party Software

The software comes with standard codecs to play videos and display images.

When you install third-party software, make sure that you have the required licenses.

Third-party software can be activated by accepting the additional third-party software agreement.

- To accept the additional third-party software agreement, **open the Software Update menu**.



- Tap **EULA**, then tap **Third-Party Software**.



- Tap **I AGREE** to accept the additional third-party software agreement



5. grandMA3 onPC

MA Lighting offers the perfect solutions for getting started with. Simply download the grandMA3 onPC software and connect to any of the grandMA3 onPC products to unlock parameters.

The amount of DMX outputs varies with each grandMA3 onPC product but all of them unlock parameters for the full grandMA3 software functionality. There is no compromise in feature set and the range of products allow you to make the right choice for your budget or individual needs.

There are three choices for onPC solutions: the grandMA3 onPC command wing, which offers the ultimate grandMA3 mobile solution, the grandMA3 onPC command wing XT, featuring a built-in MA mother board, and the grandMA3 onPC xPort Nodes, which offer the most cost-effective DMX output solution for software only show control.

Every grandMA3 onPC xPort Node is capable of handling MA-Net3, MA-Net2, sACN and Art-Net data and is fully RDM compliant. A 3.9" color display on the front face allows for simple configuration and provides a quick overview of the status of each Node.

All grandMA3 onPC xPort Nodes are equipped with a powerful processor and a 1,000Mbit/s Ethernet connection to ensure the most reliable and stable Ethernet to DMX and vice versa conversion.

The grandMA3 onPC xPort Nodes are high quality solutions that can handle busy Ethernet connections without ditching DMX or RDM messages.



5.1. System Requirements grandMA3 onPC

If you would like to run grandMA3 on your PC, here is what it takes.

Type	Minimum	Recommended
Operating system	Windows® 10 64bit version 1803 or later, Windows® 11 64bit version 21H2 or later, Apple® macOS® Catalina 10.15 or later, Admin rights are required for all operating systems.	
Processor	6th Generation Intel® Core CPU™ (processor-based platform or later) with 4 or more cores and SSE 4.2 or comparable AMD® CPU, Apple® M1® SoC CPU with 8 or more cores, Apple® M2® SoC CPU with 8 or more cores	
RAM	8 GB	16 GB
Hard disk	32 GB of available space	type SSD
Graphics card	Any graphic card with hardware acceleration, OpenGL 4.1 support, and 1024 MB VRAM, or Apple® M1® SoC GPU with 8 or more cores. Apple® M2® SoC GPU with 8 or more cores, with hardware acceleration, OpenGL 4.1 support, and 8 GB unified memory	4 096 MB VRAM 32 GB unified memory
Resolution	1 920 x 1 080 or higher.	
Network card	1000BASE-T	

Additional requirements:

- To use the grandMA3 onPC in session with grandMA3 consoles, a minimum of 16 GB of RAM (24GB Unified Memory when using Apple® M1® SoC or Apple® M2® SoC) is required.
- To use the online help and download the latest version of grandMA3 onPC, make sure that you have internet access.
- To save the software on a USB flash drive, use a USB 2.0 or 3.0 port.



Hint:

In general, we recommend more cores and more cache memory size instead of a higher processor clock speed.



Hint:

For Apple® macOS® High Sierra 10.13 or Apple® macOS® Mojave 10.14 , grandMA3 onPC version 1.8.8.2 is required.



Important:

Unfortunately AMD Radeon driver 22.11.2 creates graphical issues. We recommend to use AMD Radeon Driver 22.5.1.



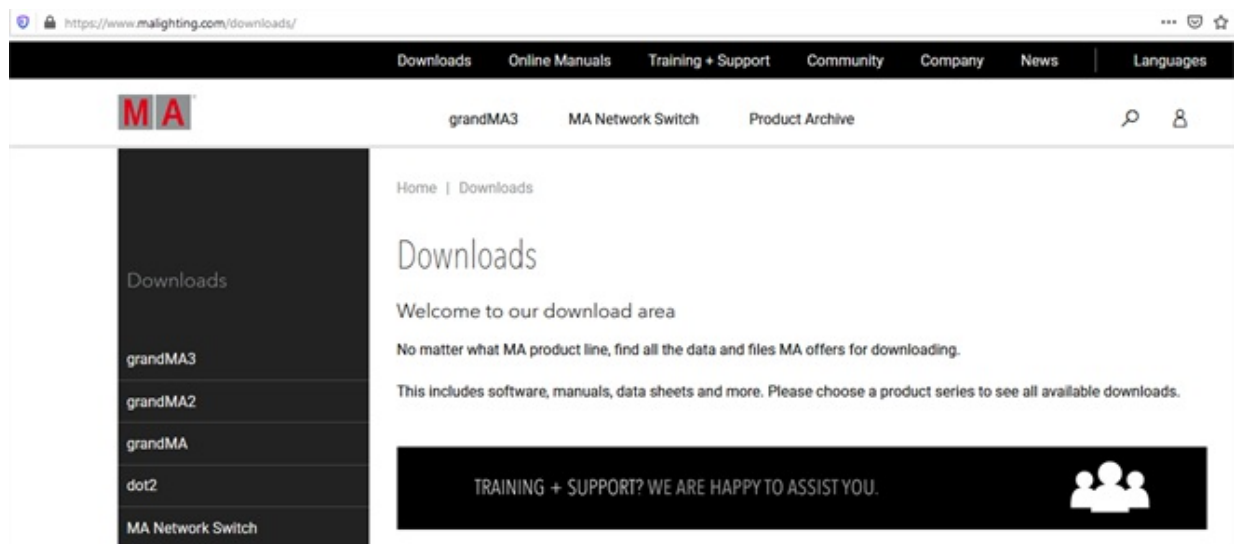
5.2. Windows™® Installation

To run the grandMA3 onPC software, copy and install the program files on your PC.

The installation is possible in every root directory or in the **standard directory** "C:\Program Files\MALightingTechnology".

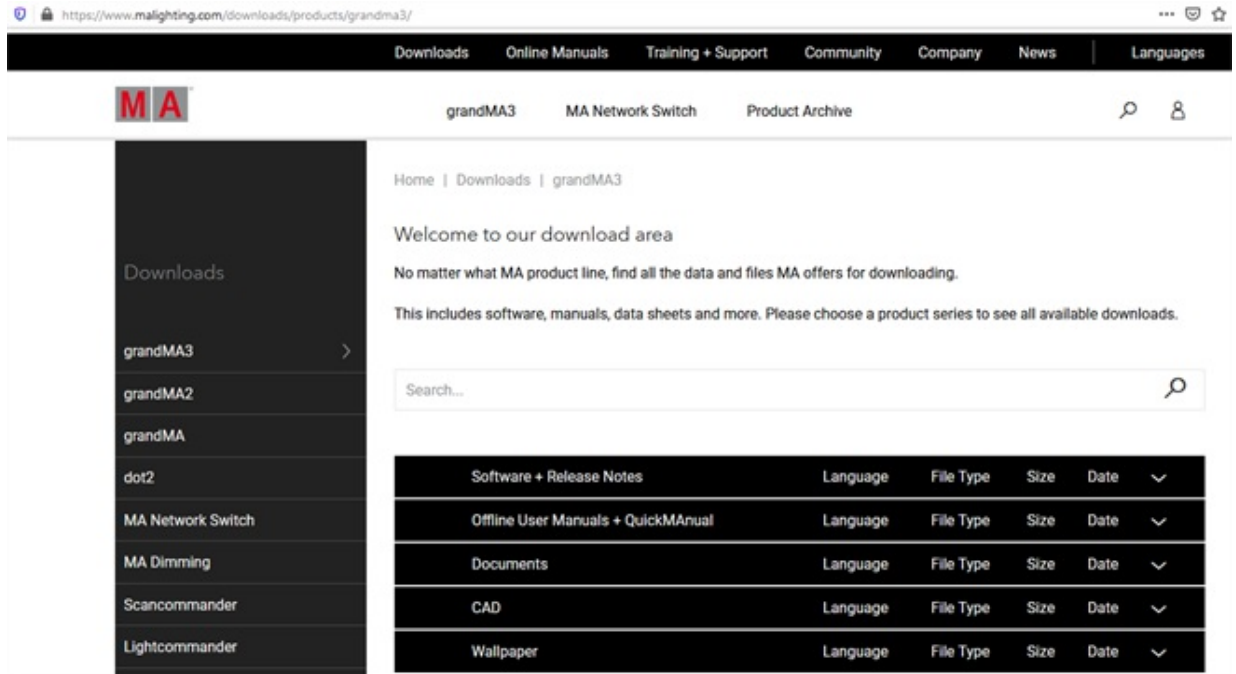
To download the grandMA3 software:

1. Go to www.malighting.com, click **Downloads**.



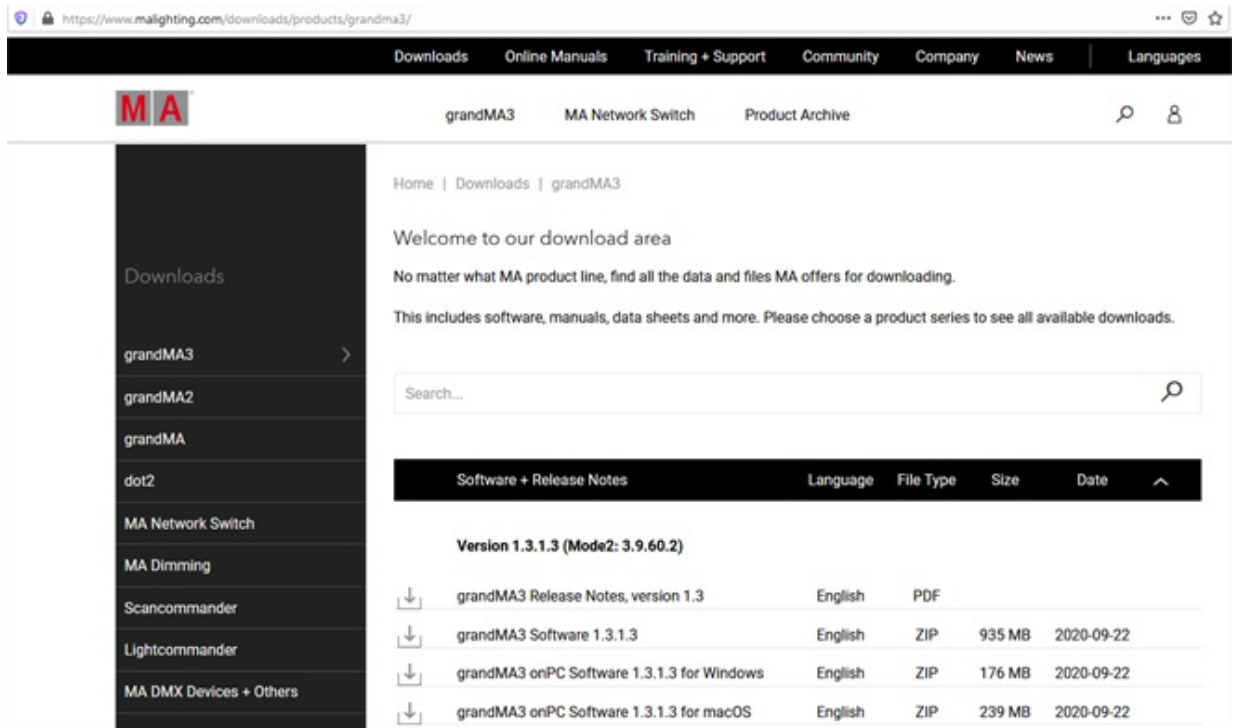
Website malighting.com Downloads

2. Click **grandMA3** in the bar on the left.



grandMA3 section

3. Click **Software + Release Notes** to find the latest software version.



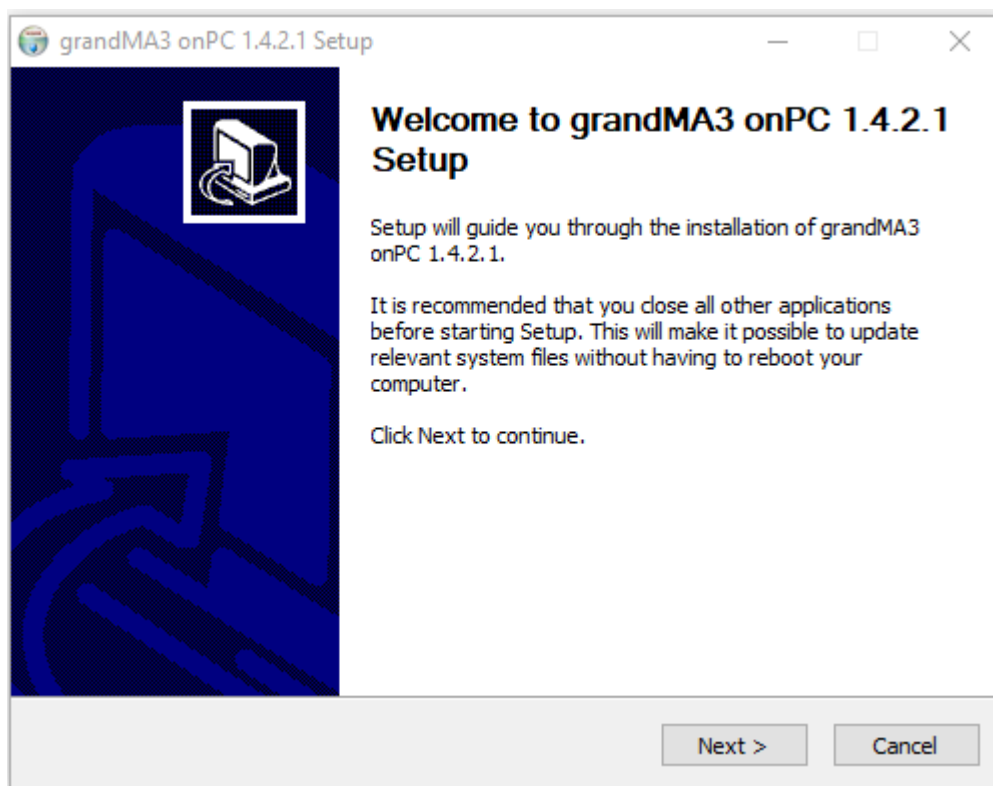
grandMA3 Software section

4. Click grandMA3 onPC Software x.x.x.x for Windows to download the desired installation package.

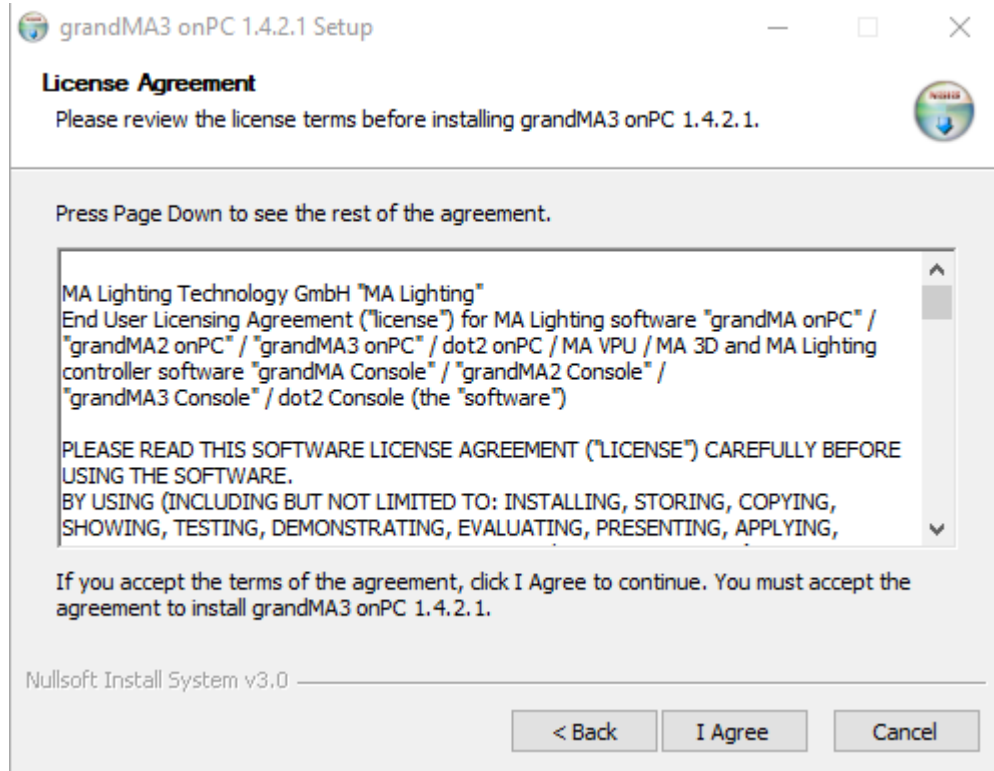
The download process starts.

- Extract the zip file.
- Go to the subfolder "ma".
- Double-click the installation program grandMA3_onPC-vx.x.x.x.exe.
- Confirm with Execute.

The installation program opens:

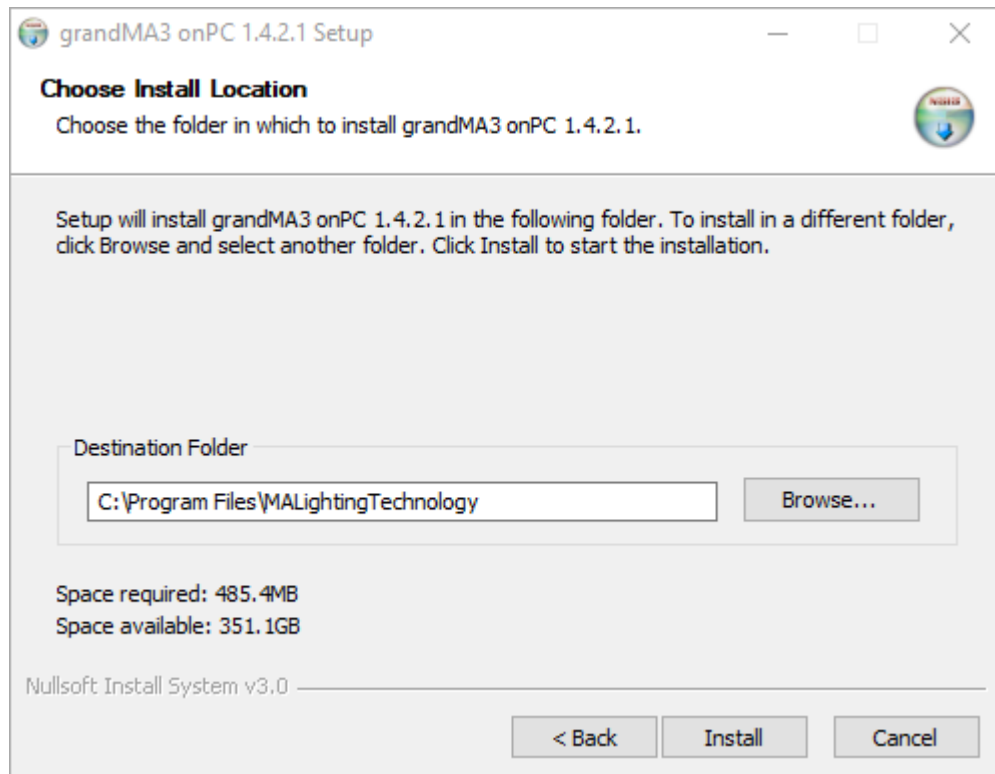


- Click **Next**. The End User License Agreement (EULA) appears:

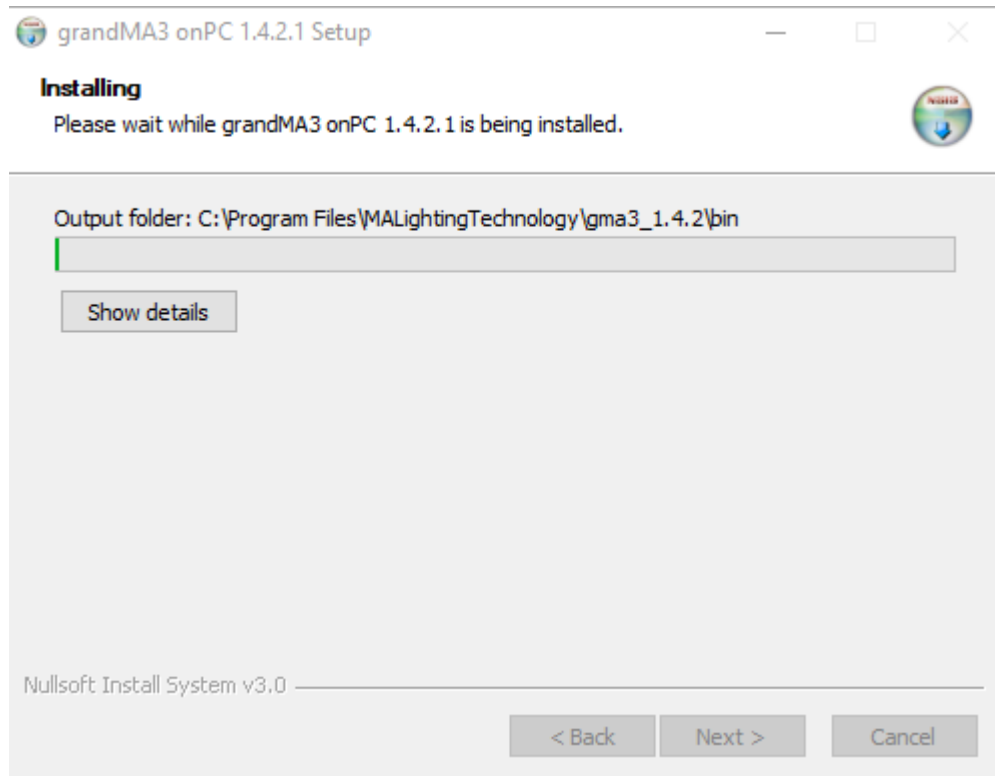


- Click **I Agree** to accept the agreement.

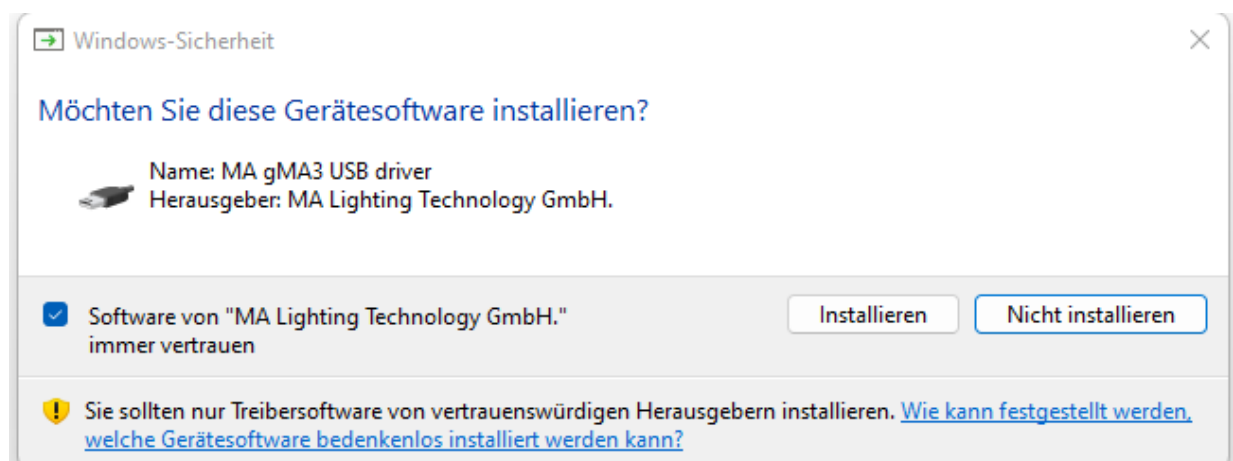
The Install Location window appears:



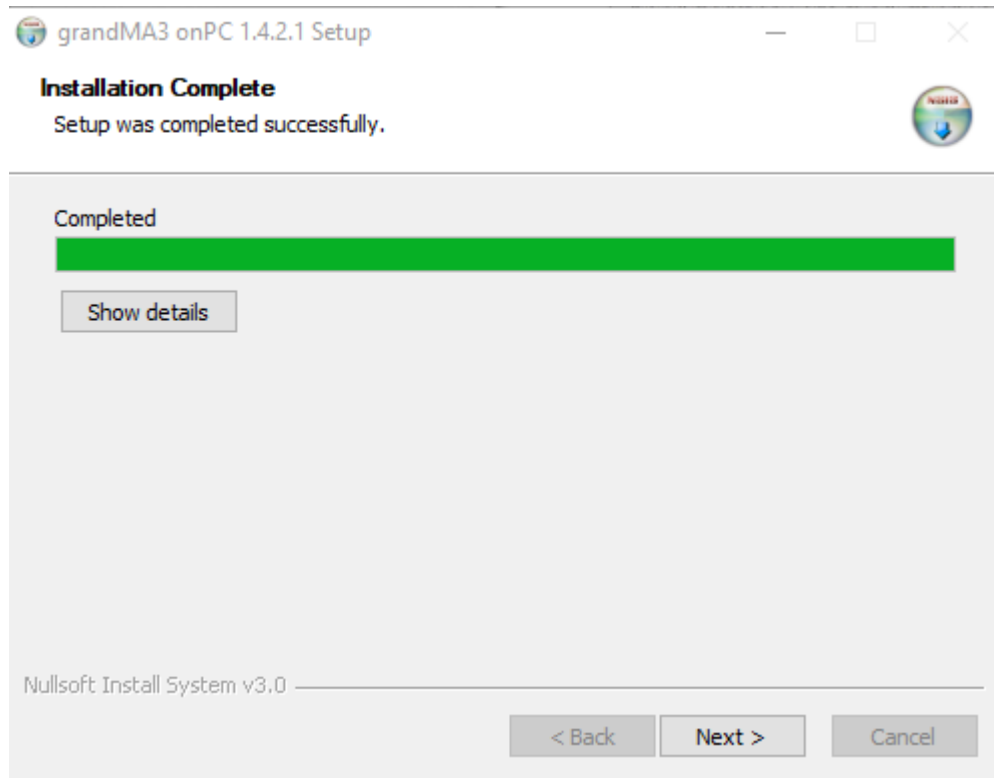
- Watch out for the suggested directory and change it if you would like to do so.
- The standard directory for the resources and user files is "C:\Program Files\MALightingTechnology".
- To learn more about the folder structure, read the **Folder Structure topic** in **File Management**.
- To confirm the installation directory, click Install.
- The program files are copied into the selected directory.



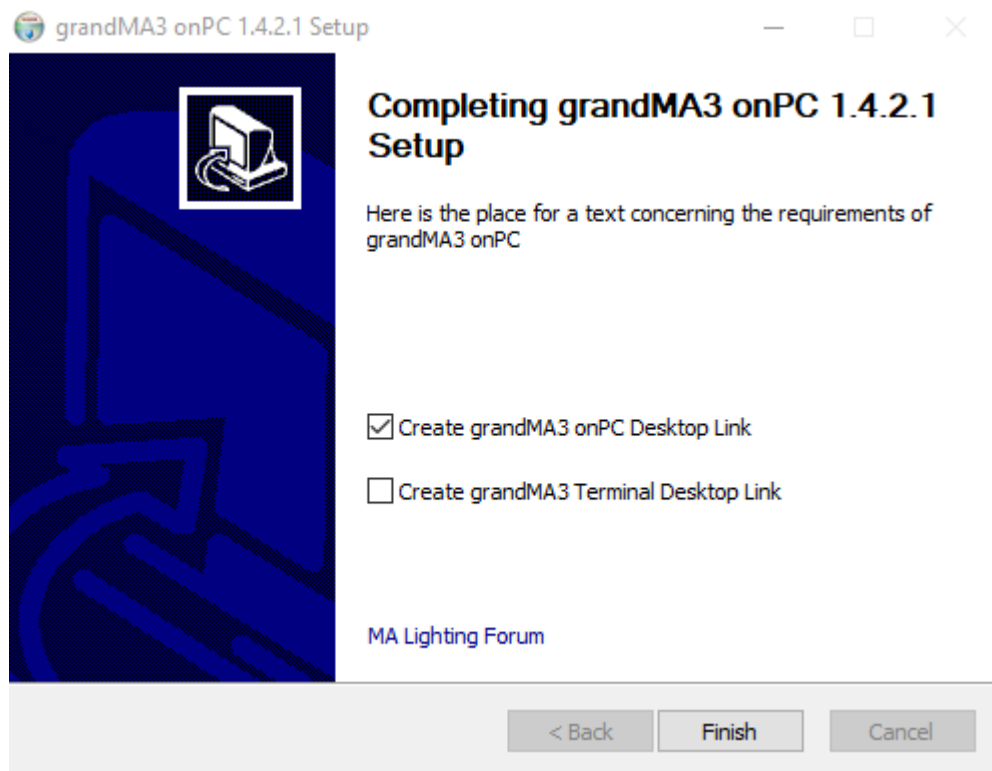
1. To install a USB driver, connect a usb stick before the installation.
2. During the installation, a pop-up window appears. Tap **Install**.



Click Next to finish the installation.

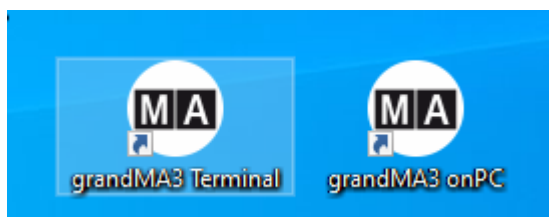


At the end of the installation process, you can choose to set a link to the terminal app or not.



It is disabled by default.

Click the checkbox if you want to create a desktop link to the terminal app.



grandMA3 onPC for Windows

As soon as the installation is finished, the MA icon appears on the desktop.

If several versions of the software have been installed, several subfolders are created in the main folder "C:\ProgramData\MALightingTechnology".



The MA icon on the desktop will always link to the latest version:

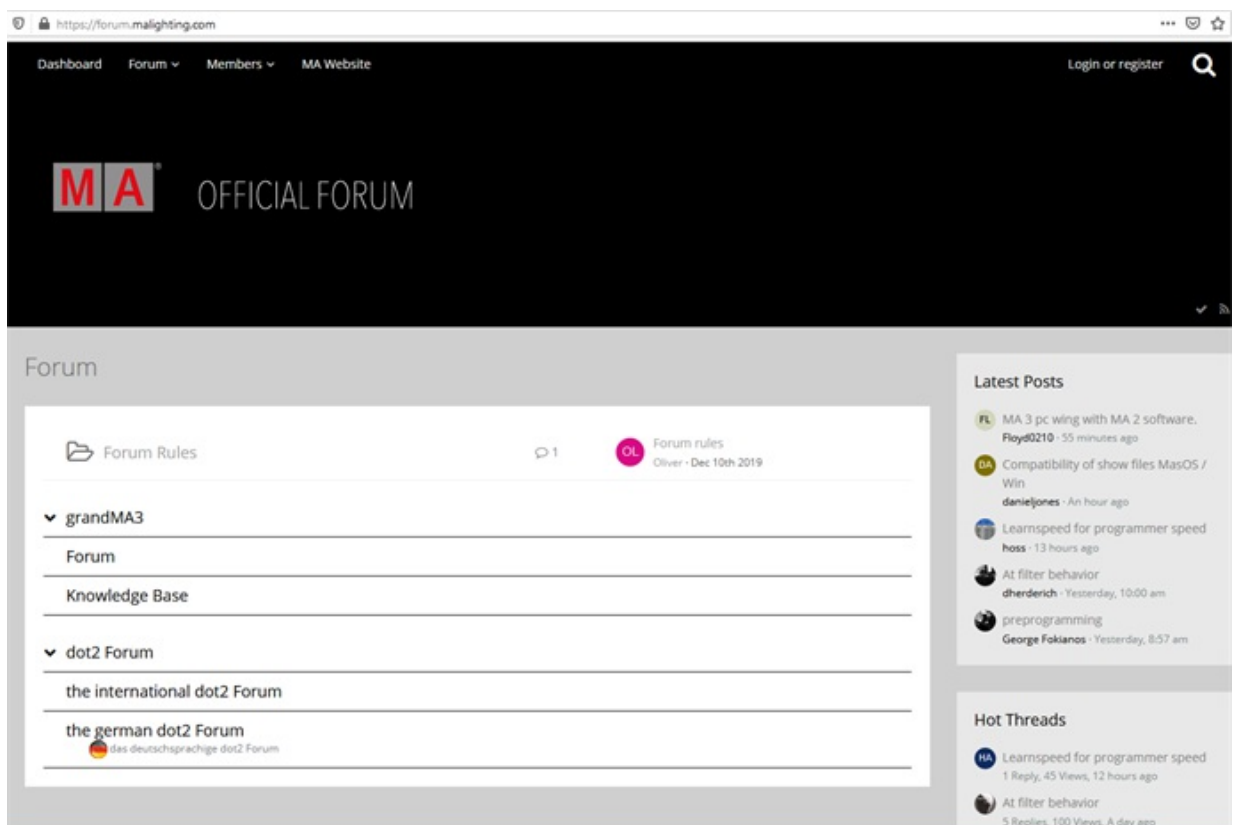


This is the main app.

MA Lighting Forum

- To visit the MA Lighting Forum on the internet, click [MA Lighting Forum](https://forum.malighting.com).

The MA Lighting Forum opens in the browser:



5.3. Optimize Windows™®



Hint:

The grandMA3 onPC software is available for Windows™® and **macOS™®!**

- Please refer to the **[system requirements topic](#)** to learn more about the computer specifications.
- For more information about the grandMA3 onPC app, the terminal app, and the different version icons, read the **[Installation of grandMA3 onPC topic](#)**.
- To run the grandMA3 onPC software even more efficiently, we recommend adjusting the following settings on your computer.

Turn on High-Performance Power Plan

To turn on the high-performance power plan:


1. Click the **Start** button.
2. Type Control Panel in the search bar.
3. In the search bar of the Control Panel, type power options.
4. Click **Power Options**.
5. Under Preferred plans, click **High performance**.

Choose or customize a power plan

A power plan is a collection of hardware and system settings (like display brightness, sleep, etc.) that manages how your computer uses power. [Tell me more about power plans](#)

Preferred plans

- Balanced (recommended)** [Change plan settings](#)
Automatically balances performance with energy consumption on capable hardware.
- High performance** [Change plan settings](#)
Favors performance, but may use more energy.

Show additional plans 

The high-performance power plan is turned on.

Optimize High-Performance Power Options

- Turn off everything that can interrupt the grandMA3 onPC.

To optimize the power options:

1. Click **Change plan settings** next to High performance



- High performance**
Favors performance, but may use more energy.

[Change plan settings](#)



2. For Turn off the display, choose **Never**.
3. For Put the computer to sleep, choose **Never**.

Change settings for the plan: High performance

Choose the sleep and display settings that you want your computer to use.

Turn off the display:

Put the computer to sleep:

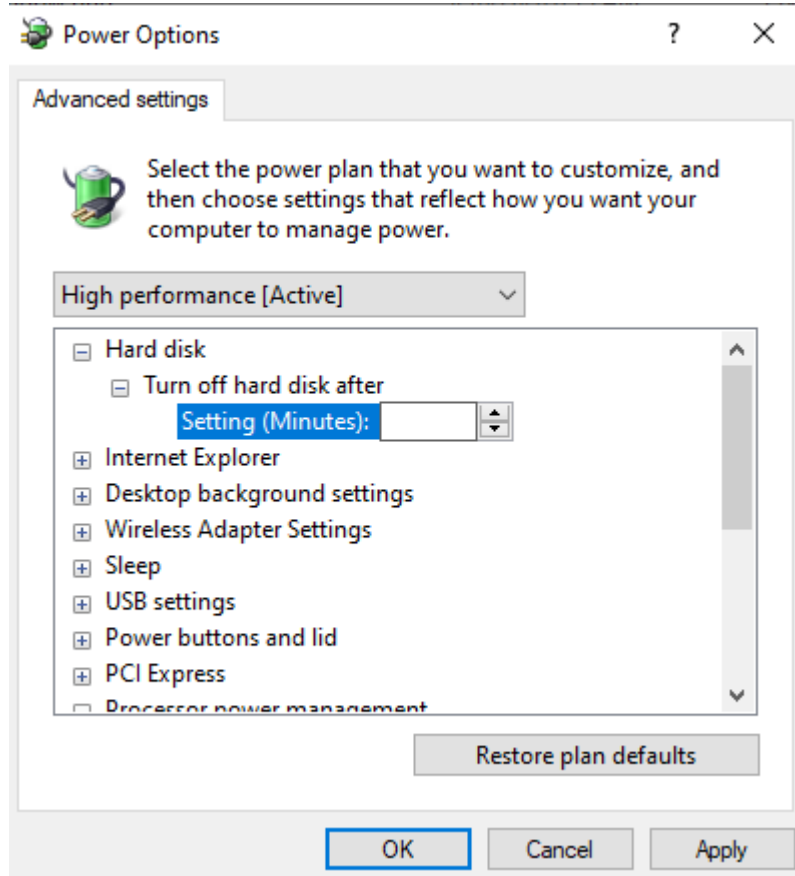
[Change advanced power settings](#)

[Restore default settings for this plan](#)

Save changes

Cancel

4. Click **Save changes**.
5. Click **Change advance power settings**.



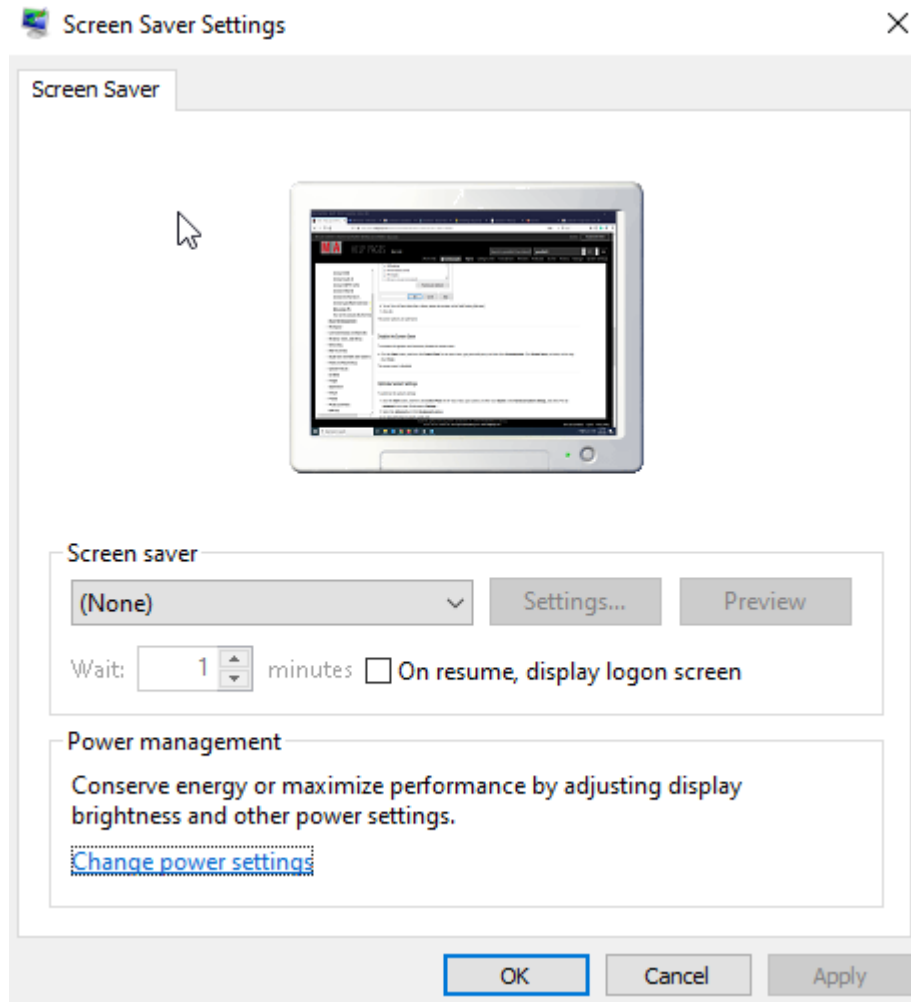
6. To set Turn off hard disk after to Never, delete the number in the field Setting (Minutes).
7. Click OK.

The power options are optimized.

Disable the Screen Saver

To conserve the graphic card resources, disable the screen saver.

1. Open the Control Panel.
2. In the search box, type Screen Saver.
3. Click **Change Screen Saver** to open the menu.
4. In the Screen saver select **(None)**.
5. Click **OK**.

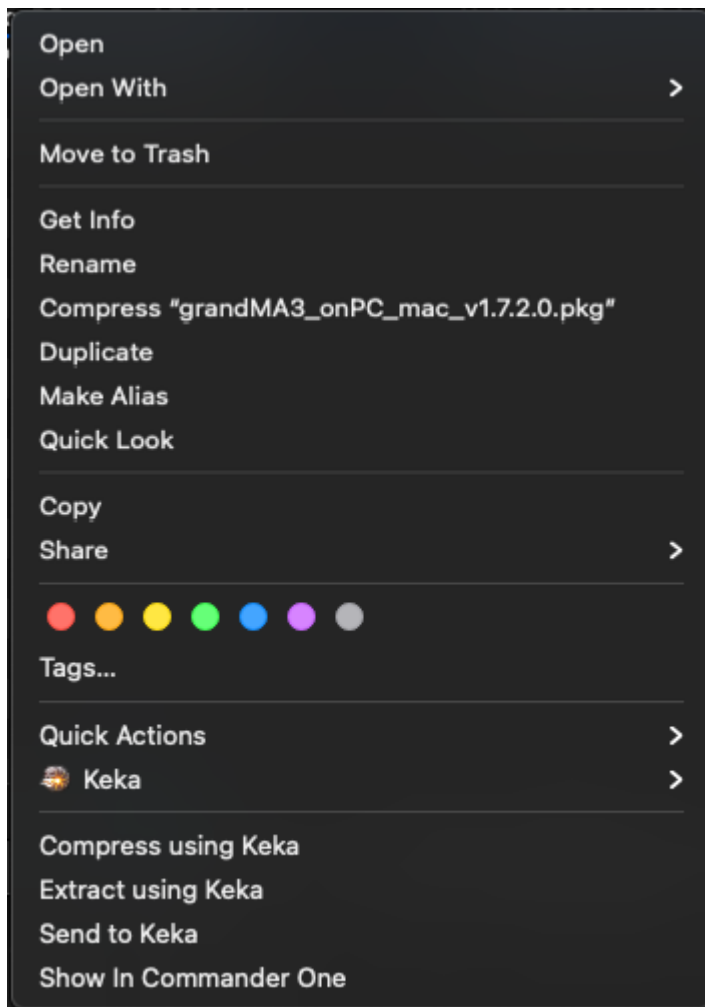


The screen saver is disabled.

5.4. macOS® Installation

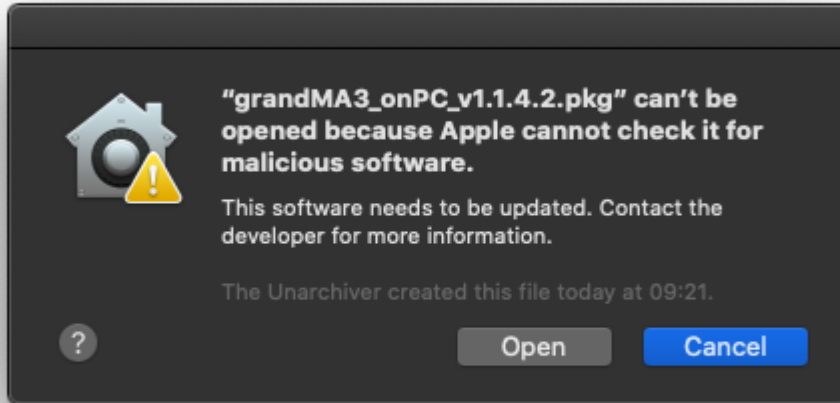
To install grandMA3 onPC on macOS®:

1. Download the installation file from www.malighting.com.
2. Click `grandMA3 onPC Software x.x.x.x for macOS` to download the installer.
3. Open the zip file, double-click it.
4. Go to the subfolder "ma".
5. Right-click the installation program `grandMA3_onPC_x.x.x.x.pkg`.
The context menu opens.

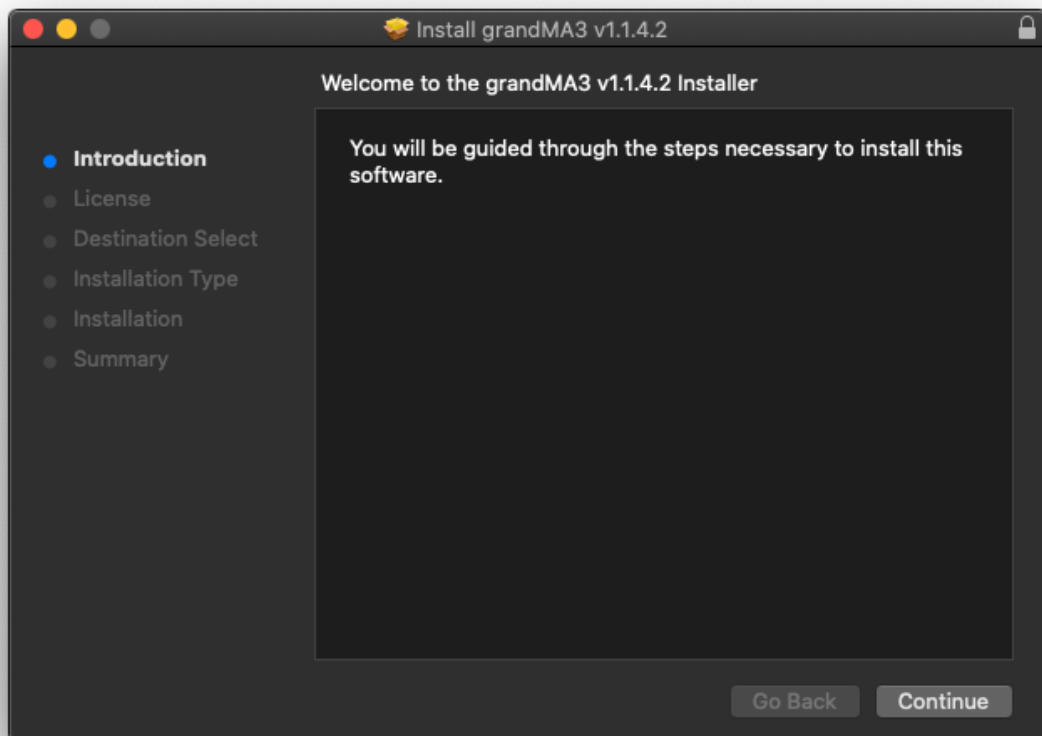


macOS context menu

6. Click `Open`.
7. Click `Open` to confirm the warning message:



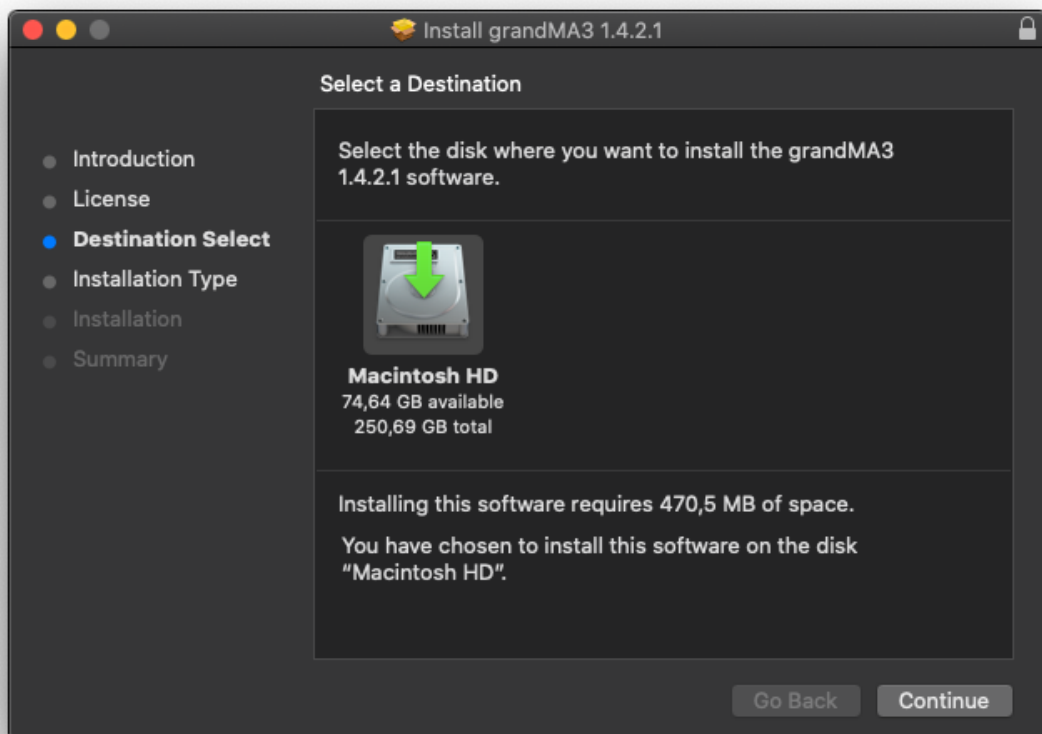
8. Click **Continue** when the installation program opens:



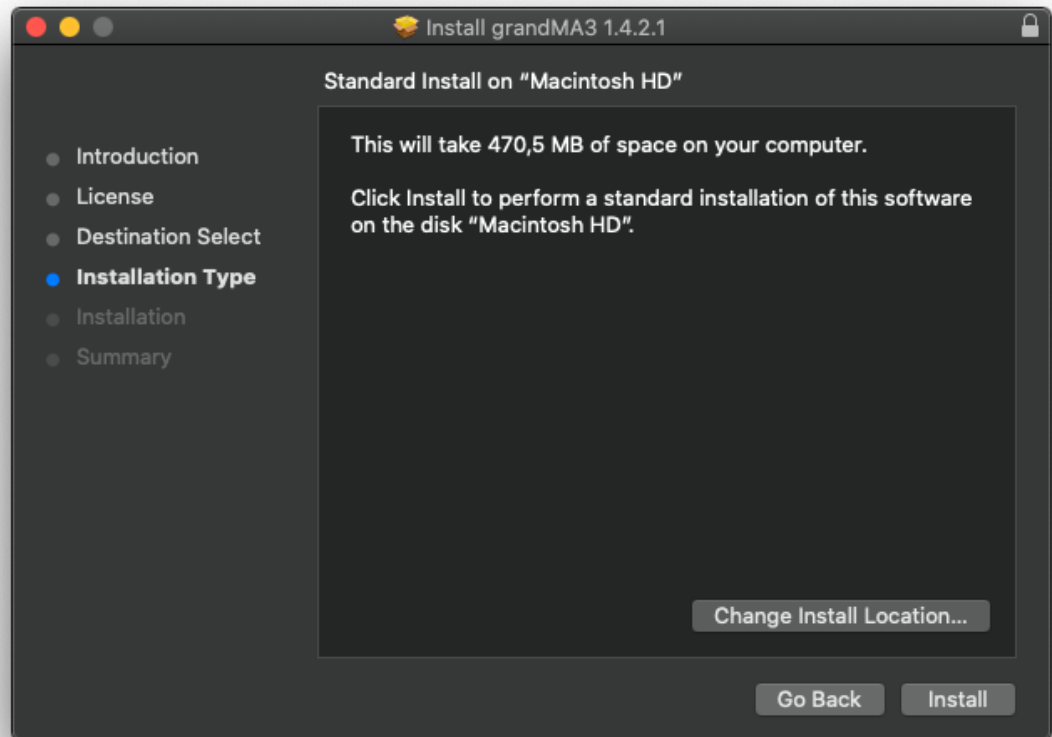
9. The software license agreement opens. Click on **Continue** and then on **Agree**.

The software is installed in the standard directory /Applications/. The resources are located in /Users/.

10. Optional: Change the destination disk, select the desired disk and click **Continue**.

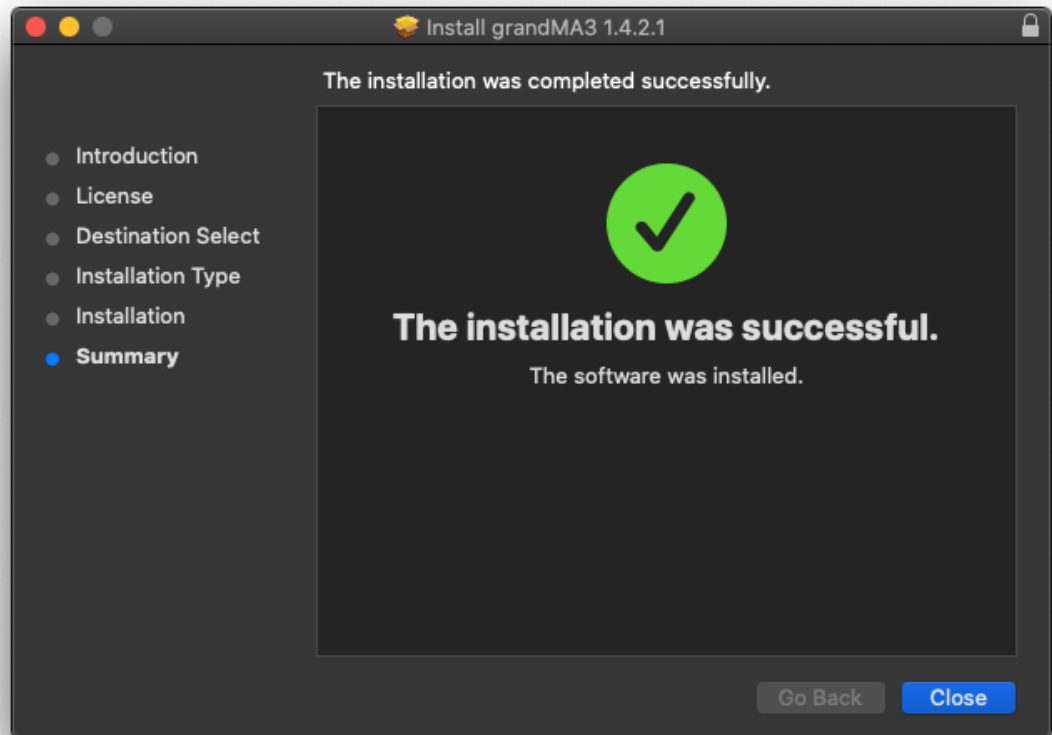


11. Click **Install** to install the application or **Change Install Location** to change the destination:



12. To install the software, click **Install**. During the installation, it is possible that you are asked for the administrator password.

13. Click **Close** after the installation is complete:

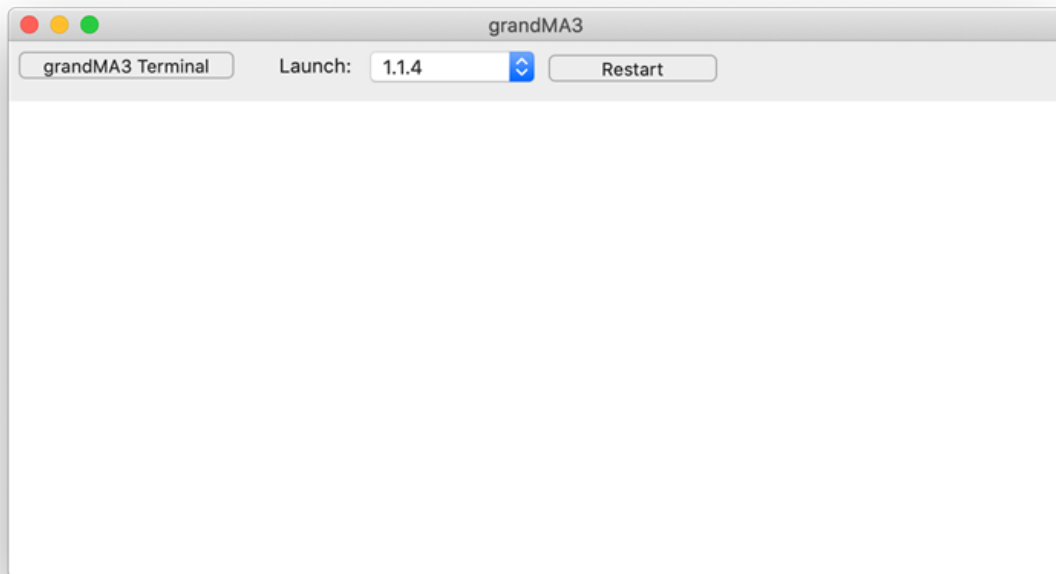


Hint:

If you receive an "Installation failed" error message during the installation, move the ma folder outside the Downloads folder, for example, to the Desktop folder, and start the installation process again.

Start the Application

1. Go to Applications in the finder and double-click grandMA3.
2. The grandMA3 launcher starts.
3. The launcher will automatically start the last installed version.



4. To choose another version, select the desired version via the launch drop-down.
5. Click **Restart** in order to start the selected version.
6. Confirm the shutdown in the running application.



5.5. Optimize macOS®

The grandMA3 onPC software is available for macOS™® and **Windows™®**.

Please refer to the **system requirements topic** to learn more about the computer specifications.

For more information about the grandMA3 onPC software and the terminal app, read the **Installation of grandMA3 onPC topic**.

To run the grandMA3 onPC software even more efficiently, we recommend adjusting the following settings on your computer.

Follow the following steps for the best results.



Hint:

The same setting options may be available for other operating system versions, too.

Apple® macOS® Catalina 10.15

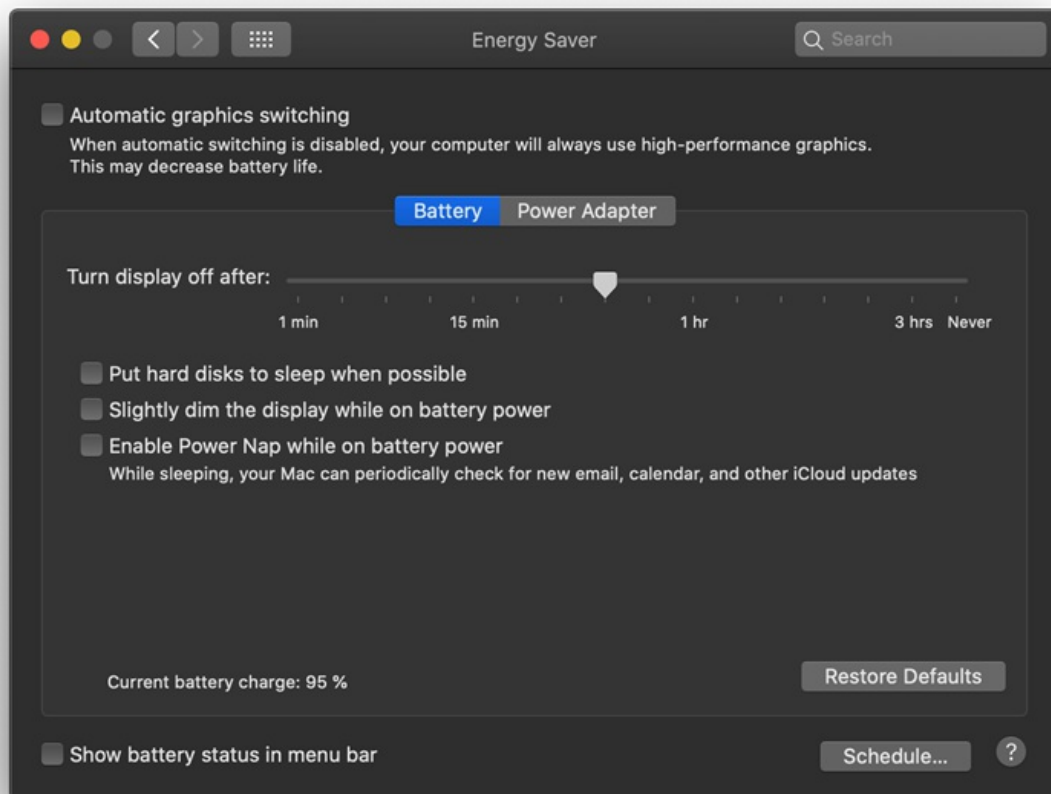
- Click **Energy Saver** in System Preferences.



System Preferences

Settings for Battery

- Automatic graphics switching: Off
- Turn display off after: Any time.
- Put hard disks to sleep when possible: Off
- Slightly dim the display while on battery power: Off
- Enable power nap while on battery power: Off



Battery



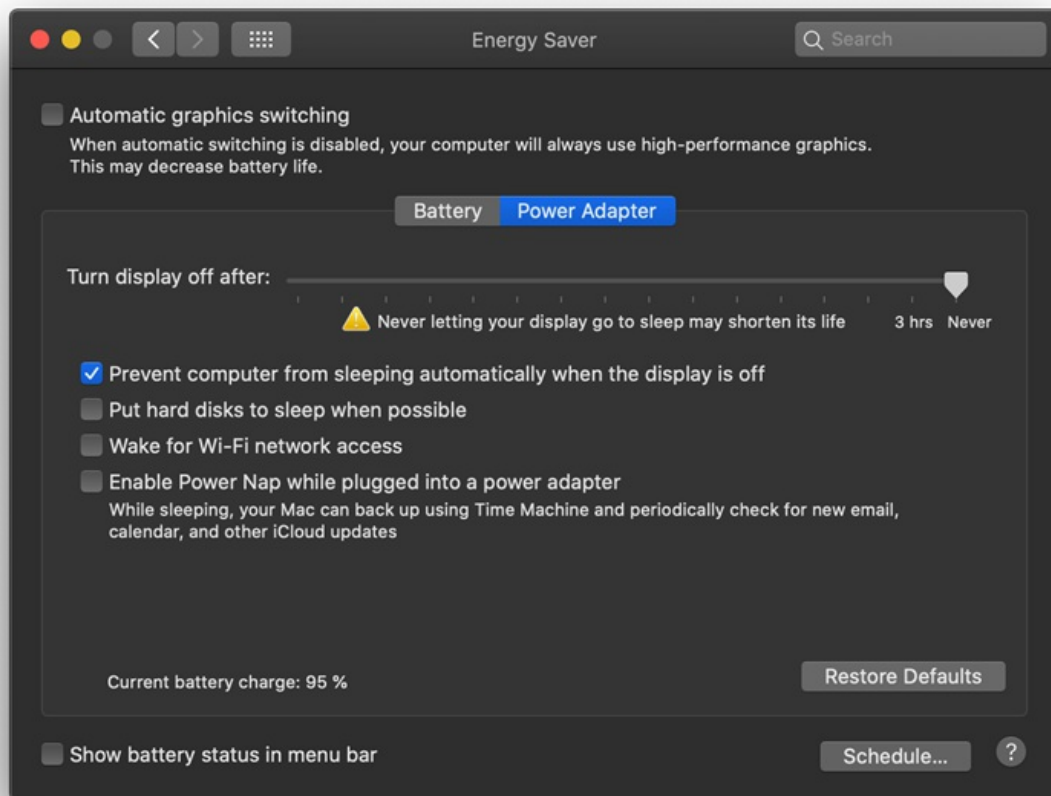
Hint:

When working with grandMA3 onPC on macOS® by using the battery only, high-performance settings would also be required.

We recommend to always connect the computer with the power adapter.

Settings for Power Adapter

- Automatic graphics switching: Off
- Turn display off after: Never.
- Prevent computer from sleeping automatically when the display is off: On
- Put hard disks to sleep when possible: Off
- Wake for Wi-Fi network access: Off
- Enable power nap while plugged into a power adapter: Off

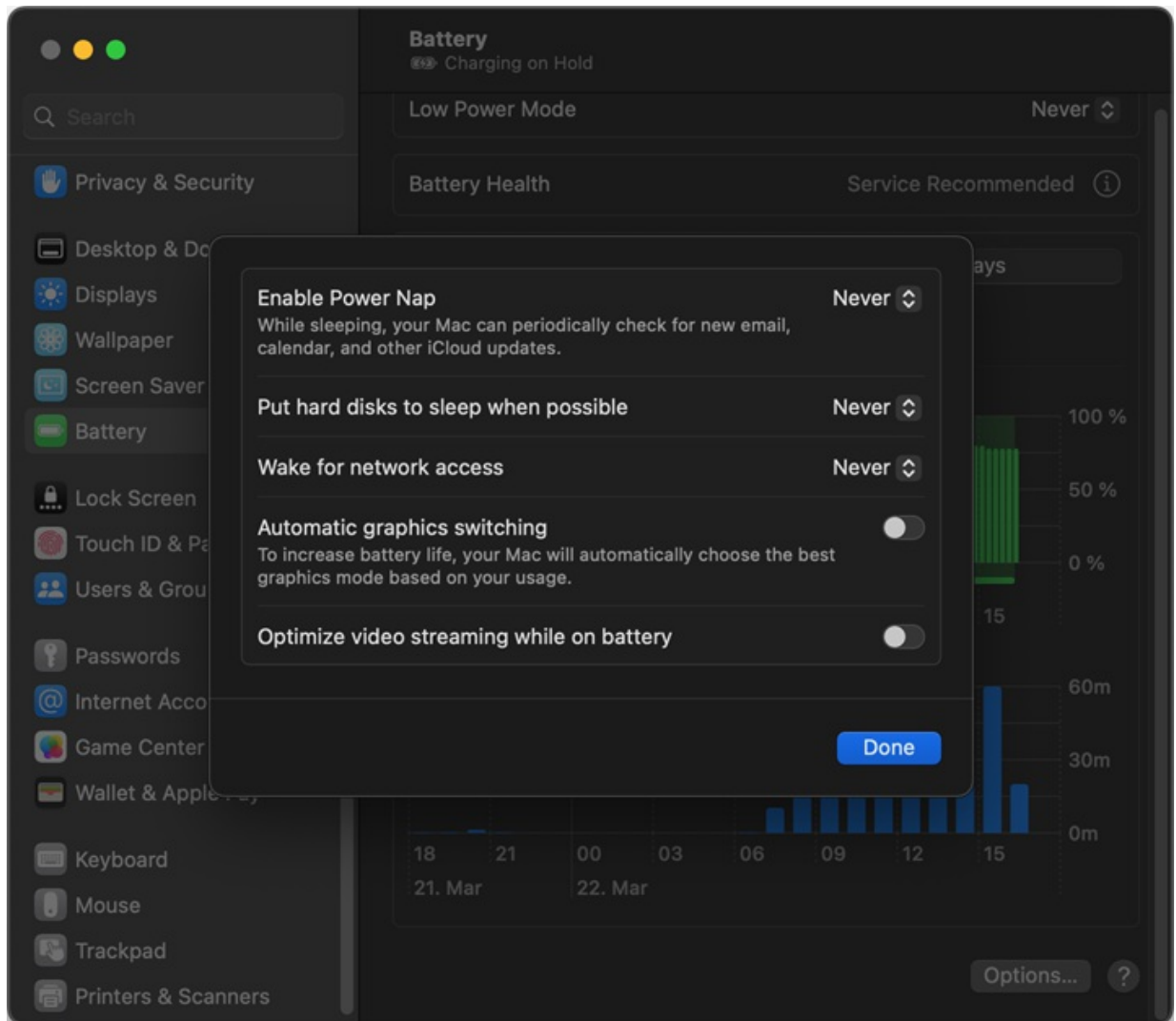


Power Adapter

Apple® macOS® Ventura 13

Settings for Battery

1. Click **Battery** in System Preferences and then click **Options...**

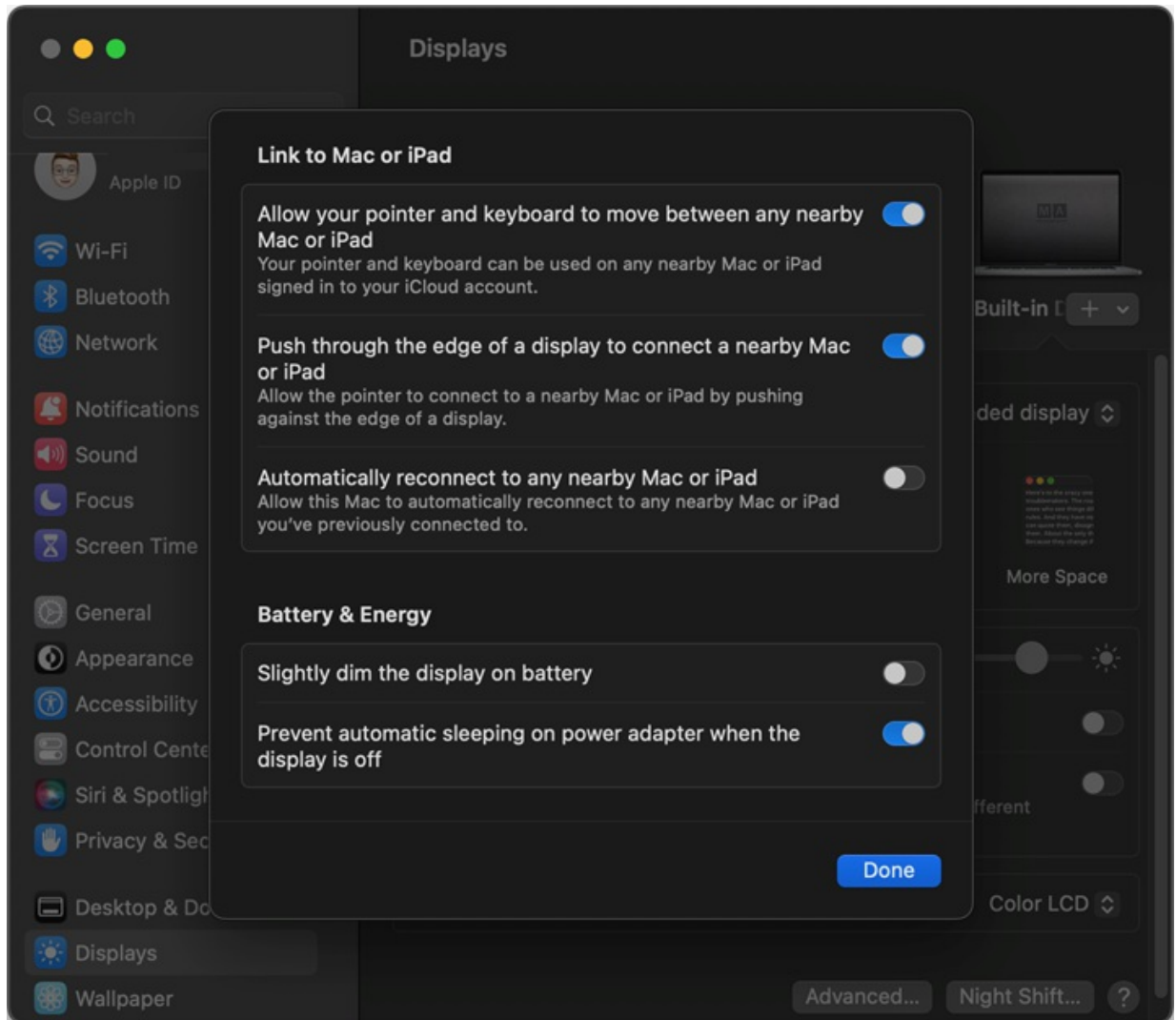


Battery settings

- Enable Power Nap: Never
- Put hard disks to sleep when possible: Never
- Wake for network access: Never
- Automatic graphics switching: Off
- Optimize video streaming while on battery: Off

Settings for Display

1. Click **Displays** in System Preferences and then click **Advanced...**

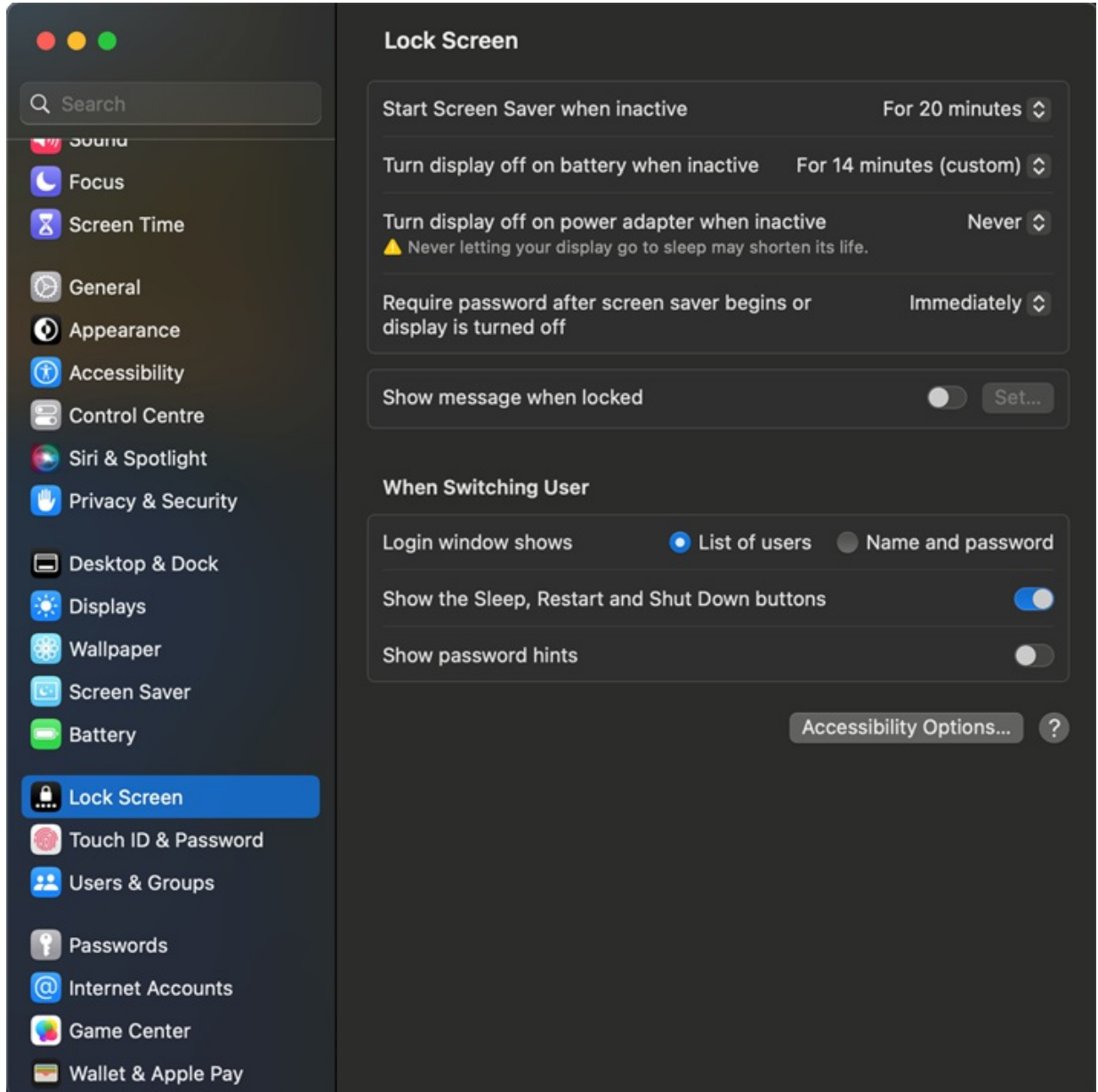


Display settings

- Slightly dim the display on battery: Off
- Prevent automatic sleeping on power adapter when the display is off: On

Lock Screen settings

1. Click **Lock Screen** in System Preferences.



Lock Screen settings

- Start Screen Saver when inactive: Any time
- Turn display off on battery when inactive: Any time
- Turn display off on power adapter when inactive: Never

5.6. grandMA3 onPC Terminal App

The terminal app can be used to connect to the same station or any other grandMA3 station inside your network with the command line or the system monitor of the main app using the Terminal.

- To monitor the software of a connected network station, start the terminal app.
- To connect to the system monitor of a station in the network, enter:

sysmon [IP Address]

Example:

1. Open the grandMA3 Terminal app on the computer.
2. Enter `sysmon 127.0.0.1` in the command section in the grandMA3 Terminal app. Press **Enter**.

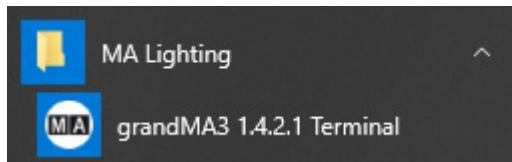
```
MA grandMA3 1.8.1.0 Terminal
TerminalApplication has created components
TerminalApplication is starting components
This application is using realtime threads. Raising process priority.
TerminalApplication has started components
Terminal started. type 'exit' to quit
Unconnected>sysmon 127.0.0.1_
```

grandMA3 Terminal app with sysmon command

3. The network is now connected to the personal system monitor.

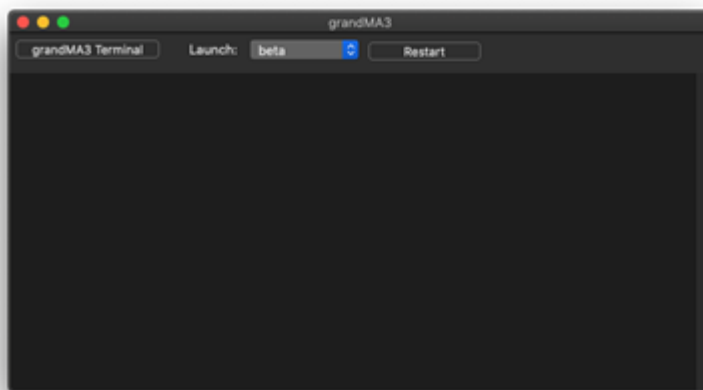
grandMA3 Terminal App on Windows

In Windows®, the grandMA3 Terminal app can be found in the start menu in the folder MA Lighting:



grandMA3 Terminal App on macOS

In macOS®, click `grandMA3 Terminal` to start the grandMA3 Terminal:




5.7. onPC Settings

onPC settings enable different settings for grandMA3 onPC. For example setting the Audio In Device or MIDI control.

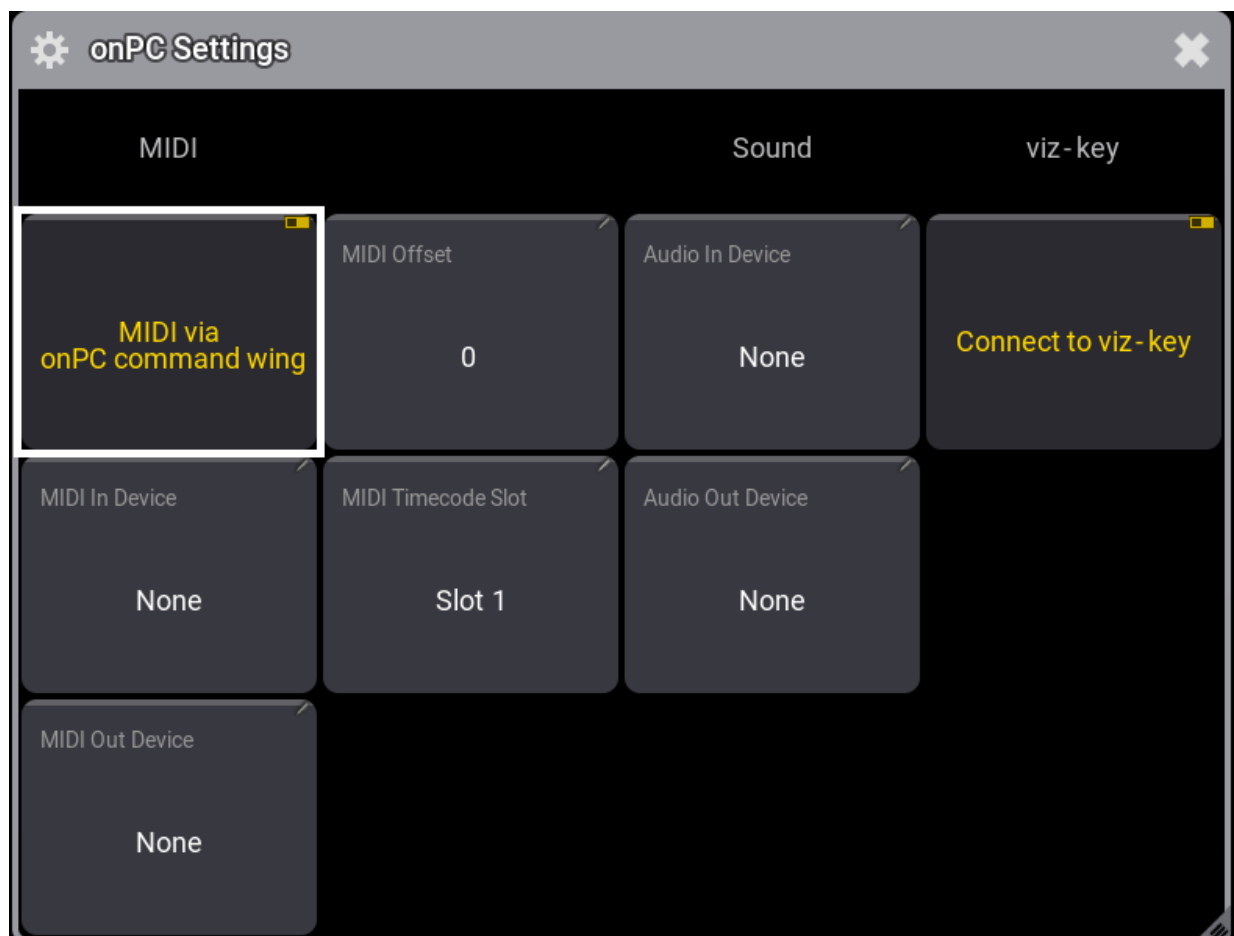
- To learn more about MIDI, read the topics [Connect MIDI](#) in [First Steps](#), and [MIDI Remotes](#) in [Remote In and Out](#).

Open onPC Settings

To open the onPC Settings:

1. Click  on the control bar. The Menu opens.
2. Click [Settings](#). The Setting pop-up opens.
3. Click [onPC Settings](#).

The onPC Settings menu opens.



onPC Settings menu

MIDI



- To enable or disable MIDI from grandMA3 onPC command wing, click **MIDI via onPC command wing** .
If disabled, the device configured in MIDI In Device and MIDI Out Device is used.
If enabled, the command wing MIDI and the device configured in MIDI In Device and MIDI Out Device are used.
- To select a MIDI In device, click **MIDI In Device** .
- To select a MIDI Out device, click **MIDI Out Device** .
- To adjust the MIDI Offset, click **MIDI Offset** .
- To adjust the MIDI Timecode Slot, click **MIDI Timecode Slot** .

Sound

- To select the Audio In device, click **Audio In Device** .
- To select the Audio Out Device, click **Audio Out Device** .

viz-key

- To attach the viz-key dongle to the onPC, enable **Connect to viz-key** .
- To attach the viz-key dongle to a visualizer that is running on the same computer as the onPC, disable **Connect to viz-key** .

6. Show File Handling

The show file contains all the information that is related to the show including:

- Patch
- Fixture profiles
- Cues
- Timings
- 3D information
- Users and user profiles



Important:

Save the old show before loading or renaming a new one, to keep changes.

The software version of a show file can only be moved forward. A show saved on a USB stick that was programmed in a previous version can be loaded in a newer version. If the show is saved again on the USB stick in this new version, it cannot be taken back to the old software.



Important:

To keep the old software version, save the show with a new name.



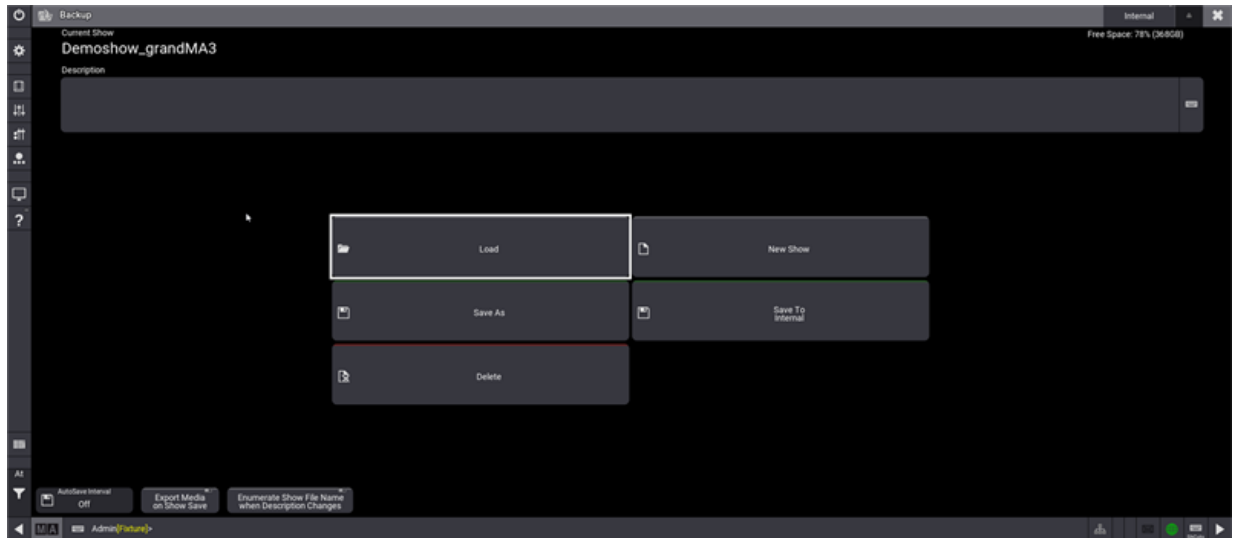
Hint:

To learn more about the folder structure of Shows, Demo Shows, Backup Shows, and Template Shows, read the topics [File Management](#) and [Folder Structure](#).

All the management of the show files is handled in the Backup menu.

1. To access the Backup menu, press **Menu**.
2. Tap **Backup**.

The Backup menu is open:



Backup menu.

Or use the **Menu command**, to call the Backup menu.



Important:

A loaded show file is limited to use 10 GB of memory.



Important:

The size of all media pools in total, is limited to a maximum of 200 MB.



Important:

The show file name is limited to a maximum of 31 characters.



Important:

The show file name must not contain the following characters: \ " \$ * ? ^ | / : < > `

To learn more about the current occupation of memory, read the **System Information** topic.

Select the Drive

The show files are stored on the internal drive or an USB stick. The internal drive is Drive 1 and the first USB stick is Drive 2. If more than one USB stick is connected the order of the connection selects the drive number. That means that the latest connected drive has the highest number.



Hint:

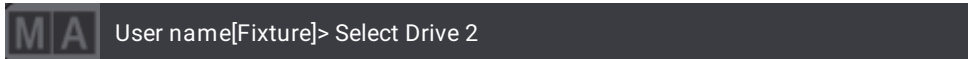
The order of connecting the USB drives to the console / onPC station determines the order of the drive numbers.

In the upper right corner of the Backup menu, it is possible to select the drive you are working on.

To select the drive:

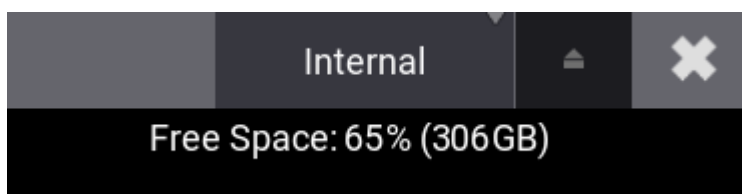
1. Tap **Internal** to toggle between the different drives connected, saved shows or saved shows in older software versions.
2. Tap and swipe right on the drive and a pop-up opens with the different opportunities.

The drive can also be selected via the command line, for example, the first USB drive:



Free Disk Space

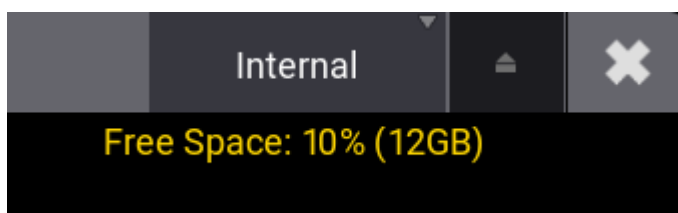
The Backup menu displays the free disk space of the selected drive in the upper right corner of the menu:



Showing free disk space in percent and gigabyte

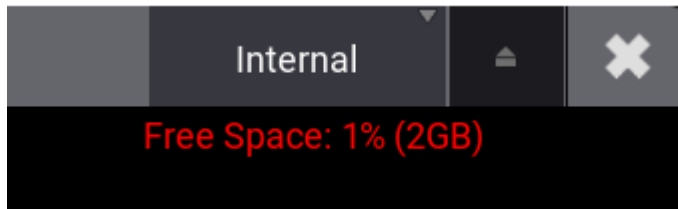
When free disk space is low:

- An **Orange** pulsating text indicates that the free disk space is less than 15 GB.



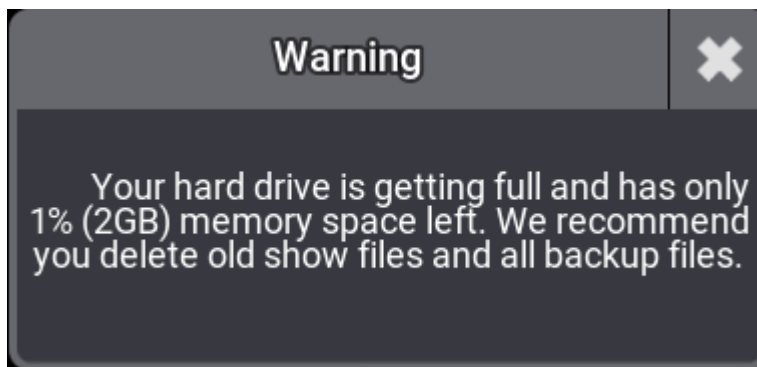
Orange indicator

- A **Red** pulsating text indicates that the free disk space is less than 5 GB.



Red indicator

- The following warning pop-up appears, when booting the software and the free disk space is less than 15 GB.



Warning pop-up.

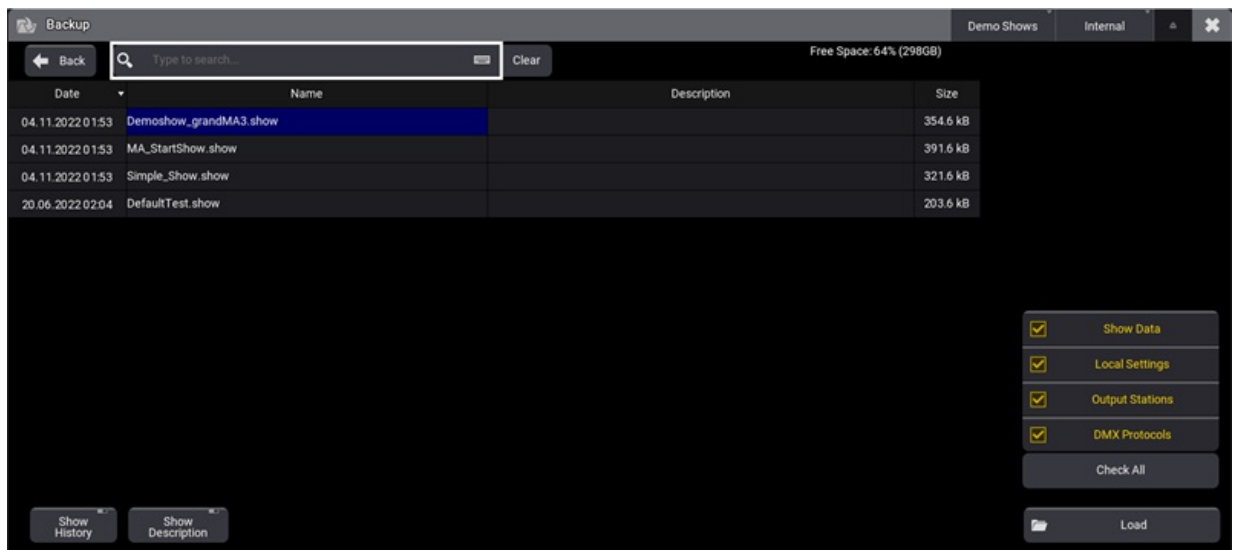
Delete a Show File

Open the Backup Menu and tap **Delete**. Tap the show file to delete. Tap **Delete** in the bottom right corner and the show file is deleted.

6.1. Load a Show File

To load a show file:

1. Open the Backup menu and tap **Load**.
2. Select a show file and tap **Load**.



Load overlay in the Backup menu with the four checkbox buttons on the right side.

To load a show via the command line, use the **LoadShow** keyword.

Load Show Segments

1. To define which segments of a show file are loaded when loading a show file, use the four checkbox buttons on the right side:

- **Show Data**: Loads the show-relevant data from the show file. This includes the patch, pool objects, user, and user profiles.
- **Local Settings**: Loads the local settings. For example, this includes web remote settings and onPC settings.
- **Output Stations**: Includes the setup of all DMX ports, the SMPTE settings, and the MIDI settings.
- **DMX Protocols**: Loads the configuration of all available DMX protocols (Art-Net and sACN).

2. Selected data will be loaded after tapping **Load**.

- **Check All** is a fast selection for all checkboxes.
- The data of the currently loaded show file will be kept for the segments, where the checkbox buttons are not enabled.



Important:

The show file needs to be saved first with grandMA3 v1.9 or higher. Show files that were saved with grandMA3 v1.8.8.2 or prior are not having the Local Settings, Output Configuration, and DMX Protocols saved inside.

6.2. Save a Show File

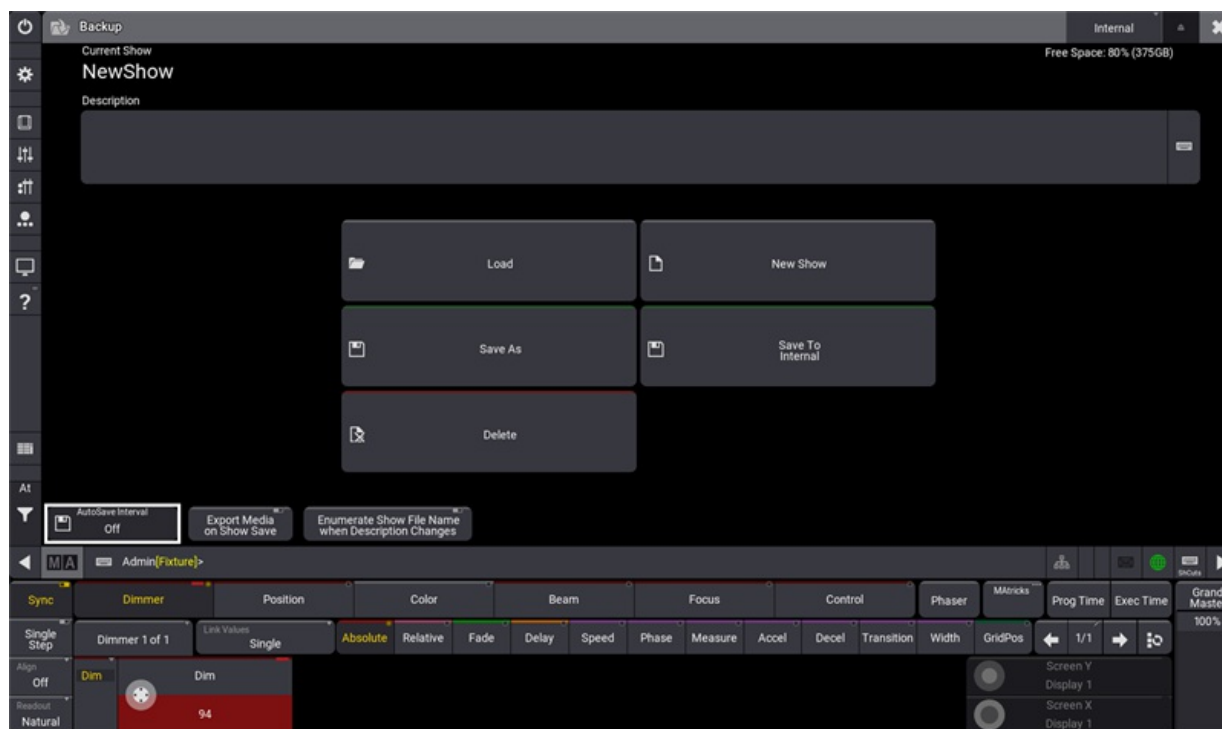
Quick Save

To save the show file quickly:

- Press **Menu** and tap **Quick Save**.
- Pressing quickly 2 x **Menu** also makes a Quick Save.

Auto Save

To save the show automatically in regular intervals, activate the Auto Save function. Tap **Backup** and **AutoSave Interval**.



Auto Save Interval

Select the desired Auto Save Interval.



Select Auto Save

Save Show As

To save a show with another name, open the Backup Menu and tap **Save As**. Write the new name of the show file in the Name field. Tap **Save as ...** and confirm pop-up window with **OK**. The show file is saved by its new name.

To save a show by the command line, use the **SaveShow keyword**.

Save To Internal

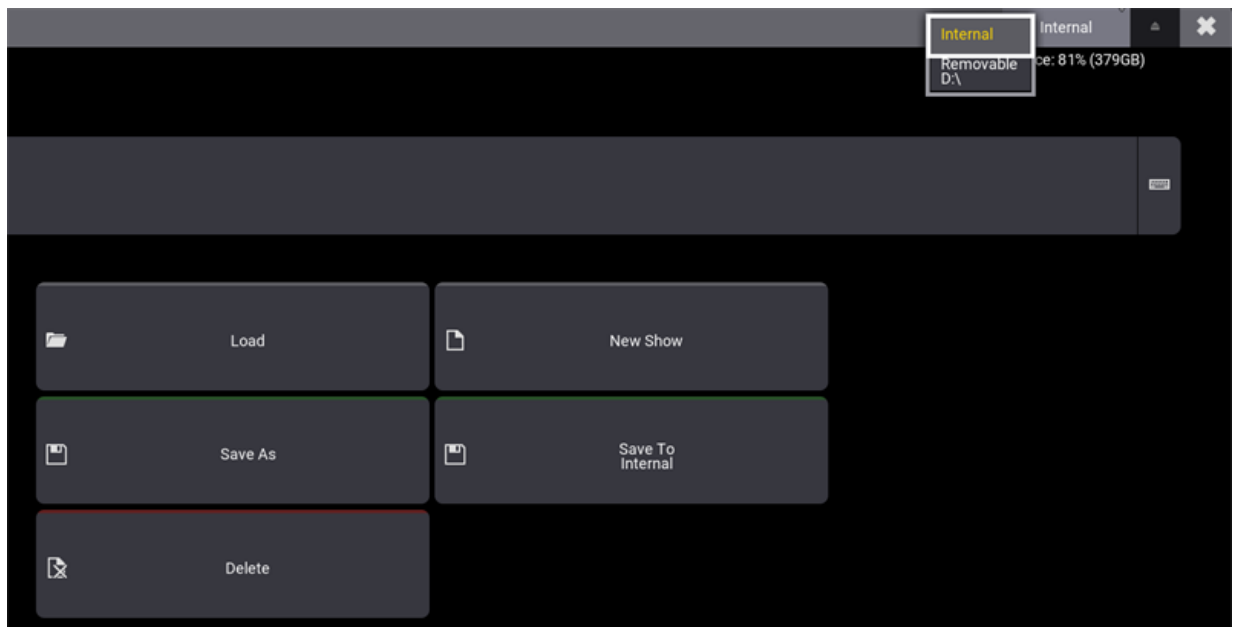


Hint:

Saving a show file to an external drive automatically saves a show file to the internal drive.

To select and change the internal/external drive to save a show file on:

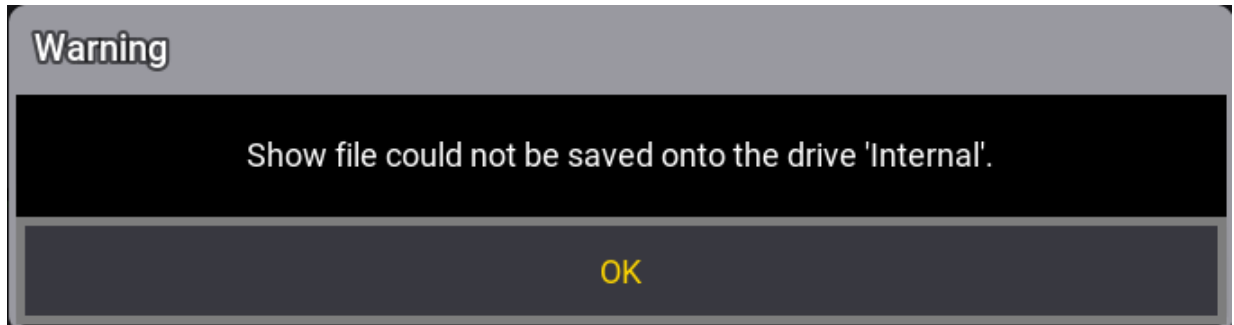
1. Tap **Internal** in the top right corner in the Backup Menu.
2. The name of **Save To Internal** button will automatically adjust to the name of the drive.
3. Tap **Save To <Name of Drive>** to save the show on the selected drive.



Save a show on an internal selected drive

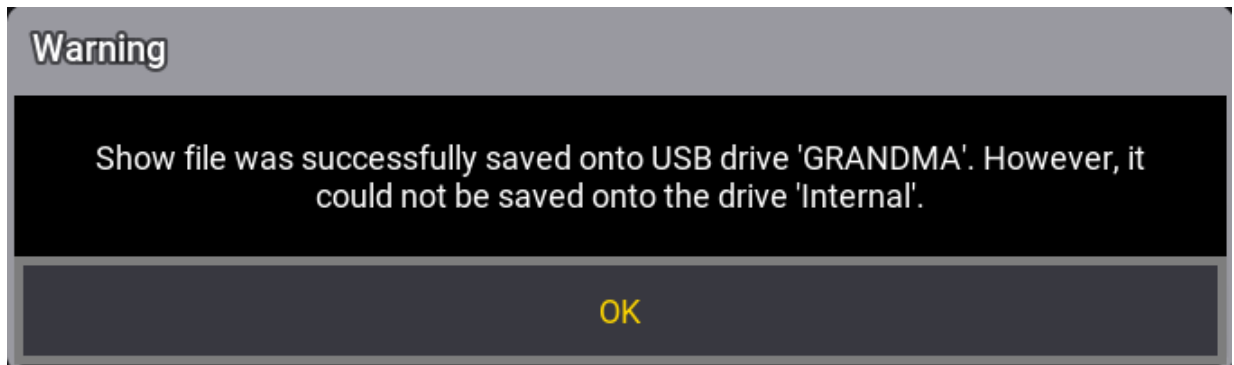
Full Disk Warning

If the show file cannot be saved due to a full disk, a warning pop-up will appear.



Warning pop-up for an 'Internal' save failure

To indicate that the show file is partially saved, a pop-up appears.

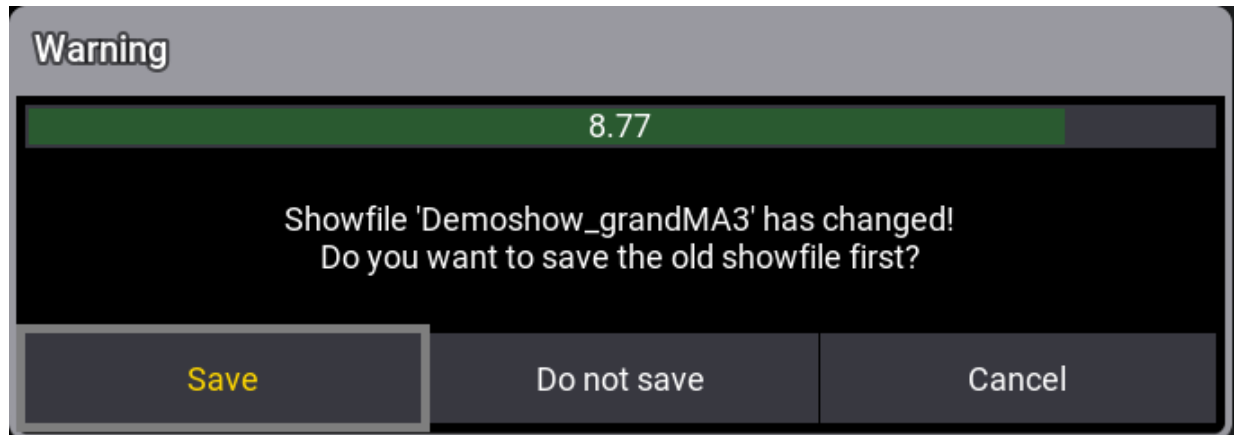


Warning pop-up for a partially saved show file

For more information about Free Disk Space, see [Show File Handling](#).

Shutting Down the System

If an attempt is made to shut down before saving the show file, a Warning pop-up appears:



Save Show File pop-up

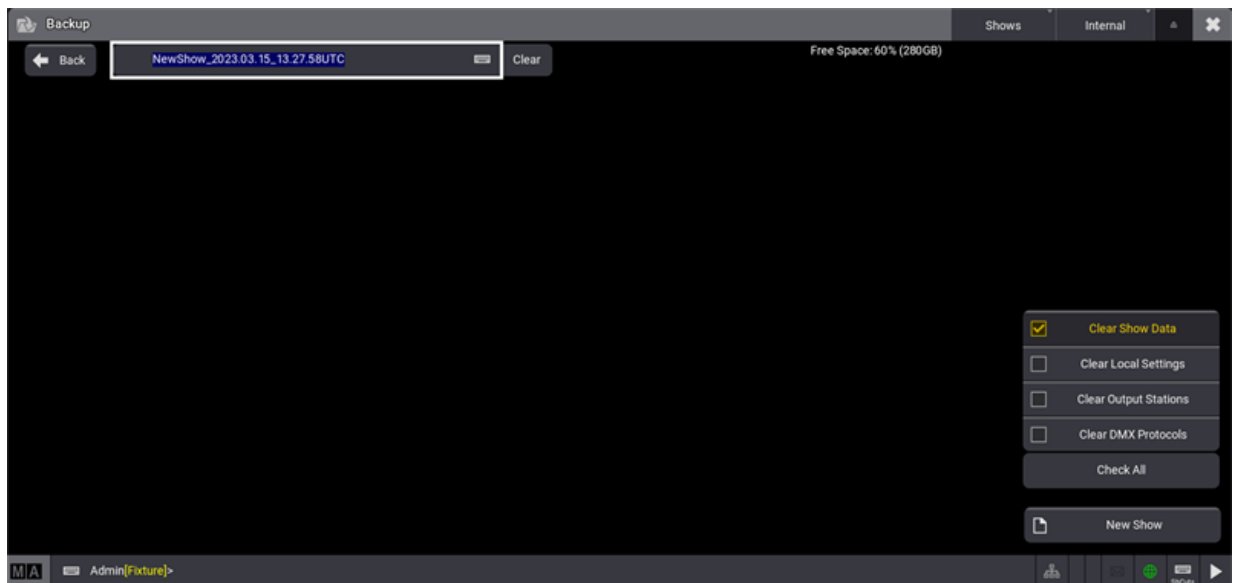
- **Save:** Saves the show file and then shuts down.
- **Do not save:** Shuts down without saving the show file.
- **Cancel:** Cancels the process.

For more information about shut down, see [Shut Down the System](#).

6.3. New Show File

To create a new show file:

1. Open the Backup menu and tap **New Show**. The New Show overlay opens:



New Show overlay in the Backup menu with automatically generated show file name

2. Type to rename the file or keep the automatically generated show file name.
3. Tap **New Show** in the bottom right corner.
4. When the currently loaded show file has changes, a Warning pop-up appears. To continue, choose an option.
When the currently loaded show file does not have any changes, the Create New Show pop-up appears. Tap **New Show**.



Hint:

It is possible to use local letters in the filename, like æ, ö, and ä.



Hint:

To create a new show by the command line, use the **NewShow keyword**.



Clear Show Segments

1. To define which segments of the current show file should be cleared for the new show, use the four checkbox buttons on the right side:

- **Clear Show Data**: Clears the show-relevant data from the show file. Includes the patch, pool objects, user, and user profiles. This checkbox is marked as default.
- **Clear Local Settings**: Clears the local settings. For example, this includes web remote settings and onPC settings.
- **Clear Output Stations**: Clears the setup of all DMX ports, the SMPTE settings, and the MIDI settings.
- **Clear DMX Protocols**: Clears the configuration of all available DMX protocols (Art-Net and sACN).

2. Selected data will be cleared after tapping **New Show**.

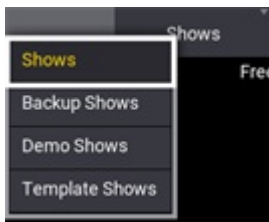
- **Check All** is a fast selection for all checkboxes.
- The data of the currently loaded show file will be kept for the segments, where the checkbox buttons are not enabled.

6.4. Backup, Demo and Template Show Files

Backup Shows and Demo Shows

To load a backup file or a demo show file:

1. Open the Backup Menu and tap **Load**.
2. Tap **Shows** to toggle between **Shows**, **Backup Shows**, **Demo Shows** and **Template Shows**.

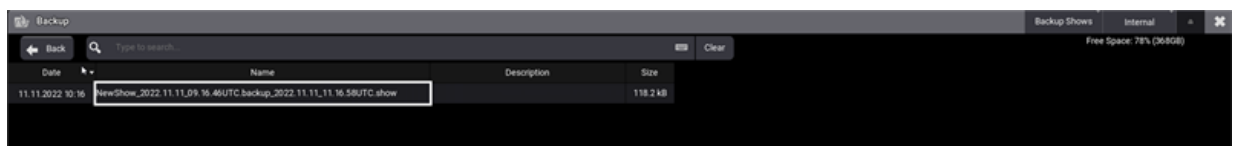


It is possible to load a previously saved backup or a demo show. It is not possible to save or create a new show file in the Backup Shows or Demo Shows folder.

It is possible to store 10 backup files from the same show file. A backup is automatically generated when saving a show file.

A backup file contains:

- Showfile Name
- Backup
- Date and Time in UTC format



Backup Files

Template Shows

The show files in the Template Shows folder can be used as a template.

Show files can be placed into MALightingTechnology/gma3_library/templateshows. For more information, see **Folder Structure**.



Important:

It is not possible to save or create a new show file in the Template Shows folder.



6.5. Organize Show Files

Show File History

To show the show file history:

1. Tap **Load** in the backup menu. Enable **Show History** in the bottom left corner.
2. Select a show file. The history of the selected show file is displayed at the bottom of the backup menu.

When loading the show file, the history gives information regarding:

- Dates, time and timezone
- Used software version
- Software type
- Name

A new entry within the history will be entered, when the show file will be saved again and one of these criteria has changed:

- Version
- Type
- Name
- Time zone
- Description

The newest entry is on top of the history list.

Date	Version	Type	Name	Description	TZ	UTC
20.02.2023 15:06	1.8.8.2	Release	Demoshow_grandMA3		+ 2	20.02.2023 13:06
03.11.2022 15:43	1.8.8.2	Release	Demoshow_grandMA3		+ 1	03.11.2022 14:43

Buttons: Show History, Show Description

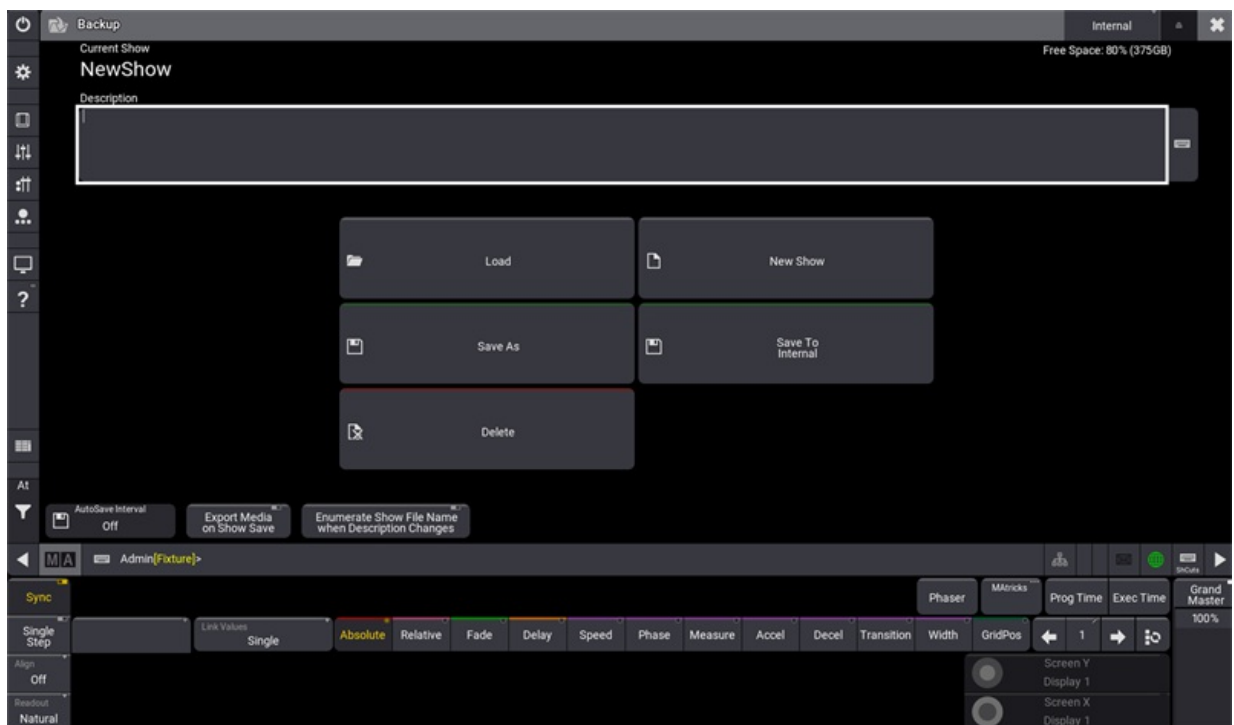
MA Admin[Fixture]>

Show file name with description and time stamp

Add a Description

To add a description to the show file, use the description area at the top of the backup menu. To add another line, press **Enter**.

Modifying or entering a description will only be applied to the show file when saving it afterwards.



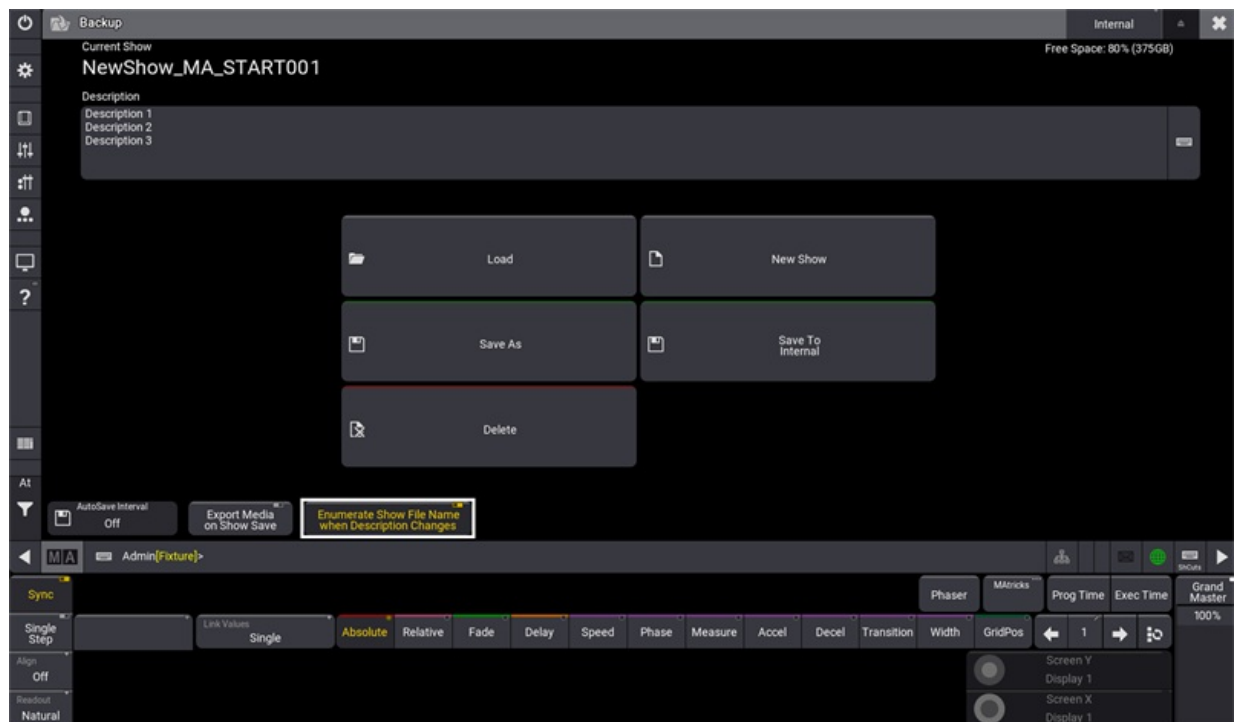
Show file name with description line

Add Automatic Numbering

It is possible to add 3-digit numbers to the show file name.

In order to separate the show file name from the number, the user can add a hyphen (-) or an underscore (_) or any other character to separate the name from the numbers at the end of the show file name.

The setting **Enumerate Show File Name when Description Changes** in the **Backup** menu provides an automatic increase of a 3-digit number. The number will be added to the end of the show file name if the description is changed when saving the show.



Show File name with automatic numbering

If the original show file name is too long to add the number, the command line will return an error. In this case, the former show file name will be used again.

Enumerating a show file can also be called with the `/Enumerate` option.

For more information see [/Enumerate option keyword](#).



Load a Show File During Startup

It is possible to specify a show file or a plugin that shall be loaded upon starting the onPC application.

To do so, add the SHOW or RUNPLUGIN parameter to the shortcut that launches the onPC application:

- "C:\Program Files\MALightingTechnology\gma3_x.x.x\bin\app_system.exe" HOSTTYPE=onPC
SHOW="startshow.show"
- "C:\Program Files\MALightingTechnology\gma3_x.x.x\bin\app_system.exe" HOSTTYPE=onPC
RUNPLUGIN="\startplugin.xml\"-y
 - x represents the grandMA3 version number.
 - y represents the number of the Lua component inside the plugin, that shall be started, e.g., 1.
 - It is not recommend to specify a path for the plugin although the plugin may be put into a subfolder.

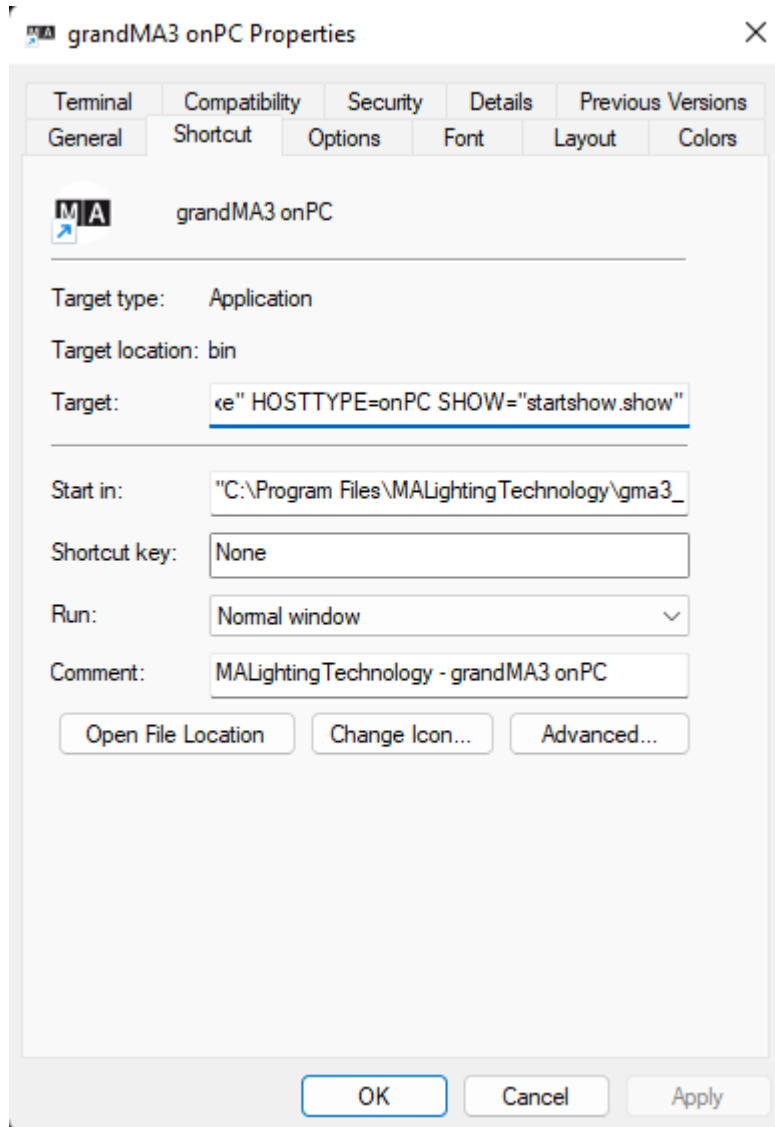
In addition, the optional parameters NOLOAD and CLEANSTART are also available when starting onPC.

- NOLOAD will not load the last show file but start with an empty show. Other device related configurations are kept.
- CLEANSTART will reset device related configuration back to default and come up with an empty show file.



Hint:

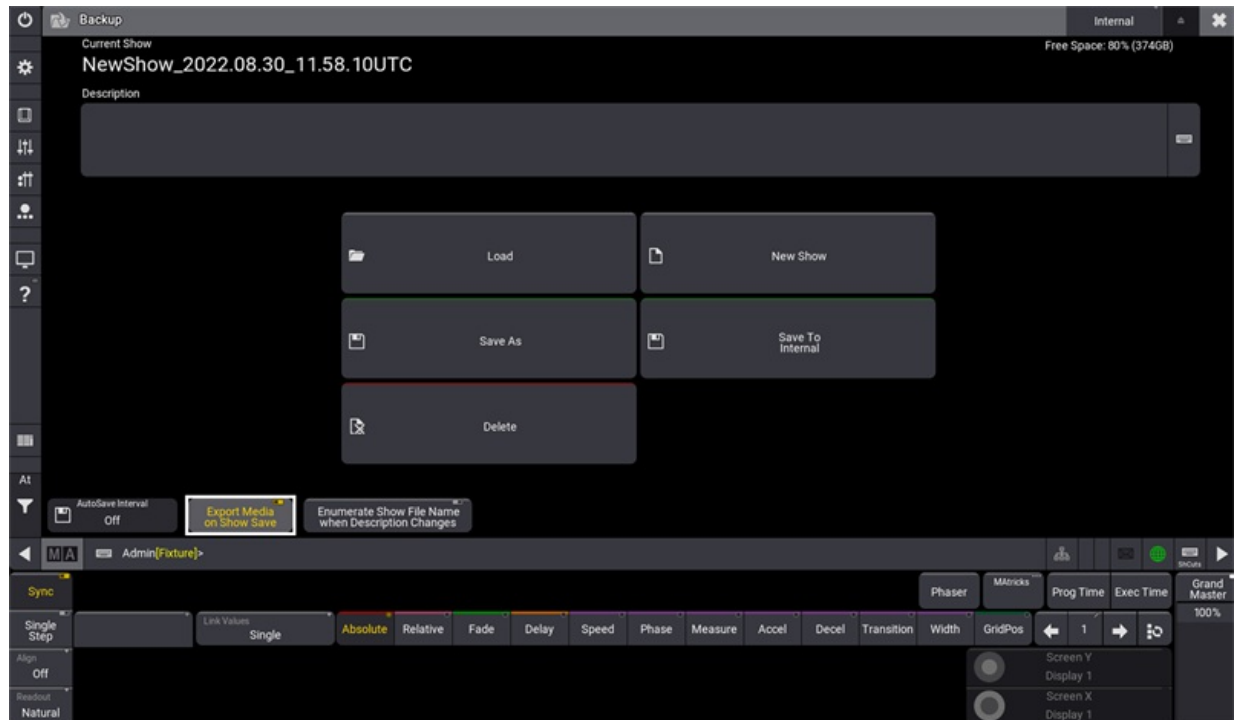
Do not use spaces in a show file name when loading a show during the startup process.



Windows 11 properties are shown as an example for the load show process.

Export Media on Show Save

In order to save media files (e.g. images, videos) to the **gma3_library folder**, activate the Export Media on Show Save function. Tap **Backup** and **Export Media on Show Save**.



A show file with Export Media on Show Save

grandMA2 to grandMA3 Show File Converter

To learn more about how to convert show files from grandMA2 to grandMA3, see **[grandMA2 to grandMA3 Show File Converter](#)**.



7. Workspace

In general, **Workspace** deals with the visual elements on screens and input modes such as:

- Windows
- Views
- Coloring
- Executor bar
- Encoder bar
- Gestures
- Keyboard shortcuts

It represents the usage of the working area in the grandMA3 console.



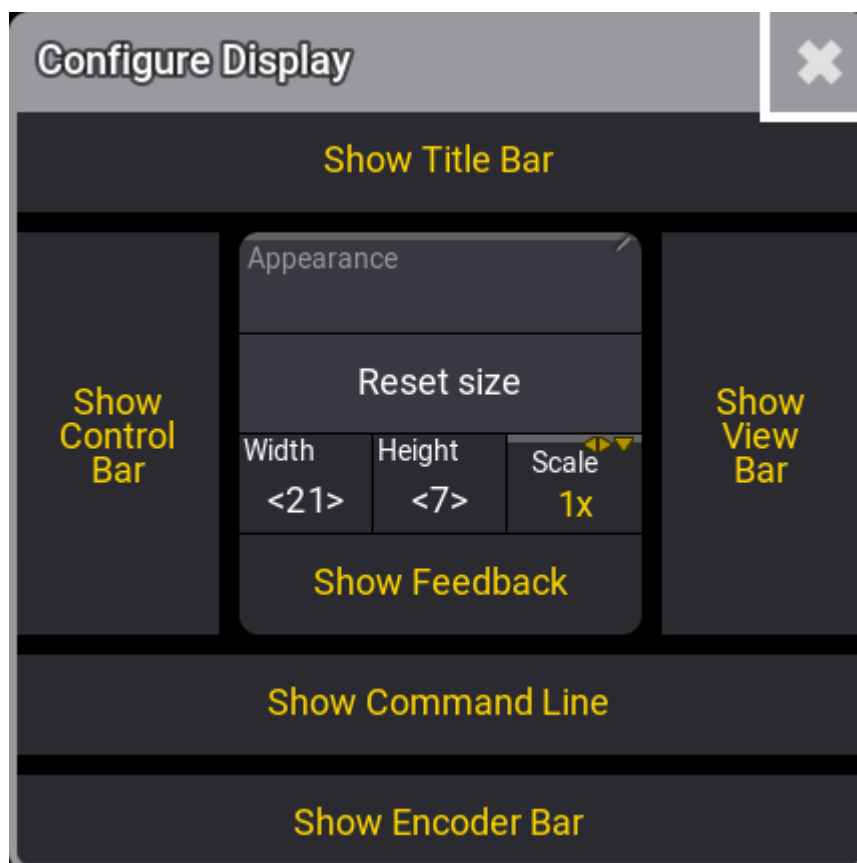
7.1. User Interface

This section covers the control elements that help you operate and control the grandMA3 in an efficient and effective way.

7.1.1. Configuration of Displays

The displays are populated with a combination of a user-defined area and a collection of additional display elements. The additional elements offer quick access to helpful features, and their availability differs from screen to screen.

Pressing **Menu** then tapping **Configure Display** on any screen (excluding screen 8, screen 9, screen 10, screen 11, and screen 12) opens the Configure Display dialog for that screen. This dialog can also be opened by tapping any user-defined area on the desired screen and tapping **Configure** in the upper-right corner of the Add Window dialog.



Tap areas to activate or deactivate them

This pop-up is used to configure or change the appearance of the display. It has different areas:

- **Show Title Bar**
Activate this button to see the window frame in the grandMA3 onPC. This is not an option in the MA hardware.
- **Show Control Bar**
Activate this to see the control bar on the left side of the display.



- **Appearance**

Tap this to open the Select Appearance pop-up and select one of the existing appearances for this display. The appearance is added as a background for the screen area. Read more in the [Appearance section](#).

- **Reset size**

Tap this to reset the size of the user-defined area. The size can be changed using the **Width** and **Height**.
Allowing each user to set the size of the user-defined area in the display. A user-defined area bigger than the display will appear within a brown frame. A user-defined area that is smaller than the screen will gray out the unused area.

- **Scale**

The scale can be used to change the scale of the interface. This option is only available on onPC and on external screens. It is very useful with high DPI screens or screens that have a lower resolution than Full HD.

- **Show Feedback**

Tap to show or hide a small feedback overlay at the bottom of the display. This feedback appears briefly when the show is saved.

- **Show View Bar**


This option hides or displays the column of view buttons on the right side of the display.

- **Show Command Line**

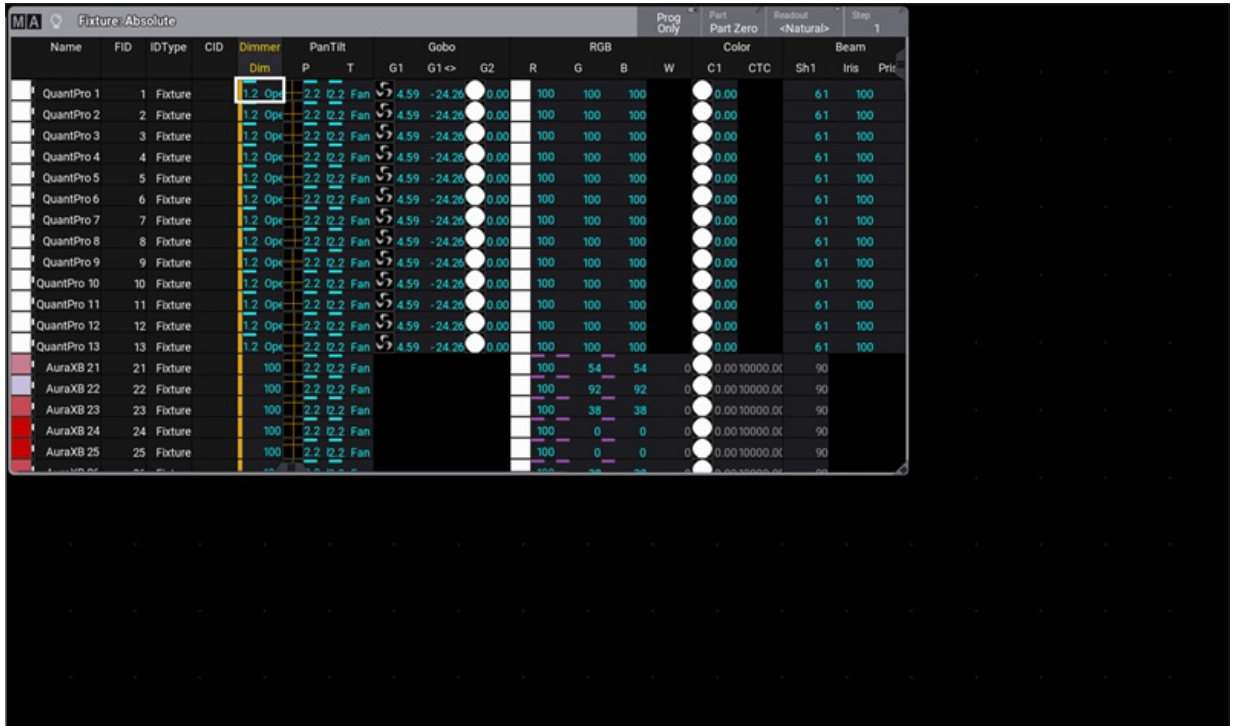
Activate this to see the command line input bar at the bottom of the display.

- **Show Encoder Bar or Show Playback Bar**

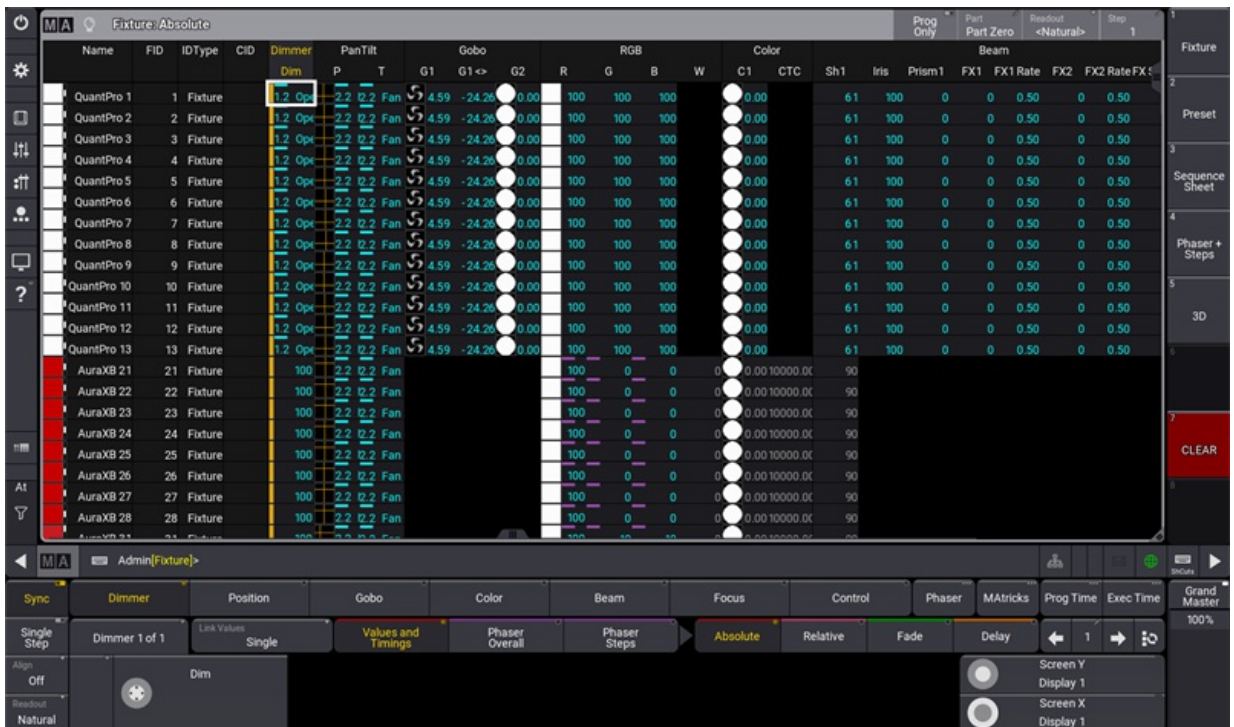
This hides or shows the encoder bar or the playback bar at the bottom of the display.

The  in the title bar closes the pop-up.

These additional elements can be hidden to increase the amount of available user-defined area or shown in different combinations on most screens, whether internal screens or external monitors.



Screen 1 with additional display elements hidden, allowing more user-defined area.



Screen 1 with all additional display elements shown.



Command line with additional feedback overlay.

Screen 1, Screen 2, and Screen 3

Screen 1, screen 2, and screen 3 are the main screen workspaces on the console. The displays on these screens offer the most available user-defined area on any of the internal screens. They can be configured with a selection of additional display elements.

By default, screen 1 includes the command line and view bar. It can also be configured to display the control bar and encoder bar.

By default, screen 2 and screen 3 include the view bar. They can also be configured to display the command line, control bar, and playback bar.

Screen 4 and Screen 5

Screen 4 and screen 5 are on the optional external monitors. The displays on these screens offer a large amount of user-defined area. They can be configured with a selection of additional display elements. They can be configured to display the view bar, command line, control bar, and playback bar.

The scale option is available on screen 4 and screen 5 to take advantage of monitors of different resolutions.

Screen 6 and Screen 7

Screen 6 and screen 7 are the right command screen and left command screen, respectively. The displays on these screens offer a small amount of user-defined area. They can be configured with a selection of additional display elements.

Screen 6 can be configured to display the view bar, command line, and control bar.

Screen 7 can be configured to display the view bar, command line, control bar, and playback bar. The playback bar available on screen 7 shows the status of objects assigned to the Xkeys.



Screen 7 with view bar and playback bar visible, showing the assignments of the Xkeys.

Screen 8, Screen 9, Screen 10, Screen 11, and Screen 12

Screen 8, Screen 9, Screen 10, Screen 11, and Screen 12 are the letterbox screens, which are available on the grandMA3 full-size, light, and extension. The displays on these screens offer no user-defined area. The configurations of these displays cannot be changed. These screens are dedicated to specific display information and user input.



Screen 8 displays the encoder bar and grand master.

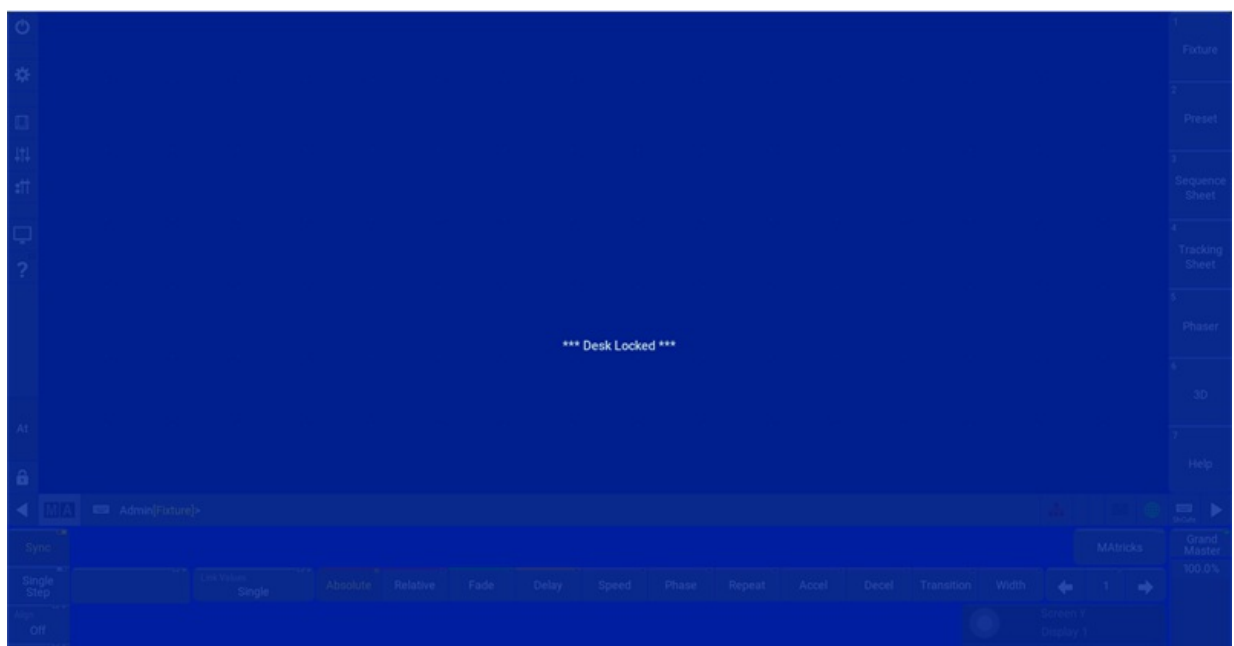
Screen 9 displays the playback bar and master section.

Screen 10, Screen 11, and Screen 12 display the playback bar and custom section encoders and wheels.

7.1.2. Desk Lock

To lock the desk, use one of these options:

- Press **MA** + **MA** + **Pause**.
- Press **Pause** on the keyboard.
- Press **F9** on the keyboard.
- Tap the  icon at the top of the **control bar** to open the **shutdown menu**. Then tap the  icon.

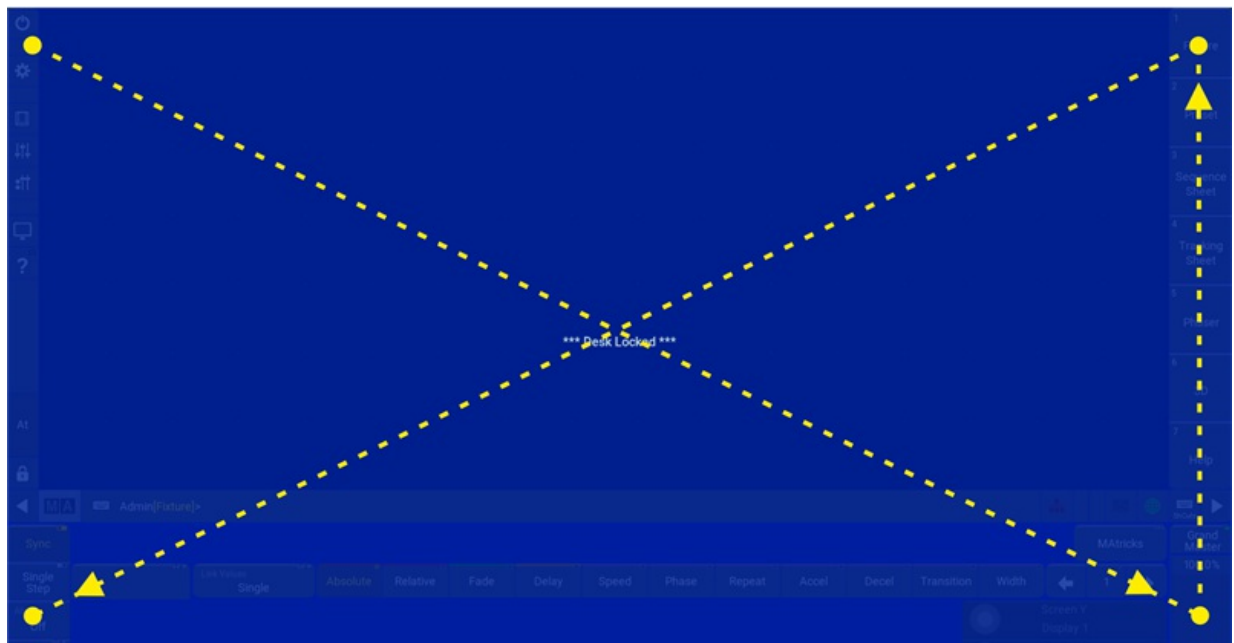


Desk Locked

The desk is locked.

To unlock the desk, use one of these options:

- Press **MA** + **MA** + **Pause**.
- Press **Pause** on the keyboard.
- Press **F9** on the keyboard.
- Tap the upper left corner, lower right corner, upper right corner, and lower left corner of any screen.



Touch pattern to unlock the desk.

The desk is unlocked.

7.1.3. User-Defined Area

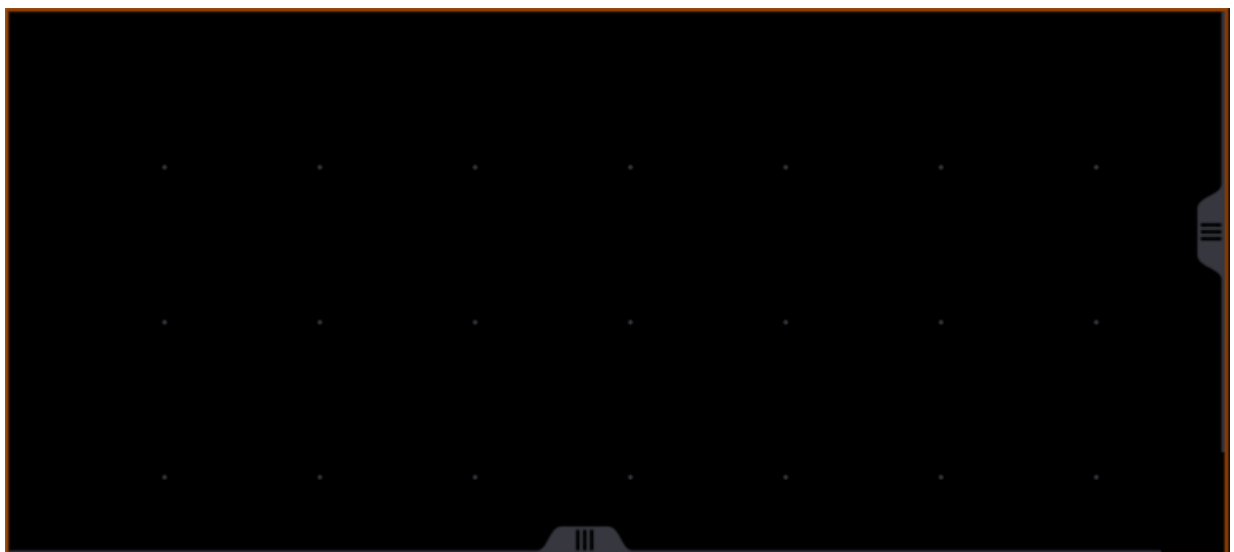
The user-defined area is located on screens 1 through 7. It appears as a grid of small dots.



Empty user-defined area with a grid of small dots

This is the area where windows can be added and arranged to be stored and recalled as views. For information on how to manage windows in the user-defined area and views, see [Windows, Views, and Menus](#).

The Configure Display menu allows adjustment of the grid size within the user-defined area. It is possible to configure the grid of the user-defined area to a size larger or smaller than the available display space. For more information on the Configure Display menu, see [Configuration of Displays](#).



Grid of the user-defined area configured larger than the available display space

If the grid is larger than the available display space, a dark orange outline appears around the edge of the user-

defined area. Horizontal and vertical scroll bars appear, allowing access to the full space of the grid. The 3 finger scroll gesture also scrolls the user-defined area within the visible portion of the display. For more information on gestures, see the [Gestures](#) topic.



Grid of the user-defined area configured smaller than the available display space

If the grid is smaller than the available display space, a partially transparent gray overlay appears over the excess area. Any previously stored views that utilize this area will still recall as stored. The overlay does allow interaction with the contents of windows underneath, but windows cannot be resized or created outside of the user-defined area.



7.1.4. Command Line

The main purpose of the command line is to display active command syntax as it is entered and to display any errors generated by failed syntax. The command line also includes several helpful status indicators and quick-access buttons to frequently used features.

By default, the command line is located at the bottom of screen 1. It can also appear at the bottom of screens 2, 3, 4, 5, 6, and 7.



Command line as it appears on screen 7

Show or Hide the Control Bar

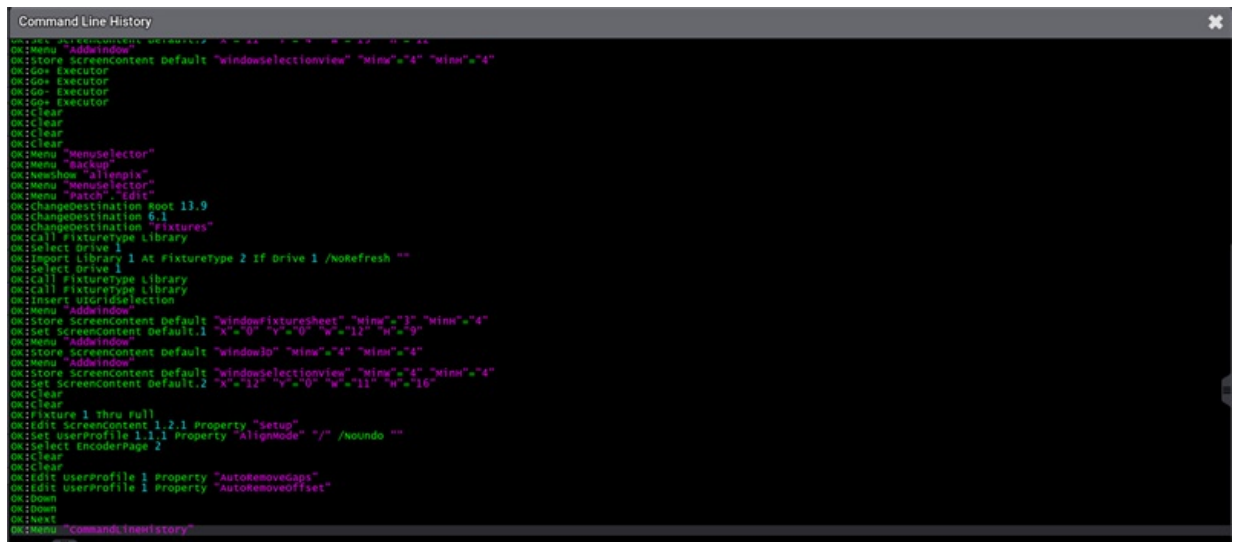
Tap the left arrow at the left edge of the command line to show or hide the control bar along the left edge of the screen.

Open the Command Line History

To open a temporary window showing the Command Line History:

- Tap **MA** on the left side of the command line

The Command Line History opens.




Command Line History

To close the window:

- Tap **MA** once again

-or-

- Tap  in the upper right corner of the window

To open the window version of the Command Line History:

1. Open the Add Window dialog
2. Tap **Common** + **Command Line History**.


To show or hide the Command Line in the Command Line History window:

1. To open the Command Line History Window Settings, tap **MA**.
2. Enable **Show Command Line**.

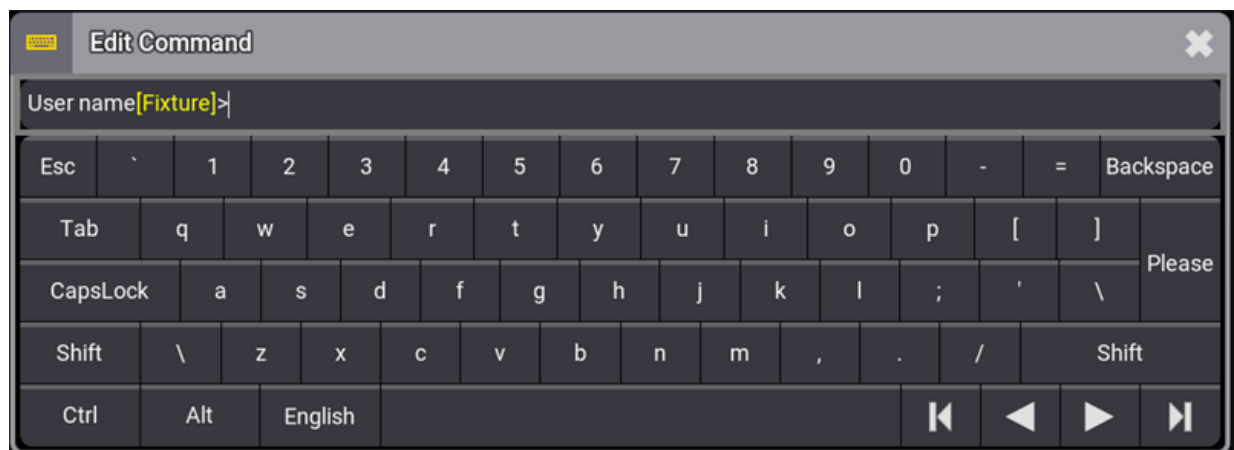
For more information, see [Add Window](#).

Open the Virtual Keyboard

To open a temporary command line window with a virtual keyboard:

- Tap  on the left of the command line

The window opens with a virtual keyboard.




Open virtual keyboard

To close the virtual keyboard:

- Tap  once again

-or-

- Tap  in the upper right corner of the window

Information in the Command Prompt

The command prompt is the point where entered syntax appears. The prompt ends with the '>' symbol, but it contains two pieces of helpful information before the symbol.

- **User Name**

The user name of the logged in user appears first in the command prompt.

- **Default Keyword**

The default keyword appears in square brackets after the user name. The console uses the displayed default keyword for any numerical syntax entered without a specified object keyword.



Important:

It is possible to navigate through the directory structure of the console using the command line. In this case, the current location in the directory replaces the default keyword displayed in the command line. When returning to the root directory, the default keyword replaces the directory path display.

Change the Default Keyword

Possible default keywords of the command line can include:

- **Fixture**
- **Channel**
- **Global**
- **Houselights**
- **NonDim**
- **Media**
- **Fog**
- **Effect**
- **Pyro**

To change the default keyword:

1. Enter the desired keyword into the command line. For example, press the **Channel** key.
2. Press **Please**.

Set the Cursor in the Command Line

When entering command line syntax using a keyboard, it is necessary to bring the keyboard focus to the command line.

To set the cursor in the command line:

- Tap the command line

- or -

- Press **Esc** one or more times. Once all open menus and pop-ups are closed, the keyboard focus returns to the command line.

- or -

- Press and hold **MA** and subsequently press **Please**





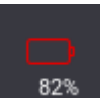
Set cursor in the command line

The cursor is set and starts blinking after the prompt and a white box surrounds the command line.

For more information, see [Command Syntax and Keywords](#).

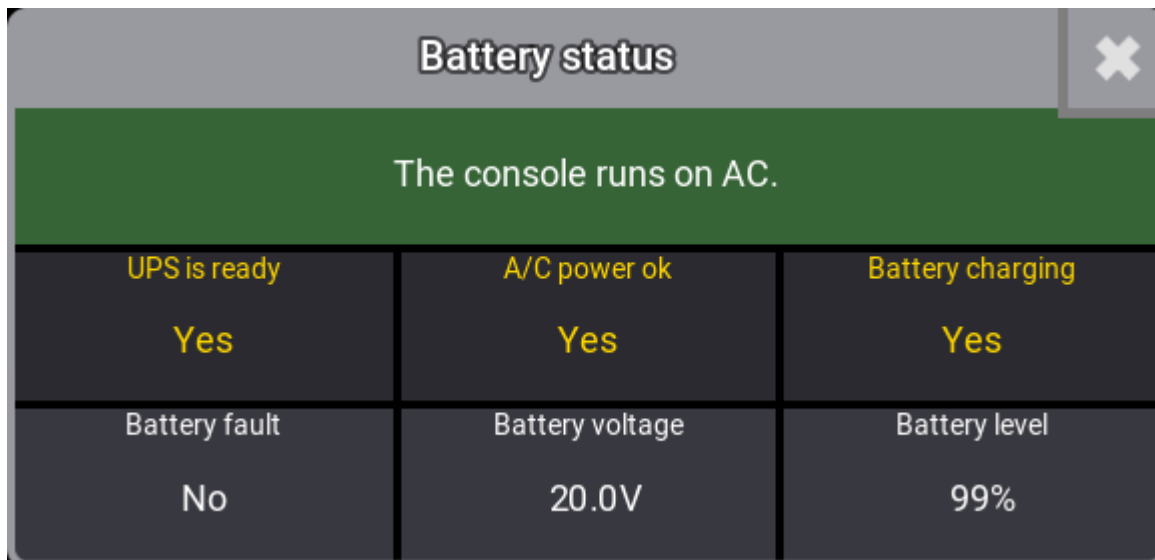
Power Status

On full-size, light, full-size CRV, and light CRV consoles, which are equipped with an internal battery backup, there is an indicator of power status on the right side of the command line. The battery percentage is always displayed at the bottom of the indicator. Status indications include the following states:

-  Power is connected and the battery is full.
-  Power is connected and the battery is charging.
-  Power is disconnected and the battery is discharging.

Battery Status Pop-up

Tap the power status indicator to open the Battery status pop-up.



Battery status pop-up

The Battery status pop-up displays multiple relevant metrics about the battery as well as A/C power.

For more information about the Power Status, see [UPS Battery](#).



Session Status and Quick Access to the Network Menu

An indicator of network session status appears on the right side of the command line. Tapping the indicator toggles the network menu open or closed. The color of the network icon indicates one of the following session states:

- **Red:** Standalone
- **Black:** Startup
- **Gray:** IdleMaster
- **Light Blue:** GlobalMaster
- **Green:** Connected

For more information about networking, see the [Networking](#) topic.

Phaser Values Workload

The left bar between  (network) and  (message center) in the command line shows the current processing workload dedicated to phaser calculation.

- To show the workload of phasers, a green indicator increases with the number of parameters with actively running phasers.

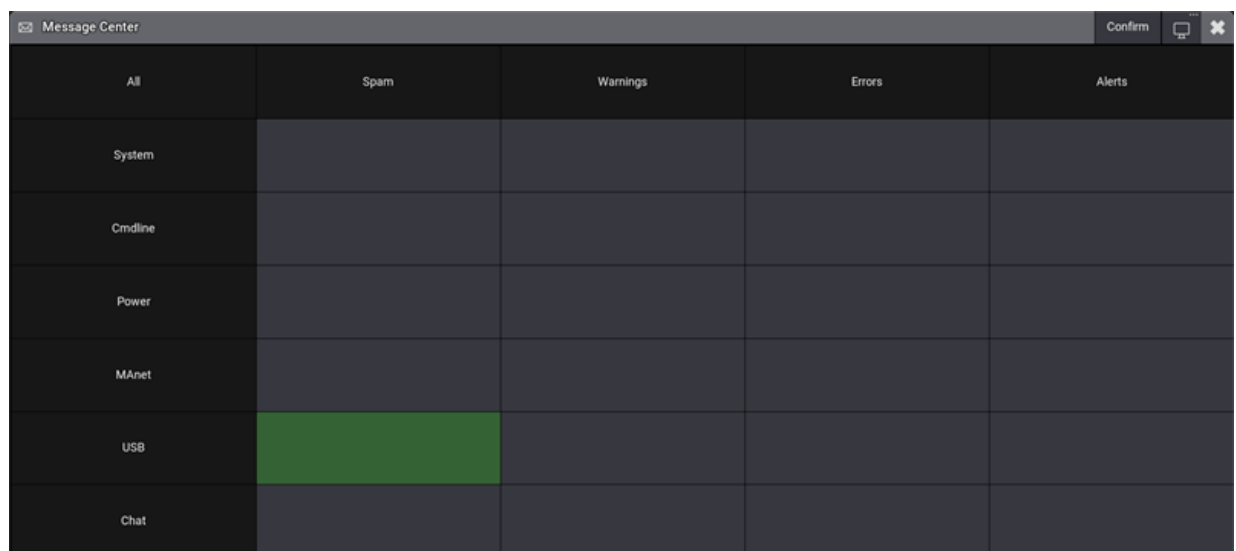
For more information on Phasers, see the [Phasers](#) topic.

FlowControl


The right bar next to the Phaser indicator shows the current workload of the network.

Open the Message Center


The envelope icon on the right side of the command line changes color to indicate new messages in the message center. Tap the icon to open a temporary version of the Message Center window.




Open message center

1. To confirm all messages, tap **Confirm**.
2. To open the message center on another screen, tap ;
3. To close the message center, tap the message center icon in the command line once again

-or-

- Tap  in the upper right corner of the window
-

World Server Indicator

The globe icon  on the right side of the command line changes color from black to green to indicate an active connection to the world server.

For more information on the world server, see the **World Server** topic.



Enable or Disable Keyboard Shortcuts

Tapping the keyboard icon, with the text "ShCuts" beneath it, on the right side of the command line toggles the keyboard shortcuts on or off.

- A white color indicates that keyboard shortcuts are disabled.
- A yellow color indicates that keyboard shortcuts are enabled.
- A red color indicates that keyboard shortcuts are enabled but temporarily blocked by a menu requiring normal keyboard input.

For more information on keyboard shortcuts, see the [Keyboard Shortcuts](#) topic.

Show or Hide the View Bar

Tap the right arrow at the right edge of the command line to show or hide the bar of view buttons along the right edge of the screen.



7.1.5. Control Bar



The control bar allows fast access to the shutdown command, essential menus, additional displays (in grandMA3 onPC), online help, and at menu.








On console displays, the control bar is hidden on all screens by default. Within the onPC application, the control bar is shown along the left edge of screens 1 through 5 by default. In consoles and the onPC application, the control bar can appear along the left edge of screens 1 through 7.

To show or hide the control bar, tap **Show Control Bar** in the configure display menu for the desired screen or press the left arrow at the left side of the command line.



Control bar

- Tap the  icon at the top of the control bar to open the **shutdown menu**.
- Tap the  icon in the control bar to open the main menu. For more information, see **Menus**.

- Tap the  icon in the control bar to open the command section overlay. For more information, see [Command Section](#).
- Tap the  icon in the control bar to open the master controls overlay. For more information, see [Master Controls](#).
- Tap the  icon in the control bar to open the playback controls overlay. For more information, see [Playback Controls](#).
- Tap the  icon in the control bar to open the custom master section overlay. For more information, see [Special Executors](#).
- Tap the  icon in the control bar for access to all of the available displays within the grandMA3 onPC application. This option is only available in grandMA3 onPC. For more information, see [Displays in grandMA3 onPC](#).
- Tap the  icon in the control bar to enter the Help keyword into the command line. For more information, see [Help keyword](#).
- Tap the  icon in the control bar to toggle the main encoder bar to the command wing bar in the grandMA3 onPC application. This option is only available in grandMA3 onPC. For more information, see [Command Wing Bar](#).
- Tap the **At** button in the control bar to open the at overlay. For more information, see below.

At Overlay

The at overlay allows quick access to various programmer functions and special values.

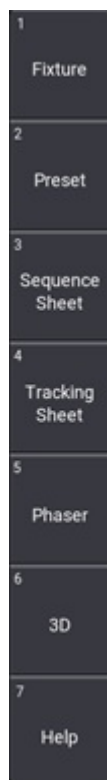


At overlay

- Tap **At Filter** to open the at filter overlay. For more information, see the [At Filter](#) topic.
- Tap **Cut Programmer**, **Copy Programmer**, and **Paste Programmer** to cut, copy, and paste the contents of the programmer. For more information, see the [What is the Programmer](#) topic.
- Tap **Delete Steps** to delete the current phaser step or steps from the programmer. For more information, see the [Phasers](#) topic.
- Tap **Clear** to access the same three clear keywords available by pressing the **Clear** key. For more information, see the [Clear Key](#) topic.
- Tap **Full**, **Zero**, **Default**, **Normal**, **On**, **Off**, **Release**, or **Remove** to apply the desired special attribute value to the selection. On and off apply to the selection. Full, zero, default, release, and remove apply to the dimmer attribute of the selection. For more information, see the Specials section in the [Calculator](#) topic.

7.1.6. View Bar

The view bar displays a number of view buttons. The view bar is visible along the right edge of screens 1 through 7 by default. To show or hide the view bar on a screen, tap **Show View Bar** in the configure display menu for the desired screen, or press the right arrow at the right side of the command line.



View bar displaying seven view buttons

View buttons allow storage and recall of user-defined views. View buttons also allow assignment and recall of macros and plugins. It is possible to:

- Store views to view buttons.
- Assign views, macros, and plugins to view buttons.
- Call views, macros, and plugins on the view buttons.
- Move and copy views, macros, and plugins to different view buttons.
- Delete views, macros, and plugins from view buttons.



Important:

Deleting an object from a view button only deletes the assignment of the view button referencing the object. The previously assigned object still exists in the appropriate view, macro, or plugin pool.

For more information on storing and calling views, see the [Windows and Views – Store and Recall Views](#) topic.



7.1.7. Tables in General

Compared to sheets that serve as an overview, tables is where you can actively enter, change, and modify values.

Tables have parents and children in form of new "object lines".

- A parent is the main object
- A child is the sub object of the parent
- A child can have several children

To disable or enable children in tables, tap **'New object' line** .

Simplify operation and edit in tables:

- **Move/Select**

-To move the focus in a table, tap to select an object and use the arrow buttons of the integrated or virtual keyboard.

- To move the selection from one column to another, tap the heading of the corresponding column.

-Tap the heading twice to select the entire column.

- **Inertial scrolling**

Use two fingers to tap and quickly brush the table in either direction.

Depending on the momentum, the scroll slows down after a while until it comes to a halt.

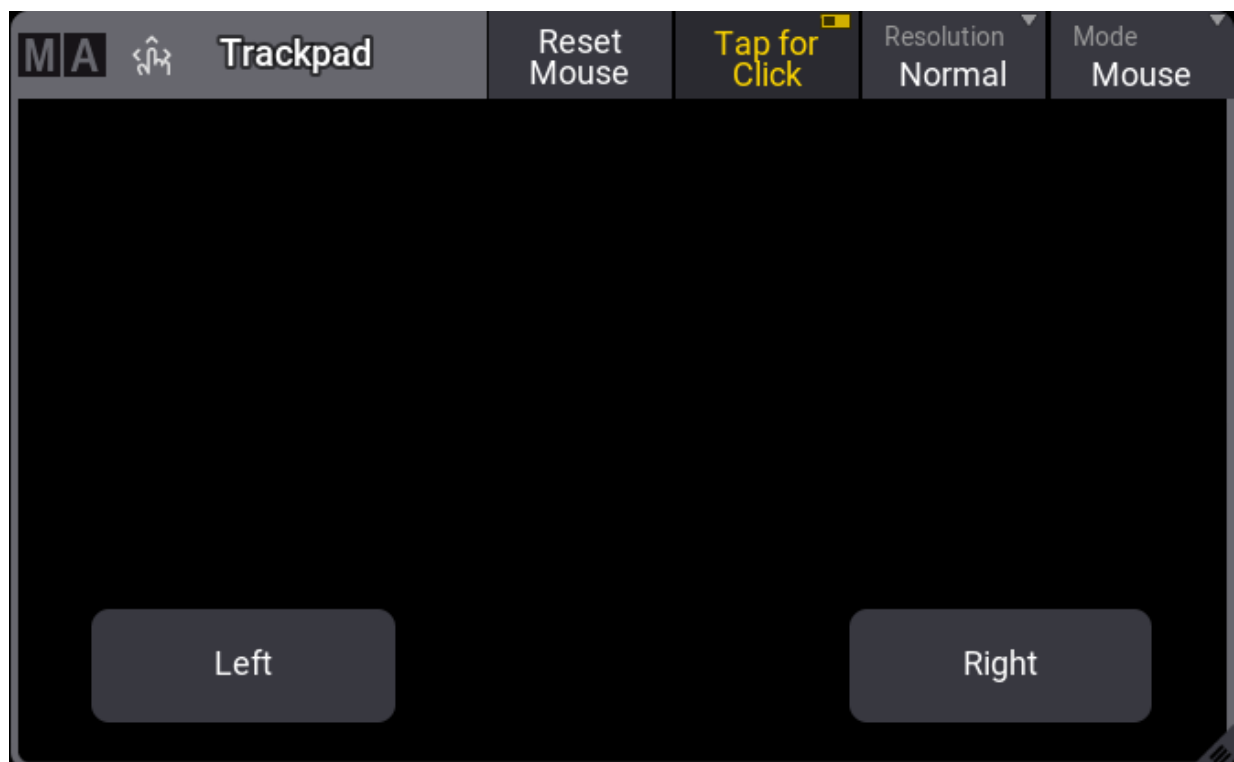
7.1.8. Trackpad Window

The trackpad window allows precise control of a mouse cursor as well as an alternate method for controlling the position of selected fixtures.

The trackpad window can appear within the user-defined area of screens 1 through 7. Views can store and recall the appearance and settings of the trackpad window. For more information on adding windows to the user-defined area, see the [Add windows](#) topic.

Tap the **Mode** button in the upper-right corner of the trackpad window to toggle between mouse mode and pan/tilt mode, or tap and swipe to see a menu of available modes.

Mouse Mode

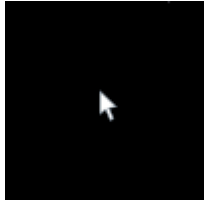


Trackpad window in mouse mode

Trackpad Mouse Cursor

While in mouse mode, tapping inside the main area of the trackpad window and swiping in any direction presents a dedicated cursor and moves that cursor in the direction of the swipe.

The main cursor in the user interface of the console appears as a black arrow with a white outline. This cursor is not affected by the trackpad window. A separate cursor appears when using the trackpad in mouse mode. The trackpad mouse cursor appears as a white arrow with a light gray outline.



Trackpad mouse cursor

The trackpad cursor initially appears slightly larger than normal. After three seconds, it reverts to its normal size. After five seconds of idle time, the trackpad cursor disappears.

The trackpad mouse cursor can move freely across the user interface of the console within screens 1 through 5. Although the trackpad window can appear on screen 6 and screen 7, the trackpad cursor cannot appear on these screens.

Trackpad Window Buttons in Mouse Mode

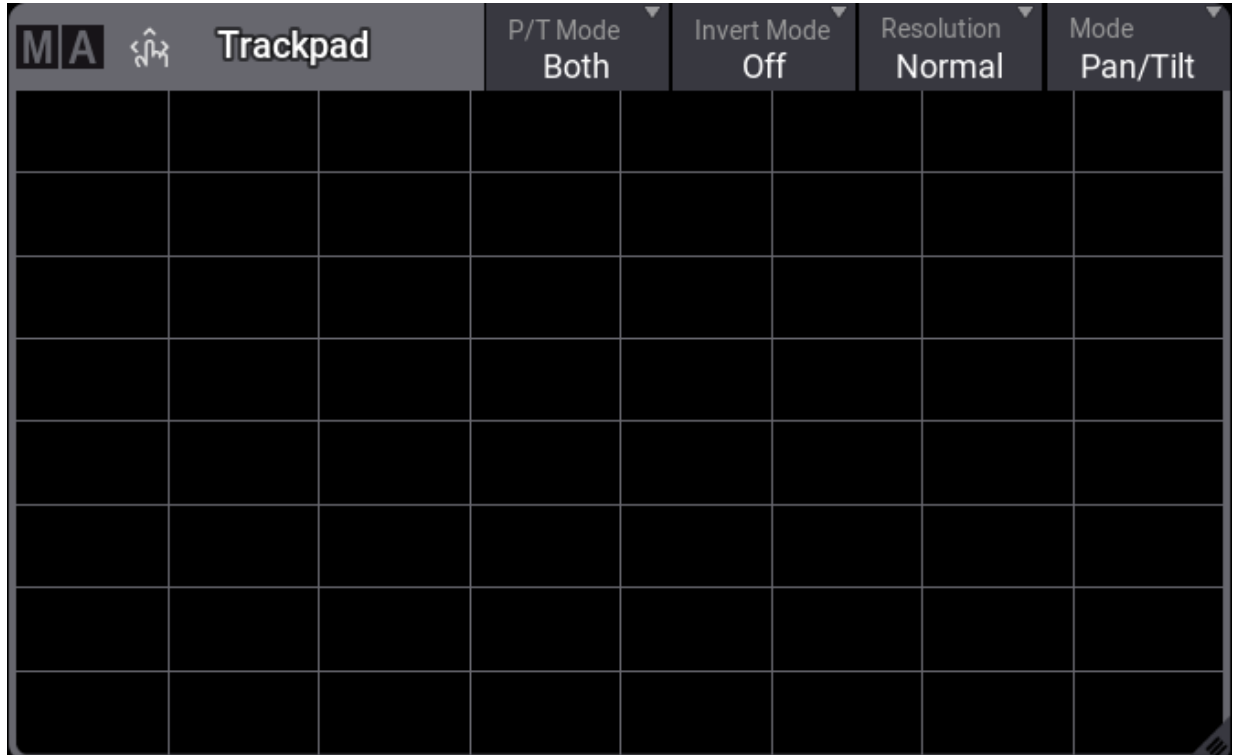
The main area of the trackpad window in mouse mode includes two buttons:

- **Left** : Tap for a left mouse click. This selects the item under the point of the trackpad cursor. Tap and hold while swiping in the main area of the trackpad window with a second finger to mimic a 1 finger swipe gesture.
- **Right** : Tap for a right mouse click. This generates the same result as a 2 finger edit gesture.

The title bar of the trackpad in mouse mode includes tools, which control the behavior of the trackpad. These tools include:

- **Reset Mouse** : Tap to respawn the trackpad mouse cursor in the center of screen 1.
- **Tap for Click** : When enabled, tapping anywhere in the main area of the trackpad window, except for the **Right** button, executes a left mouse click. When disabled, only a tap on the **Left** button executes a left mouse click.
- **Resolution** : Tap to cycle through the available resolution options for the trackpad mouse, or tap and swipe to see a menu of available resolution options. This resolution setting is separate from the resolution setting in pan/tilt mode.

Pan/Tilt Mode



Trackpad Pan/Tilt mode

While in pan/tilt mode, horizontal swipes within the main area of the trackpad window adjust the pan attribute of any selected fixtures. Vertical swipes adjust the tilt attribute. Diagonal and curved swipes adjust both pan and tilt accordingly. The main area of the trackpad window in pan/tilt mode displays a simple grid as a visual guide.

The title bar of the trackpad in pan/tilt mode includes tools, which control the behavior of the trackpad. These tools include:

- **P/T Mode** : Tap to cycle through the available pan/tilt modes, or tap and swipe to see a menu of available pan/tilt mode options. These options include:
 - **Pan Only** : This mode only allows adjustment of the pan attribute, ignoring any vertical movement.
 - **Tilt Only** : This mode only allows adjustment of the tilt attribute, ignoring any horizontal movement.
 - **Both** : This mode allows simultaneous adjustment of both the pan and tilt attributes.
- **Invert Mode** : Tap to cycle through the available invert modes, or tap and swipe to see a menu of available invert mode options. These options include:
 - **Off** : Both pan and tilt are adjusted normally.
 - **Pan Invert** : Pan inputs are interpreted in reverse of the normal direction. Tilt inputs remain normal.
 - **Tilt Invert** : Tilt inputs are interpreted in reverse of the normal direction. Pan inputs remain normal.
 - **Both** : Both pan and tilt inputs are interpreted in reverse of the normal direction.



- **Resolution** : Tap to cycle through the available resolution options for the trackpad control of pan and tilt, or tap and swipe to see a menu of available resolution options. This resolution setting is separate from the resolution setting in mouse mode.



Hint:

The **TrackpadMode**, **Mouse Resolution**, **P/T Resolution**, **P/T Invert Mode**, and **P/T Mode** settings are also available in the **Trackpad Window Settings** pop-up. To open this pop-up, tap **MA** in the upper-left corner of the **Trackpad** window.

7.2. Gestures

Use gestures on the touchscreens of the grandMA3 console to quickly navigate and adjust settings.



- **Tap**

Briefly tap the surface with a fingertip.



- **Tap & Hold**

Tap the surface and hold it for at least one second.



- **Zoom**

Tap the surface using two fingers and move them apart or toward each other.



- **Resize**

1. Tap the title bar or the title field using any finger of one hand and hold it.
 2. Briefly tap anywhere on the surface using a second finger.
 3. Release both fingers.
 4. The window is resized.
-



- **1 Finger Swipe**

Tap and brush in any direction.



- **2 Finger Scroll**

To scroll within a window, tap the surface with two fingertips and brush the surface in the desired direction.



Hint:

When using a mouse, press and hold the right and the left mouse buttons, and move the mouse accordingly.



- **3 or More Finger Scroll**

Requirement:

- Set the height and/or width of the user-defined area larger than the available screen space. For more information see the [Configuration of Displays](#) topic.

To scroll the user-defined area within the available screen space, tap the surface with three or more fingertips and brush the surface in the desired direction.



Hint:

When using a mouse, press and hold the **option** key (on macOS) or **Alt** key (on Windows), then press and hold the right and the left mouse buttons, and move the mouse accordingly.



• Drag & Drop

Use this gesture to move and resize windows within the user-defined area as well objects and elements within certain graphical editors (such as the 3D, Layout, Phaser Editor, or Timecode window).

1. Tap and hold with one finger.
2. Move the finger to the desired position.
3. Release the finger.



• 2 Finger Edit

In sheets and other similar grids:

The **Precise Edit** user profile setting defines the behavior of the **2 Finger Edit** gesture when it is used within sheets and other similar grids (such as the macro editor or patch menu).

Precise Edit disabled:

1. Tap the desired cell.
2. Tap with two fingers anywhere within the same window of the desired cell.

Precise Edit enabled:

1. Tap and hold one finger in the desired cell.
2. Tap anywhere on the same screen with a second finger.
3. Release both fingers.

In pools:

The **Precise Edit** user preference setting does not change the behavior of the **2 Finger Edit** gesture when it is used to edit objects within pools. To edit a pool object, follow the steps above under "**Precise Edit** enabled."

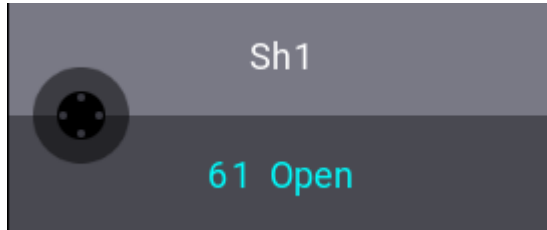
For more information about user profile settings, such as **Precise Edit**, see the [User Settings](#) topic.

• Virtual Encoder

The encoder icons in the Encoder Toolbar include a gesture for turning a virtual encoder to adjust values.

To change the value using a gesture in the user interface:

1. Tap and hold the desired encoder icon. The encoder icon turns dark.



Encoder icon


2. Draw circles around the center of the virtual encoder in the direction of the desired change. Turning clockwise will increase the value and turning counter-clockwise will decrease the value.
3. If desired, drag the pointer to another area of the screen. This moves the center of the virtual encoder to this new point. Larger circles result in finer control resolution.

For more information about the Virtual Encoder, see the [Encoder Toolbar](#) topic.

7.3. Command Area

This menu displays the Command Area of a console.

As only consoles have a command area, this menu is useful whenever you use an onPC.

To open Command Section, tap  in the control bar on the left of the screens 1 to 7 or **F3** on a keyboard.

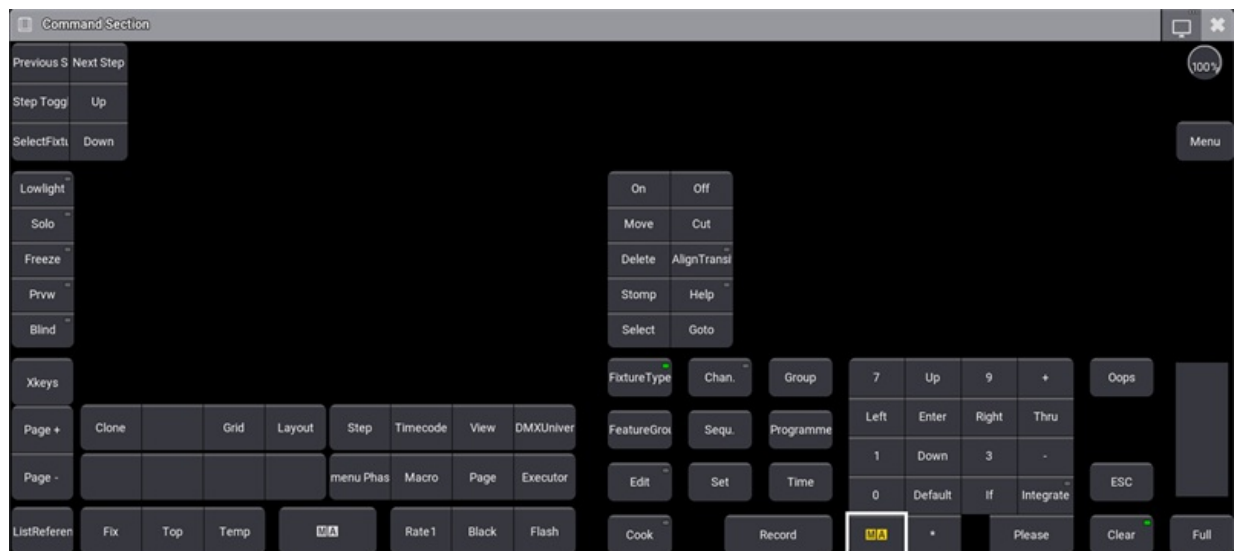


Command area

Apart from the command area this menu also displays:

1. **Xkeys**
2. **Grand master**
3. **Level wheel**

To use the latch function, tap and swipe a **MA** key. The **MA** key turns yellow and the keys of the Command Section changed as shown below:



To unlatch, tap on a **MA** key.

7.4. Oops Overlay

The Oops overlay displays the history of completed actions. Completed actions are displayed in form of a list and in chronological order. The action at the bottom of the overlay shows the latest entry.

The latest action or multiple actions can be Ooped. It is also possible to use Oops filters for different kinds of actions.

For more information about Oops, see [Oops keyword](#) and [Oops hardkey](#).



Important:

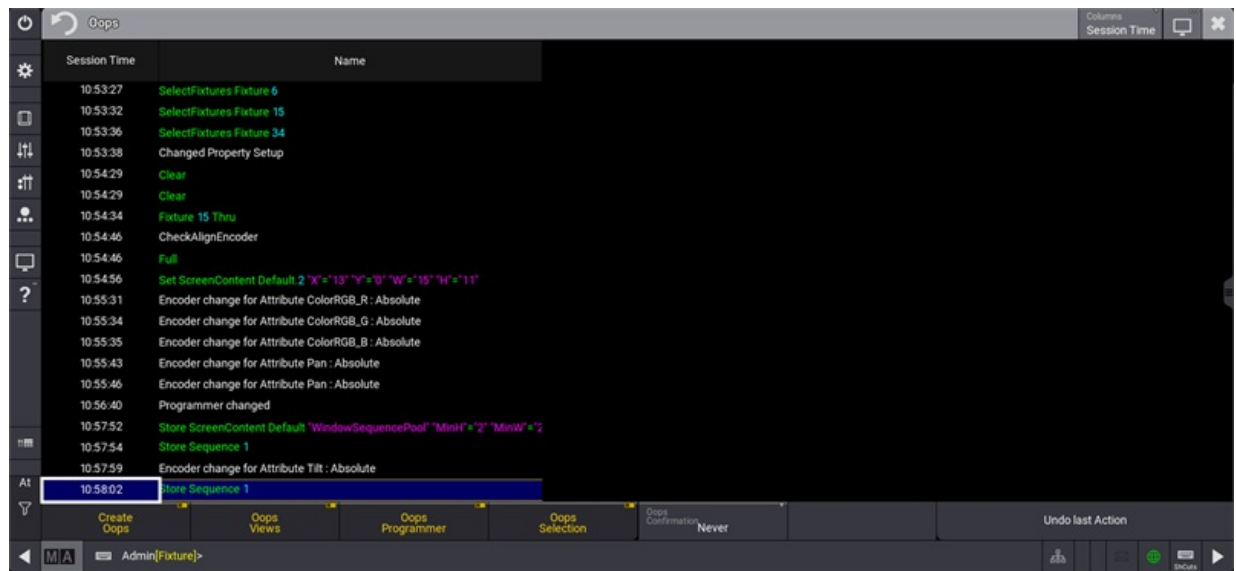
All Oops entries are cleared, when changing the Patch or the Live Patch.

To open the Oops overlay:

- Press and hold **Oops**.

or

- Execute the command **Menu "OopsOverlay"**.



Oops overlay



The overlay is separated into two columns:

- **Elapsed Time / Session Time:** Shows the bygone time since the action has been executed.
- **Name:** Shows the executed command or the action that has been done.

To toggle between different time units in the overlay, tap **Columns** in the title bar:

- **Elapsed Time:** Shows the passed time since the command has been executed.
- **Session Time:** Shows the time off the day the command has been executed.

Undo Actions



Hint:

Only the last action or a coherent amount of actions can be Oopsed. A selection of actions always includes the latest one.



Hint:

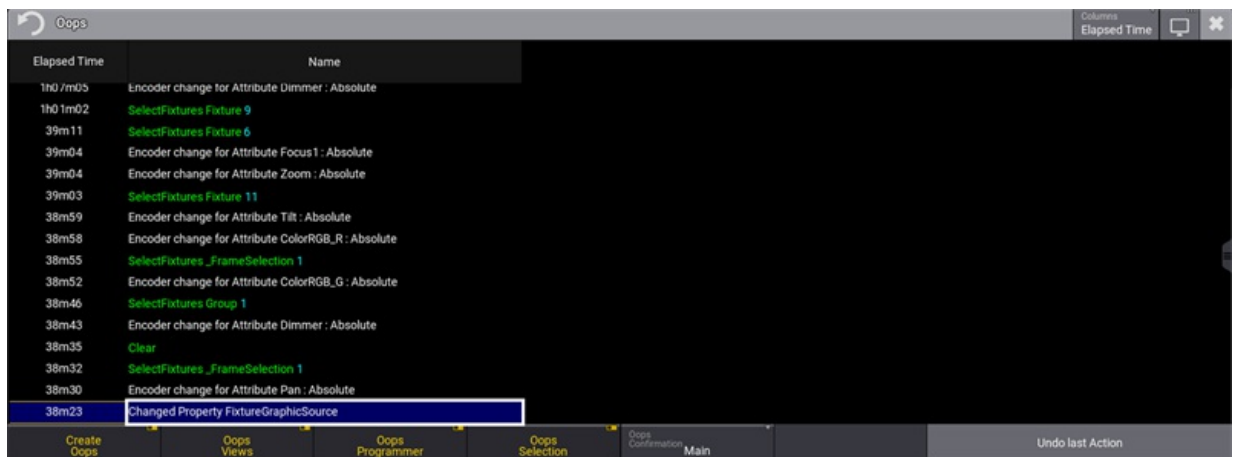
A blue background in the overlay indicates the selected action.

Requirements:

- Change at least two values in the show file to create Oops events, for example, by setting dimmer, position and color values for fixtures.

To undo the last action:

1. The action at the bottom is selected, when opening the Oops overlay.
2. Tap **Undo last Action** located the bottom right corner in the Oops menu. The latest action is undone and is vanished in the overlay.



Undo last Action

To undo multiple actions:

1. Use **2 Finger Scroll** or the scrollbar to scroll to a particular event in the Oops overlay.
2. Tap on any listed event to be undone. The latest event and all other events to the tapped event are selected.
3. Tap **Undo x Actions**. The x stands for the selected number of actions.
4. The selected actions are undone and are vanished from the overlay.



Undo multiple Actions

Toggle Oops Filters

The following filters in the Oops overlay allow to filter events:

- Create Oops
- Oops Views
- Oops Programmer
- Oops Selection



Hint:

The four filters operate the corresponding settings of the User Configuration. For more information, see [User Settings](#).



Hint:

/NoOops does not generate oops events when executing commands. For more information, see [/NoOops Option Keyword](#).

- To create no events at all, disable **Create Oops**. Oops View, Oops Programmer and Oops Selection are greyed out.



Create Oops

Oops Confirmation

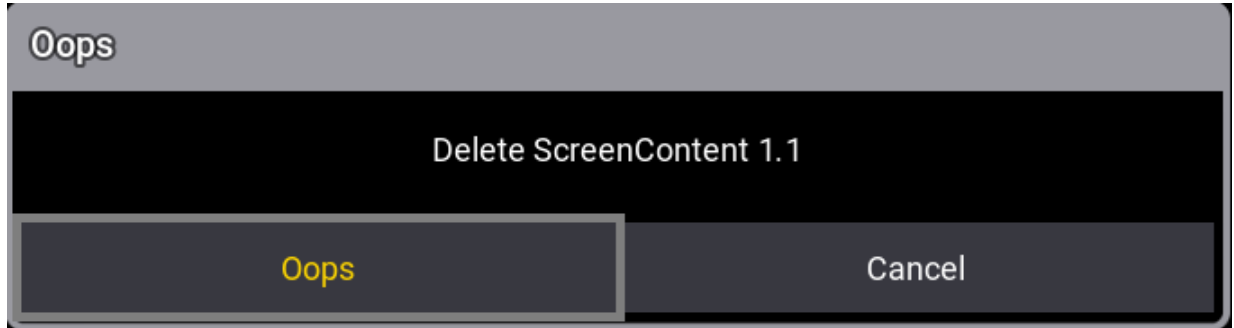
The setting **Oops Confirmation** defines when an Oopsed event has to be confirmed via a pop-up.

Oops Confirmation can be set to the following options:

- **Always**: Any Oopsed event needs to be confirmed.
- **Main**: Important actions needs to be confirmed, for example, storing, moving, or deleting objects.
- **Never**: No confirmation is needed. Oops Configuration is set to Never by default.

To change the Oops Confirmation to Always:

1. At least one Oops event is created.
2. Press and hold **Oops**. The Oops overlay opens.
3. Tap, hold and swipe **Oops Confirmation**. A dropdown menu opens.
4. Select **Always**. The dropdown menu closes.
5. Close the Oops Overlay.
6. Press **Oops**. A Oops pop-up appears:



Undo pop-up

7. Tap **Oops** to confirm. The last action is undone.

Oops in a Session

Ooping an action is possible in a session with multi-users. For more information about multi-users in sessions, see [Multi User Systems](#) and [Session](#).


Investigations about already changed objects by other stations are made, when Oops is used in a session. The first station displays a failure message in the command line, if the object was already changed by another station:

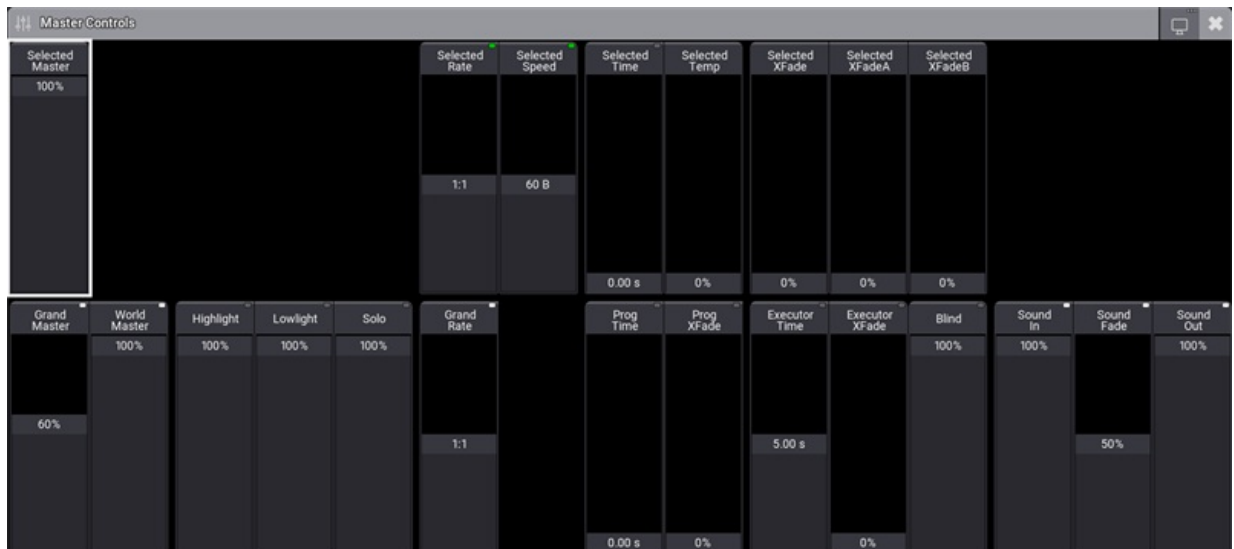


Error message

7.5. Master Controls

The menu Master Controls displays all masters.

To open Master Controls, tap  in the control bar on the left of screens 1 to 7 or press **F4** on a keyboard.



Master Controls

1. Upper section is used in the selected sequence.
2. Lower section is used for global settings.



Adjust the value of the on-screen faders depending on their type:

- On-screen faders with indicators in the right corner of the title bar:
 - To change modes, see **Masters**.
 - To adjust values, tap and slide upward or downward.
- On-screen faders without indicators:
 - To adjust values, tap and slide upward or downward.




Hint:

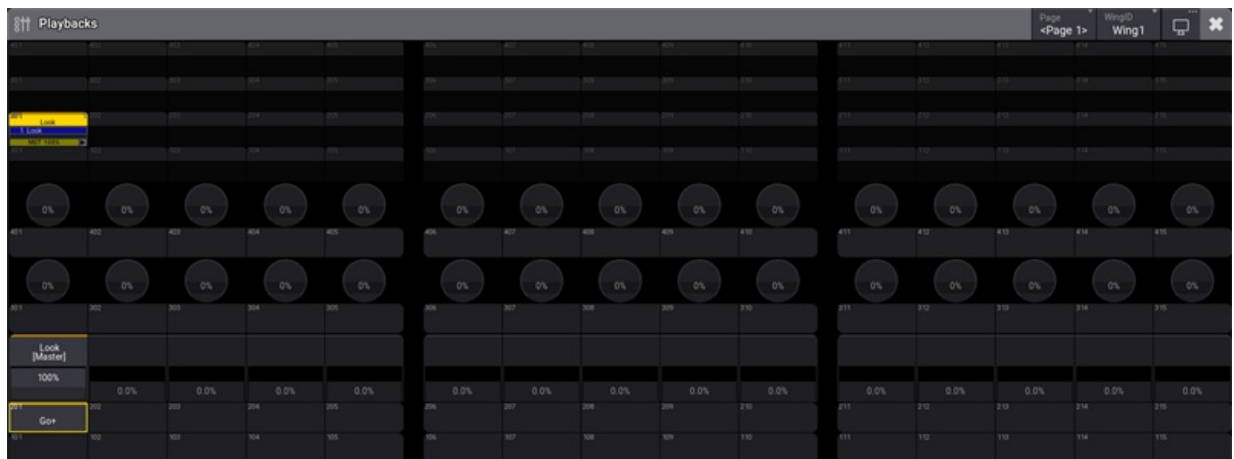
You can enable or disable the on-screen faders **Highlight**, **Solo**, and **Blind** using **Highl**, **Solo**, and **Blind** on the console.

7.6. Playback Controls

The menu Playback Controls graphically displays physical faders of a console and/or wings.

To open Playback Controls, tap  in the control bar on the left of screens 1 to 7 or **F5** on a keyboard.


For more information on the functionality of executors see [Executors](#).

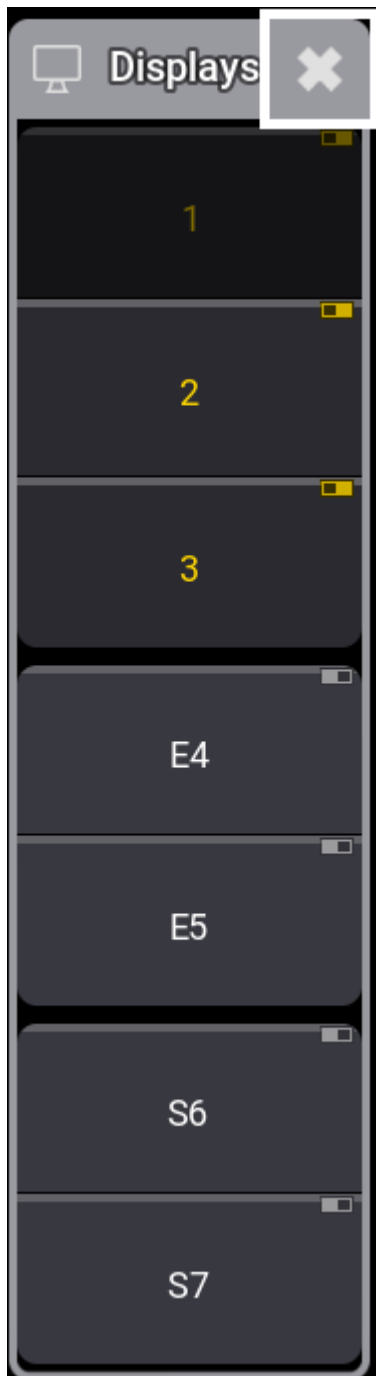


Empty Playback Controls

7.7. Displays in grandMA3 onPC

The Displays pop-up allows toggling showing, or hiding all 7 screens on the grandMA3 onPC.

To open the Displays pop-up on the grandMA3 onPC, tap  in the **control bar**.




Displays



Displays can be toggled On or Off by tapping the relevant buttons.

The number and the indicator turn yellow on the displays that are shown.

The pop-up can be closed by tapping the icon again in the control bar or by tapping the  in the upper right corner of the pop-up.


For information on the location of menus, see [Change Menu Locations](#).

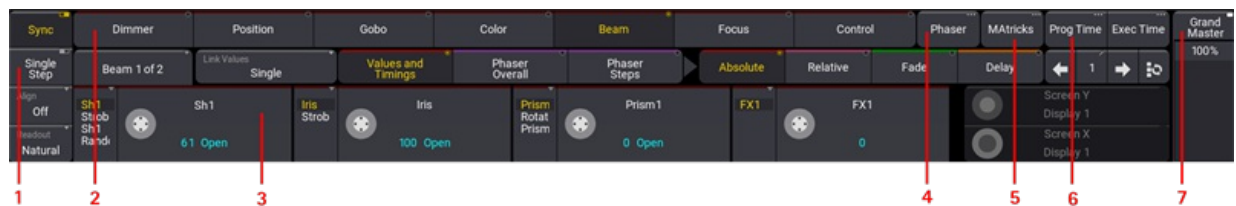
7.8. Encoder Bar

The encoder bar displays the attributes or functions linked to the encoders. The encoder bar also includes shortcuts for quick access to a selection of user profile settings and programming tools.

On the grandMA3 full-size, the grandMA3 full-size CRV, the grandMA3 light, and the grandMA3 light CRV; the encoder bar appears on screen 8. By default, the encoder bar appears at the bottom of screen 1 on all other grandMA3 consoles and the grandMA3 onPC software.

To show or hide the encoder bar on screen 1 on any grandMA3 console or the grandMA onPC software, tap **Show Encoder Bar** in the configure display menu on screen 1. For more information on the configure display menu, see the [Configuration of Displays](#) topic.

The encoder bar  is also available as a window under the **More** and **All** tabs in the **Add Window** pop-up. For more information on adding windows, see the [Add Windows](#) topic.



Encoder bar

The encoder bar includes the following main sections:

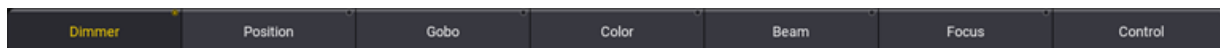
1. User profile settings and align mode.
 - Tap **Sync** to disable or enable synchronized playback of phasers in the programmer. For more information on sync, see the [Phasers](#) topic.
 - Tap **Single Step** to enable or disable single step. For more information on the single step setting, see the [User Settings](#) topic. For more information on using steps in general, see the [Phasers](#) topic.
 - Tap **Align** to cycle through align modes or tap and swipe to open a pop-up menu containing all of the available align modes. For more information on align, see the [Align](#) topic.
 - Tap **Readout** to cycle through attribute value readout options or tap and swipe to open a pop-up menu containing all of the available readout options. For more information on readout, see the [User Settings](#) topic.
2. The feature group control bar appears along the top of the encoder bar. For more information, continue to the following sub-topics.
3. The encoder toolbar occupies most of the area in the encoder bar. The layer toolbar is part of the encoder toolbar. For more information, [Feature Group Control Bar](#) and [Encoder Toolbar](#).
4. Tap **Phaser** to open a temporary version of the Phaser Editor window. For more information on the Phaser Editor window, see the [Phaser Editor](#) topic.
5. Tap **MATricks** to open a temporary version of the MATricks window. The background, top bar, and three dots in the upper-right corner of this button will change color to indicate that MATricks is active. For more information on MATricks, see the [MATricks and Shuffle](#) topic.



6. Time Control. Two on-screen fader controlling two different timing elements. They are called **Prog Time** (Program Time) and **Exec Time** (Executor Time). For more information, see [time control](#).
7. The grand master appears at the right edge of the encoder bar. For more information on the grand master, see the [Grand Master](#) topic.

7.8.1. Feature Group Control Bar

The feature group control bar appears along the top of the encoder bar. Each feature group used in the show file appears in the form of a radio button.



Feature group control bar with Dimmer selected

Adding fixtures, which use additional feature groups, to the patch automatically adds the necessary feature groups to the feature group control bar.



Hint:

When the programmer is clear, the bars across the top of the feature group buttons are grey. As the selection changes, the bars across the top of any feature groups available in the current selection change color. The color coordinates with the color of the current layer in the layer toolbar.

Select a Feature Group

To select a feature group in the control bar, tap the desired feature group. The radio button is enabled, and the encoder toolbar adjusts to display the attributes of the selected feature group.

To change the feature group using a shortcut, press and hold **Preset** + press one of the numbers from **1** to **9** on the numeric keypad. For example, to change to the color feature group, press **Preset** + **4**.

Active Programmer Values

To activate the attributes of a feature group in the programmer, tap twice on the desired feature group. The information activates and a colored marker appears on the feature group button. To deactivate, tap the feature group once again.



Important:

The color of the marker or markers on the feature group button corresponds to the active layer or layers of information. These markers will appear as long as there is active values, no matter how the values were activated.



Hint:

To activate information in a specific layer, tap the desired layer in the layer control bar in the encoder toolbar before tapping the feature group.



Feature Group as the Object of a Command

Many function keywords (including On, Off, Park, Unpark, Remove, Release, Stomp, and Default) can execute their commands using feature groups as their destination objects.

Example

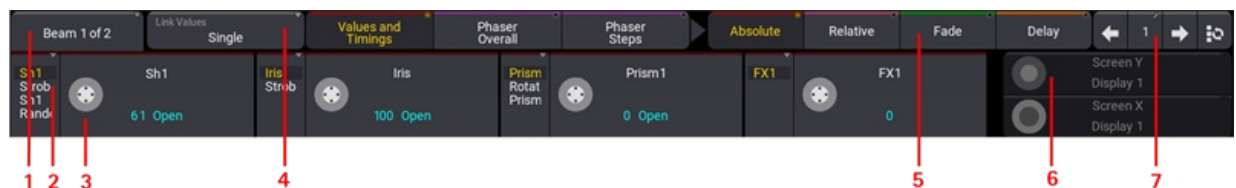
To remove all dimmer values within the current selection from the programmer:

- Press **Off** and tap **Dimmer** .

The dimmer values of the current selection are removed from the programmer.

7.8.2. Encoder Toolbar

The encoder toolbar represents the largest portion of the encoder bar. The encoder toolbar is context-sensitive and displays different sets of controls as different editors become active. The default set of controls in the encoder toolbar provides access to and display of attribute encoder information.



Encoder toolbar

Attribute controls available in the encoder toolbar include:

- Feature button:** This button displays the name of the current feature. It also indicates the total number of features available in the current feature group.
Tap to cycle through features or tap and swipe to open a pop-up menu with available features.
For more information about features and feature groups, see the [Feature Group](#) topic.
- Channel function of the attribute:** These buttons display the current channel function of the attribute displayed directly to the right. They also display additional channel functions when available.
Tap to cycle through subattributes or tap and swipe to open a pop-up menu with available subattributes.
- Attribute encoder display:** These buttons display the names of the attributes currently linked to the encoders, and show the values of those attributes.
The center of the encoder icon includes an additional image, which changes depending upon the current resolution of the encoder. For more information on encoder resolution, see the [Encoder Resolution](#) topic.
To open the calculator, tap the attribute name field or the attribute value field. For more information about attributes and subattributes, see the [Attribute Definitions](#) topic.
For more information about the gesture for rotating a virtual encoder, see the [Gestures](#) topic.
- Link button:** In cases where the same value, timing, phaser, or resolution adjustments should apply to multiple attributes, the link button offers a few options for defining multiple, simultaneous attribute destinations. For more information about the link button, see the section below.
- Layer toolbar:** The default layer is absolute. Tap any layer button in the toolbar to access the desired layer on the attribute encoders. For more information about the layer toolbar, see the [section below](#).
- Screen encoder:** If the screen encoder is enabled in the user profile settings of the current user, this area displays the basic functionality of the screen encoder. If the screen encoder is disabled in the current user profile, the functionality of the fifth dual encoder will be similar to that of the other four, and the display in this area will also be similar to the displays above the other four encoders. For more information about the screen encoder, see the [User Settings](#) topic.
- Step bar:** The left and right arrows allow navigation between phaser steps in the programmer. The step number button displays the current phaser step in the programmer. Tap the step number button to open a calculator to choose a specific step. Tap the button to the right of the right-arrow to select all phaser steps. For more information about phaser steps, see the [Phasers](#) topic.



Link Button

The link button maintains different link settings for the different types of layers found in the layer toolbar as well as encoder resolution. The text in the link button will update automatically to show the link setting of the current layer type. The available link settings are:

- **Single:** Adjustments made on one encoder apply only to the attribute on that encoder.
- **Feature:** Adjustments made to one encoder within a feature apply simultaneously to all attributes in the feature.
- **AtFilter:** Adjustments made to one encoder apply simultaneously to all attributes currently enabled in the at filter.

Link Values

Value layers include:

- Absolute
- Relative

When a value layer is selected, tap **Link Values** to cycle through the link options or tap and swipe to open a pop-up menu with all of the available link options. The available link options for value layers include:

- Single
- Feature

Link Timing

Timing layers include:

- Fade
- Delay

When a timing layer is selected, tap **Link Timing** to cycle through the link options or tap and swipe to open a pop-up menu with all of the available link options. The available link options for timing layers include:

- Single
- Feature
- AtFilter
- **Active Only (A):** If enabled, only attributes with active values in the programmer will get new values.
- **MultiStep Only (S):** If enabled, new values are only applied to attributes with two or more active steps.

Link Phaser Overall

Phaser layers include:

- Speed



- SpeedMaster
- Phase
- Measure

Link Phaser Steps

- Accel
- Decel
- Transition
- Width

When a phaser layer is selected, tap **Link Phaser** to cycle through the link options or tap and swipe to open a pop-up menu with all of the available link options. The available link options for phaser layers include:

- Single
- Feature
- AtFilter
- Active Only
- MultiStep Only

Link Resolution

While the MA key is pressed and held, tap **Link Resolution** to cycle through the link options or tap and swipe to open a pop-up menu with all of the available link options. The available link options for encoder resolution include:

- Single
- FeatureGroup

Layer Toolbar

Presets and cues can store and recall multiple layers of data for each attribute. The layer toolbar provides access to all available layers. Layers are color-coded. Markers and text backgrounds using matching colors in the attribute encoder displays, feature group buttons, fixture sheet, and sequence sheet denote active or stored data for the corresponding layer. For more information about colors, see the [Colors](#) topic.

Tap a button in the layer toolbar to access data for the desired layer. The attribute encoders and any sheet with a layer selection set to auto will display the desired layer. For more information about value and timing layers, see the [What is the Programmer](#) topic. For more information about phaser layers, see the [Phasers](#) topic. For more information about using GridPosition to create MAagic presets, see the [Create New Presets](#) topic.

When the programmer is clear, the bars across the top of the attribute encoder displays and the channel function displays are gray. The selected layer group returns to **Values and Timings** and the **Absolute** layer. The layer selection in Phaser Overall and Phaser Steps stays as selected after clearing the programmer. As the selection



changes, the bars across the top of any attributes and channel functions available in the current selection change color. The color coordinates with the color of the current layer in the layer toolbar. The lights under the dual-encoders also follow the same behavior.



Hint:

The colors for each phaser-related layer can be changed in the color theme in the ColorGroup "ProgLayer". For more information, see [color theme](#).

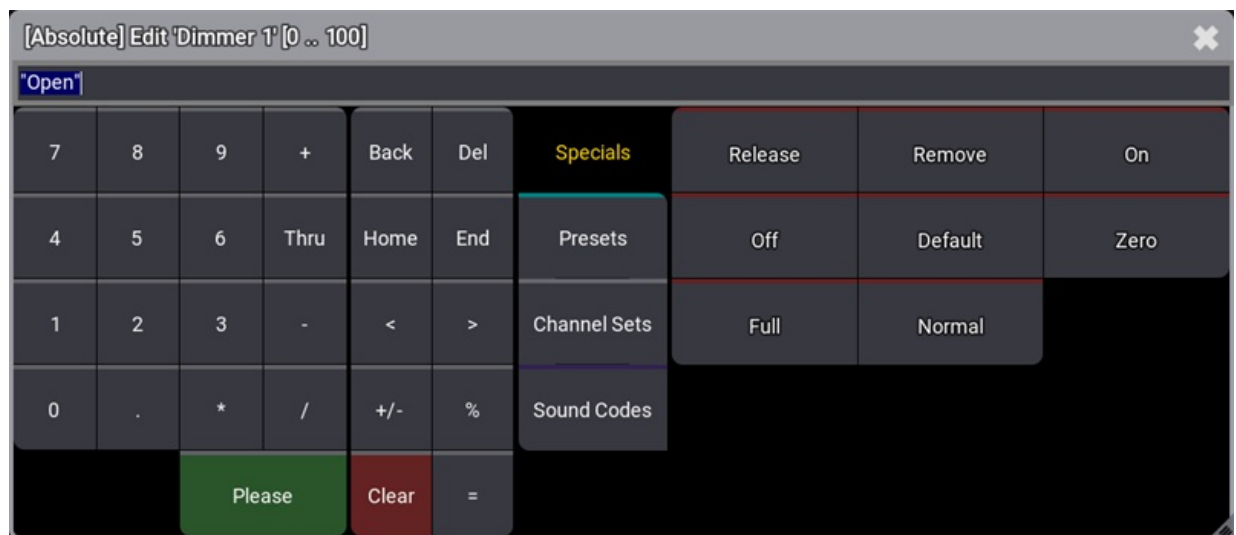
7.9. Calculator

When editing a field that accepts numeric values, the calculator appears. This includes pressing one of the dual encoders while it is displaying an editable value.

The calculator is a dynamic window, and its appearance changes depending upon the edited item and the type of value to be entered.

Example

- Edit a dimmer attribute



The calculator editing a dimmer attribute

The title bar displays several useful pieces of information about what the calculator is currently editing:

- **Layer:** If applicable, the first set of brackets contains the value layer.
- **Attribute or parameter:** The name of the attribute or parameter appears after the word "Edit."
- **Value Range:** The last set of brackets contains the allowed range of input values.

Input Field

The input field appears below the title bar. When the calculator opens, the input field displays the current value of the edited object. The text of this value is completely selected so that any immediate entry will replace the existing value. Alternatively, the text can be deselected and retained as part of the entry.

Standard Buttons

The following buttons appear in all calculators.

Number Pad

The calculator displays a number pad on the left. These buttons share the same functions in the calculator as the number keypad in the command section of the console.

Function Buttons

Function buttons appear to the right of the number pad. These buttons include:

- **Back** : Tap to delete characters to the left of the cursor.
- **Del** : Tap to delete characters to the right of the cursor.
- **Home** : Tap to set the cursor at the beginning of the input field.
- **End** : Tap to set the cursor at the end of the input field.
- **<** : Tap to move the cursor to the left.
- **>** : Tap to move the cursor to the right.
- **+/-** : Tap to insert a negative or a positive value. For more information see - **[Minus] Key** or the + **[Plus] Key**.
- **%** : Tap to enter the percent sign.
- **=** : Tap to enter the equals sign.
- **Clear** : Tap to delete the entire entry.
- **Please** : Tap to confirm and apply the value. The calculator closes.



Hint:

Entries using a standard keyboard, whether integrated under the console's armrest or connected through USB, also populate the input field of the calculator.

With one exception, keyboard shortcuts are disabled when entering text into a calculator. The exception is that **T** will enter the Thru keyword.



Hint:

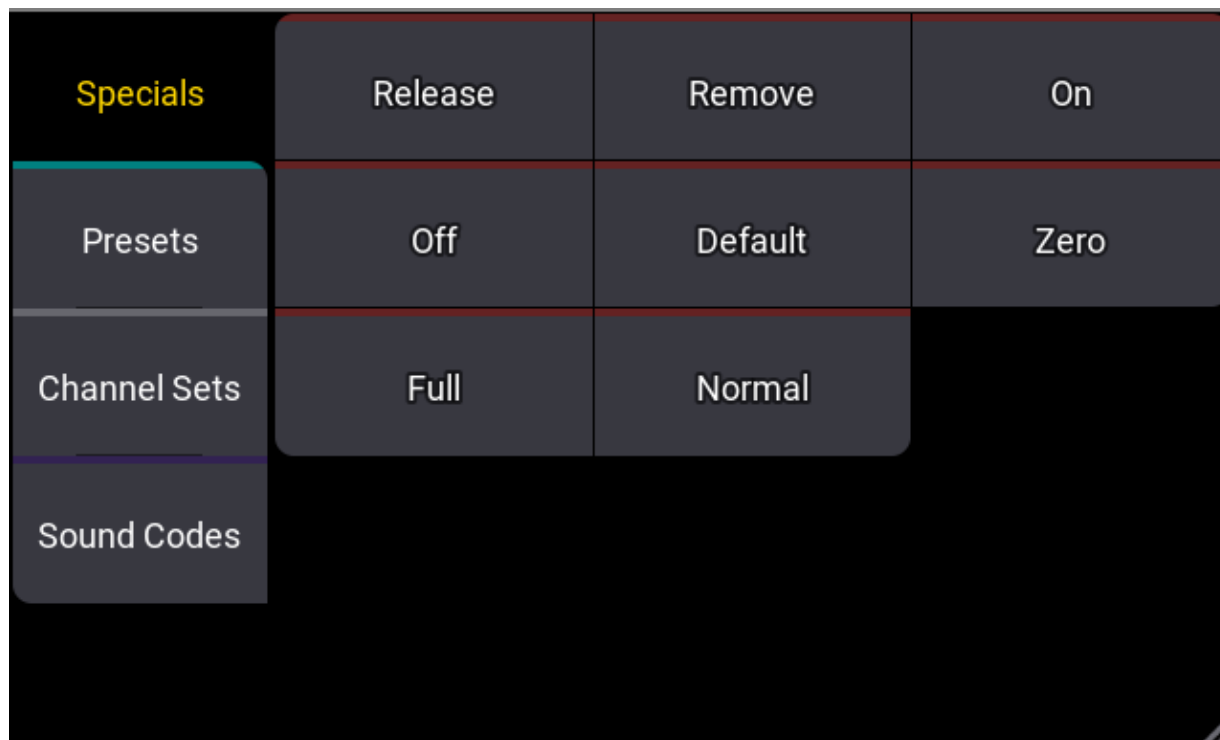
When the input readout is set to one of the hex options, an additional row of buttons appears below the number pad, allowing entries of hexadecimal values A, B, C, D, E, and F.

Additional Input Options

Four tabs of additional input options appear to the right of the function buttons.

Specials

To open the special buttons tab, tap **Specials**.



Specials in the calculator

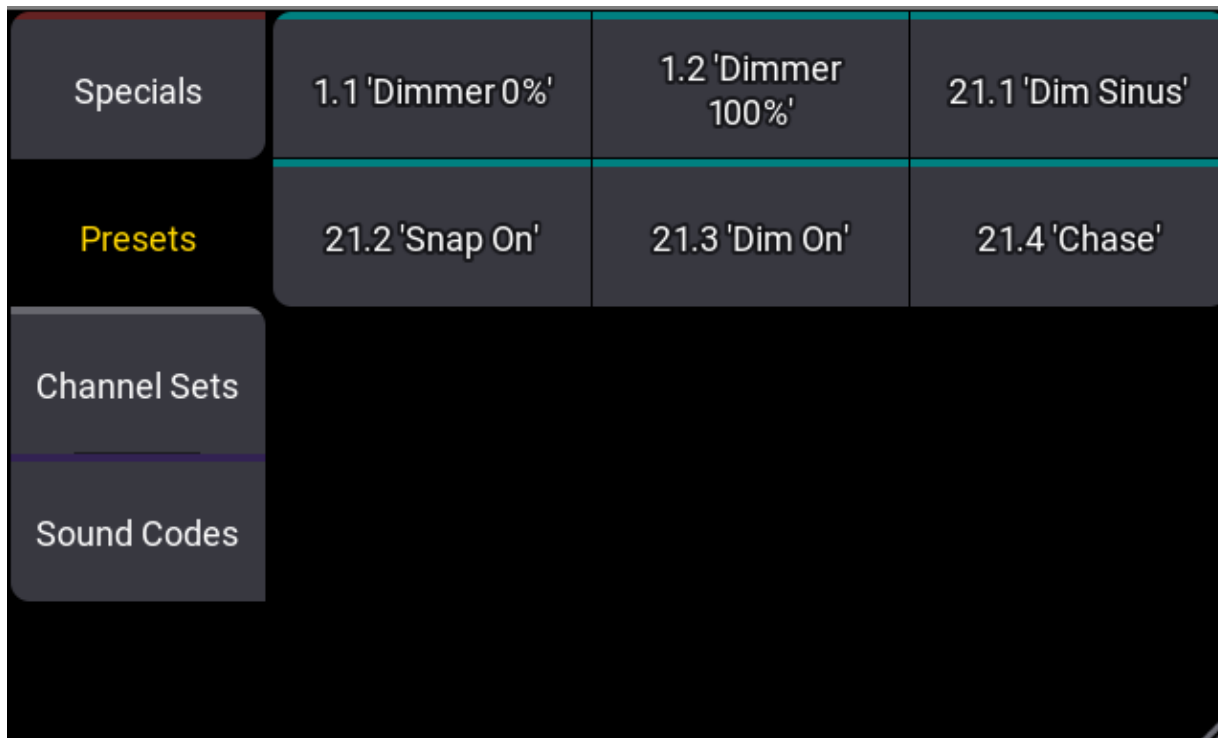
Use these buttons to enter special commands for attribute values:

- **Release** : Tap to enter a release value. For more information see the **Release Keyword**.
- **Remove** : Tap to enter a remove value. For more information see the **Remove Keyword**.
- **On** : Tap to activate values in the programmer without changing the value.
- **Off** : Tap to clear the values in the programmer.
- **Default** : Tap to enter the default value defined for the fixture type. For more information see the **Default Keyword**.
- **Zero** : Tap to set the values to 0. For more information see the **Zero Keyword**.
- **Full** : Tap to set values to 100%. For more information see the **Full Keyword**.
- **Normal** : Tap to set the normal value stored in the user profile. For more information see the **Normal Keyword**.

The at overlay in the control bar also offers quick access to limited versions of these special values. For more information, see the **Control Bar** topic.

Presets

To open the presets tab, tap **Presets**.



Presets in the calculator

Use these buttons to call presets for attribute values. This tab displays any preset from any preset pool in the currently selected data pool, which contains data that can be applied to the attribute currently edited by the calculator. Each button displays the preset number, using the format: [Feature Group].[ID], as well as the preset label. For more information on presets, see the [Presets](#) topic.

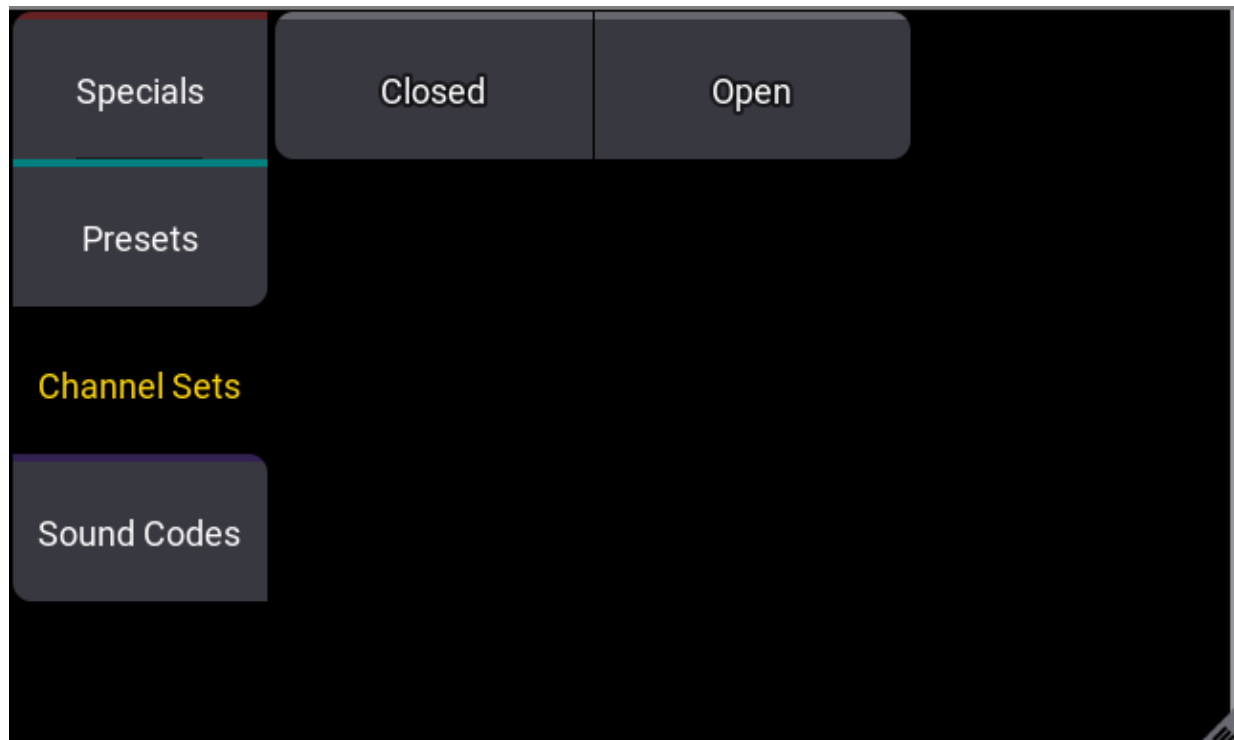


Important:

Only the buttons within the presets tab recall presets in the calculator. All other value sources in the calculator produce unlinked values, which may be more difficult to reproduce or update.

Channel Sets

To open the channel sets tab, tap **Channel Sets**.



Channel Sets in the calculator

Use these buttons to enter attribute values based on channel sets defined for the fixture type. The channel sets displayed in this tab are the same as those displayed in the smart window. For more information on the smart window, see the [Smart View](#) topic.



Hint:

When editing attributes with multiple channel functions, a button for each channel function appears between the title bar and the input field of the calculator. Selecting a channel function shows only the relevant channel sets within the channel sets tab.

Sound Codes

To open the sound codes tab, tap **Sound Codes**.

Specials	All	Bass	Mid
Presets	High	Band1	Band2
Channel Sets	Band3	Band4	Band5
Sound Codes	Band6	Band7	InvAll
	InvBass	InvMid	InvHigh

Sound codes in the calculator

Use these buttons to link the attribute values to an incoming audio signal. Values can reference the total volume of the audio input or they can reference smaller frequency bands within the signal. The available options include:

- **All** : Use for the total volume of the incoming signal.
- **Bass** , **Mid** , and **High** : Use to reference one of three broad frequency bands.
- **Band1** through **Band7** : Use to reference one of seven narrower frequency bands.
- **Inv...** : Use to reference the inverse value of any of the above options.



Hint:

Use the SoundIn and SoundFade masters to adjust how the console reacts to incoming audio signals.

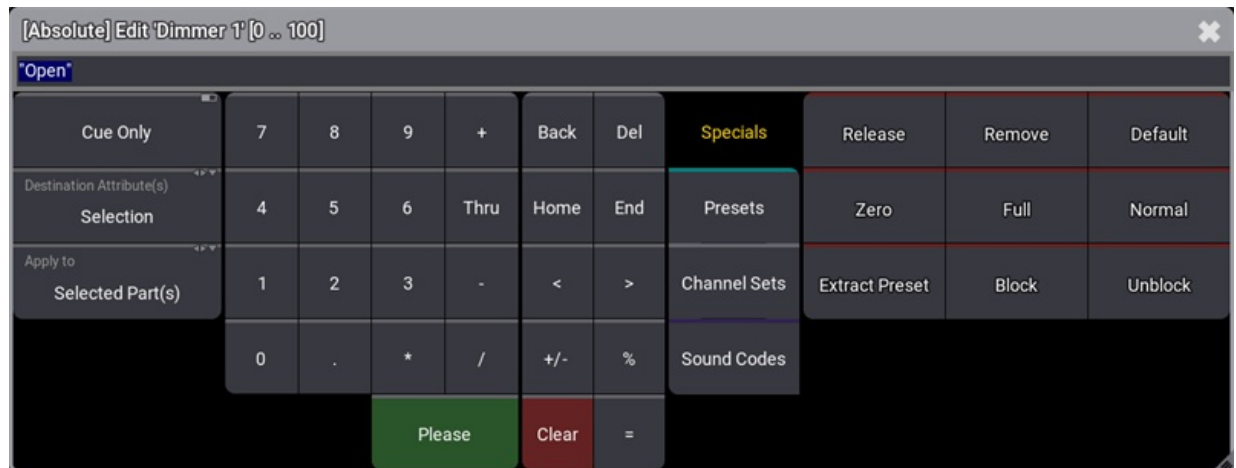
For more information about using sound input, see the [Sound Window](#) topic.

Track Sheet Calculator

When editing an attribute within the sequence sheet with the track sheet option enabled, the calculator appears with additional options to the left of the number pad. To edit attributes in the track sheet, drag the cursor across the desired attributes and up or down over the desired cues or cue parts. Then, use a 2-finger tap to edit the highlighted cells.

Example

- Edit a dimmer attribute in the track sheet



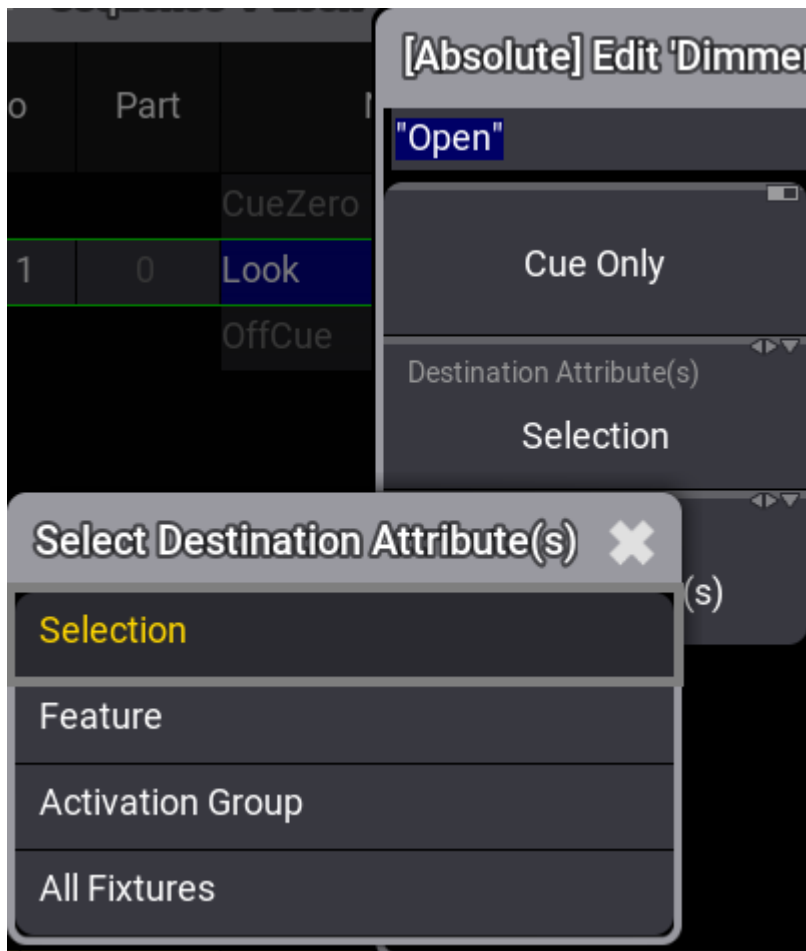
Calculator editing a dimmer in the track sheet

Cue Only

The cue only toggle appears at the top of the additional options to the left of the number pad. Toggle **Cue Only** off to allow changes to track forward. Toggle **Cue Only** on to prevent changes from tracking beyond the range of cues defined by the edit options. For more information on cue only, see the **Store Cues** topic.

Destination Attribute(s)

Tap **Destination Attribute(s)** to cycle through the available attribute destination options, or tap and swipe to see a menu of all available attribute destination options.



Attribute destination options

The select destination attributes menu includes the following options:

- **Selection** : Edits will only apply to the selected attribute cells.
- **Feature** : Edits will apply to all attributes within the feature group of the selected attribute cells, staying within the selected fixtures.
- **Activation Group** : Edits will apply to all attributes within the activation group of the selected attribute cells, staying within the selected fixtures.
- **All Fixtures** : Edits will only apply to all attributes of all fixtures within the cue part.

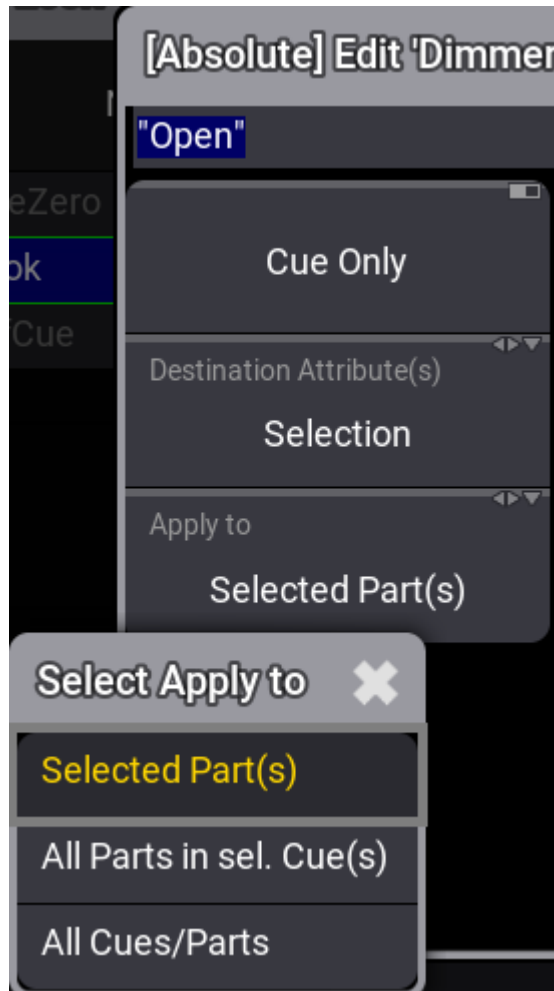


Hint:

Think of this option as expanding the effect of the edit horizontally within the track sheet.

Apply to

Tap **Apply to** to cycle through the available cue and part options, or tap and swipe to see a menu of all available cue and part options.



Apply to options

The apply to menu includes the following options:

- **Selected Part(s)** : Edits will only apply to the selected cue part cells.
- **All Parts in sel. Cue(s)** : Edits will apply to all parts within the selected cues.
- **All Cues/Parts** : Edits will apply to all parts of all cues within the sequence.



Hint:

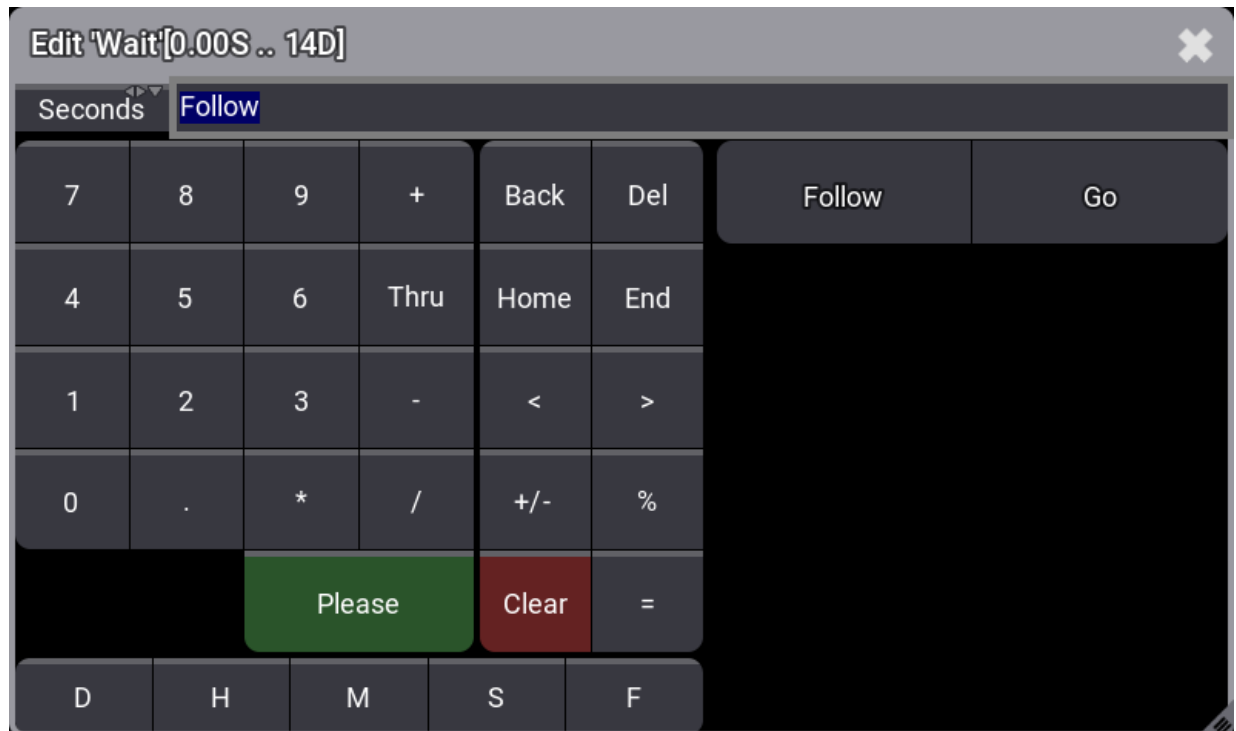
Think of this option as expanding the effect of the edit vertically within the track sheet.

Time Calculator

When editing times, the calculator appears with different readout options and additional buttons.

Example

- Edit the wait time for a line in a macro



Calculator editing the wait time for a line in a macro

Additional buttons appear below the number pad. These buttons allow for quick entry of time units:

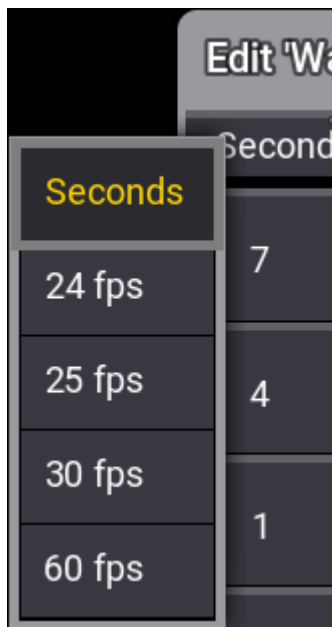
- **D**: Day
- **H**: Hour
- **M**: Minute
- **S**: Second
- **F**: Frame

Additional Input Options

Additional input options appear to the right of the function buttons. When editing a cell that can also contain triggers, this area displays trigger options.

Time Readouts

The readout selection appears to the left of the input bar. Tap the readout to cycle through the available readout options, or tap and swipe to see a menu of all available readout options.

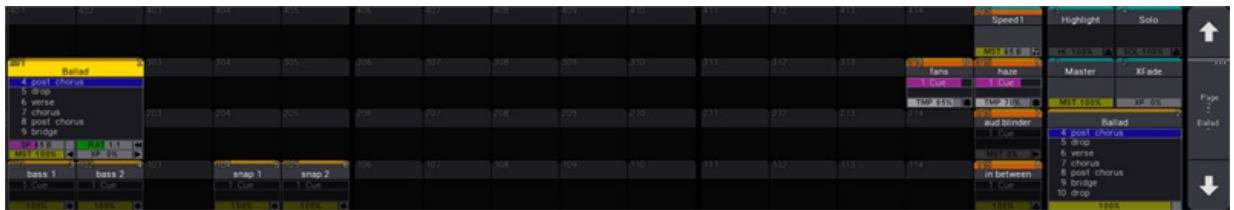


Time readout options

7.10. Playback Bar

The playback bar displays the objects currently assigned to the corresponding executors. The bar includes the playback status of each object as well as the available controls assigned to the executor keys, faders, and knobs.

The playback bar appears permanently on some screens and optionally on other screens. Depending upon where an instance of the playback bar appears, the bar may contain additional information about the assignment and status of the **Master Area** or **Custom Area** as well as page navigation controls. For more information, see [Masters](#).

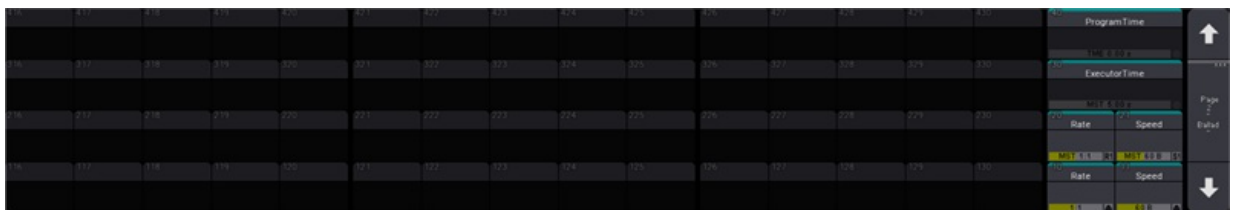


Playback bar with master area and page navigation.

- Tap the display of any executor in the playback bar to open the **Assign Menu** for that executor. For more information about the **Assign Menu**, see the [Assign Object to an Executor](#) topic.
- Tap and hold the display of any object assigned to an executor in the playback bar to edit the object. When editing a sequence, a temporary version of the [Sequence Sheet](#) appears.

The playback bar with the master area and page navigation always appear on screen 9:

- To show or hide this version of the playback bar on screen 2 on any grandMA3 console or the grandMA3 onPC software, tap **Show Playback Bar** in the **Configure Display** pop-up on screen 2.
- For more information about screen allocation, see the [Screen Allocation](#) topic.



Playback bar with custom area and page navigation.

The playback bar with the custom area and page navigation always appear on screens 10, 11, and 12:

- To show or hide this version of the playback bar on screens 3, 4, or 5 on any grandMA3 console or the grandMA3 onPC software, tap **Show Playback Bar** in the **Configure Display** pop-up on the desired screen.



Playback bar with Xkeys.

The playback bar can also appear on screen 7, showing the assignment and status of the Xkeys:

- To show or hide the playback bar on screen 7 on any grandMA3 console or the grandMA3 onPC software, tap **Show Playback Bar** in the **Configure Display** pop-up on screen 7.

A window showing the assignment and status of the Xkeys is available under the **Common** and **More** tabs in the **Add Window** pop-up. Additionally, a version of the playback bar is also available as a window under the **Common** and **More** tab in the **Add Window** pop-up. The **Playback** window does not include the custom area, master area, or the same page navigation controls as the **Playback Bar**. For more information on adding windows, see the [Add Windows](#) topic.




Important:

For more information about executors, page navigation, and the playback window and settings, see the [Executors](#) topic.

7.11. Command Wing Bar

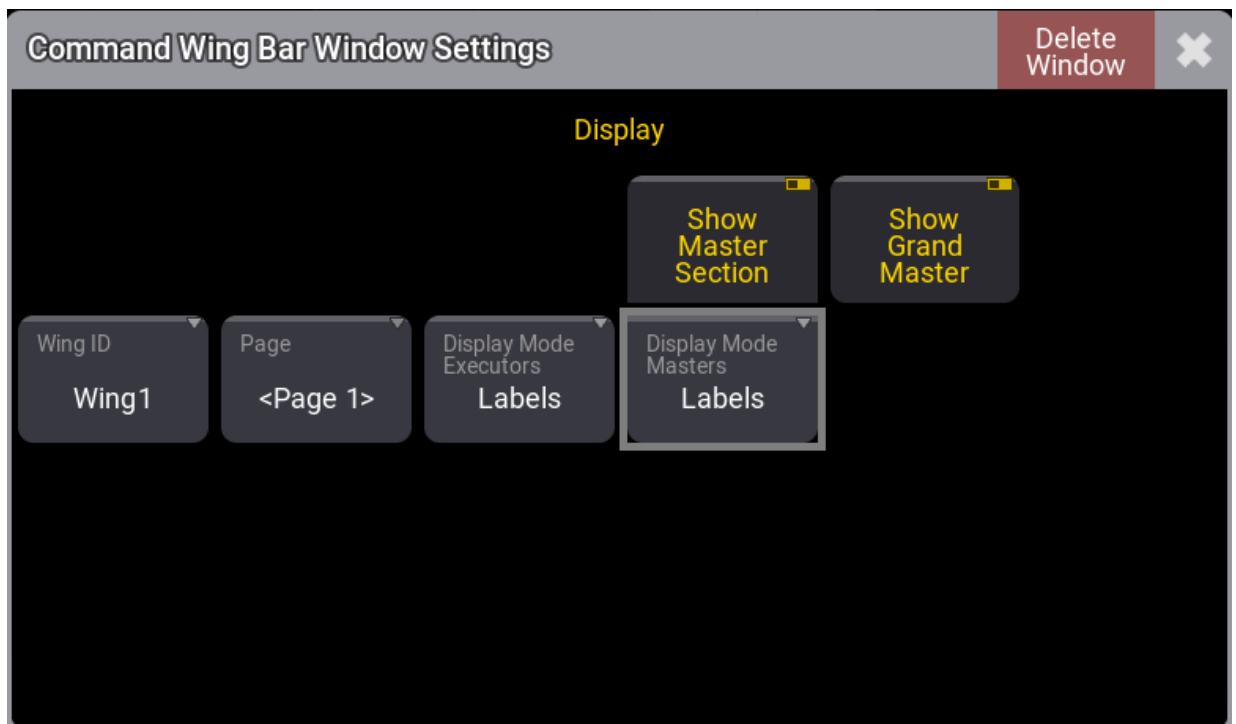
The **Command Wing Bar** combines a small version of the **Playback Bar**, including two sections of executors and the **Master Area**, with a compressed version of the **Encoder Bar**. This combined bar is helpful for optimizing screen space and minimizing the views necessary to move between programming and playback, especially when using the grandMA3 onPC software. The executors displayed on the playback side of the bar coincide with the executor sections available on the grandMA3 onPC command wing and command wing XT.

In the grandMA3 onPC software, tap the  icon in the **Control Bar** to toggle the **Encoder Bar** to the **Command Wing Bar**. The **Command Wing Bar** is also available as a window under the **More** and **All** tabs in the **Add Window** pop-up. For more information on adding windows, see the [Add Window](#) topic.



Command Wing Bar

Tap **MA** in the upper-left corner of the **Command Wing Bar Window** to access the **Command Wing Bar Window Settings** pop-up.



Command Wing Bar Window Settings pop-up

This pop-up includes the following settings:



- **Display Mode Executors** : Tap to toggle the display of the executors in the window between **Labels** and **Hardware Buttons** .
- **Display Mode Masters** : Tap to toggle the display of the special masters in the window between **Labels** and **Hardware Buttons** .

The **Window Settings** topic describes additional window settings.



7.12. Colors

grandMA3 uses a number of colors to provide information about states of values and their origin. There are two editable color themes included in the software, and it is possible to create, import, and export additional color themes. The following topics give an overview of the colors defined in the default color theme, as well as an introduction to the color theme editor.



Important:

Use caution when editing the color theme, as it is possible to make information on the screens difficult or impossible to read; for example, selecting the same color for both text and the background behind the text.



Hint:

MA + **MA** + **Clear** activates the default color theme and resets all values of that theme to the manufacturer defaults.



7.12.1. System Colors



Important:

The colors described here are included in the default color theme.
For more information about alternate color themes, see the **Color theme** topic.



Hint:

MA + **MA** + **Clear** activates the default color theme and resets all values of that theme to the manufacturer defaults.

Sheet Colors for Fixture Names and IDs

The fixture names and numbers in sheets have three different color combinations.

White:

Name	FID	IDType	CID
QuantPro 1	1	Fixture	

White names and numbers with a dark gray background

White fixture names and IDs on a dark gray background indicate fixtures that are not selected. It is the default color combination.

Yellow:

Name	FID	IDType	CID
QuantPro 1	1	Fixture	

Yellow names and numbers on a medium gray background

Yellow fixture names and IDs on a medium gray background indicate fixtures that are selected.

Dim Orange:



Name	FID	IDType	CID
QuantPro 1	1	Fixture	

Orange names and numbers on a medium gray background

Orange fixture names and IDs on a medium gray background indicate fixtures that are in the current selection, but not the current sub-selection. For example, multiple fixtures are selected and **Next** has been pressed.

Green:

Name	FID	IDType	CID
QuantPro 1	1	Fixture	

Green names and numbers on a medium gray background

Green fixture names and IDs on a medium gray background indicate fixtures with inverted encoder control of at least one attribute, based on selection and MAtricks settings. For more information about the MAtricks, see the **MAtricks and Shuffle** topic.

Light Green:

Name	FID	IDType	CID
QuantPro 1	1	Fixture	
QuantPro 2	2	Fixture	
QuantPro 3	3	Fixture	

Fixture 2 appears with light green name and number on a medium gray background

Light green fixture names and IDs on a medium gray background indicate edge fixtures, based on selection and MAtricks settings. Edge fixtures do not follow all values. For example, when creating a mirrored circle with three fixtures and two wings, the edge fixture in the center will only tilt, but not pan. This is done in order to keep the symmetry. For more information about the MAtricks, see the **MAtricks and Shuffle** topic.

Fading between Yellow and Light Red:



Name	FID	IDType	CID
QuantPro 1	1	Fixture	

Light red names and numbers on a medium gray background

Fixture names and IDs fading between yellow and light red on a medium gray background indicate at least one multipatch instance of this primary fixture is selected. For more information about the multipatch fixtures, see the [Add Multipatch Fixtures](#) topic.

Sheet Colors for Absolute Attribute Values

The cells showing attribute values in the fixture sheet and sequence sheet, with the track sheet mode enabled, have several combinations of text and background colors.

Black:

Name	FID	IDType	CID	RGB			
				R	G	B	W
QuantPro 1	1	Fixture		100	100	100	

Black background for on the W attribute

There is no object to be edited. In this case, this fixture does not include a white color mixing attribute.

Light gray text with a gray background:

Name	FID	IDType	CID	RGB			
				R	G	B	W
QuantPro 1	1	Fixture		100	100	100	

Light gray text with a gray background on R, G, and B attributes

The values of these attributes are at their default levels.

Red text with gray background:



Name	FID	IDType	CID	RGB			
				R	G	B	W
QuantPro 1	1	Fixture		100	100	100	

Red text with gray background on R, G, and B attributes

These attributes have values in the programmer, but they are not active. By default, they will not be included in any store or update actions.

White text with a red background:

Name	FID	IDType	CID	RGB			
				R	G	B	W
QuantPro 1	1	Fixture		100	100	100	

White text with a red background on R, G, and B attributes

These attributes have active values in the programmer. By default, they will be included in any store or update actions.

Cyan text with a gray background:

Name	FID	IDType	CID	RGB			
				R	G	B	W
QuantPro 1	1	Fixture		100	100	100	

Cyan text with a gray background on R, G, and B attributes

These attributes have new values in the current cue outputting from the selected sequence. A dimmer value displayed with cyan text indicates a new, higher value in the current cue outputting from the selected sequence.

Green text with a gray background:



Name	FID	IDType	CID	Dimmer	PanTilt		R
				Dim	P	T	
QuantPro 1	1	Fixture		50	50	50	

Green text with a gray background on R, G, and B attributes

A dimmer value displayed with green text indicates a new, lower value in the current cue outputting from the selected sequence. This color is only used for descending dimmer values.

Magenta text with a gray background:

Name	FID	IDType	CID	RGB			
				R	G	B	W
QuantPro 1	1	Fixture		100	100	100	

Magenta text with a gray background on R, G, and B attributes

These attributes have tracked values in the current cue outputting from the selected sequence.

White text with a gray background:

Name	FID	IDType	CID	RGB			
				R	G	B	W
QuantPro 1	1	Fixture		100	100	100	

White text with a gray background on R, G, and B attributes

These attributes have blocked values in the current cue outputting from the selected sequence.

Yellow text with a gray background:

Name	FID	IDType	CID	RGB			
				R	G	B	W
QuantPro 1	1	Fixture		100	100	100	

Yellow text with a gray background on R, G, and B attributes



These attributes have new or blocked values in the current cue outputting from a sequence that is not selected. This color combination does not appear in the sequence sheet.

Black text with a deep sea green background:

Name	FID	IDType	CID	RGB			W
				R	G	B	
QuantPro 1	1	Fixture		100	100	100	

Black text with a deep sea green background on R, G, and B attributes

These attributes have completed a move in black. For more information on move in black, see the **Move in black** topic.

Black text with a sea green background:

Name	FID	IDType	CID	RGB			W
				R	G	B	
QuantPro 1	1	Fixture		41	41	41	

Black text with a sea green background on R, G, and B attributes

These attributes are currently fading through a move in black command. This color combination does not appear in the sequence sheet. For more information on move in black, see the **Move in black** topic.

Dim yellow text with a gray background:

Name	FID	IDType	CID	RGB			W
				R	G	B	
QuantPro 1	1	Fixture		100	100	100	

Dim yellow text with a gray background on R, G, and B attributes

These attributes have tracked values in the current cue outputting from a sequence that is not selected. This color combination does not appear in the sequence sheet.

Dark blue background:



Name	FID	IDType	CID	Dimmer	PanTilt	
				Dim	P	T
QuantPro 1	1	Fixture		912.2	Fan 2.2	Fan
QuantPro 2	2	Fixture		92.2	Fan 2.2	Fan

Various colors of text with a dark blue background on Dim, P, and T attributes

These attributes are currently fading through cue transitions. Text colors continue to specify the source of the data. This background color combination does not appear in the sequence sheet.

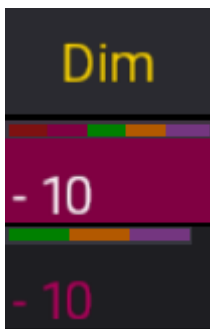
Sheet Colors for Additional Attribute Layers

Along with the absolute value, other layers of values may influence the output of an attribute. The color of the values from these layers matches the colors in the layer control bar in the encoder toolbar. For more information on the encoder toolbar, see the [Encoder Toolbar](#) topic.

The combinations of text and background colors for the additional layers all use the following pattern:

- White text on a background of the layer's color shows a value in the programmer.
- The text of the layer's color on a gray background shows a value from the current cue in the selected sequence.
- Yellow text on a gray background shows a value from a cue outputting from a sequence that is not selected.

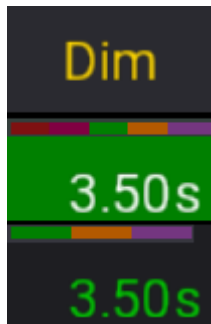
Mauve background or text:



Mauve background and text.

There is a relative value on the dimmers of these two fixtures. The top value with a mauve background comes from the programmer. The bottom value with mauve text comes from the current cue in a selected sequence.

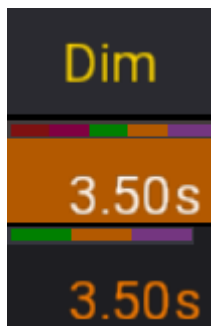
Green background or text:



Green background and text.

There is an individual fade time on the dimmers of these two fixtures. The top value with a green background comes from the programmer. The bottom value with green text comes from the current cue in a selected sequence.

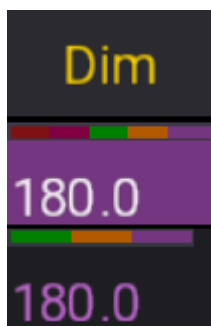
Orange background or text:



Orange background and text.

There is an individual delay time on the dimmers of these two fixtures. The top value with an orange background comes from the programmer. The bottom value with orange text comes from the current cue in a selected sequence.

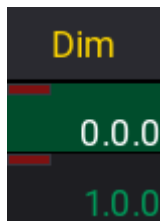
Purple background or text:



Purple background and text.

There is a phaser value on the dimmers of these two fixtures. All layers of phaser values share the same purple color. The top value with a purple background comes from the programmer. The bottom value with purple text comes from the current cue in a selected sequence.

Shamrock background or text:



Shamrock background and text.

There is a GridPos value on the dimmers of these two fixtures. Both values come from the programmer. The top value with a shamrock background shows an active GridPos value. The bottom value with shamrock text shows a deactivated GridPos value.




Hint:


Darkened versions of any of the above programmer colors indicate data from a programmer part, which is not the current part shown in the sheet.

For more information about programmer parts, see the [What is the Programmer](#) topic.



7.12.2. Markers

 **Important:**
 The colors described here are included in the default color theme.
 For more information about alternate color themes, see the **Color theme** topic.

 **Hint:**
MA + **MA** + **Clear** activates the default color theme and resets all values of that theme to the manufacturer defaults.

Markers are colored indicators in the form of a bar or a thin line. Markers appear in:

- Fixture Sheet
- Sequence Sheet with the Track Sheet option enabled
- Feature group control bar
- Attribute encoder display

The name column in the fixture sheet, the feature group control bar, and the attribute encoder display all show color-coded markers indicating values, individual timing, and phasers in the programmer. Attribute columns in the fixture sheet show markers indicating values, individual timing, and phasers either in the programmer or currently outputting from a sequence. Attribute columns in the sequence sheet show markers indicating values, individual timing, and phasers stored within the displayed cues.

Marker colors:

Name	FID	IDType	CID	Dimmer Dim
QuantPro 1	1	Fixture		21.13 F
QuantPro 2	2	Fixture		21.13 F

Fixture sheet with markers on one fixture name and both dimmer attributes

Markers above the name and dimmer value of fixture 1 indicate active information in the programmer. Markers only above the dimmer value of fixture 2, but not above the name, indicate information output from a sequence.

- Red markers indicate active absolute programmer values.
- Mauve markers indicate active relative programmer values.
- Green markers indicate individual fade times.
- Orange markers indicate individual delay times.
- Pink markers indicate phaser values.



- Bright cyan markers indicate absolute values from a preset.
- Dark cyan markers indicate relative values from a preset.
- Blue markers indicate parked attributes.

Name	FID	IDType	CID	Dimmer Dim
QuantPro 1	1	Fixture		21.6 Rr
QuantPro 2	2	Fixture		21.6 Rr

Fixture sheet with white markers

White markers indicate information in the programmer, which will not be stored when storing active values.

Small, yellow markers when editing a preset:

M A		Edit: Red		Prog Only	Part Red	Readout <Natural>	S
		3 Fixtures Selected					
Name	FID	IDType	CID	RGB			
				R	G	B	W
QuantPro 1	1	Fixture		100	0	0	
Rush Par 41	41	Fixture		100	0	0	0
LED Step 51	51	Fixture		100	0	0	

Fixture sheet while editing a global preset

Small, yellow markers in the fixture sheet while editing a global preset indicate an attribute holding the global value for that attribute for all fixtures of the same type within the preset. For more information on global presets, see the [Presets](#) topic.

Markers above the IDType:

Name	FID	IDType	CID
QuantPro 1	1	Fixture	
QuantPro 2	2	Fixture	
QuantPro 3	3	Fixture	

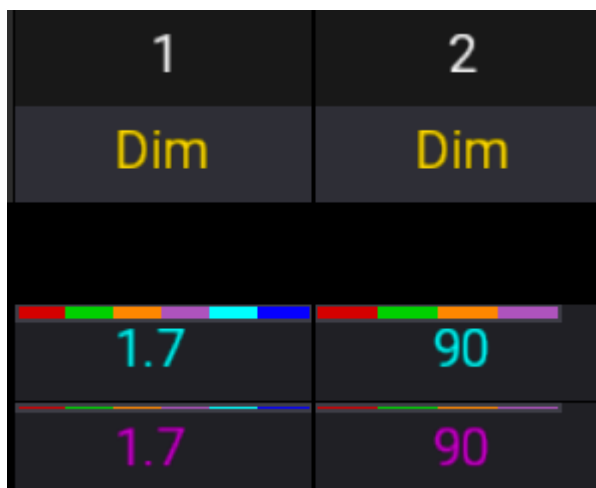
Fixture sheet with markers above the IDType.

Markers above the IDType in the fixture sheet indicate a group master is currently influencing the output of the marked fixture.

- Light pink markers indicate fixtures that are currently influenced by a Positive, Negative, or Scaling group master.
- Lavender markers indicate fixtures that are currently influenced by an Additive group master.
- Blue markers indicate fixtures with the Master React option set to "None."

For more information about group masters, see the **Group Masters** topic. For more information about the Master React option, see the **Add Fixtures to the Show** topic.

Markers in the Sequence Sheet with the Track Sheet option enabled:



The image shows a sequence sheet with two columns labeled '1' and '2'. Each column has a 'Dim' attribute. Below the attributes, there are two rows of values: '1.7' and '90'. Each value is accompanied by a horizontal bar with a multi-colored gradient (red, green, orange, purple, cyan, blue). The '1.7' values are shown in cyan, and the '90' values are shown in purple. The bars are positioned above the values, and thin lines are visible below the values, indicating tracked values.

Sequence sheet attribute columns with markers

Markers in the sequence sheet include an additional color and two different heights.

- Markers in the form of a bar indicate a new value stored in the cue.
- Markers in the form of a thin line indicate a tracked value.



7.12.3. Color Theme



Important:

Use caution when editing the color theme, as it is possible to make information on the screens difficult or impossible to read; for example, selecting the same color for both text and the background behind the text.



Hint:

MA + **MA** + **Clear** activates the default color theme and resets all values of that theme to the manufacturer defaults.

Color themes allow for customization of a wide variety of colors within the graphical user interface of the console. Alternate color themes can be activated quickly, changing the look of the interface on demand. Color themes can be exported to and imported from external sources. The console initially includes two color themes. The "default" color theme, which provides optimal readability when using the console under most lighting conditions. The "defaultDAYLIGHT" color theme creates a bolder interface for use in brighter conditions, such as daylight.



Activating Alternate Color Themes

Color themes can be imported using the menu.

1. Open the **Menu**.
2. Tap **Desk Lights & Color Theme** in the top-right corner.
3. Tap **Active Color Theme**. This button will also display the name of the current color theme. For example, 'default'.
The Import Color Theme pop-up opens.
4. Tap the desired color theme. If the color theme is on a USB drive, tap **Internal** in the title bar to select the drive.

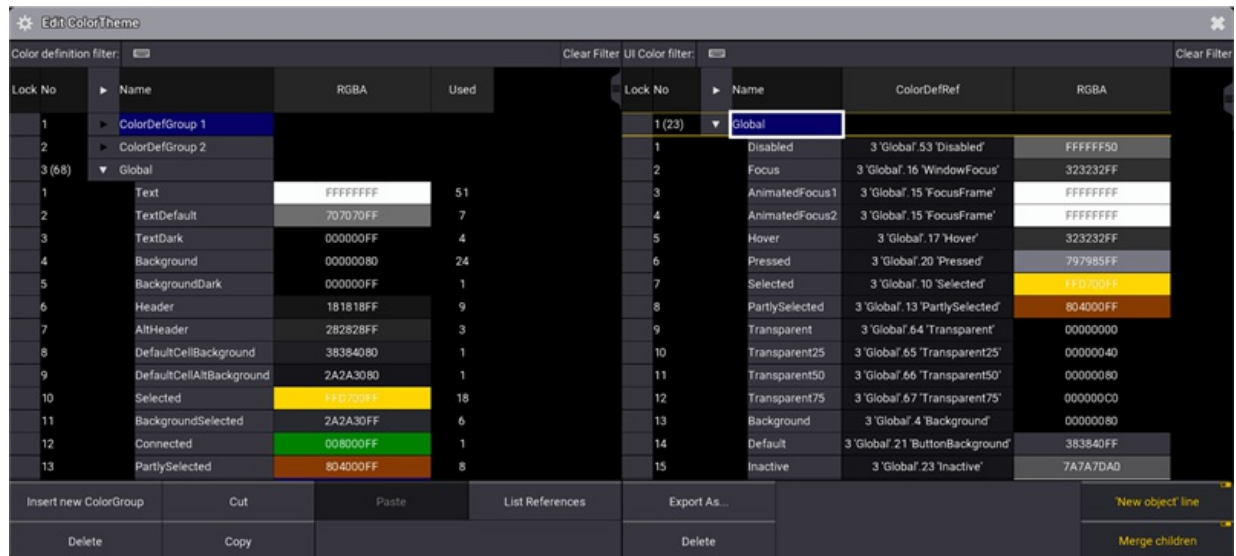
Color themes can also be imported using the command line. For example, use the following command to activate the "defaultDAYLIGHT" color theme:

```
MA User name[Fixture]>Import ColorTheme Library "defaultDAYLIGHT.xml" At  
ColorTheme /NoConfirmation
```



Editing the Current Color Theme

In the main menu, tap **Desk Lights & Color Theme** and then tap **Edit** in the bottom-right corner. The color theme editor opens:



Color theme editor

The left side of the color theme editor includes a list of color definitions. These function like color presets for user interface elements. Any changes to color definitions will be reflected in all of the UI elements referencing those definitions. This list shows the name of the color definition, the hex value of the color (including red, green, blue, and alpha values), and the number of UI elements referencing each color definition.

The right side of the color theme editor includes a list of user interface elements. This list shows the name of the UI color, a reference to a color definition, and the actual color defined by that reference. Changing the color definition reference to reference another color definition changes the color used for that element without altering any other elements. Deleting the reference for a UI element allows for a direct color definition for that element. This is analogous to storing a cue with hard values instead of referencing a preset.

Tap **Export As...** at the bottom of the menu to export the current color theme, either to the internal drive or to an external USB drive.

7.12.4. Compare Tool

The color theme compare tool is part of the Desk Lights & Color Theme menu.

The compare tool is a useful feature when it comes to:

- Comparing color themes with each other.
- Easily find distinctions when comparing color themes via color indicators and colored rows.
- Transfer colors between one color theme and another.
- Live color editing.

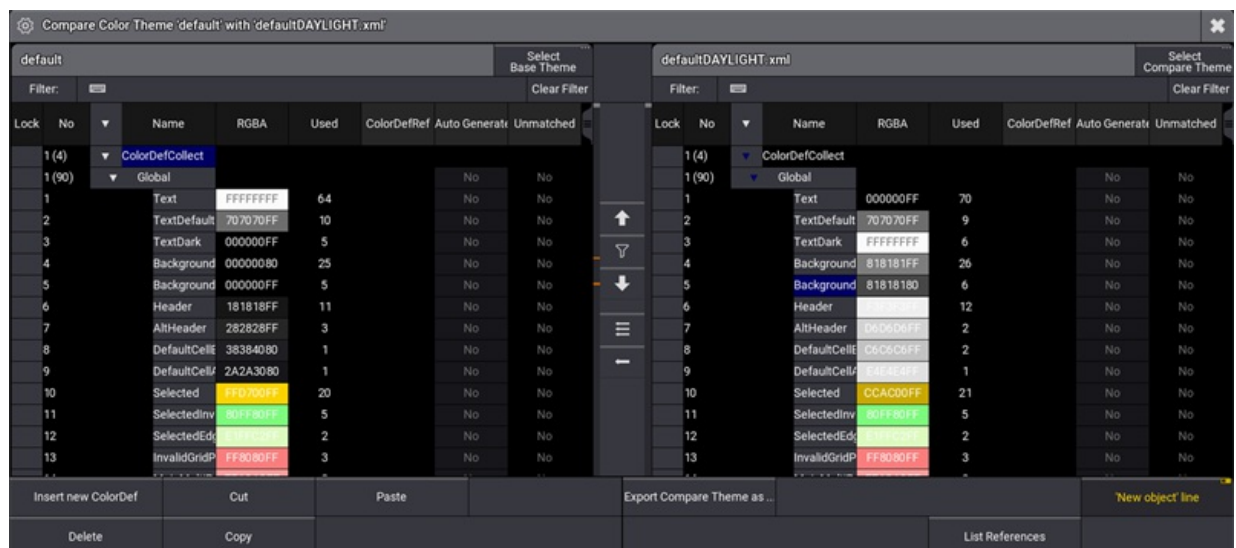
The color indicators and colored rows are defined as the following:

- **Orange:** This color exists in the base color theme, but is missing in the compared color theme, or vice versa.
- **Green:** This color was transferred from the base to the compared color theme, or vice versa.

For more information about color themes, see [Color Theme](#) topic.

To open the compare tool:








1. Open the **Menu**.
2. Tap **Desk Lights & Color Theme**.
3. Tap **Compare**. The compare tool opens. The currently active color theme will be taken as the base for comparison.
4. When opening the compare tool for the first time, a import pop-up opens. Select a color theme, for example defaultDAYLIGHT, and tap **Import**.



Compare tool is open

The different buttons of the compare tool are described below:



- **Filter:**
Filters the color name.
- **Clear Filter :**
Clears the filter.
- **Select Base Theme :**
Opens the import base color theme pop-up.
- **Select Compare Theme :**
Opens the import compare color theme pop-up.
-  (Arrow up):
Shows previous missing or merged color in the theme.
-  (Object Filter):
Shows only unmatched colors.
-  (Arrow down):
Shows next missing or merged color in the theme.
-  (Merge all left):
Merges all unmatched colors from the right into the left theme.
-  (Merge all right):
Merges all unmatched colors from the left into the right theme.
-  (Merge left):
Merges only selected unmatched colors from the left into the right theme.
-  (Merge right):
Merges only selected unmatched colors from the right into the left theme.



Hint:

To change the direction of the merge arrows, tap into the grid area of the specific color theme.

The grid is categorized in different columns:

- **Lock:**
When it is set to "Yes", no color values can be changed.



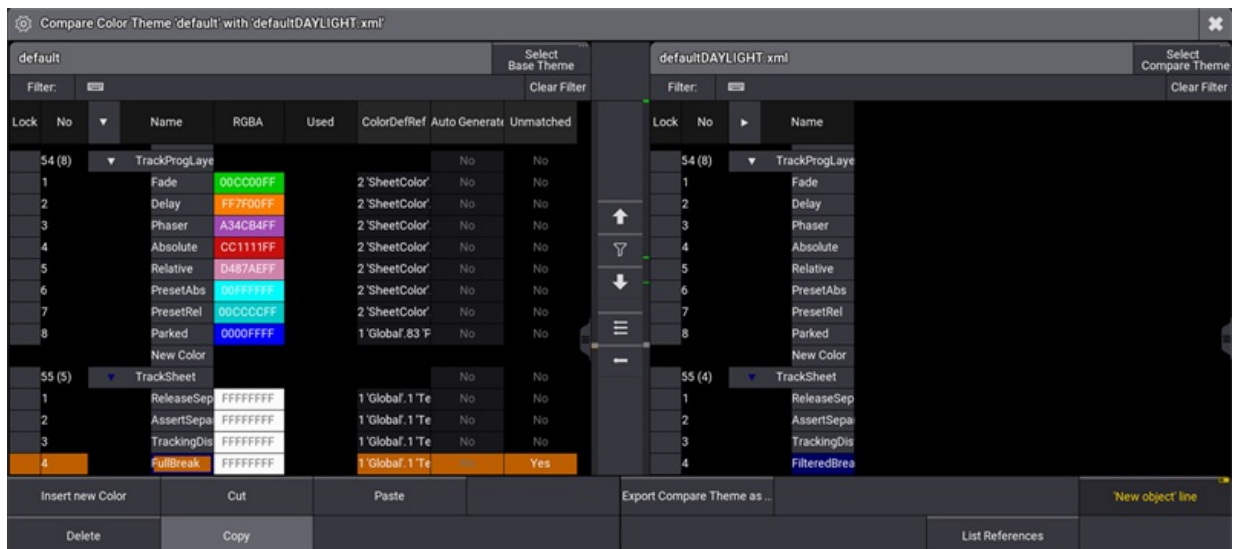
- **No:**
This is the color index number. Value in the bracket (x) shows the amount of colors in the particular structure tree.
- **Name:**
Color name.
- **RGBA:**
Shows the color definition.
- **Used:**
This is the amount of color related references.
- **ColorDefRef:**
See **Color Theme**.
- **Auto Generated:**
Shows "Yes" when a color is merged.
- **Unmatched:**
Is set to "Yes" when a color is not matching the compared theme.

Example for a Color Theme Merge

Requirements:

- Default is selected as base color theme.
- DefaultDAYLIGHT is selected as compare color theme.
- A color is deleted in the DefaultDAYLIGHT color theme. In the example below, FullBreak.

1. To quickly move the scroll bar to the next unmatched color in the color theme, tap . The unmatched color is marked in orange, as shown below:

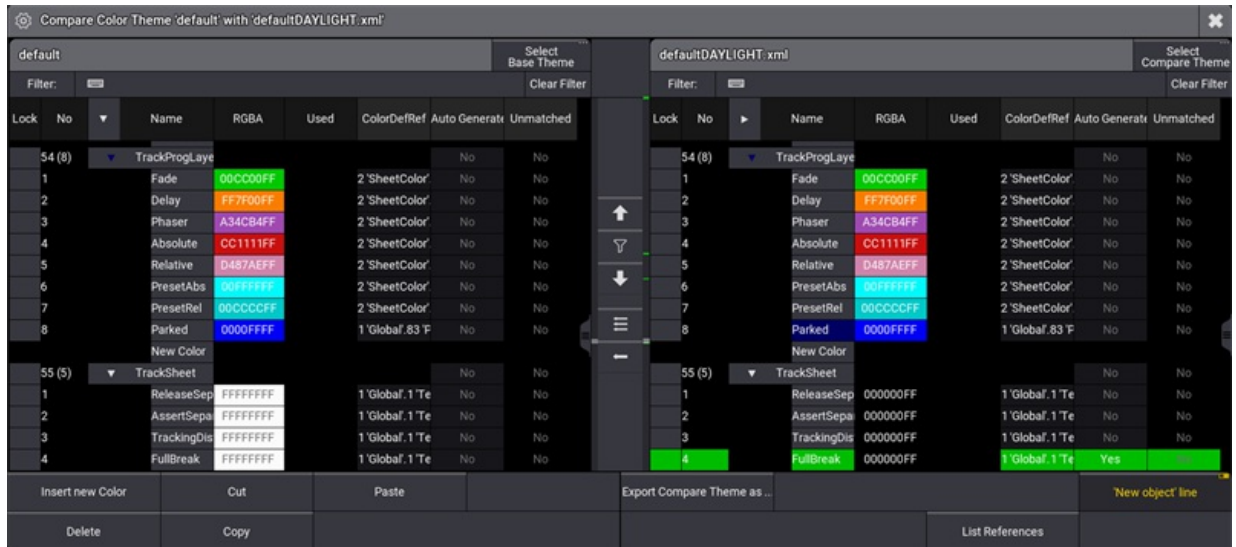


Unmatched colors are marked in orange.



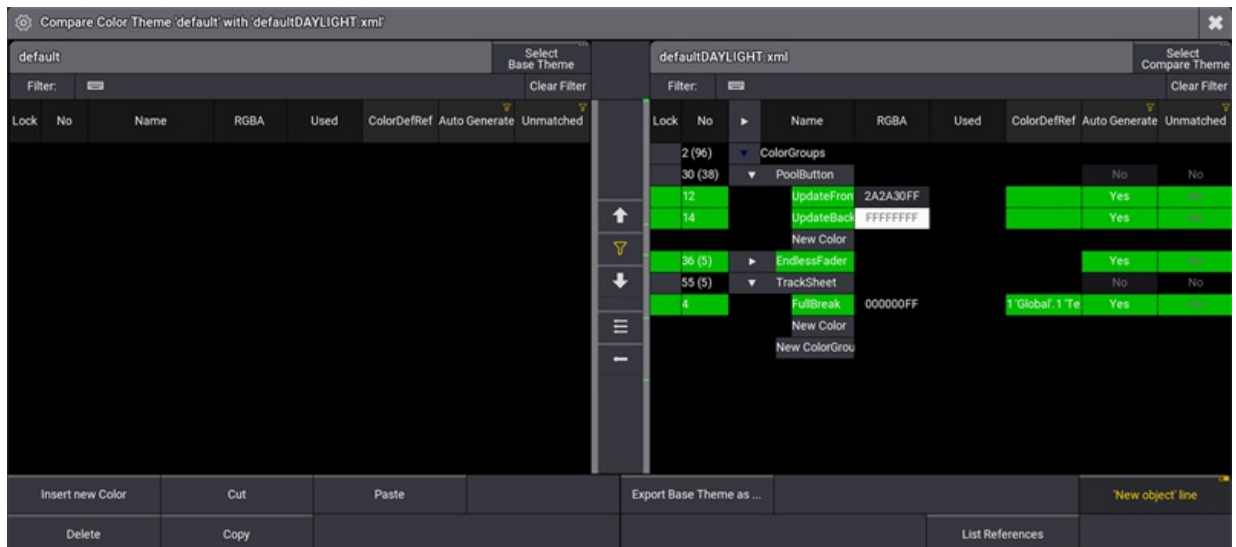
2. To merge all unmatched colors, tap

All merged colors are displayed in green rows. Notice that the column Auto Generated is set to Yes and the column Unmatched is set to No automatically in the defaultDAYLIGHT color theme as shown below:



Missing colors merged into defaultDAYLIGHT color theme are shown in green.

3. To show only Auto Generated and Unmatched colors, tap . When enabling filter, the button turns yellow and the columns names "Auto Generated" and "Unmatched" get a yellow filter icon as well:



Filter is active.

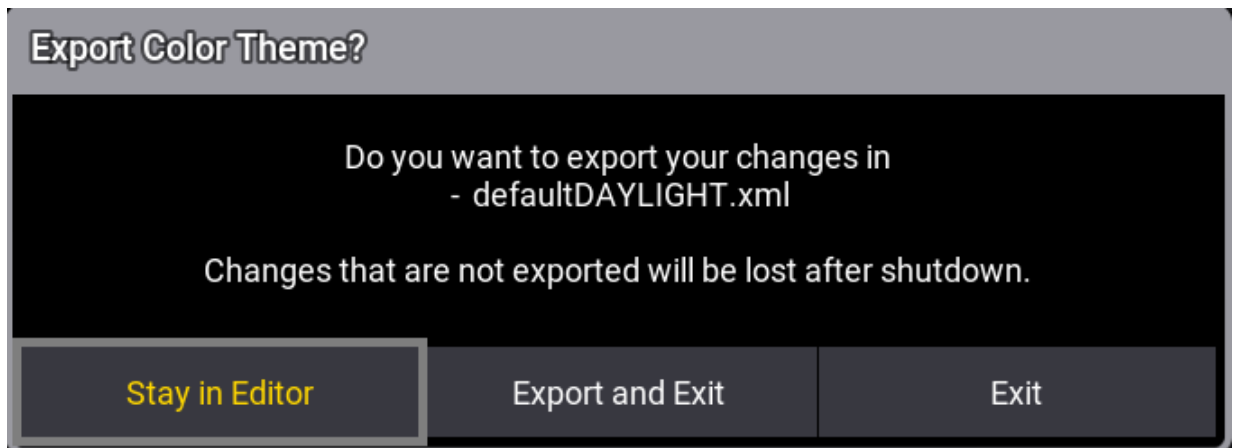
4. To change RGBA values for "FullBreak", right click the color in the RGBA column. The color editor opens.
 5. Change the R value to 20 by using the dynamic color scroll bar. Tap **Ok** to close the editor. The color value has changed:



Colors are editable in the color theme compare tool, too.

Export Color Theme

If closing the color editor without exporting the color theme first, the following pop-up will open:



Export color theme pop-up.

- **Stay in Editor** :
Pop-up closes and the color theme editor stays open.
- **Export and Exit** :
Changes are exported to the color theme listed above.
- **Exit** :
Closes the color theme editor. Changes will get lost after a shutdown.



To export the defaultDAYLIGHT.xml color theme as a new theme:

1. Tap into the right grid area of the compare tool to link the **Export** button to the corresponding defaultDAYLIGHT.xml file.
2. Tap **Export Compare Theme as ...**. The Export Compare Theme pop-up opens.
3. Type defaultDAYLIGHT_new.xml and then tap **Export**. The color theme is exported.

8. Function of the Command Line

The command line is an essential way of communication between the console and its operator.

Using keywords, special characters, and numerical identifiers is how the operator tells the console what to do.



Important:

Depending on the use case, some commands can be executed using the **Please key**.

Example

- To delete sequence 1, type:

```
MA User name[Fixture]> Delete Sequence 1
```

- Or use the keys in the command section of the console and press:

Delete + Sequ + 1 + Please

Keywords

The keywords that are mostly used have their own corresponding keys.

For more information on the multiple functions of the keys see **Keys**.



Important:

All keywords can be entered in the command line using the internal or any other connected keyboard.

To view the commands given to the console, tap **MA** in the left corner of the command line.

The **Command Line History** opens.

For more information on the usage of the command line in detail see **Command Line**.

8.1. Syntax Rules

The command line syntax is used to create valid commands.

Object keywords are used to allocate objects in the show file.

Help keywords are used to create a relation between functions and objects.

Playback keywords provide control over playback functionalities.

Fader keywords are related to anything that has to do with faders.

For more information see [Command Line History](#).

General Rules

The general rules are:



Important:

The basic syntax is as follows: **[Function] [Object]**

- All objects have a default function which is used if no function is given.
- Most functions have a default object or an object type which is used if no object is given.
- Objects are arranged in a hierarchical tree structure.
- If an object does not support the function applied, the function is passed on to a child or parent object.

Terminology

- [Square brackets]:
Description of non-literal content.
- (Parentheses):
Description of optional content.
- "Quotation marks"/'Quotation marks':
Quotation marks are used to enter a definite name or content. If the line ends after the word in quotation marks, the quotation marks at the end may be omitted. If the name or content is not a keyword and does not contain special characters, the quotation marks can be left out altogether.



Important:

If an option or any other part of the keyword command requires two types of quotation marks - single (') and double quotation marks (") - make sure to always use an equal pair. For more information see the [RemoteCommand keyword](#).

- **Capitalization:**

In general, capitalization is important. Only when using keyword commands, the console does not distinguish between upper case and lower case. In such topics, capitalization is used to improve readability only.

Use the Command Line

It is possible to abbreviate all the commands using the shortcuts of the corresponding keywords.

Each keyword has its own shortcut. Every keyword topic also names the respective shortcut.

For more information see [General Keywords](#).

Example

- To store cue 20 in sequence 8 using the overwrite function, type:

Full version of the syntax:

```
MA User name[Fixture]> Store Sequence 8 Cue 20 /Overwrite
```

Abbreviated version of the syntax:

```
MA User name[Fixture]> S Seq 8 Cue 20 /O
```

Very short version of the syntax:

- To copy cue 2 to cue 6 of the selected sequence, type:

```
MA User name[Fixture]> Co 2 At 6
```



Hint:

The examples use the full version in the manual.

8.2. Parent Child Concept

The general selection syntax is not only applying to fixtures (Parent) and their levels of sub-fixtures (Child), but furthermore the general selection syntax applies to all types of objects that use a hierarchic structure. Objects that use hierarchic structures are, for example:

- Preset Pools with Presets.
- Pages with Executors.
- Macro with Macro lines.



Hint:

To learn more about selecting fixtures and its sub-fixtures using syntax commands, see [Select Fixtures](#) topic.

To gain a faster and more logical way (for example to select, copy, move or delete objects) in the hierarchical structure, the dot (.) is an important tool in the selection syntax:

- The dot (.) ends the selection on the current level and then steps one level down in the hierarchical structure. Instead of using the dot (.) as a separator, dedicated keywords for the objects in the next level of the hierarchy can be used. For example, [Page 1 Executor 201](#).

To restart at the top level of the hierarchy within one command, the starting keyword (for example Fixture) has to be entered again. For example, [Fixture 5.2 Fixture 3](#).

To select fixtures in the same hierarchic structure within one command, (+) can be used. For example, [Fixture 5.2 + 3](#).



Dot Selection Behavior

It is not necessary to recall the subcategory of an object in a syntax command, after the subcategory is already called using the dot (.). This behavior is shown in the examples below.

Do not use the following command:

Clone Fixture 101 At Fixture 102 If Preset 21.101 Thru 21.105

Use a command with a omission of "21." instead:

Clone Fixture 101 At Fixture 102 If Preset 21.101 Thru 105

Do not use the following command:

Move Preset 1.1 Thru 1.3 At 1.5

Use a command with a omission of "1." instead:

Move Preset 1.1 Thru 3 At 1.5

Examples for Selection Syntax

To recreate the examples, the following should be prepared:

- The grandMA3 demo show file is loaded.
- A fixture sheet window is open.
- The following fixture type is patched:
 - Ayrton Alienpix - RS Ex 16 Bit
 - Quantity: 3
 - FID: 301 - 303



1. To select the second sub-sub-fixture of fixture 301.1, type:

```
MA User name[Fixture]> Fixture 301.1.2
```

Fixture Sheet: Absolute				Prog Only	Part Part Zero	Readout <Natural>	Step 1
Name	FID	IDType	CID	Dimmer Dim	PanTilt P T	Gobo G1 G1 <> G2	
LED Backwall	300	Fixture		1.2 Open			
APix 1	301	Fixture		0	0.00 0.00		
SubFixture	301.1	Fixture		100	0.00 0.00		
SubFix	301.1.1	Fixture		100			
SubFix	301.1.2	Fixture		100			
SubFix	301.1.3	Fixture		100			
SubFix	301.1.4	Fixture		100			
SubFix	301.1.5	Fixture		100			
SubFix	301.1.6	Fixture		100			
APix 2	302	Fixture		0	0.00 0.00		
SubFixture	302.1	Fixture		100	0.00 0.00		
SubFix	302.1.1	Fixture		100			
SubFix	302.1.2	Fixture		100			
SubFix	302.1.3	Fixture		100			
SubFix	302.1.4	Fixture		100			
SubFix	302.1.5	Fixture		100			
SubFix	302.1.6	Fixture		100			
APix 3	303	Fixture		0	0.00 0.00		
SubFixture	303.1	Fixture		100	0.00 0.00		
SubFix	303.1.1	Fixture		100			
SubFix	303.1.2	Fixture		100			
SubFix	303.1.3	Fixture		100			
SubFix	303.1.4	Fixture		100			
SubFix	303.1.5	Fixture		100			
SubFix	303.1.6	Fixture		100			
Univ		Universal	1	0	0.00 0.00		

Second sub-sub-fixture of fixture 301 is selected



2. To select only the sub-sub-fixtures 2 to 4 of fixtures 301 to 303, type:

```
MA User name[Fixture]> Fixture 301 Thru 303.1.2 Thru 4
```

In this example all fixtures on the top level have to be selected: **Fixture 301 Thru 303.1..**

And after the dot, the sub-fixtures are selected: **2 Thru 4.**

Fixture Sheet: Absolute				Prog Only	Part Part Zero	Readout <Natural>	Step 1
Name	FID	IDType	CID	Dimmer Dim	PanTilt P T		Gobo G1 G1<> G2
LED Backwall	300	Fixture		1.2 Open			
APix 1	301	Fixture		0	0.00	0.00	
SubFixture	301.1	Fixture		100	0.00	0.00	
SubFix	301.1.1	Fixture		100		0.00	
SubFix	301.1.2	Fixture		100		0.00	
SubFix	301.1.3	Fixture		100		0.00	
SubFix	301.1.4	Fixture		100		0.00	
SubFix	301.1.5	Fixture		100		0.00	
SubFix	301.1.6	Fixture		100		0.00	
APix 2	302	Fixture		0	0.00	0.00	
SubFixture	302.1	Fixture		100	0.00	0.00	
SubFix	302.1.1	Fixture		100		0.00	
SubFix	302.1.2	Fixture		100		0.00	
SubFix	302.1.3	Fixture		100		0.00	
SubFix	302.1.4	Fixture		100		0.00	
SubFix	302.1.5	Fixture		100		0.00	
SubFix	302.1.6	Fixture		100		0.00	
APix 3	303	Fixture		0	0.00	0.00	
SubFixture	303.1	Fixture		100	0.00	0.00	
SubFix	303.1.1	Fixture		100		0.00	
SubFix	303.1.2	Fixture		100		0.00	
SubFix	303.1.3	Fixture		100		0.00	
SubFix	303.1.4	Fixture		100		0.00	
SubFix	303.1.5	Fixture		100		0.00	
SubFix	303.1.6	Fixture		100		0.00	
Univ		Universal		1	0	0.00	0.00

Sub-sub-fixtures 2 to 4 are selected.



- To select sub-sub-fixtures 4 to 6 of fixture 301, all sub-fixtures of fixture 302, and sub-sub-fixtures 1 to 5 of fixture 303, type:

```
MA User name[Fixture]> Fixture 301.1.4 Thru 6 Fixture 302.1. Thru Fixture 303.1.  
Thru 5
```

In this example the . 4 in **Fixture 301.1.4** steps down to the sub-fixture level. **Fixture 302.1.** needs to be entered to restart at the top level with fixture 302.

If **Fixture 302** would not be entered, the second sub-fixture of sub-fixtures 4 to 6 would be selected.



M/A Fixture Sheet: Absolute				Prog Only	Part Part Zero	Readout <Natural>	Step 1	
▼	Name	FID	IDType	CID	Dimmer	PanTilt		Gobo
					Dim	P	T	G1 G1-> G2
	LED Backwall	300	Fixture		1.2 Open			
▼	APix 1	301	Fixture		0	0.00	0.00	
▼	SubFixture	301.1	Fixture		100	0.00	0.00	
	SubFix	301.1.1	Fixture		100			
	SubFix	301.1.2	Fixture		100		0.00	
	SubFix	301.1.3	Fixture		100		0.00	
	SubFix	301.1.4	Fixture		100		0.00	
	SubFix	301.1.5	Fixture		100		0.00	
	SubFix	301.1.6	Fixture		100		0.00	
▼	APix 2	302	Fixture		0	0.00	0.00	
▼	SubFixture	302.1	Fixture		100	0.00	0.00	
	SubFix	302.1.1	Fixture		100			
	SubFix	302.1.2	Fixture		100		0.00	
	SubFix	302.1.3	Fixture		100		0.00	
	SubFix	302.1.4	Fixture		100		0.00	
	SubFix	302.1.5	Fixture		100		0.00	
	SubFix	302.1.6	Fixture		100		0.00	
▼	APix 3	303	Fixture		0	0.00	0.00	
▼	SubFixture	303.1	Fixture		100	0.00	0.00	
	SubFix	303.1.1	Fixture		100			
	SubFix	303.1.2	Fixture		100		0.00	
	SubFix	303.1.3	Fixture		100		0.00	
	SubFix	303.1.4	Fixture		100		0.00	
	SubFix	303.1.5	Fixture		100		0.00	
	SubFix	303.1.6	Fixture		100		0.00	
	Univ		Universal	1	0	0.00	0.00	

Different sub-sub-fixtures are selected for each fixture type



- To select the fixtures as a replacement for a lasso selection, starting at fixture 302 and ending it at fixture 303.1 including the parent fixtures as a replacement for a lasso selection, type:

```
MA User name[Fixture]> Fixture 302. Fixture 303 Fixture 303.1
```

M/A Fixture Sheet: Absolute				Prog Only	Part Part Zero	Readout <Natural>	Step 1
Name	FID	IDType	CID	Dimmer Dim	PanTilt P T		Gobo G1 G1<> G2
LED Backwall	300	Fixture		1.2 Open			
APix 1	301	Fixture		0	0.00	0.00	
SubFixture	301.1	Fixture		100	0.00	0.00	
SubFix	301.1.1	Fixture		100		0.00	
SubFix	301.1.2	Fixture		100		0.00	
SubFix	301.1.3	Fixture		100		0.00	
SubFix	301.1.4	Fixture		100		0.00	
SubFix	301.1.5	Fixture		100		0.00	
SubFix	301.1.6	Fixture		100		0.00	
APix 2	302	Fixture		0	0.00	0.00	
SubFixture	302.1	Fixture		100	0.00	0.00	
SubFix	302.1.1	Fixture		100		0.00	
SubFix	302.1.2	Fixture		100		0.00	
SubFix	302.1.3	Fixture		100		0.00	
SubFix	302.1.4	Fixture		100		0.00	
SubFix	302.1.5	Fixture		100		0.00	
SubFix	302.1.6	Fixture		100		0.00	
APix 3	303	Fixture		0	0.00	0.00	
SubFixture	303.1	Fixture		100	0.00	0.00	
SubFix	303.1.1	Fixture		100		0.00	
SubFix	303.1.2	Fixture		100		0.00	
SubFix	303.1.3	Fixture		100		0.00	
SubFix	303.1.4	Fixture		100		0.00	
SubFix	303.1.5	Fixture		100		0.00	
SubFix	303.1.6	Fixture		100		0.00	
Univ		Universal		0	0.00	0.00	

This fixture selection is a replacement for a lasso selection.



5. To select all sub-fixtures on the first sub-fixture level for all fixtures, type:

```
MA User name[Fixture]> Fixture Thru.Thru
```

Deleting Syntax

- The following examples are used in general, and are not in correspondence with the demo show file.

To delete executor 201 on all pages, type:

```
MA User name[Fixture]> Delete Page Thru.201
```

or type:

```
MA User name[Fixture]> Delete Page Thru Executor 201
```

This second example describes the additional rule of the first rule from above: Instead of using Page x.y, the Executor keyword is used to address the executors.



8.3. General Keywords


Using keywords provides one of many possibilities to operate the grandMA3 console.

The following subtopics cover each of the available grandMA3 keywords. These are arranged in alphabetical order.

Each topic describes various means of entering the keyword into the command line, provides definition, proper syntax, and depicts examples of the keywords.



8.3.1. ; [Semicolon] Character

To enter the ; [Semicolon] in the command line, press .

Description


The semicolon separates multiple commands.

Syntax

[Command]; [Command]

Example

- To turn off sequence 5 and delete group 3, type:

```
 User name[Fixture]> Off Sequence 5; Delete Group 3
```



8.3.2. / [Slash] Character

To enter the / [Slash] in the command line, press .

Description

The slash character has two functions:

- It is used to set options during a command.
For more information see the **Option Keywords**.
- It is used as a separator in the calculator.



Important:

When adding option keywords to your commands, make sure to use the slash without spaces.

Syntax

[Command] /Option ["Option_Value"]


Example

- To store screen 2 on view button 2.7, type:

```
 User name[Fixture]> Store ViewButton 2.7 /Screen "2"
```



8.3.3. . [Dot] Character

To enter the . [Dot] in the command line, press .

Description

The dot is a separator which is used in fractional digits and hierarchic IDs of objects.



Important:

Make sure to use the dot without spaces.

Syntax

[Command] [Value].[Value_Fraction]

[Command] [Parent_Number].[Child_Number]

Examples

- To call the second color preset for fixture 31.2, type:

```
 User name[Fixture]> Fixture 31.2 At Preset 4.2
```

- To set the pan attribute to 50.5, type:

```
 User name[Fixture]> Attribute "Pan" At 50.5
```



8.3.4. = [Equal] Character

To enter the = [Equal] character in the command line, type =.

Description

The = [Equal] character is used when assigning a value to an object setting.

Many objects have different settings and these settings can be set using the setting name followed by the equal sign and then the desired value.

It is used in conjunction with the **Set keyword**.



Hint:

The equal character can be omitted in the command.
In case a command could be misinterpreted, use this character.

Syntax

[Function] [Object] ["Object_Name" or Object_Number] ["Setting"] = ["Option"]

Example


- To set the setting Enabled in macro line 1 of macro 3 to No, type:

```
MA User name[Fixture]> Set Macro 3.1 "Enabled" = "No"
```




8.3.5. <<< [GoFastBackward] Keyword

To enter the <<< [GoFastBackward] keyword in the command line, use one of the options:

- Press  <<<
- Type <<<
- Type the shortcut <

Description

The <<< [GoFastBackward] keyword is a playback keyword which is used to jump to the previous cue in a sequence, without using cue timing.


For more information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

<<< [Object] ["Object_Name" or Object_Number]

Examples

- To jump to the previous cue in sequence 3, type:

```
 User name[Fixture]> <<< Sequence 3
```

- To jump to the previous cue in the sequence assigned to executor 103 (on the current page), type:

```
 User name[Fixture]> <<< Executor 103
```

- To jump to the previous cue on all sequences assigned to executors on page 4, type:

```
 User name[Fixture]> <<< Page 4
```

For information on the key see the <<< [GoFastBackward] key.



8.3.6. >>> [GoFastForward] Keyword

To enter the >>> [GoFastForward] keyword in the command line, use one of the options:

- Press **>>>**
- Type >>>
- Type the shortcut >

Description

The >>> [GoFastForward] keyword is a playback keyword which is used to jump to the next cue in a sequence without using cue timing.

For more information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

>>> [Object] ["Object_Name" or Object_Number]

Examples

- To jump to the next cue in sequence 3, type:

```
MA User name[Fixture]> >>> Sequence 3
```

- To jump to the next cue in the sequence assigned on executor 103, type:

```
MA User name[Fixture]> >>> Executor 103
```

- To jump to the next cue on all sequences assigned to executors on page 4, type:

```
MA User name[Fixture]> >>> Page 4
```

- To jump to the next cue in the sequences assigned to executor 211 on page 5, type:

```
MA User name[Fixture]> >>> Page 5.211
```

For more information on the key see the [>>>](#) [GoFastForward] key.



8.3.7. - [Minus] Keyword

To enter the - [Minus] keyword in the command line, use one of the options:

- Press 
- Type - [Minus]

Description

The - [Minus] keyword is used to remove objects from a list or to indicate negative values.

If the - [Minus] keyword is used in order to indicate values, it will indicate absolute or relative values:

- A space between the - [Minus] and the value is automatically added. The space makes the value relative
- To obtain an absolute value, remove the space between the - [Minus]

Syntax

([Attribute] ["Attribute_Name" or Attribute_Number]) At - [Value]

[Object] ["Object_Name or Object_Number] - [Object] ["Object_Name" or Object_Number] [Number]

Examples

- To reduce the percentage readout of pan by 10 percent, type:

```
 User name[Fixture]> Attribute "Pan" At - 10
```


- To reduce 10 % from the current dimmer value in the selected fixtures, type:

```
 User name[Fixture]> At - 10
```

- To select the entire group 5 without selecting fixture 2, type:

```
 User name[Fixture]> Group 5 - Fixture 2
```

- To remove fixtures 5, 6, and 7 in the current selection of fixtures, type:

```
 User name[Fixture]> - Fixture 5 Thru 7
```





8.3.8. * [Asterisk] Character

To enter the * [Asterisk] in the command line, use one of the options:

- Press and hold **Shift** + **8** on the internal keyboard
- Press and hold **MA** + **/*** on the numeric keypad of the console

Description

The * [Asterisk] is a placeholder which is used to substitute any other character or characters in a name.

Syntax

[Command] "Name*"



Hint:

The * [Asterisk] can be positioned anywhere within the name string.

Examples

- To select the fixtures of all groups beginning with "Mac" in the group pool object name, type:

```
MA User name[Fixture]> Group "Mac*"
```

- To select all fixtures with a name beginning with "backt" and ending with "blue", type:

```
MA User name[Fixture]> Fixture "backt*blue"
```



8.3.9. % [Percent] Keyword

To enter the % [Percent] keyword in the command line, type %.

Description

The % [Percent] keyword is a helping keyword that can be used as a scaling operator.

Syntax

(Attribute ["Attribute_Name" or Attribute_Number]) At % [Value]

Examples

Requirement: Set the dimmer to 70 %

- To reduce the dimmer value from 70 % to 35 % in a selected fixture, type:

```
MA User name[Fixture]> At % 50
```


- To set the tilt to 90 % of the current value in a selected fixture, type:

```
MA User name[Fixture]> Attribute "Tilt" At % 90
```



8.3.10. + [Plus] Keyword

To enter the + [Plus] keyword in the command line, use one of the options:

- Press 
- Type +

Description

The + [Plus] keyword is a helping keyword that has various functions.

It is used to combine multiple objects in a list or to indicate relative values. When used as a relative indicator without a value, value 1 is used.

If used as a starting keyword, the + [Plus] keyword creates a selection list which is added to the current selection.


Syntax

(Attribute ["Attribute_Name" or Attribute_Number]) At + [Value]

[Function] [Object] ["Object_Name" or Object_Number] + ["Object_Name" or Object_Number]

Examples


- To delete cue 1 and 2 on the selected executor, type:

```
 User name[Fixture]> Delete Cue 1 + 2
```

- To add 5 % to the current dimmer value, type:

```
 User name[Fixture]> At + 5
```

- To add fixtures 5, 6 and 7 to the current selection, type:

```
 User name[Fixture]> + Fixture 5 Thru 7
```



8.3.11. Absolute Keyword

To enter the Absolute keyword in the command line, use one of the options:

- Press **MA** + **Time** + **Time** + **Time** + **Time**
- Type **Absolute**
- Type the shortcut **Ab**

Description

The Absolute keyword is used to address the absolute layer.

Syntax

Absolute

(Attribute ["Attribute_Name" or Attribute_Number]) At Absolute [Value]

Examples

- To set the layer to absolute, type:

```
MA User name[Fixture]> Absolute
```

- To set the color red to 10 in the programmer, type:

```
MA User name[Fixture]> Attribute "ColorRGB_R" At Absolute 10
```




8.3.12. Acceleration Keyword

To enter the Acceleration keyword in the command line, use one of the options:

- Type **Acceleration**
- Type the shortcut **Acc**

Description

The Acceleration keyword is used to set the acceleration curve of a step in the phaser.

For more information see [Phasers](#).

Syntax

Acceleration

(Attribute ["Attribute_Name" or Attribute_Number]) At Acceleration [Value]

Examples

- To set the layer to accelerate, type:

```
MA User name[Fixture]> Acceleration
```

- To set the acceleration curve of the dimmer to 70, type:

```
MA User name[Fixture]> At Acceleration 70
```



8.3.13. Action Keyword

To enter the Action keyword in the command line, use one of the options:

- Type **Action**
- Type the shortcut **Actio**

Description

The Action keyword is used to call functions that do not have a designated keyword.

Syntax

Action ["Function"]

Examples

- To store the pan/tilt position to calibration point 1 of the currently selected fixtures, type:

```
MA User name[Fixture]> Action "StoreCalibrationPoint1"
```

- To call the pan/tilt position of the currently selected fixture of calibration point 2 into the programmer, type:

```
MA User name[Fixture]> Action "CallCalibrationPoint2"
```

- To solve the stage calibration, type:

```
MA User name[Fixture]> Action "SolveCalibration"
```



8.3.14. ActivationGroup Keyword

To enter the ActivationGroup keyword in the command line, use one of the options:

- Type **ActivationGroup**
- Type the shortcut **AG**

Description

The ActivationGroup keyword represents attributes that are activated together.

For more information see [Activation Group](#).

Syntax

[Function] ActivationGroup

Example

- To list all activation groups, type:

```
MA User name[Fixture]> List ActivationGroup
```



8.3.15. Agenda Keyword

To enter the Agenda keyword in the command line, use one of the options:

- Type **Agenda**
- Type the shortcut **Age**

Description

The Agenda keyword represents the single agenda events.

For more information see [Agenda](#).

Syntax

[Function] Agenda ["Agenda_Name" or Agenda_Number] (/Option)

Option Keywords

The Agenda keyword uses the following option keywords:

- /Date

Examples

- To assign sequence 1 to the agenda event 1, type:

```
MA User name[Fixture]> Assign Sequence 1 At Agenda 1
```

- To give the first agenda event the name "Sunset", type:

```
MA User name[Fixture]> Label Agenda 1 "Sunset"
```

- To change the mode of the second agenda event to "Dawn", type:

```
MA User name[Fixture]> Set Agenda 2 Property "Mode" "Dawn"
```



8.3.16. Align Keyword

To enter the Align keyword in the command line, use one of the options:

- Press **Align**
- Type **Align**
- Type the shortcut **AI**

Description

The Align keyword is a function keyword which is used to toggle through the align modes of the attribute encoders.

- To toggle through the modes, repeatedly press **Align**.

Align Mode	Index Number
Off	0
/	1
<	2
>	3
><	4
<>	5

For more information on the align modes, see [Operate Fixtures - Align](#).

Syntax

Align ["Align_Mode" or Align_Index_Number]

Examples

- To set the encoders to the first align mode, type:

```
MA User name[Fixture]> Align "<"
```

- To disable the align mode altogether, type:

```
MA User name[Fixture]> Align 0
```



8.3.17. AlignTransition Keyword

To enter the AlignTransition keyword in the command line, use one of the options:

- Type **AlignTransition**
- Type the shortcut **AlignT**

Description

The AlignTransition keyword is used to toggle between the align transition modes:

AlignTransition Mode	Index Number
Linear	0
Sinus	1
Slow	2
Fast	3

Syntax

AlignTransition ["AlignTransition_Name" or AlignTransition_Index_Number]

AlignTransition [Function]

Examples

- To set the AlignTransition to Sinus, type:

```
MA User name[Fixture]> AlignTransition "Sinus"
```

- To toggle the AlignTransition to the previous align mode, type:

```
MA User name[Fixture]> AlignTransition Previous
```

- To toggle the AlignTransition to the next align mode, type:

```
MA User name[Fixture]> AlignTransition Next
```



8.3.18. Appearance Keyword

To enter the Appearance keyword in the command line, use one of the options:

- Press **MA** + **X4** + **X4**
- Type **Appearance**
- Type the shortcut **Ap**

Description

The Appearance keyword is used to assign an appearance to an object.

For more information see [Appearance](#).

Syntax

Assign Appearance ["Appearance_Name" or Appearance_Number] At [Object] ["Object_Name" or Object_Number]

Examples

- To assign appearance 14 to group 1, type:

```
MA User name[Fixture]> Assign Appearance 14 At Group 1
```

- To assign appearance "Sunset" to group 2, type:

```
MA User name[Fixture]> Assign Appearance "Sunset" At Group 2
```



8.3.19. Assign Keyword

To enter the Assign keyword in the command line, use one of the options:

- Press **Assign**
- Type **Assign**
- Type the shortcut **As**

Description

The Assign keyword is a function which is used to assign objects to other objects.

To set parameters for objects, use the **Set keyword**.

For more information see:

- **Assign macros to keys and buttons**
- **Create new presets**
- **Use appearances**
- **Store and recall views**

Syntax

Assign [Object] ["Object_Name" or Object_Number] At [Object] ["Object_Name" or Object_Number]

Option Keywords

The Assign keyword uses the following option keywords:

- **/Tab**

Examples

- To assign macro 2 to executor page 1 executor 402, type:

```
MA User name[Fixture]>Assign Macro 2 At Page 1.402
```

- To assign view 2 to display 2 and view button 3, type:

```
MA User name[Fixture]> Assign View 2 At ViewButton 2.3
```




- To assign appearance 1 to group 5, type:

```
MA User name[Fixture]> Assign Appearance 1 At Group 5
```

- To mirror sequence 1 in sequence 2, type:

```
MA User name[Fixture]> Assign Sequence 1 At Sequence 2
```



Hint:

In terms of playback, mirrored sequences are independent from one another. However, their content is linked.

8.3.20. At Keyword

To enter the At keyword in the command line, use one of the options:

- Press **At**
- Type **At**
- Type the shortcut **A**

Description

The At keyword is a function keyword and a helping keyword at once.

- As a function keyword it is used to apply values.
- As a helping keyword it is used along with other function keywords to indicate destination.



Hint:

At applies values live in the programmer. For information on how to apply values throughout the show file see the [Clone keyword](#).

At is "the exception that proves the rule". At is one of the few functional keywords which accept objects before the function.

As a starting keyword, At is a function that applies values in the programmer to the current selection.

If value type Fade or Delay is used, the value list will be applied as individual fade/delay times.

Following an object list, At is a function that applies values to the object list. If the object list does not support the At function, the object list is resolved into a selection list and At applies values in the programmer.

Following an object list that follows a function, At is a helping keyword for the starting function.

General Syntax

At ([Value_Type]) [Value]

At [Object] ["Object_Name" or Object_Number]

[Object] ["Object_Name" or Object_Number] At ([Value_Type]) [Value]

[Object] ["Object_Name" or Object_Number] At [Object] ["Object_Name" or Object_Number]

[Destination] At [Source] (If [Object] ["Object_Name" or Object_Number]) (/Option)

Syntax as a Helping keyword

[Function] [Object] ["Object_Name" or Object_Number] At [Object] ["Object_Name" or Object_Number]

Examples

- To set the MasterFader of the sequence 1 to 30 %, type:

```
MA User name[Fixture]> FaderMaster Sequence 1 At 30
```

- To set the dimmer attributes of the current selection to 75%, type:

```
MA User name[Fixture]> At 75
```

- To set the fixture selection to the values of cue 3 in the selected sequence, type:

```
MA User name[Fixture]> At Cue 3
```

- To set the pan attributes of the selected fixtures to 20%, type:

```
MA User name[Fixture]> Attribute "Pan" At 20
```

- To set an individual delay time of 2 seconds to attribute 2, type:

```
MA User name[Fixture]> Attribute 2 At Delay 2
```

- To copy group 4 to group 10, type:

```
MA User name[Fixture]> Copy Group 4 At 10
```

- To set a speed to 60 using the speed readout specified in the user profile (for example, BPM), type:

```
MA User name[Fixture]> At Speed 60
```

8.3.21. Attribute Keyword

To enter the Attribute keyword in the command line, use one of the options:

- Press **Preset** **Preset**
- Type **Attribute**
- Type the shortcut **Att**

Description

The Attribute keyword is an object keyword which is used to set attributes of a fixture.

The default function of attributes is Call. If you call an attribute, you can use the encoders to modify the values. Calling attributes also selects the attributes in the fixture sheet.



Important:

The number of an attribute may vary if new fixtures and attributes are added to the show file. We recommend you use the unique library name of attributes.



Hint:

- Attributes are organized by subattributes;
- Subattributes are organized by features;
- Features are organized by feature groups.

Syntax

[Function] Attribute ["Attribute_Name" or Attribute_Number]

Examples

- To view the list of attributes along with their corresponding names and numbers in the command line history, type:

```
MA User name[Fixture]> List Attribute
```

- To set the attribute "pan" to 120 degrees in the selected fixtures, type:

```
MA User name[Fixture]> Attribute "pan" At 120
```



- To knock out the first attribute, which is Dimmer, in the current selection, type:

```
MA User name[Fixture]> Off Attribute 1
```

Requirement: Load the demo show and select fixture 1.

- To set the random strobe (channel function 4) of the selected fixture to a value of 4 Hz, type:

```
MA Username[Fixture]>Attribute "Shutter1 StrobeRandom" At 4
```

8.3.22. AutoCreate Keyword

To enter the AutoCreate keyword in the command line, use one of the options:

- Type **AutoCreate**
- Type the shortcut **Au** or **Ac**

Description

The AutoCreate keyword creates objects depending on predefined source objects. It can, for example, create groups in the fixture types of the patched fixtures.



Hint:

If your selection is defined as source list, the active MAticks settings will be taken into consideration whenever executing AutoCreate.

Syntax

AutoCreate [Source_Object] ["Source_Object_Name" or Source_Object_Number] At [Destination_Object] ["Destination_Object_Name" or Destination_Object_Number] (/Option)

Option Keywords

The AutoCreate keyword uses the following option keywords:

- **/All**
- **/Merge**
- **/Overwrite**
- **/Single**

Examples

- To create single fixture groups starting with group pool object 1 using all selected fixtures, type:

```
 User name[Fixture]> AutoCreate Selection At Group 1
```

- To create single fixture groups for fixtures 1 to 10 starting with group pool object 1, type:

```
 User name[Fixture]> AutoCreate Fixture 1 Thru 10 At Group 1
```

Requirement:

1. Insert at least ten fixture types in the show file and patch at least two fixtures of every fixture type.
- To create a group in the group pool object 42 containing all fixtures of fixture type 10, type:

```
MA User name[Fixture]> AutoCreate FixtureType 10 At Group 42
```

Requirement:

1. Create layers and classes within the patch.
2. Set fixtures to these layers and classes.



Hint:

The demo show already uses these settings.

- To create a group in the group pool object 301 using all patched fixtures that are set to class "Spots" within the patch, type:

```
MA User name[Fixture]> AutoCreate FixtureClass "Spots" At Group 301
```

- To create single fixture groups starting with group pool object 401 using all fixtures of fixture type 9 that are also set to fixture class "Spots", type:

```
MA User name[Fixture]> AutoCreate FixtureType 9 + 10 If FixtureClass "Spots"
```



8.3.23. Black Keyword

To enter the Black keyword in the command line, use one of the options:

- Press **MA** + **<<<**
- Type **Black**
- Type the shortcut **Bla**

Description

Black is a playback keyword that is used to temporarily override the master level to zero on executing objects.

For more information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

Black (On or Off) [Object] ["Object_Name" or Object_Number]

Examples

- To override the master level of executor 201 to zero, type:

```
MA User name[Fixture]> Black On Executor 201
```

- To return the master level of executor 201 to the master fader, type:

```
MA User name[Fixture]> Black Off Executor 201
```




8.3.24. Blind Keyword

To enter the Blind keyword in the command line, use one of the options:

- Press **Blind**
- Type **Blind**
- Type the shortcut **BI**

Description

The Blind keyword is used to suppress the output of the live programmer. It is possible to program without putting the content live. Disabling Blind activates the programmer and the changes made during the mode.

Use the [DMX sheet](#) to see that the output is set to 0.



Hint:

Blind is a toggle function which means that entering Blind without using a helping keyword enables or disables Blind.

Syntax

Blind (On or Off)

Example

- To enable Blind and to disable the output of the programmer in return, type:

```
 User name[Fixture]> Blind
```



8.3.25. Block Keyword

To enter the Block keyword in the command line, use one of the options:

- Type **Block**
- Type the shortcut **Blo**

Description

Block is a function used to add data and prevent them from tracking. Tracked values are converted to stored values.

If the object list does not contain any references to any cues, the Block function is applied to the selected sequence.

If syntax does not contain any selection list filter, all fixtures will be used.

If syntax does not contain any attribute list filter, all attributes will be used.

Syntax

Block (Sequence ["Sequence_Name" or Sequence_Number]) Cue ["Cue_Name" or Cue_Number] (If Fixture ["Fixture_Name" or Fixture_Number] FeatureGroup ["FeatureGroup_Name" or FeatureGroup_Number] or Attribute ["Attribute_Name" or Attribute_Number] EndIf)

Examples

- To block all parameters in cue 2 of the selected sequence, type:

```
MA User name[Fixture]> Block Cue 2
```

To unblock parameters, use the **Unblock keyword**.

- To block pan and tilt of fixture 4 in cue 5 of the selected sequence, type:

```
MA User name[Fixture]> Block Cue 5 If Fixture 4 FeatureGroup "Position" EndIf
```



8.3.26. BPM Keyword

To enter the BPM keyword in the command line, use one of the options:

- Type **BPM**
- Type the shortcut **Bp**

Description

The BPM keyword is used to set the speed of a fixture selection or a speed master using the unit BPM.

Syntax

At Speed BPM [Value]

Master [Master_Number] At BPM [Value]

Examples

- To set the speed layer to 5 bpm, type:

```
MA User name[Fixture]> At Speed BPM 5
```

- To set the first speed master to 75 bpm, type:

```
MA User name[Fixture]> Master 3.1 At BPM 75
```



8.3.27. Call Keyword

To enter the Call keyword in the command line, use one of the options:

- Press **On On**
- Type **Call**
- Type the shortcut **Cal**

Description

The Call keyword is used to apply an object or its content.

Syntax

Call [Object] ["Object_Name" or Object_Number] (/Option)

Option Keywords

The Call keyword uses the following option keywords:

- **/Screen**

Examples

- To call view button 2.1, type:

```
MA User name[Fixture]> Call ViewButton 2.1
```

- To call macro 2, type:

```
MA User name[Fixture]> Call Macro 2
```



8.3.28. CancelSoftwareUpdate Keyword

To enter the CancelSoftwareUpdate keyword in the command line, use one of the options:

- Type **CancelSoftwareUpdate**
- Type the shortcut **Can**

Description

The CancelUpdateSoftware keyword is a function keyword which is used to cancel the running process of sending grandMA3 update files to other grandMA3 devices in the network.


Syntax

CancelSoftwareUpdate

Example

Requirement:

- Device must be in the network;
 - Device must send update files to other stations.
-
- To cancel the running process of sending update files to other stations, type:

```
 User name[Fixture]> CancelSoftwareUpdate
```



8.3.29. Camera Keyword

To enter this keyword into the console, use one of the options:

- Type **Camera**
- Type the shortcut **Cam**

Description

The Camera keyword selects a camera in the camera pool.

Syntax

([Function]) Camera ["Camera_Name" or Camera_Number]

Examples

- To display the auto camera of the 3D, type:

```
MA User name[Fixture]> Camera 1
```

- To select the front camera of the 3D, type:

```
MA User name[Fixture]> Camera "Front"
```

8.3.30. Capture Keyword

To enter the Capture keyword in the command line, use one of the following options:

- Press **Stomp** **Stomp**
- Type **Capture**
- Type the shortcut **Cap**

Description

The Capture keyword is a command keyword which is used to activate the current output values of specified parameters.

Capturing parameters during a running phaser or cue fade activates the actual values at the moment the command is executed. The result is a single step of static values resembling a freeze frame of the output.



Important:

The capture command produces only numeric values, losing any existing preset references.



Important:

The capture command translates the output of each parameter into the appropriate value on the absolute layer while activating a 0 value on the relative layer. For any parameter where the output was previously composed of a combination of absolute and relative values, the output will appear the same while relying only on the absolute layer with no relative offset.

Syntax

Capture [Object] ["Object_Name" or Object_Number]

Examples

- To capture all attributes of the selected fixtures, type:

```
MA User name[Fixture]> Capture
```

- To capture only the position attributes of the selected fixtures, type:

```
MA User name[Fixture]> Capture FeatureGroup "Position"
```



8.3.31. ChangeDestination Keyword

To enter the ChangeDestination keyword in the command line, use one of the options:

- Type **ChangeDestination**
- Type the shortcut **CD** or **Chang**

Description

The ChangeDestination keyword is a function keyword used to change the current destination of the command line.

Syntax

ChangeDestination ["Element_Name" or Element_Number]

ChangeDestination Root

ChangeDestination ..

Examples

- To enter the first element of the current destination, type:

```
MA User name[Fixture]> ChangeDestination 1
```

Result:

```
MA User name@MessageCenter>
```

- To enter the element of the current destination called "Sequence", type:

```
MA User name[Fixture]> ChangeDestination "Sequence"
```




- To leave the destination "Sequence", type:

```
MA User name@Menus> ChangeDestination Root
```

Result:

```
MA User name[Fixture]>
```

- To go one level back in the tree structure, type:

```
MA User name@MessageCenter/Undefined> ChangeDestination ..
```

Result:

```
MA User name@MessageCenter>
```



8.3.32. Channel Keyword

To enter the Channel keyword in the command line, use one of the options:

- Press **Channel**
- Type **Channel**
- Type the shortcut **Chann**
- Type **F + 1**

Description

The Channel keyword is an object keyword used to select fixtures of the ID type Channel.

Use the Channel keyword as a default keyword for the command line when programming with channels. For more information see [Workspace - Command Line - Change the Default Keyword](#).

Syntax

Channel ["Channel_Name" or Channel_Number]

Example

- To select channel ID 10, type:





8.3.33. ChannelSet Keyword

To enter the ChannelSet keyword in the command line, use one of the options:

- Type **ChannelSet**
- Type the shortcut **ChannelS**

Description

The ChannelSet keyword is used to take channel sets, such as gobos, actively into the programmer.

Syntax

At ChannelSet ["ChannelSet_Name" or ChannelSet_Number]

Example

- To activate the values of channel set 4 in the programmer, type:

```
MA User name[Fixture]> At ChannelSet 4
```



8.3.34. Chat Keyword

To enter the Chat keyword in the command line, use one of the options:

- Type **Chat**

Description

The Chat keyword is used to send messages via the world server in chat channels that were already joined. These chat messages are displayed in the message center.

Syntax

Chat "Text"

Example

Requirement: Join a chat channel using the **ChatJoin Keyword**

- To send the text "Cyan is a great color" as a chat message to all devices that joined the same chat channels, type:

```
MA User name[Fixture]> Chat "Cyan is a great color"
```



8.3.35. ChatJoin Keyword

To enter the ChatJoin keyword in the command line, use one of the options:

- Type **ChatJoin**
- Type the shortcut **ChatJ**

Description

The ChatJoin keyword is used to join a chat channel.

Syntax

ChatJoin ["ChatChannel_Name"]

Example

- To join the chat channel "Color", type:

```
MA User name[Fixture]> ChatJoin "Color"
```



8.3.36. ChatLeave Keyword

To enter the ChatLeave keyword in the command line, use one of the options:

- Type **ChatLeave**
- Type the shortcut **ChatL**

Description

The ChatLeave keyword is used to leave a chat channel.

Syntax

ChatLeave ["ChatChannel_Name"]

Example

Requirement: Join a chat channel first

- To leave the chat channel "Color", type:

```
MA User name[Fixture]> ChatLeave "Color"
```



8.3.37. CleanUp Keyword

To enter the CleanUp keyword in the command line, use one of the options:

- Type **CleanUp**
- Or type the shortcut **Clean**

Description

The CleanUp keyword is a command keyword that is used to delete objects that are not used and do not contain references in the show file. For example, you can clean up sequences that are not assigned to an executor.

Syntax

CleanUp [Object] ["Object_Name" or Object_Number] ((Thru or *) / (/Option))

Option Keywords

The CleanUp keyword uses the following option keywords:

- **/Selective**

Example

- To delete all unassigned sequences, type:

```
MA User name[Fixture]> CleanUp Sequence Thru
```

- To delete all unused color presets in preset pool 4, type:

```
MA User name[Fixture]> CleanUp Preset 4.*
```

- To delete all images without reference, type:

```
MA User name[Fixture]> CleanUp Image 3.1 Thru
```



8.3.38. Clear Keyword

To enter the Clear keyword in the command line, use one of the options:

- Press **Clear**
- Type **Clear**
- Type the shortcut **CL**

Description

The Clear keyword is a function keyword which is used to clear the selection, active values, or the programmer.

Depending on the status of the programmer the Clear keyword successively:

1. Clears selection (deselects all fixtures);
2. Clears active values (deactivates all values);
3. Clears all (empties the entire programmer).

For information on the key see the [Clear Key](#).

For more information on each function of the Clear keyword see [ClearSelection](#), [ClearActive](#), [ClearAll](#).

Syntax

Clear

Example

- To clear the selection, active values or the programmer depending on the status or the content of the programmer, type:

```
MA User name[Fixture]> Clear
```




8.3.39. ClearActive Keyword

To enter the ClearActive keyword in the command line, use one of the options:

- Type **ClearActive**
- Type the shortcut **Ca**

Description

The ClearActive keyword is used to deactivate all values in the programmer.

For information on the key see the [Clear Key](#).

For more information on the additional functions of the Clear keyword see [Clear](#), [ClearAll](#), [ClearSelection](#).

Syntax

ClearActive

Example

- To deactivate active values in the programmer, type:

```
MA User name[Fixture]> ClearActive
```



8.3.40. ClearAll Keyword

To enter the ClearAll keyword in the command line, use one of the options:

- Type **ClearAll**
- Type the shortcut **ClearA**

Description

The ClearAll function clears the selection and discards all values in the programmer.

For information on the key see the [Clear Key](#).

For more information on the additional functions of the Clear keyword see [Clear](#), [ClearActive](#), [ClearSelection](#).

Syntax

ClearAll

Example

- To clear the programmer, type:

```
MA User name[Fixture]> ClearAll
```



8.3.41. ClearSelection Keyword

To enter the ClearSelection keyword in the command line, use one of these options:

- Type **ClearSelection**
- Type the shortcut **Cs**

Description

The ClearSelection keyword is a function keyword which is used to deselect the selected fixtures.

For information on the key see the [Clear Key](#).

For more information on the additional functions of the Clear keyword see [Clear](#), [ClearActive](#), [ClearAll](#).

Syntax

ClearSelection

Example

- To deselect selected fixtures, type:

```
 User name[Fixture]> ClearSelection
```

8.3.42. Clone Keyword

To enter the Clone keyword in the command line, use one of the options:

- Press **MA** + **X1**
- Type **Clone**
- Type **Clo**

Description

Clone replicates data from one fixture or selection to another throughout the show file. For more information on applying values live in the programmer only, see the [At keyword](#).



Important:

If no object type is specified, using the keyword **If**, Clone only adds data within the programmer. Adding the keyword **If** changes the scope of cloning from the programmer to other objects.

Syntax

Clone [Object] ["Object_Name" or Object_Number] At [Object] ["Object_Name" or Object_Number] (If [Object] ["Object_Name" or Object_Number] /Option)

Option Keywords

The Clone keyword uses the following option keywords:

- **/MergeLowPriority**
- **/MergeHighPriority**
- **/NoDependencies**
- **/Overwrite**

Examples

- To clone the data from fixture 1 to fixture 2 in the programmer, type:

```
MA User name[Fixture]> Clone Fixture 1 At Fixture 2
```

The Clone pop-up appears, offering the choices of **Low Priority**, **High Priority**, **Overwrite**, and **Cancel**.

- To clone data from fixtures 1 and 2 to fixtures in group 10 within sequence 1 thru 10 only, type:

```
MA User name[Fixture]> Clone Fixture 1 + 2 At Group 10 If Sequence 1 Thru 10
```



- To clone all objects within data pool 1 from fixtures 1 thru 12 to fixtures 101 thru 112, type:

```
MA User name[Fixture]> Clone Fixture 1 Thru 12 At Fixture 101 Thru 112 If  
DataPool 1
```

For more information on data pools, see the [Data Pools](#).

- To clone the pan attribute from fixture 1 to fixture 2, type:

```
MA User name[Fixture]> Clone Fixture 1 Attribute "Pan" At Fixture 2
```



Hint:

When cloning preset and cue data, all presets referenced by the destination objects are automatically cloned in addition.



Hint:

The syntax:

Clone [Source_Object] ["Object_Name" or Object_Number] At [Destination_Object] ["Object_Name" or Object_Number] (/Option)

produces the same results as:

[Destination_Object] ["Object_Name" or Object_Number] At [Source_Object] ["Object_Name" or Object_Number] (/Option)

All options described above work the same for both syntax structures. When cloning without the **Clone** keyword, keep in mind that the source and destination change places within the syntax.

For more information on the **At** keyword, see the [At keyword](#).



8.3.43. CommandDelay Keyword

To enter the CommandDelay keyword in the command line, use one of the options:

- Type **CommandDelay**
- Type the shortcut **CommandD**

Description

The CommandDelay sets the delay time when executing commands in a cue.

Syntax

CommandDelay [CommandDelay_Value]

([Function]) (Sequence ["Sequence_Name" or Sequence_Number]) Cue ["Cue_Name" or Cue_Number]
CommandDelay [CommandDelay_Value]

Example

- To create cue 3 and set the cue's command delay to 4 seconds, type:

```
MA User name[Fixture]> Store Cue 3 CommandDelay 4
```

8.3.44. Collect Keyword

To enter the Collect keyword in the command line, use one of the options:

- Press **Select** **Select**
- Type **Collect**
- Type the shortcut **Coll**

Description

The Collect keyword is used to add objects of the same type to a collection. For more information on collection see [Collection Keyword](#).



Hint:

It is possible to collect empty pool objects during import or when storing.

Syntax

Collect [Object] ["Object_Name" or Object_Number]

Examples

- To add the second preset of the color preset pool to the collection, type:

```
MA User name[Fixture]> Collect Preset 4.2
```

- To add a different color preset to the collection, type:

```
MA User name[Fixture]> Collect Preset 4.10
```

- To reset a collection, type:

```
MA User name[Fixture]> Collect
```

8.3.45. Collection Keyword

To enter the Collection keyword in the command line, use one of the options:

- Press **Fixture** + **Fixture** + **Fixture**
- Type **Collection**
- Type the shortcut **Collecti**

Description

The Collection keyword is an object keyword used to represent the current collection of objects.



Important:

A collection stays always at the same spot it was collected at.



Hint:

Pool objects that are part of a collection have an orange frame around them. They also display the order of the collection which is located on top of a pool object.

Syntax

[Function] Collection At [Object] ["Object_Name" or Object_Number]

Examples

Requirement:

Create a collection of color presets

- To move the collection to the twentieth color preset, type:



```
User name[Fixture]> Move Collection At Preset 4.20
```




8.3.46. Color Keyword

To enter the Color keyword in the command line, type **Color**.

Description

The Color keyword is an object keyword which is used, for example, to adjust the color of the color theme to your liking.

Syntax

Set Color ["ColorGroup_Name"."Text"] Property ["Property_Name" "PropertyReadout"]

Examples

- To set the color of the color theme to green, type:

```
MA User name[Fixture]> Set Color "Global"."Text" Property "RGBA" "00FF00FF"
```



8.3.47. ColorDefinition Keyword

To enter the ColorDefinition keyword in the command line, use one of the options:

- Type **ColorDefinition**
- Type the shortcut **ColorD**

Description

The ColorDefinition keyword is an object keyword which is used, for example, to adjust the color definition of the color theme to your liking.

Syntax

[Function] ColorDefinition ["ColorGroup_Name"."Name"] (Property ["Property_Name"] ["Property_Value"])

Examples

- To set the color definition to white, type:

```
MA User name[Fixture]> Set ColorDefinition "Global"."Text" Property "RGBA"  
"FFFFFFF"
```



8.3.48. ColorTheme Keyword

To enter the ColorTheme keyword in the command line, use one of the options:

- Type **ColorTheme**
- Type the shortcut **Col**

Description

ColorTheme is an object keyword which is used, for example, to adjust the color theme.

Syntax

[Function] ColorTheme

Examples

- To list all the elements of the color theme, type:

```
MA User name[Fixture]> List ColorTheme
```

- To enter the ColorTheme folder in the data structure, type:

```
MA User name[Fixture]> ChangeDestination ColorTheme
```



8.3.49. Configuration Keyword

To enter the Configuration keyword in the command line, use one of the options:

- Type **Configuration**
- Type the shortcut **Con**

Description


A configuration contains a set of different button, key, fader, and encoder functions. It can be assigned to an executor.

Syntax

[Function] Configuration ["Configuration_Name or Configuration_Number] At [Object] ["Object_Name" or Object_Number]

Example

To assign configuration 2 to executor 101, type:

```
 User name[Fixture]> Assign Configuration 2 At Executor 101
```



8.3.50. Console Keyword

To enter the Console keyword in the command line, use one of the options:

- Type **Console**
- Type the shortcut **Cons**

Description

The Console keyword is an object keyword which is used to address all grandMA3 consoles in the network.

This keyword belongs to a session-related group of keywords:

- **NetworkNode keyword**
- **onPC keyword**
- **ProcessingUnit keyword**
- **Station keyword**
- **Extension keyword**
- **Join keyword**
- **Leave keyword**
- **Invite keyword**
- **Dismiss keyword**
- **Reboot keyword**
- **Restart keyword**
- **ShutDown keyword**

Syntax

[Function] Console ["Console_Name" or Console_Number]

Examples

- To list all consoles that are in the same network, type:

```
MA User name[Fixture]> List Console
```

- To invite console "FOH" to the session, type:

```
MA User name[Fixture]> Invite Console "FOH"
```



8.3.51. Cook Keyword

To enter the Cook keyword in the command line, use one of the options:

- Press **MA** + **Update**
- Type **Cook**

Description

The Cook keyword is a command keyword which is used to concoct recipes of objects without accessing the editor.

For more information see [Recipes](#).

Syntax

Cook [Object] ["Object_Name" or Object_Number] (/Option)

Option Keywords

The Cook keyword uses the following option keywords:

- **/Merge**
- **/Overwrite**
- **/Remove**

Example

- To cook the recipes of the first dimmer preset, type:

```
MA User name[Fixture]> Cook Preset 1.1
```

For more information on Recipes, see [Cue Recipes](#).



8.3.52. Copy Keyword

To enter the Copy keyword in the command line, use one of these options:

- Press **Copy**
- Type **Copy**
- Type the shortcut **Co**

Description

The Copy keyword is a function keyword which is used to create copies of an object.

If no object type is given and the command line destination is root (no destination), the default object type – **Cue** – is used for this function.

Syntax

Copy [Object] ["Source_Name" or Source_Number] At ["Destination_Name" or Destination_Number] (/Option)

Copy [Object] ["Source_Name" or Source_Number] (/Option)

Option Keywords

The Copy keyword uses the following option keywords:

- **/CopyCueDestination**
- **/CopyCueSource**
- **/Default**
- **/Merge**
- **/NoConfirmation**
- **/Overwrite**
- **/Release**

Examples

- To copy group 1 to group 5, type:

```
MA User name[Fixture]> Copy Group 1 At 5
```

- To copy group 1 to group 11; group 2 to group 12; and group 3 to group 13, type:

```
MA User name[Fixture]> Copy Group 1 Thru 3 At 11
```



- To copy group 2 to group 6, 7, and 8, type:

```
MA User name[Fixture]> Copy Group 2 At 6 Thru 8
```

- To copy cue 2 to cue 6 of the selected sequence, type:

```
MA User name[Fixture]> Copy Cue 2 At 6
```

- To copy macro 2 to macro 6, type:

```
MA User name[Fixture]> Copy Macro 2 At 6
```


8.3.53. Cue Keyword

To enter the Cue keyword in the command line, use one of the options:

- Press **Cue**
- Type **Cue**

Description



Important:

Cue is the only object type that accepts numerical ID as decimal fractions. The ID allowed for cues ranges from 0.001 to 9999.999. In all other objects, a dot indicates the ID of a parent or a child object.

Cue is an object type holding a look on stage.

Cues are arranged in sequences and are divided into parts.

For more information see the [Cue and Sequence section](#).

The **Cue** keyword is an object keyword. Object keywords must have a function keyword in front of them to create a complete command. For more information on how to use the command syntax, see [General Syntax Rules](#).

Cue has a default function called [SelfFix](#). This selects all the fixtures that have stored values in the cue.

If a sequence is not specified, then the selected sequence is used in the command.

Syntax

[Function] Cue ["Cue_Name" or Cue_Number]

[Function] Sequence ["Sequence_Name" or Sequence_Number] Cue ["Cue_Name" or Cue_Number] ([Setting] ["Setting_Value"]) (/Option)

Assign [Object] ["Object_Name" or Object_Number] At (Sequence ["Sequence_Name" or Sequence_Number]) Cue ["Cue_Name" or Cue_Number]

Settings

In some settings you have to assign an object. These can be assigned using the [Assign keyword](#).

Other settings may contain a text option or a value. Use the [Set keyword](#) for these settings.

Cues can also contain settings for MAtricks and recipe values. These can be changed using the [Set Keyword](#).

Option Keywords

The Cue keyword uses the following option keywords:



- /Active
- /ActiveForSelected
- /AddNewContent
- /All
- /AllForSelected
- /Ask
- /CopyCueDestiation
- /CopyCueSource
- /CreateSecondCue
- /CueOnly
- /Default
- /DMX
- /GridMergeMode
- /Look
- /Merge
- /NoConfirmation
- /Output
- /Overwrite
- /Programmer
- /Release
- /Remove

Examples

- To select the fixtures with values stored in cue 3 of a selected sequence, type:

```
MA User name[Fixture]> Cue 3
```

The default function for **Cue** is **SelfFix** so **Cue 3** is the same as **SelfFix Cue 3**.

- To delete cue 2.5 in the selected sequence, type:

```
MA User name[Fixture]> Delete Cue 2.5
```

- To store cue 2 in sequence 5, type:

```
MA User name[Fixture]> Store Sequence 5 Cue 2
```



- To store cue 3 in sequence 5 with a cue fade time of 7 seconds and an outfade of 11, type:

```
MA User name[Fixture]> Store Sequence 5 Cue 3 CueFade 7/11
```



8.3.54. CueAbsolute Keyword

To enter the CueAbsolute keyword in the command line, use one of the options:

- Type **CueAbsolute**
- Type the shortcut **CueA**

Description

CueAbsolute is a layer keyword. It is used to change the selected layer to the CueAbsolute layer. This layer can show information in, for instance, the fixture sheet. The layer shows the cue ID number for each parameter that has an absolute value from the active cues.

If a fixture sheet is set to **Auto** layer, then it will automatically change to show the selected layer.

A cue is composed of **[Sequence_Number].[Cue_Number];[Part_Index]**

Syntax

CueAbsolute

Examples

- To select the CueAbsolute layer, type:

```
MA User name[Fixture]> CueAbsolute
```



8.3.55. CueDelay Keyword

To enter the CueDelay keyword in the command line, use one of the options:

- Press **Time Time** (If the Time Key Target is set to Cue. For more information see [User Settings](#) and [Time Key](#).)
- Type **CueDelay**
- Type the shortcut **Cued**

Description

CueDelay can set both the indelay and the outdelay time of a cue. To do so, use a /. See examples further down.

For more information on how to set the delay times in objects see the [Delay Keyword](#).

Syntax

(Cue ["Cue_Name" or Cue_Number]) CueDelay [CueDelay_Time]

(Cue ["Cue_Name" or Cue_Number]) CueDelay [CueInDelay_Time]/[CueOutDelay_Time]

Examples

- To enter a delay of 5 seconds in the current cue of the selected sequence, type:

```
MA User name[Fixture]> CueDelay 5
```

- To set an indelay of 6 seconds and an outdelay of 12 seconds in the current cue of the selected sequence, type:

```
MA User name[Fixture]> CueDelay 6/12
```

- To adjust the CueInDelay to 3 seconds, but leave the CueOutDelay as it was, type:

```
MA User name[Fixture]> CueDelay 3/
```

- To enter a delay of 5 seconds in cues 1 to 4 of the selected sequence, type:

```
MA User name[Fixture]> Cue 1 Thru 4 CueDelay 5
```





8.3.56. CueFade Keyword

To enter the CueFade keyword in the command line, use one of the options:

- Press **Time** (If the Time Key Target is set to Cue. For more information see [User Settings](#) and [Time Key](#).)
- Type **CueFade**
- Type **Cuef**

Description

CueFade can set both the infade and the outfade time of a cue. To do so, use a /. See examples further down.

As a helping keyword for programming functions (for example Store), this keyword sets the fade time of a cue or a cue part.

Syntax

(Cue ["Cue_Name" or Cue_Number]) CueFade [CueFade_Time]

(Cue ["Cue_Name" or Cue_Number]) CueFade [CueInFade_Time]/[CueOutFade_Time]

Examples

- To enter a fade of 5 seconds in a cue, type:

```
MA User name[Fixture]> CueFade 5
```

- To set an infade of 6 seconds and an outfade of 12 seconds in the current cue of the selected sequence, type:

```
MA User name[Fixture]> CueFade 6/12
```

- To adjust the CueInFade to 3 seconds, but leave the CueOutFade as it was, type:

```
MA User name[Fixture]> CueFade 3/
```

- To enter a fade of 5 seconds in cues 1 to 4, type:

```
MA User name[Fixture]> Cue 1 Thru 4 CueFade 5
```



- To enter a fade time of 1 hour 22 minutes 56.3 seconds in cue 1 of the selected sequence, type:

```
MA User name[Fixture]> Cue 1 CueFade 1h22m56.3
```

Or press:

```
Time 1 . . . 2 2 . . 5 6 . 3
```

- To double the CueFade time, type:

```
MA User name[Fixture]> CueFade * 2
```

- To subtract 3 seconds from the CueFade time, type:

```
MA User name[Fixture]> CueFade - 3
```




8.3.57. CueInDelay Keyword

To enter the CueInDelay keyword in the command line, use one of the options:

- Type **CueInDelay**
- Type the shortcut **Cueind**

Description

CueInDelay sets the delay time of a cue.

Syntax

(Cue ["Cue_Name or Cue_Number]) CueInDelay [CueInDelay_Time]

Examples

- To enter an indelay of 5 seconds in the current cue of the selected sequence, type:

```
MA User name[Fixture]> CueInDelay 5
```

- To enter an indelay of 5 seconds in cues 1 to 4 of the selected sequence, type:

```
MA User name[Fixture]> Cue 1 Thru 4 CueInDelay 5
```



8.3.58. CueInFade Keyword

To enter the CueInFade keyword in the command line, use one of the options:

- Type **CueInFade**
- Type the shortcut **Cuein**

Description

CueInFade sets the infade time of a cue.

Syntax

(Cue ["Cue_Name" or Cue_Number]) CueInFade [CueInFade_Time]

Examples

- To enter an infade of 5 seconds in the current cue of the selected sequence, type:

```
MA User name[Fixture]> CueInFade 5
```

8.3.59. CueOutDelay Keyword

To enter the CueOutDelay keyword in the command line, use one of the options:

- Type **CueOutDelay**
- Type the shortcut **Cueoutd**

Description

CueOutDelay sets the outdelay time of a cue.



Hint:

CueOutDelay is only used by dimmer parameters that go to a lower value in the cue.

Syntax

(Cue ["Cue_Name" or Cue_Number]) CueOutDelay [CueOutDelay_Time]

Examples

- To enter an outdelay of 5 seconds in the current cue of the selected sequence, type:

```
MA User name[Fixture]> CueOutDelay 5
```

- To enter an outdelay of 5 seconds in cues 1 to 4 of the selected sequence, type:

```
MA User name[Fixture]> Cue 1 Thru 4 CueOutDelay 5
```

8.3.60. CueOutFade Keyword

To enter the CueOutFade keyword in the command line, use one of the options:

- Type **CueOutFade**
- Type the shortcut **Cueo**

Description

CueOutFade sets the outfade time of a cue.



Hint:

CueOutFade is only used by dimmer parameters that go to a lower value in the cue.

(Cue ["Cue_Name" or Cue_Number]) CueOutFade [CueOutFade_Time]

Examples

- To enter an outfade of 5 seconds in the current cue of the selected sequence, type:

```
MA User name[Fixture]> CueOutFade 5
```

- To enter an outfade of 5 seconds in cues 1 to 4 of the selected sequence, type:

```
MA User name[Fixture]> Cue 1 Thru 4 CueOutFade 5
```



8.3.61. CueRelative Keyword

To enter the CueRelative keyword in the command line, use one of the options:

- Type **CueRelative**
- Type the shortcut **CueR**

Description

CueRelative is a layer keyword. It is used to change the selected layer to the CueRelative layer. This layer can show information in, for instance, the fixture sheet. The layer shows the cue ID number for each parameter that has a relative value from the active cues.

If a fixture sheet is set to **Auto** layer, then it will automatically change to show the selected layer.

A cue is composed of **[Sequence_Number].[Cue_Number]:[Part_Index]**

Syntax

CueRelative

Example

- To select the CueRelative layer, type:

```
MA User name[Fixture]> CueRelative
```



8.3.62. CueUpdate Keyword

To enter the CueUpdate keyword in the command line, use one of the options:

- Type **CueUpdate**
- Type the shortcut **CueU**

Description

CueUpdate keyword is an object keyword which addresses all cues that can be updated.

Syntax

[Function] CueUpdate [CueUpdate_Number]

Examples

- To list all the cues that can be updated in the command line feedback, type:

```
MA User name[Fixture]> List CueUpdate
```

It is the same list shown in the [Update menu](#).



Important:

The number relates to the first element in the list. It does not represent the cue number.

- To update the first cue in the list, type:

```
MA User name[Fixture]> Update CueUpdate 1
```



8.3.63. CurrentEnvironment Keyword

To enter the CurrentEnvironment keyword in the command line, use one of the options:

- Type **CurrentEnvironment**
- Type the shortcut **CurEnv**

Description

The CurrentEnvironment keyword addresses the currently selected environment of the current user - Live or Preview.

Syntax

[Function] CurrentEnvironment (Property ["Property_Name"] ["Value"])

Example

- To set the align mode to >< (butterfly) in the current environment, type:

```
 User name[Fixture]> Set CurrentEnvironment Property "AlignMode" "><"
```



8.3.64. CurrentUser Keyword

To enter the CurrentUser keyword in the command line, use one of the options:

- Type **CurrentUser**
- Type the shortcut **Cu**

Description

CurrentUser represents the user that is currently logged in.

Syntax

[Function] CurrentUser (["Property"] ["Value"])

Examples

- To list all properties and their values of the current user, type:

```
MA User name[Fixture]> List CurrentUser
```

- To set the right of the current user to admin, type:

```
MA User name[Fixture]> Set CurrentUser "Rights" "Admin"
```




8.3.65. CurrentUserProfile Keyword

To enter the CurrentUserProfile keyword in the command line, use one of the options:

- Type **CurrentUserProfile**
- Type the shortcut **CUP**
- Or type the shortcut **CurrentUserP**

Description

The CurrentUserProfile keyword represents the profile of the current user.

Syntax

[Function] CurrentUserProfile ("Property") ["Value"]

Examples

- To set single step in the current user profile, type:

```
MA User name[Fixture]> Set CurrentUserProfile "SingleStep" "Yes"
```

- To set the wheel resolution to normal in the current user profile, type:

```
MA User name[Fixture]> Set CurrentUserProfile "Wheelresolution" "Normal"
```



8.3.66. Cut Keyword

To enter the Cut keyword in the command line, type **Cut**.

Description

The Cut keyword is a function keyword used to specify the source objects for a two-step action.

Syntax

Cut [Object] ["Object_Name" or Object_Number]

The object is temporarily stored for later use as source object for the following Paste command.

For more information on Cut & Paste see the [Paste Keyword](#).

Example

- To prepare the color preset 1 to be moved using the Paste keyword, type:

```
MA User name[Fixture]> Cut Preset 4.1
```



8.3.67. DataPool Keyword

To enter the DataPool keyword in the command line, use one of the options:

- Press **MA** + **Preset** + **Preset**
- Type **DataPool**
- Type the shortcut **Dat**

Description

The DataPool keyword is used to address a data pool or objects outside the data pool you are currently in.

Syntax

[Function] DataPool [DataPool_Number] ([Object] ["Object_Name" or Object_Number])

Example

- To go to the next cue in sequence 2 in the data pool 2, type:

```
MA User name[Fixture]> Go+ DataPool 2 Sequence 2
```



8.3.68. Deceleration Keyword

To enter the Deceleration keyword in the command line, use one of the options:

- Type **Deceleration**
- Type the shortcut **Dec**

Description

The Deceleration keyword is used to set the deceleration curve of a step in the phaser.

For more information see [Phasers](#).

Syntax

Deceleration

[At] Deceleration [Value]

Examples

- To set the layer to deceleration, type:

```
MA User name[Fixture]> Deceleration
```

- To set the deceleration curve of the dimmer to 85, type:

```
MA User name[Fixture]> At Deceleration 85
```



8.3.69. Decimal8 Keyword

To enter the Decimal8 keyword in the command line, use one of the options:

- Type **Decimal8**
- Type the shortcut **Deci**

Description

The Decimal8 keyword is used to set the values of a fixture selection using the 8bit decimal notation.

Syntax

At [Layer] Decimal8 [Value]

Examples

- To set the absolute layer to 255 in decimal8, type:

```
MA User name[Fixture]> At Absolute Decimal8 255
```

- To set the pan value in the absolute layer to 128 in Decimal8, type:

```
MA User name[Fixture]> Attribute "pan" At Absolute Decimal8 128
```



8.3.70. Decimal16 Keyword

To enter the Decimal16 keyword in the command line, use one of the options:

- Type **Decimal16**
- Type the shortcut **Decimal1**

Description

The Decimal16 keyword is used to set the values of a fixture selection using the 16bit hexadecimal notation.

Syntax

At [Layer] Decimal16 [Value]

Examples

- To set the dimmer for the selected fixtures on the absolute layer to 65535 in Decimal16, type:

```
MA User name[Fixture]> At Absolute Decimal16 65535
```

- To set the pan attribute for the selected fixtures on the absolute layer to 32768 in Decimal16, type:

```
MA User name[Fixture]> Attribute "Pan" At Absolute Decimal16 32768
```



8.3.71. Decimal24 Keyword

To enter the Decimal24 keyword in the command line, use one of the options:

- Type **Decimal24**
- Type the shortcut **Decimal2**

Description

The Decimal24 keyword is used to set the values of a fixture selection using the 24bit decimal notation.

Syntax

At [Layer] Decimal24 [Value]

Examples

- To set the absolute layer to 1677721 in decimal24, type:

```
MA User name[Fixture]> At Absolute Decimal24 1677721
```

- To set the pan value in the absolute layer to 800000 in Decimal24, type:

```
MA User name[Fixture]> Attribute "pan" At Absolute Decimal24 800000
```



8.3.72. Default Keyword

To enter the Default keyword in the command line, use one of the options:

- Press **MA** + **.**
- Type **Default**
- Type the shortcut **Def**

Description

The Default keyword is used to reset the attributes of your fixture selection to default values. If there is no attribute list, all attributes of the selected fixtures will be set to their default values.

Syntax

Default

Fixture ["Fixture_Name" or Fixture_Number] At Default (FeatureGroup ["FeatureGroup_Name" or FeatureGroup_Number])

Store Default (/Option)

Examples

- To set the dimmer of fixture 1 to its default values, type:

```
MA User name[Fixture]> Fixture 1 At Default
```

- To set the position attribute of fixture 2 to default, type:

```
MA User name[Fixture]> Fixture 2 At Default FeatureGroup 2
```

Example: Store Special Values

Requirement:

Activate values in the programmer:

- Call a preset or turn the attribute encoders.

- To store the currently active values as default values, type:



```
MA User name[Fixture]> Store Default
```

- To reset the special values in the currently active values, type:

```
MA User name[Fixture]> Store Default /Remove
```

For more information on the special values see [Parameter List](#).



8.3.73. Delay Keyword

To enter the Delay keyword in the command line, use one of the options:

- Press **Time Time** (If the Time Key Target is set to Fixture. For more information see [User Settings](#) and [Time Key](#).)
- Press **MA Time Time Time**
- Type **Delay**
- Type the shortcut **Dela**

Description

Delay sets the delay time of an object.

For more information on the delay times for cues see [CueDelay Keyword](#).

Syntax

Delay [Delay_Time]

Example

- To set an individual delay time of 4 seconds in the current selection, type:

```
 User name[Fixture]> Delay 4
```



8.3.74. Delete Keyword

To enter the Delete keyword in the command line, use one of the options:

- Press **Delete**
- Type **Delete**
- Type the shortcut **Del**

Description

The Delete keyword is a function keyword which is used to remove data in a show file.

Syntax

Delete [Object] ["Object_Name" or Object_Number]

Option Keywords

The Delete keyword uses the following option keywords:

- **/NoConfirmation**

Examples

- To delete group 1 in the group pool, type:

```
MA User name[Fixture]> Delete Group 1
```

- To delete cue 2 in the selected sequence, type:

```
MA User name[Fixture]> Delete Cue 2
```



8.3.75. DeleteOtherVersions Keyword

To enter the DeleteOtherVersions keyword in the command line, use one of the options:

- Type **DeleteOtherVersions**
- Type the shortcut **Deleteo**

Description

The DeleteOtherVersions keyword is a function keyword that is used to delete the installation files of the software versions in a grandMA3 device.

Syntax

DeleteOtherVersions "release_type_x.x.x.x.xml;/Path/to/MALightingTechnology/installation_packages"

Example

- To delete the installation files of the version 1.6.3.67 on your Windows® computer, type:

```
MA User name[Fixture]> DeleteOtherVersions  
"release_stick_grandMA3_software_v1.6.3.67.xml;C:/ProgramData/MALightingT
```



8.3.76. DeleteGlobalVariable Keyword

To enter the DeleteGlobalVariable keyword in the command line, use one of the options:

- Type **DeleteGlobalVariable**
- Type the shortcut **DeleteG**

Description

The DeleteGlobalVariable keyword is used to delete global variables in a show.

Syntax

DeleteGlobalVariable ["Name of Variable"]

Example

- To delete the global variable Urban Blues 3, type:

```
MA User name[Fixture]> DeleteGlobalVariable "Urban Blues 3"
```



8.3.77. DeleteUserVariable

To enter the DeleteUserVariable keyword in the command line, use one of the options:

- Type **DeleteUserVariable**
- Type the shortcut **DeleteU**

Description

The DeleteUserVariable keyword is used to delete user-specific variables.

Syntax

DeleteUserVariable ["Name of Variable"]

Example

- To delete the user-specific variable "Green 5", type:

```
MA User name[Fixture]> DeleteUserVariable "Green 5"
```



8.3.78. Dismiss Keyword

To enter the Dismiss keyword in the command line, use one of the options:

- Type **Dismiss**
- Type the shortcut **Di**

Description

The Dismiss keyword is used to throw stations out of your session.

Syntax

Dismiss [DeviceType] ["Device_Name"]

Dismiss [DeviceType] [Device_Number]

Dismiss IP [IP_Address]

Find the device number in the **Network menu**:

- To show the devices, unfold the device type.
- Along with other information each device displays number, IP address, and name.

Examples

- To dismiss the console with the name "FOH2", type:

```
MA User name[Fixture]> Dismiss Console "FOH2"
```

- To dismiss the Processing Unit with the number 1, type:

```
MA User name[Fixture]> Dismiss ProcessingUnit 1
```

- To dismiss the station with the IP address 192.168.0.10, type:

```
MA User name[Fixture]> Dismiss IP 192.168.0.10
```



8.3.79. Display Keyword

To enter the Display keyword in the command line, use one of the options:

- Type **Display**
- Type the shortcut **Disp**

Description

Display is an object keyword which is used to manage the displays.

Syntax

[Function] Display

[Function] Display [Display_Number]

Examples

- To list all open displays, type:

```
MA User name[Fixture]> List Display
```

- To close display 2, type:

```
MA User name[Fixture]> Delete Display 2
```

- To retrieve display 2, type:

```
MA User name[Fixture]> Store Display 2
```




8.3.80. DMXAddress Keyword

To enter the DMXAddress in the command line, use one of the options:

- Press **MA** + **X8 | DMX** + **X8 | DMX**
- Type **DMXAddress**
- Type the shortcut **DMXa**

Description

The DMXAddress keyword is used to access DMX addresses directly using an absolute numbering method.

Syntax

[Function] DMXAddress [DMXAddress_Number]

DMXAddress [DMXAddress_Number] At ([Readout]) [Value]

Examples

- To select the fixture patched to universe 2, address 1, type:

```
MA User name[Fixture]> SelectFixtures DMXAddress 513
```

- To output 50% on the third DMX channel of universe 1 using the DMX testing function, type:

```
MA User name[Fixture]> DMXAddress 3 At 50
```

- To output 42% on the DMX channels 8 to 15 on universe 2 using the DMX testing function, type:

```
MA User name[Fixture]> DMXAddress 520 Thru 527 At 42
```

- To disable the DMX testing function on all DMX channels of all universes, type:

```
MA User name[Fixture]> Off DMXAddress Thru
```



8.3.81. DMXLayer Keyword

To enter the DMXLayer keyword in the command line, use one of the options:

- Type **DMXLayer**
- Type the shortcut **DMXL**

Description

DMXLayer is a layer keyword. It is used to change the selected layer to the DMXLayer. This layer can show information in, for instance, the fixture sheet.

The DMXLayer shows the values of the DMX sheet.

If a fixture sheet is set to **Auto** layer, then it will automatically change to show the selected layer.

Syntax

DMXLayer

Examples

- To select the DMXLayer, type:

```
 User name[Fixture]> DMXLayer
```



8.3.82. DMXReadout Keyword

To enter the DMXReadout keyword in the command line, use one of the options:

- Type **DMXReadout**
- Type the shortcut **DMXR**

Description


The DMXReadout keyword is used to set the readout of DMX values in the user profile. It affects the display of values when editing a fixture type or the normal value in the user profile.

Syntax

DMXReadout ["DMXReadout_Name" or DMXReadout_Number]

Example

- To set the DMXReadout to Hex8, type:

```
 User name[Fixture]> DMXReadout "Hex8"
```



8.3.83. DMXUniverse Keyword

To enter the DMXUniverse in the command line, use one of the options:

- Press **MA** + **X8 | DMX**
- Type **DMXUniverse**
- Type the shortcut **DMX**

Description

The DMXUniverse keyword is used to access DMX universes or all DMX channels of a universe.

Syntax

[Function] DMXUniverse ["DMXUniverse_Name" or DMXUniverse_Number]

DMXUniverse ["DMXUniverse_Name" or DMXUniverse_Number] At ([Readout]) [Value]

Examples

- To patch all DMX channels of universe 1 to universe 11, type:

```
MA User name[Fixture]> Move DMXUniverse 1 At DMXUniverse 11
```

- To select the fixture patched to universe 2.001, type:

```
MA User name[Fixture]> SelectFixtures DMXUniverse 2.001
```

- To output 50% on the third DMX channel of universe 1 using the DMX testing function, type:

```
MA User name[Fixture]> DMXUniverse 1.3 At 50
```

- To output 42% on the DMX channels 8 to 15 on universe 2 using the DMX testing function, type:

```
MA User name[Fixture]> DMXUniverse 2.8 Thru 15 At 42
```

- To disable the DMX testing function on all DMX channels of all universes, type:



```
MA User name[Fixture]> Off DMXUniverse Thru
```



8.3.84. DoubleSpeed Keyword

To enter the DoubleSpeed keyword in the command line, use one of the options:

- Type **DoubleSpeed**
- Type the shortcut **DS** or **Dou**

Description

The DoubleSpeed keyword is a playback keyword which is used to change the speed of the speed masters, presets, or sequences to double the speed.

For more information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

DoubleSpeed [Object] ["Object_Name" or Object_Number]

Example

- To double the speed of sequence 2, type:

```
MA User name[Fixture]> DoubleSpeed Sequence 2
```



8.3.85. Down Keyword

To enter the Down keyword in the command line, use one of the options:

- Press **Down**
- Type **Down**
- Type the shortcut **Do**

For more information see the [Down Key](#).

Description

The Down keyword is a function keyword which is used to address the subfixture within the main fixture.

Syntax

Down

Example

To select the subfixtures in the selected fixtures in the programmer, type:

```
MA User name[Fixture]> Down
```



8.3.86. DumpLog Keyword

To enter the DumpLog keyword in the command line, use one of the options:

- Type **DumpLog**
- Type the shortcuts **DU**

Description

DumpLog is a function keyword that is used to deposit the output of the system monitor in form of a text file in case you should experience any inconsistencies or errors in the software. Hence, use this keyword if you are prompted by the tech support.



Hint:

Use the **Select Drive command** to select the drive you would like to export your file to. For more information see the **Select Keyword** and the **Drive Keyword**. The system monitor displays the path the file is exported to.

Syntax

DumpLog

Example

- To deposit the output of the system monitor, type:

```
MA User name[Fixture]> DumpLog
```




8.3.87. Drive Keyword

To enter the Drive keyword in the command line, use one of the options:

- Type **Drive**
- Type the shortcut **Dr**

Description

The Drive keyword is used to address physical drives, for instance, USB flash drives and the internal hard drive.

Syntax

[Function] Drive ["Drive_Name" or Drive_Number]

[Command] If Drive ["Drive_Name" or Drive_Number]

Examples

- To display all available drives, type:

```
MA User name[Fixture]> List Drive
```

- To load the show "Timecode" from the internal drive, type:

```
MA User name[Fixture]> LoadShow "Timecode" If Drive 1
```

8.3.88. DropOwnership Keyword

To enter the **DropOwnership** keyword in the command line, use one of the options:

- Type **DropOwnership**
- Type the shortcut **Dro**

Description

The DropOwnership keyword is used when a user wants to edit an object that is currently owned by a different user. DropOwnership sends a request to release the ownership of an object.



Important:

If a running process owns the object to protect data integrity, DropOwnership might not work right away. If this should be the case, we recommend you use the command again.

Syntax

DropOwnership [Object] ["Object_Name" or Object_Number]

Examples

- To request dropping ownership of group 1, type:

```
MA User name[Fixture]> DropOwnership Group 1
```

- To request dropping ownership of macro 1, line 1, type:

```
MA User name[Fixture]> DropOwnership Macro 1
```

or type:

```
MA User name[Fixture]> DropOwnership Macro 1.1
```

or:

```
MA User name[Fixture]> DropOwnership Macro 1.*
```



8.3.89. Echo Keyword

To enter the Echo keyword in the command line, use one of these options:

- Type **Echo**
- Type the shortcut **Ec**

Description

The Echo keyword is used to display feedback in the windows **Command Line History** and the **System Monitor**.

Syntax

Echo ["Feedback_Text"]

Example

- To feedback "Hello, world!", type:

A screenshot of a terminal window with a dark background. On the left side, there is a small logo with the letters 'M' and 'A' in red. To the right of the logo, the text 'User name[Fixture]> Echo "Hello, world!"' is displayed in white.

Result:

A screenshot of a terminal window with a dark background. The text 'Hello, world!' is displayed in white.



8.3.90. Edit Keyword

To enter the Edit keyword in the command line, use one of the options:

- Press **Edit**
- Type **Edit**
- Type the shortcut **E**

Description

Edit is a function keyword which is used to modify values.

Syntax

Edit [Object] ["Object_Name" or Object_Number]

Option Keywords

The Edit keyword uses the following option keywords:

- **/Tab**

Example

- To edit sequence 1, type:

```
MA User name[Fixture]> Edit Sequence 1
```

The Edit Sequence pop-up opens.



8.3.91. EditSetting Keyword

To enter the EditSetting keyword in the command line, use one of the options:

- Press **Edit** **Edit**
- Type **EditSetting**
- Type the shortcut **EditS**

Description

EditSetting is a function keyword which is used to modify the object settings.

Syntax

EditSetting [Object] ["Object_Name" or Object_Number]

Option Keywords

The EditSetting keyword uses the following option keywords:

- **/Tab**

Examples

- To edit the object settings of group 1, type:

```
MA User name[Fixture]> EditSetting Group 1
```

The group settings pop-up opens.

- To edit the settings of object 5 in preset pool 21, type:

```
MA User name[Fixture]> EditSetting Preset 21.5
```

The Preset settings pop-up opens.

- To change the Cue Part default, enter EditSetting into the command line and then tap the desired preset.
The Preset settings pop-up opens.
- Tap **Cue Part**, enter the new value and apply it by tapping **Please**.



8.3.92. Eject Keyword

To enter the Eject keyword in the command line, use one of the options:

- Type **Eject**
- Type the shortcut **Ej**

Description

The Eject keyword is used to eject a USB flash drive.

Syntax

Eject Drive ["Drive_Name" or Drive_Number]

Examples

- To eject drive 2 (USB flash drive) from the console, type:

```
MA User name[Fixture]> Eject Drive 2
```

- To eject drive Enceladus, type:

```
MA User name[Fixture]> Eject Drive "Enceladus"
```



8.3.93. Effect Keyword

To enter the Effect keyword in the command line, use one of the options:

- Press **Channel** until Effect appears in the command line
- Type **F + 7**
- Type **Effect**
- Type the shortcut **Ef**



Hint:

To use the Effect keyword, make sure you assign the Effect ID type and a custom ID to a fixture first.

Description

The Effect keyword is an object keyword which is used to call fixtures of the ID type Effect in the programmer.

The name of this keyword can change because it is a custom ID type. For more information see [Custom ID Type](#).

Syntax

Effect ["Effect_Name" or Effect_Number]

Example

- To select the fixture that uses the Effect ID 10, type:

```
MA User name[Fixture]> Effect 10
```



8.3.94. EncoderPage Keyword

To enter the EncoderPage keyword in the command line, use one of the options:

- Type **EncoderPage**
- Type the shortcut **EP** or **En**

Description

The EncoderPage keyword addresses the encoder pages.

Syntax

[Function] EncoderPage ["FeatureGroup_Name" or FeatureGroup_Number].["EncoderPage_Name" or EncoderPage_Number]

Examples

Requirement: Patch a RGBAWUv fixture and select it.

- To activate R, G, B, A in the selected fixtures, type:

```
MA User name[Fixture]> On EncoderPage "Color".1
```

- To activate W, Uv in the selected fixtures, type:

```
MA User name[Fixture]> On EncoderPage "Color".2
```


8.3.95. EndIf Keyword



Important:

If the If keyword is already entered in the command line, pressing **If** again makes it an EndIf.

To enter the EndIf keyword in the command line, use one of the options:

- Type **EndIf**
- Type the shortcut **End**

Description

EndIf is a helping keyword that indicates the end of an If statement.

It enables If statements to be entered in the middle of a syntax. Upon processing, the If statement is moved to the end of the syntax, and is used as a filter or condition. This enables If syntax to be used in conjunction with pool items.

For more information see [If Keyword](#).

Syntax

[Function] If [Object] ["Object_Name" or Object_Number] EndIf [Object] ["Object_Name" or Object_Number]

Example

- To create preset 1.1 using fixtures of group 5 type:

```
MA User name[Fixture]> Store If Group 5 EndIf Preset 1.1
```

Result in the command line history:

```
Store Preset 1.1 If Group 5
```



8.3.96. Environment Keyword

To enter the Environment keyword in the command line, use one of the options:

- Type **Environment**
- Type the shortcut **Env**

Description

The Environment keyword is used to grant access to the two environments of the user profile - normal and preview. Each environment has its own programmer, selection, At filter and other.

Syntax

[Function] Environment [Environment_Number] (Property ["Property_Name"] ["Property_Value"])

Examples

- To list the environments, type:

```
MA User name[Fixture]> List Environment
```

- To enable Single Step for Preview, type:

```
MA User name[Fixture]> Set Environment 2 Property "SingleStep" "Yes"
```



8.3.97. Exchange Keyword

To enter the Exchange keyword in the command line, use one of the options:

- Press **Move** **Move**
- Type **Exchange**
- Type the shortcut **Exc**

Description

The Exchange keyword is a function keyword used to swap places.

Syntax

Exchange [Object] ["Object_Name" or Object_Number] At [Object] ["Object_Name" or Object_Number]

Example

- To exchange macro 1 for macro 3, type:

```
MA User name[Fixture]> Exchange Macro 1 At Macro 3
```



8.3.98. Executor Keyword

To enter the **Executor** keyword in the command line, use one of the options:

- Press **MA** + **X16**
- Type **Executor**
- Type the shortcut **Ex**

Description

The Executor keyword is an object keyword used as a control handle for other objects.

The default function for Executor objects is **Select**. This means that calling executors without any function specified selects the object assigned to the executor. This selection is now also controllable with the 100 mm fader section.

If you apply a function or reference a property not supported by the Executor object, the command will be passed on to its child: key, fader, or the object assigned to the executor.

Syntax

Executor [Executor_ID]

Select Page [Page_ID] Executor [Executor_ID]

Set Executor [Executor_ID] [Setting] = [Setting_Option]

Settings

The following table displays the settings that can be set using the command line:

Setting	Setting Options	Description
Key	Go+ etc.	Executor key assignment
Fader	Master etc.	Executor fader assignment
Encoder	Master etc.	Executor encoder assignment
EncoderLeft	<<< etc.	Executor encoder assignment when turning the encoder counterclockwise
EncoderRight	>>> etc.	Executor encoder assignment when turning the encoder clockwise
KeyCmd	Go+ etc.	Command run when executor button is pressed
EncoderRightCmd	<<< etc.	Command run when the encoder is turned counterclockwise
EncoderLeftCmd	>>> etc.	Command run when the encoder is turned clockwise



Setting	Setting Options	Description
MAKey	Go+ etc.	Executor key assignment when pressing it together with the MA key
MAFader	Master etc.	Executor fader assignment when pressing the MA key
MAEncoder	Master etc.	Executor encoder assignment when the MA key is pressed
MAEncdoerRight	<<< etc.	Executor encoder assignment when turning the encoder counterclockwise while pressing the MA key
MAEncoderLeft	>>> etc.	Executor encoder assignment when turning the encoder clockwise while pressing the MA key
MAKeyCmd	Go+ etc.	Command run when executor button is pressed together with the MA key
MAEncoderRightCmd	<<< etc.	Command run when the encoder is turned counterclockwise together with the MA key
MAEncoderLeftCmd	>>> etc.	Command run when the encoder is turned clockwise together with the MA key
PrimaryAssignmentChanged		This is information only. See the description below
SecondaryAssignmentChanged		This is information only. See the description below
Width	1-5	Executor width
Height	1-4	Executor height
Object		The object the executor controls
Config		The executor configuration used by the executor
TotalPrimaryAssignmentChanged		This is information only. See the description below
TotalSecondaryAssignmentChanged		This is information only. See the description below

The four information settings are related to changes made in relation to the used executor configuration. Primary assignments are the assignments the executor has when the MA key is not pressed. The Secondary assignments are the assignments the executor has while the MA key is pressed. The two properties beginning with "Total" are for the entire combined executor. This is relevant when the executor is part of a combined executor with more than one executor in height and/or width. The information settings cannot be changed, they are automatically updated by the software.

For setting the executor assignments using the interface, please read the [Assign Object to an Executor topic](#) and the [Executor Configurations topic](#).

Examples

- To remove executor 205 on the current page, type:

```
MA User name[Fixture]> Delete Executor 205
```



It does not delete the object assigned to the executor. It just deletes the assignment.

- To delete cue 3 of the sequence assigned to executor 205, type:

```
MA User name[Fixture]> Delete Executor 205 Cue 3
```

- To select executor 102 on page 4, type:

```
MA User name[Fixture]> Select Page 4.102
```

- To set the setting "Key" of executor 201 to "Flash", type:

```
MA User name[Fixture]> Set Executor 201 "Key" = "Flash"
```

For more information see [Executors](#).

8.3.99. Export Keyword

To enter the Export keyword in the command line, use one of the options:

- Type **Export**
- Type the shortcut **Exp**

Description

Export is a function keyword which is used to save objects from the current show file as a smaller file.

If no file name is used in the command, the file name will use the name of the object.

By default, files will be exported to the relevant folder within the library folder structure, either on the local drive of the console or on PC station, or on a selected USB drive. For more information on grandMA3 folders see [Folder Structure](#).



Important:

When exporting several objects at a time without indicating a file name for each pool object, a separate XML file will be exported.
When exporting several objects at a time indicating a file name, all pool objects will be exported into a shared XML file.



Hint:

The Import/Export menu offers a graphical user interface for import and export functions. For more information about the Import/Export menu, see [Import/Export Menu](#).

Syntax

Export [Object] ["Object_Name" or Object_Number] (If Drive [Drive_Number]) (/Option) ("Option_Value")

Option Keywords

The Export keyword uses the following option keywords:

- [/Gaps](#)
- [/GDTF](#)
- [/NoDependencies](#)
- [/Path](#)
- [/Type](#)

Examples

- To export macro 1 as the XML file "test", type:



```
MA User name[Fixture]> Export Macro 1 "test"
```

- To export macro 1 using the macro name, type:

```
MA User name[Fixture]> Export Macro 1
```

- To export several macros to single XML files at a time, type:

```
MA User name[Fixture]> Export Macro 1 Thru 42
```

- To export macro 1 "test" to the first connected USB drive, type:

```
MA User name[Fixture]> Export Macro 1 "test" If Drive 2
```




8.3.100. Extract Keyword

To enter the **Extract** keyword in the command line, use one of the options:

- Press **MA** + **At** + **At**
- Type **Extract**
- Type the shortcut **Ext**

Description

The Extract keyword is a command keyword used to call the values of presets without references.

Syntax

Extract [Object] ["Object_Name" or Object_Number] (/Option)

Option Keywords

The Extract keyword uses the following option keywords:

- **/Single**

Example

- To call color preset 2, type:

```
 User name[Fixture]> Extract Preset 4.2
```



8.3.101. Extension Keyword

To enter the Extension keyword in the command line, use one of the options:

- Type **Extension**
- Type the shortcut **Exte**

Description

The Extension keyword is an object keyword.

This keyword belongs to a session-related group of keywords.

Syntax

[Function] Extension ["Extension_Name" or Extension_Number]

Examples

To list all extensions that are currently in the same network, type:

```
MA User name[Fixture]> List Extension
```

To invite extension "FOH3" to the station where you execute the command, type:

```
MA User name[Fixture]> Invite Extension "FOH3"
```



8.3.102. Fade Keyword

To enter the **Fade** keyword in the command line, use one of the options:

- Press **Time** (If the Time Key Target is set to Fixture. For more information see [User Settings](#) and [Time Key](#).)
- Press **MA Time Time**
- Type **Fade**

Description

The Fade keyword is a helping keyword which is used to indicate fade times.

As a helping keyword for playback functions (for example Goto), this keyword sets the time which is used to execute the function.

If it is used as a starting keyword, Fade will apply individual timing in the programmer for the current selection and attributes.



Hint:

As long as the command starts with a function, the Fade keyword and the fade value can be randomly used within the command.

To set the fade times for cues, read more in the [CueFade keyword topic](#).

Syntax

([Function] [Object] ["Object_Name" or Object_Number]) Fade [Fade_Value]

Examples

- To crossfade to cue 3 in the selected sequence in 4 seconds, type:

```
MA User name[Fixture]> Goto Cue 3 Fade 4
```

- To set the individual fade time of 2 seconds to the dimmer of the current selection, type:

```
MA User name[Fixture]> Fade 2
```

- To set the dimmer value of the current fixture selection to 50 % and apply an individual fade time of 2 seconds to the dimmer of the current selection, type:



```
MA User name[Fixture]> At 50 Fade 2
```



8.3.103. FaderCrossFade Keyword

To enter the FaderCrossFade keyword in the command line:

- Type **FaderCrossFade**
- Type the shortcut **FaderC** or **FaderX**

Description

The FaderCrossFade keyword represents the crossfade function of a sequence.

Crossfade gradually activates the next cue of a sequence in accordance with the position of the fader.

For more information see [Executors](#).

Syntax

[Function] FaderCrossFade At [Object] ["Object_Name" or Object_Number]

FaderCrossFade [Object] ["Object_Name or Object_Number] At [Value]

Example

- To assign FaderCrossFade as executor 302, type:

```
MA User name[Fixture]> Assign FaderCrossFade At Executor 302
```

- To set the FaderCrossFade value of the fader range to 10% in sequence 5, type:

```
MA User name[Fixture]> FaderCrossFade Sequence 5 At 10
```



8.3.104. FaderCrossFadeA Keyword

To enter the FaderCrossFadeA keyword in the command line:

- Type **FaderCrossFadeA**
- Type the shortcut **FaderXA**

Description

The FaderCrossFadeA keyword represents the crossfade A function of a sequence.

Crossfade A gradually fades out dimmer attributes of a current cue in a sequence in accordance with the position of the fader.

For more information see [Executors](#).

Syntax

[Function] FaderCrossFadeA At [Object] ["Object_Name" or Object_Number]

FaderCrossFadeA [Object] ["Object_Name" or Object_Number] At [Value]

Example

- To assign FaderCrossFadeA as fader function as executor 303, type:

```
MA User name[Fixture]> Assign FaderCrossFadeA At Executor 303
```

- To set the FaderCrossFadeA value to 10% of the fader range for sequence 5, type:

```
MA User name[Fixture]> FaderCrossFadeA Sequence 5 At 10
```



8.3.105. FaderCrossFadeB Keyword

To enter the FaderCrossFadeB keyword in the command line:

- Type **FaderCrossFadeB**
- Type the shortcut **FaderXB**

Description

The FaderCrossFadeB keyword represents the crossfade B function of a sequence.

Crossfade B gradually fades in dimmer attributes of the next cue in a sequence in accordance with the position of the fader.

For more information see [Executors](#).

Syntax

[Function] FaderCrossFadeB At ["Object_Name" or Object_Number]

FaderCrossFadeB [Object] ["Object_Name" or Object_Number] At [Value]

Example

- To assign FaderCrossFadeB as fader function to executor 304, type:

```
MA User name[Fixture]> Assign FaderCrossFadeB At Executor 304
```

- To set the FaderCrossFadeB value of the fader range to 10% in sequence 5, type:

```
MA User name[Fixture]> FaderCrossFadeB Sequence 5 At 10
```



8.3.106. FaderMaster Keyword

To enter the FaderMaster keyword in the command line, use one of the options:

- Type **FaderMaster**
- Type the shortcut **FaderM**
- Press **MA** + **X16 | Exec** + **X16 | Exec** + **X16 | Exec**

Description

The FaderMaster keyword applies the Master function to an executor.

The master function controls the intensity of the assigned object.

For more information see the [Executors](#).

Syntax

Assign FaderMaster At Executor [Executor_Number]

FaderMaster [Executor_Number] At [Value] (Fade [Time])

Only sequences and presets can move the master using a fade time. Other objects snap to the value.

Examples

- To assign FaderMaster to executor 204, type:

```
MA User name[Fixture]> Assign FaderMaster At Executor 204
```

- To move FaderMaster 205 at position 50% in a 5 seconds fade:

```
MA User name[Fixture]> FaderMaster 205 At 50 Fade 5
```




8.3.107. FaderRate Keyword

To enter the FaderRate keyword in the command line, use one of the options:

- Type **FaderRate**
- Type the shortcut **FaderR**

Description

The FaderRate keyword applies the Rate function to an executor.

Rate divides or multiplies the fade and delay time in a sequence by the value of the fader. If Speed from Rate is enabled, it is then also valid for the speed stored in cues.

For more information see [Executors](#).

Syntax

Assign FaderRate At Executor ["Executor_Name" or Executor_Number]

FaderRate [Object] ["Object_Name" or Object_Number] At [Value]

Examples

- To assign FaderRate to executor 205, type:

```
MA User name[Fixture]> Assign FaderRate At Executor 205
```

- To move the rate fader of sequence 1 to 1:1, type:

```
MA User name[Fixture]> FaderRate Sequence 1 At 100
```



8.3.108. FaderSpeed Keyword

To enter the FaderSpeed keyword in the command line, use one of the options:

- Type **FaderSpeed**
- Type the shortcut **FaderS**

Description

The FaderSpeed keyword applies the Speed function to an executor.

It controls the speed of a phaser in a cue.

For more information see [Executors](#).

Syntax

Assign FaderSpeed At Executor ["Executor_Name" or Executor_Number]

FaderSpeed [Object] ["Object_Name" or Object_Number] At [Speed_Readout] [Speed_Value]

Examples

- To assign FaderSpeed to executor 206, type:

```
MA User name[Fixture]> Assign FaderSpeed At Executor 206
```

- To set the FaderSpeed of sequence 2 to 6 BPM, type:

```
MA User name[Fixture]> FaderSpeed Sequence 2 At BPM 6
```



8.3.109. FaderTemp Keyword

To enter the FaderTemp keyword in the command line, use one of the options:

- Type **FaderTemp**
- Type the shortcut **FaderT**

Description

The FaderTemp keyword applies the Temp function to an executor.

Temp crossfades the cue on when pulled up, and off when pulled down.

For more information see the [Executors](#).

Syntax

Assign FaderTemp At Executor ["Executor_Name" or Executor_Number]

FaderTemp [Object] ["Object_Name" or Object_Number] At [Value]

Examples

- To assign FaderTemp to executor 301, type:

```
MA User name[Fixture]> Assign FaderTemp At Executor 301
```

- To set the temp fader of sequence 2 to 42%, type:

```
MA User name[Fixture]> FaderTemp Sequence 2 At 42
```



8.3.110. FaderTime Keyword

To enter the FaderTime keyword in the command line, use one of the options:

- Type **FaderTime**
- Type the shortcut **FaderTi**

Description

The FaderTime keyword applies the Time function to a sequence. It is used to overwrite the stored cue part times by setting a time value and activating the time function.

For more information about cue time overwriting see [Cue Timing](#).

For more information on how to activate and deactivate the function see [Time](#).

Syntax

Assign FaderTime At [Object] ["Object_Name" or Object_Number]

FaderTime [Object] ["Object_Name" or Object_Number] At [Value]

Examples

- To assign FaderTime function to executor 207 on the selected page, type:

```
MA User name[Fixture]> Assign FaderTime At Executor 207
```

- To set the FaderTime value to 50% of the time range for the selected sequence, type:

```
MA User name[Fixture]> FaderTime At 50
```

- To set the FaderTime value to 10% of the time range for sequence 5, type:

```
MA User name[Fixture]> FaderTime Sequence 5 At 10
```



8.3.111. FeatureGroup Keyword

To enter the FeatureGroup keyword in the command line, use one of the options:

- Press **MA** + **Preset**
- Type **FeatureGroup**
- Type the shortcut **Fg**

Description

The FeatureGroup keyword is used to address feature groups.

Syntax

[Function] FeatureGroup ["FeatureGroup_Name" or FeatureGroup_Number]

Example

- To set the value Remove in all attributes of the FeatureGroup "Dimmer" in the current selection, type:

```
MA User name[Fixture]> Remove FeatureGroup "Dimmer"
```



8.3.112. Filter Keyword

To enter the Filter keyword in the command line, use one of the options:

- Press `Group` `Group` `Group`
- Type **Filter**
- Type the shortcut **Fil**

Description

The Filter keyword is used to call a filter. It represents an attribute and a layer filter.

For more information see [Worlds and Filters](#).

Syntax

[Function] Filter ["Filter_Name" or Filter_Number] (/Option)

Option Keywords

The Filter keyword uses the following option keywords:

- **/Overwrite**

Example

- To call filter 4, type:

```
MA User name[Fixture]> Filter 4
```



8.3.113. Fix Keyword

To enter the Fix keyword in the command line, use one of the options:

- Press **MA** + **Pause**
- Type **Fix**

Description

Fix is a function keyword which is used to latch executors on a page. The executors that are latched will always be displayed, even if you change the page.

Fix is a toggle function. This means that using Fix without any helping keyword toggles the fixing of the executors on and off.

Syntax

Fix (On or Off) Executor [Executor_Number]

Fix (On or Off) Page ["Page_Name" or Page_Number].[Executor_Number]

Examples

- To fix executor 101 through 105 on the current page, type:

```
MA User name[Fixture]> Fix On Executor 101 Thru 105
```

- To toggle executor 103 between changing page, or not changing page, type:

```
MA User name[Fixture]> Fix Executor 103
```



8.3.114. FixtureClass Keyword

To enter the FixtureClass keyword in the command line, use one of the options:

- Press **MA** + **Fixture** + **Fixture** + **Fixture**
- Type **FixtureClass**
- Type the shortcuts **FC**

Description

FixtureClass is an object keyword which addresses the fixture classes of a show file.

Syntax

[Function] FixtureClass ["FixtureClass_Name" or FixtureClass_Number]

Example

Requirement: Create the class "Spots" in the patch and link fixtures to it.

- To create a group in the group pool object 301 that contains all patched fixtures that are set to class "Spots" in the patch, type:

```
MA User name[Fixture]> AutoCreate FixtureClass "Spots" At Group 301
```

- To select all fixtures that are set to the class "Spots" in the patch, type:

```
MA User name[Fixture]> SelfFix FixtureClass "Spots"
```




8.3.115. FixtureLayer Keyword

To enter the FixtureLayer keyword in the command line, use one of the options:

- Press **MA** + **Fixture** + **Fixture**
- Type **FixtureLayer**
- Type the shortcuts **FL**

Description

FixtureLayer is an object keyword which addresses the layers of fixtures in a show file.

Syntax

[Function] FixtureLayer ["FixtureLayer_Name" or FixtureLayer_Number]

Examples

Requirement: Create the layer "Backtruss" in the patch and link fixtures to it.

- To create single fixture groups starting in group pool object 201 using all patched fixtures that are set to the layer "Backtruss" within the patch, type:

```
MA User name[Fixture]> AutoCreate FixtureLayer "Backtruss" At Group 201 /Single
```

- To select all fixtures that are set to layer "Backtruss" within the patch, type:

```
MA User name[Fixture]> SelectFixtures FixtureLayer "Backtruss"
```



8.3.116. Fixture Keyword

To enter the Fixture keyword in the command line, use one of the options:

- Press **Fixture**
- Type **Fixture**
- Type the shortcut **F**

Description

The Fixture keyword is used as an object keyword to access fixtures that have a fixture ID.

Syntax

Fixture ["Fixture_Name" or Fixture_Number]

Fixture ["Fixture_Name" or Fixture_Number],["SubFixture_Name" or SubFixture_Number]

Examples

- To select fixture 2, type:

```
MA User name[Fixture]> Fixture 2
```

- To select the fifth subfixture of fixture 10, type:

```
MA User name[Fixture]> Fixture 10.5
```

8.3.117. FixtureType Keyword

To enter the FixtureType keyword in the command line, use one of the options:

- Press **MA** + **Fixture**
- Type **FixtureType**
- Type the shortcuts **FT** or **FixtureT**

Description

FixtureType is an object keyword which addresses the fixture types of a show file.



Important:

Most edits and command line actions with the keyword FixtureType has to be done while in the Edit Setup mode. For more information, see [ChangeDestination keyword](#).

Syntax

[Function] FixtureType ["FixtureType_Name" or FixtureType_Number]

Examples

Requirement:

- Enter the Patch menu first.
For more information see [Patch and Fixture Setup](#).

- To assign fixture type 2 to fixtures 1 through 4, type:

```
MA User name@ShowData/Patch/Stages/Stage 1> Assign FixtureType 2 At 1 Thru 4
```

- To select all patched fixtures of fixture type 3, type:

```
MA User name[Fixture]> SelectFixtures FixtureType 3
```

- To create single fixture groups starting in group pool object 401 using all fixtures of the fixture type 9, type:

```
MA User name[Fixture]> AutoCreate FixtureType 9 At Group 401 /Single
```



8.3.118. Flip Keyword

To enter the Flip keyword in the command line, use one of the options:

- Type **Flip**
- Type the shortcut **Fli**

Description

The Flip keyword is used to access the different pan/tilt combinations that direct a moving head in the same direction.

Flip adds 180 degrees to the pan value of the fixtures and inverts the tilt angle. If the fixtures reach their physical breakpoint, the pan and tilt values will be set to the smallest possible value.

That is, Flip directs the fixture in the same direction using a different pan/tilt combinations.



Hint:

- If no selection list is entered, Flip is applied to the fixture selection.
- If no number is entered, the function toggles through the different possible combinations.
- The number of combinations depends on the possible degree value the fixture can pan in.

Syntax

Flip ([Flip_Number] [Object] ["Object_Name" or Object_Number])

Examples

- To set the pan and tilt of the fixture selection to the next pan/tilt combination, type:

```
MA User name[Fixture]> Flip
```

- To set the pan and tilt of group 7 to the second pan/tilt combination that directs the fixtures in the same direction, type:

```
MA User name[Fixture]> Flip 2 Group 7
```



8.3.119. Flash Keyword

To enter the Flash keyword in the command line, use one of the options:

- Press **MA** + **>>>**
- Type **Flash**
- Type the shortcut **Fla**

Description

The Flash keyword is a playback keyword which is used to temporarily overwrite master level in order to set it to full on executing objects without using times.

If the executor is disabled and Flash is applied, the executor is temporarily activated using zero timing.

For more information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

Flash (On/Off) [Object] ["Object_Name" or Object_Number]

Examples

- To overwrite the master level of executor 201 and to start in first step, type:

```
MA User name[Fixture]> Flash On Executor 201
```

- To recall the overwriting, and set the executor to master fader and disable it, type:

```
MA User name[Fixture]> Flash Off Executor 201
```



8.3.120. Fog Keyword

To enter the **Fog** keyword in the command line, use one of the options:

- Press **Channel** until Fog appears in the command line
- Type **F + 6**
- Type **Fog**
- Type the shortcut **Fo**



Hint:

To use the Fog keyword, make sure you assign the Fog ID type and a custom ID to a fixture first.

Description

The Fog keyword is an object keyword which is used to call fixtures of the ID type Fog in the programmer.

The name of this keyword can change because it is a custom ID type. For more information see the [Custom ID Type](#).

Syntax

Fog ["IDType_Name" or IDType_Number]

Example

- To select the fixture that uses the Fog ID 7, type:

```
MA User name[Fixture]> Fog 7
```



8.3.121. Freeze Keyword



Important:

If Freeze is on, programmer values have a higher priority as playbacks.
Only executor priority **Super** has a higher priority than **Freeze**.

To enter the Freeze keyword in the command line, use one of the options:

- Press **Freeze**
- Type **Freeze**
- Type the shortcut **Fr**

Description

Freeze is a function keyword which is used to change the priority of the programmer.

Freeze is a toggle function. This means that entering Freeze without any helping keyword toggles the Freeze mode on/off.

Syntax

Freeze (On/Off)

Example

To turn on the freeze mode, type:

```
MA User name[Fixture]> Freeze On
```




8.3.122. Full Keyword



Important:

Pressing **Full** instantly executes the keyword.

To enter the **Full** keyword in the command line, use one of the options:

- Type **Full**
- Type the shortcut **Fu**

Description

Full is a function keyword which is used to set the dimmer values to 100 %.

The default function is At. This means that entering Full sets the dimmer attributes of the current selection to 100 %.

Syntax

([Object] ["Object_Name" or Object_Number]) Full

Examples

- To select fixture 1 thru 10 and set the dimmer to 100 %, type:

```
MA User name[Fixture]> Fixture 1 Thru 10 Full
```

- To select channel 53 and set the dimmer to 100 %, type:

```
MA User name[Fixture]> Channel 53 Full
```

8.3.123. Gel Keyword

To enter the **Gel** keyword in the command line, use one of the options:

- Press `Preset` `Preset` `Preset`
- Type **Gel**

Description

The keyword Gel provides the opportunity to edit or apply swatch book colors using the command line.

Syntax

[Function] Gel ["Swatch_Name"."Gel_Name"]

[Function] Gel [Swatch_Number].[Gel_Number]

[Function] Gel [Swatch_Number]

[Function] Gel ["Swatch_Name"]

Examples

- To set the color of the selected fixtures to Lee's color Mauve, type:

```
MA User name[Fixture]> At Gel "Lee"."Mauve"
```

- To set the color of the selected fixtures to the 44th color of the eighth swatch book, type:

```
MA User name[Fixture]> At Gel 8.44
```

- To export the "Lee" swatch book, type:

```
MA User name[Fixture]> Export Gel 8
```



8.3.124. GetGlobalVariable Keyword

To enter the GetGlobalVariable keyword in the command line, use one of the options:

- Type **GetGlobalVariable**
- Type the shortcut **Getg**

Description

The GetGlobalVariable keyword is used to display global variables along with their values in the command line history.

Syntax

GetGlobalVariable ["Variable _Name"]

Examples

- To display all variables in the command line history, type:

```
MA User name[Fixture]> GetGlobalVariable "*"
```

- To display all variables beginning with the letter f, type:

```
MA User name[Fixture]> GetGlobalVariable "f*"
```



8.3.125. GetUserVariable Keyword

To enter the GetUserVariable keyword in the command line, use one of the options:

- Type **GetUserVariable**
- Type the shortcut **GetU**

Description

The GetUserVariable keyword is used to display user-specific variables along with their values in the command line history.

Syntax

GetUserVariable ["Variable_Name"]

Examples

- To display all user-specific variables in the command line history, type:

```
MA User name[Fixture]> GetUserVariable "*"
```

- To display all user-specific variables beginning with the letter f, type:

```
MA User name[Fixture]> GetUserVariable "f*"
```



8.3.126. Go+ Keyword

To enter the Go+ keyword in the command line, use one of the options:

- Press **Go+ | Temp**
- Type **Go+**
- Type the shortcut **Go**

Description

The Go+ keyword is a playback keyword and it operates as a start signal for commands to be executed.

If the target object has cues, it will proceed to the next cue.

For more information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

Go [Object] ["Object_Name" or Object_Number]

Examples

- To go to the next cue of executor 101, type:

```
MA User name[Fixture]> Go+ Executor 101
```

- To start macro 2, type:

```
MA User name[Fixture]> Go+ Macro 2
```



8.3.127. Go- Keyword

To enter the Go- keyword in the command line, use one of the options:

- Press **Go- | Top**
- Type **Go-**

Description

The Go- keyword is a playback keyword that is used to activate the previous cue.

If the target object has cues, it will go back to the previous cue.

For more information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

Go- [Object] ["Object_Name" or Object_Number]

Example

- To go to the previous cue of executor 101, type:

```
MA User name[Fixture]> Go- Executor 101
```



8.3.128. Goto Keyword

To enter the **Goto** keyword in the command line, use one of the options:

- Press **Goto**
- Type **Goto**
- Type the shortcut **Got**

Description

Goto is a function keyword that is used to jump in a list.

Syntax

Goto [Object] ["Object_Name" or Object_Number]

Examples

- To go to cue 103 of the selected executor, type:

```
MA User name[Fixture]> Goto Cue 103
```

- To go to cue 105 of executor 104, type:

```
MA User name[Fixture]> Goto Cue 105 Executor 104
```



8.3.129. Grid Keyword

To enter the Grid keyword in the command line, use one of these options:

- Press **MA** + **X3**
- Type **Grid**
- Type the shortcut **Gri**

Description

The Grid keyword is used to set the grid cursor or span a grid area in the selection grid window. After the cursor is set, it is possible to arrange fixtures in the selection grid window. To position fixtures with a z-axis in the selection window, use the Grid keyword.

To set the grid cursor using the user interface see [Selection Grid Window](#).

The grid command supports a space or a / (slash) as delimiters for command line input.

Syntax

Grid [x-axis]/[y-axis]/[z-axis]


Grid [x-axis] [y-axis] [z-axis]

Examples

- To move the blue grid cell cursor in the selection window to column 2, row 2, type:

```
 User name[Fixture]> Grid 2/2
```

- To position fixtures in a defined area in the selection window, type:

```
 User name[Fixture]> Grid 2/2 Thru 4/4
```




8.3.130. GridPosition Keyword

To enter the GridPosition keyword in the command line, use one of the options:

- Type **GridPosition**
- Type the shortcut **GridP**

Description

GridPosition is a layer keyword which represents the xyz grid coordinates of fixtures at a time as values were set in attributes. It is used to change the selected layer to GridPosition. This layer can show information in, for instance, the fixture sheet.

Syntax

GridPosition

Example

- To select the layer GridPosition, type:

```
MA User name[Fixture]> GridPosition
```



8.3.131. Group Keyword

To enter the Group keyword in the command line, use one of the options:

- Press **Group**
- Type **Group**
- Type the shortcut **G**

Description

The Group keyword is an object type that contains a certain selection of fixtures in a specific order along with x-, y-, z-coordinates used in the **Selection window**.

The default function of the Group keyword is SelfFix. This means that calling groups without specifying any function selects the fixtures within the group.

For more information see the **SelfFix keyword**.

Syntax

([Function]) Group ["Group_Name" or Group_Number]

Option Keywords

The Group keyword uses the following option keywords:

- **/GridMergeMode**
- **/Merge**
- **/Overwrite**
- **/Remove**

Examples

- To select the fixtures stored in group 3, type:

```
MA User name[Fixture]> Group 3
```

- To list all stored groups of the group pool in the command line history, type:

```
MA User name[Fixture]> List Group
```



8.3.132. HalfSpeed Keyword

To enter the HalfSpeed keyword in the command line, use one of the options:

- Type **HalfSpeed**
- Type the shortcut **HS** or **Hal**

Description

The HalfSpeed keyword is a playback keyword which is used to change the speed of the speed masters, presets, or sequences to half the speed.

For more information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

HalfSpeed [Object] ["Object_Name" or Object_Number]

Example

- To half the speed of the first speed master, type:

```
 User name[Fixture]> HalfSpeed Master 3.1
```



8.3.133. Help Keyword

To enter the Help keyword in the command line, use one of the options:

- Press **Help**
- Type **Help**
- Type the shortcut **H**



Hint:

Use the context-sensitive help:

-Press and hold **Help** or tap **?** in the control bar and tap a UI element to open the corresponding help topic.

Description

Help is a function used to list all command keywords, list of operators, layer keywords, cue timing keywords, readout keywords, the playback commands, etc. The keywords and what they do is displayed in white. The corresponding shortcuts are displayed in green.

Syntax

Help ([Filter])

Examples

- To list all keywords in the Command Line History, type:

```
MA User name[Fixture]> Help
```

- To list all keywords starting with an f, type:

```
MA User name[Fixture]> Help f*
```



8.3.134. HelpLua Keyword

To enter the HelpLua keyword in the command line, use one of the options:

- Type **HelpLua**
- Type the shortcut **HelpL**

Description

HelpLua is a function keyword which is used to export a list of all grandMA Lua functions.

To learn more about Lua functions, read the [Plugins](#) topic.

Syntax

HelpLua

Example

- To list all specific grandMA3 Lua functions, type:

```
MA User name[Fixture]> HelpLua
```

The Lua functions are exported into the file "grandMA3_lua_functions.txt" in the **grandMA3 library folder**.



8.3.135. Hex8 Keyword

To enter the Hex8 keyword in the command line, use one of the options:

- Type **Hex8**
- Type the shortcut **Hex**

Description

The Hex8 keyword is used to set the values of a fixture selection using the 8bit hexadecimal notation.

Syntax

At [Layer] Hex8 ["Value"]

Examples

- To set the absolute layer to FF in Hex8, type:

```
MA User name[Fixture]> At Absolute Hex8 "FF"
```

- To set the pan value in the absolute layer to DE in Hex8, type:

```
MA User name[Fixture]> Attribute "pan" At Absolute Hex8 "DE"
```



8.3.136. Hex16 Keyword

To enter the Hex16 keyword in the command line, use one of the options:

- Type **Hex16**
- Type the shortcut **Hex1**

Description

The Hex16 keyword is used to set the values of a fixture selection using the 16bit hexadecimal notation.

Syntax

At [Layer] Hex16 ["Value"]

Examples

- To set the dimmer for the selected fixtures on the absolute layer to FFFF in Hex16, type:

```
MA User name[Fixture]> At Absolute Hex16 "FFFF"
```

- To set the pan attribute for the selected fixtures on the absolute layer to 8000 in Hex16, type:

```
MA User name[Fixture]> Attribute "Pan" At Absolute Hex16 "8000"
```



8.3.137. Hex24 Keyword

To enter the Hex24 keyword in the command line, use one of the options:

- Type **Hex24**
- Type the shortcut **Hex2**

Description

The Hex24 keyword is used to set the values of a fixture selection using the 24bit hexadecimal notation.

Syntax

At [Layer] Hex24 ["Value"]

Examples

- To set the absolute layer to FFFFFFFF in Hex24, type:

```
MA User name[Fixture]> At Absolute Hex24 "FFFFFFF"
```

- To set the pan value in the absolute layer to 800000 in Hex24, type:

```
MA User name[Fixture]> Attribute "pan" At Absolute Hex24 "800000"
```


8.3.138. Highlight Keyword

To enter the Highlight keyword in the command line, use one of the options:

- Press **Highlt**
- Type **Highlight**
- Type the shortcut **Hi**

Description

The Highlight keyword is a playback keyword that is used to apply the defined highlight values on fixtures that are selected.

It is a toggle function. This means, entering Highlight without a helping keyword enables or disables the Highlight mode.



Hint:

To manually fade from the current value to the highlight value and back, use the Highlight fader. For more information see [Master Controls](#).

For information on location see the [Highlt Key](#).

For information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

([Function]) Highlight (On/Off)

Example

- To toggle the highlight mode, type:

```
MA User name[Fixture]> Highlight
```

Example: Special Values

Requirement:

Activate values in the programmer:

- Call a preset or turn the attribute encoders.

- To store the currently active values as highlight values, type:



```
MA User name[Fixture]> Store Highlight
```

For more information on the special values see [Parameter List](#).



8.3.139. Houselights Keyword

To enter the **Houselights** keyword in the command line, use one of the options:

- Press **Channel** until Houselights appears in the command line
- Type **F + 3**
- Type **Houselights**
- Type the shortcut **Ho**



Hint:

To use the Houselights keyword, make sure you assign the Houselights ID type and a custom ID to a fixture first.

Description

The Houselights keyword is an object keyword which is used to call fixtures of the ID type Houselights in the programmer.

The name of this keyword can change because it is a custom ID type. For more information see the [Custom ID Type](#).

Syntax

Houselights ["IDType_Name" or IDType_Number]

Example

- To select the fixture that uses the Houselights ID 5, type:

```
MA User name[Fixture]> Houselights 5
```



8.3.140. Hz Keyword

To enter the Hz (Hertz) keyword in the command line, type: **Hz**

Description

The Hz (Hertz) keyword is used to set the speed of a fixture selection or a speed master using the unit hertz.

Syntax

At Speed Hz [Value]

Master [Master_Number] At Hz [Value]

Examples

- To set the speed layer to 60 hertz, type:

```
MA User name[Fixture]> At Speed Hz 60
```

- To set the first speed master to 3 hertz, type:

```
MA User name[Fixture]> Master 3.1 At Hz 3
```

8.3.141. If Keyword



Important:

To end an if statement in the middle of a longer syntax, use the EndIf keyword.
For more information see [EndIf Keyword](#).

To enter the If keyword in the command line, use one of the options:

- Press **If**
- Type If

Description

The If keyword is a function keyword which is used to deselect fixtures in a selection.

As a helping keyword If sets a filter for the operation.

As a helping keyword for the Clone function If sets the scope of cloning.

For more information see [Clone Keyword](#).

Syntax

([Function] [Object] ["Object_Name" or Object_Number]) If [Object] ["Object_Name" or Object_Number]

Clone [Object] ["Object_Name" or Object_Number] At [Object] ["Object_Name" or Object_Number] If [Object] ["Object_Name" or Object_Number]

Examples

- To deselect fixtures which are not in group 5, type:

```
MA User name[Fixture]> If Group 5
```

- To deselect fixtures which are not in group 3 and group 5, type:

```
MA User name[Fixture]> Group 3 If Group 5
```

- To delete channel 4 in cue 3, type:

```
MA User name[Fixture]> Delete Cue 3 If Channel 4
```



- To delete attribute "Pan" of fixture 4 in cue 3, type:

```
MA User name[Fixture]> Delete Cue 3 If Fixture 4 Attribute "Pan"
```

- To clone fixture 1 to fixture 2 only in sequence 1, type:

```
MA User name[Fixture]> Clone Fixture 1 At Fixture 2 If Sequence 1
```

8.3.142. IfActive Keyword

To enter the IfActive keyword in the command line, use one of the options:

- Press **If** **If**
- Type **IfActive**
- Type the shortcut **Ifa**

Description

IfActive is a function keyword that selects fixtures with active values in the programmer.



Important:

Executed on its own, the IfActive keyword only selects fixtures of the current command line default.



Important:

IfActive only works with dimmer values.

If no filter is entered, IfActive will select all fixtures with active values.

If a filter is entered, IfActive will select all fixtures which have this filter and also fixtures that have active values in the programmer.

Syntax

IfActive ([Object] ["Object_Name" or Object_Number])

Examples

- To select all fixtures with active values in the programmer, type:

```
MA User name[Fixture]> IfActive
```

- To select fixtures within group 5 which also have active values in the programmer, type:

```
MA User name[Fixture]> IfActive Group 5
```

8.3.143. IfOutput Keyword

To enter the IfOutput keyword in the command line, use one of the options:

- Press **If**
- Type **IfOutput**
- Type the shortcut **IfO**

Description

IfOutput is a function keyword that selects fixtures based on their current output. This function works with presets and sequences only.



Important:

Executed on its own, the IfOutput keyword only selects fixtures of the current command line default.

Syntax

IfOutput ([Object] ["Object_Name" or Object_Number])

Examples

- To select all fixtures of the current command line default which have output on stage, type:

```
MA User name[Fixture]> IfOutput
```

- To select all fixtures that output the values stored in sequence 1, type:

```
MA User name[Fixture]> IfOutput Sequence 1
```

- To select all fixtures that use the color preset "Green", type:

```
MA User name[Fixture]> IfOutput Preset "Color"."Green"
```


8.3.144. IfProgrammer Keyword

To enter the IfProgrammer keyword in the command line, use one of the options:

- Press **If If If**
- Type **IfProgrammer**
- Type the shortcut **Ifp**

Description

IfProgrammer is a function keyword that selects fixtures that contain values in the programmer.



Important:

Executed on its own, the IfProgrammer keyword only selects fixtures of the current command line default.



Important:

IfProgrammer only works with dimmer values.

If no filter is entered, IfProgrammer selects all fixtures containing values in the programmer.

If a filter is entered, IfProgrammer selects the fixtures which are in the filter and also which contain values in the programmer.

Syntax

IfProgrammer ([Object] ["Object_Name" or Object_Number])

Examples

- To select all fixtures containing values in the programmer, type:

```
MA User name[Fixture]> IfProgrammer
```

- To only select the fixtures of group 5 which contain values in the programmer, type:

```
MA User name[Fixture]> IfProgrammer Group 5
```



8.3.145. Image Keyword

To enter the Image keyword in the command line, use one of the options:

- Type **Image**
- Type the shortcut **Ima**

Description

The keyword Image is an object keyword. It is used to manage images using the command line.

Syntax

[Function] Image [ImagePool_Number].[Image_Number]

Examples

- To edit image 1 in the custom image pool, type:

```
MA User name[Fixture]> Edit Image 3.1
```

- To delete image 2 in the custom image pool, type:

```
MA User name[Fixture]> Delete Image 3.2
```



8.3.146. Import Keyword

To enter the Import keyword in the command line, use one of the options:

- Type **Import**
- Type the shortcut **I**

Description

Import is a function keyword that incorporates small portions of exported show file data, available as XML files, into the current show file.

The Import command loads data into the specified destination occupying the first free pool object when no certain pool object is given.

By default, files will be imported from the relevant folder within the library folder structure, either on the local drive of the console or on PC station, or on a selected USB flash drive. For more information on this folder structure see **Folder Structure**.

Syntax

Import [Object] Library "File Name.xml" (If Drive [Drive_Number]) (At ["Object_Name" or Object_Number]) (/Option) ("Option_Value")

Option Keywords

The Import keyword uses the following option keywords:

- **/Path**
- **/Gaps**
- **/NoDependencies**
- **/NoRefresh**
- **/Type**

Examples

Requirements: Destination is changed to fixture types.

- To import a MAC Aura XB as a new fixture type in the show file, type:

```
 User name@ShowData/Patch/FixtureTypes> Import Library "MAC Aura XB"
```

Requirements: Destination is changed to macros.

- To import the macro color.xml to the first free pool object, type:



```
MA User name@ShowData/DataPools/Default/Macros> Import Library "color.xml"
```

- To import the same macro to macro 42 in the macro pool without changing the command line destination, type:

```
MA User name[Fixture]> Import Macro Library "color.xml" At Macro 42
```

Requirements: Destination is changed to macros.

- To import all macros from the library into the show file, type:

```
MA User name@ShowData/DataPools/Default/Macros> Import Library "*.xml"
```



8.3.147. Index Keyword

To enter the Index keyword in the command line, use one of the options:

- Type **Index**
- Type the shortcut **Ind**

Description

Index keyword is a command keyword that addresses values where a property can have more than one value. For example, a fixture with at least two DMX breaks. The patch property has 2 values for this fixture.

The Index Keyword is zero nominated.

Syntax

Set [Object] ["Object_Name" or Object_Number] ["Property"] Index [Index_Number] ["Value"]

Example

Requierment:

- Fixture 1 has to have at least two DMX breaks.
For example, the fixture type LED Wall 20x20
- To set the patch address of the second break to universe 2 address 1 without affecting the first address, type:

```
MA User name[Fixture]> Set Fixture 1 "Patch" Index 1 "2.1"
```



8.3.148. Insert Keyword

To enter the **Insert** keyword in the command line, use one of the options:

- Press `Copy Copy Copy`
- Type **Insert**
- Type the shortcut **Ins**

Description

The Insert keyword is a function keyword which is used to insert pool objects between two other and already occupied pool objects. The following pool objects will be moved to the next empty destination.

Syntax

Insert [Object] ["Object_Name" or Object_Number] At ["Object_Name" or Object_Number]

Example

- To move Group 5 onto Group 9 and move the latter one cell further, type:

```
MA User name[Fixture]> Insert Group 5 At 9
```



8.3.149. Integrate Keyword

To enter the Integrate keyword in the command line, use one of the options:

- Press **MA** + **At**
- Type **Integrate**
- Type the shortcut **Int**

Description

The Integrate keyword integrates step 1 of a preset into other steps of the programmer.

These steps can be stored into presets and cues.

Syntax

Integrate Preset ["FeatureGroup_Name" or FeatureGroup_Number],["Preset_Name" or Preset_Number] Step [Step_Number]

Examples

Requirement: Example 1

- There are already active values in step 1 in the programmer
- To integrate step 1 of preset 1.1 into step 2 of the programmer, type:

```
MA User name[Fixture]> Integrate Preset 1.1 Step 2
```

Requirement: Example 2

- Select fixtures and create preset 4.5
- Follow steps 1 to 4

1. To call preset 4.5 into the programmer, type:

```
MA User name[Fixture]> At Preset 4.5
```

2. To create the next step, type:



```
MA User name[Fixture]> Next Step
```

3. To integrate step 1 of preset 4.2, type:

```
MA User name[Fixture]> Integrate Preset 4.2
```

4. To store the changes in the preset, type:

```
MA User name[Fixture]> Store Preset 4.5
```




8.3.150. Interface Keyword

To enter the Interface keyword in the command line, use one of the options:

- Type **Interface**
- Type the shortcut **Inter**

Description

Interface is an object keyword which is used to set the network interface using the command line.

For more information see [Interfaces and IP](#).

Syntax

[Function] Interface

Interface [Number]

Examples

- To list all network interfaces, enter the Interface folder in the data structure first:

```
MA User name[Fixture]> ChangeDestination Interface
```

- To list the interfaces, type:

```
MA User name@Interfaces> List
```



8.3.151. Invert Keyword

To enter the **Invert** keyword in the command line, use one of the options:

- Type **Invert**
- Type the shortcut **Inve**

Description

Invert is a function to invert the selection status of fixtures.

If a fixture is selected, using this keyword deselects the fixtures. If a fixture is not selected, using the keyword Invert selects the fixtures.

Syntax

Invert Fixture ["Fixture_Name" or Fixture_Number]

Example

Requirement: Fixtures 1, 3 and 5 are selected

- To invert the state of selection in the fixtures 1 to 6, type:

```
MA User name[Fixture]> Invert Fixture 1 Thru 6
```

8.3.152. Invite Keyword

To enter the Invite keyword in the command line, use one of the options:

- Type **Invite**
- Type the shortcut **Inv**

Description

Invite is a function keyword which is used to invite other stations into the session.

This keyword belongs to a session-related group of keywords.

For more information see:

- [Console keyword](#)
- [Node keyword](#)
- [OnPC keyword](#)
- [ProcessingUnit keyword](#)
- [Station keyword](#)
- [Extension keyword](#)
- [Join keyword](#)
- [Leave keyword](#)
- [Dismiss keyword](#)
- [Reboot keyword](#)
- [Restart keyword](#)
- [ShutDown keyword](#)

Syntax

Invite IP [Device_IP]

Invite [Device_Type] ["DeviceType_Name" or DeviceType_Number]



Hint:

The device types are: **Console**, **Node**, **onPC**, **ProcessingUnit**, **Station**, **Extension**.

Examples

- To invite console 6 to the session, type:

```
MA User name[Fixture]> Invite Console 6
```



- To invite the node named "Truss" to the session, type:

```
MA User name[Fixture]> Invite Node "Truss"
```

- To invite the station with the IP address 192.168.0.10, type:

```
MA User name[Fixture]> Invite IP 192.168.0.10
```



8.3.153. IP Keyword

To enter the IP keyword in the command line, type **IP**.

Description

IP is an object keyword used to manage devices with IP addresses.

It interacts with other function keywords and the IP address of the corresponding device.

Syntax

[Function] IP [IP Address]

Examples

For information on IP examples see:

- **Dismiss keyword**
- **Reboot keyword**
- **RemoteCommand keyword**
- **ShutDown keyword**



8.3.154. JoinSession Keyword

To enter the JoinSession keyword in the command line, use one of the options:

- Type **JoinSession**
- Type the shortcut **J**

Description

JoinSession is a function keyword which is used to join a session.

This keyword belongs to a session-related group of keywords.

For more information see:

- [Console keyword](#)
- [Node keyword](#)
- [OnPC keyword](#)
- [ProcessingUnit keyword](#)
- [Station keyword](#)
- [Extension keyword](#)
- [Dismiss keyword](#)
- [Invite keyword](#)
- [LeaveSession keyword](#)
- [Reboot keyword](#)
- [Restart keyword](#)
- [ShutDown keyword](#)

Syntax

JoinSession IP [Device_IP]

JoinSession [DeviceType] [DeviceType_ID]



Hint:

The device types are: **Console**, **Node**, **onPC**, **ProcessingUnit**, **Station**, **Extension**.

Examples

- To join the session of console 6, type:

```
MA User name[Fixture]> JoinSession Console 6
```



- To join the session on the station with the IP address 192.168.10.21:

```
MA User name[Fixture]> JoinSession IP 192.168.10.21
```

- To join the session of the node "Truss", type:

```
MA User name[Fixture]> JoinSession Node "Truss"
```



8.3.155. Key Keyword

To enter the Key keyword in the command line, use one of the options:

- Type **Key**
- Type the shortcut **K**

Description

The Key keyword is used to address the network keys.

For more information on what network keys are see [Create a Custom Key](#).

Syntax

[Function] Key ["Key_Name" or Key_Number] (Property ["Property_Name"] ["Value"])

Examples

- To store a new key, type:

```
MA User name[Fixture]> Store Key 2
```

- To list all keys, type:

```
MA User name[Fixture]> List Key
```

- To set a different password for the newly created key, type:

```
MA User name[Fixture]> Set Key 2 Property "Seed" "Concord Dawn"
```

- To not use key 2 for MAnet, type:

```
MA User name[Fixture]> Set Key 2 Property "MANET" "No"
```




8.3.156. Keyboard Keyword

To enter the Keyboard keyword in the command line, use one of the options:

- Type **Keyboard**
- Type the shortcut **Keyb**

Description

Keyboard is an object keyword which is used to change the language of the on-screen keyboard.

Syntax

[Function] Keyboard

Keyboard ["Keyboard_Language" or Keyboard_Number]

Examples

- To list the keyboard languages, type:

```
MA User name[Fixture]> List Keyboard
```

- To set the keyboard to German, type:

```
MA User name[Fixture]> Keyboard 1
```

- To set the keyboard to English, type:

```
MA User name[Fixture]> Keyboard "English"
```



8.3.157. KeyboardShortcut Keyword

To enter the KeyboardShortcuts keyword in the command line, use one of the options:

- Type **KeyboardShortcut**
- Type the shortcut **KeyboardS**

Description

The KeyboardShortcut keyword is an object keyword that is used to work with keyboard shortcuts using the command line.

Syntax

[Function] KeyboardShortcut [KeyboardShortcut_Number] (Property ["Property_Name"] ["Value"])

Examples

- To list all keyboard shortcuts, type:

```
MA User name[Fixture]> List KeyboardShortcut
```

- To assign, for example, the property "Shortcut" to the value "Space", type:

```
MA User name[Fixture]> Set KeyboardShortcut 161 Property "Shortcut" "Space"
```

- To delete the keyboard shortcut number 160, type:

```
MA User name[Fixture]> Delete KeyboardShortcut 160
```



8.3.158. Kill Keyword

To enter the **Kill** keyword in the command line, use one of the options:

- Press **MA** + **Go-** + **Go-**
- Type **Kill**
- Type the shortcut **Ki**

Description

The Kill keyword is a function keyword that executes the Go+ function on sequences and presets and disables all other played back sequences and presets.

If the sequences and presets are not protected against kill, they will be disabled by the Kill keyword.

For more information on Kill Protect see [Sequence Settings](#).

Syntax

Kill [Object] ["Object_Name" or Object_Number]

Example

- To jump to the next cue on executor 102 and disable all other playbacks, type:

```
MA User name[Fixture]> Kill Executor 102
```



8.3.159. KnockIn Keyword

To enter the KnockIn keyword in the command line, use one of the options:

- Type **KnockIn**
- Type the shortcut **Kn**

Description

KnockIn is used to take values that are located for example in sequences or presets actively into the programmer.

This is useful to the effect that you can actively take values that have already been stored into the programmer.

Syntax

Fixture ["Fixture_Name" or Fixture_Number] KnockIn ["Attribute_Name" or Attribute_Number]/["FeatureGroup_Name" or FeatureGroup_Number]

KnockIn Fixture ["Fixture_Name" or Fixture_Number]

Attribute ["Attribute_Name" or Attribute_Number] At KnockIn

Examples

- To knock in the attribute Color RGB_R in fixture 1, type:

```
MA User name[Fixture]> Fixture 1 KnockIn Attribute "ColorRGB_R"
```

- To knock in all attributes of fixture 2, type:

```
MA User name[Fixture]> KnockIn Fixture 2
```

- To knock in only the dimmer values of fixtures 11 through 15, type:

```
MA User name[Fixture]> Fixture 11 Thru 15 At KnockIn
```

8.3.160. KnockOut Keyword

To enter the Off keyword in the command line, use one of the options:

- Type **KnockOut**
- Type the shortcut **Knocko**



Hint:

The KnockOut keyword is used as a synonym of the Off Keyword.

Description

The KnockOut keyword is used as a function and value keyword to:

- Knock out selections in the programmer
- Knock out active attributes in the programmer

Syntax

Fixture ["Fixture_Name" or Fixture_Number] KnockOut ["Attribute_Name" or Attribute_Number]/["FeatureGroup_Name" or FeatureGroup_Number]

KnockOut Fixture ["Fixture_Name" or Fixture_Number]

Attribute ["Attribute_Name" or Attribute_Number] At KnockOut

Examples

- To knock out the parameters of fixture 2 and 4 in the programmer, type:

```
MA User name[Fixture]> KnockOut Fixture 2 + 4
```

- To knock out the attribute Color RGB_R in the selected fixtures, type:

```
MA User name[Fixture]> Attribute "ColorRGB_R" At KnockOut
```

- To knock out the dimmer attribute in fixture 5, type:

```
MA User name[Fixture]> Fixture 5 At KnockOut
```

8.3.161. Label Keyword

To enter the Label keyword in the command line, use one of the options:

- Press **Assign** **Assign**
- Type **Label**
- Type the shortcut **L**

Description

The Label keyword is used to give objects a name.

If multiple objects are labeled, and the name contains a free-standing number, the number will be enumerated for each object.

If you do not label an object, a pop-up appears.

Syntax

Label [Object] ["Object_Name" or Object_Number] ["Name"]



Important:

The name must not contain the following characters: \ " \$ & * ? , . ; ^ { | } ~

Examples

- To label group 3 "Higgs Boson", type:

```
MA User name[Fixture]> Label Group 3 "Higgs Boson"
```

- To label fixtures 1 to 10 as "Mac700 1", "Mac700 2" and so on, type:

```
MA User name[Fixture]> Label Fixture 1 Thru 10 "Mac700 1"
```

- To rename the color preset "Red" to "Dark Red", type:

```
MA User name[Fixture]> Label Preset "Color"."Red" "Dark Red"
```



8.3.162. Layout Keyword

To enter the **Layout Keyword** in the command line, use one of the options:

- Press **MA** + **X4**
- Type **Layout**
- Type the shortcut **Lay**

Description

Layout is an object keyword representing a layout of fixtures and other objects.

Syntax

[Function] Layout ["Layout_Name" or Layout_Number]

Examples

- To create layout 5 and add fixture selection to this layout, type:

```
MA User name[Fixture]> Assign Layout 5
```

- To add group 5 as a button in layout 4, type:

```
MA User name[Fixture]> Assign Group 5 At Layout 4
```



8.3.163. LearnSpeed Keyword

To enter the LearnSpeed keyword in the command line, use one of the options:

- Press **Learn | Rate1**
- Type **LearnSpeed**
- Type the shortcut **Learn**

Description

The LearnSpeed keyword is a playback keyword that is used to set the speed in phasers.

For more information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

LearnSpeed [Object] ["Object_Name" or Object_Number]

Example

To set the speed in a phaser:

- Press and hold **Learn | Rate1** and press the executor key several times using the speed you would like to have your phaser at.

The phaser learns the speed you press the executor key at.



8.3.164. LeaveSession Keyword

To enter the LeaveSession keyword in the command line, use one of the options:

- Type **LeaveSession**
- Type **Leave**
- Type the shortcut **Le**

Description

LeaveSession is a function keyword which is used to leave the current session.

Other session related keywords are:

- **JoinSession**
- **Invite**
- **Dismiss**

Syntax

LeaveSession

Example

- To leave the current session, type:

```
MA User name[Fixture]> LeaveSession
```



8.3.165. Library Keyword

To enter the **Library** keyword in the command line, use one of the options:

- Type **Library**
- Type the shortcut **Lib**

Description

Library is an object keyword that is used to access the corresponding library folder down to the root on the hard drive.

Syntax

[Function] Library

Examples

Requirements: Change destination to macros.

For more information see [ChangeDestination Keyword](#).

- To list all macros stored in the macro library, type:

```
MA Admin@Root/ShowData/Pool/Global/Macros> List Library
```

- To import the first macro library, type:

```
MA Admin@Root/ShowData/Pool/Global/Macros> Import Library 1
```

Requirements: Change destination to fixture types.

For more information see [ChangeDestination Keyword](#).

- To list all fixture types stored in the fixture types library, type:

```
MA Admin@Root/ShowData/LivePatch/FixtureTypes> List Library
```

8.3.166. List Keyword

To enter the List keyword in the command line, use one of these options:

- Press **List**
- Type **List**
- Type the shortcut **Li**

Description

The List keyword displays show data, including objects and their associated properties, in the command line history.

It is possible to list any object or any property of:

- Cues of the selected executor
- Groups
- Presets

If the list command does not specify any type of object, the data of the current command line destination is displayed.

The List command can also display the contents of library folders on the console, onPC station, or connected USB drives.



Hint:

The /Path option instructs the software to access folders other than the defaults. For more information on the default folder structure, see the **Folder Structure** topic.

The List keyword is a function keyword.

Syntax

List [Object] ["Object_Name" or Object_Number]

List [Object] Library (If Drive [Drive_Number]) (/Path [FolderPath]) (/Type [FileType])



Important:

When specifying a type, the word "System" or "User" must be capitalized. For more information see the option keyword **/Type**.

Examples

- To list the first ten fixtures of the Fixture Edit Setup, type:



```
MA User name[Fixture]> List Fixture Thru 10
```

- To list all existing attributes in the show file, type:

```
MA User name[Fixture]> List Attribute
```

- To list the first 5 groups of the group pool, type:

```
MA User name[Fixture]> List Group Thru 5
```

- To list the macros available to import from the default macro library folder on the first connected USB drive, type:

```
MA User name[Fixture]> List Macro Library If Drive 2
```

Example - List Properties of a Parent and Their Children's Properties



Important:

In case the children of an object should have various properties, they will be listed using names and numbers when executing List within the parent object.

To display the properties of a child directly, you have to specifically list the child.

However, the steps described here can also be used in other constellations consisting of a parent and their children.

In the following example we use StationSettings as the parent and DeskLightsCollect as one of their children.

1. Change the destination of the command line first:

```
MA User name[Fixture]> ChangeDestination StationSettings
```

Result:

```
MA User name@StationSettings>
```

2. To list the properties of the station settings, type:

```
MA User name@StationSettings> List
```

The children are now displayed in the command line history:

```
OK:changeDestination "stationSettings"  
LOCK NO      NAME  
      1      TimeConfig  
S      2      LocalSettings  
S      3 (2)  DefaultDisplayPositionsCollect  
S      4 (5)  DeskLightsCollect  
      5      DisplaySurfaces  
      6 (5)  GridColumnRegistry  
OK:List  
OK:Menu "commandLineHistory"
```

The children of the Station Settings are listed

3. To list the properties of one child, for example DeskLightsCollect, type:

```
MA User name@StationSettings> List DeskLightsCollect
```



The properties of the child DeskLightsCollect are now displayed in the command line history:

```
OK>List "deskLightcollect"
LOCK NO      NAME                      LEDFEEDBACKDURATION
S      4 (5) deskLightscollect 30
OK>List "deskLightscollect"
```

The properties of DeskLightsCollect are listed

- To access the folder DeskLightsCollect within Station Settings, type:

```
MA User name@StationSettings> ChangeDestination DeskLightsCollect
```

Result:

```
MA User name@StationSettings/DeskLightsCollect>
```

- To list the properties of the children of DeskLightsCollect, type:

```
MA User name@StationSettings/DeskLightsCollect> List
```

The children's properties are now displayed.

```
LOCK NO NAME      MASTER      MASTERADDRESS LEDENCODER LEDFADER LEDEXEC LEDKEYBOARD
1 DeskLights % 50.0000029802 % 0.0000000000 % 0.0000000000 % 0.0000000000 % 0.0000000000
2 Screens % 100.0000059605 % 0.0000000000 % 0.0000000000 % 0.0000000000 % 0.0000000000
3 LedMaster % 100.0000059605 % 100.0000059605 % 100.0000059605 % 100.0000059605 % 100.0000059605
4 LedBackground % 50.0000029802 % 0.0000000000 % 0.0000000000 % 0.0000000000 % 50.0000029802
5 LedFeedback % 50.0000029802 % 100.0000059605 % 50.0000029802 % 100.0000059605 % 100.0000059605
OK>List
```

Properties of DeskLightsCollect listed in the command line history



8.3.167. ListOwnership Keyword

To enter the ListOwnership keyword in the command line, use one of the options:

- Type **ListOwnership**
- Type the shortcut **Listo**

Description

ListOwnership is used as a troubleshooting keyword in case of a multi-user access conflict.

If a multi-user access conflict occurs, use the ListOwnership keyword. It lists the objects that are currently locked for all users in the session, hence causing the conflict.

Syntax

ListOwnership ([Object] ["Object_Name" or Object_Number])

Examples

- To display all objects that are locked for all users and that might cause a multi-user conflict, type:

```
MA User name[Fixture]> ListOwnership
```

- To display the ownership in macro 1 line 1, type:

```
MA User name[Fixture]> ListOwnership Macro 1
```

or type:

```
MA User name[Fixture]> ListOwnership Macro 1.1
```

or:

```
MA User name[Fixture]> ListOwnership Macro 1.*
```



8.3.168. ListReference Keyword

To enter the ListReference keyword in the command line, use one of the options:

- Press **MA** + **List**
- Type **ListReference**
- Type the shortcut **Listr**

Description

ListReference is used to display references and/or dependencies in your show file.

Examples of references between objects, only to mention a few:

- Layout to group, to macro, or to fixture
- Preset to cue, to preset (embedded preset)

These references and/or dependencies depend on the use of the object in the show file.

Syntax

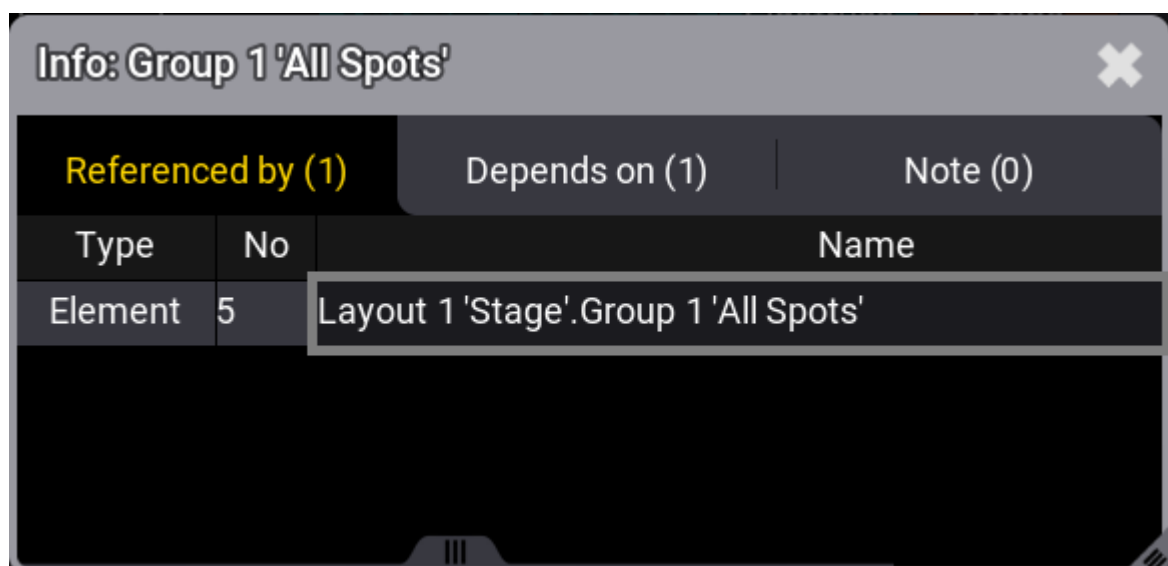
ListReference [Object] ["Object_Name" or Object_Number]

Example

- To list the references and dependencies of group 1, type:

```
MA User name[Fixture]> ListReference Group 1
```

The info pop-up opens listing references and/or dependencies:



Current references, dependencies, or notes





8.3.169. Load Keyword

To enter the Load keyword in the command line, use one of the options:

- Press **Goto** **Goto**
- Type **Load**

Description

The Load keyword is a playback keyword which is used to prepare an executor to jump to another cue rather than jumping to the next cue when a **Go+** is performed on the executor.

For more information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

Load [Object] ["Object_Name" or Object_Number]

Examples

- To load cue 3 on the **selected** executor, type:

```
MA User name[Fixture]> Load Cue 3
```

Cue 3 is loaded. To indicate that a cue is loaded, the display toggles between Cue 3 and Cue "Name".

- To load cue 5 on executor 114, type:

```
MA User name[Fixture]> Load Executor 114 Cue 5
```

- To load cue +2 on executor 114, type:

```
MA User name[Fixture]> Load +2 Executor 114
```

- To load the previous cue on executor 114, type:

```
MA User name[Fixture]> Load Previous Executor 114
```



8.3.170. Loaded Keyword



Important:

MA + **Go+ [large]** immediately executes the **Go+ Loaded** command.

To enter the Loaded keyword in the command line, use one of the options:

- Press **MA** + **Goto** **Goto**
- Type **Loaded**

Description

The Loaded keyword is a function keyword which is used to simultaneously handle several cues in a sequence that are in the Load status.

For more information on Load see the [Load Keyword](#).

Syntax

[Function] Loaded

Examples

Requirement: At least one cue has to be in the Load status.

- To start several loaded cues, type:

```
MA User name[Fixture]> Go+ Loaded
```

- To disable sequences that are loaded, type:

```
MA User name[Fixture]> Off Loaded
```



8.3.171. LoadShow Keyword

To enter the LoadShow keyword in the command line, use one of the options:

- Type **LoadShow**
- Type the shortcut **Loa**

Description

The LoadShow keyword is a function keyword that loads a show from the folders: shows, demo shows, or backup.

For more information on the folder structure of shows, demo shows, and backups, see [Show File Handling](#) and [Folder Structure](#).

Syntax

LoadShow ["Show_Name"] (/Option)


Option Keywords

The LoadShow keyword uses the following option keywords:


- **/All**
- **/DMXProtocols**
- **/LocalSettings**
- **/NoSave**
- **/NoShowData**
- **/OutputStations**
- **/Path**
- **/Save**
- **/Type**

Examples

- To load the show file with the file name "MacBeth", type:

```
 User name[Fixture]> LoadShow "MacBeth"
```

- To create a new show file with the name "William", type:

```
 User name[Fixture]> LoadShow "William"
```



Important:

To load the new show again, save the new show file after loading it. For more information see **SaveShow keyword**.



8.3.172. Lock Keyword

To enter the Lock keyword in the command line, use one of the options:

- Type **Lock**
- Type the shortcut **Loc**

Description

The Lock keyword is a function keyword that locks objects.

Objects that are locked are indicated by a padlock.

Syntax

Lock [Object] ["Object_Name" or Object_Number]

Examples

- To lock macro 1, type:

```
MA User name[Fixture]> Lock Macro 1
```

To unlock objects see the **Unlock Keyword**.

- To lock the second color preset 2, type:

```
MA User name[Fixture]> Lock Preset 4.2
```

- To lock sequence 3, type:

```
MA User name[Fixture]> Lock Sequence 3
```



8.3.173. LogIn Keyword

To enter the LogIn keyword in the command line, use one of the options:

- Type **LogIn**
- Type the shortcut **Log**

Description

The LogIn keyword is a function keyword to log in another user.

Syntax

LogIn ["User_Name"] ["Password"]

Example

To log out the current user and log in as user Jimmy Page with the password mac, type:

```
MA User name[Fixture]> LogIn "Jimmy Page" "mac"
```



8.3.174. LogOut Keyword

To enter the LogOut keyword in the command line, use one of the options:

- Type **LogOut**
- Type the shortcut **Logo**

Description

The LogOut keyword is a function keyword which is used to log out the user and change to guest user.

Syntax

LogOut

Example

- To log out the user and change to guest user, type:

```
MA User name[Fixture]> LogOut
```




8.3.175. Lowlight Keyword

To enter the Lowlight keyword in the command line, use one of the options:

- Press **MA** + **Highlt**
- Type **Lowlight**
- Type the shortcut **Low**

Description

The Lowlight keyword is a playback keyword that is used to apply the defined lowlight values in fixtures that are subselected when highlight is enabled.

Lowlight is also a toggle function meaning that entering Lowlight without using a helping keyword enables or disables the Lowlight mode.

For more information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

(Function) Lowlight (On/Off)

Example

- To apply the lowlight values to selected fixtures, type:

```
MA User name[Fixture]> Lowlight
```

Example: Special Values

Requirement:

Activate values in the programmer:

- Call a preset or turn the attribute encoders.

- To store the currently active values as lowlight values, type:

```
MA User name[Fixture]> Store Lowlight
```

For more information on the special values see [Parameter List](#).



8.3.176. Lua Keyword

To enter the Lua keyword in the command line, type **Lua**.

Description

The Lua keyword is used to execute commands in the script language Lua.

After entering Lua into the command line, type in the script language Lua version 5.4. The grandMA3 will directly execute the commands written in Lua.

To learn more about the grandMA3 specific Lua functions, see [Plugins](#) .

Syntax

Lua ["LuaCode"]

Examples

- To output "Hello World" in the Command Line History, type:

```
MA User name[Fixture]> Lua "Printf('Hello World')"
```

Result:

```
Hello World
```

- To output "Hello World" in the system monitor, type:

```
MA User name[Fixture]> Lua "Echo('Hello World')"
```

Result:

```
Hello World
```



8.3.177. LuaFile Keyword

To enter the LuaFile keyword in the command line, use one of the options:

- Type **LuaFile**
- Type the shortcut **LuaF**

Description

The LuaFile keyword is used to execute a Lua file directly without the need to import it into the grandMA3 show file.

Syntax

LuaFile ["AbsolutePathToLuaFile/FileName.lua"]

Example

- To execute the Lua file "execute_example.lua", type:

```
MA User name[Fixture]> LuaFile  
"C:\ProgramData\MALightingTechnology\gma3_1.6.3\shared\resource\lib_plug
```



8.3.178. Macro Keyword

To enter the **Macro** keyword in the command line, use one of the options:

- Press **MA** + **X14**
- Type **Macro**
- Type the shortcut **M**

Description

The Macro keyword is an object keyword which is used to access Macros.

The default function for macro is Go+. This means calling macros without specifying any function starts the macro.

For more information see [Macros](#).

Syntax

[Function] Macro ["Macro_Name" or Macro_Number]

[Function] DataPool ["DataPool_Name" or DataPool_Number] Macro ["Macro_Name" or Macro_Number]

Examples

- To start macro 5, type:

```
MA User name[Fixture]> Macro 5
```

- To set the wait time of macro 3 line 4 to Go, type:

```
MA User name[Fixture]> Edit Macro 3.4 "Wait" "Go"
```

- To store a new and empty macro 2, type:

```
MA User name[Fixture]> Store Macro 2
```



8.3.179. MArker Keyword

To enter the MArker keyword in the command line, use one of the options:

- Press **Channel** until MArker appears in the command line
- Type **MArker**
- Type the shortcut **MAr**



Hint:

To use the MArker keyword, make sure you assign the MArker ID type and a custom ID to a fixture first.

Description

The MArker keyword is an object keyword that is used to select fixtures of the ID type MArker.

Syntax

MArker ["MArker_Name" or MArker_Number]

Example

- To select MArker 30, type:

```
 User name[Fixture]> MArker 30
```



8.3.180. Master Keyword

To enter the **Master** keyword in the command line, use one of the options:

- Type **Master**
- Type the shortcut **Mas**

Description

The Master keyword is an object keyword that is used to address the different master functions, e.g., for the selected sequence, grand masters, speed masters and playback masters.

Syntax

[Function] Master ["MasterCategory_Name" or MasterCategory_Number].["Master_Name" or Master_Number]

Examples

- To assign the master to the selected sequence on executor 206, type:

```
MA User name[Fixture]> Assign Master 1.1 At Executor 206
```

- To assign the grand master to executor 207, type:

```
MA User name[Fixture]> Assign Master 2.1 At Executor 207
```

- To label the second speed master "Great Speed", type:

```
MA User name[Fixture]> Label Master 3.2 "Great Speed"
```

- To assign the playback master 3 to executor 209 on page 6, type:

```
MA User name[Fixture]> Assign Master 4.3 At Page 6.209
```

For more information on how to assign objects to executors see [Assign Object to an Executor](#).



8.3.181. MATricks Keyword

To enter the MATricks keyword in the command line, use one of the following options:

- Type **MATricks**
- Type the shortcut **MA**t

Description

The MATricks keyword behaves as an object type.

Used with an ID, MATricks represents MATricks objects stored in the MATricks pool.

With the helping keywords On, Off, and Toggle, the MATricks of the two selections may temporarily be enabled or disabled.

Furthermore, you can set the values of the MATricks of the two selections.

Syntax

Set Selection (Selection_Number) MATricks (Property) ["Property_Name" or Property_Value]

[Function] MATricks ["MATricks_Name" or MATricks_Number]

Properties

X	Y	Z
XBlock	YBlock	ZBlock
XGroup	YGroup	ZGroup
XWings	YWings	ZWings
XWidth	YWidth	ZWidth
XShuffle	YShuffle	ZShuffle
XShift	YShift	ZShift

Examples

- To set the MATricks X to 2 in the active selection, type:

```
MA User name[Fixture]> Set Selection MATricks "X" 2
```

- To set the MATricks XBlock to 4 in selection 2, type:

```
MA User name[Fixture]> Set Selection 2 MATricks "XBlock" 4
```



- To disable MATricks in the active selection, type:

```
MA User name[Fixture]> Off Selection MATricks
```

- Toggle to activate MATricks in the active selection, press **Set** or type:

```
MA User name[Fixture]> Toggle Selection MATricks
```

- To reset the MATricks in the first selection, type:

```
MA User name[Fixture]> Reset Selection 1 MATricks
```

- To call the first MATricks object in the MATricks pool, type:

```
MA User name[Fixture]> Call MATricks 1
```

- To label MATricks 2 "Great", type:

```
MA User name[Fixture]> Label MATricks 2 "Great"
```

- To assign the fourth MATricks object of the pool to the first recipe in cue 1 part 0 of the selected sequence, type:

```
MA User name[Fixture]> Assign MATricks 4 At Cue 1 Part 0.1
```


8.3.182. Measure Keyword

To enter the Measure keyword in the command line, use one of the options:

- Type **Measure**
- Type the shortcut **Mea**

Description

The Measure keyword is used together with the phaser speed (speed layer) to define the length of time of a phaser.

For more information see [Phasers](#).



Hint:

The measure layer also affects how the width of each step is calculated.



Important:

When using the measure layer, multiple width value combinations can produce identical results. To ensure predictable timing, it is recommended to consider the width value of a step as the percentage of a beat and ensure that the total width of all steps in the phaser equals the number of beats specified in the measure layer.

Syntax

Measure


[At] Measure [Value]

Examples

Requirement:

To set values in the measure layer, create at least 2 steps in the programmer.

- To set the selected layer to measure, type:

```
 User name[Fixture]> Measure
```

- To set the measure of the selected feature group to 4 beats, type:



```
MA User name[Fixture]> At Measure 4
```

Result:

The phaser cycle now lasts 4 beats.



8.3.183. Media Keyword

To enter the **Media** keyword in the command line, use one of the options:

- Type **F + 5**
- Type **Media**
- Type the shortcut **Med**



Hint:

To use the Media keyword, make sure you assign the Media ID type and a custom ID to a fixture first.

Description

The Media keyword is an object keyword used to call fixtures of the ID type Media in the programmer.

The name of this keyword can change because it is a custom ID type. For more information see the [Custom ID Type](#).

Syntax

Media ["IDType_Name" or IDType_Number]

Example

- To select the fixture that uses the Media ID 2, type:

```
MA User name[Fixture]> Media 2
```



8.3.184. MemoryInfo Keyword

To enter the **MemoryInfo** keyword in the command line, use one of the options:

- Type **MemoryInfo**
- Type the shortcut **Mem**

Description

The MemoryInfo keyword is a command keyword that is used to indicate how much memory is occupied.

Syntax

MemoryInfo

Example

- To check how much memory is occupied, type:

```
MA User name[Fixture]> MemoryInfo
```



8.3.185. Menu Keyword

To enter the **Menu** keyword in the command line, use one of the options:

- Type **Menu**
- Type the shortcut **Men**

Description

The Menu keyword is an object keyword used to address menus.

Syntax

([Function]) Menu ["Menu_Name" or Menu_Number]

Examples

- To open the menu Network, type:

```
MA User name[Fixture]> Menu "Network"
```

- To list all menus, type:

```
MA User name[Fixture]> List Menu
```

The command line feedback window lists all menus you can open using this keyword.

- To display the rear panel connectors on the letterbox screens, type:

```
MA User name[Fixture]> Menu "Connectorview"
```



8.3.186. Mesh Keyword

To enter the **Mesh** keyword in the command line, use one of the options:

- Type **Mesh**

Description

The Mesh keyword is an object keyword which is used to address meshes.

Syntax

[Function] Mesh ["Mesh_Name" or Mesh_Number]

Examples

- To list all meshes of the show file, type:

```
MA User name[Fixture]> List Mesh
```



8.3.187. Move Keyword

To enter the Move keyword in the command line, use one of the options:

- Press **Move**
- Type **Move**
- Type the shortcut **Mo**

Description

The Move keyword is a function keyword used to move objects to another location giving them a new object ID.

If the destination is already occupied, the object will be moved to the target object. The previous target object will be moved to the next object until an empty object will be occupied.

Syntax

Move [Object] ["Object_Name" or Object_Number] At ["Object_Name" or Object_Number]

Example

- To move macro 1 to macro 3, type:

```
MA User name[Fixture]> Move Macro 1 At 3
```

8.3.188. Multipatch Keyword

To enter the Multipatch keyword in the command line, use one of the options:

- Press **Channel** until Multipatch appears in the command line
- Type **Multipatch**
- Type the shortcut **MU**



Hint:

To use the Multipatch keyword, make sure you create a Multipatch in an existing fixture first.

Description

The Multipatch keyword is a keyword which is used to address the multipatch fixtures of a fixture.

Syntax

Multipatch [Absolute_Multipatch_ID]

Fixture ["Fixture_Name" or Fixture_Number] Multipatch [Multipatch_ID]

Examples

- To select the second multipatch fixture of fixture 4, type:

```
MA User name[Fixture]> Fixture 4 Multipatch 2
```

- To patch the third multipatch fixture of fixture 2 to DMX address 6 in DMX universe 42, type:

```
MA User name[Fixture]> Patch Fixture 2 Multipatch 3 42.6
```


8.3.189. MyRunningMacro Keyword

To enter the MyRunningMacro keyword in the command line, use one of the options:

- Type **MyRunningMacro**
- Type the shortcut **MyRunningM**

Description

The MyRunningMacro keyword is used when there are several users in a session.

It allows you to list or disable your own running macros in a show file.



Important:

The numbers of the running macros are not equal to the numbers of the macro in the macros data pool.

Syntax

[Function] MyRunningMacro ["Macro_Name" or Macro_Number]

Examples

- To disable the first macro that is running and which you started, type:

```
MA User name[Fixture]> Off MyRunningMacro 1
```

- To list all your running macros, change destination to the running macros first:

```
MA User name[Fixture]> ChangeDestination MyRunningMacro
```

Result:

```
MA User name@Temp/MyRunningMacros>
```

- To list all your running macros, type:

```
MA User name@Temp/MyRunningMacros> List
```

8.3.190. MyRunningPreset Keyword

To enter the MyRunningPreset keyword in the command line, use one of the options:

- Type **MyRunningPreset**
- Type the shortcut **MyRunningP**

Description

The MyRunningPreset keyword is used when there are several users in a session.

It allows you to disable your own running presets in a show file.



Important:

The numbers of the running presets are not equal to the numbers in the preset pool.

Syntax

[Function] MyRunningPreset ["Preset_Name" or Preset_Number]

Examples

- To disable your second preset of the gobo preset pool that is running, type:

```
MA User name[Fixture]> Off MyRunningPreset 3.2
```

- To list all your running presets, change destination to the running presets first and type:

```
MA User name[Fixture]> ChangeDestination MyRunningPresets
```

Result:

```
MA User name@Temp/MyRunningPresets>
```

- To list all your running presets, type:

```
MA User name@Temp/MyRunningPresets> List
```

8.3.191. MyRunningSequence Keyword

To enter the MyRunningSequence keyword in the command line, use one of the options:

- Type **MyRunningSequence**
- Type the shortcut **My**

Description

The MyRunningSequence keyword is used when there are several users in a session.

It allows you to disable your own running sequences in a show file.



Important:

The numbers of the running sequences are not equal to the numbers of sequences in the sequences data pool.

Syntax

[Function] MyRunningSequence [Number]

Examples

- To disable the first sequence that is running and which you started, type:

```
MA User name[Fixture]> Off MyRunningSequence 1
```

- To list all your running sequences, change destination to the running sequences first:

```
MA User name[Fixture]> ChangeDestination MyRunningSequence
```

Result:

```
MA User name@Temp/MyRunningSequences>
```

- To list all your running sequences, type:

```
MA User name@Temp/MyRunningSequences> List
```

8.3.192. MyRunningTimer Keyword

To enter the MyRunningTimer keyword in the command line, use one of the options:

- Type **MyRunningTimer**
- Type the shortcut **MyRunningT**

Description

The MyRunningTimer keyword is used when there are several users in a session.

It allows you to list or disable your own running timers in a show file.



Important:

The numbers of the running timers are not equal to the numbers of timers in the timers data pool.

Syntax

[Function] MyRunningTimer ["Timer_Name" or Timer_Number]

Examples

- To disable the first timer that is running and which you started, type:

```
MA User name[Fixture]> Off MyRunningTimer 1
```

- To list all your running timers, change destination to the running timers first:

```
MA User name[Fixture]> ChangeDestination MyRunningTimer
```

Result:

```
MA User name@Temp/MyRunningTimers>
```

- To list all your running timers, type:

```
MA User name@Temp/MyRunningTimers> List
```



8.3.193. Natural Keyword

To enter the Natural keyword in the command line, use one of the options:

- Type **Natural**
- Type the shortcut **Na**

Description

The Natural keyword is used to set the values of a fixture selection using the notation that is defined as natural in the different attributes.

For information on how to define which readout is used as natural readout in each attribute in the user interface, see [Attribute Definitions](#).

Syntax

Attribute ["Attribute_Name"] At Natural [Value]

Example

- To set the pan attribute of the selected fixtures to value 26 using the readout that is defined as natural readout, type:

```
MA User name[Fixture]> Attribute "Pan" At Natural 26
```



8.3.194. NDI Keyword

To enter the NDI keyword in the command line, use one of the options:

- Type **NDI**
- Type the shortcut **ND**

Description

The keyword NDI is an object keyword that addresses NDI streams within the NDI folder of the video pool.

For more information on NDI and how to use it see [Video - Use an NDI Source](#).

Syntax

[Function] NDI

Examples

- To list all NDI streams that are used in the show file, type:

```
 User name[Fixture]> List NDI
```



8.3.195. NewShow Keyword

To enter the NewShow keyword in the command line, use one of the options:

- Type **NewShow**
- Type the shortcut **New**

Description

The NewShow keyword is a function keyword that is used to create new shows.

Syntax

NewShow ["Show_Name"] (/Option)

Option Keywords

The NewShow keyword uses the following option keywords:

- **/All**
- **/DMXProtocols**
- **/LocalSettings**
- **/NoShowData**
- **/OutputStations**
- **/Path**

Example

- To create a new show file with the file name "MacBeth", type:

```
MA User name[Fixture]> NewShow "MacBeth"
```



8.3.196. Next Keyword

To enter the **Next** keyword in the command line, use one of the options:

- Press **Next**
- Type **Next**
- Type the shortcut **N**



Important:

To execute the command directly, press **Next**.

Description

If no fixtures are selected and the default keyword is Fixture, the Next keyword selects the fixture with the lowest fixture ID.

If only one fixture is selected and the default keyword is Fixture, the Next keyword selects the following fixture.

If multiple fixtures are selected, the Next keyword selects the following fixture one after the other within the selected block of fixtures.

For more information see [MAticks and Shuffle](#).

Syntax

Next

Option Keywords

The Next keyword uses the following option keywords:

- **/Wrap**

Example

- To step through single fixtures in the selected block of fixtures, press **Next**.



Name	FID	IDType	CID	Dimmer	PanTilt	
				Dim	P	T
AS QWO 1	1	Fixture		60	50	50
AS QWO 2	2	Fixture		60	50	50
AS QWO 3	3	Fixture		60	50	50
AS QWO 4	4	Fixture		60	50	50
AS QWO 5	5	Fixture		60	50	50
AS QWO 6	6	Fixture		60	50	50
AS QWO 7	7	Fixture		60	50	50
AS QWO 8	8	Fixture		60	50	50
AS QWO 9	9	Fixture		60	50	50
AS QWO 10	10	Fixture		60	50	50
AS QWO 11	11	Fixture		0	50	50
AS QWO 12	12	Fixture		0	50	50
AS QWO 13	13	Fixture		0	50	50

Using Next in a selected block of fixtures



8.3.197. NextY Keyword

To enter the **NextY** keyword in the command line, use one of the options:

- Type **NextY**

Description

The NextY keyword is a command keyword that is used to move the main selection from fixture to fixture on the y-axis. This can be observed in the selection grid for example.

Syntax

NextY

Example

Requirement:

- Select more than one fixture on the y-axis:

```
MA User name[Fixture]> Grid 0/0 Thru 4/4
```

```
MA User name[Fixture]> Fixture 1 Thru 25
```

- To move this selection downward, that is, on the y-axis, type:

```
MA User name[Fixture]> NextY
```



8.3.198. NextZ Keyword

To enter the **NextZ** keyword in the command line, use one of the options:

- Type **NextZ**

Description

The NextZ keyword is a command keyword that is used to move the main selection from fixture to fixture on the z-axis. This can be observed in the selection grid for example.

Syntax

NextZ

Example

Requirement:

- Select more than one fixture on the z-axis:

```
MA User name[Fixture]> Grid 0/0/0 Thru 4/0/4
```

```
MA User name[Fixture]> Fixture 1 Thru 25
```

- To move this selection upward, that is, on the z-axis, type:

```
MA User name[Fixture]> NextZ
```



8.3.199. NetworkNode Keyword

To enter the NetworkNode keyword in the command line, use one of the options:

- Type **NetworkNode**
- Type the shortcuts **Net** or **Node**

Description

The NetworkNode keyword is an object keyword which is used to display the grandMA3 nodes in the network.

This keyword belongs to a session-related group of keywords:

- Console keyword
- onPC keyword
- ProcessingUnit keyword
- Station keyword
- Extension keyword
- Join keyword
- Leave keyword
- Invite keyword
- Dismiss keyword
- Reboot keyword
- Restart keyword

Syntax

[Function] NetworkNode ["NetworkNode_Name" or NetworkNode_Number]

Examples

To list all nodes that are currently in the same network, type:

```
MA User name[Fixture]> List NetworkNode
```

To invite node "FOH3" into the session, type:

```
MA User name[Fixture]> Invite NetworkNode "FOH3"
```



8.3.200. NonDim Keyword

To enter the **NonDim** keyword in the command line, use one of the options:

- Press **Channel** until NonDim appears in the command line
- Type **F + 4**
- Type **NonDim**
- Type the shortcut **Non**



Hint:

To use the NonDim keyword, make sure you assign the NonDim ID type and a custom ID to a fixture first.

Description

The NonDim keyword is an object keyword used to call fixtures of the ID type NonDim in the programmer.

The name of this keyword can change because it is a custom ID type. For more information see the [Custom ID Type Topic](#).

Syntax

NonDim ["NonDim_Name" or NonDim_Number]

Example

- To select the fixture that uses the NonDim ID 1, type:

```
MA User name[Fixture]> NonDim 1
```



8.3.201. Normal Keyword



Important:

Pressing **At** twice directly executes the Normal keyword.

To enter the **Normal** keyword in the command line, use one of the options:

- Press **At** **At**
- Type **Normal**
- Type the shortcut **No**

Description

Normal is an object keyword which is used to set the dimmer of the fixtures to the defined normal value.

For information on how to change the normal value see [User Settings](#).

Syntax

Normal

Example

- To set the dimmer of the selected fixtures to the normal value, type:

```
MA User name[Fixture]> Normal
```



8.3.202. Note Keyword

To enter the Note keyword in the command line, use one of the options:

- Type **Note**
- Type the shortcut **Not**

Description

The Note keyword is used to add or edit a short note in an object.

For more information see [Notes](#).

Syntax

Note [Object] ["Object_Name" or Object_Number] ("Text")

Examples

- To add a note in macro 5, type:

```
MA User name[Fixture]> Note Macro 5
```

- To add a note called "Important" in plugin 42, type:

```
MA User name[Fixture]> Note Plugin 42 "Important"
```



8.3.203. Off Keyword

To enter the Off keyword in the command line, use one of the options:

- Press **Off**
- Type **Off**
- Type the shortcut **Of**

Description

The Off keyword is used as a function keyword to:

- Stop an executor
- Knock out parameters in the programmer
- Knock out selections in the programmer
- Knock out active attributes in the programmer

For information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

Off [Object] ["Object_Name" or Object_Number]

Examples

- To knock out the parameters of fixture 2 and 4 in the programmer, type:

```
MA User name[Fixture]> Off Fixture 2 + 4
```

- To knock out the values of sequence 1, type:

```
MA User name[Fixture]> Off Sequence 1
```

- To turn off all sequences containing loaded cues, type:

```
MA User name[Fixture]> Off Loaded
```




8.3.204. Offset Keyword



Hint:

- Store Offset is more precise when the resulting offsets are as small as possible.
- It is also possible to use relative values in the programmer.

To enter the Offset keyword in the command line, use one of the options:

- Press **MA** + **Update**
- Type **Offset**
- Type the shortcut **Offs**

Description

The Offset keyword is a helping keyword that helps calculate the pan and tilt offset in fixtures.

Syntax

[Function] Offset

Examples

Requirement:

1. Call a preset into the programmer or play a cue in the fixtures where offset is to be calculated;
 2. The position of the fixtures in 3D have to be aligned with the fixtures on real stage;
 3. Adjust the pan and tilt values in the programmer so that the fixtures are in the same position on real stage.
- To calculate and use offset in the selected fixtures, type:

```
MA User name[Fixture]> Store Offset
```



8.3.205. On Keyword

To enter the On keyword in the command line, use one of the options:

- Press **On**
- Type **On**

Description

The On keyword is used as a playback or as a helping keyword.

Use the On keyword as a playback keyword to:

- Start or restart an executor
- Activate selection in the programmer
- Activate attributes in the programmer

Use the On keyword as a helping keyword to:

- Indicate the start of a temporary function
- Enable the state of a toggle function

For information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

Function keyword:

On [Object] ["Object_Name" or Object_Number]

Helping keyword:

[Function] On

Examples

- To activate the attributes of fixture 2 and 4 in the programmer, type:

```
MA User name[Fixture]> On Fixture 2 + 4
```

- To activate sequence 3, type:

```
MA User name[Fixture]> On Sequence 3
```



8.3.206. onPC Keyword

To enter the onPC keyword in the command line, use one of the options:

- Type **onPC**
- Type the shortcut **onP**

Description

The onPC keyword is an object keyword which is used to display all onPC stations in the network.

This keyword belongs to a session-related group of keywords:

- **NetworkNode keyword**
- **onPC keyword**
- **ProcessingUnit keyword**
- **Station keyword**
- **Extension keyword**
- **Join keyword**
- **Leave keyword**
- **Invite keyword**
- **Dismiss keyword**
- **Reboot keyword**
- **Restart keyword**
- **ShutDown keyword**

Syntax

[Function] onPC ["onPC_Name" or onPC_Number]

Examples

- To list all onPC stations that are currently in the same network, type:

```
MA User name[Fixture]> List onPC
```

- To invite the onPC station "FOH3" to the session, type:

```
MA User name[Fixture]> Invite onPC "FOH3"
```



8.3.207. OSC Keyword

To enter the **OSC** keyword in the command line, use one of the options:

- Type **OSC**
- Type the shortcut **Os**

Description

The OSC keyword represents the open sound control data.

OSC keyword lists, exports or imports open sound control data in the Remote Inputs.

Syntax

[Function] OSC

Examples

1. List all OSC data in an interface

1. Change destination to OSC:

```
MA User name[Fixture]> ChangeDestination OSC
```

Result:

```
MA User name@ShowData/OSCBase>
```

2. To list OSC data in the interface, type:

```
MA User name@ShowData/OSCBase> List
```

2. Import OSC data in Remote Input

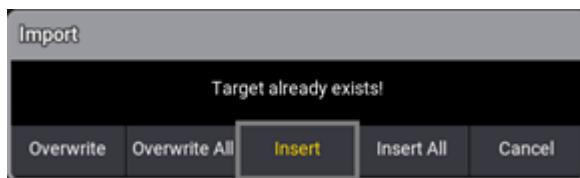
Requirement: Change destination to OSC.

For more information see [Example 1](#).

- To import OSC data, type:

```
MA User name@ShowData/OSCBASE> Import
```

If the target already exists, the import pop-up will appear asking you to choose an option:



Target is occupied

- Choose an option.

The OSC is imported.

8.3.208. Oops Keyword



Hint:

Use Undo as a synonym of the Oops keyword.

To enter the Oops keyword in the command line, use one of the options:

- Press **Oops**
- Type **Oops**
- Type the shortcut **O**

Description

The Oops keyword is a function keyword which is used to cancel:

- The last operation made in the command line;
- The last fixture selection;
- The last operation made in the programmer.

Syntax

Oops

Option Keywords

The Oops keyword uses the following option keywords:

- **/NoConfirmation**

Example

- To undo the last operation, type:

```
MA User name[Fixture]> Oops
```



8.3.209. OutputLayer Keyword

To enter the OutputLayer keyword in the command line, use one of the options:

- Type **OutputLayer**
- Type the shortcut **OU**

Description

OutputLayer is a layer keyword. It is used to change the selected layer to the OutputLayer. This layer can show information in, for instance, the fixture sheet.

Syntax

OutputLayer

Example

- To select the OutputLayer, type:

```
 User name[Fixture]> OutputLayer
```

8.3.210. Page Keyword

To enter the Page keyword in the command line, use one of the options:

- Press **MA** + **X15**
- Type **Page**
- Type the shortcut **P**



Important:

Before calling a page, create a page in the page pool first.

Description

The Page keyword is an object keyword which is used to access pages.

Syntax

([Function]) Page ["Page_Name" or Page_Number] ([/Option] ["Option_Value"])

Option Keywords

The Page keyword uses the following option keywords:

- **/Tab**

Examples

- To change to page 2, type:

```
MA User name[Fixture]> Page 2
```

- To label page 2 "Ray", type:

```
MA User name[Fixture]> Label Page 2 "Ray"
```

- To call the page with the name "Ray", type:

```
MA User name[Fixture]> Page "Ray"
```

- To change to the next page, type:



 User name[Fixture]> Next Page



8.3.211. Part Keyword

To enter the Part keyword in the command line, use one of the options:

- Press **Cue** **Cue**
- Type **Part**
- Type the shortcut **Par**

Description

The Part keyword is an object keyword which is used to segment cues in parts.

Parts are useful for assignment and edit of different timings of groups of fixture parameters.

Syntax

[Function] Part ["Part_Name" or Part_Number]

Examples

- To create a second part of cue 3 in the selected executor, type:

```
MA User name[Fixture]> Store Cue 3 Part 2
```

- To delete part 3 of cue 1, type:

```
MA User name[Fixture]> Delete Cue 1 Part 3
```

8.3.212. Park Keyword

To enter the Park keyword in the command line, use one of the options:

- Press **Pause** **Pause**
- Type **Park**

Description

The Park keyword is a command keyword which is used to prevent DMX channels of fixtures to change their value.

Syntax

Park [Object] ["Object_Name" or Object_Number]

Examples

- To park fixture 1 with all its attributes, type:

```
MA User name[Fixture]> Park Fixture 1
```

- To park the current selection, type:

```
MA User name[Fixture]> Park
```

It is also possible to enter Park into the command line and tap a cell in the fixture sheet to park a certain attribute.

To unpark fixtures and/or attributes, see the **Unpark Keyword**.



Important:

When parking a fixture and/or attribute it will park the corresponding DMX channel.

- To park all DMX channels of fixture 1 at 50%, type:

```
MA User name[Fixture]> Park Fixture 1 At 50
```



Hint:

The command **Park Fixture At** will park all DMX channels of the fixture to the value that is set.

- To park only DMX channels for FeatureGroup 1 at 50, type:

```
MA User name[Fixture]> Park Fixture 1 At 50 If FeatureGroup 1
```

- To park DMX universe 2, type:

```
MA User name[Fixture]> Park DMXUniverse 2
```

To specify the universe:

1. Enter the Park keyword in the command line.
2. Tap the universe in the universe pool.

- To park DMX channel 20 on the first universe, type:

```
MA User name[Fixture]> Park DMXUniverse 1.20
```

To specify the DMX channel:

1. Enter the Park keyword in the command line.
2. Tap the channel in the DMX sheet.



Hint:

If there are parked channels in a universe, they will be indicated by a blue **P** icon in the universe pool.



8.3.213. Paste Keyword

To enter the Paste keyword in the command line, use one of the options:

- Press **Copy** **Copy**
- Type **Paste**
- Type the shortcut **Pas**

Description

The Paste keyword pastes or moves previously cut or copied objects.

For more information see the [Cut keyword](#) and the [Copy keyword](#).

Syntax

Paste Object ["Object_Name" or Object_Number] (/Option)

Option Keywords

The Paste keyword uses the following option keywords:

- **/Date**

Examples

- To cut and then paste group 1 to group 5, type:

```
MA User name[Fixture]> Cut Group 1
```

```
MA User name[Fixture]> Paste Group 5
```

- To copy and then paste cue 5 to cue 15, type:

```
MA User name[Fixture]> Copy Cue 5
```

```
MA User name[Fixture]> Paste Cue 15
```



8.3.214. Patch Keyword

To enter the **Patch** keyword in the command line, use one of the options:

- Type **Patch**
- Type the shortcut **Pat**

Description

The Patch keyword is used to edit the patch address of single fixtures or entire fixture selections.

Syntax

Patch Fixture ["Fixture_Name" or Fixture_Number] [Universe.Address]

Patch Fixture ["Fixture_Name" or Fixture_Number] Thru ["Fixture_Name" or Fixture_Number]

Examples

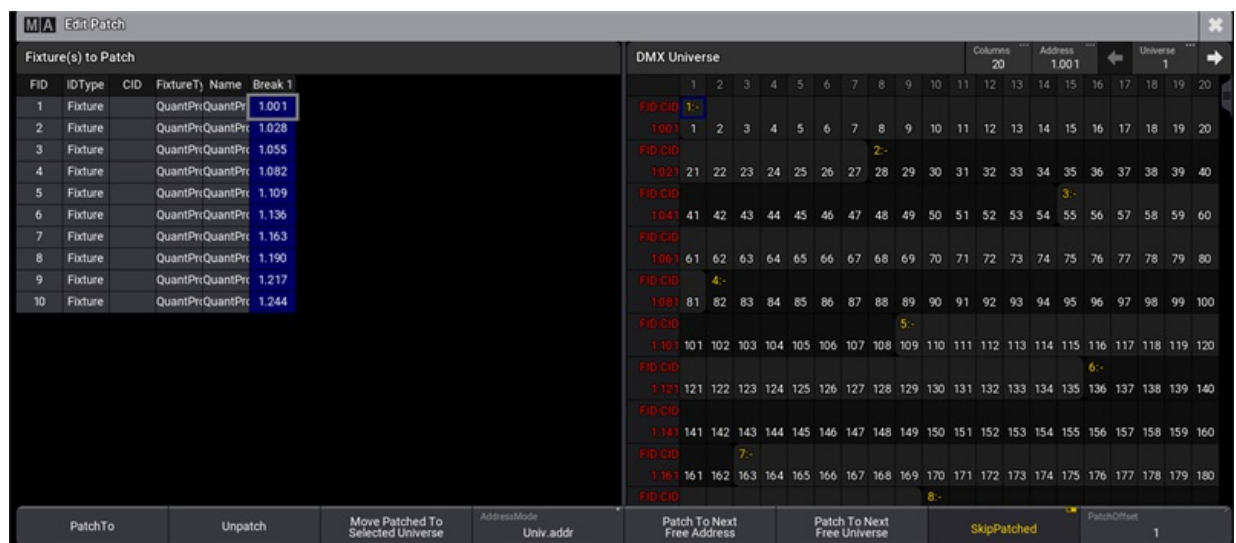
- To patch fixture 2 to the patch address 123 on universe 3, type:

```
MA User name[Fixture]> Patch Fixture 2 3.123
```

- To edit the patch address of fixtures 1 through 10, type:

```
MA User name[Fixture]> Patch Fixture 1 Thru 10
```

The Edit Patch opens:



Edit patched fixtures



You can now patch the fixtures to a new patch address or unpatch them using the user interface.

8.3.215. Pause Keyword

To enter the Pause keyword in the command line, use one of the options:

- Press **Pause**
- Type **Pause**
- Type the shortcut **Pau**

Description

The Pause keyword is a playback keyword which is used to pause:

- Crossfades between steps and cues
- Timecode shows
- Macros



Hint:

Pause is a toggle function between **Pause On** and **Pause Off**.

For information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

Pause (On/Off) [Object] ["Object_Name" or Object_Number]

Examples

- To pause an active phaser in the sequence on executor 201, type:

```
MA User name[Fixture]> Pause Executor 201
```

- To restart the phaser in the sequence of executor 201, type:

```
MA User name[Fixture]> Pause Executor 201
```




8.3.216. Percent Keyword

To enter the Percent keyword in the command line, use one of the options:

- Type **Percent**
- Type the shortcut **Pe**

Description


The Percent keyword is used to set the values of a fixture selection using the percent (%) notation.

Syntax

At [Layer] Percent [Value]

Example

- To set the pan value in the absolute layer to 50 in percent, type:

```
 User name[Fixture]> Attribute "pan" At Absolute Percent 50
```



8.3.217. PercentFine Keyword

To enter the PercentFine keyword in the command line, use one of the options:

- Type **PercentFine**
- Type the shortcut **Percentf**

Description

The PercentFine keyword is used to set the values of a fixture selection taking into consideration two decimal places.

Syntax

At [Layer] PercentFine[Value]

Examples

- To set the absolute layer to 50.35 in PercentFine, type:

```
MA User name[Fixture]> At Absolute PercentFine 50.35
```

- To set the pan value in the absolute layer to 75.50, type:

```
MA User name[Fixture]> Attribute "pan" At Absolute PercentFine 75.50
```



8.3.218. Phase Keyword

To enter the Phase keyword in the command line, use one of the options:

- Type **Phase**
- Type the shortcut **Ph**

Description

Phase is an object keyword that is used to set the Phase layer.

For more information see [Phasers](#).

Syntax

Phase

At Phase [Value] Thru [Value]

Examples

- To change to the phase layer, type:

```
MA User name[Fixture]> Phase
```

Requirement:

In this example, at least 2 steps have to be active in the programmer.

- To set a phase of 0 through 360 in the selected attribute, type:

```
MA User name[Fixture]> At Phase 0 Thru 360
```



8.3.219. Physical Keyword

To enter the Physical keyword in the command line, use one of the options:

- Type **Physical**
- Type the shortcut **Phy**

Description

The Physical keyword is used to set the physical values of a fixture selection using the Physical notation. It comprises RPM (rounds per minute), Hz (Hertz), degrees, or intensity.

Syntax

At [Layer] Physical [Value]

Examples

- To set the dimmer value to 1.0 in the absolute layer using Physical, type:

```
MA User name[Fixture]> At Absolute Physical 1.0
```

- To set the pan value in the absolute layer to 75.60 degrees using Physical, type:

```
MA User name[Fixture]> Attribute "pan" At Absolute Physical 75.60
```



8.3.220. Plugin Keyword

To enter the Plugin keyword in the command line, use one of the options:

- Press **MA** + **X14** + **X14**
- Type **Plugin**
- Type the shortcut **PI**

Description

The Plugin keyword is an object keyword used to access plugins.

The default function of the Plugin keyword is **Go+**.

Syntax

[Function] Plugin [ID]

Examples

- To edit plugin 2, type:

```
MA User name[Fixture]> Edit Plugin 2
```

- To label plugin 1 "Weakon," type:

```
MA User name[Fixture]> Label Plugin 1 "Weakon"
```

Call a Plugin and Specify an Argument

Requirement:

- Create a plugin that uses an argument when calling a function.

In this example our plugin is plugin 1 of the plugin pool and the argument is called name in the definition of the function.

```
function main(display_handle, name)
```

```
    Printf("My name is "..name)
```

```
end
```

```
return main
```



- To generate the sentence "My name is Richard Roe" in the command line history, type:

```
MA User name[Fixture]> Plugin 1 "Richard Roe"
```

Result:

```
OK: Plugin 1 "Richard Roe"  
My name is Richard Roe  
OK: Menu "commandLineHistory"
```

Response in the command line history

"My name is Richard Roe" is now displayed in the command line history.

Call a Dedicated LuaComponent

Requirement:

- Create at least two lua components in the plugin.

For more information on how to create lua components see [Plugins](#).

- To call the second LuaComponent of plugin 1, type:

```
MA User name[Fixture]> Plugin 1.2
```



8.3.221. Preset Keyword

To enter the Preset keyword in the command line, use one of the options:

- Press **Preset**
- Type **Preset**
- Type the shortcut **Pres**

Description

With the Preset keyword, it is possible to:

- Select the fixtures stored in a preset;
- Apply the At function on the preset within the fixture or channel selection;
- Set a property in a preset.

A command containing only the preset keyword and the preset ID performs the default function for a preset - **SelfFix**. For more information see [SelfFix Keyword](#).

To give the fixture attributes a link to the preset, the At keyword needs to be added in front of the Preset keyword and the preset ID. For more information see [At Keyword](#).

Syntax

Preset ["FeatureGroup_Name" or FeatureGroup_Number].["Preset_Name" or Preset_Number]

[Function] Preset ["FeatureGroup_Name" or FeatureGroup_Number].["Preset_Name" or Preset_Number] ([Setting] ["Setting_Value"] [/OptionKeyword])

Assign [Object] ["Object_Name" or Object_Number] At Preset ["FeatureGroup_Name" or FeatureGroup_Number]. ["Preset_Name" or Preset_Number]

Settings

Some settings need an object assigned. These can be assigned using the [Assign Keyword](#).

Other settings contain a text option or a value. The [Set keyword](#) is used for these settings.

The following table displays the settings that need an object:

Setting	Object	Description
Appearance	"Appearance 1"	Assigns the appearance for the pool object.
InputFilter	"Filter 12"	Assigns a filter or world as an input filter for the pool object.
Scribble	"Scribble 1"	Assigns the scribble for the preset pool object.



The following table displays the settings that need an option or value:

Setting	Option/Value	Description
Name	"Preset Name"	Sets the preset pool object name.
MoveGridCursor	"Yes" or "No"	This defines if the grid cursor is moved after calling the preset.
CuePart	"Default" or a specific part number	This is the programmer cue part the preset will be called into.
MAGic	"Yes" or "No"	This defines if the preset is a magic preset or not.
PresetMode	Read only	This is information only about the preset mode.
StoredData	Read only	This is information only about the stored data.

Presets can also contain settings for MAtricks and recipe values. These can be changed using the Set Keyword.

Option Keywords

Some option keywords are relevant when storing presets. Learn about them by following the links.

- [**/Active**](#)
- [**/ActiveForSelected**](#)
- [**/AddNewContent**](#)
- [**/All**](#)
- [**/AllForSelected**](#)
- [**/Ask**](#)
- [**/Auto**](#)
- [**/DMX**](#)
- [**/Embed**](#)
- [**/ForceGlobal**](#)
- [**/Global**](#)
- [**/GridMergeMode**](#)
- [**/InputFilter**](#)
- [**/KeepActivation**](#)
- [**/Look**](#)
- [**/MAtricks**](#)
- [**/Output**](#)
- [**/Overwrite**](#)
- [**/PhaserData**](#)
- [**/Programmer**](#)
- [**/Selective**](#)
- [**/Universal**](#)

Examples



- To select the fixtures that can use preset 5 of the dimmer feature group, type:

```
MA User name[Fixture]> SelFix Preset 1.5
```

- To select the fixtures stored in any preset with the name "DarkRed", type:

```
MA User name[Fixture]> SelFix Preset *."DarkRed"
```

- To call a reference to preset 21.45 ("All 1" preset number 45) to the attributes of the selected fixtures, type:

```
MA User name[Fixture]> At Preset 21.45
```

- To set the name of the position preset 3 to be "Stage Left", type:

```
MA User name[Fixture]> Set Preset 2.3 Name "Stage Left"
```

- To assign world 5 as an input filter on position preset 4, type:

```
MA User name[Fixture]> Assign World 5 At Preset 2.4
```



8.3.222. PresetUpdate Keyword

To enter the PresetUpdate keyword in the command line, use one of these options:

- Type **PresetUpdate**
- Type the shortcut **PresetU**

Description

PresetUpdate contains all presets that could be updated at the moment.

It needs the **List keyword** to show the preset list.

The PresetUpdate keyword is an Object keyword.

Syntax

List PresetUpdate

Example

- To list all the presets that can be updated in the command line feedback, type:

```
 User name[Fixture]> List PresetUpdate
```

It is the same list visible in the **Update menu**.



8.3.223. Press Keyword

To enter the Press keyword in the command line, type **Press**.

Description

The Press keyword is used to simulate the pressed state of a key.

It is possible to assign this state to macros.

For more information see [Macros](#).

For information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

[Function] Press [Object] ["Object_Name" or Object_Number]

Example

- To execute Flash on the executor 203, type:

```
MA User name[Fixture]> Flash Press Executor 203
```

To unpress a key, see [Unpress Keyword](#).



8.3.224. Preview Keyword

To enter the Preview keyword in the command line, use one of the options:

- Press **Preview**
- Type **Preview**
- Type the shortcut **Previe**

Description

The Preview keyword is used to toggle the preview mode. For more information on preview see [What is the Programmer](#).

Syntax

Preview

Preview On

Preview Off

Example

- To toggle the preview mode, type:



The title bar and the frame around the windows that display the preview mode turn red.



8.3.225. Previous Keyword

To enter the Previous keyword in the command line, use one of the options:

- Press **Prev**
- Type **Previous**
- Type the shortcut **Prev**

Description

The Previous keyword has several functions:

- If no fixture is selected, the fixture with the highest ID number will be selected.
- If one fixture is selected, the fixture before this fixture will be selected.
- If MAticks are active, the Previous keyword has special functions depending on the selected MAticks options.

For more information on MAticks see [MAticks and Shuffle](#).

Syntax

Previous

Option Keywords

The Previous keyword uses the following option keywords:

- **/Wrap**

Example

- To select the last fixture of the selected fixtures in the selection order, type:

```
MA User name[Fixture]> Previous
```



8.3.226. PreviousY Keyword

To enter the **PreviousY** keyword in the command line, use one of the options:

- Type **PreviousY**

Description

The PreviousY keyword is a command keyword that is used to move the main selection from fixture to fixture back to the previous position on the y-axis. This can be observed in the selection grid for example.

Syntax

PreviousY

Example

Requirement: Select more than one fixture on the y-axis

```
MA User name[Fixture]> Grid 0/0 Thru 4/4
```

```
MA User name[Fixture]> Fixture 1 Thru 25
```

- To move this selection to the previous position on the y-axis, type:

```
MA User name[Fixture]> PreviousY
```



8.3.227. PreviousZ Keyword

To enter the **PreviousZ** keyword in the command line, use one of the options:

- Type **PreviousZ**

Description

The PreviousZ keyword is a command keyword that is used to move the main selection from fixture to fixture back to the previous position on the z-axis. This can be observed in the selection grid for example.

Syntax

PreviousZ

Example

Requirement: Select more than one fixture on the z-axis

```
MA User name[Fixture]> Grid 0/0/0 Thru 4/0/4
```

```
MA User name[Fixture]> Fixture 1 Thru 25
```

- To move this selection to the previous position on the z-axis, type:

```
MA User name[Fixture]> PreviousZ
```



8.3.228. Programmer Keyword

To enter the Programmer keyword in the command line, use one of the options:

- Press **MA** + **Cue**
- Type **Programmer**
- Type the shortcut **Prog**

Description

The Programmer keyword represents the programmer playback. It is used to enable, disable, or pause values in the programmer.

Syntax

On Programmer

Pause Programmer

Off Programmer

Examples

Requirement: Phasers

- To pause all phasers and transitions running in the programmer, type:

```
MA User name[Fixture]> Pause Programmer
```

- To create programmer part 42, type:

```
MA User name[Fixture]> Store Programmer 42
```




8.3.229. Property Keyword

To enter the Programmer keyword in the command line, use one of the options:

- Type **Property**
- Type the shortcut **Prop**

Description

The Property keyword is an object keyword which is used to communicate with the console for you to set a specific property.

Syntax

[Function] [Object] Property ["Property_Name"] [Value]

Example

- To set the ValueReadout to Hex8 in the current user profile, type:

```
 User name[Fixture]> Set CurrentUserProfile Property "ValueReadout" "Hex8"
```



8.3.230. ProcessingUnit Keyword

To enter the ProcessingUnit keyword in the command line, use one of the options:

- Type **ProcessingUnit**
- Type the shortcut **PU** or **Proc**

Description

The ProcessingUnit keyword is an object keyword which is used to address the grandMA3 processing units in the network.

This keyword belongs to a session-related group of keywords:

- Console keyword
- onPC keyword
- NetworkNode keyword
- Station keyword
- Extension keyword
- Join keyword
- Leave keyword
- Invite keyword
- Dismiss keyword
- Reboot keyword
- Restart keyword
- Shutdown keyword

Syntax

[Function] ProcessingUnit ["ProcessingUnit_Name" or ProcessingUnit_Number]

Examples

To list all processing units that are currently in the same network, type:

```
MA User name[Fixture]> List ProcessingUnit
```

To invite processing unit "FOH3" into the session, type:

```
MA User name[Fixture]> Invite ProcessingUnit "FOH3"
```



8.3.231. Pyro Keyword

To enter the Pyro keyword in the command line, use one of the options:

- Press **Channel** until Pyro appears in the command line
- Type **F + 8**
- Type **Pyro**
- Type the shortcut **Py**



Hint:

To use the Pyro keyword, make sure you assign the Pyro ID type and a custom ID to a fixture first.

Description

The Pyro keyword is an object keyword used to call fixtures of the ID type Pyro in the programmer.

The name of this keyword can change because it is a custom ID type. For more information see the [Custom ID Type Topic](#).

Syntax

Pyro [ID]

Example

- To select the the fixture that uses the pyro ID 8, type:

```
MA User name[Fixture]> Pyro 8
```



8.3.232. Quickey Keyword

To enter the Quickey keyword in the command line, use one of the options:

- Press MA + `X14 | Macro` + `X14 | Macro` + `X14 | Macro`
- Type **Quickey**
- Type the shortcut **Q**

Description

The Quickey keyword is an object keyword that is used to address quickeys in the pool.

For more information see [Quickeys](#).

Syntax

[Function] Quickey ["Quickey_Name" or Quickey_Number]

Example

- To edit quickey 1 in the quickey pool, type:

```
MA User name[Fixture]> Edit Quickey 1
```



8.3.233. Rate1 Keyword

To enter the Rate1 keyword in the command line, use one of the options:

- Press **MA** + **Learn**
- Type **Rate1**
- Type the shortcut **Ra**

Description

The Rate1 keyword is a playback keyword used to reset the rate of a phaser to 1:1.

For information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

Rate1 [Object] ["Object_Name" or Object_Number]

Example

- To reset the rate of executor 105 back to 1:1, type:

```
MA User name[Fixture]> Rate1 Executor 105
```



8.3.234. RDM Keyword

To enter the RDM keyword in the command line, use one of the options:

- Type **RDM**
- Type the shortcut **RD**

Description

The keyword RDM is an object keyword that addresses RDM objects within the RDM folder.

For more information on RDM and how to use it see [RDM](#).

Syntax

[Function] RDM

Examples

- Access the RDM folder first, type:

```
MA User name[Fixture]> ChangeDestination RDM
```

Result:

```
MA User name@ShowData/RDMData>
```

- To list all RDM objects in this folder, type:

```
MA User name@ShowData/RDMData> List
```



8.3.235. Readout Keyword



Important:

Changing the Readout is a user profile setting. That is, the Readout will be automatically set throughout the entire user profile.

For more information see [User Settings](#).



Hint:

-The Readout values are:

Decimal8, Decimal24, Hex8, Hex24, Percent, PercentFine, Physical.

-The names of the readout are case-sensitive, e.g., "Hex8", "Percent".

To enter the Readout keyword in the command line, use one of the options:

- Type **Readout**
- Type the shortcut **Rea**

Description

The Readout keyword is used to set the readout of fixtures in the usual notation.

Syntax

Readout [Name]

Example

- To set the Readout to Hex8, type:

```
MA User name[Fixture]> Readout "Hex8"
```

8.3.236. Reboot Keyword

To enter the Reboot keyword in the command line, use one of the following options:

- Type **Reboot**
- Type the shortcut **R**

Description

The Reboot keyword is a function keyword used to shut down the station in use and boot it up again.

A confirmation pop-up opens on the station in use.

This keyword belongs to a session-related group of keywords.

For more information see:

- [Console keyword](#)
- [Node keyword](#)
- [OnPC keyword](#)
- [Processing Unit keyword](#)
- [Station keyword](#)
- [Extension keyword](#)
- [Join keyword](#)
- [Leave keyword](#)
- [Invite keyword](#)
- [Dismiss keyword](#)
- [Restart keyword](#)
- [Shutdown keyword](#)



Hint:

The device types are: [Console](#), [Node](#), [onPC](#), [PU](#), [Station](#), [Extension](#).

Syntax

[Reboot IP \[IP_Address\] \(/Option\)](#)

[Reboot \[Device_Type\] \(/Option\)](#)

Option Keywords

The Reboot keyword uses the following option keywords:

- [/NoConfirmation](#)
- [/NoSave](#)



- **/Save**

Examples

- To reboot the connected grandMA3 processing unit 1, type:

```
MA User name[Fixture]> Reboot ProcessingUnit 1
```

- To reboot the connected grandMA3 processing unit called "Stage Right", type:

```
MA User name[Fixture]> Reboot ProcessingUnit "Stage Right"
```

- To reboot the console that uses the IP address 192.168.0.4, type:

```
MA User name[Fixture]> Reboot IP 192.168.0.4
```



8.3.237. Recast Keyword

To enter the Recast keyword in the command line, use one of the options:

- Press **MA** + **X1** + **X1**
- Type **Recast**
- Type the shortcut **Reca**

Description

The Recast keyword is a command keyword which is used to update attributes that were added or removed in the presets. It will add or remove these values in cues where the preset is used.

Furthermore, recast can be used when configuring executors. When an executor configuration is used on several executors, and the assignment of handle for one of these executors changes, the changes will not automatically be transmitted to other executors using this configuration. When storing the changes into the executor configuration, it is possible to recast the executor configuration. All other executors using this configuration will then get the new handle assignment. For more information on executor configurations see the [Executor Configurations](#).



Known Limitation:

Recast will only recast presets to cues where a preset link exists for the absolute layer.

Syntax

Recast Preset ["FeatureGroup_Name" or FeatureGroup_Number],["Preset_Name" or Preset_Number]

Recast Configuration ["ExecutorConfiguration_Name" or ExecutorConfiguration_Number]

Example

- The dimmer is open and the color is red in ten fixtures in the All preset 21.1. This preset is used in sequence 1. We now add position to the preset. To recast preset 21.1, type:

```
MA User name[Fixture]> Recast Preset 21.1
```



8.3.238. Record Keyword

To enter the Record keyword in the command line, use one of the options:

- Press **MA** + **Store**
- Type **Record**
- Type the shortcut **Rec**

Description

The Record keyword is used to record a timecode show using the command line.

For information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

Record (On/Off) Timecode ["Timecode_Name" or Timecode_Number]

Example

- To start recording timecode 3 in the timecode pool, type:

```
 User name[Fixture]> Record Timecode 3
```



8.3.239. Relative Keyword

To enter the Relative keyword in the command line, use one of the options:

- Press **MA** + **Time**
- Type **Relative**

Description

The Relative keyword is an object keyword which is used to set the relative layer.

Syntax

Relative

Attribute ["Attribute_Name"] At Relative [Value]

Examples

- To set the layer to relative, type:

```
MA User name[Fixture]> Relative
```

- To set the color red to 10 in the programmer, type:

```
MA User name[Fixture]> Attribute "ColorRGB_R" At Relative 10
```



8.3.240. Release Keyword

To enter the **Release** keyword in the command line, use one of the options:

- Press `Delete Delete Delete`
- Type **Release**
- Type the shortcut **Rel**

Description

The Release keyword enters release values in the programmer depending on attributes specified and the fixtures that were selected.

Release values that are stored using the merge function release previously tracked values in the tracking list. The fixtures use their default values.

If you release an object, the release is applied to the corresponding layer.

Syntax

Release [Object] ["Object_Name" or Object_Number]

Examples

- To enter release values in the programmer in all attributes of the fixture selection in the corresponding layer, type:

```
MA User name[Fixture]> Release
```

- To enter release values in the programmer in the attribute PAN of the fixture selection in the corresponding layer, type:

```
MA User name[Fixture]> Release Attribute "Pan"
```

- To enter release values in the programmer in fixture 1, type:

```
MA User name[Fixture]> Release Fixture 1
```



8.3.241. ReloadAllPlugins Keyword

To enter the ReloadAllPlugins keyword in the command line, use one of the options:

- Type **ReloadAllPlugins**
- Type the shortcut **ReloadP**
- Or the shortcut **RP**

Description

The ReloadAllPlugins keyword is a function keyword used to restart the LUA engine.

It is necessary to restart the LUA engine after you did changes in LUA, because these changes could influence how LUA behaves.

You may want to test the integrity of the LUA system to make sure that it behaves as expected next time you load the show. This is important as the show file saved does not contain a snapshot of the LUA memory. It only contains the integrated functions and the code in the defined plugins.

When the show file is loaded, the LUA engine and the code of the plugins are reloaded. That is, the code of the LUA file on the harddrive is reread and loaded into the show file again. This then may result in a different state than that after you powered down the console or saved the show file.



Hint:

Double-check the executed command in the system monitor.

Syntax

ReloadAllPlugins

Example

- To restart the LUA engine after programming using LUA, type:

```
MA User name[Fixture]> ReloadAllPlugins
```



8.3.242. ReloadUI Keyword

To enter the ReloadUI keyword in the command line, use one of the following options:

- Type **ReloadUI**
- Type **Relo**
- Type the shortcut **RU**

Description

The ReloadUI keyword is a function keyword used to refresh the user interface of the console.

Syntax

ReloadUI

Example

- To refresh the user interface of the console, type:

```
MA User name[Fixture]> ReloadUI
```



8.3.243. Remote Keyword

To enter the Remote keyword in the command line, use one of the options:

- Type **Remote**
- Type the shortcut **Rem**

Description

The Remote keyword is an object keyword used to access the remote input types.

You can store or delete remote input types and set parameters.

Syntax

[Function] Remote [Remote Input Type ID].[Row Number] ["Parameter"] ["Parameter Value"]

Assign [Object] [Object ID] At Remote [Remote Input Type ID].[Row Number]


The following table displays the available remote input types along with their remote input type ID.

Remote Input Type	Remote Input Type ID
DC Remote	1
MIDI Remote	2
DMX Remote	3

In input type the IDs have to be in an order and have to start with 1.

Parameter

The following table displays the parameters you can set using the command line with the help of the **Set Keyword**.

 **Hint:**
 If an option or any other part of the keyword command requires two quotation marks, the outer quotation marks are "+" and the inner quotation marks are '+'.

Parameter	Parameter Value	Description
Lock	"Yes", "No"	Sets the lock status.
Name	"This is the name of the remote"	Sets the name of the remote.
Target	"World", "Sequence", "Macro", "Group", "Menu", "Plugin", "View", "Master"	Sets the target of the action when the contact is active.
Fader	"Master", "X", "Temp", and all the fader functions.	Sets the fader the console should activate.



Parameter	Parameter Value	Description
Key	"Fix", "Select", "Selfix", and all the key functions.	Sets the key the console should activate.
TriggerOn	"0%...100%"	Sets when the trigger will be set to on.
TriggerOff	"0%...100%"	Sets when the trigger will be set to on.
InFrom	"0%...100%"	Sets the starting point of the range of the used incoming signal.
InTo	"0%...100%"	Sets the end point of the range of the used incoming signal.
OutFrom	"0%...100%"	Sets the starting point of the range of the used outgoing signal.
OutTo	"0%...100%"	Sets the end point of the range of the used outgoing signal.
Enabled	"Yes", "No"	Sets the status to enabled or not enabled.
Only for DMX remotes: Address	1.001...1024.512 [universe].[dmx address]	Sets the DMX universe and address.
Only for DMX remotes: Resolution	"8bit", "16bit", "24bit"	Sets the DMX resolution. The DMX channels have to be consecutive for 16 bit and 24 bit.
Only for MIDI remotes: MIDIChannel	"1, 2, 3, ..., 16"	Sets the MIDI channel.
Only for MIDI remotes: MIDIIndex	"1, 2, 3, ..., 128"	Sets the MIDI index.
Only for MIDI remotes: MIDIType	"Note", "NoteAttack", "NoteAttackDecay", "Control"	Sets the MIDI type. Note = MIDI note only NoteAttack = MIDI note and uses the velocity to regulate the master except note off NoteAttackDecay = MIDI note and uses the velocity to regulate the master with note off Control = Control change (CC) messages.



Parameter	Parameter Value	Description
Only for DC		
remotes:		
DC start signal	"1, 2, 3, ..., 64"	Sets the DC start signal.

The start signal and the MIDI offset of the desired input console for **DC Remotes** and **MIDI Remotes** have to be set in the **output configuration menu**.

Examples

- To set the key of the first DMX remote to Go+, type:

```
MA User name[Fixture]> Set Remote 3.1 "Key" "Go+"
```

- To store a new MIDI remote, type:

```
MA User name[Fixture]> Store Remote 2.1
```

- To assign sequence 2 to the second DMX remote, type:

```
MA User name[Fixture]> Assign Sequence 2 At Remote 3.2
```



8.3.244. RemoteHID Keyword

To enter the RemoteHID keyword, use one of the options:

- Type **RemoteHID**

Description

The RemoteHID keyword connects a mouse and/or external keyboard to different stations.

Syntax

RemoteHID IP [IP]

RemoteHID onPC [station name]

Example

- To connect to the station with the IP 192.168.0.4, type:

```
MA User name[Fixture]> RemoteHID IP 192.168.0.4
```

- To connect to the onPC station that has the name "3D", type:

```
MA User name[Fixture]> RemoteHID onPC "3D"
```



8.3.245. RemoteCommand Keyword

To enter the RemoteCommand keyword, use one of the options:

- Type **RemoteCommand**
- Type **RemoteC**

Description

The RemoteCommand keyword remotely sends commands to other stations.

Syntax

RemoteCommand IP [IP] ["Command to be Executed"]

RemoteCommand [Station Type] [ID/"Name"] ["Command to be Executed"]

Example

- To remotely execute the command "Call ViewButton 2.1" on the station with the IP address 192.168.0.10, type:

```
MA User name[Fixture]> RemoteCommand IP 192.168.0.10 "Call ViewButton 2.1"
```

- To remotely lock the desk on the station with the IP address 192.168.0.10, type:

```
MA User name[Fixture]> RemoteCommand IP 192.168.0.10 'Menu "DeskLock" '
```

Alternatively type:

```
MA User name[Fixture]> RemoteCommand IP 192.168.0.10 "Menu 'DeskLock' "
```

For more information on the usage of quotation marks see [General Syntax Rules](#).

For more mutual examples see the [Station Keyword](#).



8.3.246. Remove Keyword

To enter the **Remove** keyword in the command line, use one of the options:

- Press **Delete Delete**
- Type **Remove**
- Type the shortcut **Remov**

Description

The Remove keyword enters remove values in the programmer depending on the attributes specified and the fixtures that were selected.

Remove values that are stored using the merge function remove previously stored values.

If a stored value is removed, values from the previous cue will be tracked again.

Syntax

Remove Object ["Object_Name" or Object_Number]

Examples

- To set all attributes of the current selection to Remove, type:

```
MA User name[Fixture]> Remove
```

- To set the Remove value in all attributes of the feature group Position, however in only the selected layer, type:

```
MA User name[Fixture]> Remove FeatureGroup "Position"
```



8.3.247. RenderQuality Keyword

To enter the RenderQuality keyword in the command line, use one of the options:

- Type **RenderQuality**
- Or type the shortcut **Ren**

Description

The RenderQuality keyword is an object keyword that is used to address the objects within the render qualities pool.



Hint:

The first eight render quality objects are locked by default.

Syntax

[Function] RenderQuality ["RenderQuality_Name" or RenderQuality_Number]

Examples

- To list all render qualities, type:

```
MA User name[Fixture]> List RenderQuality
```

- To label the RenderQuality 9 "Fabulous", type:

```
MA User name[Fixture]> Label RenderQuality 9 "Fabulous"
```



8.3.248. Reset Keyword

To enter the Reset keyword in the command line, use one of the options:

- Type **Reset**
- Type the shortcut **Rese**

Description

The Reset keyword is a function keyword used to clear the settings of objects, for example the MAtricks.

Syntax

Reset [Object]

Example

- To reset the MAtricks settings of the current selection, type:

```
MA User name[Fixture]> Reset Selection MAtricks
```



8.3.249. Restart Keyword

To enter the Restart keyword in the command line, use one of the options:

- Type **Restart**
- Type the shortcut **Res**

Description

The Restart keyword is a function keyword that is used to restart the application. Restart behaves the same as closing the program and reopening it without shutting down the console.

Syntax

Restart (/Option)

Restart [Device_Type] (/Option)

Restart IP [IP_Address] (/Option)

Option Keywords

The Restart keyword uses the following option keywords:

- **/NoConfirmation**
- **/NoSave**
- **/Save**

Examples

- To restart the application of the console, type:

```
MA User name[Fixture]> Restart
```

- To restart the application of the station using the IP address 192.168.0.32, type:

```
MA User name[Fixture]> Restart IP 192.168.0.32
```




8.3.250. Root Keyword



Important:

We recommend using the index name of the root. If you use root index numbers, double-check after every release if the root index number is still valid. If the root index number has changed, adjust your macros.

To enter the Root keyword in the command line, use one of the options:

- Type **Root**
- Type the shortcut **Ro**

Description

The Root keyword is an object keyword used to access the root in the object tree.

Syntax


Root [root-index]

Root [root-index].[sub-index]

Root [root-index].[sub-index].[sub-sub-index]

Example

- To change the destination back to root, type:

```
 User name@Root/ShowData/LivePatch/ID Types/Fixture> CD Root
```



8.3.251. RealtimeChannel Keyword

To enter the RealtimeChannel keyword in the command line, use one of the options:

- Type **RealtimeChannel**
- Type the shortcut **Real**

Description


The RealtimeChannel keyword is used to address realtime channels.

Syntax

[Function] RealtimeChannel

Example

- To list all realtime channels in a show file, type:

```
 User name[Fixture]> List RealtimeChannel
```

Realtime channels are also displayed in the parameter list of the patch.

For more information see **Patch and Fixture Setup**.

8.3.252. RunningMacro Keyword

To enter the RunningMacro keyword in the command line, use one of the options:

- Type **RunningMacro**
- Type the shortcut **RunningM**

Description

The RunningMacro keyword is useful when there are several users in a session.

It allows you to list or disable all running macros in a show file.



Important:

The numbers of the running macros are not equal to the numbers of macros in the macros data pool.

Syntax

[Function] RunningMacro [Number]

Example

- To disable the first macro that is currently running, type:

```
MA User name[Fixture]> Off RunningMacro 1
```

- To list all running macros, change destination to the running macros first:

```
MA User name[Fixture]> ChangeDestination RunningMacro
```

Result:

```
MA User name@Temp/RunningMacros>
```

- To list all running macros, type:

```
MA User name@Temp/RunningMacros> List
```

8.3.253. RunningPreset Keyword

To enter the RunningPreset keyword in the command line, use one of the options:

- Type **RunningPreset**
- Type the shortcut **RunningP**

Description

The RunningPreset keyword is useful when there are several users in a session.

It allows you to list or disable all running presets in a show file.



Important:

The numbers of the running presets are not equal to the numbers in the preset pool.

Syntax

[Function] RunningPreset ["Preset_Name" or Preset_Number]

Example

- To disable the first color preset that is currently running, type:

```
MA User name[Fixture]> Off RunningPreset 4.1
```

- To list all running presets, change destination to the running presets first:

```
MA User name[Fixture]> ChangeDestination RunningPreset
```

Result:

```
MA User name@Temp/RunningPreset>
```

- To list all running presets, type:

```
MA User name@Temp/RunningPreset> List
```

8.3.254. RunningSequence Keyword

To enter the RunningSequence keyword in the command line, use one of the options:

- Type **RunningSequence**
- Type the shortcut **Run**

Description

The RunningSequence keyword is useful when there are several users in a session.

It allows you to list or disable all running sequences in a show file.



Important:

The numbers of the running sequences are not equal to the numbers of sequences in the sequences data pool.

Syntax

[Function] RunningSequence [Number]

Example

- To disable the first sequence that is currently running, type:

```
MA User name[Fixture]> Off RunningSequence 1
```

- To list all running sequences, change destination to the running sequences first:

```
MA User name[Fixture]> ChangeDestination RunningSequence
```

Result:

```
MA User name@Temp/RunningSequences>
```

- To list all running sequences, type:

```
MA User name@Temp/RunningSequences> List
```

8.3.255. RunningTimecode Keyword

To enter the RunningTimecode keyword in the command line, use one of the options:

- Type **RunningTimecode**
- Type the shortcut **RunningTimec**

Description

The RunningTimecode keyword addresses the running timecodes in the show file.



Important:

The numbers of the running timcodes are not equal to the numbers of timecodes in the timecodes data pool.

Syntax

[Function] RunningTimcode ["RunningTimecode_Name" or RunningTimecode_Number]

Example

- To disable the first timecode that is currently running, type:

```
MA User name[Fixture]> Off RunningTimecode 1
```

- To list all running timecodes, change destination to the running timecodes first:

```
MA User name[Fixture]> ChangeDestination RunningTimecodes
```

Result:

```
MA User name@Temp/RunningTimecodes>
```

- To list all running timecodes, type:

```
MA User name@Temp/RunningTimecodes> List
```

8.3.256. RunningTimer Keyword

To enter the RunningTimer keyword in the command line, use one of the options:

- Type **RunningTimer**
- Type the shortcut **RunningT**

Description

The RunningTimer keyword grants access to all running timers.

It allows you to list or disable all running timers in a show file.



Important:

The numbers of the running timers are not equal to the numbers of timers in the timers data pool.

Syntax

[Function] RunningTimer [Number]

Example

- To disable the first timer that is currently running, type:

```
MA User name[Fixture]> Off RunningTimer 1
```

- To list all running timers, change destination to the running timers first:

```
MA User name[Fixture]> ChangeDestination RunningTimer
```

Result:

```
MA User name@Temp/RunningTimers>
```

- To list all running timers, type:

```
MA User name@Temp/RunningTimers> List
```



8.3.257. SaveShow Keyword

To enter the SaveShow keyword in the command line, use one of the options:

- Type **SaveShow**
- Type the shortcut **Sa**

Description

The SaveShow keyword is a function keyword used to save the current show file.

If you do not enter a new show name, the show will be saved using the name of the current show.



Important:

If there is already a show file with the same show name, the console overwrites the existing show file.

For more information on the folder structure of shows, demo shows, and backups, see [Show File Handling](#).

Syntax

SaveShow ("Show_Name") (/Option)

Option Keywords

The SaveShow keyword uses the following option keywords:

- **/Enumerate**

Examples

- To save the show as "Rhapsody", type:

```
MA User name[Fixture]> SaveShow "Rhapsody"
```

- To save the show that is currently opened, type:

```
MA User name[Fixture]> SaveShow
```


8.3.258. ScreenConfiguration Keyword



Important:

It is possible to create screen configurations for single user profiles. However, every user of the same user profile can use a different screen configuration.

By default, every user profile has 2 screen configurations:

- Default
- 3D

To enter the ScreenConfiguration keyword in the command line, use one of the options:

- Type **ScreenConfiguration**
- Type the shortcut **ScreenConf**

Description

The ScreeConfiguration keyword is an object keyword that addresses the different screen configurations of a user profile. For example, one setup for the operation of a console, and a second setup for the operation of a dedicated onPC station.

Syntax

[Function] ScreenConfiguration ["ScreenConfiguration_Name" or ScreenConfiguration_Number]

Examples

- To store a new screen configuration called "Average Joe", type:

```
MA User name[Fixture]> Store ScreenConfiguration 3 "Average Joe"
```

- To enter screen configuration 2, type:

```
MA User name[Fixture]> ScreenConfiguration 2
```

- To assign the screen configuration "Average Joe" to ViewButton 1.6, type:

```
MA User name[Fixture]> Assign ScreenConfiguration "Average Joe" At ViewButton 1.6.
```



For more information on ViewButtons see [Store and Recall Views](#).



8.3.259. ScreenContent Keyword

To enter the ScreenContent keyword in the command line, use one of the options:

- Type **ScreenContent**
- Type the shortcut **Scre**

Description

The ScreenContent keyword is used to represent the windows of a display.

Syntax

[Function] ScreenContent

[Function] ScreenContent [ID].[ID] ["Property"] ["Property Value"]

Examples

- To delete all windows on all screens, type:

```
MA User name[Fixture]> Delete ScreenContent *.*
```

- To delete all windows on screen 1, type:

```
MA User name[Fixture]> Delete ScreenContent 1.*
```

- To set the width of the first window you created to 12 half columns in screen 1, type:

```
MA User name[Fixture]> Set ScreenContent 1.1 "W" "12"
```



8.3.260. Scribble Keyword

To enter the Scribble keyword into the command line, use one of the options:

- Press **MA** + **X4** + **X4** + **X4**
- Type **Scribble**
- Type the shortcut **Scr**

Description

Scribbles are little drawings that are used as visualizations instead of labels.

The scribble keyword is an object keyword that is used to access scribbles with a scribble ID.

It is also possible to label scribbles and address them by their label.

For more information about scribbles see [Scribbles](#).

Syntax

[Function] Scribble [ID/"Name"] At [Object List]

Examples

- To edit scribble 1, type:

```
MA User name[Fixture]> Edit Scribble 1
```

The edit scribble pop-up opens.

- To assign scribble 1 to group 5, type:

```
MA User name[Fixture]> Assign Scribble 1 At Group 5
```

- To delete scribble 1, type:

```
MA User name[Fixture]> Delete Scribble 1
```



8.3.261. Seconds Keyword

To enter the Seconds keyword in the command line, use one of the options:

- Type **Seconds**
- Type the shortcut **Sec**

Description

The Seconds keyword is used to set the speed of a fixture selection using the unit seconds.

Syntax

[At] [Speed] Seconds [Value]

Examples

- To set the speed layer to 20 seconds, type:

```
MA User name[Fixture]> At Speed Seconds 20
```

- To set the first speed master to 6 seconds, type:

```
MA User name[Fixture]> Master 3.1 At Seconds 6
```



8.3.262. Select Keyword

To enter the Select keyword in the command line, use one of the options:

- Press **Select**
- Type **Select**
- Type the shortcut **Sel**

Description

The Select keyword selects objects as default objects.

Every selected object is indicated by a yellow frame.

- The **selected sequence** is the target for all sequence-related commands, e.g., **Store Cue 4**.
- The **selected sequence** is displayed in the Sequence Sheet, if the **Link Type** is set to **Selected**.
- The **selected camera** is displayed in the 3D View, if the **Camera Selection** is set to **Link Selected**.

For information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

Select [Object] ["Object_Name" or Object_Number]

Example

- To select the sequence assigned on executor 105, type:

```
MA User name[Fixture]> Select Executor 105
```

- To select page 2, type:

```
MA User name[Fixture]> Select Page 2
```

- To select world 1, type:

```
MA User name[Fixture]> Select World 1
```



- To select data pool 3, type:

```
MA User name[Fixture]> Select DataPool 3
```



8.3.263. Selection Keyword

To enter the Selection keyword in the command line, use one of the options:

- Press `Fixture` `Fixture`
- Type **Selection**
- Type the shortcut **Sel**

Description

The Selection keyword is an object keyword used to represent the current selection of fixtures in the programmer.

Syntax

Selection

Example

- To disable all programmer values of the current fixture selection, type:

```
MA User name[Fixture]> Off Selection
```




8.3.264. SelectFixtures Keyword

To enter the **SelectFixtures** keyword in the command line, use one of the options:

- Press **SelfFix**
- Type **SelectFixtures**
- Type the shortcut **Self**

Description

The **SelectFixtures** keyword is a function keyword used to create selections of fixtures in the programmer.

If only fixtures are selected, the **SelectFixtures** keyword adds additional fixtures to the selection.

If fixtures are selected and activated in the programmer, the **SelectFixtures** keyword replaces the selection by the **SelectFixtures** selection.

If the exact same **SelectFixtures** command is successively used:

- using it the second time activates all attributes of the selected fixtures in the programmer
- using it the third time deactivates all attributes of the selected fixtures in the programmer

SelectFixtures is the default function of most objects, for example, fixture or group or preset.

To clear the selection, press **Clear**.

Syntax

SelectFixtures [Object] ["Object_Name" or Object_Number]

Examples

- To select all fixtures or channels stored in a sequence of the executor 101, type:

```
MA User name[Fixture]> SelectFixtures Executor 101
```

- To select all fixtures stored in dimmer preset 1.1, type:

```
MA User name[Fixture]> SelectFixtures Preset 1.1
```



8.3.265. Sequence Keyword

To enter the Sequence keyword in the command line, use one of the options:

- Press **Sequ**
- Type **Sequence**
- Type **Se**

Description

The sequence keyword is an object keyword that is used to select sequences in the sequence pool.

Syntax

[Function] Sequence ["Sequence_Name" or Sequence_Number]

Option Keywords

The Sequence keyword uses the following option keywords:

- **/CreateSecondCue**
- **/DMX**
- **/Merge**
- **/Output**
- **/Overwrite**
- **/Programmer**
- **/Release**
- **/Remove**
- **/Selection**

Examples

- To select sequence 5 in the sequence pool, type:

```
MA User name[Fixture]> Sequence 5
```

- To block all cues in sequence 5, type:

```
MA User name[Fixture]> Block Sequence 5
```

8.3.266. SendMIDI Keyword



Important:

If you do not define a channel value, the MIDI messages will be automatically addressed to channel 1.

To enter the SendMIDI keyword in the command line, use one of the options:

- Type **SendMIDI**
- Type **Sendm**

Description

The SendMIDI keyword is a command keyword that is used to output MIDI notes, MIDI control change messages, and MIDI program change messages.

Syntax

SendMIDI "Note" (Channel/)Note (Velocity) (Status)

SendMIDI "Program" (Channel/)Value

SendMIDI "Control" (Channel/)Controller (Value)

Examples

- To send out the MIDI note 42 on MIDI channel 2 with a velocity of 99, type:

```
MA User name[Fixture]> SendMIDI "Note" 2/42 99
```

- To send out the MIDI note 37 on MIDI channel 1, type:

```
MA User name[Fixture]> SendMIDI "Note" 37
```

- To send out the MIDI control value 64 to MIDI channel 1 controller 8, type:

```
MA User name[Fixture]> SendMIDI "Control" 1/8 64
```

- To send out a MIDI program change of 12, type:



```
MA User name[Fixture]> SendMIDI "Program" 12
```



8.3.267. SendOSC Keyword

To enter the **SendOSC** keyword in the command line, use one of the options:

- Type **SendOSC**
- Type the shortcut **Sen**

Description

The SendOSC keyword is a command keyword used to send an OSC command.

To learn more about OSC, read the [Remote In and Out topic](#).

Syntax

SendOSC [ID] "[OSCAddress],[OSC Type],[Value]"

The supported types are:

- Int(i)
- Float(f)
- Blob(b)
- String(s)
- True(T)
- False(F)
- Null(N)
- Impulse(l)
- Timetag(t)

It is not necessary to set a value (Payload) for:

- True
- False
- Null
- Impulse
- Timetag



Hint:

When using the OSC types True, False, Nil/Null, Impulse and Timetag it is not necessary to enter a value.



Hint:

Several values can be sent at once when separating them with commas.



Important:

When addressing an executor, a page must be specified as well.

Instead of using page and executor numbers it is also possible to address them by name.

When addressing executor keys, a value of 0 will be interpreted as not pressed. Values greater 0 will be interpreted as button press.

If a prefix is specified for an OSCData entry, then this very prefix will be added to the sent string when using the OSCSend command.



Hint:

The supported OSC types to control faders, executor knobs, and buttons are: Integer32, Float32, True, False and Nil/Null.

A True will be interpreted as 1, while a False will be interpreted as 0.



Hint:

The addresses defined for Page, Fader, ExecutorKnob, and Key are case-sensitive.

Examples

- To send an OSC command using the first configuration in the OSC menu with integer value 50 to fader 201 on page 1, type:

```
MA User name[Fixture]> SendOSC 1 "/Page1/Fader201,i,50"
```

- To send an OSC command using the first configuration in the OSC menu with integer value 100 to fader 201 on page 1 and a fade time of 5s, type:

```
MA User name[Fixture]> SendOSC 1 "/Page1/Fader201,ii,100,5"
```

- To send commands via OSC to a second grandMA3 station, the OSC address /cmd can be used. To store cue 1 via OSC, type:

```
MA User name[Fixture]> SendOSC 1 "/cmd,s,Store Cue 1"
```





8.3.268. Set Keyword

To enter the Set keyword in the command line, use one of the options:

- Press **MA** + **Assign**
- Type **Set**

Description

The Set keyword sets values to the properties of objects.

Syntax

Set ["Object_Name" or Object_Number] ["Property_Name"] [Property_Value]

Option Keywords

The Set keyword uses the following option keywords:

- **/PatchOffset**

Examples

- To set sequence 8 to priority HTP, type:

```
MA User name[Fixture]> Set Sequence 8 "Priority" 3
```




8.3.269. SetGlobalVariable Keyword

To enter the SetGlobalVariable keyword in the command line, use one of the options:

- Type **SetGlobalVariable**
- Type the shortcut **Setg**

Description

The SetGlobalVariable keyword is used to set global variables in a show.

Syntax

SetGlobalVariable ["Name of Variable"] [Numeric Value]

SetGlobalVariable ["Name of Variable"] ["Text"]

Example

- To set the global variable Urban Blues to the value of 3, type:

```
MA User name[Fixture]> SetGlobalVariable "Urban Blues" "3"
```



8.3.270. SetUserVariable Keyword

To enter the SetUserVariable keyword in the command line, use one of the options:

- Type **SetUserVariable**
- Type the shortcut **Setu**

Description

The SetUserVariable keyword is used to set user-specific variables.

Syntax

SetUserVariable ["Name of Variable"] [Numeric Value]

SetUserVariable ["Name of Variable"] ["Text"]

Example

- To set the user variable Green to the value 5, type:

```
MA User name[Fixture]> SetUserVariable "Green" 5
```

8.3.271. Shuffle Keyword

To enter the Shuffle keyword in the command line, use one of the options:

- Press **Selfix** **Selfix**
- Type **Shuffle**
- Type the shortcut **Shuf**

Description

Shuffle is a command keyword used to shuffle the order of the fixture selection. Shuffle is part of the MAtricks toolset.

For more information, read the [MAtricks and Shuffle topic](#).

Syntax

MAtricks [Axis] [Value]

MAtricks [Axis] +



Hint:

The plus is a replacement for the value. You can either use plus/minus or a value.

Examples

To shuffle the current selection on the y-axis, type:

```
MA User name[Fixture]> MAtricks "YShuffle" +
```

To shuffle the current selection on the z-axis, type:

```
MA User name[Fixture]> MAtricks "ZShuffle" +
```

It is also possible to set a certain value to any of the three shuffle settings. To set the shuffle to 4 for the x-axis, type:

```
MA User name[Fixture]> MAtricks "XShuffle" 4
```



Hint:

When deactivating or resetting the MAtricks, the original selection order will be restored.

8.3.272. ShutDown Keyword

To enter the ShutDown keyword in the command line, use one of these options:

- Type **ShutDown**
- Type the shortcut **Sh**

Description

The ShutDown keyword powers down the grandMA3 console or closes the grandMA3 onPC.

A shutdown requires a confirmation in the local station.

The shutdown can be canceled within 10 seconds using a remote station.

This keyword belongs to a session-related group of keywords.

For more information see:

- [Console keyword](#)
- [OnPC keyword](#)
- [PU keyword](#)
- [Station keyword](#)
- [Extension keyword](#)
- [Join keyword](#)
- [Leave keyword](#)
- [Invite keyword](#)
- [Dismiss keyword](#)
- [Reboot keyword](#)
- [Restart keyword](#)



Hint:

The device types are: [Console](#), [onPC](#), [PU](#), [Station](#), [Extension](#).

Syntax

ShutDown [Device_Type] (/Option)

ShutDown IP [IP_Address] (/Option)

Option Keywords

The ShutDown keyword uses the following option keywords:

- [/NoAutoClose](#)



- **/NoConfirmation**
- **/NoSave**
- **/Save**

Examples

- To shut down the current station and provoke a countdown pop-up, type:

```
MA User name[Fixture]> ShutDown
```

- To shut down the station with the IP address 192.168.0.4, type:

```
MA User name[Fixture]> ShutDown IP 192.168.0.4
```



8.3.273. SnapDelay Keyword

To enter the SnapDelay keyword in the command line, use one of the options:

- Type **SnapDelay**
- Type the shortcut **Sn**

Description

The SnapDelay keyword sets a snap time.

The snap time is a delay time for attributes that do not fade, for example a gobo wheel or a color wheel.

Syntax

SnapDelay [Value]

Examples

- To set the snap time of the current cue of the selected sequence to 4 seconds, type:

```
MA User name[Fixture]> SnapDelay 4
```



8.3.274. SoftwareImport Keyword

To enter the SoftwareImport keyword in the command line, use one of the options:

- Type **SoftwareImport**
- Type the shortcut **SoftwareI**

Description

The SoftwareImport keyword is a function keyword which is used to import installation packages of a grandMA3 installer permanently to the hard drive of your grandMA3 console or the grandMA3 onPC using a given path.

Syntax

SoftwareImport "release_type_vx.x.x.xml;Path/to/the/location/of/the/installer"

Example

- To import the installation packages of the grandMA3 stick v1.6.3.7 on the USB drive "Software" that is recognized as drive D on a Windows computer, and where the files are located within the ma folder, type:

```
MA User name[Fixture]> SoftwareImport  
"release_stick_v1.6.3.7.xml;D:/Software/ma"
```



8.3.275. SoftwareUpdate Keyword

To enter the SoftwareUpdate keyword in the command line, use one of the options:

- Type **SoftwareUpdate**
- Type the shortcut **SoftwareU**

Description

The UpdateSoftware keyword is a function keyword used to update the software of every MA device or program in the network.

For more information on how to update the software and requirements see [Update grandMA3 Consoles](#).

Syntax

SoftwareUpdate [StationType] [ID/"Name"]

"release_type_x.y.z.a.xml;/Path/to/MALightingTechnology/installation_packages"

Example

Requirement:

1. The grandMA3 onPC runs on Windows®
 2. Copy the files of the ma folder of the grandMA3_stick_v1.6.3.7.zip file to C:\ProgramData\MALightingTechnology\installation_packages
- To update the first console within your network to grandMA3 v1.6.3.7, type:

```
MA User name[Fixture]> SoftwareUpdate Console 1  
"release_stick_1.6.3.7.xml;C:/ProgramData/MALightingTechnology/installation_p
```




8.3.276. SoundChannel Keyword

To enter the SoundChannel keyword in the command line, use one of the options:

- Type **SoundChannel**
- Type the shortcut **SoundC**

Description


The SoundChannel keyword represents sound codes in the attribute calculators.

Syntax

[Attribute List] [At] SoundChannel [ID/"Name"]

Example

- To set the value for Pan of the selected fixtures to Band1, type:

```
 User name[Fixture]> Attribute "Pan" At SoundChannel 5
```

For more information about using sound input, see the [Sound Window](#) topic.



8.3.277. Solo Keyword

To enter the Solo keyword in the command line, use one of the options:

- Press **Solo**
- Type **Solo**
- Type **So**

Description

The Solo keyword is a function keyword used to set the intensity values of fixtures that are not selected to zero.

Only the fixtures that are selected generate visible output on stage.

If Solo is used standalone, it toggles between Solo On and Solo Off.

To manually fade from the current value to solo and vice versa, use the Grand Solo.

For information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

Solo (On/Off) [Object] ["Object_Name" or Object_Number]

Example

- To enable Solo, type:

```
MA User name[Fixture]> Solo On
```



8.3.278. SpecialExecutor Keyword

To enter the SpecialExecutor keyword in the command line, use one of the options:

- Press **MA** + **X16|Exec** + **X16|Exec**
- Type **SpecialExecutor**
- Or type the shortcut **SE**

Description

The SpecialExecutor keyword addresses special executors.

For more information on the configuration see [Executor Configurations](#).

Syntax

[Function] SpecialExecutor [ID/"Name"]

Examples

- To open the menu **Special Executor Configuration**, type:

```
MA User name[Fixture]> Assign SpecialExecutor 7
```

- To assign the functionality of grand master to SpecialExecutor 2, type:

```
MA User name[Fixture]> Assign Master 2.1 At SpecialExecutor 2
```



8.3.279. Speed Keyword

To enter the Speed keyword in the command line, type **Speed**.

Description

The Speed keyword is used to set the speed of phasers.

For more information see [Phasers](#).

Requirement:

To set values for the speed, there must be at least 2 steps in the programmer.

Syntax

[At] Speed [Value]

Examples

- To set the selected layer to speed, type:

```
MA User name[Fixture]> Speed
```

- To set the speed of the dimmer to 70 hertz, type:

```
MA User name[Fixture]>At Speed Hz 70
```



8.3.280. Speed1 Keyword

To enter the Speed1 keyword in the command line, use one of the options:

- Type **Speed1**
- Type the shortcut **Spee**

Description

The Speed1 keyword is used to reset the speed to 60 BPM.

For information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

Speed1 [Object] ["Object_Name" or Object_Number]

Example

- To reset the speed master of the sequence that is assigned to executor 201, type:

```
MA User name[Fixture]> Speed1 Executor 201
```



8.3.281. SpeedMaster Keyword

To enter the SpeedMaster keyword in the command line, use one of the options:

- Type **SpeedMaster**
- Type the shortcut **Speedm**

Description

The SpeedMaster is a layer keyword. It defines which speed master the phaser is to adapt its speed to in each attribute.


Syntax

SpeedMaster


(Attribute ["Attribute_Name" or Attribute_Number]) At SpeedMaster [SpeedMaster_Number]

Examples

- To set the layer to speed master, type:

```
 User name[Fixture]> SpeedMaster
```

- To set attribute "Pan" to SpeedMaster 1, type:

```
 User name[Fixture]> Attribute "Pan" At SpeedMaster 1
```



8.3.282. Stage Keyword

To enter the Stage keyword in the command line, use one of the options:

- Type **Stage**
- Type the shortcut **Sta**

Description

The Stage keyword addresses stages in the 3D space.

Syntax

[Function] Stage [ID/"Name"] ["Property"] ["Value"]

Example

- To store a new stage, type:

```
MA User name[Fixture]> Store Stage 2
```

- To list all stages in the command line history, type:

```
MA User name[Fixture]> List Stage
```

- To set the center of stage 1 to 5 meters in the x direction of the Cartesian coordinate system, type:

```
MA User name[Fixture]> Set Stage 1 "POSX" "5"
```



8.3.283. Station Keyword

To enter the Station keyword in the command line, use one of the options:

- Type **Station**
- Type the shortcut **Stat**

Description

The Station keyword is an object keyword used to display all stations in the network, but you can also **invite** stations into your session or **dismiss** them.

This keyword belongs to a session-related group of keywords.

For more information see:

- Console keyword
- Node keyword
- onPC keyword
- PU keyword
- Extension keyword
- Join keyword
- Leave keyword
- Invite keyword
- Dismiss keyword
- Reboot keyword
- Restart keyword
- Shutdown keyword

Syntax

Station [ID]

Station [Name]

Examples

- To list all stations that are currently in the same network, type:

```
 User name[Fixture]> List Station
```

- To invite station "FOH3" into the session, type:



```
MA User name[Fixture]> Invite Station "FOH3"
```



8.3.284. StationSettings Keyword

To enter the StationSettings keyword in the command line, use one of the options:

- Type **StationSettings**
- Type the shortcut **Stations**

Description

StationSettings keyword is used to address settings of your station.

Syntax

[Function] StationSettings ("StationSettings_Name" or StationSettings_Number] Property ["Property_Name"] [Value])

Examples

- To list all station settings, type:

```
MA User name[Fixture]> List StationSettings
```

- To set your desk light to 25%, type:

```
MA User name[Fixture]> Set StationSettings "DesklightsCollect"."DeskLights"  
Property "Master" 25
```



8.3.285. Step Keyword

To enter the Step keyword in the command line, use one of the options:

- Press **MA** + **X5**
- Type **Step**

Description

The Step keyword is a command keyword which is used to define the step of the At command.

Syntax

[Function] Step [Step_Number]

Option Keywords

The Step keyword uses the following option keywords:

- **/Wrap**

Example

Requirement:

- Step 1 has to contain information
- To create the next step in the programmer, type:

```
MA User name[Fixture]> Next Step
```



8.3.286. Stomp Keyword

To enter the Stomp keyword in the command line, use one of the options:

- Press **Stomp**
- Type **Stomp**
- Type the shortcut **Stom**

Description

The Stomp keyword is a command keyword used to stop a running phaser.

Stomp now knocks in the last used single-step value for the absolute and/or the relative layer of the attribute. This can be a value from the programmer or from a playback. If no single-step value is found, it will use the default value.

Syntax

Stomp ([Object] ["Object_Name" or Object_Number])

Examples

- To stomp the phasers in the selected fixtures, type:

```
MA User name[Fixture]> Stomp
```

- To stomp the phasers for the feature group "Position" in the selected fixtures, type:

```
MA User name[Fixture]> Stomp FeatureGroup "Position"
```

- To stomp the running phasers in fixtures 1 to 10, type:

```
MA User name[Fixture]> Stomp Fixture 1 Thru 10
```

- To stomp the phasers in all fixtures that are stored in the group "All Spots", type:

```
MA User name[Fixture]> Stomp Group "All Spots"
```



8.3.287. Store Keyword

To enter the **Store** keyword in the command line, use one of the options:

- Press **Store**
- Type **Store**
- Type the shortcut **S**

Description

The Store keyword is a function keyword used to store objects in the show file.

If no object type or destination is given, the object type **Cue** will be used for the selected sequence.

Syntax

Store [Object] ["Object_Name" or Object_Number] (/Option)

Option Keywords

The Store keyword uses the following option keywords:

- /Active
- /ActiveForSelected
- /All
- /AllForSelected
- /Ask
- /Auto
- /CreateSecondCue
- /CueOnly
- /Embed
- /ForceGlobal
- /Global
- /InputFilter
- /KeepActivation
- /Look
- /MAticks
- /Merge
- /NoConfirmation
- /Overwrite
- /PhaserData
- /Remove
- /Screen



- **/Selective**
- **/Universal**

To learn more about store options, read the **Store Options and Store Preferences** topic.

Examples

- To store cue 2 in the selected sequence, type:

```
MA User name[Fixture]> Store 2
```

For more information see **Store Cues**.

- To store the programmer values as cue 1 through cue 10 and cue 20 through cue 30, type:

```
MA User name[Fixture]> Store Cue 1 Thru 10 + 20 Thru 30
```

- To store the programmer values as cue 42 of the selected sequence and directly label it, type:

```
MA User name[Fixture]> Store Cue 42 "Return of the Paranoid Android"
```



8.3.288. SwitchTograndMA2Software Keyword

To enter the SwitchTograndMA2Software keyword in the command line, use one of the options:

- Type **SwitchTograndMA2Software**
- Type the shortcut **SwgMA2**

Description


The SwitchTograndMA2Software keyword is a command keyword that is used to switch to Mode2 on a grandMA3 station.

Syntax

SwitchTograndMA2Software

Example

- To switch from the grandMA3 software to Mode2, type:

```
 User name[Fixture]> SwitchTograndMA2Software
```



8.3.289. SwitchTograndMA3Software Keyword

To enter the SwitchTograndMA3Software keyword in the command line, use one of the options:

- Type **SwitchTograndMA3Software**
- Type the shortcut **SwgMA3**

Description

The SwitchTograndMA3Software keyword is a command keyword that is used to remotely switch to grandMA3 software on a grandMA3 station that is currently running in Mode2.

Syntax

SwitchTograndMA3Software IP [IP_Address]

Example

- To remotely switch a grandMA3 station with the IP 192.168.0.4 from Mode2 to grandMA3 software, type:

```
 User name[Fixture]> SwitchTograndMA3Software IP 192.168.0.4
```




8.3.290. Swap Keyword

To enter the Swap keyword in the command line, use one of the options:

- Type **Swap**
- Type the shortcut **Swa**

Description

The Swap keyword is a playback keyword that starts the playback of a sequence and pulls down the dimmer of all other fixtures that are not part of the sequence or that are protected against Swap.

For more information on how to assign executors see [Assign Objects to an Executor](#).

- To disable the executor, release the executor key.

Pressing the executor button plays back the sequence using the swap functionality.

If you swap sequences that have the same playback master, one will go to zero should it not be swap-protected.

For more information on Swap Protect see [Sequence Settings](#).

Syntax

Swap (On/Off) [Object] ["Object_Name" or Object_Number]

Example

- To playback sequence 5 and pull down the dimmers of all other running playbacks to 0, type:

```
MA User name[Fixture]> Swap Sequence 5
```



8.3.291. Temp Keyword

To enter the Temp keyword in the command line, use one of the options:

- Press **MA** + **Go+**
- Type **Temp**
- Type the shortcut **Te**

Description

The Temp keyword is a function keyword that enables an executor as long as you hold the executor key. The Temp keyword follows cue timing, off timing, and the position of the master fader on the executor.

- To disable the executor, release the executor key.

For information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

Temp (On/Off) [Object] ["Object_Name" or Object_Number]

Examples

- To temporarily enable executor 104, type:

```
MA User name[Fixture]> Temp On Executor 104
```

- To temporarily disable executor 104, type:

```
MA User name[Fixture]> Temp Off Executor 104
```



8.3.292. Thru Keyword

To enter the Thru keyword in the command line, use one of the options:

- Press **Thru**
- Type **Thru**
- Type the shortcut **T**

Description

The Thru keyword indicates the range of objects or values.

If the beginning or the end of the range is not defined, the objects or values that were used last are used again.

Syntax

[Beginning of Range] Thru [End of Range]

Examples

- To select fixture 3 through 6, type:

```
MA User name[Fixture]> Fixture 3 Thru 6
```

- To select all fixtures starting with the first fixture and ending with fixture 10, type:

```
MA User name[Fixture]> Fixture Thru 10
```

- To delete all cues, beginning with cue 3 in the selected sequence, type:

```
MA User name[Fixture]> Delete Cue 3 Thru
```

- To select all fixtures in the fixture sheet of the current world, type:

```
MA User name[Fixture]> Fixture Thru
```

- To disable all executors in the current page, type:



```
MA User name[Fixture]> Off Thru
```

8.3.293. Time Keyword

To enter the Time keyword in the command line, type **Time**.

Description

The Time keyword is used to toggle the time overwrite function on sequences.

For more information see [Cue Timing topic](#).

See the [FaderTime keyword](#) to learn how to set the time.

For information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

Time (On/Off) [Object] ["Object_Name" or Object_Number]



Hint:

If the sequence ID is not defined, the sequence that is selected will be affected.

Examples

- To toggle the time function on the selected sequence, type:

```
MA User name[Fixture]> Time
```

- To enable the time overwrite for sequence 4, type:

```
MA User name[Fixture]>Time On Sequence 4
```



8.3.294. Timecode Keyword

To enter the Timecode keyword in the command line, use one of the options:

- Press **MA** + **X6**
- Type **Timecode**
- Type the shortcut **Ti** or **TC**

Description

The Timecode keyword is an object keyword used to select the timecode by default.

Using the Timecode keyword you can:

- store
- play (go)
- record
- edit
- label
- set properties of
- rewind (top)

timecode shows.

For more information see [Timecode](#).

Syntax

[Function] Timecode ["Timecode_Name" or Timecode_Number] ["Property_Name" or "Property_Value"]



Property

The following table displays the properties that can be assigned using the **Set Keyword**.

Property	Property Value	Description
Name	"Timecode show name"	Set the name of timecode show.
Time		Set position of the time cursor.
Duration	0s to 255h59m58.96s	Set the entire length of timecode show.
Offset	0s to 255h59m58.96s	To move the entire timecode show forward, set an offset in the timecode show.
Loop Mode	Loop	Basic settings for Loop and how to pause or stop.
	Pause	
	Off	
Loop Count	"Endless Repeat" (0),	Only for timecode shows syncing to the internal clock. Set if the timecode show runs:
	"No Repeat" (1),	Endless (Endless Repeat)
	2..1000	Once and stop (No Repeat)
		A specific number of times (number between 2 and 1000)
TimeMarkers		Timemarkers are used to select the track. They will be stored.
TCSlot	"Internal" (-2), "Link Selected" (-1), "Slot1" (0), "Slot2" (1), "Slot3" (2), etc.	Set the timecode show to a timecode slot.
AutoStart (only available with external timecode slot)	"Yes", "No"	Only available when syncing to an external source. If a timecode signal is received, the timecode show switches from the off mode into the play mode.
AutoStop	"Yes", "No"	Only available when syncing to an external source. If a timecode signal is received, the timecode show switches from the play mode into the off mode.
User Bits	0 .. FFFFFFFF, 0 .. 4294967296	Only available when syncing to an external source. To transmit several kinds of information, for example, a second Timecode Stream, set user bits in hex or decimal. So several incoming Timecode Streams can be discerned.

Examples

- To set timecode show "Intro" to a duration of 55 seconds, type:

```
MA User name[Fixture]> Set Timecode "Intro" "Duration" "55"
```



- To store a new timecode show called "Napalm Skies" in the timecode pool, type:

```
MA User name[Fixture]> Store Timecode "Napalm Skies"
```

- To rename timecode show "Intro" "Prelude", type:

```
MA User name[Fixture]> Set Timecode "Intro" "Name" "Prelude"
```

- To start the timecode show "Prelude", type:

```
MA User name[Fixture]> Go Timecode "Prelude"
```

- To rewind the timecode show "Prelude", type:

```
MA User name[Fixture]> Top Timecode "Prelude"
```

- To label timecode show 3 in the timecode pool, type:

```
MA User name[Fixture]> Label Timecode 3
```

The label pop-up opens and you can now label the timecode show.



8.3.295. TimecodeSlot Keyword

To enter the TimecodeSlot keyword in the command line, use one of the options:

- Press **MA** + **X6** + **X6**
- Type **TimecodeSlot**
- Type **TimecodeS**

Description

The TimecodeSlot keyword is an object keyword used to address the timecode slots.

For more information see [What are Timecode Slots.](#)

Syntax

[Function] TimecodeSlot ["TimecodeSlot_Name" or TimecodeSlot_Number]

Examples

- To select timecode slot 4, type:

```
MA User name[Fixture]> Select TimecodeSlot 4
```

- To start the timecode generator of TimecodeSlot 2, type:

```
MA User name[Fixture]> Go+ TimecodeSlot 2
```

- To stop the timecode generator of timecode slot 2, type:

```
MA User name[Fixture]> Off TimecodeSlot 2
```



8.3.296. Timer Keyword

To enter the Timer keyword in the command line, use one of the options:

- Press **MA** + **X6 | TC** + **X6 | TC** + **X6 | TC**
- Type **Timer**

Description

Timer is an object keyword used to access the timer objects.

Timers are stored in the Timer pool. There are two types of timers:

- **Stopwatch** - counting up from zero.
- **Countdown** - counting down to zero.

They can be controlled using the Timer pool or by the command line.

Learn more about timers in the [Timers topic](#).

Syntax

[Function] Timer ["Timer_Name" or Timer_Number] (Property ["Setting"] ["Setting_Value"])

Settings

Some settings need an object assigned. These can be assigned using the [Assign Keyword](#).

Other settings contain a text option or a value. The [Set keyword](#) is used for these settings.

The following table displays the settings that need an object:

Setting	Object	Description
Appearance	"Appearance 1"	Assigns the appearance for the pool object.
Scribble	"Scribble 1"	Assigns the scribble for the preset pool object.



The following table displays the settings that need an option or value:

Setting	Option/Value	Description
Name	"Timer Name"	Sets the timer pool object name.
Note	"This is a note"	This is the note to the timer pool object.
TimerMode	"Stopwatch" / "Countdown"	The mode for the timer.
TimeReadout	One of the four readout options	The readout option for the timer.
RestartOption	"Continue on Go+" / "Reset on Go+"	Sets if the timer should restart or continue with a new Go+.
TimerLinkType	One of the five link options	Timers can be linked to sequences and can be activated or toggled.
CountdownAlertType	One of the four alert type options	Defines what should happen when a countdown is finished.
CountdownAlertRange	"Local" / "All Stations"	Defines where an alert pop-up should appear.
CountdownDuration	Time value	Sets the countdown duration time.
AlertCommand	"Command Line Input"	The command that can be executed when the countdown is finished.
AlertText	"Text that appears in a pop-up"	The text is displayed in an alert pop-up.
FrameFormat	One of the five frame format options	A setting that defines which of five formats is used with the timer.
AlertDuration	Time value	The time duration the alert pop-up is displayed.

Read the **Timer topic** for more explanations about the settings.

Examples

- Run timer 1 with this command:

```
MA User name[Fixture]> Go Timer 1
```

- Change the timer mode for timer 2:

```
MA User name[Fixture]> Set Timer 2 "TimerMode" "Countdown"
```

- Set timer 2 to a countdown time of 25 seconds:

```
MA User name[Fixture]> Set Timer 2 "CountdownDuration" 25
```



8.3.297. Toggle Keyword

To enter the Toggle keyword in the command line, use one of the options:

- Press and hold **MA** and press **Go+** **Go+**
- Type **Toggle**
- Type the shortcut **To**

Description

The Toggle keyword is a playback and a helping keyword that behaves as an On or Off keyword, depending on the on/off status of an object, function, or mode it is applied to.

If an object, function, or mode is enabled, Toggle disables it. If an object, function, or mode is disabled, Toggle enables it.

For more information see the [On](#) and [Off Keywords](#).

For information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

Toggle [Object] ["Object_Name" or Object_Number]

Examples

- To disable the running executor 104, type:

```
MA User name[Fixture]> Toggle Executor 104
```



8.3.298. TopUp Keyword

To enter the TopUp keyword in the command line, use one of the options:

- Press **Set** + **Up**
- Type **TopUp**
- Type the shortcut **Topu**

Description

The TopUp keyword is a function keyword used to select the highest level in the hierarchy of multi-instance fixtures whenever subfixtures are selected.

Syntax

TopUp

Example

- To jump to the main instance of the fixture within which subfixtures are selected, type:

```
MA User name[Fixture]> TopUp
```



8.3.299. Top Keyword

To enter the Top keyword in the command line, use one of the options:

- Press **MA** + **Go-**
- Type **Top**

Description

The Top keyword is a playback keyword used to jump to the beginning of a cue list or to set a timecode marker at the beginning of a timecode show.

For information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

Top [Object] ["Object_Name" or Object_Number]

Examples

- To jump to the top of the cue list on executor 105, type:

```
MA User name[Fixture]> Top Executor 105
```

- To jump to the top of the cue list on executor 105 in 3 seconds, type:

```
MA User name[Fixture]> Top Executor 105 Fade 3
```

- To jump to the top of the Timecode show 1, type:

```
MA User name[Fixture]> Top Timecode 1
```



8.3.300. Transition Keyword

To enter the Transition keyword in the command line, use one of the options:

- Type **Transition**
- Type the shortcut **Tr**

Description

The Transition keyword is used to determine the length of the passage in a step.

For more information see [Phasers](#).

Syntax

Transition

[At] Transition [Value]

Examples

- To set the layer to transition, type:

```
MA User name[Fixture]> Transition
```

- To shorten the transition of a step to a half, type:

```
MA User name[Fixture]>At Transition 50
```



8.3.301. Type Keyword

To enter the Type keyword in the command line, use one of the options:

- Type **Type**
- Type the shortcut **Ty**

Description

The Type keyword is used to address children of an object.

Syntax

[Function] [Object] [Number/"Name"] Type "Name of Type"

Example

- To store cue part 1 in a newly created sequence without specifying a cue number, type:

```
MA User name[Fixture]> Store Part 1 Type "Part"
```



Hint:

To circumvent the pop-up "Select type of object", press an executor immediately after typing this command.



8.3.302. UIChannel Keyword

To enter the UIChannel keyword in the command line, use one of the options:

- Type **UIChannel**
- Type the shortcut **Ui**

Description

The UIChannel is an object keyword that represents an attribute in the fixture sheet.



Important:

Depending on the fixtures that are patched in the show file, the attributes that are represented in a UIChannel may differ.

Syntax

[List] UIChannel

UIChannel [Number of UIChannel] At [Value]

Examples

- To list all the UIChannels in the command line, type:

```
MA User name[Fixture]> List UIChannel
```

- To set the UIChannel 9 to a value of 42, type:

```
MA User name[Fixture]> UIChannel 9 At 42
```



8.3.303. UIGridSelection Keyword

To enter the UIGridSelection keyword in the command line, use one of the options:

- Type **UIGridSelection**
- Type the shortcut **UIG**

Description

UIGridSelection keyword is used by the system to hold information about what objects are selected in a grid window. Grid windows include sheets. It does **not** include the Selection Grid window.

The keyword is primarily used internally by the system, but the selection can be used for normal operations that are done on the selected objects.



Important:

The grid with the selection needs to have the focus for the UIGridSelection command to work.

Syntax

[Function] UIGridSelection

Example

Requirement:

- Some cues in a sequence.
- A macro with a **Copy UIGridSelection** command. This is important to be able to execute the command without moving the focus.
- To copy a selection of the cues to cue 41, using the UIGridCommand do the following steps:
 1. Select the desired cues in the sequence sheet.
 2. Run the macro using the command keys (Important to use the keys to keep the focus in the cue selection).
 3. Now the following command can be entered and executed in the command line:



```
User name[Fixture]> Paste Cue 41
```

8.3.304. Unblock Keyword



Important:

To block parameters, use the **Block keyword**.

To enter the Unblock keyword in the command line, use one of the options:

- Type **Unblock**
- Type the shortcut **UB** or **Unb**

Description

Unblock is a function that converts blocked values into tracking values in cues.

If the object list does not contain any references to any cues, the unblock function is applied to the selected sequence.

If unblock does not contain any selection list filters, all fixtures will be used.

If unblock does not contain any attribute list filters, all attributes will be used.

Syntax

Unblock (Sequence ["Sequence_Name" or Sequence_Number]) Cue ["Cue_Name" or Cue_Number] (If Fixture ["Fixture_Name" or Fixture_Number] FeatureGroup ["FeatureGroup_Name" or FeatureGroup_Number] or Attribute ["Attribute_Name" or Attribute_Number] EndIf)

Examples

- To unblock all parameters of the selected sequence, type:

```
MA User name[Fixture]> Unblock
```

- To unblock pan and tilt of fixture 4 in cue 5 of the selected sequence, type:

```
MA User name[Fixture]> Unblock Cue 5 If Fixture 4 Attribute "Position" EndIf
```



8.3.305. Universal Keyword

To enter the **Universal** keyword in the command line, use one of the options:

- Press **Channel Channel**
- Type **Universal**
- Type the shortcut **Uni**
- Press **F + 2** on a keyboard

Description

The Universal keyword is an object keyword used to call fixtures of the ID type Universal in the programmer.

Syntax

Universal [ID]

Example

- To select the fixture that uses the Universal ID 1, type:

```
MA User name[Fixture]>Universal 1
```

8.3.306. Unlock Keyword



Hint:

A padlock indicates a locked object.



Hint:

To lock objects, use the **Lock Keyword**.

To enter the Unlock keyword in the command line, use one of the options:

- Type **Unlock**
- Type the shortcut **UI** or **Unl**

Description

The Unlock keyword is a function keyword used to unlock objects that were previously locked.

Syntax

Unlock [Object] ["Object_Name" or Object_Number]

Examples

- To unlock cue 3 to edit the cue, type:

```
MA User name[Fixture]> Unlock Cue 3
```

- To unlock world 3 to edit the world, type:

```
MA User name[Fixture]> Unlock World 3
```

- To unlock the macro line 2 of the macro 1, type:

```
MA User name[Fixture]> Unlock Macro 1.2
```

8.3.307. Unpark Keyword

To enter the Unpark keyword in the command line, use one of the options:

- Press **Go+** **Go+**
- Type **Unpark**
- Type **Unp**

Description

The Unpark keyword is a command keyword used to release a parked DMX channel or universe.



Hint:

If you unpark a universe, the blue **P** icon disappears in the universe pool.

Syntax

Unpark [Object] ["Object_Name" or Object_Number]

Examples

- To unpark fixture 1 with all its attributes, type:

```
MA User name[Fixture]> Unpark Fixture 1
```

- To unpark the current selection, type:

```
MA User name[Fixture]> Unpark
```

- To unpark DMX universe 2, type:

```
MA User name[Fixture]> Unpark DMXUniverse 2
```



8.3.308. Unpress Keyword

To enter the Unpress keyword in the command line, use one of the options:

- Type **Unpress**
- Type the shortcut **Unpr**

Description

The Unpress keyword is used to simulate the unpressed state of a key.

It is possible to assign this state to macros.

For more information see [Macros](#).

For information on how to assign executors see [Assign Objects to an Executor](#).

Syntax

[Function] Unpress [Object] ["Object_Name" or Object_Number]

Example

- To disable Flash on the executor 203, type:

```
MA User name[Fixture]> Flash Unpress Executor 203
```



8.3.309. Up Keyword

To enter the Up keyword in the command line, use one of the options:

- Press **Up**
- Type **Up**

Description

The Up keyword is a function keyword used to navigate upward in the fixture structure.

Syntax

Up

Example

- To go back to the main fixture in the fixture sheet in an upward manner, type:

```
MA User name[Fixture]> Up
```




8.3.310. Update Keyword

To enter the Update keyword in the command line, use one of the options:

- Press **Update**
- Type **Update**
- Type the shortcut **U**

Description

The Update keyword is a function keyword used to update values in their source objects.

Syntax

Update [Object List]

Option Keywords

The Update Keyword uses the following option keywords:

- **/AddNewContent**
- **/ForceGlobal**
- **/Global**
- **/InputFilter**
- **/OriginalContentOnly**
- **/Selective**
- **/Universal**

Example

- To update Preset 4.1, type:

```
MA User name[Fixture]> Update Preset 4.1
```



8.3.311. UpdateContent Keyword

To enter the UpdateContent keyword in the command line, use one of the options:

- Type **UpdateContent**
- Type the shortcut **UC** or **UpdateC**

Description

The UpdateContent keyword is used to scan a media pool, for example images, and create XML files for media files.

Syntax

UpdateContent [Object] ["Object_Name" or Object_Number]

Example

The following example is explained using images.

Requirement:

1. A media file, for example an image, was added to the corresponding media folder, for example in grandMA3_lib/media/images, without an XML file.
2. Enter the media pool folder and then the image folder.
For more information see [ChangeDestination keyword](#).

- To create the XML files that are missing in the image folder of the media pool, type:

```
MA User name@ShowData/MediaPools/Images> UpdateContent Image
```



8.3.312. User1 Keyword



Important:

The **User1** keyword can only be used on the grandMA3 compact, the grandMA3 compact XT, and the grandMA3 onPC command wing where the user key is located on the right of Special Executor 1 and 2.



Hint:

The **User1** keyword must be used as one word.

To enter the **User1** keyword in the command line, use one of the options:

- Press **U1**
- Type **User1**

Description

The **User1** keyword is a function keyword that toggles between the encoder bar and the Xkeys bar.

Syntax

User1

Example

- To toggle the encoder bar to the Xkeys bar, type:

```
 User name[Fixture]> User1
```



8.3.313. User2 Keyword



Important:

The **User2** keyword can only be used on the grandMA3 compact, the grandMA3 compact XT, and the grandMA3 onPC command wing where the user key is located on the right of Special Executor 1 and 2.



Hint:

The **User2** keyword must be used as one word.

To enter the **User2** keyword in the command line, use one of the options:

- Press **U2**
- Type **User2**

Description

The **User2** keyword is a function keyword that toggles between the encoder bar and the executor bar.

Syntax

User2

Example

- To toggle the encoder bar to the executor bar, type:

```
 User name[Fixture]> User2
```



8.3.314. User Keyword

To enter the User keyword in the command line, use one of the options:

- Type **User**
- Type the shortcut **Us**

Description

The User keyword is used for login or to change the user settings.

Learn more about the User and the different setting in the [User Setting topic](#).

Syntax

User [User_Number or "User_Name"]

[Function] User [User_Number or "User_Name"] ([Setting] [Setting_Value])

Settings

The User keyword has some settings. The settings can be changed using the [Set Keyword](#). When the setting needs a pool object, then the [Assign keyword](#) might be used to assign an object to the setting.

These are the settings that can be viewed and changed:

Setting	Object/Option/Value	Description
Name	Text	The name of the user. This can be used as user ID.
Scribble	Scribble pool object	The Scribble assigned to the User object.
Appearance	Appearance pool object	The Appearance assigned to the User object.
Password	Text	The password is not shown in clear text.
Profile	UserProfile object	The assigned User Profile.
ScreenConfig	Screen configuration object	The active screen configuration.
Rights	"Admin", "Setup", "Program", "Preset", "Playback", "View", or "None"	The access right assigned to the user.
Language	"de", "en", "ru", or "dk"	The language assigned to the user.
Keyboard	"German", "English", "Russian", or "Danish"	The keyboard layout assigned to the users onscreen keyboard.

Example

- To list the details of all users, type:

```
MA User name[Fixture]> List User
```



- To login as "Admin" user, type:

```
MA User name[Fixture]> User Admin
```

Alternatively see the [Login Keyword](#).

- To assign appearance number 1 to User 2

```
MA User name[Fixture]> Assign Appearance 1 At User 2
```

- To change the rights to Playback for user Remote, type:

```
MA User name[Fixture]> Set User "Remote" "Rights" "Playback"
```



8.3.315. UserProfile Keyword

To enter the UserProfile keyword in the command line, use one of the options:

- Type **UserProfile**
- Type **Userp**
- Type the shortcut **UPR**

Description

The keyword UserProfile is an object keyword. It can be used to perform some functions on the UserProfile, for instance, create, delete, or change a setting.

Learn more about the user profiles in the [Create User](#) topic and learn about the different settings in the [User Settings](#) topic.

Syntax

[Function] UserProfile [UserProfile_Number or "UserProfile_Name"] ([Setting] [Setting_Value])

Settings

The UserProfile keyword has some settings. The settings can be changed using the [Set Keyword](#).

These are the settings that can be viewed and changed:

Setting	Object/Option/Value	Description
Name	Text	The name of the user profile. This can be used as the user profile ID.
DMXReadout	"Percent", "Dec8", "Dec16", "Dec24", "Hex8", "Hex16", or "Hex24"	This is the default DMX readout.
NormalValue	A percent value	This is the value assigned to the intensity attribute when the Normal keyword is used.
WheelResolution	"Coarse", "Normal", or "Fine"	The resolution of the level wheel on the consoles.
WheelMode	"Additive", "Incremental", "Prop.+", or "Prop.-"	This defines the level wheel mode.
PreciseEdit	"Yes" or "No"	This turns the precise edit mode On or Off.
ScreenEncoder	"Yes" or "No"	This enables or disables the screen encoder on the fifth dual encoder .
TimeKeyTarget	"Cue" or "Fixture"	This defines how the Time key functions.
TCSlot	"TCSlot 1" .. "TCSlot 8"	This is the user profile's selected timecode slot.



Setting	Object/Option/Value	Description
ValueReadout	"Natural", "Percent", "PercentFine", "Physical", "Decimal8", "Decimal16", "Decimal24", "Hex8", "Hex16", or "Hex24"	This is the default value readout.
SpeedReadout	"Hertz", "BPM", or "Seconds"	This is the default speed readout.
PresetReadout	"ID", "ID+Name", "ID+Name+Value", "Name", "Name+Value", or "Value"	This is the default readout for preset in sheets.
OverlayFade	Time value in milliseconds	This is the time used by pop-ups and overlays for fading into visibility.
TimeReadout	"10d11h23m45", "251h23m45", "10.11:23:45", or "251:23:45"	This is the default time readout.
FrameReadout	"Seconds", "24 fps", "25 fps", "30 fps", or "60 fps"	This is the default time frame readout.
UndoProgrammer	"Yes" or "No"	This turns On or Off if programmer actions are oops'able.
UndoGeneral	"Yes" or "No"	This turns On or Off if general actions are oops'able.
AutoRemoveGaps	"Yes" or "No"	This turns On or Off the auto-remove gaps function in the selection grid.
MirrorSpecialExecutorPages	"Yes" or "No"	This turns On or Off the mirror function on extension wings.
ShowAppearancelnCueInput	"Yes" or "No"	This turns On or Off the cue appearance in cue selection pop-ups.
ShowSettingsInEditors	"Yes" or "No"	This turns On or Off if settings are visible in editors as a default.
ExecConfigSequence	Executor Configuration Object	Default executor configuration for sequences.
ExecConfigMacro	Executor Configuration Object	Default executor configuration for macros.
ExecConfigView	Executor Configuration Object	Default executor configuration for views.
ExecConfigWorld	Executor Configuration Object	Default executor configuration for worlds.
ExecConfigGroup	Executor Configuration Object	Default executor configuration for groups.
ExecConfigPreset	Executor Configuration Object	Default executor configuration for presets.
ExecConfigPlugin	Executor Configuration Object	Default executor configuration for plugins.
ExecConfigUser	Executor Configuration Object	Default executor configuration for users.
ExecConfigSound	Executor Configuration Object	Default executor configuration for sounds.
ExecConfigScreenConfig	Executor Configuration Object	Default executor configuration for screen configurations.
ExecConfigMaster	Executor Configuration Object	Default executor configuration for masters.



Setting	Object/Option/Value	Description
ExecConfigSpeedMaster	Executor Configuration Object	Default executor configuration for speed masters.
ExecConfigPlaybackMaster	Executor Configuration Object	Default executor configuration for playback masters.
ShowConnectors	"Yes" or "No"	This turns On or Off if the connector overlay is visible when the "Output Configuration" menu is visible.
FixtureLibShowMA	"Yes" or "No"	This turns On or Off if MA fixtures are shown in the "Insert New Fixture" pop-up.
FixtureLibShowUser	"Yes" or "No"	This turns On or Off if User fixtures are shown in the "Insert New Fixture" pop-up.
FixtureLibShowShare	"Yes" or "No"	This turns On or Off if Share fixtures are shown in the "Insert New Fixture" pop-up.

Explanations of the different settings are explained in more detail in the [User Settings topic](#).

Example

- To list all available user profiles, type:

```
MA User name[Fixture]> List UserProfile
```

- To turn Off the screen encoder for the Default user profile, type:

```
MA User name[Fixture]> Set UserProfile "Default" "ScreenEncoder" "No"
```



8.3.316. Video Keyword

To enter the Video keyword in the command line, use one of the options:

- Type **Video**
- Type the shortcut **Vid**

Description

The Video keyword is an object keyword. It is used to manage videos using the command line.

Syntax

Video [ID/Name]

[Function] Video [ID/Name]

Examples

- To edit video 1 in the video pool, type:

```
MA User name[Fixture]> Edit Video 1
```

- To delete video 1 in the video pool, type:

```
MA User name[Fixture]> Delete Video 1
```



8.3.317. Version Keyword

To enter the Version keyword in the command line, use one of the options:

- Type **Version**
- Type the shortcut **Ve**

Description

The Version keyword is a function keyword used to grant access to details of the version in the console or in the onPC software.

It displays the following details in the command line history:

- Version number
- Release type
- Build type
- Code type
- Host type
- Host sub type
- Station serial number
- Date of compilation
- Repository branch
- Repository hash
- Version number of Lua
- Version number of Lua sockets
- Version number of Lua file system

Syntax

Version

Example

- To view the details of the version, type:





```
OK:Menu "CommandLineHistory"  
Application version is 1.5.2.3  
Release type Release  
BuildType Release  
HostType onPC, HostSubType wing-onPC  
Station SN is 1551000000-900009  
Compiled at Aug 3 2021 11:17:37  
Repository branch unknownBranch  
Repository hash 63e789d047b6d2f224122576247907938c591258  
water mark  
Lua core : Lua 5.4  
Lua sockets : LuaSocket 3.0-rc1  
Lua file system : 1.8.0  
OK:Version
```

Version details in the command line history window

The version details are displayed in the command line history window.



8.3.318. View Keyword

To enter the View keyword in the command line, use one of the options:

- Press **MA** + **X7 | View**
- Type **View**
- Type the shortcut **V**

Description

The View keyword calls views on a screen.

The option **/Screen** specifies specific screens related to the command. For instance, calling a view on a specific screen or saving the content of a specific screen or multiple screens.

Syntax

View [View_Number or "View_Name"] (/Option ["Option_Value"])

[Function] View [View_Number or "View_Name"] (/Option ["Option_Value"])

Settings

The following table displays the properties that can be assigned using the **Set keyword**.

Setting	Option/Value	Description
Name	View name, for instance, "Stage External"	The name of the view.
Scribble	Scribble pool object	The assigned scribble object.
Appearance	Appearance pool object	The assigned appearance object.
ScreenContentMask	1 to 127	An internal number that indicates the screens used when it was saved.
RequestedW	1 to 327	The width of the view (grid width).
RequestedH	1 to 327	The height of the view (grid height).

Option Keywords

The **/Screen option keyword** with a value can be used both when storing and calling a view.

Examples

- To call view 2 from the view pool on the screen where it is stored, type:

```
MA User name[Fixture]> View 2
```



- To call view 5 on screen 2, type:

```
MA User name[Fixture]> View 5 /Screen "2"
```

- To store the content of screen 2, 3, and 4 in the view object 5 with the label "Phaser Creation", type:

```
MA User name[Fixture]> Store View 5 "Phaser Creation" /Screen "2, 3, 4"
```



8.3.319. ViewButton Keyword

To enter the ViewButton keyword in the command line, use one of the options:

- Press **MA** + **X7|View** + **X7|View**
- Type **ViewButton**
- Or type the shortcut **ViewB**

Description

The ViewButton keyword is an object keyword used to call the object assigned on the view button.

The call function works only if the assigned object supports it.

For more information see the [Call Keyword](#).

Syntax

ViewButton [Screen_Number].[ViewButton_Number or "ViewButton_Name"] (/Option "[Option_Value]")

[Function] ViewButton [Screen_Number].[ViewButton_Number or "ViewButton_Name"] (/Option "[Option_Value]")

Option Keywords

The [/Screen option keyword](#) with a value can be used both when storing and calling a view using the ViewButton. It selects which screen is used as source or destination.

Examples

- To call the view assigned on ViewButton 4 on screen 2, type:

```
MA User name[Fixture]> ViewButton 2.4
```



Important:

If a screen location is not specified using the /Screen option, then the view will be called on the screen that currently has focus.

- To call the view assigned on ViewButton 1, screen 2 on screen 3, type:

```
MA User name[Fixture]> ViewButton 2.1 /Screen "3"
```

- To assign the user pool object "Guest" to the ViewButton 1.1, type:



```
MA User name[Fixture]> Assign User "Guest" At ViewButton 1.1
```

- To remove the object assigned on ViewButton 4, screen 1, type:

```
MA User name[Fixture]> Delete ViewButton 1.4
```

- To label the view that is assigned to ViewButton 5 on screen 2 "Layout", type:

```
MA User name[Fixture]> Label ViewButton 2.5 "Layout"
```




8.3.320. Width Keyword

To enter the Width keyword in the command line, use one of the options:

- Type **Width**
- Type the shortcut **Wi**

Description

The Width keyword is used to determine the width incorporating transition of the entire step.

For more information see [Phasers](#).

Syntax

Width

[At] Width [Value]

Examples

- To set the layer to width, type:

```
MA User name[Fixture]> Width
```

- To create a PWM effect, type:

```
MA User name[Fixture]> At Transition 0 + Width 50
```



8.3.321. World Keyword

To enter the World keyword in the command line, use one of the options:

- Press **Group** **Group**
- Type **World**
- Or type the shortcut **W**

Description

The World keyword is an object keyword used to call worlds along with their filters, and limit the access to the parameters in the world.

By default, World 1 is always locked and can neither be edited nor deleted. World 1 includes all parameters (fixtures and attributes) of the show.

Syntax

[Function] World ["World_Name" or World_Number]

Option Keywords

The World keyword uses the following option keywords:

- **/GridMergeMode**
- **/Merge**
- **/Overwrite**
- **/Remove**

Examples

- To call world 3, type:

```
MA User name[Fixture]> World 3
```

- To label the world 3 "All Fixtures", type:

```
MA User name[Fixture]> Label World 3 "All Fixtures"
```



8.3.322. Zero Keyword

To enter the Zero keyword in the command line, use one of the options:

- Press **. .**
- Type **Zero**
- Type the shortcut **Z**

Description

The keyword Zero is a helping keyword used to set the intensity of the dimmer to zero if fixtures or channels are selected.

Syntax

[Object List] Zero

Examples

- To set the intensity of the selected fixtures or channels to zero, type:

```
MA User name[Fixture]> Zero
```

- To set the intensity of fixture 1 through 10 to zero, type:

```
MA User name[Fixture]> Fixture 1 Thru 10 Zero
```

8.4. Option Keywords

Option keywords can be seen as criteria. These criteria are used in commands to temporarily determine their result. The result is always a single event - that is, it will not be set permanently.

To make it easier for you to understand how such option keywords work, we will use an everyday situation at a pizza place.

Example

- Pizza order

Let's say you are a regular at your favorite pizza place where you always order your pizza with Mozzarella topping. The owner of the place knows that you always eat this pizza. Whenever you call and say you are having a pizza they already arrange everything and the only thing you have to do is go and pick up your food.

But what if one day you decide to have a Basil topping instead?! Duh dum!

You call your pizza guy and explicitly say that you are having a Basil pizza this time. The next time around your pizza guy will stick to your "default Mozzarella topping", except you explicitly tell them that you would like to have a different topping this time.



Hint:

Mind you, you can always use your favorite dish in this example - burgers, kebab, steamed buns, sauerbraten, varenyky, borscht...



8.4.1. /Active Option Keyword

To enter the **/Active** option keyword in the command line, use one of the options:

- Type **/Active**
- Type the shortcut **/Ac**

Description

The **/Active** option keyword stores all active values of the programmer no matter the selection.

Syntax

[Function] [Object] ["Object_Name" or Object_Number] /Active

General Keywords

General keywords that use the **/Active** option keyword:

- **Cue keyword**
- **Preset keyword**
- **Store keyword**

Example

Requirement:

- Active values in the programmer
- To store all active values in cue 1 of the selected sequence, type:

```
MA User name[Fixture]> Store Cue 1 /Active
```



8.4.2. /ActiveForSelected Option Keyword

To enter the **/ActiveForSelected** option keyword in the command line, use one of the options:

- Type **/ActiveForSelected**
- Type the shortcut **/ActiveF**

Description

The **/ActiveForSelected** option keyword stores all active attributes of the selected fixtures in the programmer.

Syntax

[Function] [Object] ["Object_Name" or Object_Number] /ActiveForSelected

General Keywords

General keywords that use the **/ActiveForSelected** option keyword:

- **Cue keyword**
- **Preset keyword**
- **Store keyword**

Example

Requirement:

- Selected fixtures with active values in the programmer
- To store all active attributes of the selected fixtures in cue 1, type:

```
MA User name[Fixture]> Store Cue 1 /ActiveForSelected
```



8.4.3. /AddNewContent Option Keyword

To enter the **/AddNewContent** option keyword in the command line, use one of the options:

- Type **/AddNewContent**
- Type the shortcut **/Ad**

Description

The **/AddNewContent** option keyword updates existing values of attributes and adds new values in presets or cues.

Syntax

Update [Object] ["Object_Name" or Object_Number] /AddNewContent

General Keywords

General keywords that use the **/AddNewContent** option keyword:

- **Cue keyword**
- **Preset keyword**
- **Update keyword**

Example

Requirement:

- Create presets or cues
- Playback presets or cues or activate presets in the programmer
- At least one attribute must have active values in the programmer

- To update existing and new attribute values in Preset 1.1, type:

```
 User name[Fixture]> Update Preset 1.1 /AddNewContent
```



8.4.4. /All Option Keyword

To enter the **/All** option keyword in the command line, use one of the options:

- Type **/All**
- Type the shortcut **/AI**

Description

The **/All** option keyword stores the values of all attributes when using Data Source Output or DMX.

Syntax

[Function] [Object] ["Object_Name" or Object_Number] /All

General Keywords

General keywords that use the **/All** option keyword:

- **AutoCreate keyword**
- **Cue keyword**
- **LoadShow keyword**
- **NewShow keyword**
- **Preset keyword**
- **Store keyword**



Hint:


The **/All** option keyword also works in conjunction with the **/Look option keyword**.

Examples

- To store all attributes in cue 1, type:

```
 User name[Fixture]> Store Cue 1 /All
```

- To load the entire data of the show "Fiddler on the Roof", type:

```
 User name[Fixture]> LoadShow "Fiddler on the Roof" /All
```

- To create a new show file called "Rocky Horror Picture Show" and clear all previous settings, type:



```
MA User name[Fixture]> NewShow "Rocky Horror Picture Show" /All
```

Requirement:

1. Select eight fixtures.
 2. Set MAtricks XGroup to 2.
 3. Set MAtricks X to 0.
- To create group 21 in the group pool containing all main fixtures of the selection (odd fixtures of the current selection), type:

```
MA User name[Fixture]> AutoCreate Selection At Group 21 /All
```



8.4.5. /AllForSelected Option Keyword

To enter the **/AllForSelected** option keyword in the command line, use one of the options:

- Type **/AllForSelected**
- Type the shortcut **/Allf**

Description

The **/AllForSelected** option keyword stores all attributes of the selected fixtures in the programmer.

Syntax

[Function] [Object] ["Object_Name" or Object_Number] /AllForSelected

General Keywords

General keywords that use the **/AllForSelected** option keyword:

- **Cue keyword**
- **Preset keyword**
- **Store keyword**




Hint:

The **/AllForSelected** option keyword also works in conjunction with the **/Look option keyword**.

Example

Requirement:

- Selected fixtures with values in the programmer
- To store all attributes of the selected fixtures in cue 1, type:

```
 User name[Fixture]> Store Cue 1 /AllForSelected
```



8.4.6. /Ask Option Keyword

To enter the **/Ask** option keyword in the command line, use one of the options:

- Type **/Ask**
- Type the shortcut **/A**

Description

The **/Ask** option keyword is a default setting which is used to define the store option in an object that already has values. The **/Ask** option keyword triggers a pop-up where it is possible to set the store options.

Syntax

Store [Object] ["Object_Name" or Object_ID] /Ask

General Keywords


General keywords that use the **/Ask** option keyword:

- Cue keyword
- Filter keyword
- Group keyword
- MAtricks keyword
- Preset keyword
- Sequence keyword
- Store keyword
- World keyword

Example

Requirement: An existing cue

- To store the new value to the existing cue and define how to store data to the destination, type:

```
 User name[Fixture]> Store Sequence 3 Cue 1 /Ask
```



8.4.7. /Auto Option Keyword

To enter the **/Auto** option keyword in the command line, use one of the options:

- Type **/Auto**
- Type the shortcut **/Au**

Description

The **/Auto** option keyword is used to store and update presets. Executing the **/Auto** option keyword sets the preset mode to **Auto**. For more information see [Presets](#).

Syntax

[Function] Preset ["FeatureGroup_Name" or FeatureGroup_ID].["Preset_Name" or Preset_ID] /Auto

General Keywords

General keywords that use the **/Auto** option keyword:

- [Preset keyword](#)
- [Store keyword](#)
- [Update keyword](#)

Example

- To store the active values into color preset 3 using the Preset Mode "Auto", type:

```
 User name[Fixture]> Store Preset 4.3 /Auto
```



8.4.8. /CopyCueDestination Option Keyword

To enter the **/CopyCueDestination** option keyword in the command line, use one of the options:

- Type **/CopyCueDestination**
- Type the shortcut **/CopyCueD** or **/CCD**

Description

The **/CopyCueDestination** option keyword defines how the copied data is used in the destination cue when copying cues.

For more information see [Copy Cues](#).

Syntax

Copy (Sequence ["Source_Sequence_Name" or Source_Sequence_ID]) Cue ["Source_Cue_Name" or Source_Cue_ID] At (Sequence ["Destination_Sequence_Name" or Destination_Sequence_ID]) Cue ["Destination_Cue_Name" or Destination_Cue_ID] /CopyCueDestination "Value"

General Keywords

General keywords that use the **/CopyCueDestination** option keyword:

- [Cue keyword](#)
- [Copy keyword](#)

Values

The **/CopyCueDestination** option keyword uses these values:

- Merge
- Overwrite

Example

Requirement: Create cues 1 and 2

- To copy cue 1 to cue 2 and merge the data of cue 1 with cue 2, type:

```
MA User name[Fixture]> Copy Cue 1 At Cue 2 /CopyCueDestination "Merge"
```



8.4.9. /CopyCueSource Option Keyword

To enter the **/CopyCueSource** option keyword in the command line, use one of the options:

- Type **/CopyCueSource**
- Type the shortcut **/Cop**, **/CC** or **/CCS**

Description

The **/CopyCueSource** option keyword defines how data is extracted in the source cue when copying cues.

For more information see [Copy Cues](#).

Syntax

Copy (Sequence ["Source_Sequence_Name" or Source_Sequence_ID]) Cue ["Source_Cue_Name" or Source_Cue_ID]
At (Sequence ["Destination_Sequence_Name" or Destination_Sequence_ID]) Cue ["Destination_Cue_Name" or
Destination_Cue_ID] /CopyCueSource "Value"

General Keywords

General keywords that use the **/CopyCueSource** option keyword:

- [Cue keyword](#)
- [Copy keyword](#)

Values

The **/CopyCueSource** option keyword uses these values:

- Content
- Status
- Look

Example

- To copy the status of cue 1 to cue 2, type:

```
MA User name[Fixture]> Copy Cue 1 At Cue 2 /CopyCueSource "Status"
```



8.4.10. /CreateReferenceObject Option Keyword

To enter the **/CreateReferenceObject** option keyword in the command line, use one of the options:

- Type **/CreateReferenceObject**
- Type the shortcut **/CreateR**

Description

The **/CreateReferenceObject** option keyword creates a referenced object when an object is imported. For example, an appearance which uses the imported image.

For more information on importing references see [Import/Export Menu](#).

Syntax

**Import [Object Type] Library "File Name.file_type" At [Object Type] ([Object_Number] [Object ID])
/CreateReferenceObject**

General Keywords

General keywords that use the **/CreateReferenceObject** option keyword:

- [Import keyword](#)
- [Image keyword](#)

Example

- To import an image and automatically create an appearance which references to the imported image, type:

```
MA User name[Fixture]> Import Image Library "Cloud.png" At Image 3.11  
/CreateReferenceObject
```



8.4.11. /CreateSecondCue Option Keyword

To enter the **/CreateSecondCue** option keyword in the command line, use one of the options:

- Type **/CreateSecondCue**
- Type the shortcut **/C**

Description

The **/CreateSecondCue** option keyword stores a cue using the next whole number provided the sequence has only one cue.

Syntax

[Function] (Sequence ["Sequence_Name" or "Sequence_ID"]) /CreateSecondCue

General Keywords

General keywords that use the **/CreateSecondCue** option keyword:

- **Cue keyword**
- **Store keyword**

Example

- To store a second cue in the selected sequence, type:

```
MA User name[Fixture]> Store /CreateSecondCue
```




8.4.12. /CueOnly Option Keyword

To enter the **/CueOnly** option keyword in the command line, use one of the options:

- Type **/CueOnly**
- Type the shortcuts **/Co** or **/Cu**

Description

The **/CueOnly** option keyword blocks tracked values in the next cue or cue part to preserve the previous look on stage.

For more information see [Store Cues](#).

Syntax

[Function] (Sequence ["Sequence_Name" or Sequence_Number]) Cue ["Cue_Name" or Cue_Number] /CueOnly


General Keywords

General keywords that use the **/CueOnly** option keyword:

- [Cue keyword](#)
- [Store keyword](#)
- [Update keyword](#)

Example

- To store the current programmer values in cue 8 and to preserve the previous look in the following cue after cue 8, type:

```
 User name[Fixture]> Store Cue 8 /CueOnly
```



8.4.13. /Date Option Keyword

To enter the **/Date** option keyword in the command line, use one of the options:

- Type **/Date**
- Type the shortcut **/Da**

Description

The **/Date** option keyword changes the start date of an agenda event that is to be pasted from clipboard. All other settings of this agenda event will not be changed and stay the same.

Syntax

Paste Agenda [Agenda_Number] /Date "DD.MM.YYYY"

General Keywords

General keywords that use the **/Date** option keyword:

- **Agenda keyword**
- **Paste keyword**

Example

Requirement:

- An existing agenda event was copied to the clipboard.
- To copy agenda event 1 to the clipboard, type:

```
MA User name[Fixture]> Copy Agenda 1
```

- To paste agenda event 1 back from the clipboard to create an agenda event 2 with a new start date 11.11.2011 , type:

```
MA User name[Fixture]> Paste Agenda 2/Date "11.11.2011"
```



8.4.14. /Default Option Keyword

To enter the **/Default** option keyword in the command line, use one of the options:

- Type **/Default**
- Type the shortcut **/D**

Description

The **/Default** option keyword is used when copying cues. It replaces non-existing data in the destination cue by default values.

Syntax

Copy Cue ["Cue_Name" or Cue_Number] /Default

General Keywords

General keywords that use the **/Default** option keyword:

- **Copy keyword**
- **Cue keyword**

Example

- To copy cue 1 to cue 5 using default values, type:

```
MA User name[Fixture]> Copy Cue 1 At Cue 5 /Default
```



8.4.15. /DiscardChanges Option Keyword

To enter the **/DiscardChanges** option keyword in the command line, use one of the options:

- Type **/DiscardChanges**
- Type the shortcut **/DC** or **/Di**

Description

The **/DiscardChanges** option keyword is used to leave the Patch without storing the changes.

Syntax

ChangeDestination [Object] ("Object_Name" or Object_Number) /DiscardChanges

General Keywords

General keywords that use the **/DiscardChanges** option keyword:

- **ChangeDestination keyword**

Example

Requirement:

- You are in the Patch and you made edits.
- To leave the patch without storing your recent edits in Stage1, type:

```
MA User name@ShowData/Patch/Stages/Stage1/Fixtures> ChangeDestination  
Root /DiscardChanges
```



8.4.16. /DMX Option Keyword

To enter the **/DMX** option keyword in the command line, use one of the options:

- Type **/DMX**
- Type the shortcut **/DM**

Description

The **/DMX** option keyword is used to store incoming DMX data to cues or presets.

For more information on data sources see [Store Settings and Preferences](#).

Syntax

Store [Object] ["Object_Name" or Object_Number] /DMX [/AllForSelected or /All]

General Keywords

General keywords that use the **/DMX** option keyword:

- [Cue keyword](#)
- [Preset keyword](#)
- [Sequence keyword](#)

Example

- To store the incoming DMX data to sequence 1 cue 42 in all attributes of the selected fixtures, type:

```
MA User name[Fixture]> Store Sequence 1 Cue 42 /DMX /AllForSelected
```



8.4.17. /DMXProtocols Option Keyword

To enter the **/DMXProtocols** option keyword in the command line, use one of the options:

- Type **/DMXProtocols**
- Type the shortcut **/DMXP**

Description

The **/DMXProtocols** option keyword is used to load the DMX protocol settings (Art-Net and sACN) of the show file.

Syntax

[Function] ["Show_Name"] /DMXProtocols

General Keywords

General keywords that use the **/DMXProtocols** option keyword:

- **LoadShow**
- **NewShow**

Examples

- To load the DMX protocols of the show file "A Midsummer Night's Dream" and its show data, type:

```
MA User name[Fixture]> LoadShow "A Midsummer Night's Dream" /DMXProtocols
```

- To create a new show and clear all DMX protocols, type:

```
MA User name[Fixture]> NewShow "Phobos" /DMXProtocols
```



8.4.18. /Embed Option Keyword

To enter the **/Embed** option keyword in the command line, use one of the options:

- Type **/Embed**
- Type the shortcut **/E** or **/EB**

Description

The **/Embed** option keyword is used to store presets that are active in the programmer into a preset. The values in the new preset reference the preset that was active in the programmer.

Syntax

Store Preset ["FeatureGroup_Name" or FeatureGroup_Number],["Preset_Name" or Preset_Number] /Embed

General Keywords

General keywords that use the **/Embed** option keyword:

- Preset keyword
- Store keyword

Example

Requirement:

- Presets are activated in the programmer.
- To store the preset values that are currently active in the programmer together with the reference of these values into the second dimmer preset, type:

```
MA User name[Fixture]> Store Preset 1.2 /Embed
```



8.4.19. /Enumerate Option Keyword

To enter the **/Enumerate** option keyword in the command line, use one of the options:

- Type **/Enumerate**
- Type the shortcut **/En**

Description

The **/Enumerate** option keyword is used to add a count to the name of a show file.



Important:

The number is a three-digit number. If the name of the show file along with the enumeration reaches the limit of 31 characters, the excess characters will be cut and replaced by the enumeration instead.

Syntax

SaveShow "Show_Name" /Enumerate

General Keywords

General keywords that use the **/Enumerate** option keyword:

- **SaveShow keyword**

Example

- To enumerate a show file type:

```
MA User name[Fixture]> SaveShow /Enumerate
```




8.4.20. /File Option Keyword

To enter the **/File** option keyword in the command line, use one of the options:

- Type **/File**
- Type the shortcut **/Fi**

Description

The **/File** option keyword is used to specify the file name of objects during import or export.

Syntax

[Function] [Object] ["Object_Name" or Object_Number] (If Drive [Drive_Number]) /File ["File_Name"]

General Keywords

General keywords that use the **/File** option keyword:

- **Export keyword**
- **Import keyword**

Example

- To export the preset file Endor, type:

```
 User name[Fixture]> Export Preset 2.5 /File "Endor"
```



8.4.21. /ForceGlobal Option Keyword

To enter the **/ForceGlobal** option keyword in the command line, use one of the options:

- Type **/ForceGlobal**
- Type the shortcut **/ForceG**

Description

The **/ForceGlobal** option keyword adds global data to a preset already containing selective data. Selective data which is not used will be removed in the preset of fixtures of the same fixture type. **/ForceGlobal** removes selective data when updating or storing using the merge option.

Syntax

[Function] Preset ["FeatureGroup_Name" or FeatureGroup_Number].["Preset_Name" or Preset_Number] /ForceGlobal

General Keywords

General keywords that use the **/ForceGlobal** option keyword:

- **Preset keyword**
- **Store keyword**
- **Update keyword**

Example

- To replace the selective data by global data in all fixtures of the same fixture type in the second dimmer preset, type:

```
MA User name[Fixture]> Store Preset 1.2 /ForceGlobal
```



8.4.22. /GDTF Option Keyword

To enter the **/GDTF** option keyword in the command line, use one of the options:

- Type **/GDTF**
- Type the shortcut **/GD**

Description

The **/GDTF** option keyword is used during export of a fixture type that uses the GDTF format.

Syntax

Export FixtureType ["FixtureType_Name" or FixtureType_Number] "filename.gdtf" /GDTF

General Keywords


General keywords that use the **/GDTF** option keyword:

- **Export keyword**

Example

Requirement: Choose a drive you would like to export your fixture to.

- To export a Mac Aura XB to the currently selected drive, type:

```
 User name[Fixture]> Export FixtureType "Mac Aura XB" "filename.gdtf" /GDTF
```



8.4.23. /Gaps Option Keyword

To enter the **/Gaps** option keyword in the command line, use one of the options:

- Type **/Gaps**
- Type the shortcut **/Ga**

Description

The **/Gaps** option keyword retains or suppresses empty spaces when importing or exporting a range of pool objects.



Hint:

/Gaps or **/Gaps "Yes"** retains empty spaces when importing or exporting a range of pool objects.
/Gaps "No" suppresses empty spaces when importing or exporting a range of pool objects.

Syntax

[Function] [Object] ["Object_Name" or Object_Number] /Gaps ("Yes" or "No")

General Keywords

General keywords that use the **/Gaps** option keyword:

- **Import keyword**
- **Export keyword**

Example

- To import all macros from the library "mymacros.xml," starting with macro 11, while suppressing any empty spaces included in the library, type:

```
MA User name[Fixture]> Import Macro Library "mymacros.xml" At Macro 11 /Gaps  
"No"
```



8.4.24. /Global Option Keyword

To enter the **/Global** option keyword in the command line, use one of the options:

- Type **/Global**
- Type the shortcut **/G**

Description

The **/Global** option keyword is used to store or update global data into presets regardless their mode.

Syntax

[Function] Preset ["FeatureGroup_Name" or FeatureGroup_Number].["Preset_Name" or Preset_Number] /Global

General Keywords

General keywords that use the **/Global** option keyword:

- **Preset keyword**
- **Store keyword**
- **Update keyword**

Example

- To store the current programmer content as global data in the second dimmer preset, type:

```
MA User name[Fixture]> Store Preset 1.2 /Global
```



8.4.25. /GridMergeMode Option Keyword

To enter the **/GridMergeMode** option keyword in the command line, use one of the options:

- Type **/GridMergeMode**
- Type the shortcut **/Gr** or **GMM**

Description

The **/GridMergeMode** option keyword is used when storing fixtures with their selection grid data into objects using the merge mode.

For more information see [Store Settings and Preferences](#).

Syntax

Store [Object] ["Object_Name" or Object_Number] /Merge /GridMergeMode "Value"

General Keywords

General keywords that use the **/GridMergeMode** option keyword:

- Cue keyword
- Group keyword
- Preset keyword
- World keyword

Values

The **/GridMergeMode** option keyword uses these values:

- AppendX
- Off

Example

- To merge the currently selected fixtures in group 27 and append the new fixtures in the selection grid on the x-axis following the fixtures that were already stored, type:

```
MA User name[Fixture]> Store Group 27 /GridMergeMode "AppendX"
```

- To merge the currently selected fixtures in group 28 and place them on their original position in the selection grid, type:

```
MA User name[Fixture]> Store Group 28 /GridMergeMode "Off"
```





8.4.26. /Indirect Option Keyword

To enter the **/Indirect** option keyword in the command line, use one of the options:

- Type **/Indirect**
- Type the shortcut **/Ind**

Description



Hint:

The /Indirect Option keyword can be used with all object types to which you can assign other objects.

The /Indirect option keyword is used to directly edit an assigned object. Executing a command containing this option keyword opens the editor for the object to be modified.

Syntax

Edit [Object] ["Object_Name" or Object_Number] Property ["Property_Name"] /Indirect

General Keywords

General keywords that use the /Indirect option keyword:

- **Edit keyword**
- **Property keyword**

Examples

- To open the appearance editor and edit the appearance that is assigned to group 8, type:

```
MA User name[Fixture]> Edit Group 8 Property "Appearance" /Indirect
```

- To edit the object that is triggered by the first agenda entry, type:

```
MA User name[Fixture]> Edit Agenda 1 Property "Object" /Indirect
```




8.4.27. /InputFilter Option Keyword

To enter the **/InputFilter** option keyword in the command line, use one of the options:

- Type **/InputFilter**
- Type the shortcut **/I**

Description

The **/InputFilter** option keyword is used to enable or disable the filter of a feature group when storing or updating presets.

Syntax

[Function] Preset ["FeatureGroup_Name" or FeatureGroup_Number].["Preset_Name" or Preset_Number] /InputFilter ["Option_Value"]

General Keywords

General keywords that use the **/InputFilter** option keyword:

- **Preset keyword**
- **Store keyword**
- **Update keyword**

Example

- To disable the input filter when storing the third dimmer preset, type:

```
 User name[Fixture]> Store Preset 1.3 /InputFilter "No"
```



8.4.28. /KeepActivation Option Keyword

To enter the **/KeepActivation** option keyword in the command line, use one of the options:

- Type **/KeepActivation**
- Type the shortcut **/KA** or **/K**

Description

The **/KeepActivation** option keyword is used to keep the programmer content still active after storing the preset.

Syntax

Store Preset ["FeatureGroup_Name" or FeatureGroup_Number].["Preset_Name" or Preset_Number] /KeepActivation

General Keywords

General keywords that use the **/KeepActivation** option keyword:

- **Preset keyword**
- **Store keyword**

Example

- To store the current programmer content in the third dimmer preset and keep this data active in the programmer, type:

```
MA User name[Fixture]> Store Preset 1.3 /KeepActivation
```



8.4.29. /LocalSettings Option Keyword

To enter the **/LocalSettings** option keyword in the command line, use one of the options:

- Type **/LocalSettings**
- Type the shortcut **/Loc**

Description

The **/LocalSettings** option keyword is used to load the local settings of a show file.

Syntax

[Function] ["Show_Name"] /LocalSettings

General Keywords

General keywords that use the **/LocalSettings** option keyword:

- **LoadShow**
- **NewShow**

Example

- To load local settings together with the show data of the show file "A Midsummer Night's Dream", type:

```
MA User name[Fixture]> LoadShow "A Midsummer Night's Dream" /LocalSettings
```



8.4.30. /Look Option Keyword

To enter the **/Look** option keyword in the command line, use one of the options:

- Type **/Look**
- Type the shortcut **/Loo**

Description

The **/Look** option keyword stores all dimmer values of all fixtures in the show.

If the dimmer value of a fixture is zero, the **/Look** option keyword only stores dimmer values of the fixture as there is no visible output onstage.

If the dimmer value of a fixture is above zero, the **/Look** option keyword stores the values of all attributes of the fixture.

Syntax

[Function] [Object] ["Object_Name" or Object_Number] /Look

General Keywords

General keywords that use the **/Look** option keyword:

- **Cue keyword**
- **Preset keyword**
- **Store keyword**



Hint:

The **/Look** option keyword can be used together with the **/All option keyword** or the **/AllForSelected** option keyword.

Examples

- To store the dimmer values of cue 1, type:

```
MA User name[Fixture]> Store Cue 1 /Look
```

- To store the dimmer values and all active attributes of the selected fixtures in the programmer to the second preset in the first All preset pool, type:

```
MA User name[Fixture]> Store Preset 21.2 /AllForSelected /Look
```





8.4.31. /MAtricks Option Keyword

To enter the **/MAtricks** option keyword in the command line, use one of the options:

- Type **/MAtricks**
- Type the shortcut **/MA**

Description

The **/MAtricks** option keyword is used to store the MAtricks settings in the preset whenever the MAtricks are active.

Syntax

Store Preset ["FeatureGroup_Name" or FeatureGroup_Number],["Preset_Name" or Preset_Number] /MAtricks

General Keywords

General keywords that use the **/MAtricks** option keyword:

- Preset keyword
- Store keyword

Example

- To store the currently active MAtricks settings in the third dimmer preset, type:

```
 User name[Fixture]> Store Preset 1.3 /MAtricks
```



8.4.32. /Merge Option Keyword

To enter the **/Merge** option keyword in the command line, use one of these options:

- Type **/Merge**
- Type the shortcut **/m**

Description

The **/Merge** option keyword merges new values with existing values.

The most recent values have a higher priority than the preexisting and will effectively overwrite these.

Syntax

[Function] [Object] ["Object_Name" or Object_Number] /Merge

General Keywords

General keywords that use the **/Merge** option keyword:

- **Assign keyword**
- **AutoCreate keyword**
- **Cook keyword**
- **Copy keyword**
- **Cue keyword**
- **Group keyword**
- **Layout keyword**
- **MAtricks keyword**
- **Preset keyword**
- **Store keyword**
- **World keyword**

Example

- To merge the values of cue 1, type:

```
 User name[Fixture]> Store Cue 1 /Merge
```



8.4.33. /MergeHighPriority Option Keyword

To enter the **/MergeHighPriority** option keyword in the command line, use one of these options:

- Type **/MergeHighPriority**
- Type the shortcut **/MergeH**

Description

The **/MergeHighPriority** option keyword adds all data that is in the source to destination. Data originating in the source overwrite the entire data at destination.

Syntax

[Function] [Source_Object] ["Object_Name" or Object_Number] At [Destination_Object] ["Object_Name" or Object_Number] /MergeHighPriority

General Keywords

General keywords that use the **/MergeHighPriority** option keyword:

- **Clone keyword**

Example

- To clone fixture 1 to fixture 2 overwriting the entire content of fixture 2 in the programmer, type:

```
MA User name[Fixture]> Clone Fixture 1 At Fixture 2 /MergeHighPriority
```




8.4.34. /MergeLowPriority Option Keyword

To enter the **/MergeLowPriority** option keyword in the command line, use one of these options:

- Type **/MergeLowPriority**
- Type the shortcut **/MergeL**

Description

The **/MergeLowPriority** option keyword adds data that is in the source to the destination only where it does not contain data. This option keyword preserves existing data at destination and is the least extreme form of merging.

Syntax

[Function] [Object] ["Object_Name" or Object_Number] /MergeLowPriority

General Keywords

General keywords that use the **/MergeLowPriority** option keyword:

- **Clone keyword**

Example

- To clone data from fixture 1 to fixture 2 as a low priority merge in the programmer, type:

```
MA User name[Fixture]> Clone Fixture 1 At Fixture 2 /MergeLowPriority
```



8.4.35. /MoveValues Option Keyword

To enter the **/MoveValues** option keyword in the command line, use one of the options:

- Type **/MoveValues**
- Type the shortcut **/MV** or **/Mo**

Description

The **/MoveValues** option keyword is used to move values to part 0 when deleting cue parts.

Syntax

Delete Cue ["Cue_Name" or Cue_Number] Part ["Part_Name" or Part_Number] /MoveValues

General Keywords

General keywords that use the **/MoveValues** option keyword:

- **Cue keyword**
- **Delete keyword**
- **Part keyword**

Example

- To delete part 5 of cue 7, and move the values to part 0 of cue 7, type:

```
MA User name[Fixture]> Delete Cue 7 Part 5 /MoveValues
```



8.4.36. /NoAutoClose Option Keyword

To enter the **/NoAutoClose** option keyword in the command line, use one of the options:

- Type **/NoAutoClose**
- Type the shortcuts **/NoA**

Description

The **/NoAutoClose** option keyword is used to suppress the countdown in pop-ups.

Syntax

[Function] ("Object_Name" or Object_Number) /NoAutoClose


General Keywords

General keywords that use the **/NoAutoClose** option keyword:

- **ShutDown keyword**

Example

- To shutdown the station you are working with at the moment suppressing the countdown in the shutdown pop-up, type:

```
 User name[Fixture]> Shutdown /NoAutoClose
```



8.4.37. /NoConfirmation Option Keyword

To enter the **/NoConfirmation** option keyword in the command line, use one of the options:

- Type **/NoConfirmation**
- Type the shortcuts **/N** or **/NC**

Description

The **/NoConfirmation** option keyword is used to suppress the confirmation pop-ups when storing objects or shutting down the station.

Syntax

[Function] ["Object_Name" or Object_Number] /NoConfirmation

General Keywords

General keywords that use the **/NoConfirmation** option keyword:

- **Delete keyword**
- **Oops keyword**
- **Reboot keyword**
- **Restart keyword**
- **ShutDown keyword**
- **Store keyword**

Examples

- To shut down the station you are working with without provoking a confirmation pop-up, type:

```
MA User name[Fixture]> Shutdown /NoConfirmation
```

- To store the programmer values as cue 2 without displaying the confirmation pop-up, type:

```
MA User name[Fixture]> Store Cue 2 /NoConfirmation
```



8.4.38. /NoDependencies Option Keyword

To enter the **/NoDependencies** option keyword in the command line, use one of the options:

- Type **/NoDependencies**
- Type the shortcuts **/NoD**

Description

The **/NoDependencies** option keyword is used to import or export data without the dependent objects.

Syntax

[Function] ["Object_Name" or Object_Number] /NoDependencies

General Keywords

General keywords that use the **/NoDependencies** option keyword:

- **Clone keyword**
- **Export keyword**
- **Import keyword**

Examples

- To import the sequence in the file "Mimas.xml" to sequence 4 without dependent objects, type:

```
MA User name[Fixture]> Import Sequence Library "Mimas.xml" At Sequence 4  
/NoDependencies
```

Requirement: Fixture 1, which uses a selective preset, is stored in a cue of sequence 1. Only fixture 1 is part of this preset.

- To clone the data of fixture 1 to fixture 2 and use hard values in sequence 1 in fixture 2 where fixture 1 uses the selective preset, type:

```
MA User name[Fixture]> Clone Fixture 1 At Fixture 2 If Sequence 1  
/NoDependencies
```



8.4.39. /NoOops Option Keyword

To enter the **/NoOops** option keyword in the command line, use one of the options:

- Type **/NoOops**
- Type the shortcut **/NU**

Description

The **/NoOops** option keyword does not generate oops events when executing commands.

Syntax


[Command] /NoOops

General Keywords

Almost all of the keywords in the grandMA3 software can be combined with the **/NoOops** option keyword.

Example

- To store a new dimmer preset without creating an oops event for this action, type:

```
 User name[Fixture]> Store Preset 1.4 /NoOops
```

8.4.40. /NoRefresh Option Keyword

To enter the **/NoRefresh** option keyword in the command line, use one of the options:

- Type **/NoRefresh**
- Type the shortcut **/NoR**

Description

The **/NoRefresh** option keyword is used to suppress the refresh of libraries during import or export of objects.



Hint:

When listing the library files of a certain type and/or path, it may take a while to type the path and the options into the command line. When a file is to be imported after a type and/or path has been specified during a previous command, the type and/or path will typically need to be entered again as part of the import command. Using the **/NoRefresh** option, it is not necessary to reenter **/Type** and/or **/Path** as part of the import command.

Syntax

[Function] Object ["Object_Name" or Object_Number] (If Drive [Drive_Number]) (/Option) ("/Option_Value")

General Keywords

General keywords that use the **/NoRefresh** option keyword:

- **Export keyword**
- **Import keyword**

Example

To use the **/NoRefresh** option to avoid reentering drive and path specifications:

- List all macro libraries within a specific path on a specific drive:

```
MA User name[Fixture]> List Library If Drive 2 /Path  
"/My_grandMA3_files/macro/archive"
```

- To import the second library from the list without reentering the drive and path, type:

```
MA User name[Fixture]> Import Library 2 /NoRefresh
```

8.4.41. /NoSave Option Keyword

To enter the **/NoSave** option keyword in the command line, use one of the options:

- Type **/NoSave**
- Type the shortcut **/NS**

Description

The **/NoSave** option keyword defines for the show file not to be saved when loading a show, or shutting down, restarting, or rebooting the console or your onPC.



Important:

The **/NoSave** option keyword only works in the conjunction with the **/NoConfirmation option keyword**.

Syntax

[Function] ([Object] ["Object_Name" or Object_Number]) /NoConfirmation /NoSave


General Keywords

General keywords that use the **/NoSave** option keyword:

- **LoadShow keyword**
- **Reboot keyword**
- **Restart keyword**
- **ShutDown keyword**

Examples

- To shut down the grandMA3 console without saving the show file, type:

```
 User name[Fixture]> ShutDown /NoConfirmation /NoSave
```




8.4.42. /NoShowData Option Keyword

To enter the **/NoShowData** option keyword in the command line, use one of the options:

- Type **/NoShowData**
- Type the shortcut **/NoSh**

Description

The **/NoShowData** option keyword is used to keep current show data when loading shows.

Syntax

[Function] ["Show_Name"] /NoShowData

General Keywords

General keywords that use the **/NoShowData** option keyword:

- **LoadShow**
- **NewShow**

Example

- To load all data, except the show data of the show file "A Midsummer Night's Dream" and keep the current show data, type:

```
MA User name[Fixture]> LoadShow "A Midsummer Night's Dream" /NoShowData
```

8.4.43. /NoSubfolders Option Keyword

To enter the **/NoSubfolders** option keyword in the command line, use one of the options:

- Type **/NoSubfolders**
- Type the shortcut **/NoS**

Description

The **/NoSubfolders** option keyword is used to exclude subfolders when importing library files or listing libraries.



Hint:

Subfolders are created by the users.

Syntax

[Function] [Object] ["Object_Name" or Object_Number] /NoSubfolders

General Keywords

General keywords that use the **/NoSubfolders** option keyword:

- **List keyword**
- **Import keyword**

Examples

- To only show color themes in the main folders when listing, type:

```
MA User name[Fixture]> List ColorTheme /NoSubfolders
```

It might as well happen that there are two files with the same name located in the main folder and a subfolder.

- To import the macro library file "bestmacro.xml" located in the main folder, type:

```
MA User name[Fixture]> Import Macro Library "bestmacro.xml" At Macro 5 If Drive  
2 /NoSubfolders
```



8.4.44. /OriginalContentOnly Option Keyword

To enter the **/OriginalContentOnly** option keyword in the command line, use one of the options:

- Type **/OriginalContentOnly**
- Type the shortcut **/Or**

Description

The **/OriginalContentOnly** option keyword updates already existing values of attributes in their original location in presets or cues.

Syntax

Update [Object] ["Object_Name" or Object_Number] /OriginalContentOnly

General Keywords

General keywords that use the **/OriginalContentOnly** option keyword:

- **Update keyword**

Example

Requirement:

- Create presets or cues
 - Playback presets or cues or activate presets in the programmer
 - The different attributes must have active values in the programmer
-
- To only update the existing values in Preset 1.1, type:

```
MA User name[Fixture]> Update Preset 1.1 /OriginalContentOnly
```



8.4.45. /Output Option Keyword

To enter the **/Output** option keyword in the command line, use one of the options:

- Type **/Output**
- Type the shortcut **/Ou**

Description

The **/Output** option keyword is used to store incoming output data to cues or presets.

For more information on data sources see [Store Settings and Preferences](#).

Syntax

Store [Object] ["Object_Name" or Object_Number] /Output [AllForSelected or /All]

General Keywords

General keywords that use the **/Output** option keyword:

- [Cue keyword](#)
- [Preset keyword](#)
- [Sequence keyword](#)

Example

- To store the incoming output data to sequence 1 cue 42 in all attributes of the selected fixtures, type:

```
MA User name[Fixture]> Store Sequence 1 Cue 42 /Output/AllForSelected
```



8.4.46. /OutputStations Option Keyword

To enter the **/OutputStations** option keyword in the command line, use one of the options:

- Type **/OutputStations**
- Type the shortcut **/OutputS**

Description

The **/OutputStations** option keyword is used to overwrite the current output stations with the output stations of the show file that will be loaded.

Syntax

[Function] ["Show_Name"] /OutputStations

General Keywords

General keywords that use the **/OutputStations** option keyword:

- **LoadShow**
- **NewShow**

Example

- To load the output stations of the show file "A Midsummer Night's Dream" together with its show data, type:

```
MA User name[Fixture]> LoadShow "A Midsummer Night's Dream" /OutputStations
```



8.4.47. /Overwrite Option Keyword

To enter the **/Overwrite** option keyword in the command line, use one of the options:

- Type **/Overwrite**
- Type the shortcut **/O**

Description

The **/Overwrite** option keyword is used to overwrite existing values.

Syntax

[Function] [Object] ["Object_Name" or Object_Number] /Overwrite

General Keywords

General keywords that use the **/Overwrite** option keyword:

- **Cook keyword**
- **Clone keyword**
- **Cue keyword**
- **Filter keyword**
- **Group keyword**
- **MAtricks keyword**
- **Preset keyword**
- **Sequence keyword**
- **Store keyword**
- **World keyword**

Example

- To overwrite the existing values of cue 5 in sequence 1, type:

```
MA User name[Fixture]> Store Sequence 1 Cue 5 /Overwrite
```

- To clone the programmer data of fixture 1 to fixture 2 and overwrite all the data of fixture 2 in the programmer, type:

```
MA User name[Fixture]> Clone Fixture 1 At Fixture 2 /Overwrite
```



8.4.48. /PatchOffset Option Keyword

To enter the **/PatchOffset** option keyword in the command line, use one of the options:

- Type **/PatchOffset**
- Type the shortcut **/Patc**

Description

When patching the **/PatchOffset** option keyword sets the offset of DMX addresses in fixtures. The DMX address has to be specified using breaks of the fixture type.

Syntax

Set (Object) [Object_Number] Property "Break[Number]" ["DMXAddress"] /PatchOffset [PatchOffset_Value]

General Keywords

General keywords that use the **/PatchOffset** option keyword:

- **Set keyword**

Example

Requirement:

- Go to the Live Patch

For information on the Live Patch and how to use it in the grandMA3, see **Live Patch**.

- To set an offset of 50 starting at DMX address 10.1 in the first 13 fixtures in the patch, type:

```
 User name[Fixture]> Set 2 Thru 14 Property "Break1" "10.1" /PatchOffset 50
```

8.4.49. /Path Option Keyword

To enter the **/Path** option keyword in the command line, use one of the options:

- Type **/Path**
- Type the shortcut **/Pa**

Description

The **/Path** option keyword defines the folder path where an imported or exported file is saved.



Hint:

- Enter a path beginning with a letter or number if the path is incorporated with the default folder structure.
- Enter a path beginning with the forward-slash (/) character if the path begins at the root of the device.
- Entering a path that does not already exist creates the necessary folders.

Syntax

[Function] [Object] ["Object_Name" or Object_Number] (If Drive [Drive_Number]) /Path "The/Path/To/My/Files"

General Keywords

General keywords that use the **/Path** option keyword:

- **Export keyword**
- **Import keyword**

Example

- To export macro 1, labeled "test," to a folder labeled "myfavorites" at the root of the first connected USB drive, type:

```
 User name[Fixture]> Export Macro 1 "test" If Drive 2 /Path "/myfavorites"
```




8.4.50. /PhaserData Option Keyword

To enter the **/PhaserData** option keyword in the command line, use one of the options:

- Type **/PhaserData**
- Type the shortcut **/Ph**

Description

The **/PhaserData** option keyword is used to define if attribute data will be stored in presets.

Syntax

Store Preset ["FeatureGroup_Name" or FeatureGroup_Number].["Preset_Name" or Preset_Number] /PhaserData ["Option_Value"]

General Keywords

General keywords that use the **/PhaserData** option keyword:

- **Preset keyword**
- **Store keyword**

Example

Requirement:

1. Select 10 fixtures.
2. Set XWings to 2.
3. Type in the command line:

```
MA User name[Fixture]> At 0 Thru 100
```

- To store all data, except the active programmer data, into the first preset of the third All preset pool, type:

```
MA User name[Fixture]> Store Preset 23.1 /PhaserData "No"
```



8.4.51. /Programmer Option Keyword

To enter the **/Programmer** option keyword in the command line, use one of the options:

- Type **/Programmer**
- Type the shortcut **/P**

Description

The **/Programmer** option keyword is used to store active programmer data to cues or presets.

For more information on data sources see [Store Settings and Preferences](#).

Syntax

Store [Object] ["Object_Name" or Object_Number] /Programmer

General Keywords

General keywords that use the **/Programmer** option keyword:

- **Cue keyword**
- **Preset keyword**
- **Sequence keyword**

Example

- To store the programmer data to sequence 1 cue 42, type:

```
MA User name[Fixture]> Store Sequence 1 Cue 42 /Programmer
```



8.4.52. /Release Option Keyword

To enter the **/Release** option keyword in the command line, use one of the options:

- Type **/Release**
- Type the shortcut **/Rel**

Description

The **/Release** option keyword is used when copying cues. It replaces non-existing data in the destination cue by released values.

Syntax

Copy Cue ["Cue_Name" or Cue_Number] /Release

General Keywords

General keywords that use the **/Release** option keyword:

- **Copy keyword**
- **Cue keyword**

Example

- To copy cue 1 to cue 2 using released values, type:

```
MA User name[Fixture]> Copy Cue 1 At Cue 2 /Release
```



8.4.53. /Remove Option Keyword

To enter the **/Remove** option keyword in the command line, use one of the options:

- Type **/Remove**
- Type the shortcut **/R**

Description

The **/Remove** option keyword is used to remove values that are stored in attributes and active programmer values.

Syntax

[Function] ["Object_Name" or Object_Number] /Remove

General Keywords

General keywords that use the **/Remove** option keyword:

- **Assign keyword**
- **Cook keyword**
- **Cue keyword**
- **Group keyword**
- **Layout keyword**
- **MAtricks keyword**
- **Preset keyword**
- **Store keyword**
- **Update keyword**
- **World keyword**

Examples

- To remove the values that are active in the programmer and that are stored in cue 5 of sequence 1, type:

```
MA User name[Fixture]> Store Sequence 1 Cue 5 /Remove
```

- To remove all cooked data from sequence 1, type:

```
MA User name[Fixture]> Cook Sequence 1 /Remove
```



8.4.54. /Save Option Keyword

To enter the **/Save** option keyword in the command line, use one of the options:

- Type **/Save**
- Type the shortcut **/S**

Description

The **/Save** option keyword defines the show file to be saved when loading a show, shutting down, restarting, or rebooting the console or your onPC.



Important:

The **/Save** option keyword only works in the conjunction with the **/NoConfirmation option keyword**.

Syntax

[Function] ([Object] ["Object_Name" or Object_Number]) /NoConfirmation /Save

General Keywords

General keywords that use the **/Save** option keyword:

- **LoadShow keyword**
- **Reboot keyword**
- **Restart keyword**
- **ShutDown keyword**

Examples

- To shut down the grandMA3 console and saving the current show file suppressing the confirmation pop-up, type:

```
MA User name[Fixture]> ShutDown /NoConfirmation /Save
```



8.4.55. /Screen Option Keyword

To enter the **/Screen** option keyword in the command line, use one of the options:

- Type **/Screen**
- Type the shortcut **/S**

Description

The **/Screen** option keyword addresses screens when storing or calling views.

Syntax

[Function] [Object] ["Object_Name" or Object_Number] /Screen

General Keywords

General keywords that use the **/Screen** option keyword:

- **Call keyword**
- **Store keyword**
- **View keyword**
- **ViewButton keyword**

Example

- To store screen 1 on the view button 1, type:

```
 User name[Fixture]> Store View 1 /Screen "1"
```



8.4.56. /Selection Option Keyword

To enter the **/Selection** option keyword in the command line, use one of the options:

- Type **/Selection**
- Type the shortcut **/Selectio**

Description

The **/Selection** option keyword defines if the current selection will be stored in a recipe.

For more information on how recipes work see [Recipes](#).

Syntax

Store Preset ["FeatureGroup_Name" or FeatureGroup_Number].["Preset_Name" or Preset_Number].["Recipe_Name" or Recipe_Number] /Selection ["Option_Value"]

Store Sequence ["Sequence_Name" or Sequence_Number] Cue ["Cue_Name" or Cue_Number] Part ["Part_Name" or Part_Number].["Recipe_Name" or Recipe_Number] /Selection ["Option_Value"]

General Keywords

General keywords that use the **/Selection** option keyword:

- [Preset keyword](#)
- [Sequence keyword](#)
- [Store keyword](#)

Example

- In order not to store the current selection into the recipe, type:

```
MA User name[Fixture]> Store Preset 3.2.1 /Selection "No"
```



8.4.57. /Selective Option Keyword

To enter the **/Selective** option keyword in the command line, use one of the options:

- Type **/Selective**
- Type the shortcut **/Se**

Description

The **/Selective** option keyword is used to store or update selective data into presets regardless their mode.

Syntax

[Function] Preset ["FeatureGroup_Name" or FeatureGroup_Number].["Preset_Name" or Preset_Number] /Selective

General Keywords

General keywords that use the **/Selective** option keyword:

- **CleanUp keyword**
- **Preset keyword**
- **Store keyword**
- **Update keyword**

Examples

- To store the current programmer content as selective data in the second dimmer preset, type:

```
MA User name[Fixture]> Store Preset 1.2 /Selective
```

- To remove selective data in the first color preset, type:

```
MA User name[Fixture]> Cleanup Preset 4.1 /Selective
```


8.4.58. /Single Option Keyword

To enter the **/Single** option keyword in the command line, use one of the options:

- Type **/Single**
- Type the shortcut **/Single**

Description

The **/Single** option keyword is used when extracting presets or when autocreating single fixture groups.

When extracting embedded presets or phaser presets where presets are integrated in the steps, Extract will call directly the values of the source presets.

Using the **/Single** option keyword together with the Extract keyword makes it possible to extract one level down in the hierarchy of presets.



Important:

The combination of the Extract keyword together with the **/Single** option keyword currently works with presets that are active in the programmer.

Syntax

[Function] [Object] ["Object_Name" or Object_Number] /Single

General Keywords

General keywords that use the **/Single** option keyword:

- **AutoCreate keyword**
- **Extract keyword**

Examples

Extract Example

Requirement:

1. Create 2 color presets.
 2. Embed these color presets into a different color preset:
Embed the second color preset into the All preset 21.1.
 3. Select fixtures and apply the All preset on them.
- To call the second color preset into the programmer, type once:



```
User name[Fixture]> Extract Selection /Single
```

- To call the first color preset into the programmer, type a second time around:

```
MA User name[Fixture]> Extract Selection /Single
```

- To generate activate hard values, type a third time:

```
MA User name[Fixture]> Extract Selection /Single
```

or:

```
MA User name[Fixture]> Extract Selection
```

AutoCreate Example

Requirement:

- Insert at least ten fixture types in the show file and patch at least two fixtures of every fixture type.
- To create single fixture groups starting with group pool object 42 that contain all patched fixtures of the fixture type 9, type:

```
MA User name[Fixture]> AutoCreate FixtureType 9 At Group 42 /Single
```

- To create single fixture groups that contain all patched fixtures of fixture types 9 and 10 starting with group pool object 101, type:

```
MA User name[Fixture]> AutoCreate FixtureType 9 + 10 At Group 101 /Single
```

Requirement:

1. Create layers and classes within the patch.
2. Set fixtures to these layers and classes.



- To create single fixture groups starting with group pool object 201 using all patched fixtures that are set to the layer "Backtruss" within the patch, type:

```
MA User name[Fixture]> AutoCreate FixtureLayer "Backtruss" At Group 201 /Single
```



8.4.59. /Tab Option Keyword

To enter the **/Tab** option keyword in the command line, use one of the options:

- Type **/Tab**
- Type the shortcut **/T**

Description

The **/Tab** option keyword is used to define which tab will open in the assign menu.

Syntax

[Function] Page ["Page_Name" or Page_Number].["Executor_Name" or Executor_Number] /Tab ["Option_Value"]

General Keywords

General keywords that use the **/Tab** option keyword:

- **Assign keyword**
- **Edit keyword**
- **EditSetting keyword**
- **Page keyword**

Values

The **/Tab** option keyword uses these values:

- Edit
- EditSetting
- Handle
- Object

Examples

- To open the edit tab of executor 201 on page 1 using the assign command, type:

```
MA User name[Fixture]> Assign Page 1.201 /Tab "Edit"
```

- To open the assign menu and display the handle tab using the edit command, type:

```
MA User name[Fixture]> Edit Page 1.201 /Tab "Handle"
```



8.4.60. /Type Option Keyword

To enter the **/Type** option keyword in the command line, use one of the options:

- Type **/Type**
- Type the shortcut **/Ty**

Description

The **/Type** option keyword can have different values depending on the keyword it is combined with.

The Import keyword and the Export keyword can be both used with the **/Type** "User" and "System".

The LoadShow keyword can be used with the **/Type** "Demo".

In a nutshell - the Import and the Export keyword both use library files and the LoadShow keyword uses show files.

When importing the desired file may be either a "User" file, which has been previously exported, or a "System" file, which is predefined and included with the system software. If this option is not defined within the import syntax, the console will first search the user library for the specified file name. If the file does not exist within the user library, the console will then search within the system files.

- **/Type "User"** restricts the console to only search within the user library of the selected drive.
- **/Type "System"** restricts the console to only search within the system files, ignoring the user library.
- **/Type "Demo"** restricts the console to only search within demo shows folder.
- **/Type "Template"** restricts the console to only search within the template shows folder.



Important:

When no type is specified, the type "User" has priority.

Syntax

[Function] [Object] ["Object_Name" or Object_Number] (If Drive [Drive_Number]) /Type "Value"

General Keywords

General keywords that use the **/Type** option keyword:

- **Export keyword**
- **Import keyword**
- **LoadShow keyword**



Examples

- To import the save_show macro from the system library instead of the user library to macro 21, type:

```
MA User name[Fixture]> Import Macro Library "save_show.xml" At Macro  
21 /Type "System"
```

- To load the demo show from the demo shows folder, type:

```
MA User name[Fixture]> LoadShow "Demoshow_grandMA3.show" /Type "Demo"
```



8.4.61. /Universal Option Keyword

To enter the **/Universal** option keyword in the command line, use one of the options:

- Type **/Universal**
- Type the shortcut **/U**

Description

The **/Universal** option keyword is used to store or update universal data into presets regardless their mode.

Syntax

[Function] Preset ["FeatureGroup_Name" or FeatureGroup_Number].["Preset_Name" or Preset_Number] /Universal

General Keywords

General keywords that use the **/Universal** option keyword:

- **Preset keyword**
- **Store keyword**
- **Update keyword**

Example

- To store the current programmer content as universal data in the second dimmer preset, type:

```
MA User name[Fixture]> Store Preset 1.2 /Universal
```



8.4.62. /Wrap Option Keyword

To enter the **/Wrap** option keyword in the command line, use one of the options:

- Type **/Wrap**
- Type the shortcut **/W**

Description

The **/Wrap** option keyword loops through all steps and selects the first step after reaching the last step.

This option keyword works in both directions using **Next** and **Previous**.

Syntax

[Function] Step /Wrap

General Keywords

General keywords that use the **/Wrap** option keyword:

- **Next keyword**
- **Previous keyword**
- **Step keyword**

Example

- To loop through all steps in the selected fixtures and subsequently select the first step in the selection, type:

```
MA User name[Fixture]> Next Step /Wrap
```


8.6. Extended Command Line Syntax Options

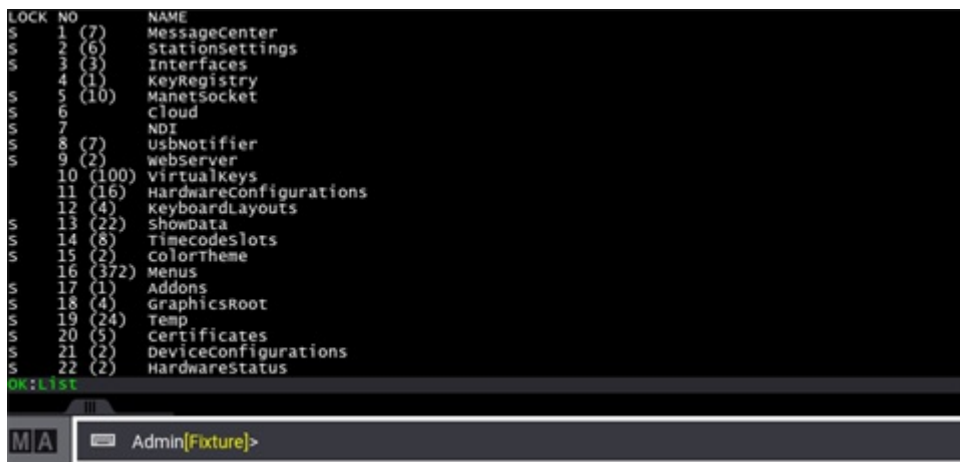
To control the input in the command line use the **List keyword** any time.

For more information about the command line and the Command Line History, read the **command line topic**.

To list all available subfolders, type:



```
MA User name [Fixture]> List
```



```
LOCK NO      NAME
S 1 (7)      MessageCenter
S 2 (6)      StationSettings
S 3 (3)      Interfaces
S 4 (1)      KeyRegistry
S 5 (10)     ManetSocket
S 6          Cloud
S 7          NDI
S 8 (7)      UsbNotifier
S 9 (2)      webserver
S 10 (100)   Virtualkeys
S 11 (16)    Hardwareconfigurations
S 12 (4)     KeyboardLayouts
S 13 (22)    ShowData
S 14 (8)     Timecodeslots
S 15 (2)     colorTheme
S 16 (372)   Menus
S 17 (1)     Addons
S 18 (4)     GraphicsRoot
S 19 (24)    Temp
S 20 (5)     Certificates
S 21 (2)     Deviceconfigurations
S 22 (2)     Hardwarestatus
OK:List
MA Admin[Fixture]>
```

List view in Command Line History



Hint:

When the List command returns multiple objects of different types, the settings for those objects are not displayed. To view the settings of a specific object, include the name or number of the object with the List keyword. For more information, see the **List keyword** topic.

To list all available keywords, type:



```
MA User name [Fixture]> Help
```



```

Command Line History
/ack Option to set the ack mode
/ccd-/CopyCueDst Option to perform cue copy operation
/ccs-/CopyCueSrc Option to perform cue copy operation
/createSecondcue Option to define store mode
/cue-/CueOnly Option to perform cue only operation
/default Option to use defaults
/discardChanges Option discard changes
/embed Option store embedded preset references
/file Option to specify a file name for an import/export command
/gdtf Option to request an export in GDTF format
/global Option to call a preset global
/gridMergeMode Option to perform group store merge operation
/indirect Option to edit indirect
/inputFilter Option to enable preset store input filter
/keepActivation Option keep programmer content active after store
/look Option to define store operation
/merge Option to define store mode
/mergeHighPriority Option to define clone mode
/mergeLowPriority Option to define clone mode
/noAutoClose Option to suppress auto close
/noConfirm Option to suppress confirmation
/noDependencies Option to request an export without dependencies
/noUndo Option to perform an operation without undo
/originalContentOnly Option to define update mode
/originalFixtureOnly Option to define update mode
/overwrite Option to define store mode
/path Option to specify a path for an import/export command
/release Option to define store mode
/remove Option to define store mode
/screen Option to define used screens
/selective Option to call a preset selective
/type Option to specify a type to call
/universal Option to call a preset universally
/wrap Option to wrap around with next previous
List of help keywords:
EndIf Indicates the end of a filter expression
If Defines a filter for an operation
IfActive selects fixtures depending on active output
IfOutput selects fixtures depending on active output
IfProg selects fixtures depending on active output
OK:Help
MA Admin[Fixture]>
    
```

Keyword list in Command Line History

For more information about the help keyword, read the [help keyword topic](#).

Examples

Change the Value of a Sequence Property

The included data is categorized in directories. In order to change a directory, use the command "Change Destination".

- To change the destination to Sequence, type:

```

MA User name [Fixture]> ChangeDestination Sequence
OK:changeDestination Sequence
MA Admin@ShowData/DataPools/Default/Sequences>
    
```



- To display the options in the command line feedback, type:

```
MA User name [Fixture]> List
```

LOCK NO	NAME	SCRIBBLE APPEARANCE	AUTOSTART	AUTOSTOP	AUTOFIX	AUTOSTOMP	RELEASEFIRSTCUE
1 (2)	Default		Yes	Yes	No	Off	Yes
2 (4)	Sequence 2		Yes	Yes	No	Prio	Yes
3 (3)	Sequence 3		Yes	Yes	No	Prio	Yes
4 (3)	Sequence 4		Yes	Yes	No	Prio	Yes
5 (3)	Sequence 5		Yes	Yes	No	Prio	Yes

```
OK:List
```

```
MA Admin@ShowData/DataPools/Default/Sequences>
```

- To change Autostomp from Off to Prio, type:

```
MA User name [Fixture]> Set Sequence 1 Property "Autostomp" "Prio"
```

```
OK:Set Sequence 1 Property "Autostomp" "Prio"
```

```
MA Admin@ShowData/DataPools/Default/Sequences>
```

- To doublecheck the options in the command line feedback, type:

```
MA User name [Fixture]> List
```

LOCK NO	NAME	SCRIBBLE APPEARANCE	AUTOSTART	AUTOSTOP	AUTOFIX	AUTOSTOMP	RELEASEFIRSTCUE
1 (2)	Default		Yes	Yes	No	Prio	Yes
2 (4)	Sequence 2		Yes	Yes	No	Prio	Yes
3 (3)	Sequence 3		Yes	Yes	No	Prio	Yes
4 (3)	Sequence 4		Yes	Yes	No	Prio	Yes
5 (3)	Sequence 5		Yes	Yes	No	Prio	Yes

```
OK:List
```

```
MA Admin@ShowData/DataPools/Default/Sequences>
```

Autostomp is now set to Prio.

- To return to the destination root, type:

```
MA User name [Fixture]> ChangeDestination Root
```

```
OK:changeDestination Root
MA Admin[Fixture]>
```

Store a Cue

Using the command line enables to combine a keyword with different options.

To overwrite cue 5 with the current programmer content and rename the cue to "Great Look" and storing the data as cue only, type:

```
MA User name [Fixture]> Store Cue 5 "Great Look" /CueOnly /Overwrite
OK:store cue 5 "Great Look" /cueOnly "" /overwrite ""
MA Admin[Fixture]>
```



9. Windows, Views, and Menus

Windows are created on the **Screens**. The screens are the monitors. The different sizes of grandMA3 hardware have different amounts of screens. The grandMA3 onPC has quick access to open 5 displays with a large screen area and 2 displays with a small screen area. This offers access to the same primary screens as a grandMA3 full-size. For more information, see [Screen Allocation](#).

There are several possible **Screen Configurations** for each user profile. A user always has a screen configuration assigned. Each screen configuration stores a set of view button assignments and the current window set up on the screens. The screen configurations can be selected in the [User menu](#).

Most screens are empty in a new show. An empty screen is a blank space where each user can create their own arrangement of windows. This arrangement can be stored as a **View**.

On the right side of each screen, there can be a number of **View Buttons**. Views can be assigned to these buttons. This allows for fast access to recall a stored view or update a view by storing it again.

A new show has six factory-made views assigned to the first six view buttons on each of the big screens. The two smaller screens have four and two views assigned. These can be changed or deleted.

A **Menu** is a big pop-up that covers most of the screen. There are several menus in the software that gives access to the setup of the console, system, fixtures, and much more. For more information, see [Menus](#).

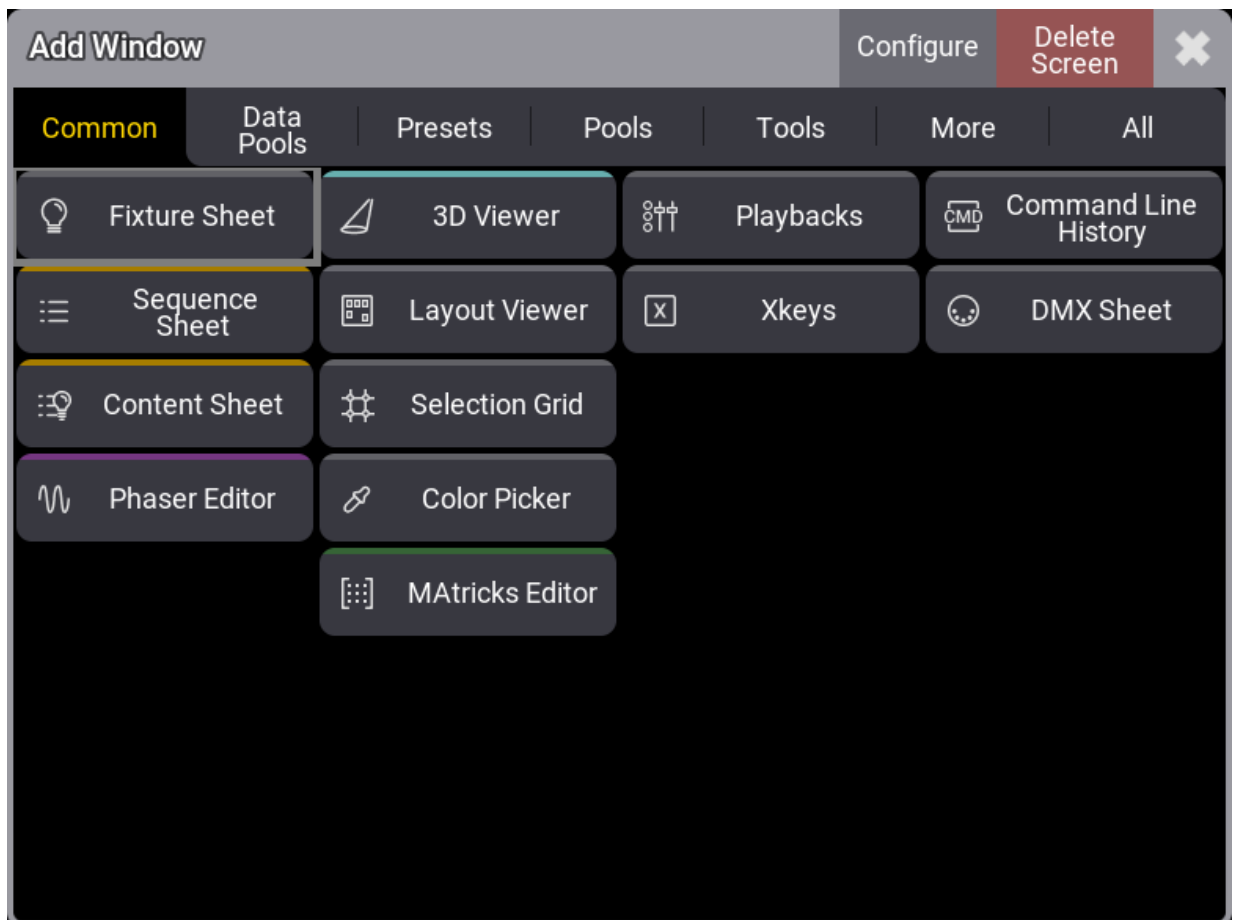
9.1. Add Window

Windows are added to the **user definable areas** on the screens.

The best way to add a window is by tapping inside an empty screen area. The window is added with the windows upper left corner where the screen is tapped.

Another option is to tap and draw a square on the empty screen to define the window size.

Tap the screen to open the **Add Window** pop-up:



Tap a button with the desired window to create it

Windows are divided into different tabs. The tabs are **Common, Data Pools, Presets, Pools, Tools, More, and All**. Tapping the different tabs reveals different buttons with the windows for each section.

Tap the desired button to create the window.

The title bar of the pop-up has two buttons. The red **Delete Screen** button removes all windows from the screen.

The **Configure** button opens the **Configure Display** pop-up. This can be used to customize some settings



regarding tools on the side of the display and change the size of the user area. Read more in the **Configurations of Displays topic**.



List of current windows that can be created using the **Add Window** pop-up:

Common

A fixed selection of commonly used windows. (These windows are also represented in other tabs):

- Fixture Sheet
- Sequence Sheet
- Content Sheet
- Phaser Editor
- 3D Viewer
- Layout Viewer
- Selection Grid
- Color Picker
- MAtricks Editor
- Playbacks
- Xkeys
- Command Line History
- DMX Sheet

Data Pools

All object pools that are part of a data pool:

- **Executor Configurations**
- **Filters**
- **Groups**
- **Layouts**
- **Macros**
- **MAtricks**
- **Pages**
- **Plugins**
- **Quickeys**
- **Sequences**
- **Timecodes**
- **Timers**
- **Worlds**



Presets

All preset pools. ([One link to all about presets](#)):

- Dynamic
- All 1
- All 2
- All 3
- All 4
- All 5
- Dimmer
- Position
- Gobo
- Color
- Beam
- Focus
- Control
- Shapers
- Video

Pools

All pools that are part of the show data or the user profiles:

Show Data:

- [Appearances](#)
- [Data Pools](#)
- [Gels](#)
- [Scribbles](#)
- [Timecode Slots](#)
- [Universes](#)
- [Users](#)

Media:

- [Gobos](#)
- [Images](#)
- [Meshes](#)
- Sounds
- [Symbols](#)
- [Videos](#)



User Profile:

- **Cameras**
- **Render Qualities**
- **Views**

Tools

General windows for displaying show data:

Sheets:

- **Content Sheet**
- **DMX Sheet**
- **Fixture Sheet**
- **Sequence Sheet**

Programmer Tools:

- **At Filter**
- **Color Picker**
- **Programmer Parts**
- **Selection Grid**
- **sMArt**

Viewers and Editors:

- **3D Viewer**
- **Agenda Viewer**
- **Clock Viewer**
- **Layout Viewer**
- **MAtricks Editor**
- **Phaser Editor**
- **RDM Devices Viewer**
- **Sound Viewer**
- **Timecode Viewer**

More

All different bars, playback windows, or info and system-related windows:

Bars:

- **Align Bar**
- **command wing Bar**



- **Encoder Bar**
- **Selection Bar**
- **Step Bar**

Playback:

- **Custom Master Section**
- **Running Playbacks**
- **Playback**
- **Xkeys**

Info and System:

- **Command Line History**
- **Help**
- **Info**
- **Message Center**
- **System Info**
- **System Monitor**
- **Trackpad**

All


The **All** tab contains all windows in an alphabetical list that can be filtered and sorted.

9.2. Rearrange Windows

Tapping a free space on a screen opens the **Add Window** pop-up. The tapped grid field is used as the upper left corner of the created window. From here the window takes all the available space it can, and becomes as big as possible trying to fill all available space on the screen.

Tap, hold and drag on an empty screen draws a square. Releasing the screen opens the **Add Window** pop-up and the selected window fills the drawn square.

The window size can often be made smaller.

In the lower right corner of each window there is a small resize corner: 

Tap or click and holding this corner allows it to be moved. Moving it resizes the window.

Windows can also be resized by tap and hold in the title bar and meanwhile tap another location on the screen. The window will fill the available space.

Windows can also be moved or resized by moving the entire window. Tap or click and hold the title bar while moving it on the screen will show the grid and the window can be dragged and dropped to a new position.

Windows cannot overlap each other, so it can only be as big as the free space allows. If a window is moved up against another window then it will resize the window.

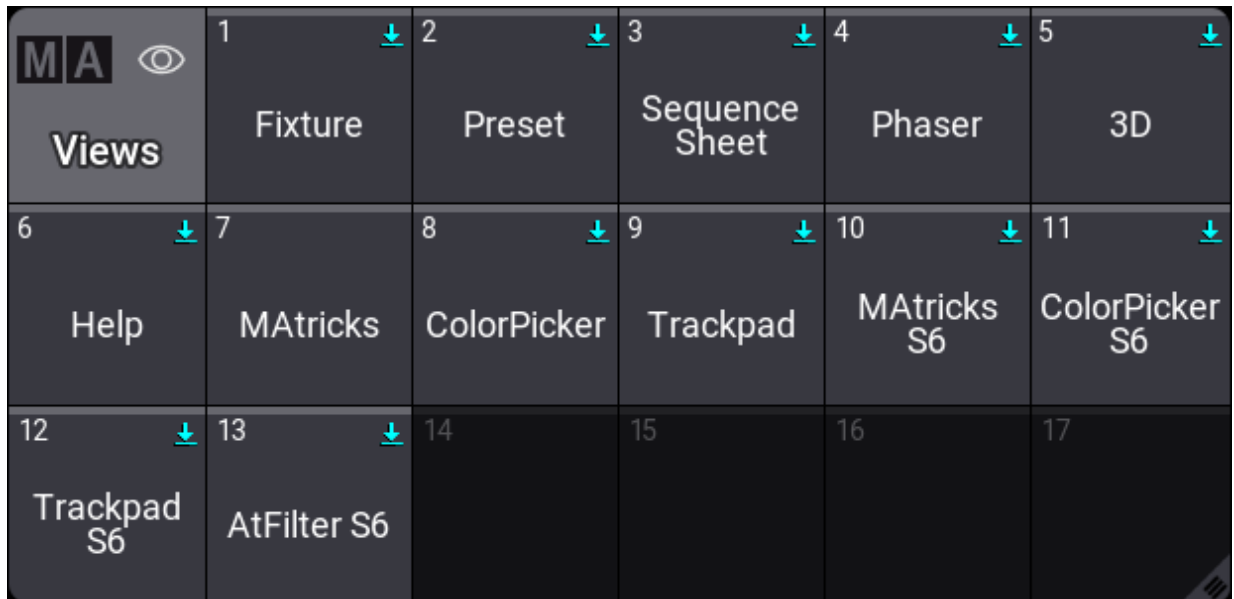
The windows snap to a dotted grid. This grid is used when windows are created, resized, and moved. There are half size grid dots, but pools and preset windows are limited to a full grid size when resized.

Some windows have a minimum size limit. For instance, a Fixture Sheet window needs to be a minimum of 3 x 2 grid fields.



9.3. Store and Recall Views

The arrangement of windows is called a **View**. Views are stored in a **View Pool**. The pool can be created as a window.



View pool window

Views can be assigned to **ViewButtons** and **Executors**.

Store Views

A view is stored using the standard store syntax and the **View keyword**. This means it is possible to use the keys, keys in combination with buttons on the screens, or the command line.

Storing a view opens the **Store View Options** pop-up.



Select what screens to store in a view.

This is used to directly label the view and select which screens should be stored in the view. Each screen has a button that can be turned On or Off. The last touched screen is selected as a default. There are two buttons at the bottom that gives fast access to select **All** or **None** of the screens. Tap **OK** to store the window arrangement of the selected screens or press **Please** to confirm the options.

If the command line is used, the screens can be specified using the **/Screen option keyword**. For more information, please read the **Store keyword** and **View keyword** topics.

Store Using Keys

1. Press **Store**.
2. Press and hold **MA** while pressing **X7 | View**.
This puts the **View keyword** in the command line.
3. Use the numeric keys to type the view number.
4. Execute the command by pressing **Please**.



5. Select the desired screens in the pop-up and press **Please** to confirm the options.

If point 3 is skipped (not adding a number), then the first available view is stored.



Store Using a Combination of Keys and Touch Screens

Requirement:

- A visible **View Pool** on one of the screens.

To store a new view:

1. Press **Store**.
2. Tap an available pool object in the view pool to create a new view or one of the existing pool objects to overwrite the existing view.
3. Select the desired screens in the pop-up and tap **OK** to confirm the options.

The last touched object (the object with the white frame) can be labeled if you start typing on the keyboard. Any of the labeling methods described in the [Label Pool Objects](#) topic can also be used.

Store Using Command Line

Storing using the command line is very simple. The two keywords needed are **Store** and **View**.

For example, storing the window arrangement on screens 1 and 2 as view 15 with the "layout" label:

```
MA User name[Fixture]> Store View 15 "layout" /Screen "1,2"
```

See more in the [View keyword](#) topic.

Store a View Directly on a View Button

A new view can be stored on a **View Button**. This creates a view in the view pool and immediately assigns it to the view button.

1. Press **Store**.
2. Tap a view button.
3. Select the desired screens in the pop-up and tap **OK** to confirm the options.

Now there is a new view in the pool and it is assigned to the tapped view button.

Any of the [described label functions](#) can be used with the view buttons.



Assign Existing View to a View Button

The above method is for creating a new view and having it available on a view button. Existing views can also be assigned to the view buttons.

Again there are three primary ways to do it: Keys, keys and screens, and command line.

Assign View Using Keys

1. Press **Assign**.
2. Press and hold **MA** while pressing **X7 | View** once.
3. Use the numeric keys to type the view number.
4. Press **At**.
5. Press and hold **MA** while pressing **X7 | View** twice (this gives the **ViewButton** keyword).
6. Use the numeric keys to type the view button number.
7. Execute the command by pressing **Please**.

Example:

To assign view 4 at view button 7 on screen 1 the following key presses are needed:

Assign MA + X7 | View 4 At MA + X7 | View MA + X7 | View 1 . 7 Please

This is the command result:

```
MA User name[Fixture]> Assign View 4 At ViewButton 1.7
```

Assign View Using Keys and Screens

Requirement:

- A visible **View Pool** on one of the screens and visible view buttons.

To assign a view:

1. Press **Assign**.
2. Tap the desired view in the view pool.
3. Tap the desired view button.

Assign View Using the Command Line

The **Assign**, **View**, and **ViewButton** keywords are needed for this command.

Assign View [View_ID] At Viewbutton [Display_ID.ViewButton_ID]



Assign Existing Views to an Executor

Existing views can be assigned to executors.

Again there are three primary ways to do it: Keys, keys and screens, and the command line.

Assign View Using Keys

When objects are assigned to executors on a specific page using the keys and command line, then the object needs to be assigned to the child of the page using the **Page** keyword. The executors are children of the page.

If the object needs to be assigned to executors on the active page, it can be assigned to the executor using the **Executor** keyword.

1. Press **Assign**.
2. Press and hold **MA** while pressing **X7 | View** once.
3. Use the numeric keys to type the view number.
4. Press **At**.
5. Press and hold **MA** while pressing **X16 | Exec** once.
6. Use the numeric keys to type the executor number (executor on the active page).
7. Execute the command by pressing **Please**.

Example:

To assign view 9 at executor X1 (executor 291) on executor page 3, the following key presses are needed:

Assign MA + X7 | View 9 At MA + X15 | Page 3 . 2 9 1 Please

This is the command result:

```
MA User name[Fixture]> Assign View 9 At Page 3.291
```

Assign View Using Keys and Screens

Requirement:

- A visible **View Pool** on one of the screens and access to executors.

To assign a view:

1. Press **Assign**.
2. Tap the desired view in the view pool.
3. Tap the desired executor.



Assign View Using the Command Line

The **Assign**, **View**, and **Page** or **Executor** keywords are needed for this command.

Assign View [View_ID] At Executor [Executor_ID]

Assign View [View_ID] At Page [Page_ID].[Executor_ID]

Recall Views

Stored views are recalled to load the view.

If a view is recalled without specifying a destination (which screen should the view be recalled to), then it, by default, opens on the screen where it was stored. The destination can be set using the /Screen preference. See the example below and learn more in the **View keyword** topic. If the view is stored with two or more screens, then the views are always recalled on the stored screens.

When the view is assigned to a view button, then pressing the view button recalls the view on the same screen as the view button.

Tapping a view in the pool recalls the view on the same screen as the tapped view pool.

Pressing an executor to recall a view uses the default behavior described above.

Recalling the view using the command line without specifying a destination, then it is recalled on the screen where the command line has focus (The command line can be on several screens). Specifying a destination in the command line overrides this default behavior.

(Call) View [View_ID]

(Call) View [View_ID] /Screen "[Screen_ID]"

Update Views

A view is updated simply by storing it again and overwriting the existing view. This can be done directly in the view pool or on the view button using any of the methods described above.



9.4. Remove Windows from a Screen

There are three different ways to remove or delete a window from a screen: A single window can be deleted, all windows on a screen can be deleted, or all windows can be deleted on all screens

This can be done using the command line or the GUI. This topic is about the GUI method. For information about the command method, please see the [Delete keyword](#) and the [ScreenContent keyword](#).

Delete a Single Window on a Screen

Requirement:

A window needs to be on the screen.

1. Tap the MA icon in the windows title bar
 - The settings pop-up appears.
2. Tap the **Delete Window** button in the title bar of the settings pop-up.

The window is now deleted from the screen.

Delete All Windows from a Screen

There are two methods to do this. Using the **Add Window pop-up** or the **Display pop-up**.

Add Window Method


Requirement:

An empty area on the screen.

1. Tap the empty area on the screen.
2. Tap the **Delete Screen** button in the **Add Window** pop-up.


All windows are now deleted from the screen.

Display Method

1. Open the Menu and Display pop-up by pressing the **Menu** key or tap the  icon in the **Control Bar**.
2. In the Display pop-up, tap the **Delete This Screen** button on the screen that needs to be deleted.

All windows are now deleted from the screen.

Delete All Windows from All Screens

1. Open the Menu and Display pop-up by pressing the **Menu** key or tap the  icon in the **Control Bar**.
2. In the Display pop-up, tap the **Delete All Screens** button on any of the screens.

All windows are now deleted from all screens.

9.5. Window Settings

All windows have settings.


The settings can be accessed by tapping the MA logo in the title bar or title field.

The settings vary depending on the window. The settings are organized in different sections, visualized by different tabs.

The **Display** section looks like this for the **Fixture Sheet**:



Fixture Sheet Settings - Display tab

There are some common buttons in the settings title bar. The red **Delete Window** button deletes the window from the screen. The  closes the settings pop-up.

The **Save** button stores the current settings as a user preference. The **Load** button is used to load stored preferences. Read more [below](#) to learn more about how to store and load preferences.

The number of tabs changes depending on the window. Some windows have a lot of settings; some have fewer.

Changing a setting does not close the settings pop-up.



Some settings open small selection pop-ups listing the different properties for the setting.

Display / Sheet

This tab has settings regarding the way the window displays sheet information. The settings here vary a lot depending on the window. The settings that are special for a single window type are described in the topic about the window. Some of these settings are in a Display tab in the Phaser Editor some of the settings are in the Sheet tab.

The following is a list of the settings that are shared between two or more windows:

- **#Columns:**
This input button sets how many columns a sheet should display (the settings **Transpose** and **Adjust Columns** must be switched on in all other sheets than the **DMX Sheet**).
- **Adjust Columns:**
This On/Off button makes a sheet adjust the column width to match the window size and the number of columns.
- **Appearance:**
Tapping this button opens a **Select Appearance** pop-up that lists all the defined appearances and the possibility of creating a new appearance. Selecting one will apply that appearance to the window.
- **Color Mode:**
This switches the color readout between RGB and CMY. The default value is to follow the setting in the **User Profile**. The user profile setting is shown between "<>".
- **Cue Only:**
It defines if the cue only function is On/Off when editing values in the **Sequence Sheet in track sheet mode** (only visible when that mode is On) and in the **Content Sheet**.
- **Executors:**
This is an On/Off button in the **Playback window** and the **Xkeys window**. It is used to show or hide the executors in the window. If executors are hidden, and labels are shown, then it looks like the executor labels in the Playback Bar on the letterbox screens.
- **Feature Graphic:**
Shows or hides a small graphic next to each feature in the **Fixture Sheet**, **Sequence Sheet**, and **Content Sheet**.
- **Feature Sort:**
This On/Off button activates feature sorting. The selected feature is moved before the other features in the **Fixture Sheet**, **Sequence Sheet**, and **Content Sheet**.
- **Fixed Target:**
This setting defines the sequence a sheet displays if the **Link Type** is **Fixed**. Tapping this setting opens an **Assignment Editor** pop-up where a sequence can be selected.
- **Fixture Appearance:**
This On/Off button shows or hides the appearance of the fixture in the **Fixture Sheet** and the **Content Sheet**.



- **Fixture Graphic:**

This defines which graphics are displayed in front of the name column in **Fixture Sheets** and **Content Sheets**. Resizing the name column to a small size will hide the graphic. This setting has the following options:

 - **None:**

No graphic is shown.
 - **Flip:**

Adds the flip indicator, for fixtures with position attributes, on the left side of the **Name** column.
 - **Simple:**

Adds a simple square graphic indicating combined color and dimmer values next to the flip indicator in the **Name** column.
 - **Gobo:**

Adds a gobo image on the simple graphic. It only displays the gobo of one gobo wheel at a time. Gobo wheels in ascending order define which gobo is displayed. For example, when Gobo 1 is set to open, then the gobo of Gobo 2 is displayed.
- **Fixture Sort:**

This On/Off button activates the sorting of fixtures. The fixtures are sorted in the selection order to the top of the **Fixture Sheet**, **Content Sheet**, and **Sequence Sheet** in track mode.
- **Font Size:**

This selects the font size in the window. It is a swipe button that opens a list of sizes from 10 to 32. There is also a **Default** property. This is the same as size 18.
- **Frame Readout:**

This defines the frame readout for this window. It can be used to overwrite the **user profile default** readout.
- **Labels:**

This is an On/Off button in the **Playback window** and the **Xkeys window**. It is used to show or hide the labels in the window.
- **Layer:**

It selects which layer is displayed in the window. It is a swipe button that opens a list of the layers. A special property is **Auto**. This property makes the window follow the selected layer in the **Encoder Bar**.
- **Layer Toolbar:**

This On/Off button shows or hides a **layer toolbar** at the bottom with the different Layers. This is a property for the **Fixture Sheet**, **Sequence Sheet**, **Content Sheet**, and **Phaser Editor**.



- **Link Type:**

This setting defines what sequence is shown in **Sequence Sheets** and **Content Sheets**. There are three different link types. This button gives access to choosing the desired one. The options are:

 - **Fixed:**

The sheet displays the information from a specific sequence. The selection is made in the Sheet Settings. Read about the **Fixed Target** setting above.
 - **Selected:**

The sheet displays information from the selected sequence.
 - **LastGo:**

This automatically shows the latest sequence to receive one of the trigger commands (<<<, >>>, Go+, Go-, Goto, Load, On, Select, Top, Temp, Flash, Toggle On, Pause). This includes if the sequence is triggered from a running timecode recording. A sequence can be excluded from LastGo by turning Off the **Include Link Last Go** setting in the **Sequence Settings**. LastGo only shows sequences triggered by the same user profile.

- **Merge Cells:**

When this is On, the values of a feature are merged to only be shown once if the two or more adjacent values are the same. For instance, if all red, green, and blue values are 100, then there is only shown 100 once.

- **Page:**

This is used in the **Playback window**, the **Command Wing Bar** window, and the **Xkeys window**. It is used to change which executor page they relate to.

- **Preset:**

This defines how the preset information is displayed in the **Fixture Sheet**, **Sequence Sheet**, and **Phaser Editor**. There are six properties which are different combinations of these three elements:

 - **ID:**

Shows the ID number of the preset.
 - **Name:**

Shows the name of the preset.
 - **Value:**

Shows the values stored in the preset.

- **Readout:**

This selects the value readout for fixture attributes. It is a swipe button that opens a list of readout types. Here is also an **Auto** property. It makes the sheet follow the selected readout in the **Encoder Bar**.

- **Setup:**

This changes the **3D Viewer**, **Agenda Viewer**, **Layout Viewer**, and **Timecode Viewer** into a mode where the different elements in the two views can be changed.

- **Show Empty Rows:**

This is a sheet setting in the **Phaser Editor**. It shows or hides the empty attribute and layers rows in the sheet view of the editor.

- **Show Grand Master:**

This shows or hides the Grand Master section in the **Custom Master Section** Window, the **command wing Bar** window, and the **Encoder Bar** Window.



- **Show Master Area or Show Master Section:**
This shows or hides the Master area in the **Custom Master Section** Window and the **command wing Bar** window.
- **Show Title Bar:**
This shows or hides the windows title bar in **Layout Viewer**, **Clock Viewer**, and **3D Viewer**. It is On by default. If it is Off, then the title bar can be shown temporarily by pressing both **MA** keys in the control area. In grandMA3 on PC, the title bar can be temporarily shown by pressing **Ctrl + Alt** on Windows and **Ctrl + Option** on Mac.
- **Speed:**
Setting in the **Fixture Sheet** and the **Phaser Editor**. It sets how the speed value is displayed. It has the following options: Auto (following the User Profile setting), Hertz, BPM (Beats Per Minute), and Seconds.
- **Step:**
It selects what step to display. Steps are used with **Phasers**. It is a property input button that opens a calculator pop-up. This is a property for the **Fixture Sheet** and **Sequence Sheet**.
- **Time Format:**
This defines the time format for the **Fixture Sheet** and the **Sequence Sheet** windows. This can be used to select a different format than the **user profile default**.
- **Transpose:**
This On/Off button flips the columns and rows in windows.
- **View Mode:**
The view mode defines how the different data and information are displayed in the view. This setting is also a button in the title bar in the two relevant windows.
The **Agenda Viewer** has the following view modes: Sheet, Year, Month, Week, and Day.
The **Phaser Editor** has the following view modes: Auto, 2D, 1D, and Step.
The **Sound Viewer** has the following view modes: Wave, Sound, and Beat.
The **Timecode Viewer** has the following view modes: Text, Timeline, and Both.
- **Wing ID:**
Defines which wing the **command wing Bar** window and the **Playback** window displays. Tap this setting to open a small **Select WingID** pop-up where the desired wing can be selected.

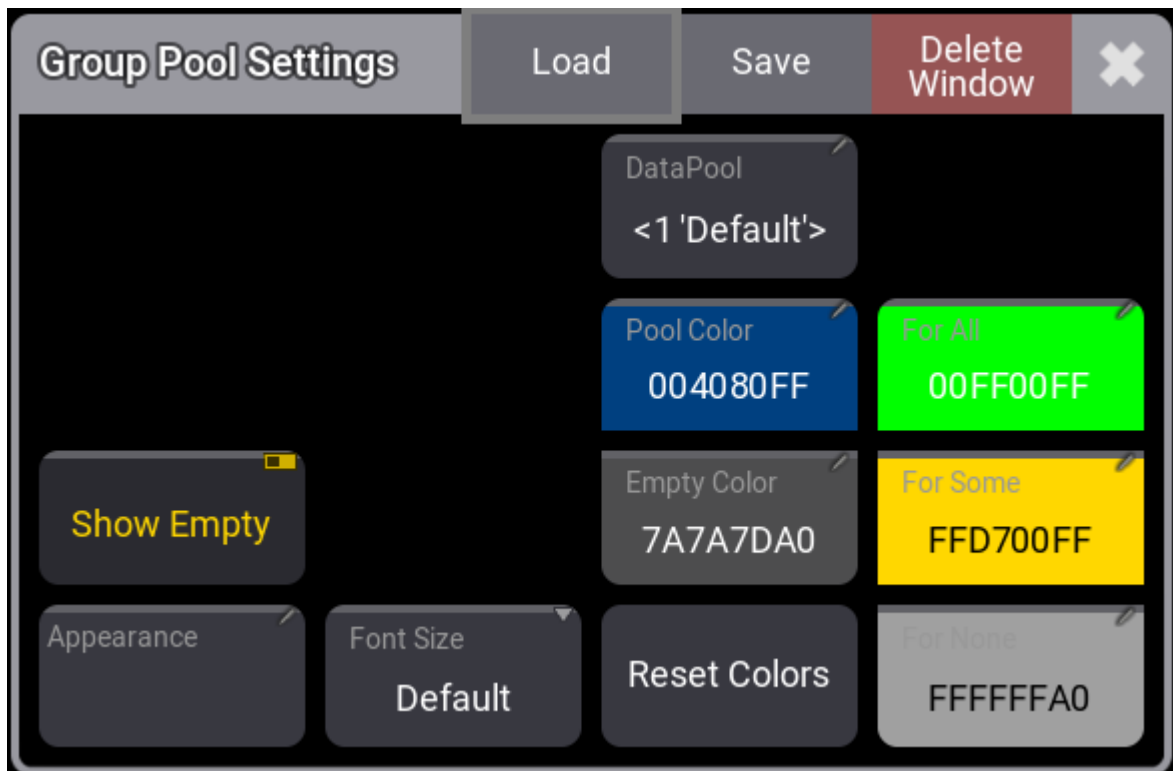


Hint:

Showing the Layout View or the 3D window in a true full-screen version can be achieved by hiding all other visual elements on the screen using the **Configure Display pop-up** and then turning Off the **Show Title Bar** setting for the window.

Pool Settings

Most pools have the same few settings. Some pools might have extra settings. The Group and Preset pools have three extra settings regarding the colored bar above each pool object. The **Smart window** behaves much like a pool and shares most of the pool settings.



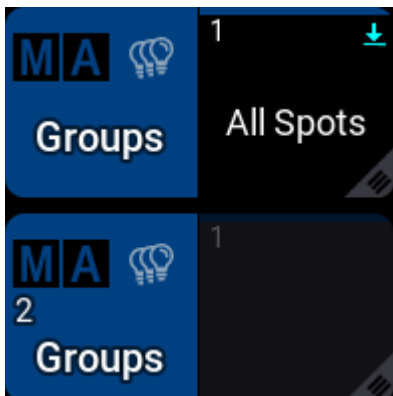
General pool window settings

These are the common settings for pools:

- **Show Empty:**
This toggle button can hide or show empty pool objects.
- **Appearance:**
The appearance is applied behind the pool objects.
- **Font Size:**
There are some different font size properties from 10 to 32. There is also a default property. This is the same as size 18. This simply changes the font size on the pool objects.
- **DataPool** (only available for some pools):
This defines what data pool the pool window shows data from. This makes it possible to have pools showing objects from different data pools. For instance, a group pool window from the default data pool can be shown next to a different group pool window showing groups from a different data pool.
- **Pool Color:**
This is the color for the title button in the pool.

- **Empty Color:**
This color is applied to empty pool objects.
- **Reset Colors:**
This resets the colors to the colors in the default color theme.

The data pool number is shown in the lower right corner of the pool title button. The number is only shown if the pool window supports showing different data pools and if there is more than one data pool in the show.



Two group pools with different data pools

Preset and group pools also have:

- **For All:**
This color is used when the preset can be used by all of the selected fixtures.
- **For Some:**
This color is used when some of the selected fixtures can use the preset or when more fixtures can use the preset than the current selection.
- **For None:**
This color is used when the preset is not useable by any of the selected fixtures or none of the selected fixtures are in the group.

Tapping one of the color settings opens an **Edit Color** pop-up.

The buttons in the title bar are the same as the ones for other window settings - read above.

Default Pool Action

Some pools have a default **Action** setting. This defines the default action executed when a pool object is tapped without a (relevant) keyword in the command line.

A small icon in the upper right corner of the pool title object indicates the default action.



Pools with default pool actions

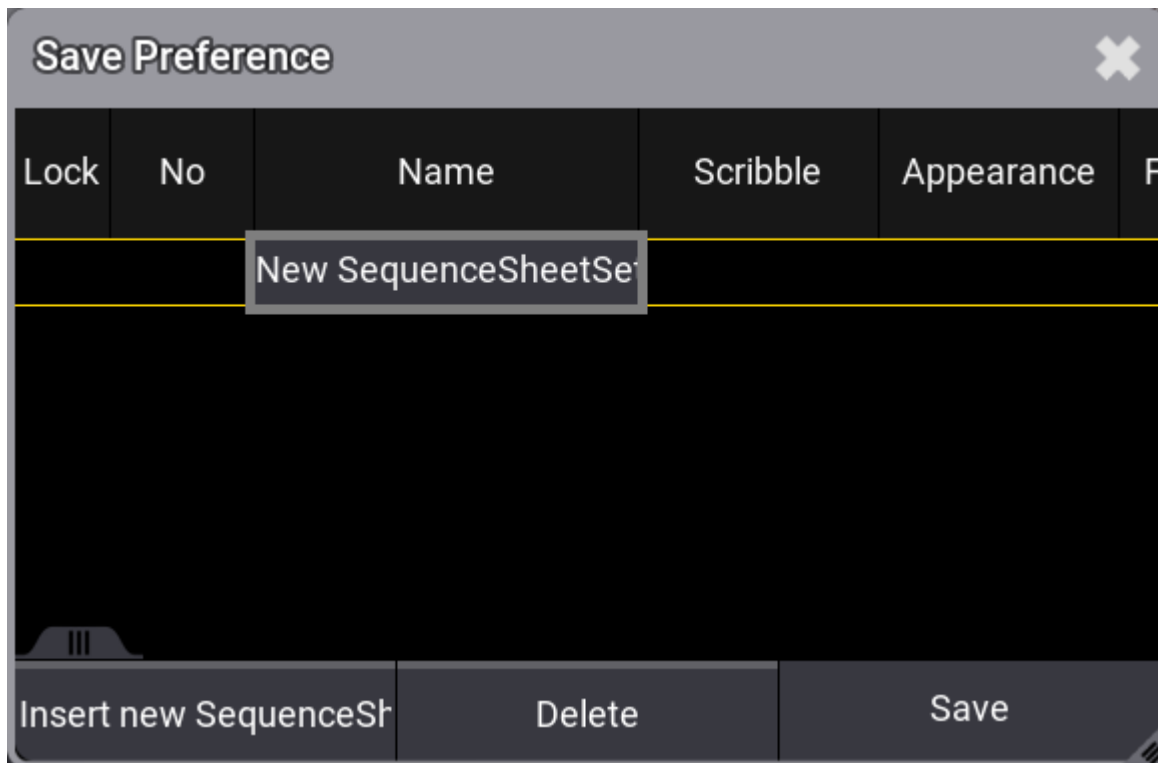
Pools can have the following actions:

- **At** (@):
When there is no selection in the programmer, tapping a preset does nothing. When the programmer has a selection, tapping it calls the preset into the programmer.
- **SelfFix/At** (no icon) - default action for presets:
When there is no selection in the programmer, tapping a preset will select the fixture that can use the preset. Tapping it again calls the preset into the programmer.
When the programmer has a selection, tapping a preset the first time will call the preset into the programmer.
- **Select** (S) - default action for sequences:
Tapping the pool object selects it.
- **Toggle** (🔌):
Tapping a pool object plays it back or switches it off, depending on its current playback state.
- **Go+** (▶):
Starts playback of the pool object or goes to the next cue in the sequence.
- **Flash** (⬆):
Flashes a pool object as long as it is tapped. Flash ignores fade times.
- **Temp** (⬇):
Plays back a pool object as long the pool object is pressed. Temp respects the fade times.

Store and Load Preferences

Setting preferences can be stored and loaded for each window and pool. These preferences are stored in the user profile. This means that exporting and importing a user profile includes these preferences.

Tapping **Save** opens a **Save Preferences** pop-up:



Save Preference pop-up

Save a New Preference

1. Open the **Settings** pop-up.
2. Tap **Save** in the settings title bar.
3. Tap **Insert new [Type]** (this button changes a little for each window, and the type shows what type of settings are being inserted).
4. Edit the name field and give it a name.
5. Tap **Save** in the preference pop-up.



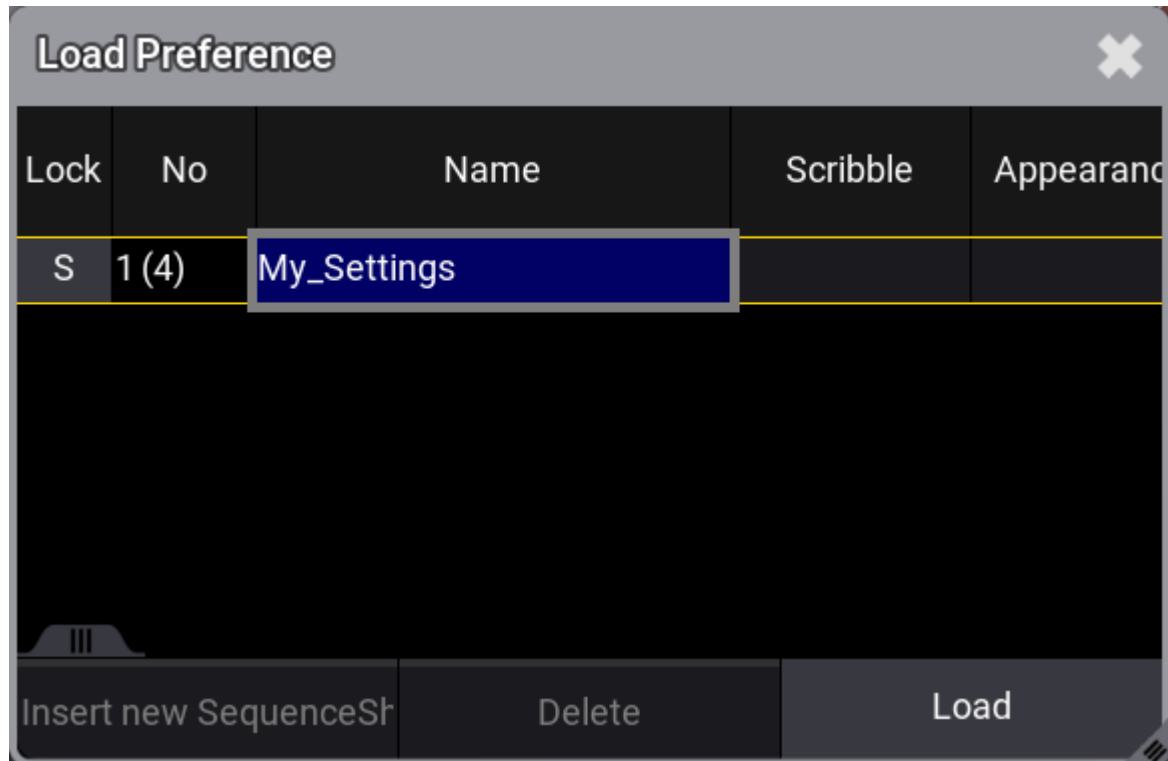
Hint:

When saving a new preference, no preference exists yet, and the focus is on "New ...", it is enough to tap **Save** to save the preference.

Update the Preference

1. Open the **Settings** pop-up.
2. Tap **Save** in the settings title bar.
3. Tap the desired preference in the list.
4. Tap **Save** in the preference pop-up.


Load a Preference



Load preference pop-up

1. Open the **Settings** pop-up.
2. Tap **Load** in the settings title bar.
3. Select the desired preferences.
4. Tap **Load** in the preference pop-up to confirm.

Delete a Preference

1. Open the **Settings** pop-up.
2. Tap **Save** in the settings title bar.
3. Select the desired preferences.
4. Tap **Delete** in the preference pop-up.
5. Close the pop-up by tapping the  in the upper right corner.

9.6. Title Bar Configuration

Many windows with a title bar can edit the arrangements of the buttons in the title bar.

These settings follow the user profile so that different users can have different default preferences. Also, the setting is stored with the view. For instance, a view can have several fixture sheets with some buttons in one window and others in a different window.

The title bar buttons can be edited in the window settings. Open the settings by tapping the **MA** logo in the upper left corner.

The settings title bar has an **Edit Title Bar**. Toggling this On changes how the setting pop-up works and looks.



Fixture Sheet Setting with Edit Title Bar On

Some settings can have a button in the title bar. These settings can be toggled to **Yes** in the different tabs in the setting pop-up.

The number on the lower right side of the settings shows the order of the button counting from right to left. The latest activated button will have the highest number and be on the left side in the title bar.


The number in the tab displays how many settings are set to **Yes** out of the possible settings in the tab. For instance, "3/11" is three out of 11 possible settings with a button in the title bar.

Turning the edit mode On also adds an extra tab called **Title Buttons**.



Title Buttons tab

This tab is split into two sides.

The left side lists all the active title bar buttons. The Selected button can be moved up or down using the arrow buttons on the left side. The selected button can also be deleted by tapping the  icon.

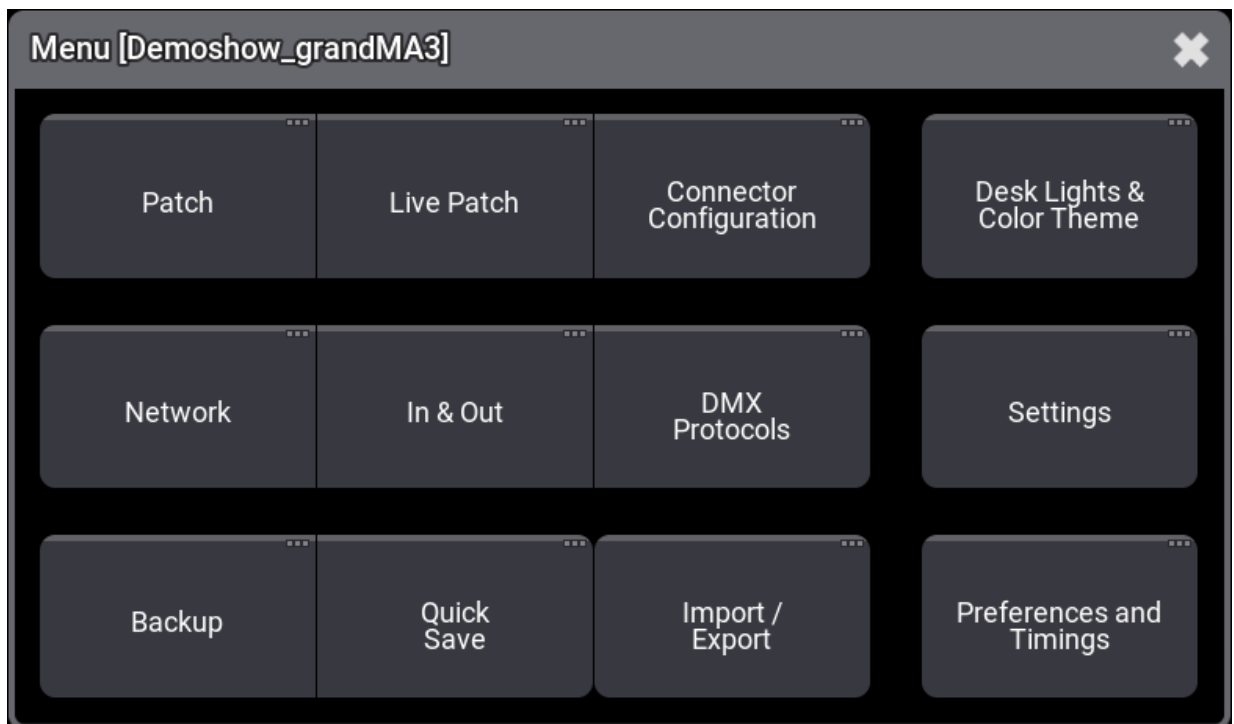
The top button is on the right side of the title bar.

The right side of the tab has two buttons. **Load Defaults** loads the default button configuration for the window. **Delete All** deletes all the buttons from the title bar.

9.7. Menus


There are many different menus in the grandMA3 software. Most are described in relevant sections. For instance, the patch menu is described in the [Patch section](#).


There is a **Menu pop-up** that gives access to many of the different menus.



Menu pop-up with buttons that opens a lot of menus

This pop-up can be opened in multiple ways:

- On physical hardware, there is a **Menu** key. Press this to open the pop-up.
- In the **Control Bar** on the left side of most displays, on the onPC, there is a gear icon . Tap this to open the pop-up.
- It can be opened using the command line:

```
 User name[Fixture]> Menu "MenuSelector"
```

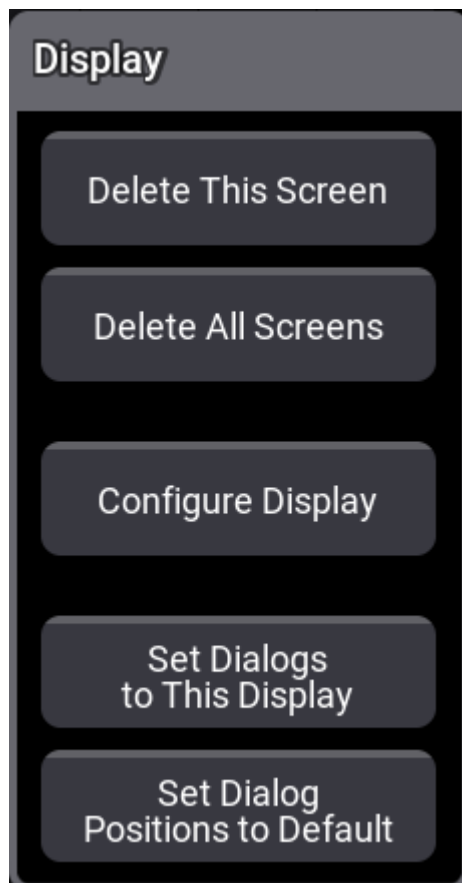


The menu pop-up gives access to the following menus:

- **Patch:**
This is where fixtures are added and all about fixture setup - Read more in the [Patch and Fixture Setup section](#).
- **Live Patch:**
This menu gives access to the fixture setup that can be changed without a new show upload - Read more in the [Patch and Fixture Setup section](#).
- **Connector Configuration:**
The [connector configuration](#) changes the local physical DMX ports on the MA equipment.
- **Desk Lights & Color Theme:**
This opens a pop-up that gives access to [customize the look of the software](#).
- **Network:**
This menu is used to set up all about the network and sessions - read more in the [Networking section](#).
- **In & Out:**
This opens the [In & Out menu](#) with settings for DC remote, MIDI remote, DMX remotes, OSC, and PSN.
- **DMX Protocols:**
This menu is used to set up DMX [network protocols](#).
- **Settings:** This opens a small sub-menu that has these options:
 - **User Configuration:**
This menu is used to set up users and user profiles - read more in the [Single User and Multi-User Systems section](#).
 - **Date and Time:**
This menu is used to change the time and date in the console - read more in the [Date and Time topic](#).
 - **USB Configuration:**
This is used to configure USB devices. This should normally not be changed manually.
 - **Software Update:**
This menu is used to [update the software](#).
 - **Touch Configuration:**
This menu is used to assign touch screens to the USB inputs.
 - **Extension Configuration:**
This menu shows the connected grandMA3 extensions.
 - **onPC Settings:**
This is a pop-up with [onPC settings](#).
- **Backup:**
The backup menu is used to save and load shows - read more in the [Show File Handling topic](#).
- **Quick Save:**
This is not a menu but a shortcut to saving the show with the same show file name.
- **Import / Export:**
This opens an [Import and Export menu](#) that can be used to import and export different objects of the show file.
- **Preferences and Timings:**
This is where the defaults are set up for cues and sequences - read more in the [Cue Timing topic](#).

Display Overlay

Opening the **Menu pop-up** also opens a **Display overlay**.



Display overlay in the right-hand lower corner of all screens

This pop-up appears on all screens except the letterbox screens.

Most of the actions performed using this pop-up relate to the specific screen where the menu is touched. For instance, deleting a screen (removing the windows from the screen).

- **Delete All Screens:**
This clears all windows from screens - read more in the [Remove Windows from a Screen topic](#).
- **Delete This Screen:**
This clears all windows from this screen - read more in the [Remove Windows from a Screen topic](#).
- **Configure Display:**
This opens the Configure Display pop-up - read more in the [Configuration of Displays topic](#).
- **Set Dialogs to This Display:**
This makes all menus open on this screen - read more in the [Change Menu Locations topic](#).



- **Set Dialog Positions to Default:**

This resets all menus to open on the default screen - read more in the [Change Menu Locations topic](#).

9.8. Change Menu Locations

Some menus can be moved between the screens.

Menus that can be moved have a display icon in the title bar:



Display icon next to the close X

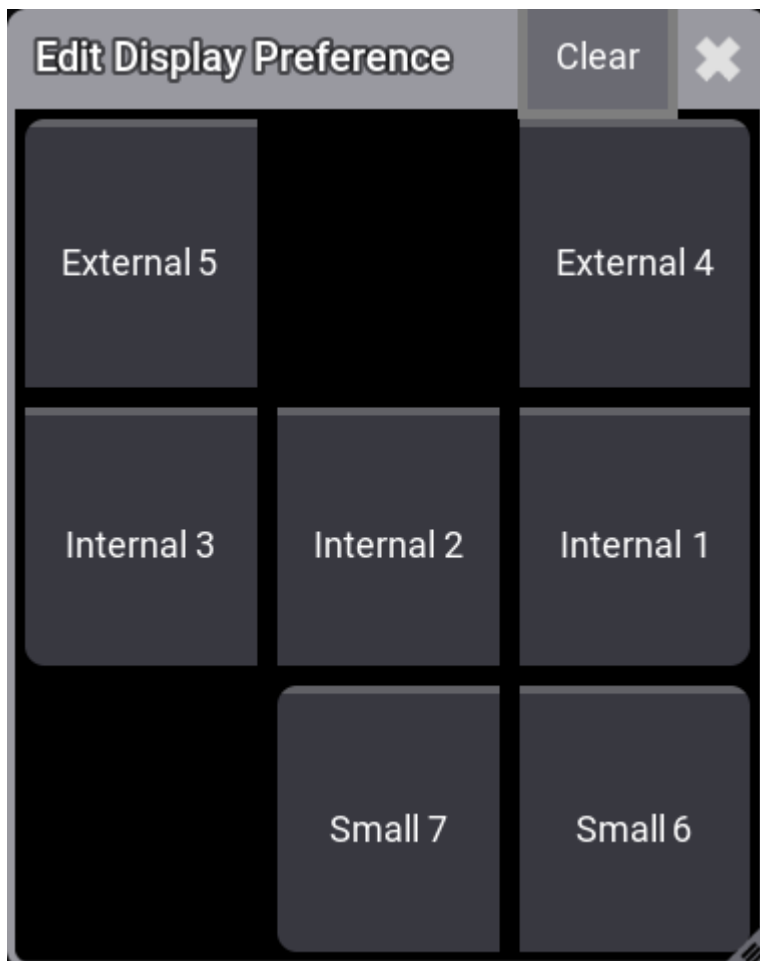
Move a Menu to a Different Screen

Requirement:

- Open a menu that can be moved. For instance the **Playback Controls** Menu.

1. Tap the display icon in the title bar.

The **Edit Display Preference pop-up** appears:




Use Edit Display Preference pop-up to select a new location

2. Tap the button corresponding to the desired screen.



Hint:

Tap the **Clear** button or the  to cancel the move.

Move all Menus to a Screen

It is possible to select a screen where all moveable menus opens.

1. Open the **Menu and Display pop-up**.
2. In the Display pop-up tap the **Set Dialog to This Display** button on the screen where the menus should open.

Reset Movable Menus Screen

The screen selection can be reset. This is useful if a show changes from a smaller hardware platform. For instance, if the show is programmed on a grandMA3 full-size and menus are moved to screen 3 and now the show is opened on a grandMA3 light without screen 3.

1. Open the **Menu and Display pop-up**.
2. In the Display pop-up tap the **Set Dialog Positions to Default** button on the screen where the menus should open.

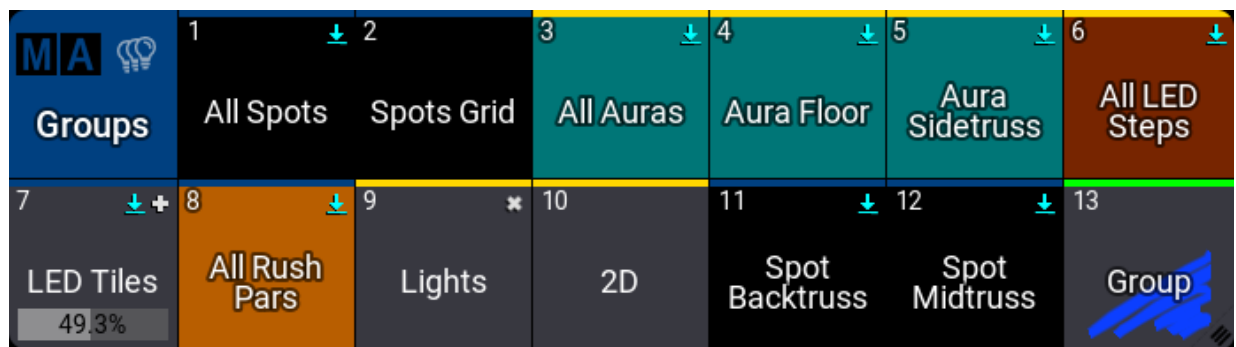
9.9. Pool Windows

Most of the show data in grandMA3 are organized in pools - these pools are part of a **Data Pool**. So, in essence, everything except the patch and fixture setup are stored inside a Data Pool. This makes it easy to have several shows with the same fixture setup but different data pools.

Pools contain a lot of different data. For instance, the **Groups pool** contains information about the selections of fixtures. The **Views pool** contains information about the arrangement of windows on a screen.

Pools are made like any other window - read more in the **Add Windows topic**. Pool arrangement and resizing are just like any other window - read more in the **Rearrange topic**.

This is an example of the Groups pool:



Groups pool

Each pool has a title field. In the example above it is the first blue square with the MA logo, the pool icon (if available), and the name of the pool.

Tapping the logo opens the settings for the entire pool - read more about the common window settings in the **Window Settings** topic. For details about pool or preset specific settings, read the relevant topics.

Some pools have specific use. This topic and the following subtopics describe the general concepts. Read the specific topics to learn the details of each type of pool.

Pool Object

All other fields except the title field contain a **Pool Object**. If something is stored in the pool object, then it has a lighter gray color (default color), and there is a label (see the example pool above). It can also contain pool specific information like a master level or icons that show information about the content or pool object settings.

The number in the upper left corner of each pool object is the unique pool number identifier. It can be used when the object is called. For instance, using **Group 1** in the command line is the first object in the groups pool. The name of the pool object can also be used to call the group. For example, **Group "All Spots"** calls (all) the groups with that name. Asterisk can be used as a joker sign when using object names. Typing **Group "All*"** calls all the group objects where the name starts with **All**.

Working with or using pool objects can be done in many ways. The examples in the subtopics use the command line, but many of the operations could also be performed using the keys, screens, or any combination thereof.

Some pools have **Selected** pool objects. It is visualized with a yellow frame around the selected pool object.



World 1 is selected

A different pool object can usually be selected by tapping the object in the pool or using the **Select keyword**.

Swipey Commands

Each pool object can open a set of **Swipey Commands**. They give quick access to some common operations with the pool object.

This is how to open the swipey commands:

1. Tap and hold the pool object.
2. Move the finger/cursor outside the pool object while keep pressing the screen.



The open Swipey Commands

3. Keep the screen pressed and move the finger/cursor to the desired command.
4. Release the screen.



These are the available commands with the swipecs:

- **Assign:**
This adds "Assign" and the pool object type and number in the command line. Waiting for more user input.
- **Edit:**
This starts the edit mode for the pool object - this is about the content of the pool object.
- **Edit Setting:**
This opens a small editor that gives access to editing this pool object's settings - this is about the settings for the pool object.
- **Store:**
This executes "Store" plus the pool object type and number. This is useful for storing new pool objects - for instance, groups.
- **Label:**
This executes "Label" plus the pool object type and number.
- **Delete:**
This executes "Delete" plus the pool object type and number - The result is deleting the pool object.
- **Move:**
This adds "Move" plus the pool object type and number in the command line. Waiting for the user to tap the new location.
- **Copy:**
This adds "Copy" plus the pool object type and number in the command line. Waiting for the user to tap the new location.



9.9.1. Create Pool Objects

This is a general description of how to create pool objects.

The **Store keyword** is used when creating pool objects. This stores the relevant information in the pool object if the relevant information is available.

For instance, to store a preset, there need to be correct values in the programmer. If the relevant values are not available, then an empty pool object is created.

Some pool objects are a bit more complex. For instance, storing a pool object in the image pool. The pool that can be added on a screen, is not just called "Images" as the title suggests. It is actually image "pool" 3.

So storing something on pool object 2 needs the following input:

```
MA User name[Fixture]> Store Image 3.2
```

But this only stores an empty pool object.

Some objects are better to **Edit** when creating them. This opens the relevant editor to create the object.

Images are an example of such an object.

```
MA User name[Fixture]> Edit Image 3.2
```

This command creates the object and immediately opens the editor for the image pool object. Read more about images in the **Images section**.

Some pool objects can be created by pressing and holding an empty pool object. If there is relevant information in the programmer, then this might be stored in the new pool object.

For instance, having a selection of fixtures in the programmer and pressing and holding an empty group pool object for about 2 seconds will create a new group with the fixture selection.

9.9.2. Label Pool Objects

All pool objects can be labeled. The label contains a name, a scribble, and an appearance.

The name can be changed using the **Label** keyword. It can be written in the command line. Write **Label** followed by the object you want to name and then the name in quotation marks.

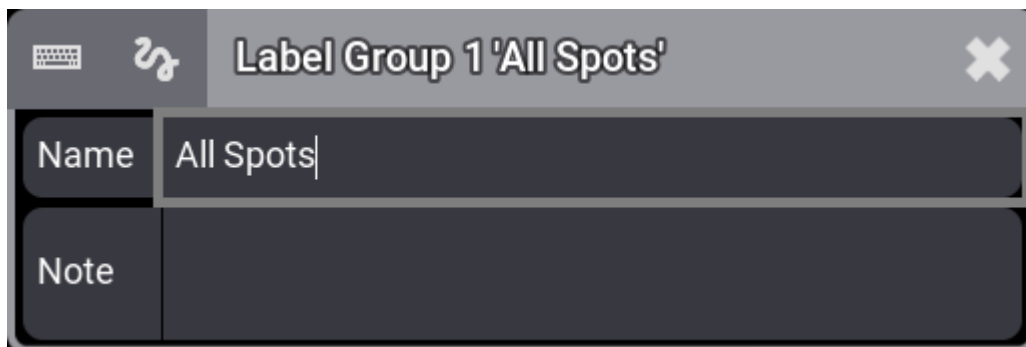
For instance, if group 1 needs to be named "Stage Right Spots" use the following command:

```
MA User name[Fixture]> Label Group 1 "Stage Right Spots"
```


The label command can be accessed by pressing the **Assign** **key** twice. The syntax is the same no matter how the command is created:


Label [Object_Type] ["Current_Object_Name" or Object_Number] "New_Object Name"

If the object name is omitted in the command, an **Edit Label** pop-up appears:



Enter a label

The keyboard icon () opens an on-screen keyboard. If the keyboard is visible, then the icon is yellow.

The scribble icon () opens the **Scribble pad**. If the Scribble pad is visible, then the icon is yellow.

The label pop-up also has easy access to add a note to the object. Learn more in the **Notes topic**.

The last touched pool object has a white frame around it. If the keyboard is used, the keyboard text input is often interpreted as a new label input, which can be used to label all pool objects. This is the fastest way to label newly created objects.

Scribbles

Scribbles are small drawings that can be used together with or instead of a label.

Scribbles from the Scribble pool can be assigned to a pool object. This is the syntax:

Assign Scribble ["Scribble_Name" or Scribble_Number] At [Object_Type] ["Object_Name" or Object_Number]



For instance, if scribble 3 is to be used with Sequence 10, then the following command can be used:

```
MA User name[Fixture]> Assign Scribble 3 At Sequence 10
```

Learn more about scribbles in the [Scribble section](#).

Appearance

An **Appearance** can be assigned to pool objects.

This can also serve as a labeling function. This is the syntax :

Assign Appearance ["Appearance_Name" or Appearance_Number] At [Object_Type] ["Object_Name" or Object_Number]

For instance, if appearance 5 is to be used for world 4, then the following command can be used:

```
MA User name[Fixture]> Assign Appearance 5 At World 4
```

Learn more about appearances in the [Appearance section](#).




9.9.3. Move Pool Objects

Pool objects are moved using the **Move** keyword.

A single object can be moved or it can be a selection of several objects moved at once.

Example

Macro 5 needs to be moved to the empty macro pool object 20:

```
 User name[Fixture]> Move Macro 5 At 20
```

If the destination is not empty, then the existing object will move one place further. If this object is also occupied, it will move further, and so on until no occupied object has to move anymore. Thus potentially moving all the arranged pool objects.



Important:

Please have a look at the **Exchange keyword** for grandMA2 move functionality.



9.9.4. Insert Pool Objects

Pool objects can be inserted between other pool objects of the same type. This can be useful for organizing the pools.

It is done using the **Insert** keyword.

The inserted pool object is inserted before the destination object number.

Example

For example inserting view 42 between view 9 and 10:

```
MA User name[Fixture]> Insert View 42 At 10
```

The old view 10 and other views from 10 to the next empty pool object is then moved one number up.



9.9.5. Copy Pool Objects

Pool objects can be copied to an empty location using the **Copy** keyword.

A single object can be copied or it can be a selection of several objects copied at once. They are positioned at the new location using the selection order.

Example

Copy sequence 2 at pool object 22 using the following command:


```
MA User name[Fixture]> Copy Sequence 2 At 22
```

If the destination is not empty then the old object is overwritten!

9.9.6. Lock and Unlock Pool Objects

Lock Pool Objects

Pool objects can be locked using the **Lock** keyword.

If a pool object is locked then there is a small lock icon  in the upper right corner of the pool object.

Predefined pool objects are locked from the factory and cannot be unlocked.



Hint:

The big red lock icon that can appear on pool objects is not a locked object using this lock mechanism. It is temporarily blocked by a user editing the object. Learn more in the [Object Ownership topic](#).

Example

Locking appearance number 4 is done using the following command:

```
MA User name[Fixture]> Lock Appearance 4
```

Unlock Pool Objects

Objects can be unlocked using the **Unlock** keyword. The syntax is the same as locking except the keyword.


Example

Unlocking appearance number 4 is done using this command:

```
MA User name[Fixture]> Unlock Appearance 4
```


9.9.7. Delete Pool Objects

Removing pool objects can be done using the **Delete** keyword.

 **Restriction:**
When a pool object is deleted, then all links to the object are lost. This means that if the pool object is returned using the oops function, then links to where it was used are not returned even though the pool object is back in the pool.

When an object is deleted, then grandMA3 will try to make everything look the same. For instance, deleting a preset will move the values stored in the preset into the different cues where the preset was used. This is not always possible, for instance, if an executor is controlling a sequence and the sequence is deleted, then it is gone and the executor is empty.

Example

Delete view 42:

```
MA User name[Fixture]> Delete View 42
```

It can also be a range of numbers for instance:

```
MA User name[Fixture]> Delete View 42 Thru 90
```

Deleting pool objects can be undone with **Oops**. It will bring back the pool objects but it might not restore links to the object.

For instance, deleting an appearance will remove the link to it where it might be used. Oops'ing the deletion will bring back the appearance, but it will not reassign the appearance where it was used before deletion.



10. Networking

Networking allows for the expansion of a single standalone console to a complete system with DMX nodes and extra processing powers.

There are three Ethernet connectors on the rear panel of all the consoles. They can be used to connect to three different networks.

Networking is use for:

- Connecting several MA3 devices in a session
- Output and input Ethernet-based DMX
- Internet connection

MA Session

Sessions are a way to expand a single console. It allows for connecting multiple consoles in a multi-user setup where several programmers can work in the same show. grandMA3 Processing Units can be added to move DMX calculations away from the consoles and into processors located where needed. grandMA3 xPort Nodes are listening to a session and function as DMX interfaces allowing decentralized synchronized DMX distribution.

A session is needed to connect grandMA3 devices. Read about [sessions here](#).

Ethernet DMX

grandMA3 can output DMX using Art-Net and sACN. It is possible to set up a specific Ethernet port to output the network DMX. Read more in the [DMX In and Out of the system topic](#).

Internet Connection

The grandMA3 system is designed to be able to connect to the Internet. More specifically connect to World Servers. These servers are able to provide different services to the connected stations.

It is recommended to separate the light network from the internet and use one of the three Ethernet connectors on the back of the MA hardware stations to connect to a network with internet access. This will maintain a separation between the internet and the light network and still provide access to the services on the world server.

For more information see the [World Server topic](#) and the [Interfaces and IP topic](#).

Internal Connections

A console uses internal network connections to have different sections of the internal components connected. This is visible in the Network menu, but should never be changed by the users.



Enabling or Disabling the Network Connection

The network needs to be enabled to communicate. This includes transmitting DMX using Ethernet.

Turning network On or Off is done from the **Network menu** - read about the network menu in the [Sessions topic](#).

In the lower right corner of the network menu, there is a button to toggle the network connection.



Network enable button

If the icon is red then the network is turned Off. If it is green then it is On.

10.1. Interfaces and IP

The Network Interface menu lists the available interfaces, and in the grandMA3 hardware, it can be used to change the settings.


Open the Network Interface Menu

1. Press **Menu**.
- Opens the **menu select pop-up**.
2. Tap **Network**.
- Opens the Network menu.
3. Tap **My Interfaces**.
- Opens the Network Interfaces pop-up:



Lock	No	Name	Link	DHCP	Slow	IP	Mask	Gateway	MAC
S	1 (1)	Con1	Yes	Yes	No	192.168.1.124	255.255.255.0	192.168.1.1	E4:4F:29:00:90:13
S	2 (1)	Con2	Yes	No	No	10.33.144.18	255.0.0.0		E4:4F:29:00:90:12
S	3 (1)	Con3	No	Yes	No	No Cable	255.0.0.0		E4:4F:29:00:90:11
S	4 (1)	loopback	Yes	No	No	127.0.0.1	255.0.0.0		

See the Network Interfaces

 **Restriction:**
If you see an interface with "imx6" then it is for internal communication inside the console. Do not make any changes to this interface.

Every cell with a light gray background can be edited.

The grandMA3 consoles have three network interfaces, with connectors on the rear panel, that can be used to **connect with external equipment**. The connectors are named **Con1**, **Con2**, and **Con3** in the Network Interface menu.

In the grandMA3 onPC, it is not possible to edit the IP addresses in the Network Interface menu. This needs to be done in the computer operating system.

Short Description of the Columns

- **Lock:**
Changing the value to **Yes** in this cell locks the interface from being edited.
- **No:**
This is the interface number.



- **Name:**
This is the name of the interface.
- **Link:**
This cell says **Yes** if there is an active connection on the connector. It says **No** in red if there is no connection.
- **DHCP:**
This cell says **Yes** if DHCP is activated for the interface. No means that DHCP is turned Off, and it is possible to manually set a static IP address information. Read about DHCP below.
- **Slow:**
This cell says **Yes** if the detected network speed is below 1 Gbit/s.
- **IP:**
This is the IP address for the interface. If DHCP is set to "No", this cell can be edited on the console to change the IP address for the interface. CIDR notation is accepted as input to also set the subnet mask. It says "No Cable" if it is disconnected.
- **Mask:**
This is the subnet mask for the interface. If DHCP is set to "No", this cell can be edited on the console to change the subnet mask for the interface.
- **Gateway:**
This is the gateway address for the interface. If DHCP is set to "No", this cell can be edited on the console to change the gateway address for the interface.
- **MAC:**
This is the MAC address for the interface. This is information only and cannot be edited.

What is DHCP

DHCP is a system where IP addresses are distributed from a DHCP server. If this is On (the field says "Yes"), it is impossible to set the IP address locally.

Edit the field to turn On or Off DHCP on the console.

Set the IP Address in the Console



Restriction:

The IP address range 192.168.33.x is not allowed to be used for the network interfaces Con1, Con2, or Con3.

1. Navigate to the Network Interfaces menu (read above).
2. Make sure DHCP is turned Off (the cell says "No") for the network connector (read above).
3. Edit the IP cell and write the new IP address (IPv4 only). The CIDR notation can be used to set the subnet mask while typing the IP address. For example, **192.168.101.11/24** gives the subnet mask to **255.255.255.0**.
4. Edit the Mask cell and write the subnet mask.
5. Optionally edit the Gateway cell to set a gateway address if needed.
6. Tap **Apply Changes** to use the new settings.



The changed IP address is used without the need for a reboot.



Hint:

For more information about the **Auto** option of the MA Net interface, see [Session](#).

10.2. Session

MA devices are connected in **Sessions**.

Controlling devices (grandMA3 onPC, grandMA3 replay unit, and grandMA3 consoles) are called **Stations**.

Multiple sessions can exist in the same network and are identified by a session name and location.



Hint:

Learn more about different ways the grandMA3 system can work in a networked environment in the [System Overview](#).

Session Rules

The grandMA3 Session is an essential part of a working system.

The rules to accomplish and maintain a stable session are listed below to understand the system's behavior.

- A session always creates a **Master** station.
- A session always creates a unique, matching **Session Index**, which the user cannot set.
- A session always needs a unique, matching **Session Name** across all members.
- A session always needs a matching **Location** across all members.
- All session members (stations) are displayed with a green background in the **Stations** tab in the **Network Menu**.
- All members must have the same **Streaming Version** of the software - the first three numbers of a software version number.
- Devices can be invited or intentionally join the session.
- Devices can be protected from being invited.
- Devices that lost the connection to their session will be auto-invited when coming back online - if stated in the network list of the show file and the Invite button is activated.
- Two devices with the same Session Index in the same network will cause a session data conflict when both are in Master states. The session data conflict only needs user interaction if the system cannot solve it automatically.
- All protocols will be output from the **Master** station only.
- A maximum of 32 sessions are possible in one network domain with a maximum of 1 fully loaded session (262 144 Parameter) as a total.
- Consoles in session with the same user logged in will work in **Full-Tracking Mode**.
- Consoles in session with different users logged in will work in **Multi-User Mode**.
- The network needs a latency of less than 2ms for data-package transmission (for synchronization) to avoid output jitter!
- The session timeout (before action will be taken when a station loses connection) is 5s!

Many of the elements above can be seen and edited in the network menu.



MA devices recognized in the network can be seen in the network menu.

The menu can be accessed in multiple ways.

Open the Menu Using the Command Line

Type the following command in the command line input and execute it:

```
MA User name[Fixture]> Menu "Network"
```

Open the Menu from the Menu Pop-up

This is a combination of keys and buttons. It executes the same command as above.

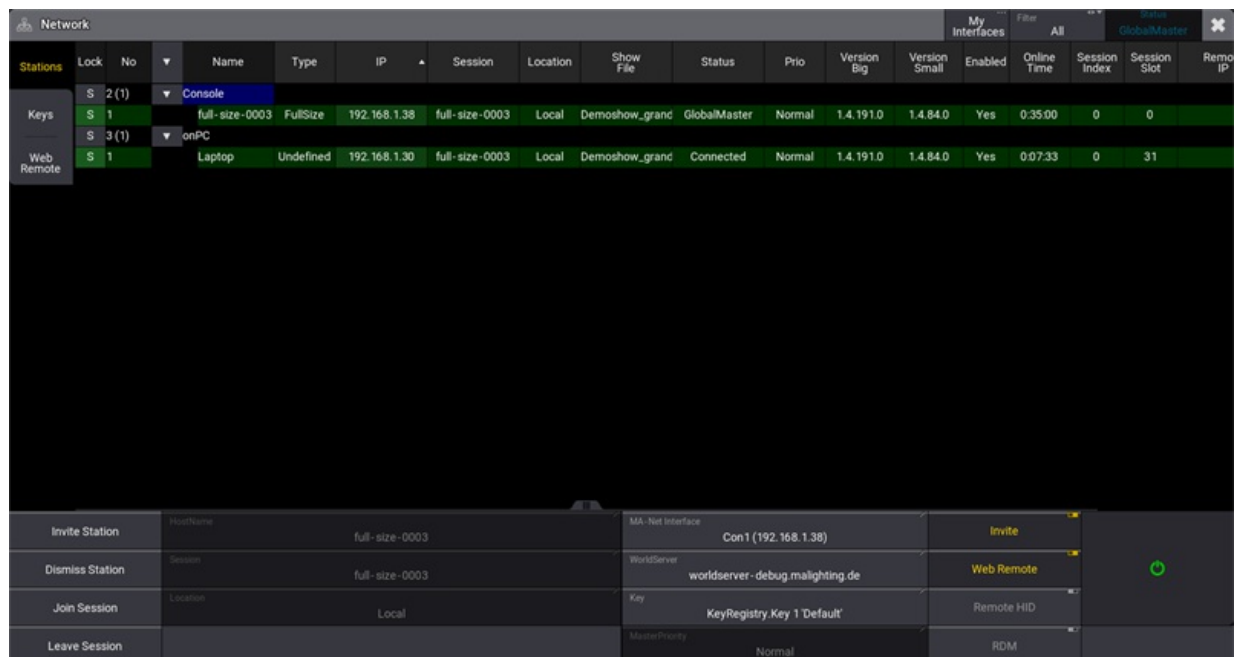
1. Press **Menu**.
2. Tap **Network** on the menu selection pop-up.

Open the Menu Using the Dedicated Button

The **Command Line** has a dedicated button for the network menu:



Tap this button to open the menu.



The Network menu lists all MA devices in the network



The title bar has the normal close button (✕). Next to this is a button that displays the current status for this device, a filter button, and a button that opens a **My Interfaces pop-up** where the network interfaces are listed and might be modified.

The possible filters are:

- **All:**
This lists all stations.
- **My Session:**
This only lists the stations that are in the same session as this station.
- **Not My Session:**
This lists the stations that are not part of this station's session.
- **Wrong Version:**
This lists stations that have a different software version number than this station.
- **My Location:**
This lists the stations that have the same **Location** value. Read about the location setting below.

The tool buttons on the left side have three buttons: Stations, **Keys**, and **Web Remote**.

There are four buttons on the left side at the bottom: **Invite Station**, **Dismiss stations**, **Join Session**, and **Leave Session**.

Some input buttons change the session settings for the station being operated.

This is a short explanation of each field:

- **HostName:**
The name of this device. The name can be changed freely. It does not influence the session. It can be used to identify individual devices. It can only be edited when the station is not in a session.
- **Session:**
The name of the session. Three elements need to match before devices can connect in the session. The session name, the **Location**, and the **Key**. It can only be edited when the station is not in a session.
- **Location:**
The session location. It can be anything. It does not need to be the actual location for the session. It is used together with the session name to identify the session. It can only be edited when the station is not in a session.
- **MA-Net Interface:**
Tapping the Interface field opens a small **Select Interface pop-up**. This lists the possible network interfaces, including an Auto option (Read more about this option **below**). Read more about changing the IP addresses in **Interfaces and IP**.
- **WorldServer:**
This input field is used to define what world server to contact. Read more about it in the **World Server topic**.
- **Key:**
The key is a password for the session. Tapping this field opens a small **Select Key pop-up** that lists the possible keys - including the option to create a new one. Read more about key creation in **Create a custom key**.

- **Master Priority:**

This field is used to set the priority for this device. Tapping this field opens the small **Select MasterPriority pop-up**. It can only be edited when the station is not in a session.

There are several levels of priority: **Never** (not able to be master), **Very Low**, **Low**, **Normal**, and **High**. If stations have the same priority, then the station with the lowest slot number becomes master. Slot numbers are automatically assigned depending on the order of devices joined in the session. The slot number is listed in the station table.



Hint:

Each station in a session has a priority. The session needs to have one station that is the master in the session. If there is only a station with the never priority, then the session is terminated since this station can never become the master.

- **Invite:**

This On/Off button toggles **Invites**. Disabling this prevents this station from being invited into a session. It is still possible to join a session from this station, but it is an action that needs to be performed locally at the station.

- **Web Remote:**

This On/Off button toggles the **Web Remote** access. Turning this off prevents web remote connections.

- **Remote HID:**

This On/Off button toggles **Remote HID**. Learn more about this in the [Remote HID topic](#).

- **RDM:**

This is a global RDM On/Off button. Learn more about RDM in the [RDM section](#).

- **Enable button** (🔌):

This button **enables or disables** the network connections.

- A red icon means the network connection is disabled. A text beneath the icon describes if a session is created or joined if the network is enabled.
- A green icon means the network connection is enabled.
- A green icon with a red background means the connection is enabled, but the interface is not connected.

Auto Option in MA-Net Interface Select Interface Pop-up

The dropdown for selecting the interface used for MA-Net in the network menu has an **Auto** option.

When the MA-Net interface is set to Auto, the grandMA3 software determines the interface to use. In this case, the IP of the selected interface is set into angle brackets on the **MA-Net Interface** button, for example, <192.168.0.4>.

The order of the following rules specifies the automatically determined interface in Auto mode:

1. When a Class C IP address (192.168.x.y) is found, the interface with this IP will be taken. No matter the link state of this interface. (With onPC on Mac, the interface link state must be active.)
2. When no Class C IP is found, the software searches for a Class B IP (172.16.x.y). Only if the interface has an active link state.



3. If also no Class B IP address is available, a Class A IP in the range of 10 (10.x.y.z) will be searched only if the interface has an active link state.
4. If this also fails, a Class A IP in the range of 2 (2.x.y.z) will be searched only if the interface has an active link state.
5. The loopback interface will be used if no Class A IP is available.

Changing the selected interface from Auto to the preferred interface is always possible.

On grandMA3 consoles, Processing Units, and xPort Nodes, the first interface will still be selected as the default interface after a Full Install.

Per default, grandMA3 onPC on Windows and macOS will be set to Auto.

The **Stations** view of this menu displays all the MA devices organized in types. The devices with the green background are connected devices.

This is a short description of the columns:

- **Lock:**
A **Yes** in this column indicates that the row is locked. The fields show "S" if they are unlocked.
- **No:**
This is the row number. Notice that a row can be a child of a parent object. If there is a number in parentheses, then it is the number of children in the parent object.
- **Name:**
This is the name of the device or device type.
- **Type:**
This shows the type for each device. OnPCs might be listed as "Undefined".
- **IP:**
This is the IP address for the MA-Net interface for the device.
- **Session:**
This is the name of the session the device is a part of.
- **Location:**
This is the location for the session. Both session name, location, and key have to match for devices to connect.
- **Show File:**
This is the name of the active show file in the device.
- **Status:**
This is the session status for the device.
- **Prio:**
This is the priority for the station.
- **Version Big:**
This is the software version for the device. The **Version Big** number indicates the version number that needs to match for stations and processing units to connect in a session.
- **Version Small:**
This is the software number that needs to match for grandMA xPortNodes to be able to connect to a session.



- **Enabled:**
This shows if the network connection is enabled.
- **Online Time:**
This is the time the device has been online in the network.
- **Session Index:**
Each session is automatically assigned a number. This is the session number for each device. There is a maximum of 32 sessions in the same network.
- **Session Slot:**
Each station in a session has a unique slot ID. This is the ID number for each device.
- **Remote IP:**
Some MA devices are connected to a station. For instance, the grandMA3 extension. This field shows the IP address of the station these devices are connected.
- **Mask:**
This is the Sub-Net mask of the station.

Session Maximums

There are some maximum numbers for a session. These describe some limits in a session that can be good to know.

There should only be one fully loaded session in a network domain. If more is needed, great care should be taken to ensure the network can manage the traffic without creating bottlenecks.

These numbers are for one session.

A session can control up to **262 144 parameters**. Learn more about this in the [Parameters topics](#). These topics also explain how the parameters can be expanded up to the limit.

MA Devices

All MA devices belong to a device category.

Category A - maximum **31 devices** in a session.

These are actively calculating stations and devices. This includes all grandMA3 consoles, grandMA3 replay units, grandMA3 onPC's, and grandMA3 processing units.

A grandMA3 onPC counts as one station, no matter how many wings there are connected. The grandMA3 onPC wings does not count as devices in a network.

Category B - maximum **64 devices** in a session.

These stations and devices are not calculating anything related to the output of the session. It is currently not defined when a station can be in a non-calculating mode.

Category C - maximum **128 devices** in a session.



These are MA devices that might use the calculated parameters and are in the session. It includes MA Nodes running in grandMA3 mode and MA extensions.



Important:

Network nodes that run in sACN or Art-Net mode do not count against the limitation.



10.2.1. Create a Session

Requirement:

Network connections need to be made, and the correct IP address should be set - For more information, see **Interfaces and IP.**

Create a Session with the User Interface

1. Open the Network menu.
2. Tap the buttons at the bottom that needs to be edited.
 - Make sure that the correct interface is selected.
 - Make sure the Session name and Location values are correct.
 - Make sure the correct Key is selected.
3. If the network connection is disabled (the network icon (📶) in the **Command Line** is red), then tap the big enable/disable button in the lower right corner of the Network menu.
 - The network icon should become green.



10.2.2. Join a Session

Requirement:

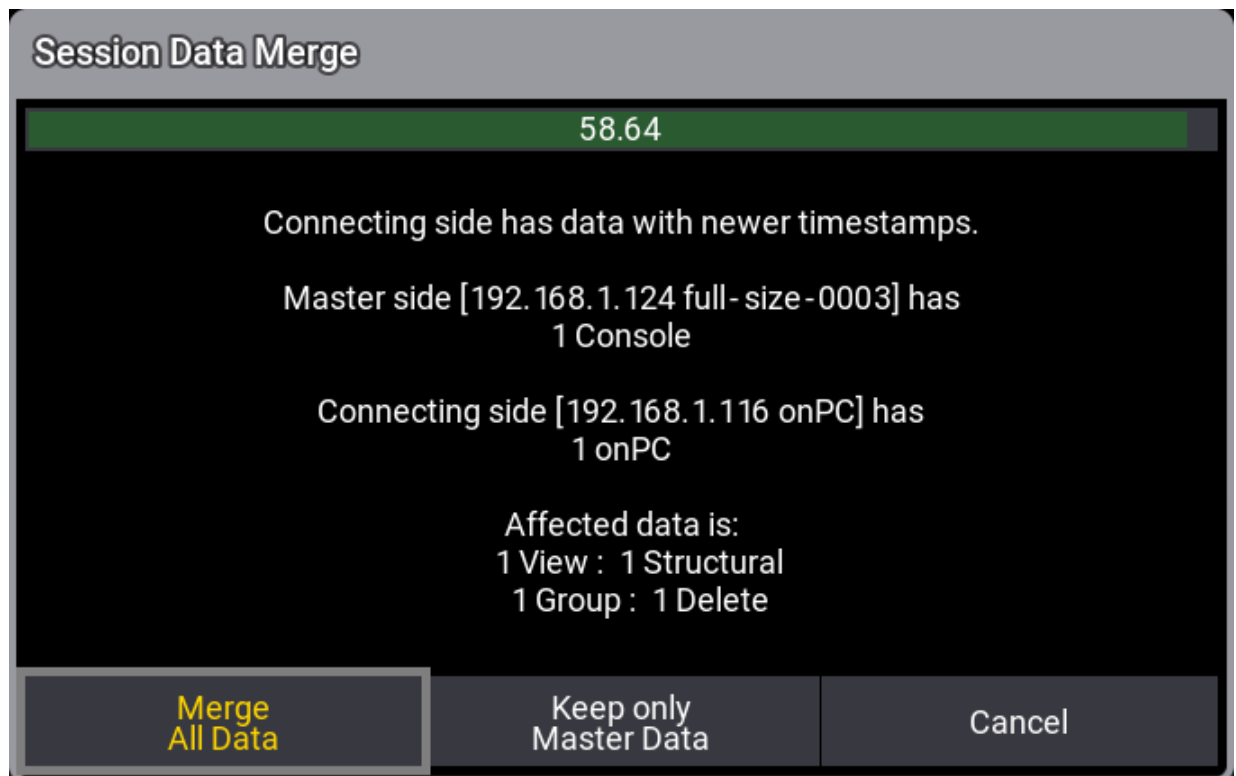
Network connections need to be made, and the correct IP address should be set - For more information, see [Interfaces and IP](#). There needs to be a running session that can be joined.

When a console or onPC station wants to join a session or gets invited into a session, the session recognizes that the station joining was already a part of the session at a previous point.

If this is not the case, the station joins the session and gets the show file.

If the same show file is loaded on both sides, the grandMA3 software can merge both show files. This is useful in undetected scenarios, such as a station being disconnected for a time from the rest of the session (maybe because of a defect in the network cable or other causes of failure). If on both sides (the remaining session and the disconnected station) users worked further on the show file, users have to decide to choose between one of the two show files. Meaning they had to reprogram the changes of the lost show file in the one that is kept. The **Merge All Data** option combines the data of both show files in one show file.

This is the small pop-up asking what to do:



Session Data Merge pop-up

The **Session Data Merge** pop-up is visible for 60 seconds. After 60 seconds without a response, the session data merge will be canceled.



To pause the countdown, tap in the text area of the pop-up. When a station, as described above, joins the session again, a pop-up informs the user about a Session Data Merge.

The pop-up lists the affected data. It is the elements that are different between the shows.

There are three options to choose from:

- **Merge All Data:**
Merges the changes of all connecting stations into one show file.
- **Keep only Master Data:**
The changes in the show file data of the connecting devices are ignored. The show file of the Master side is maintained. Before overriding the show file of the connecting devices, the show file will be saved automatically on the connecting stations.
- **Cancel:**
The connecting stations will not join the session of the Master side and be made Standalone. Then it can be investigated which show file is correct, back up each show file, etc.

Merging the data of two show files is supported for these object types at the moment:

- Sequences with Cue and Cue Part
- Presets
- Recipes
- Programmer and Programmer Parts
- Timecodes
- Macros, Plugins, and PluginComponents
- Pages and Executors
- Images and Videos
- DataPools
- Groups
- All other pools not explicitly mentioned here



Known Limitations:

- Changes made to the patch during the disconnection of a station disables the possibility of merging.
- Merging the deletion of parts of objects during the disconnection of a station is not fully working yet. (for example, deleting one macro line or deleting a recipe line)
- Merging the movement of objects during the disconnection of a station creates duplicates: At the original spot and the new spot. This may result, for example, in having different sequences assigned to the same executor on different stations.

Join a Session Using the User Interface

1. Open the Network menu.



2. Tap the buttons at the bottom that needs to be edited:
 - Make sure that the correct interface is selected.
 - Make sure the correct Key is selected.
3. Tap the station with the desired running session.
4. Tap **Join Session**.
 - The network icon and the stations should become green.
5. If the **Session Data Collision pop-up** appears, then select the desired option.

Join a Session Using the Command Line

1. Use the **JoinSession keyword** to join existing or create a new session.



Restriction:

It is only possible to join a session if the joining station got the same key as the currently active key on the selected station (step 3 above) currently in the session.

This means that the station that wants to join a session selects a station already in the session. The station already in the session got a selected key. The joining station needs to have the same key selected to be able to join the session.



10.2.3. Leave a Session

Requirement:

The station needs to be a session member to be able to leave the session.

Leave a Session Using the User Interface

1. Open the **Network menu**.
2. Tap **Leave Session** or turn the network Off using the network connection button.

Leave a Session Using the Command Line

- Use the **LeaveSession keyword** to leave the session.



Important:

When leaving the session during an active show download, the station will fall back to the previously loaded show file that was loaded before the show download started.

If there are other stations in the session with a priority above never, then the session is still active.

10.2.4. Invite a Station into a Session

Stations can be invited into a session if **EnableInvite** is turned On in the **Network Settings** on the station being invited.



Important:

If the priority of the joining station is higher than the current session master, the show file from the joining station will be used instead of the show file used in the session before the invite.

Requirement:

A session needs to be running, and the station being operated needs to be part of the session. See how in the **Create a Session topic**.

Using the User Interface

Stations are located and invited using the **Network menu**.

1. Navigate to the Network menu - see how, using the link above.
2. Locate and tap the station in the list of available stations (stations in the network but not connected to a session).
3. Tap **Invite Station** at the bottom of the Network menu.

Using the Command Line

Stations are invited using the **Invite keyword**.



Restriction:

Inviting a station into a session is only possible if the joining station has the same key in the key registry as the one currently selected on the inviting station when the **Invite** button is tapped.



10.2.5. Dismiss Stations from a Session

Stations that are part of a session, can be dismissed by another station.

Requirement:

A session needs to be running and the station being operated needs to be part of the session. The station to be dismissed also needs to be in the same session. See how to create sessions in the [Create a Session topic](#).

Using the User Interface

Stations are located and dismissed using the [Network menu](#).

1. Navigate to the Network menu - see how using the link above.
2. Locate and tap the station in the list of stations that are connected to the session (stations with a green background color).
3. Tap **Dismiss Station** at the bottom of the Network menu.

The selected station is now kicked out of the session.

Using the Command Line

Stations are dismissed using the [Dismiss keyword](#).

10.2.6. Create a Custom Key

Keys are passwords for sessions.

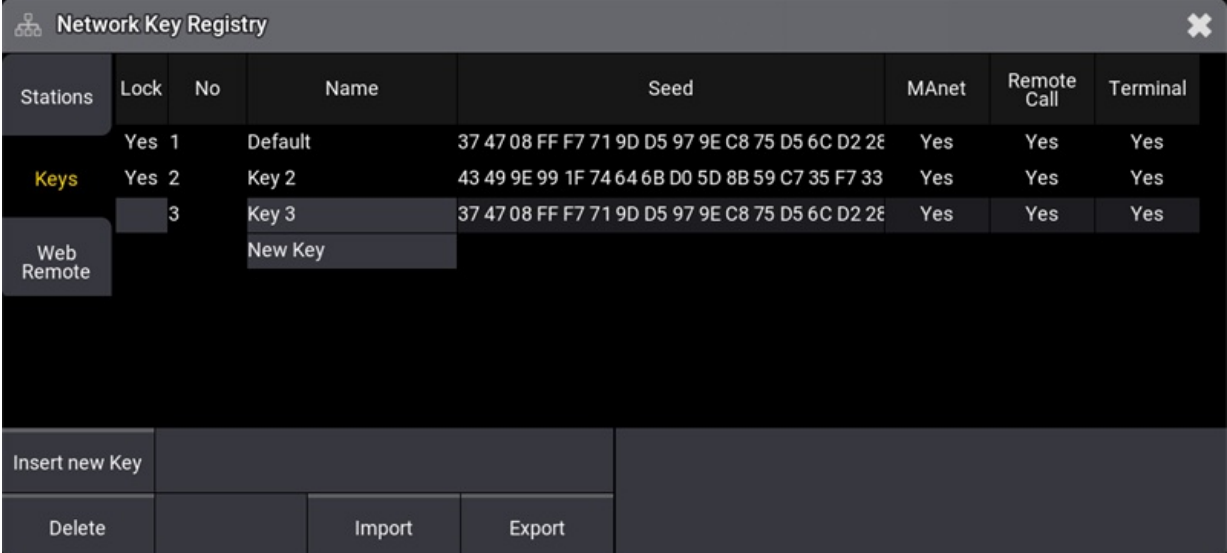
The same default key is a part of every new show file. It is used unless a custom key is created and used instead.

The key is a long hexadecimal hash value generated by a seed word or a set of characters.

Keys can be Inserted/Created, Deleted, Imported, and Exported. Locked keys cannot be edited or deleted.

Keys are used when a **session is created** and when stations are **invited**.

All the described key operations are done from the **Network Key Registry** menu.



Stations	Lock	No	Name	Seed	MAnet	Remote Call	Terminal
	Yes	1	Default	37 47 08 FF F7 71 9D D5 97 9E C8 75 D5 6C D2 2E	Yes	Yes	Yes
Keys	Yes	2	Key 2	43 49 9E 99 1F 74 64 6B D0 5D 8B 59 C7 35 F7 33	Yes	Yes	Yes
		3	Key 3	37 47 08 FF F7 71 9D D5 97 9E C8 75 D5 6C D2 2E	Yes	Yes	Yes
Web Remote			New Key				

Buttons: Insert new Key, Delete, Import, Export

Open the Network Key Registry from the Network menu

Open the Network Key Registry Menu

1. Press the **Menu** key.
2. Tap the **Network** button.
3. Tap the **Keys** button on the left menu.

This opens a menu like the one in the image above.

There are a few columns in the menu:

- **Lock:**
This column indicates if the key is locked. The default key is locked as a default. When a seed is changed, the key is automatically locked. Edit this field to unlock a key.
- **No:**
This is the key number.



- **Name:**
This is a user-friendly name for the key. The name can be edited if the key is unlocked.
- **Seed:**
The seed is a hash value. This can be edited on unlocked keys. Any text input can be a seed to create a hash value. The hash value cannot be manually input as this would create a new hash using the input as a seed. The original seed input needs to be known to be able to recreate the hash manually. The default value for a new key is the same seed value as the default key.
- **MAnet:**
When this is set to **Yes**, the key gives access to the MAnet area. This is, for instance, the possibility of joining a session.
- **Remote Call:**
When this is set to **Yes**, the key gives access to the Remote Call area, which is, for instance, the possibility to use the web remote into a station, network updating the software, **RemoteCommand**, and other commands that affect other stations (for instance, reboot and shut down of other devices).
- **Terminal:**
When this is set to **Yes**, the key gives access to the Terminal area. This is, for instance, access to a station's terminal interface.



Important:

Please leave the **MAnet**, **Remote Call**, and **Terminal** columns set to "Yes" for the default key. Changing these might give unexpected results.

Using Keys in Sessions

The key is selected before the session is started. If a custom key is selected, other stations need to have the key in the registry list to join or be invited into a session.

The keys can be exported to a USB memory stick and imported into other stations.

Or, if the seed is known, the custom key can be recreated on other stations using the same seed value.

If a station does not have the key used when the session was created, it cannot be a part of it. The joining station does not need to have the key selected in the **Network Menu** but must be in the **Network Key Registry** list.

Requirement for the next few operations:

The **Network Key Registry** menu is opened.

Create a new Key

Keys are created like most new elements.

1. Select the row where the key should be inserted.
2. Tap **Insert**.
3. (Optional) Edit the **Name** field to give the key a useful name.

4. Edit the **Seed** field and type a word or a random set of characters. This input is converted to a hash number.

Notice that the row is locked as soon as the seed field changes. It can be unlocked and edited afterward. The lock is to prevent accidental change of the key.



Hint:

The key can be reproduced on other stations if the same seed text is applied.

Delete a Key

A locked key cannot be deleted. Follow these steps to delete a key.

1. Select the row with the key to be deleted.
2. Edit the **Lock** field to unlock the row - the field should be empty for unlocked.
3. Tap **Delete**.

The key is now deleted.

Export a Key

Keys can be exported and brought to other stations manually.

1. Select the row with the key to be exported.
2. Tap **Export**.
3. Select the desired drive in the pop-ups title bar.
4. Give the exported file a name.
5. Tap **Export** in the pop-up.

The key is now exported to the selected drive.

Import a Key

Keys can be imported into a station.

1. Select the "New Key" row.
2. Tap **Import**.
3. Select the desired drive in the pop-ups title bar.
4. Select the desired file.
5. Tap **Import** in the pop-up.

The key is now imported to the station.



10.2.7. Session Master Selection

Some rules define what station becomes the master of a session.

A key element to this is the stations' priorities.

Inviting Stations into a Session

A session always has a master. If there is only one station in the session, it is **IdleMaster**. If there are more stations, it is **GlobalMaster**.

Adding another station to the session can have different outcomes:

- The joining station has a **lower priority**:
The station joins the session and downloads the showfile. The station's status becomes **Connected**.
- The joining station has the **same priority**:
The station joins the session and downloads the showfile. The station's status becomes **Connected**.
- The joining station has a **higher priority**:
The station joins the session, takes over the **GlobalMaster** status, and uploads its show to all connected stations. The show is replaced.

Session Master Selection Rules

A set of rules takes effect in a situation where a new master needs to be selected among the remaining stations.

This situation can appear if the current session master disappears. For instance, if the network connection disappears.

It also happens when several stations booting up were previously part of a session.

Three rules define how the master is selected. They are in prioritized order:

1. The station with the highest priority wins. If there is more than one station with the same priority, go to rule two.
2. The stations' online time defines the master. The highest online time (+/- 5 seconds) wins. If more than one station has the same online time (+/- 5 seconds), go to rule three.
3. The station with the lowest IP address becomes the master.

The online time can be seen in the **Network Menu**.

10.3. Web Remote

The stations can be remote-controlled by any modern browser connected to the system. This includes Wi-Fi-connected devices.

A maximum of five devices can connect to one station simultaneously.

Each remote device connects to a specific station. The remote browser shows the same controls and encoder view at the bottom of the display as the grandMA3 Compact, grandMA3 Compact XT, and grandMA3 onPC display 1. The user-definable area is the same as screen 1 on any console or onPC. The web remote will try to log in as the user **Remote** when connecting, and the view and windows on screen 1 follow the logged-in user. The web remote can log in as a different user than the one currently logged in on the connected station. Learn more about multiuser setups in the [Single User and Multi User Systems section](#).

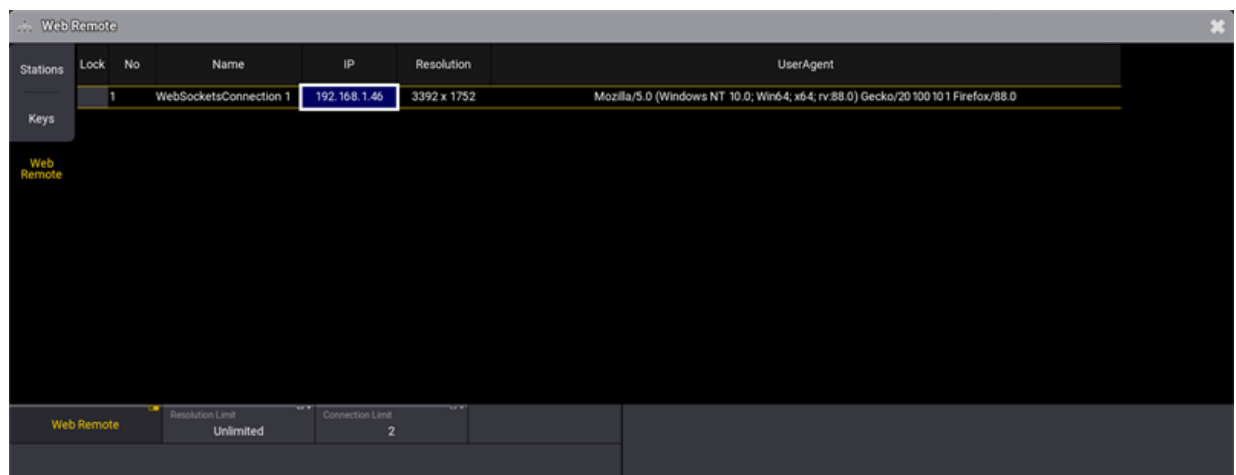
Any window or view that can be created on a console or onPC can be used in the remote. The system will try to provide a high refresh rate of the content, but it is affected by the connection speed and the resolution.

Connect with a Web Remote

Web remotes need to be enabled in the station. This can be done in the **Network Menu** - see the [Session topic](#) to learn how to access this menu.

The network menu has a toggle button that turns the web remote On or Off.

The same setting is mirrored in the actual **Web Remote** menu that can be accessed by tapping **Web Remote** on the left side of the network menu.




Web Remote menu with one connection

1. Make sure web remote is turned On in the station.
2. Open a browser on a network-connected device.
3. Type the IP address of the station and include **:8080** after the address to specify the port number.
The result should be a browser window with a grandMA3 display.

Now there is access to remote control the station with the user rights provided by the logged-in user.

A new show file has a user called **Remote**. This uses the default user profile but with a different screen configuration. Remote is also a keyword, so if it is desired to log in as this user, type the following in the command line:

```
MA User name[Fixture]>Login "Remote"
```

 **Hint:**
Putting the user name in quotation marks is essential for the software to treat it as a name, not as a keyword.

The Web Remote menu shows the connected devices. It is just information about the IP address, operating system, and browser.

Three different settings can be changed in the menu.

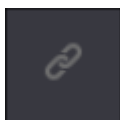
The remote connection can be turned On or Off.

A **Resolution Limit** can be set. There are options to limit it to 480p, 720p, 1080p, or unlimited. This limit is the same for every device that connects. If unlimited is selected, the resolution is adopted to the device screen.

Connection Limit can be used to set a number between 1 and 5. This is the maximum amount of simultaneous connections allowed.

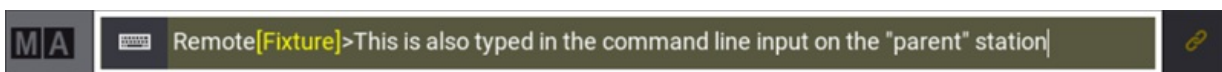
Linked Command Lines

A button on the connected remote device is next to the host command line input.



This can be activated to link the remote command line input to the host command line input.

Linked command lines change the background color on both the remote and the host station. It could look like this:



Linked command line example

Notice that the command input in the image above is not a valid command. It is just an example.



Any command input on any linked command line is immediately shown on all the linked command lines where the users are logged in with the same user profile. Any executed commands are executed on the host station.



10.4. Network Design

Creating a good working network for a grandMA3 system can be complex when the system expands.

Small systems might not require expensive switches and infrastructure to work, but pushing the grandMA3 system also pushes the network infrastructure.

General Control System Requirements

- The output of an MA Lighting control system only changes by intention (of the user).
- The output of an MA Lighting control system complies with the standards, proprietary open standards, and protocols (see a complete list in the **Regulations and Standards topic**):
 - DMX512-A.
 - sACN.
 - SMPTE Timecode.
 - Open Sound Control (OSC).
 - DIN SPEC 15800 (GDTF).
 - Art-Net.
 - PSN.
 - MVR.
- The output of an MA Lighting control system is synchronized for all physical output ports over all attached MA devices and all transmitted protocols.
- The output of an MA Lighting control system is considered to be “real-time” (The term “real-time” can be defined by: All latencies between input events and output reactions are determined and known.)
- Show files are upwards compatible, and exceptions shall be announced. (This might not always be possible due to rare technical reasons. If so, MA will state this with the release notes of the software update as ‘known limitations’!)

The list above defines a set of rules that must be true within a correctly configured network.

Network Speed

The amount of data sent in a grandMA3 network can vary depending on the number of grandMA3 devices and parameters.

A grandMA3 system is always defined based on one (1) maximum utilized session in a network. This maximum is defined in the **Session topic**.

With this in mind, a network speed of 1 Gbit/s is required for the MA-Net3 protocol. This means that the ports on the switches should be configured as “Access” ports and have a minimum capacity of 1 Gbit/s.

The maximum latency on the “broadcast domain” should be 2 milliseconds or less.

The MA-Net3 protocol uses multicast, and all grandMA3 devices support IGMP membership requests.



All devices should use the same bandwidth capacity. This means that if there is an older network node translating a DMX streaming protocol (like Art-Net) only capable of 100 Mbit/s speed, this protocol should be separated to a different VLAN (or network) limited to this speed.

In a Gigabit network, the average bandwidth reserved for MA-Net3 is 200 Mbit/s. MA-Net3 does not allow for traffic shaping.

Cable Quality and Lengths

Since the network speed is defined as 1 Gbit/s, the quality of the copper cabling should be Cat.5e or better.

The cable length is defined in a set of standards for fixed installations (EN 50173). These specify a set of rules for installations that should be followed.

For "flying installations" or "one-offs", a different type of cable is used (stranded core / "patch" cables). This type does not support the same lengths as a proper fixed installation (using solid core cables). Using a maximum length of 75 meters / 246 feet between devices is recommended for these systems. This length is tested by MA Lighting using new quality cables. If the cable is damaged, it should not be used.

These lengths are about the distance between devices, such as a grandMA3 console and the switch.

Connecting switches can have higher demands. The speed between the switches might be higher than 1 Gbit/s and thus have higher demands.

Switch Settings

If more than one switch is used, and the switches are used for more than MA-Net3, then it is recommended to separate the MA-Net3 into a different VLAN.

Please consider that the MA-Net3 should have 1 Gbit/s available, so if the switches do other things as well, there needs to be more than 1 Gbit/s connection between the switches.

When using multiple switches, please ensure the switches support Protocol Independent Multicast (PIM) to route the multicast correctly between the switches.

Switches should have all ports go online simultaneously after a reboot. Delayed port-by-port reboot behavior is not supported at this time.

The IP addresses used in a MA system should follow the recommendation RFC-1918 about address allocation in private internets.

If the MA Worldserver is desired, then a VLAN (or separate network) should be used. This network segment should be routed to the Internet.



Bandwidth Calculation for Best Practise Networks

The DMX output of MA devices needs a refresh rate of 30Hz. Therefore all “real-time” data needs to be present within 33ms.

To avoid any visible jitter, the synchronization data packet is allowed to have a maximum latency over the whole system of 2ms.

Within each refresh of 33ms, all necessary data must be transferred to the end devices.

The maximum possible will be 262 144 parameters with 24-bit DMX data. Only a mixed calculation is possible with a 1 024 DMX universe limit. The maximum for 262 144 parameters each of 16bit will end up in 1024 DMX universes and therefore:

$$1\ 024\ \text{Universes} * 512\ \text{packets} * 10\ \text{bit} * 30\ \text{Hz} = 158\ \text{MBit/s}$$

This will be the average bandwidth needed to allow the “real-time” traffic (as described in the requirements above) to pass through. At the same time, the latency cannot be more than 2ms. This also explains the QoS definitions. This is a worst-case scenario with the maximum possible DMX data.

There is more data to be transmitted than just the DMX data; therefore, the average bandwidth for Gigabit systems is set to 200 MBit/s.

Higher bandwidth for faster show upload might be required, especially in bigger installations.

More bandwidth is needed if additional DMX-over-Ethernet protocols are used in parallel and/or Web-Remote(s).

The calculation will be the same, and the bandwidth of additional DMX-over-Ethernet data must be added.

Testing the Network Environment

A grandMA3 network system will be recognized as a smooth operating system if there are no data retransmissions and no DMX dropouts on any device in the session.

Any retransmission of data packages in the overall system will be shown in the System Monitor of every device within the session. The Session stability and reliability is a measure of network quality and vice versa.

Retransmission means a data package has not reached its destination in order or at all. The question arises of which retransmission count is acceptable. In other words, when to worry about network stability.

The answer cannot be a quantity – the answer is a situation depending measure.

Here are some guidelines for quality insurance of the network session handling:

- A few Retransmissions (0 – 50) on a regular or irregular basis – The system is considered stable.
- Multiple Retransmissions (> 50) on an irregular basis – The system has issues.
- Multiple Retransmissions (>100) on a regular basis – The system needs investigations!



- High number of Retransmissions (> 200) on an irregular basis – The system needs investigations!



Important:

During a show upload, the number of retransmissions can be very high due to the heavy use of communication on all layers, and retransmissions are expected. The system is considered to be stable.

DMX dropouts are not allowed within the session at any time.

10.4.1. Protocol Details

The grandMA3 system supports different network protocols.



Hint:

See a list of protocols in the **Regulations and Standards topic**.

The following are some details about each protocol that might be useful for setting up rules in network switches and routers.

MA-Net3

General Traffic

Transport layer: UDP - Multicast

Port number: 30020

IP addresses:

- 236.4.1.0 - Administration group for all devices and sessions
- 236.4.1.1 - General communication (session data)
- 236.4.1.2 - Extended communication (session data)
- 236.4.1.3 - Extended DMX (session data)
- 236.4.1.4 - Reserved (session data)

Additional Information:

The IP addresses mentioned above belonging to the session are for session index number 1. Subsequent session indexes use different addresses using the formula "IP+4*X" where "X" is the session index number, and "IP" is the address mentioned above. This only changes the last octet of the address.

So the IP addresses specific for session index 3 are:

- 239.4.1.13
- 239.4.1.14
- 239.4.1.15
- 239.4.1.16

This allows for 32 sessions in a network.



Important:

According to RFC5771 (IANA Guidelines for IPv4 Multicast Address Assignments), the used Multicast addresses are within an IANA reserved block.

MA Lighting is currently investigating how to solve the issue.

For the current version of grandMA3 software, the exception needs to be made to accept the Multicast addresses in the local area network.

Alternate Traffic

Transport layer: TCP - Unicast

Port number: 30022+

Additional Information:

This is used when a station needs to send a lot of information to a different station, for instance, with a network update.

MA Worldserver

Transport layer: TCP - Unicast

Port number: 30021

Additional Information:

This must be routed to the Internet to access the official worldserver from MA Lighting.

MA Web-Remote - HTTP

Transport layer: TCP - Unicast

Port number: 80

MA Web-Remote - Websocket

Transport layer: TCP - Unicast

Port number: 8080

Art-Net (revision 1.4db)

Transport layer: UDP - Broadcast

Port number: 6454

and

Transport layer: TCP - Unicast



Port number: 6454

Additional Information:

In Auto-Mode, grandMA3 Art-Net will be sent as Unicast unless there are more than five receivers (default value in Broadcast Threshold) of the same DMX universe in the network. Then the protocol switches automatically to Broadcast.

The default Broadcast Threshold is set to five but can be changed in the **Art-Net menu**.

sACN

Transport layer: UDP - Multicast

Port number: 5568

IP addresses: 239.255.x.y

and

Transport layer: TCP - Unicast

Port number: 5568

Additional Information:

The default uses a Multicast address for each sACN DMX universe corresponding to a specific universe. The first two numbers in the IPv4 address are fixed. The two following numbers are calculated to match the universe number.

For instance, sACN universe 1 is on IP 239.255.0.1, and sACN universe 256 is on IP 239.255.1.0

PSN - **PosiStageNet**

Transport layer: UDP - Multicast

Port number: 65565

IP addresses: 236.10.10.10

Additional Information:

The Multicast address and port number can be changed in the **PSN menu**.

OSC - **OpenSoundControl**

Transport layer: TCP / UDP

Port number: 8000



Additional Information:

The transport layer protocol, address, and port number can be changed in the [OSC menu](#).

NDI - Network Device Interface

mDNS communication

Transport layer: UDP - Multicast

Port number: 5353

IP addresses: 224.0.0.251

Alternate Discovery

Transport layer: TCP - Unicast

Port number: 5959

Base TCP Connection for Stream

Transport layer: TCP - Unicast

Port number: 5961+

UDP Shared with TCP Connections

Transport layer: UDP - Multicast

Port number: 5960+

UDP / TCP Receiving

Transport layer: UDP / TCP

Port number: 6960+

UDP / TCP Sending

Transport layer: UDP / TCP

Port number: 7960+

SFTP

Transport layer: TCP - Unicast

Port number: 22



SNMP

Transport layer: UDP - Multicast

Port number: 161 and 162

Additional Information:

SNMP v1 and v2c. Public community.

AVAHI

Transport layer: UDP - Multicast

Port number: 5353



10.4.2. Wireless Networks WLAN - WiFi

A wireless connection should only be used with the web remote.

Do not try to run a session on a WiFi connection. Please read the **Network Design topic** and understand why it is not possible.

Wireless networks (WiFi) manipulate the data packages to optimize the wireless network. This works for standard HTTP and HTTPS traffic but not for MA-Net3.

Please use proper access points for the wireless network.

The access points shall be connected to a port on a network switch, and all access points should be in a separate VLAN. This ensures that all multicast traffic can be filtered out of the wireless network.

The access points must be filtered to only allow MA Web Remote traffic “on air” to avoid delays caused by too much traffic between the wireless device used for MA Web Remote and the rest of the MA System. This filtering could be done in the switch or the access points.

In the software, the maximum resolution transmitted for the Web Remote can be set (and limited) to allow for better performance in a slow wireless network - see the **Web Remote topic**.



10.5. Regulations and Standards

The grandMA3 system conforms to and suggests a series of network-related regulations and standards.

- ISO/IEC 11801:2002 - International standard for structured data cabling
- EN 50173 - European standard for structure data cabling
- ANSI/TIA-568 - Standard for commercial building cabling products and services
- ANSI/IEEE 802.3 IEEE - Standard for Information Technology (Cabling)
- ANSI/IEEE 802.11 IEEE - Standard for Information Technology (WiFi)
- RFC 1918 - Address Allocation for Private Internets
- RFC 2236 - Internet Group Management Protocol
- RFC 4594 - Configuration Guidelines for DiffDerv Service Classes
- ANSI E1.11 - 2008 (R2018) Entertainment Technology USITT DMX512-A Asynchronous Serial Digital Data Transmission Standard for Controlling Lighting Equipment and Accessories (DMX)
- ANSI E1.31 - 2018 Entertainment Technology Lightweight streaming protocol for transport of DMX512 using ACN (sACN)
- ANSI E1.20 - 2010 Entertainment Technology RDM Remote Device Management Over DMX512 Networks (RDM)
- Art-Net 4 - revision 1.4db
- PSN 2.03
- NDI 4
- DIN SPEC 15800:2022 – Entertainment Technology - General Device Type Format (GDTF)

Some of the protocols have some more details in the [Protocol Details topic](#).



11. DMX In and Out

There are several ways to get DMX in and out of the grandMA3 system.

DMX ports on stations can generate DMX as a standalone device.

DMX ports on grandMA3 devices part of a session can also be DMX outputs and inputs.

This is set up from the **Output Configuration** menu. Please read about it in the **DMX Port Configuration topic**.

DMX can also be transferred using standard network protocols. This is **Ethernet DMX**.

There are different rules for the different kinds of DMX.

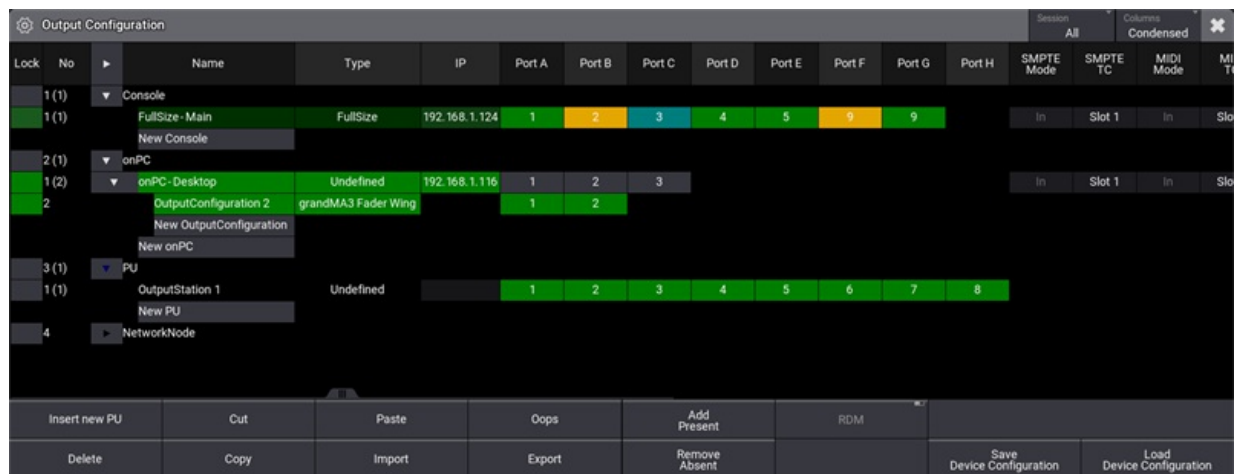
Generally, each DMX port on a grandMA3 device can be an input or an output.



11.1. DMX Port Configuration

The DMX outputs and inputs are configured in the **Output Configuration** menu. They are labeled from **A** and alphabetically upwards.

There are also columns for SMPTE, MIDI, DC remote inputs, and Ethernet interfaces, but these elements are not described in this topic. Please read about them in the [Timecode section](#), [Remote In and Out topic](#), and [Ethernet DMX](#).



Output configuration menu with console onPC and PU

Access the menu by pressing the **Menu** key and then tap **Connector Configuration**.

-OR-

Use the command line to open the menu:



Hint:
See [Ethernet DMX](#) for how to send and receive DMX using Ethernet.

The menu lists grandMA3 devices. There is a button in the title bar called **Session**. This filters the list to show **All** devices in the network or only the ones **In Session**.

The devices are organized in Console, onPC, PU, and NetworkNode sections that can be unfolded to see the devices in each category. Real hardware devices are matched to devices in this list by IP addresses.

The menu has different modes for displaying columns. There is a button in the title bar called **Columns**. This has three modes:

- **Full:**
This shows all columns.
- **Condensed:**
This mode condenses the XLR columns to one for each port and displays all other columns.
- **XLR Only:**
This only shows the XLR ports in a condensed version.



Hint:

Read the [What are Timecode Slots topic](#) to learn about the SMPTE and MIDI settings.
Read the [DC Remotes topic](#) to learn about the DC Start setting.
Read the [Ethernet DMX topics](#) to learn about the Art-Net and sACN settings.
Read the [OSC topic](#) and the [PSN topic](#) to learn about these settings.

DMX Ports

Each DMX port has four different settings (only visible when in **Full** mode):

Mode:

Each port can have four different modes:

- **Off** (gray background color in the cell):
The port is turned Off.
- **Out** (green background color in the cell):
DMX is sent out without RDM traffic. The universe is defined in the XLR field. This is the default mode.
- **RDM** (dark sea green background color in the cell):
DMX is sent out, and RDM is active. The universe is defined in the XLR field.
- **In** (yellow background color in the cell):
DMX is received and merged into the universe specified by the XLR number. The merge uses the priority defined in Prio.

XLR:

This is a number that defines what universe the port relates to. Fixtures can be patched in universes 1 to 1 024.

Merge:

This option is only visible when the port mode is **In**.

The options are:

- **Off:**
The incoming DMX is not merged.
- **Prio:**
The priority set in the **Prio** column is used. The highest priority wins.

- **HTP:**
The highest DMX value is used.
- **LowTP:**
The lowest DMX value is used.

Prio (Priority):

The priority is used for merging DMX inputs. It is only used and can only be edited when the port mode is **In**, and the merge is set to **Prio**.

Editing this field opens a small select pop-up with the options.

The options are (from highest priority to lowest):

- **Super:**
This priority is the LTP priority above any other playbacks and even above the programmer.
- **Prog:**
The Programmer priority is between Super and the other priorities.
- **Highest:**
Highest LTP priority - like LTP but higher than both LTP and High.
- **High:**
High LTP priority - like LTP but a higher priority than normal LTP.
- **LTP (Latest Takes Precedence):**
This is the normal LTP priority. The newest attribute value is prioritized over the old value.
- **Low:**
Low LTP - This is a lower LTP priority.
- **Lowest:**
Lowest LTP - This is a lower priority than both LTP and Low.



Important:

The port configuration is stored in each device where it is set.

Import and Export of Entire Device Configuration

The device configuration for all the devices in the list can be stored in a single configuration file. This configuration can then be loaded again to restore the settings based on the file. This makes storing and loading configurations for different shows or setups easy.



Hint:

The device configuration file includes the Output Configuration and the DMX Protocol (**Ethernet DMX** protocols) settings.



Save Device Configuration

1. Make sure the configuration is as desired.
2. Tap **Save Device Configuration**. This opens the **Save Device Configuration as:** pop-up.
3. Type a name for the configuration and ensure the desired drive is selected in the title bar of the pop-up.
4. Tap **Save**.

Load Device Configuration

1. Tap **Load Device Configuration**. This opens the **Load Device Configuration from:** pop-up.
2. Select the desired configuration file.
3. Tap **Load**.

Import and Export of Output Configuration for Single Devices

Each device keeps its own output configuration. The configuration of single devices can be exported to a file and then imported again.

Export an Output Configuration

1. Select the devices that need to be exported.
2. Tap the **Export** button -> a file browser opens.
3. Select the desired drive.
4. Give the file a name and tap enter/please.

A file can contain the configuration for one or multiple devices. It is just a matter of selecting one or several devices before the export.

Import an Output Configuration

1. Select the desired devices.
2. Tap the **Import** button -> a file browser opens.
3. Select the desired drive.
4. Select the files that have a configuration for the selected devices.

Add and Remove Devices

The list of devices is automatically updated with new devices in the network.

Tapping **Add Present** updates the list with new devices.

Tapping **Remove Absent** removes devices from the list that are no longer in the network.

Devices can be added without being present in the network. Unfold the desired device type in the list and tap the **New [device type]** followed by **Insert new [device type]**. The grandMA3 onPC's can be expanded further by adding up to two grandMA3 Fader Wings to each grandMA3 onPC.

Show Connectors

On the light and full-size consoles, there is a button called **Show Connectors**. This can be toggled On and shows a connector overlay on the letterbox screens. This indicates where the different connectors are located on the back of the console. The XLR ports are labeled DMX A to DMX G in this overlay.



Connector overlay on the right letterbox screen (grandMA3 full-size model)

The overlay can also be toggled using the following command:

```
MA User name[Fixture]> Menu "ConnectorView"
```

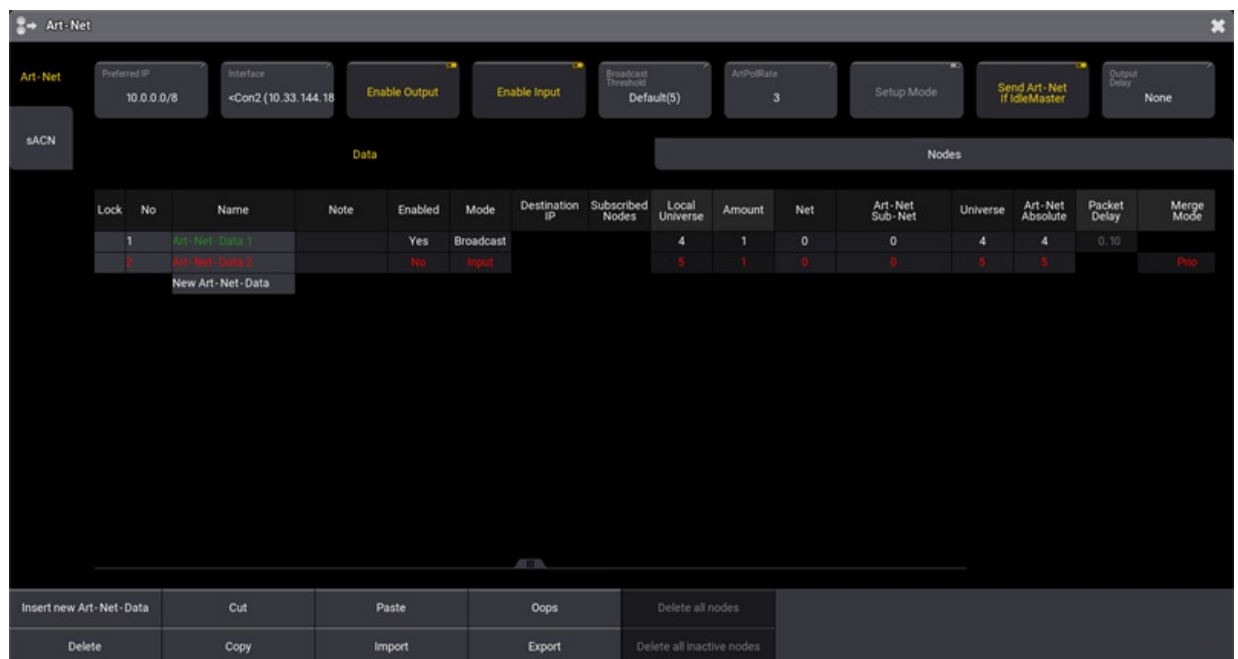


11.2. Ethernet DMX

There are currently two supported protocols for sending and receiving DMX using the network: sACN and Art-Net.

It is the session master station that outputs the network DMX. The visual feedback regarding sending or receiving network DMX can look different on the stations in the session, so it should be viewed and changed on the master station to see the actual output/input status.

Both are set up and changed in the **DMX Protocols** menu:



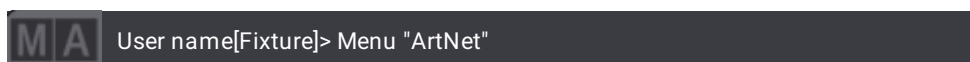
DMX protocol configuration for Art-Net

Access the menu by pressing **Menu** and then tap **DMX Protocols**.

Tapping the **DMX Protocols** button opens the menu for the first protocol. In the example above, it is the Art-Net menu.

The menus can also be opened using the command line:

For Art-Net:



For sACN:





On the left side menu, there are two buttons, one for each menu.

See the subtopics for information about the two protocols.

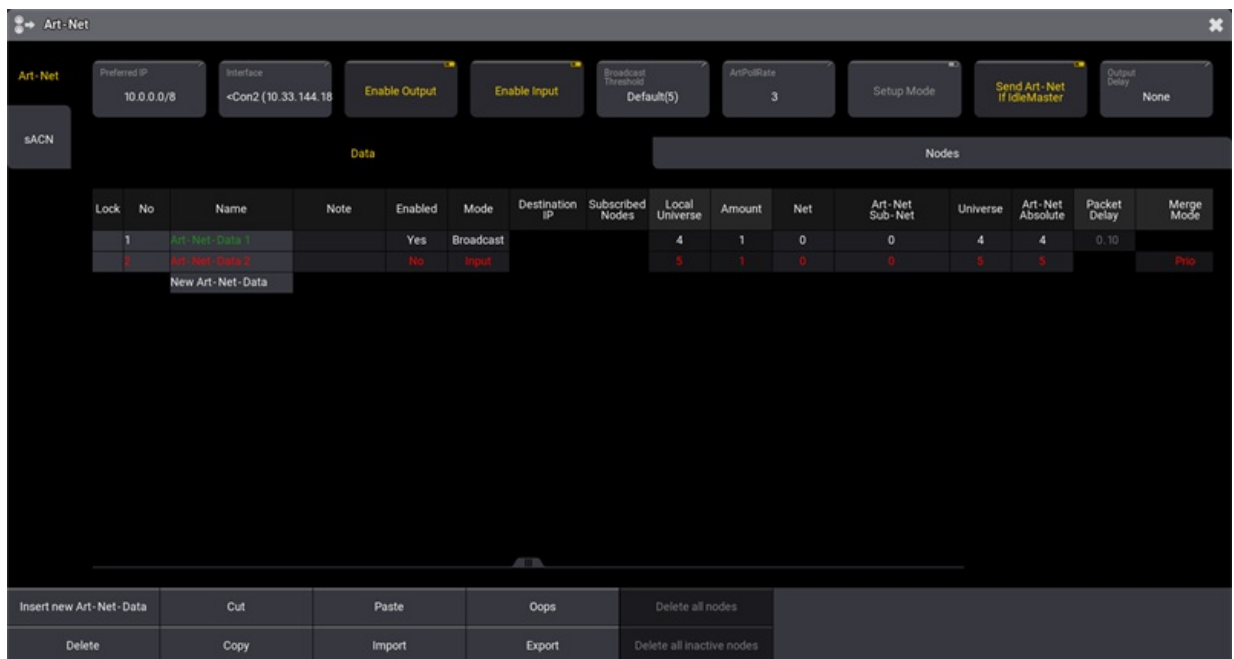


11.2.1. Art-Net Menu

Art-Net is a royalty-free protocol developed by Artistic Licence (<http://www.artisticlicence.com>).

grandMA3 supports Art-Net 4.

It is configured in the Art-Net menu:



Art-Net configuration menu

See the [Ethernet DMX topic](#) for information on how to open this menu.

See the [Transmit DMX using Art-Net topic](#) for an explanation of the steps needed to transmit Art-Net.

Config Buttons

There are some buttons at the top of the menu:

- **Preferred IP:**

This is the preferred IP address or address range used by the Art-Net protocol. The Interface can be set to **Auto**, allowing this setting to select the interface that matches the preferred setting. Tap this input button to open a small number input pop-up. The preferred number has to be input as an IP address with the subnet mask written in CIDR notation. For instance, 10.0.0.0/8 tells the system that it prefers an IP address that starts with 10. The rest of the numbers do not matter (the "/8" is the same as subnet mask 255.0.0.0). This setting is individual for each station.



- **Interface:**

Tap this button to open the Select Interface pop-up to select the desired network interface. This interface will be used for all Art-Net in and out. The **Auto** option can be used to allow the **Preferred IP** setting to select the interface. The interface is written inside "<" and ">" when it is selected by the preferred setting. This setting is individual for each station.
- **Enable Output:**

This On/Off button must be On for Art-Net to be transmitted. Data also needs to be configured for output - read more below.
- **Enable Input:**

This On/Off button must be On for Art-Net to be received. Data also needs to be configured for input - read more below.
- **Broadcast Threshold:**

This input button sets the amount of Art-Net receivers for a universe before the master starts to send the universe as broadcast. This is only valid if the mode is set to **Auto**. If the number of receivers is below the threshold, the universe is sent as Unicast. If it is above, then the universe is sent as Broadcast.
- **ArtPollRate:**

This input sets the time between each ArtPollRequest packet sent by the master station.
- **Setup Mode:**

This On/Off button is used to toggle the setup mode. This mode can be used to transmit and receive configuration data only. If the output and input are turned Off, only the configuration data is transmitted and received.
- **Send Art-Net If IdleMaster:**

This On/Off button defines if the station transmits Art-Net data when it is Idle Master. In a session, it is the Global master who transmits the network DMX. If the station is not in a session with other devices, it is Idle Master. Turning this setting On will make the station output network DMX when it is Idle Master. This must be On if a single station is to output networked DMX. Learn more about standalone devices and networked devices in the [System Overview section](#). This setting is individual for each station.
- **Output Delay:**

This can set an output delay between 0ms and 30ms. This delays the entire Art-Net output compared to the outputs coming from MA-Net devices.



Important:

- Enabling Setup Mode allows Art-Net configuration data to be sent and received even when **Enable Output** and **Enable Input** is Off.
- When Setup Mode is Off and output is enabled, DMX and configuration data are transmitted.
- When Setup Mode is Off and input is enabled, DMX and configuration data are received.

There are two tabs below the buttons. They are **Data** and **Nodes**. Data is used to set up output and input. Nodes can be used to see the discovered nodes in the network. If the nodes support it, this can also be used to configure the ports on the node.



Data Tab

The data tab is a grid of rows and columns. Each row is an Art-Net configuration. The text is red if the row is invalid or not enabled. The name text flashes green when Art-Net is transmitted. It flashes dark yellow when an Art-Net input signal is received.

This is a short description of the columns:

- **Lock:**
The row can be locked to prevent changes.
- **No:**
This is the row number.
- **Name:**
Each row can have a name. This can be used as short info for the row.
- **Note:**
A note can be added to each row for information purposes.
- **Enabled:**
This **Yes** or **No** field is used to enable the row transmitting or receiving Art-Net. No is the default.
- **Mode:**
The mode defines what the row is doing. There are four options:
 - **Broadcast:**
This transmits Art-Net using broadcast.
 - **Unicast:**
This transmits Art-Net using unicast.
 - **Auto:**
This transmits Art-Net. It uses the **Broadcast Threshold** number to determine if universes should be transmitted using broadcast or unicast. If the number of universe subscribers (determined by ArtPollRequests) are below the threshold, then it is sent using unicast. If it is above the threshold, then it is broadcasted.
 - **Input:**
The row is receiving Art-Net and merging it into the defined universe.
- **Destination IP:**
This field is only active if the unicast mode is selected. This is the IPv4 address of the receiving device.
- **Subscribed Nodes:**
This field is information about the amount of subscribing devices. This is determined using ArtPollRequests.
- **Local Universe:**
This is the grandMA3 universe to be transmitted or the universe that should receive incoming Art-Net DMX. This is the first universe in the range if the amount is more than one.
- **Amount:**
This is the amount of grandMA3 universes to be transmitted or received.



- **Net:**

The net number is a value between 0 and 127. There are 128 different nets since Art-Net III. Each net is a complete group of **Sub-Nets** and **Universes**. This allows for addressing a total of 32 768 Art-Net universes. To be compatible with Art-Net I and Art-Net II devices, please use net 0.
- **Art-Net Sub-Net:**

There are 16 sub-nets in Art-Net. They can be input in decimal numbers from 0 to 15 or 0 to F in hex numbers.
- **Universe:**

There are 16 universes in each sub-net. They can be input in decimal numbers from 0 to 15 or 0 to F in hex numbers.
- **Art-Net Absolute:**

This is the absolute universe number. It is calculated based on the net, sub-net, and universe numbers. It can also work the other way. A universe number can be input here, and then the net, sub-net, and universe numbers are calculated based on the input.
- **Packet Delay:**

A delay can be set up between each transmitted universe. This can be helpful for older nodes with slower network cards. Sending many universes at once can flood the node. Adding a small delay helps. This setting is only available for outputting modes.
- **Merge Mode:**

The Merge Mode defines how incoming sACN merges into universes. When changing the Merge Mode for a configuration line, all universes defined by Local Universe and Amount are changed together. In the **DMX Universes** -tab of the Patch and Live Patch menu or by editing a universe in the universe pool, it is still possible to independently change the Merge Mode for single universes. As soon as two or more universes of a configuration line have different Merge Modes, the Merge Mode cell will display ... to indicate this. The Merge Mode and Input Priority are described in the **DMX Port Configuration topic**.
- **Input Priority:**

This is the priority of the received Art-Net. This can only be changed when the mode is input. Follow the link above to learn more.
- **Timecode Slot:**

Any received ArtTimeCode is sent to the timecode slot number defined here. The mode does not need to be set to Input to receive the timecode, but Input needs to be enabled. This setting can only be changed when the mode is input. There are a total of 8 timecode slots. Read more about them in the **What are timecode slots topic**.
- **Enable RDM:**

RDM via Art-Net can be enabled for the Art-Net universes specified in the row.

Nodes Tab

Each node detected is a row. Each node has one or more sub-rows with "binds". Each bind can only have 4 ports. If a device has more than 4 ports, then the device has multiple binds.

- **Lock:**

The row can be locked to prevent changes.
- **No:**

This is the row number.



- **Name:**
Each row gets a name from the device. The name can be edited if the device supports it.
- **IP:**
The IP address of the device. The address can be edited if the device supports it.
- **Net:**
Each bind has a net number. The number can be edited if the device supports it.
- **Sub-Net:**
Each bind has a sub-net number. The number can be edited if the device supports it.
- **Output Port:**
This is the universe number of output ports on the device. The number can be edited if the device supports it.
- **Input Port:**
This is the universe number of input ports on the device. The number can be edited if the device supports it.

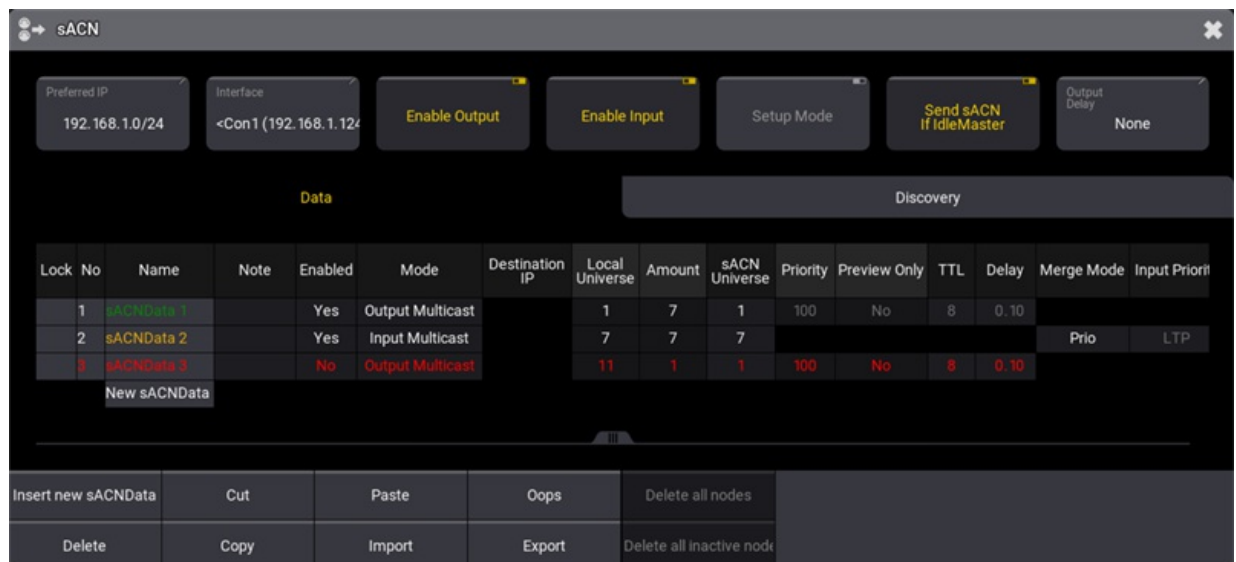


11.2.2. sACN (streaming ACN) Menu

ACN (and streaming ACN) is an ANSI/ESTA international standard. Further readings:
http://en.wikipedia.org/wiki/Architecture_for_control_networks.

ACN (Architecture for Control Networks) is a suite protocol. It uses a lot of elements that are currently not supported by grandMA3. But the ACN protocols also have a version for transporting DMX data. It is called 'Lightweight streaming protocol for transport of DMX512 using ACN' or more popular "streaming ACN" or "sACN". It is the international standard number E1.31.

It is configured in the sACN menu:



sACN menu

See the [Ethernet DMX topic](#) for information on how to open this menu.

Config Buttons

There are some buttons at the top of the menu:

- **Preferred IP:**

This is the preferred IP address or address range used by the sACN protocol. The Interface can be set to Auto, allowing this setting to select the interface that matches the preferred setting. Tap this input button to open a small number input pop-up. The preferred number has to be input as an IP address with the subnet mask written in CIDR notation. For instance, 192.168.1.0/24 tells the system that it prefers an IP address that starts with 192.168.1. The last number does not matter (the "/24" is the same as subnet mask 255.255.255.0). This setting is individual for each station.



- **Interface:**
Tap this button to open the **Select Interface pop-up** to select the desired network interface. This interface will be used for all sACN in and out. The **Auto** option can be used to allow the **Preferred IP** setting to select the interface. The interface is written inside "<" and ">" when the preferred setting selects it. This setting is individual for each station.
- **Enable Output:**
This On/Off button must be On for the master to transmit sACN. Data also needs to be configured for output - read more below.
- **Enable Input:**
This On/Off button must be On for the master to receive sACN. Data also needs to be configured for input - read more below.
- **Setup Mode:**
This On/Off button is used to toggle the setup mode for nodes. This allows configuration data to set up the nodes, without sending sACN DMX data into the network.
- **Send sACN If Idle Master:**
This On/Off button defines if the station transmits sACN data when it is Idle Master. In a session, it is the Global Master who transmits the network DMX. If the station is not in a session with other devices, it is Idle Master. Turning this setting On will make the station output network DMX when it is Idle Master. This must be On if a single station is to output networked DMX. Learn more about standalone devices and networked devices in the [System Overview section](#). This setting is individual for each station.
- **Output Delay:**
This can set an output delay between 0ms and 30ms. This delays the entire sACN output compared to the outputs coming from MA-Net devices.



Important:

- Enabling Setup Mode allows sACN configuration data to be sent and received even when **Enable Output** and **Enable Input** is Off.
- When Setup Mode is Off and output is enabled, then DMX and configuration data are transmitted.
- When Setup Mode is Off and input is enabled, then DMX and configuration data are received.

There are two tabs below the buttons. They are **Data** and **Discovery**. Data is used to set up output and input. Discovery can be used to see the transmitting nodes in the network.

Data Tab

The data tab is a grid of rows and columns. Each row is an sACN configuration. If the row is not valid or not enabled, then the name is in red text. The name text flashes green when sACN data is transmitted.

This is a short description of the columns:

- **Lock:**
The row can be locked to prevent changes.



- **No:**
This is the row number.
- **Name:**
Each row can have a name. This can be used as short info for the row.
- **Note:**
A note can be added to each row for multiline information.
- **Enabled:**
This Yes or No field enables the row to transmit or receive sACN. Yes is the default.
- **Mode:**
The mode defines what the row is doing. There are four options:
 - **Output Multicast:**
When choosing Output Multicast, sACN will be sent as multicast to the relevant multicast addresses.
 - **Output Unicast:**
When choosing Output Unicast, a valid IP address has to be entered in the **Destination IP** column. Universes configured in this row will be sent as unicast to this IP address.
 - **Input Multicast:**
Input Multicast will join the Multicast group of the relevant DMX Input Universe.
Input Multicast is limited to max. 10 Universes. If more than 10 rows are configured as Input Multicast, all rows beyond multicast input row 10 will be invalid.
 - **Input Unicast:**
Input Unicast is not limited and receives sACN data for the relevant universe without joining any multicast group.
- **Destination IP:**
This field is only active if the output unicast mode is selected. This is the IPv4 address of the receiving device.
- **Local Universe:**
This is the grandMA3 universe to be transmitted or the universe that should receive incoming sACN DMX. If the amount is more than one, then this is the first universe in the range.
- **Amount:**
This is the amount of grandMA3 universes to be transmitted or received.
- **sACN Universe:**
This is the sACN universe number the grandMA3 universes is transmitted to or the universe number that is listened to if Input is selected. If the amount is more than one, then this is the first universe in the range.
- **Priority:**
The allowed value is 0 to 200. The highest number has the highest priority. The default value is 100. This priority is used for transmitted sACN.
- **Preview Only:**
sACN data can be sent as preview data. This can, for instance, be used to send DMX to visualizers.
- **TTL (Time To Live):**
Time To Live is a number used to tell routers and some switches how far through the network the sACN data should be transmitted. This is only relevant for output modes. The default value is 8 and this should usually not be changed.



- **Delay:**
A delay can be set up between each transmitted universe. This can be helpful for older nodes with slower network cards. Sending many universes at once can flood the node. Adding a small delay helps.
- **Merge Mode:**
The Merge Mode defines how incoming sACN merges into universes. When changing the Merge Mode for a configuration line, all universes defined by Local Universe and Amount are changed together. In the **DMX Universes** tab of the Patch and Live Patch menu or by editing a universe in the universe pool, it is still possible to change the Merge Mode for single universes independently. As soon as two or more universes of a configuration line have different Merge Modes, the Merge Mode cell will display ... to indicate this. The Merge Mode and Input Priority are described in the [**DMX Port Configuration topic**](#).
- **Input Priority:**
This is the priority of the received sACN. sACN input of grandMA3 ignores sACN priorities and uses this priority instead.

Discovery Tab

This tap displays the transmitting nodes in the network. Each node detected is a row. Each node has one or more sub rows with "pages".

- **Lock:**
The row can be locked to prevent changes.
- **No:**
This is the row number.
- **Name:**
Each row gets a name from the device. The name cannot be edited.
- **Universe List:**
This is a list of the universes the node transmits.



11.2.3. Transmit DMX Using Art-Net

This is a short example of setting up DMX transmission using Art-Net. Learn more about the Art-Net menu in the **Art-Net Menu topic**.

The idea in this example is some lights are patched in universe one to five. The fixtures are connected to Art-Net nodes and the nodes need to get the Art-Net data from the grandMA3. There is also a media server that uses a single universe. It is patched in universe 21 and needs to receive Art-Net universe 21 (hex number 1:5).

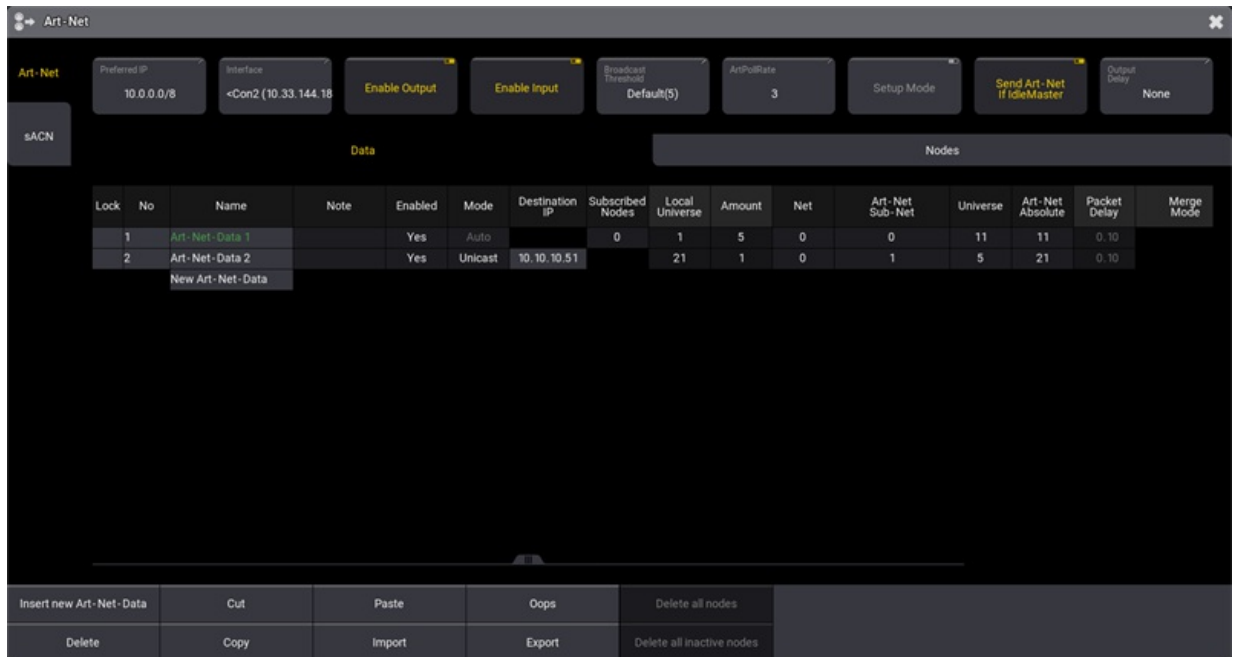
The Art-Net nodes are set up to receive from Art-Net universe 11 (hex number 0:B), so the grandMA3 universe one needs to be sent to this Art-Net universe. The media server is a specific device with a defined Art-Net IP address and this data can be sent using unicast to this device. In this example the media server IP address is set to 10.10.10.51.

The example assumes a patched setup that matches the universes. For demonstration purposes, this is not actually needed. If the grandMA3 onPC is used to try this example, then there might not be parameters to allow actual output.

These are the steps needed:

1. Open the Art-Net menu by pressing **Menu** and then tap **DMX Protocols**. Make sure the Art-Net menu is visible by tapping **Art-Net**.
2. The first line is going to be universe one to five. Edit the following cells to match these options:
 1. Enabled = Yes
 2. Mode = Auto
 3. Local Universe = 1
 4. Amount = 5
 5. Art-Net Absolute = 11
3. A second line is needed for the universe that needs to go to a specific destination. Tap **New Art-Net-Data** below line 1.
4. Tap **Insert new Art-Net-Data** in the button menu at the bottom of the screen.
5. Now the line exists. Edit the new line to match these options:
 1. Enabled = Yes
 2. Mode = Unicast
 3. Destination IP = 10.10.10.51
 4. Local Universe = 21
 5. Amount = 1
 6. Art-Net Absolute = 21 (notice that this automatically calculates the correct Art-Net Sub-Net and Universe number)
6. Finally, tap the **Enable Output** until it is On to activate the transmission of DMX data via Art-Net

That was it. It should look something like this:



Final Art-Net example setup

If the station is not connected to other stations or nodes in a session, it is **Idle Master**. If the station needs to output Art-Net in a setup like this, the **Send Art-Net if Idle Master** setting needs to be turned On. If the station is in a session with other stations, then it is the **Global Master** in the session that transmits the Art-Net data.

The menu can be closed by tapping the cross (✕) in the upper right corner.



12. Single User and Multi User Systems

The grandMA3 system can handle small and big systems with one or many users simultaneously.

grandMA3 systems are often referred to as a **Session**.

An Ethernet network can have several sessions running.

A session can have one, some, or many MA devices as members or listeners.

This page describes some differences between the two primary ways to have a system.

Single User Session

In a single-user session, there is one operator/programmer. This person might have one or several stations (consoles or onPC).

When there is only one user, all stations are logged in as the same **User**.

This is also called **Full Tracking Backup**. The stations are all in sync and share views. The programmer content is the same on all stations.

This is useful for the single operator. If one station fails, the operator can move to another station and continue working without losing anything.

This operator will usually have full admin rights to the entire system.

Multi User Session

In a multi-user session, there are several operators. They might use the same stations and take turns (working in shifts or at different process phases). It can also be a system with multiple users working simultaneously on different stations.

For this setup, the operators must create more **Users** and, most likely, more **User Profiles**. Each operator will then log into the station using their User and may control the complete system or just a part of the system.

The stations share the show file, and the sequences are all in sync, but the users can have different views on the screens, and their programmer is not shared - the output is.

Sometimes systems are a combination of multi-users and full tracking backups; There are several operators (Multi User), but each operator has two (or more) stations logged in with their user (Full Tracking Backup).

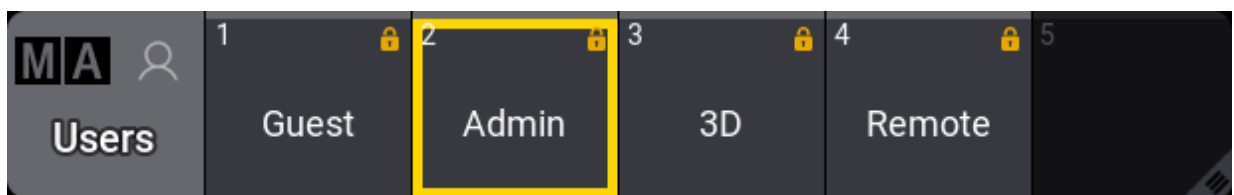
In a multi-user system, it is possible to have users with different operator rights. There are several levels of rights. They span from complete access to only being allowed to change the view.

12.1. Create User

A **User** is also a login name.

Some things are connected to the user. For instance, each user can have different operating rights and different languages. Read more in the [User Settings topic](#).

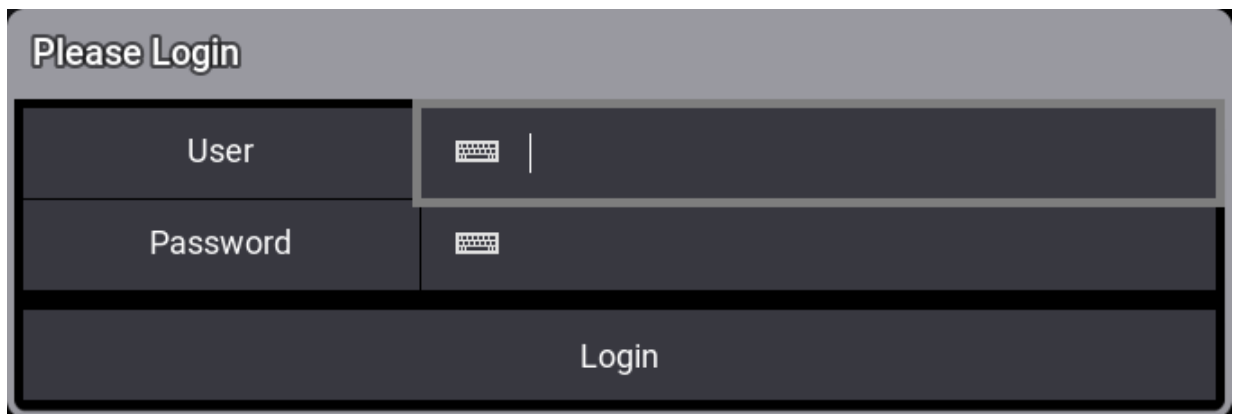
The users are stored in a **User pool**. Read more about creating windows in the [Add Windows topic](#).



The User pool with logged-in Admin user

The currently logged-in user has a yellow frame. It is the selected user.

Tapping one of the other users logs in as this user. If the user has a password, then a **Please Login pop-up** appears.



Please Login pop-up asking for password



Important:

The password is case-sensitive.

Type the correct password to log in.

A show file always has some default users. They use the same **User Profile**.

User Profiles are the ones that hold information about programmer content, selected sequence, current page, views, cameras, etc.

If two users share the same user profile, they share all the information stored in it.

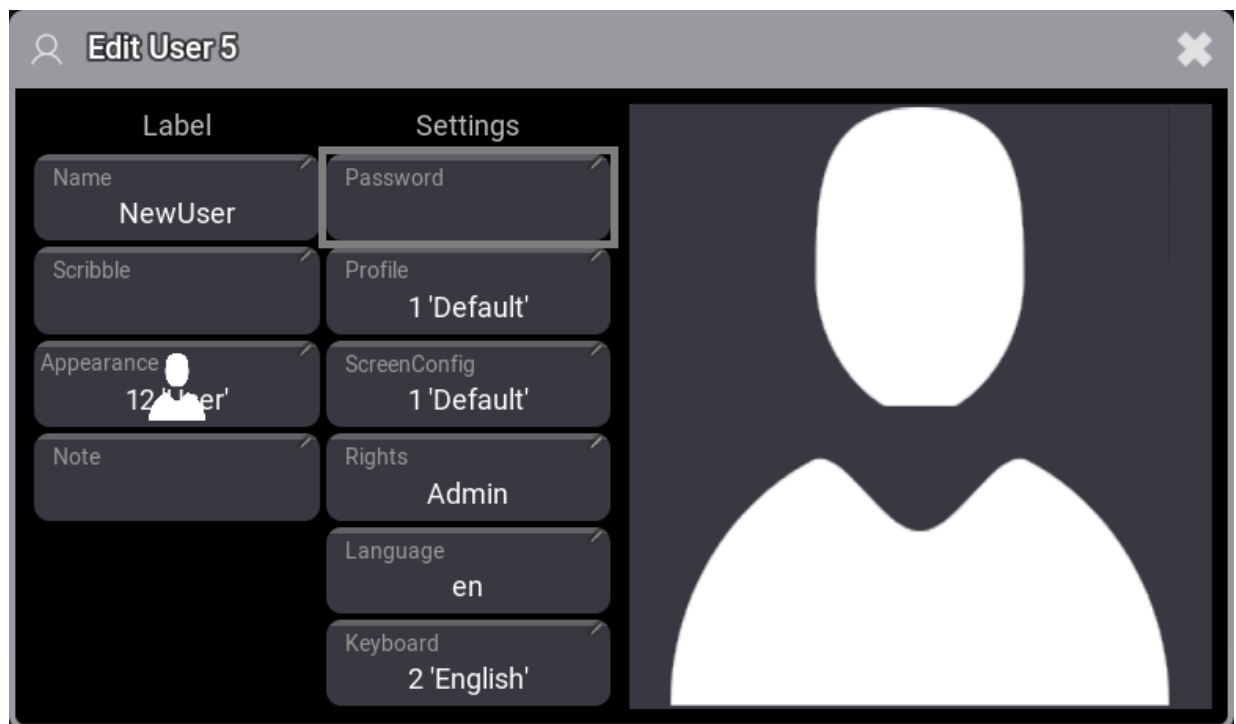
If the user profiles differ, two operators can have different selections, programmer values, views, etc. - **Multi User System**.

Each **User** must have a **User Profile** assigned - Read more about this and how to create user profiles in the **User Settings topic**.

Create a New User

The easiest way to create a new user is to edit an empty object in the user pool, for instance, by using swipecs.

This opens the **Edit User pop-up**



Edit User pop-up with a new user

Here the name should be changed as a minimum. This is also the name used when logging in using the **LogIn keyword**.

The big area on the right side of the pop-up shows the selected Appearance for the user (the image above shows an appearance).

The rest of the settings in the editor is explained in the **User Settings topic**.



12.2. User Settings

Many settings are connected to the **User** and the **User Profile**.

User Settings

There are some different user settings.

- **Name:**
This is the name of the user. This is the login name used with the **Login keyword**.
- **Scribble:**
A user can have a scribble assigned. The scribble is only used in the **User pool**.
- **Appearance:**
A user can have an appearance. The appearance is used in the user pool and when the station is locked.
- **Note:**
A note can be added to the user profile.
- **Password:**
This is an optional password that can be set to the user. It needs to be provided when the user is logging in.
- **Profile:**
Each user has a User Profile assigned. Read more **below**.
- **ScreenConfig:**
Different screen configurations can be created. This is the configuration called when the user logs in.
- **Rights:**
There are six different levels of rights. Read more **below**.
- **Keyboard:**
The on-screen keyboard can have different layouts. This setting can be used to select one of the available layouts.
- **Privacy Policy:**
This setting can be toggled between Yes and No. Yes means that this user has agreed to the **Privacy Policy**.
The policy can be found in **Menu** - **Settings** - **Software Update** - **Privacy Policy**.

There are two places in the GUI where the settings can be viewed and edited. The settings are the same.

The first place is the Users pool. Editing a pool object opens an editor where the settings can be changed. Learn more about the Users pool in the **Create User topic**.

The second place is the **User Configuration**. Navigate to it using these steps:

1. Press the **Menu** key.
2. Tap the **Settings** button.
3. Tap the **User Configuration** button.

This opens the list of users in the show. Edit any field to change the setting.



User Profile Settings

The user profiles contain most of the settings relevant to the users. The programmer information, views, selected elements, values, preferences, and much more.

Many of these elements are stored and selected by using the software. Some settings can be set in the User Profiles menu.

Navigate to the menu:

1. Press the **Menu** key.
2. Tap the **Settings** button.
3. Tap the **User Configuration** button.
4. Tap the **Profiles** button on the left menu.

Some settings here can also be changed in relevant places in the software. For instance, the value readout setting can be changed in the **Encoder Bar**.

- **Name:**
This is the name of the user profile.
- **Note:**
This is a note for the user profile.
- **DMX Readout:**
This is used to change the readout of DMX values. This is useful when editing **fixture types**.
- **Normal Value:**
This is the intensity value used when the **Normal keyword** is used - typically by pressing the **At key** twice. It is a DMX value, so the DMX readout setting affects how to input a value in this field.
- **Wheel Resolution:**
This can be used to change the resolution of the wheels on the consoles. The options are: Coarse, Normal, and Fine.
- **Wheel Mode:**
This setting changes how the wheels work. Read more **below**.
- **Precise Edit:**
This function is used when editing values in sheets. It can be difficult to hit a field precisely in a sheet with fingers. Turning this Off makes it easier to select a field without accidentally creating a new selection. Learn more about this in the **Gesture topic**.
- **Screen Encoder:**
When this option is enabled, the rightmost dual encoder becomes a screen encoder. Turning the inner encoder moves the focus in a vertical direction while the outer encoder scrolls in a horizontal direction. To create a lasso selection in a grid, like in the patch menu, press, hold and turning the inner encoder. Press and release the inner encoder or the dual encoder key to edit the currently focussed object.
The screen encoder label in the encoder also displays which display the encoder is currently active. This option is enabled by default.



- **Time Key Target:**

The **Time key** can have two different targets: Cue or Fixture. This defines if the key defaults to keywords relating to cue timing or fixture layers.
- **TCSlot:**

This displays the currently selected timecode slot. Read more in the **[What are Timecode Slots](#)** topic.
- **Value Readout:**

This is the default value readout. Many sheets can be set to show a readout, which can be a specific readout type or follow this default type. This setting can be changed in the **[Encoder Bar](#)**.
- **Speed Readout:**

This is the default speed readout. The options are Hertz, BPM, and Seconds. Speed is used in **[Phasers](#)**.
- **Preset Readout:**

This changes the way presets are displayed in Fixture and Sequence Sheets. The preset can be displayed with a combination of three elements:

 - **ID:** The ID number of the preset.
 - **Name:** The name of the preset.
 - **Value:** The values in the preset.
- **Help Popup Zoom Factor:**

This is the default zoom factor in the help pop-up.
- **Overlay Fade:**

This time sets a fade time used by pop-ups and menus in the user interface. The default time is 250 ms.
- **Time Readout:**

This is the default time readout for the user profile. Read more about the readout **[below](#)**.
- **Frame Readout:**

This is the default frame readout for the user profile. Read more about the readout **[below](#)**.
- **Color Readout:**

This is the default color readout. This default is used in the **[encoder toolbar](#)** and most sheets - unless a different readout is selected for the sheet.
- **Oops Views:**

This setting toggles if view operations are oopsable. **Create Oops** needs to be enabled for this function to work.
- **Oops Programmer:**

This setting toggles if programmer operations are oopsable. **Create Oops** needs to be enabled for this function to work.
- **Oops Selection:**

This setting toggles if selection operations are oopsable. **Create Oops** needs to be enabled for this function to work.
- **Create Oops:**

This setting toggles if the Oops function in general is turned On or Off. "Yes" means that the Oops function is On.



- **Oops Confirmation:**
 - **Always:** All oops actions need confirmation.
 - **Main:** This will require a confirmation for the following actions:
 - Create
 - Delete
 - Exchange
 - File actions while loading from a file.
 - Insert
 - Layout
 - Move
 - Remove
 - **Never:** Actions are oopsed without confirmation.
- **Auto Remove Gaps:**

This setting is for the selection grid. Please read the [Selection Grid](#) topic for more information.
- **Auto Remove Offset:**

This setting is for the selection grid. Please read the [Selection Grid](#) topic for more information.
- **Mirror SpecialExecutor Pages:**

This setting is for the custom section of the [grandMA3 extension](#). If the setting is "Yes" then the extension has the same assignment as the custom section of the station the extension is connected to. If the setting is "No", then each extension can have its own assignments in the custom section. grandMA3 extensions with the same WingID are always mirrored - they are essentially defined as the same. Learn more about connecting grandMA3 extensions in the [Connect grandMA3 extension topic](#).
- **Show Appearance In Cue Input:**

This setting is used to define if the cue appearances are shown in pop-ups where cues can be selected. For instance, the pop-ups that appear using the [Goto](#) and [Load](#) commands without a specified target.
- **Show Settings In Editors:**

Toggling this shows or hides the settings in some editors. This setting is also in the title bar of the editors where this can be toggled.
- **Show Connectors:**

This setting shows or hides the Connector Overlay. Learn more about the Connector Overlay in the [Output Configuration topic](#).
- **DMX Tester Address Mode:**

The following settings relate to the [DMX Tester](#).

 - **Uni:** This splits the DMX address into universes and universe addresses.
 - **Abs:** This shows the DMX address as an absolute DMX address.
- **DMX Tester Retain Mode:**

Toggles the retain mode in the DMX tester.



- **DMX Tester Mode:**
 - All: All DMX addresses are available for testing.
 - Patched: Only patched addresses can be tested.
 - Unpatched: Only unpatched addresses can be tested.
- **DMX Tester Test Value:**

This is the test values used by the DMX tester.

User Attribute Preferences

From the profile setting, the **User Attribute Preferences** can be accessed by tapping **Edit Encoder Bar**.

This is used to set the user profiles preferred readout type for the natural readout. It also has different resolutions and encoder press resolution multipliers.

Resolution multipliers define the factor by which an encoder changes a value when the encoder is pressed or, in the case of the dual encoders, the difference between the inner and outer ring.

At the top, there are some settings:

- **Time Layer Resolution:**

Sets the encoder resolution for the time-related layers.
- **Phase Layer Resolution:**

Sets the encoder resolution for the phaser-related layers.
- **Dual Encoder Factor:**

This defines the multiplier the outer encoder ring uses in relation to the inner encoder ring.
- **Dual Encoder Press Factor:**

This defines the multiplier for the outer encoder ring when the dual encoder key is pressed while turning the outer encoder ring.
- **Link Resolution:**

This setting defines how the encoder resolution is linked between features. The options are **Single** and **Feature Group**. Learn more about this setting in the [Encoder Toolbar topic](#).

The rest of the pop-up lists all the attributes in rows.

Three different columns with values can be changed for each attribute:

- **Natural Readout:**

The Natural readout allows defining the desired readout type per attribute. For instance, dimmer values are best displayed in percentage readout. In contrast, pan or tilt values are better readable using the physical readout, which displays the pan and tilt values as degrees. This setting selects the desired readout when Natural is selected in sheets and encoders.

The user-defined readout has a higher priority than the readout defined for the attribute. The user-defined readout can be linked to the default (from the attribute definition). This option is called Default, and the actual readout is shown in angle brackets.



- **Encoder Resolution:**

The encoder resolution defines how big an attribute value change will be when turning a dual encoder by 1 click. This setting can also be changed by pressing MA and tapping the channel function area in the encoder toolbar. Learn how in the [Encoder Resolution topic](#).

- **Encoder Press Factor:**

This multiplier is used when the inner encoder is pressed and turned for this attribute.

Learn more about the different resolutions available in the [Encoder Resolution topic](#).

Wheel Mode

There are four different level wheel modes:

Additive:

Additive keeps the difference between dimmer values until they reach 0% or 100%, using the level wheel. After 0% or 100 % is reached, the values will be leveled out.

Example:

Fixture 1 has a value of 50, and fixture 2 is at 60. When the level wheel is used to take them both up to 100 and then back down, they both go down from 100 at the same time and at the same level.

Incremental:

Incremental would always keep the difference between the dimmer values, even if you reached 0% or 100% by using the dimmer wheel.

Example:

Fixture 1 has a value of 50, and fixture 2 is at 60. When the level wheel is used to take them both up to 100 and then back down, fixture 1 will start coming down first, and then fixture 2 will follow when fixture 1 is at 90.

Prop.+ (Proportional positive):

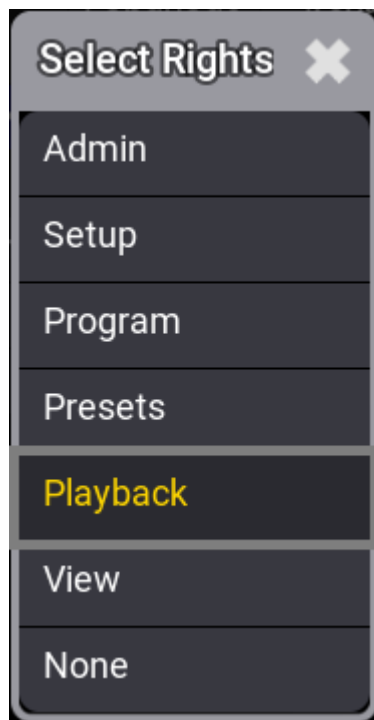
When using the level wheel to turn up the dimmer values, the difference in the dimmer values will **decrease**. Turning up to 100% will make all channels reach 100% at the same time.

Prop.- (Proportional negative):

When using the level wheel to turn up the dimmer values, the difference in the dimmer values will **increase**. Turning down the values will make all channels reach 0% at the same time.

User Rights

There are seven different levels of user rights in the system. They are a user setting - read more above.



Select Rights pop-up is used to select one of the rights levels

The settings are:

- **Admin:**
This is the right to change everything in the console, system, and show.
- **Setup:**
This will limit access to some of the elements in the console. There are other console settings that can be accessed.
- **Program:**
At this level, the user cannot make major changes to the patch. It only gives access to the "Live Patch". Most programming operations can be done.
- **Presets:**
This level allows for updating existing presets. But the user cannot edit the cue content.
- **Playback:**
This level allows playback and running a programmed show. But the user cannot store anything.
- **View:**
With this user right, it is not allowed to use a programmer. The user is allowed to call views and log in as a different user.
- **None:**
The user is only allowed to log in as a different user.

Time and Frame Readout

The applied values are used in most places the time is displayed.



Hint:

The Readout can be changed for individual sheets in the **window settings**.

There is a hierarchy of the readout/format settings. The default is set in the user profile. If this default is changed in a window, then the window setting is used.

Timecode is an example of an area with many layers in the hierarchy. Read more about the timecode in the **Timecode section**.



Hint:

The user profile defined time readout is used in the encoder bar even if the readout for an, for instance, fixture sheet is changed separately.

The timing calculator value indicator is aligned with the preselected frame readout.

To provide better optical representation, the trailing zeros from frames or seconds are always suppressed. When the frame readout is set to a frame unit, the times will always display trailing zeros.

To easily distinguish between fractions of a second and frames, fractions of a second are separated from second using a dot (.), while frames are separated from seconds using a colon (:).

The time options are:

- **10d11h23m45:**
The time is separated into days, hours, minutes, and seconds using letters as separators.
- **251h23m45:**
This is separated into hours, minutes, and seconds using letters as separators. The hour number can become more than 24 if time is more than a day.
- **10.11:23:45:**
The time is separated into days, hours, minutes, and seconds using a dot and colons as separators.
- **251:23:45:**
This is separated into hours, minutes, and seconds using colons as separators. The hour number can become more than 24 if time is more than a day.

The frame options are:

- **Seconds**
- **24 fps**
- **25 fps**



- **30 fps**
- **60 fps**



Hint:

If frame readouts (24, 25, 30, 60 fps) are used, fractions are separated by a colon, and fractions of seconds are divided by dots.

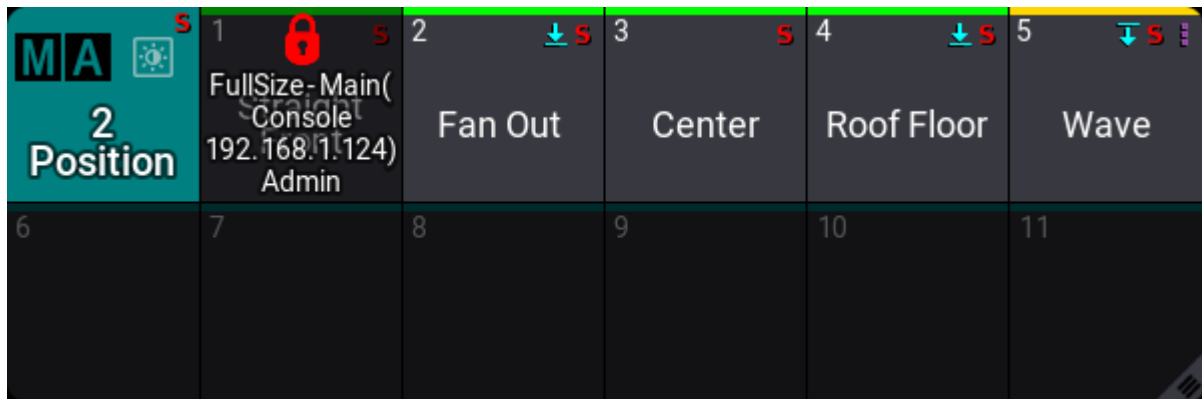
Fps means frames per second.

12.3. Object Ownership

A multi-user setup presents a situation where multiple users can try to edit the same object. This could cause unpredicted problems.

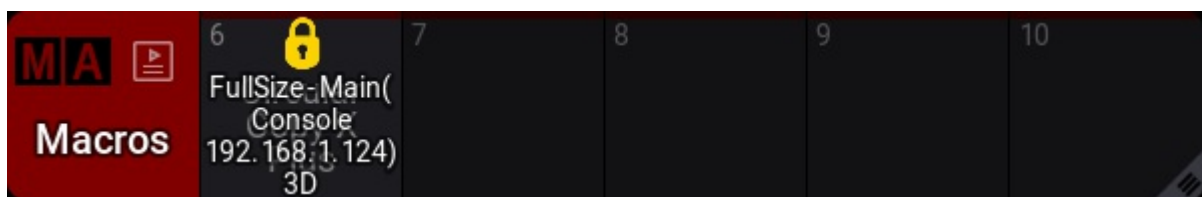
The grandMA3 system has a locking mechanism that temporarily prevents multiple users from editing the same object.

When a user starts to edit an object, the other users are presented with a pulsing lock icon and some text offering information about the station and the user editing the object.



Position preset 1 is being edited by a different user

An object can be fully locked by another user editing the entire object. A red lock indicates this (see the example above). A yellow lock indicates that an object is partially locked. This means that only a part of the object is being edited and is currently locked. Other elements of the object can still be edited.



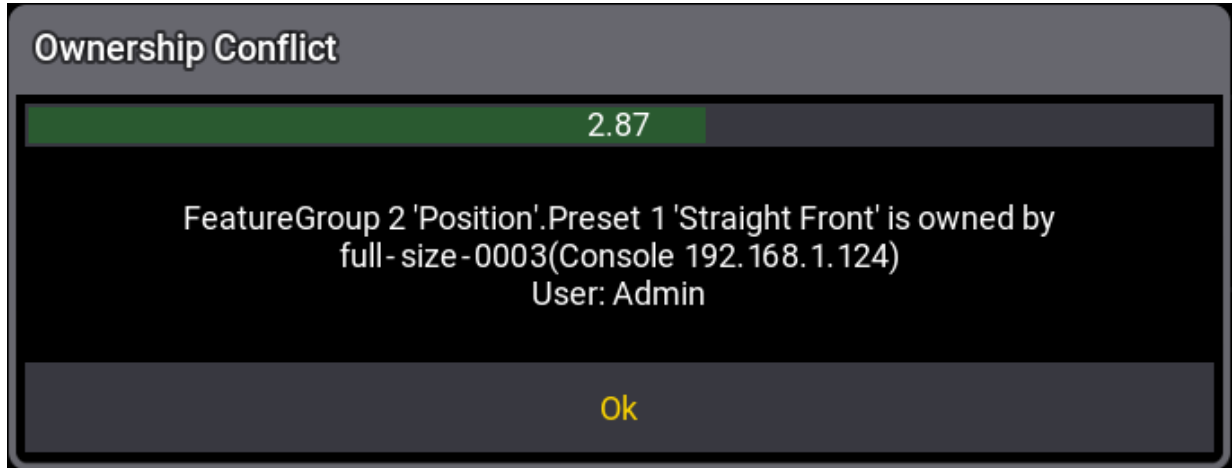
A macro line is being edited by a different user

All users can use the objects while it is being edited, but using the object before the edit is done might not reflect the edited values.

The user editing the object has ownership of the object while it is being edited. The station being used to edit the object will have a green frame in the relevant window while it is being edited.

Suppose the editing user is logged into several stations. In that case, other stations with this user display a yellow frame around the windows to indicate the user has started editing an object on a different station.

If a different user tries to edit a temporarily locked object, a pop-up appears with additional information:

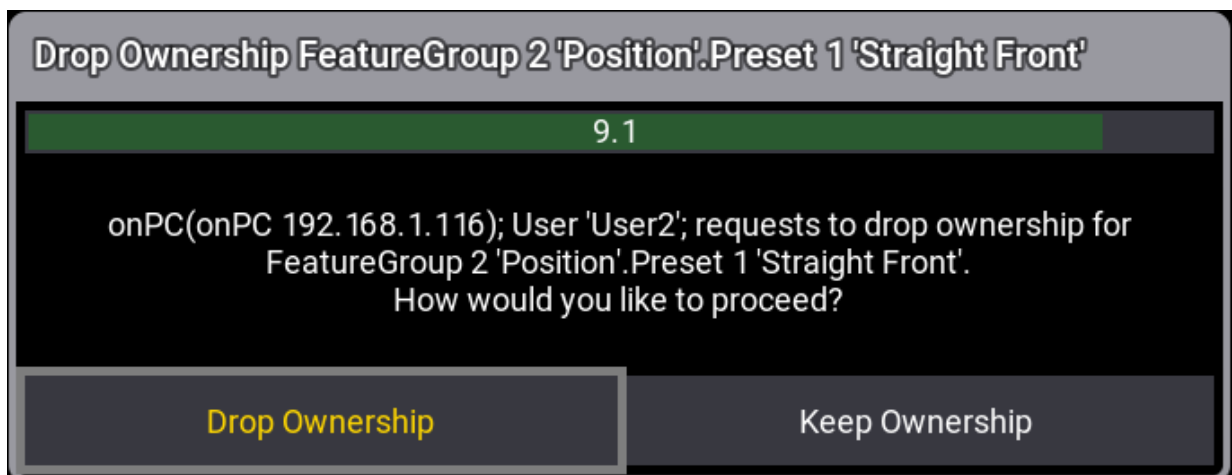


Ownership Conflict pop-up

The pop-up has a 10 seconds countdown that automatically closes the pop-up. It can be closed before the countdown ends by tapping **Ok**.

A user can try to get ownership of a temporarily locked object using the **DropOwnership keyword**.

This presents a pop-up at the stations belonging to the user who owns the object:



Drop Ownership pop-up

In this pop-up, the current owner can choose to drop the ownership by tapping **Drop Ownership** or keep it by tapping **Keep Ownership**.

This pop-up also has a countdown of 10 seconds. If the time runs out, then the ownership is dropped. This is the default action that allows other users to get objects currently owned by a user who has left the stations and simultaneously allows a user to keep ownership as an active choice.

The **ListOwnership keyword** can list all or specific ownerships in a session. The result is shown in the **Command Line History window**.



Hint:

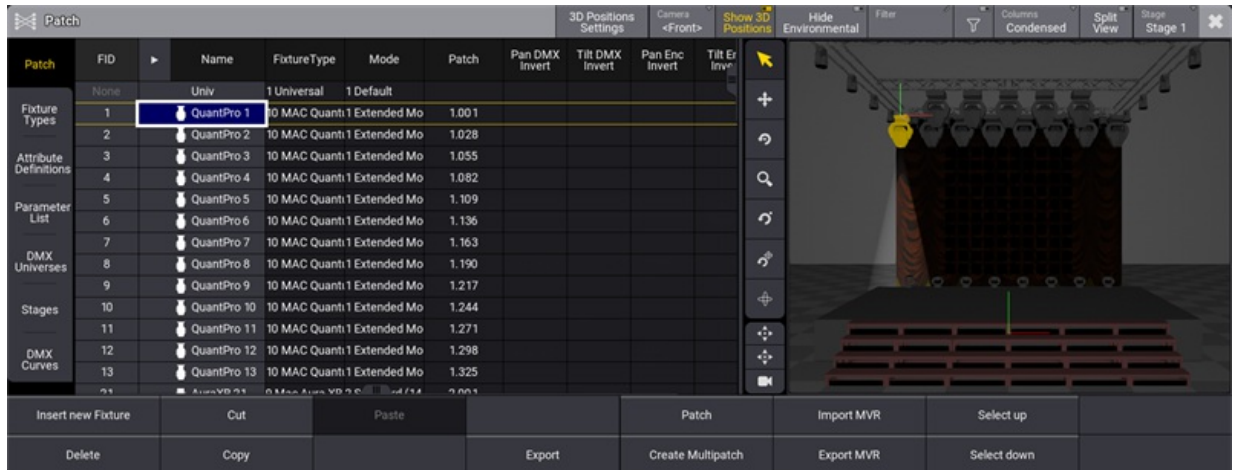
Objects can also be locked by automatic processes that start as a follow-up of the user's actions. For example, when a user updates a preset, the Total Reference Update mechanism locks the needed objects with ownership while running. The automatic process owns the objects while the update process is running.



13. Patch and Fixture Setup

Fixtures need to be added to the show file before they can be controlled or **operated**.

Fixtures are added to the show using the **Patch** menu.



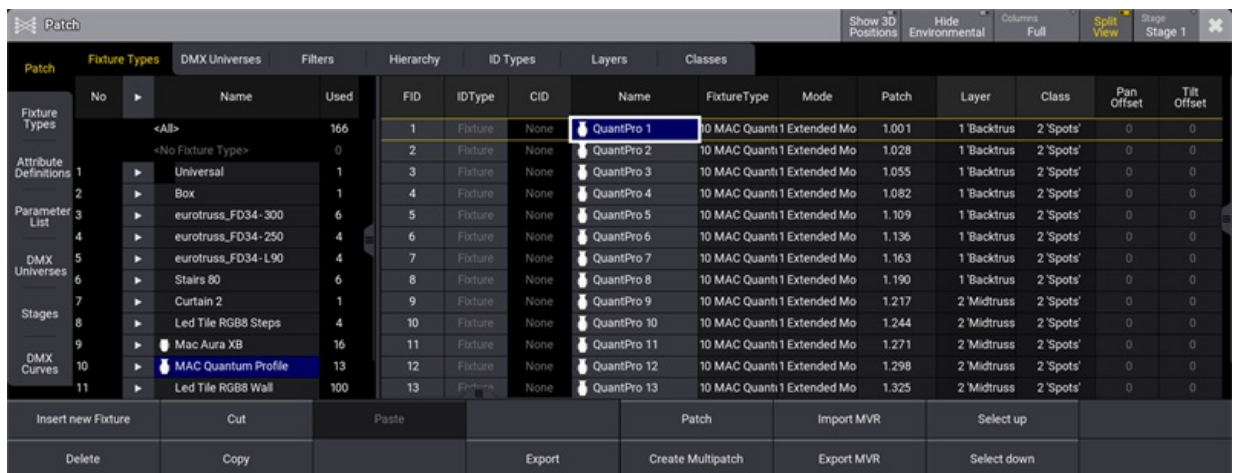
The patch menu with some fixtures patched

The patch menu gives access to **Fixture Types**, **Attribute Definitions**, **Parameter List**, **DMX Universes**, **Stages**, and **DMX Curves**.

The patch menu has two different modes. The image above shows the **Condensed** mode. This mode hides a lot of columns that might not be needed. The other mode is called **Full**. This shows all columns in the patch menu. The mode can be toggled by tapping **Columns** in the title bar.

The fixture list can also be filtered using the filter settings in the title bar or by activating **Split View**.

Split view filters the fixtures by different column properties.



The full Patch menu in Split View



The different split options appear as tabs that can be selected to filter the fixture list.

The list of fixtures is separated into two sides. The left side lists the different elements of the selected split filter. The right side lists the fixtures that qualify based on the object selected on the left side.

The tabs and split options (in Full mode) are:

- **Fixture Types:**
The left side lists the imported fixture types, including <All> (all fixtures) and <No Fixture Type> (fixtures without an assigned fixture type). Learn more about fixture types in the [Fixture Types section](#).
- **DMX Universes:**
The left side lists the DMX universes, including <All> (all universes) and <Unpatched> (fixtures without an assigned DMX address). Learn more about DMX universes in the [DMX Universes topic](#).
- **Filters:**
The left side lists the filters in the Filter pool, including <All> (all fixtures). Learn more about filters in the [World and Filters section](#).
- **Hierarchy:**
The left side lists the hierarchical structure of the patch. It lists the parent elements and can be unfolded. The right side displays the elements directly dependent on the selected object on the left side.
- **ID Types:**
The left side lists the different defined ID types, including <All> (all fixtures) and <No ID> (fixtures without an assigned FID or CID). Learn more about ID types in the [What are Fixtures topic](#).
- **Layers:**
The left side lists the different defined layers, including <All> (all fixtures) and <No Layer> (fixtures without an assigned layer). Learn more about layers in the [Classes and Layers topic](#).
- **Classes:**
The left side lists the different defined classes, including <All> (all fixtures) and <No Class> (fixtures without an assigned class). Learn more about classes in the [Classes and Layers topic](#).

It is possible to have a small version of the 3D stage open in the patch. It can be toggled On or Off with [Show 3D Positions](#) in the title bar. It is visible in the first example image above.

When it is On, it is visible on the right side of the menu. It can be adjusted in size by moving the separator. Also, when it is On, a [3D Positions Settings](#) appears in the title bar. Tapping this opens a limited version of the settings for the 3D viewer.

The settings have two tabs: **Misc** and **Label**.

The tab Misc offers these settings:

- **Wireframe:**
Enables the rendering of the 3D scene as wireframe. By default, this setting is disabled.
- **Beam:**
Changes the visualization of the beam. Available values are "Simple" and "Line". Simple is the default value.



- **BodyQuality:**

Changes the visualization of the fixture body:

- **Box:**

The whole bounding box of the fixture is visualized as one box.

- **Standard** (Default):

The fixture is visualized with its original meshes until the vertex limit of 1 200 vertices is reached. If the overall vertex count of the fixture exceeds 1 200 vertices, the fixture is visualized with default meshes.

- **Ultra:**

The fixture is visualized with its original meshes, no matter its vertex count.

The Label settings are the same as for the normal 3D (link below).

When the 3D area has focus (for instance, when tapped), the encoder toolbar changes to the 3D encoder toolbar.

The 3D area in the patch does not visualize the current DMX output. The selected fixture in the patch shows an intensity output of 100%. The zoom of a fixture is visualized with the default value of the zoom attribute or, respectively, with the value of the Beam Angle of its Beam Geometry if a fixture does not have a zoom attribute. Other functions of the fixture (for example, color, iris, or gobo) are not visualized in the 3D area in the patch.

Learn more about 3D and the setting in the [3D Viewer topic](#).



13.1. What are Fixtures

Fixtures are the different devices that can be controlled by the grandMA3.

Fixtures are added using the patch menu. Each fixture is a row in the patch menu. Each row has some settings organized in columns. Two of the columns are **Fixture Type** and **Mode**. Fixture Type is the definition or description of the physical fixture and the DMX definition for the fixture might be separated into different modes. Each fixture type can have several different modes. Only one mode can be selected for each fixture in the patch. Read more about editing and creating fixture types in the [Fixture Types section](#).

Fixtures contain different **Attributes** or **Parameters**. The parameters can have different values. Individual parameters are the ones manipulated when controlling the fixtures' attributes. For instance, changing the dimmer attribute for ten fixtures changes the dimmer parameter on each of the ten fixtures. Read more about attributes in the [Attribute Definitions topic](#).

Fixtures in the grandMA3 are selected using an ID. There are 11 different **ID Types**. These ID Types can be used to organize the fixture list. Five of the ID Types are **Universal**, **Fixture**, **Channel**, **MArker**, and **Multipatch** - they are locked and cannot be renamed. **Universal** is a fixture type that is automatically created and should not be edited manually. **Multipatch (MP)** ID type is a special type used to create multi-patch versions of a fixture - learn more in the [Add Multipatch Fixture](#) topic.

The remaining six ID types can be renamed to match the needs of the show. These are called **Custom ID** because they can be customized. The default names are **Houselights**, **NonDim**, **Media**, **Fog**, **Effect**, and **Pyro**. Each of these is a keyword. Renaming them changes the corresponding keywords. Read about the ID type keywords following these links: [Fixture](#), [Channel](#), [Universal](#), [Houselights](#), [NonDim](#), [Media](#), [Fog](#), [Effect](#), [Pyro](#), [MArker](#), and [Multipatch](#).

A fixture can have a **Fixture ID (FID)** and one other **ID Type (CID)** with the same or a different ID number. A fixture needs at least one ID to be **operated**.

Fixtures are added to a stage. In a new empty show, there is a standard stage (Stage 1). More stages can be added to organize the setup. Read more about Stages in the [Stages topic](#) and about placing fixtures in the [Position Fixtures in the 3D Space topic](#).

Fixtures can have **Class** and **Layer** information. These are two different ways to organize or group the fixtures. Read more in the [Classes and Layers topic](#). The patch menu can be filtered to hide or show fixtures, for instance, using specific classes or layers.

Another way to organize the patch is to add fixtures as a "child" (structurally below a "parent") to another fixture. A special **Grouping** fixture is very useful for this. It is an "empty" fixture in the generic manufacturer group. This can be useful if there are many fixtures of one type, for instance, 100 LED panels. A grouping fixture would be added to the patch and the LED panels are added inside or below the grouping fixture. The grouping fixture can have an ID. Selecting this ID then controls the LED panels inside or below the grouping fixture. An example of this structure can be seen in the demo showfile.



Fixtures also need to be patched to a DMX address in the grandMA3. If it does not have a DMX address, then the system does not know where to send the parameter values.

Example 1

A row is added to the patch menu. A Robe MegaPointe is assigned to the row. Now the fixture exists in the patch menu. The fixture needs an ID to be able to be selected and controlled. The fixture gets FID (Fixture ID) 1. Now the fixture can be selected using the **Fixture 1** command. For the fixture to react it needs to have the same DMX address as set in the fixture itself and it needs to be connected to a DMX port set to the correct universe. The fixture gets address 3.54. The universe number is 3 and the DMX address in that universe is 54 - this needs to match the fixture setting.

Example 2

A row is added to the patch menu. A generic dimmer (8 bit) is assigned as the mode. The fixture is a part of the building and is used for house lights. The ID Type is changed to **Houselights** and CID (Channel ID / Custom ID depending on the ID type) is changed to 101. It does not get an FID. It can now be selected by pressing **Channel** three times to get the houselights keyword and the numbers **1 0 1**.

The fixture gets the Patch address 1.101 (universe 1, address 101).



Hint:

The channel key is used to get all other ID types than **Fixture**. Pressing it multiple times toggles through the ID types.

13.2. Add Fixtures to the Show

This is the process for adding new fixtures (or devices) to the show and the patch, using the GUI.



Quick Steps in a new show:

1. Open the Patch Menu.
2. Follow the steps in the wizard:
 - 2A. Select the fixture type.
 - 2B. (optional) Add a custom name.
 - 2C. Type the quantity.
 - 2D. Type the first ID number.
 - 2E. (If in **Full** mode) Select a Layer and class (**None** is an option).
 - 2F. Type the patch address for the first fixture.
 - 2G. Tap **Apply**.
3. Close the Patch menu and tap **Ok** to save the changes.

These are steps needed for adding the first fixtures to a new show - read below for details.



Quick Steps in a show with existing fixtures:

1. Open the Patch Menu.
2. Select the row where the new fixtures should be inserted and tap the **Insert New Fixture** button.
3. Make sure the desired source is selected.
4. Select the desired fixture type (use the filter to limit the list) and tap **Select**.
4. Fill in the rest of the fields in the wizard and tap **Create!**.
5. Edit any desired field in the patch grid before closing the Patch menu.
7. Close the Patch menu and save the changes.

These are steps needed to add more fixtures in an existing show - read below for details.

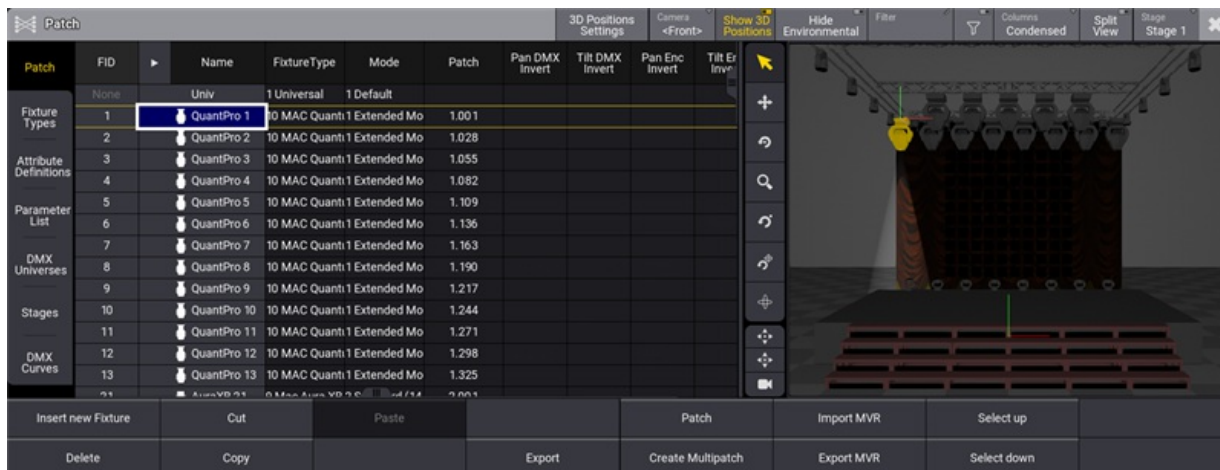
Everything about adding fixtures is done from the Patch menu.

Navigate to the Patch Menu

The Patch menu needs to be open to add fixtures.

1. Press the **Menu** key.
2. Tap the **Patch** button in the menu pop-up.

The patch menu is now open.



The open Patch menu - in condensed mode - with some fixtures

When the menu is opened the first time, a wizard helps to add the first fixtures to the show, and instead of the patch menu, there is a guide through the fixture selection and the **Insert New Fixtures** pop-up (see below).

Insert a Device in the Patch

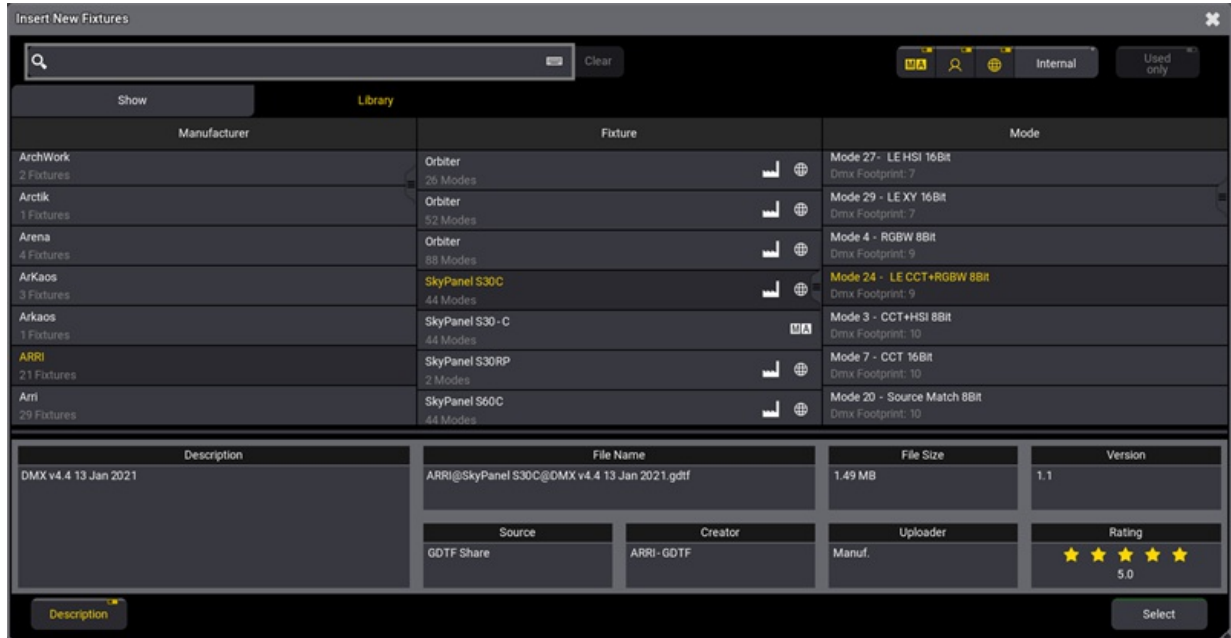
Each fixture needs a row in the patch menu.

The fixtures belong to a "parent" stage object. The default parent is **Stage 1**. Several stages can be created - read more in the [Stage topic](#).

The selected stage can be changed using the stage button in the title bar.

1. Select the stage where the fixtures should be added using the button in the title bar.
2. Tap the **New Fixture** area on the list if it needs to be added to the bottom, or tap an existing fixture to insert new fixtures above the selected fixture.
3. Tap **Insert new Fixture**.

This opens the **Insert New Fixture** wizard pop-up on the **Select DMX Mode to use** part.



Select a fixture in the libraries

Below the search bar are two tabs: **Show** and **Library**. Show is the fixture types already imported into the show file. This list can be limited to only showing the patched fixture types. This can be done by activating the **Used only** button. The **Library** tab lists the different fixture types that can be used. With library selected, there are extra buttons available on the right side.

One button selects the drive. This makes it possible to select the internal drive, previously installed versions library, or an external USB drive. In the image above, it is the **Internal** button. The button is labeled to show the selected drive. Next to this button, there are small toggle buttons that select different sources.

The sources are MA (grandMA2 converted fixture files and grandMA3 fixture files), User and Shares (GDTF share and grandMA3 fixture share). The share button is only available when there is an active connection to a World Server (learn more in the [World Server topic](#)).

Restriction:
 grandMA2 fixture files cannot be loaded directly into the grandMA3. The grandMA2 converted source mentioned above is the grandMA2 library converted to match the grandMA3 structure. Fixture files are converted when a grandMA2 file is stored as a grandMA3 file. Learn more in the grandMA2 manual / Using the Backup Menu / Save as grandMA3.

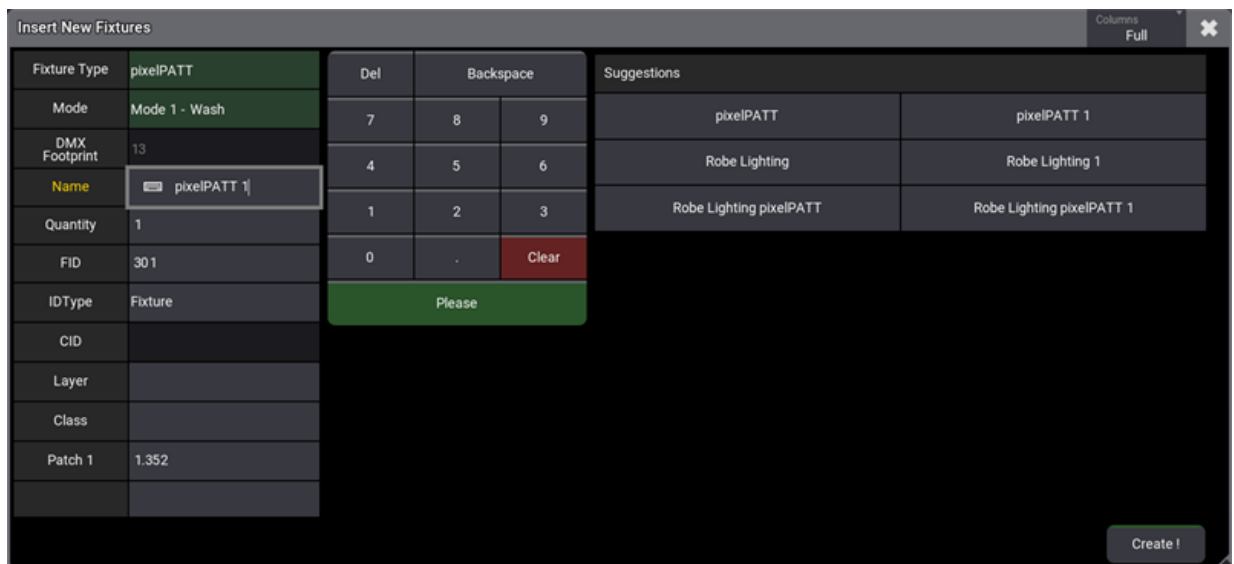
4. Select the desired drive and activate the desired sources.

The fixture libraries are sorted by manufacturers in the left column. Selecting a manufacturer lists the fixtures from that manufacturer in the center column. Selecting a fixture lists the available modes in the right column.

At the bottom, a **Description** button can show an area with extra information about the selected fixture type file.

5. If the list is long, filtering it by typing a word in the **Search** input might make sense. The search is across manufacturers, fixtures, and modes (If a fixture has multiple modes, it will show them all, but only fixtures that include the searched mode).
6. Select the desired fixture and mode from the list and tap **Select**.

This reveals the **Insert New Fixtures Wizard**. This has a list of elements that needs information to add the fixtures to the patch. The wizard comes in two different modes, depending on the **Columns** mode of the patch. This mode can be toggled in the **Columns** button in the title bar. When the mode is **Full** then, there are more elements (ID Type, CID, Layer, and Class). This is the full version of the pop-up:



Fill out all the desired fields in the right column to insert fixtures

7. All the fields in the right column can be filled with information. The required fields have a value suggestion. These can be edited to suit the needs. All the information can also be edited afterward in the patch menu. The area on the right side adapts to help fill out the selected field. It might have suggestions for input.
8. Accept the suggested values, or edit them to suit the needs.
9. Tap **Create !** to add the fixtures.

A row now represents each new fixture in the patch menu. Read below for an explanation of each column in the patch menu and detailed information about changing some values.

Short Explanation of Each Column in the Patch Menu

The patch menu has a lot of columns. This is a short explanation of each. Remember that the patch menu has a condensed and full mode. Select Full to see all the columns.

- **FID:**

This is the Fixture ID of the fixture. Read more in the **Assign an ID to fixtures** below.



- **IDType:**
This is the ID Type of the fixture. Read more in the **Assign an ID to fixtures** below.
- **CID:**
This is the fixture's CID. Read more in the **Assign an ID to fixtures** below.
- **Name:**
This is the name of the fixture. If there are sub-fixtures, then a right-pointed arrow can be tapped to unfold the sub-fixtures.
- **FixtureType:**
This is the name of the selected fixture type.
- **Mode:**
This is the mode of the selected fixture.
- **Patch:**
This is the first DMX address of the fixture. Read more in the **Assign DMX address to fixtures** below.
- **Layer:**
Fixtures can be organized in Layers. This is the layer information for the fixture. Read more in the **Classes and Layers topic**.
- **Class:**
Fixtures can be organized in Classes. This is the class information for the fixture. Read more in the **Classes and Layers topic**.
- **Pan Offset:**
This offset value can offset the DMX output for the Pan attribute. The offset value is applied just before the DMX output. This means the offset is not shown in the programmer, cues, or presets. The offset can be useful if, for instance, a fixture is hung differently than what has been programmed, especially in a touring situation where it is not a permanent change.
- **Tilt Offset:**
This is the same as the Pan Offset, but for the tilt attribute. Read above.
- **Pan DMX Invert:**
The Pan DMX output can be inverted by editing this field. This can be useful depending on how the fixtures are mounted.
- **Tilt DMX Invert:**
The Pan DMX output can be inverted by editing this field. This can be useful depending on how the fixtures are mounted.
- **Pan Enc Invert:**
Pan can be inverted for the encoder rotation by editing this field. This can be useful depending on how the fixtures are mounted.
- **Tilt Enc Invert:**
Tilt can be inverted for the encoder rotation by editing this field. This can be useful depending on how the fixtures are mounted.
- **Pos X:**
This is the fixture's position on the X-axis in the 3D window.
- **Pos Y:**
This is the fixture's position on the Y-axis in the 3D window.



- **Pos Z:**
This is the fixture's position on the Z-axis in the 3D window.
- **Rot X:**
It is the rotation of the fixture on the X-axis in the 3D window.
- **Rot Y:**
It is the rotation of the fixture on the Y-axis in the 3D window.
- **Rot Z:**
It is the rotation of the fixture on the Z-axis in the 3D window.
- **Scale X:**
This can scale the fixture/object on the X-axis in the 3D window. The scale value is a factor where 1 is the default. A higher number makes the object bigger. A smaller number makes it smaller.
- **Scale Y:**
This can scale the fixture/object on the Y-axis in the 3D window. The scale value is a factor where 1 is the default. A higher number makes the object bigger. A smaller number makes it smaller.
- **Scale Z:**
This can scale the fixture/object on the Z-axis in the 3D window. The scale value is a factor where 1 is the default. A higher number makes the object bigger. A smaller number makes it smaller.
- **Gel Color:**
Here a color that will be added to the output from the fixture can be defined. It is useful for adding gels to conventional fixtures. This is visualized in the console and 3D Viewer.
- **Gel:**
This column does not currently have a function.
- **Note:**
This adds a multiline note to the fixture.
- **Beam Angle:**
Here a beam angle can be defined for the fixture. This is useful for conventional fixtures where different angles might be needed. This is visualized in the 3D window.
- **Cast Shadow:**
Each fixture casts a shadow or not. Turning this on makes the 3D Viewer look more real world, but it also increases the calculations needed for the 3D visualization.
- **Follow Target:**
Turning this to Yes makes the fixtures or 3D objects something that can be selected or picked using the **Follow** function in the 3D Viewer.
- **Selectable 3D:**
This is a Yes or No (text is hidden) field. Yes means that the fixture can be selected using the selection tool in the **3D Viewer**. Turning this off (No option) for stage elements that do not need to be controlled can be useful.
- **Visible 3D:**
This is a Yes or No (text is hidden) field. Yes means that it is shown in the **3D Viewer**.
- **Target Space:**
The MAker fixtures use this. This is the selected target space for the MAker fixture. It defines the available space where fixtures can move their light beams when selecting this MAker fixture. Learn more in the **MAker Fixture topic**.



- **Movement Space:**

The MAker fixtures use this. This is the selected movement space for the MArker fixture. It defines the space wherein the MAker fixture can move. Learn more in the [MAker Fixture topic](#).

- **Master React:**

This setting has three options: **None**, **Group**, and **Grand**. This defines if the intensity is affected by a master. If Grand is selected, then group masters are also affecting the fixture. If Group is selected, then they are not affected by the grandmaster. None makes the fixture ignore both groups and grandmasters.

- **Appearance:**

An appearance can be assigned to the fixture.

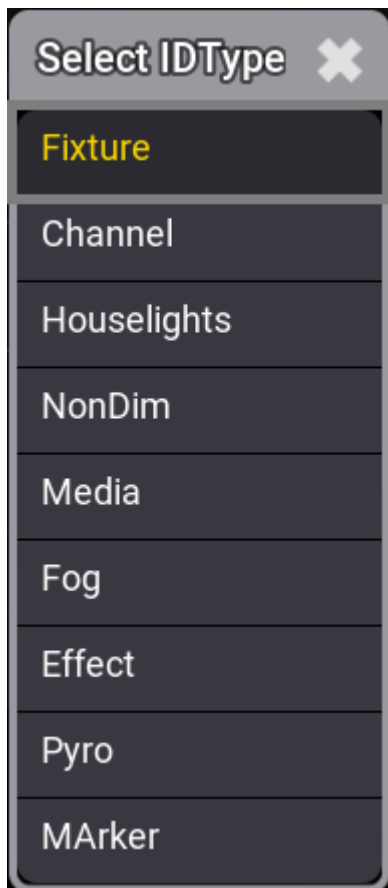
Assign an ID to Fixtures

The fixtures need at least one ID to be selected and controlled.

There are two types of ID numbers for each fixture. Fixtures can have both or just one of the two - but it needs at least one.

The **FID** is the default fixture ID. The number here is used with the [Fixture](#) keyword.

The **CID** is the second ID. This can be used if the **IDType** differs from "Fixture". Editing the IDType field opens the **Select IDType** pop-up.



Select the desired IDType

This lists the different valid IDTypes. Select one that is not "Fixture" to use the CID.



Important:

There are other hidden types called **Universal** and **Multipatch**. Universal is a special type the software automatically creates for a special universal fixture added to the patch. Multipatch is used for special multipatch versions of fixtures. These IDTypes cannot be used for normal fixtures.

1. Select the ID fields in one of the two ID columns (FID or CID) for the fixture rows where an ID is to be assigned. The selection order is important.
2. Type a number on the keyboard and assign the number by pressing **Please**.

Now the fixtures have an ID. The numbers are assigned sequentially based on the selection order.

Assign DMX Address to Fixtures

The fixtures need to be assigned a DMX universe and address before being able to create any output.



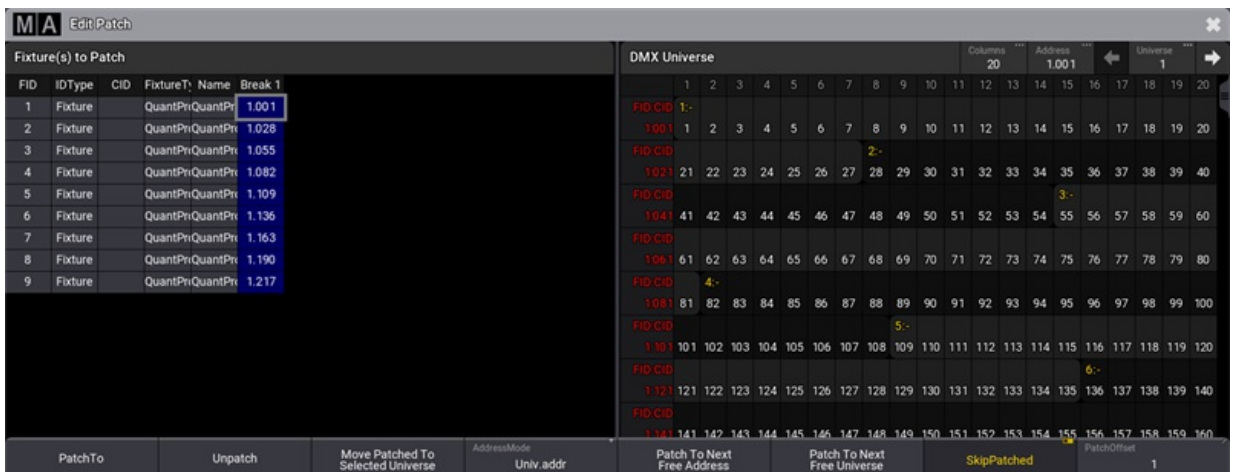
There are two primary options to give the fixtures an address. One is to just type the desired address directly in the patch field. The other is to use the dedicated **Edit Patch** pop-up.

Type the number:

1. Select the fields in the Patch column for the fixture rows where the address will be assigned. The selection order is important.
2. Type the DMX universe and address separated by a dot (for example, **2.1**) on the keyboard and assign the address by pressing **Please**.

Use the Edit Patch menu:

1. Select the fields in the Patch column for the fixture rows where the address will be assigned. The selection order is important.
2. Right-click the blue selected areas or tap the **Patch** button - this opens the **Edit Patch** pop-up.



Use the *Edit Patch pop-up* to get a visual overview of the DMX universe

This pop-up is divided into two sides. On the left side, there are all the selected fixtures. On the right side is a view with all the DMX universes and addresses. The purpose of the menu is that fixtures can be selected on the left side, and the universe list on the right side shows where there is available space in the universes. There are several ways to navigate universes and assign the fixtures to the selected DMX address.

The numbers on the right side are red if the universe is not granted.

The pop-up also opens a dedicated encoder toolbar:



Edit patch encoder toolbar is used for easy navigation

This toolbar can be used to select fixtures, universe, and address.



This is one way to do it:

1. Select the fixtures in the correct order.
2. Use the encoders to navigate to the desired address.
3. Tap the **Address** encoder to assign the fixtures to the address.

The Edit Patch pop-up has several buttons at the bottom:

- **PatchTo:**
Opens a small pop-up where the desired patch address can be typed.
- **Unpatch:**
This removes the patch address from the fixtures.
- **Move Patched To Selected Universe:**
This moves the fixtures to the selected universe and keeps the DMX address.
- **AddressMode:**
This toggle button changes how the DMX address is displayed. They can be split up into universes and a range DMX address from 1 through 512 - this option is called **Univ.addr**. The other option is **Absolute**, which shows the addresses continuously starting from number 1 and upwards.
- **Patch To Next Free Address:**
This gives the fixtures the next available DMX addresses.
- **Patch To Next Free Universe:**
This patches the fixtures to the next empty universe.
- **SkipPatched:**
This will skip addresses that have patched fixtures when scrolling through the universes.
- **PatchOffset:**
This can set the desired number of DMX channels between the fixtures. If the number is lower than the number of channels the fixtures use, they are patched as close as possible. If the number is more than the used channels, then the PatchOffset number is used.


The **Edit Patch** menu auto closes when all the selected fixtures are patched.

Filtering the Patch Menu

Filtering in the patch menu can be useful when there are a lot of fixtures and stage elements.

There are several ways to filter the fixtures in the patch menu.

An existing filter can be assigned using the **Filter** input in the title bar. Tapping this opens a small select pop-up that lists all existing filters (read about creating a filter in the **Create a Filter topic**). Two other options are **None** (no filter) and **New** (create new filter). Select the desired filter.

Filtering needs to be turned On for the filter to be active. Tap the filter icon () (between Filter and Columns) to turn it On. This button toggles On or Off.

Turning filters On also exposes column filters. An extra row is displayed above the fixture rows just below the column titles. Any column value can filter the patch simply by editing the filter field for a column. Only one filter value can be applied to each column.

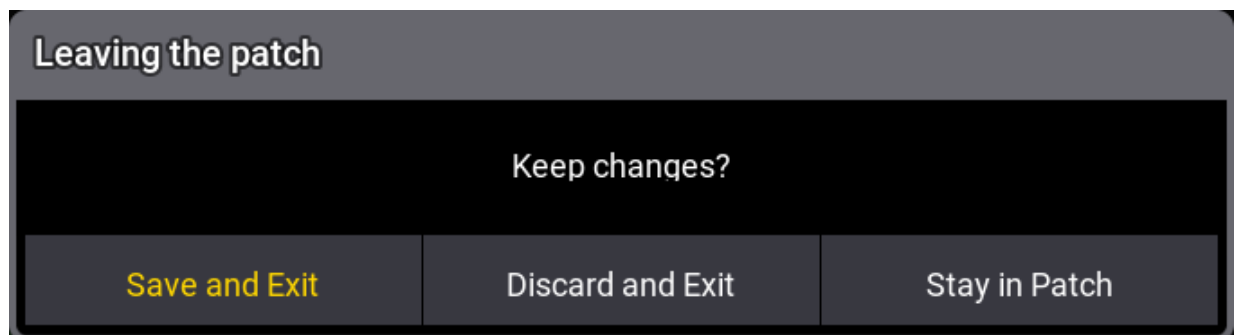
Edit a column filter field to assign a value. Some fields open a small selection pop-up giving a list of available values. Others are input fields where a text needs to be written.

The column filters can be combined with a filter from the pool.

The **Split View** is also a way to filter the patch. Learn more about the split view in the [Patch and Fixture Setup topic](#).

Closing the Patch Menu

There is a pop-up asking what to do when exiting the patch menu.



Leave patch pop-up with options

This asks if the changes should be kept.

Tap **Save and Exit** to save the changes and leave the patch menu. Tap **Discard and Exit** to cancel any changes and leave the patch menu. Tap **Stay in Patch** to stay in the patch menu.



13.3. Add Multipatch Fixtures

Multipatch fixtures are virtual fixtures where every parameter follows the primary fixture (the fixture they mirror).

These multipatch fixtures can be used when there are multiple physical fixtures controlled by the same ID. The virtual fixtures do not cost any extra parameters as long as the primary fixture has a patched and a granted DMX address. They can have their own patch address and there will be DMX output to these addresses. For more information about Granted in DMX Universes, see [DMX Universes](#).

The multipatch fixtures can be assigned to [layouts](#) and they can be positioned in the [3D](#). They are not part of the selection grid and can therefore not be used in MATricks.

Changing the parameters on a selected multipatch fixture actually changes the parameters of the primary fixture. All multipatched fixtures to this primary fixture will reflect the changed values.

Changing the location of the primary fixtures in the patch or adding multipatch fixtures to other fixtures might renumber all the multipatch fixtures.

Multipatch fixtures can be selected individually. The selection color is light red instead of yellow (default colors). In the fixture sheet, the primary ID will flash between yellow and light red when a multipatch fixture is selected.

They can be selected using the command line either using their unique multipatch CID or by an index number of the primary fixture.

Example

Have a setup like this:

FID		Name	FixtureType	Mode	Patch
None		Univ	1 Universal	1 Default	
1		QuantPro 1	10 MAC Quanti	1 Extended Mode	1.001
MP 1		<QuantPro 1>	<10 MAC Quan	<1 Extended Mode	6.001
MP 2		<QuantPro 1>	<10 MAC Quan	<1 Extended Mode	6.028
2		QuantPro 2	10 MAC Quanti	1 Extended Mode	1.028
MP 1		<QuantPro 2>	<10 MAC Quan	<1 Extended Mode	5.001
MP 2		<QuantPro 2>	<10 MAC Quan	<1 Extended Mode	5.028

Patch with two fixtures with two multipatched fixtures each



The first multipatch fixture on FID 2 has the hidden multipatch CID number of 3 - it is the third multipatch fixture in the entire patch.

It can be selected by this syntax: **Multipatch 3** or **Fixture 2 Multipatch 1**.



Hint:

The hidden multipatch CID should not be used to select the fixtures. The CID number will possibly change if changes are made to patch. The number is not fixed. It is recommended to use the index number based on the primary fixture (the second example above).

Learn more about the keyword in the **Multipatch keyword** topic.


Add Multipatch Fixtures

Requirement:

Have some fixtures patched to the show.

1. Enter the Patch menu by pressing **Menu** and then tap **Patch**.
2. Select the primary fixture that needs multipatched fixtures.
3. Tap **Create Multipatch**.
4. Enter the amount of needed multipatch fixtures in the **Amount of MP fixtures** pop-up and confirm it by tapping **Please**.

The new multipatched fixtures can now get a DMX patch address. The multipatch fixtures are limited in what can be changed, but name and location information is part of what can be changed for the fixtures.

5. When all needed information is assigned to the new multipatch fixtures, the patch menu can be closed by tapping  and saving the changes.

13.4. My Virtual Rig


MVR (My Virtual Rig) is a file format that is used to share data for a stage set up between a lighting console, a visualizer, a CAD program or similar tools. This allows for transferring parametric and geometric data between different programs.

It is a complementary system to the GDTF files.

MVR is described in the GDTF help pages (external link): <https://gdtf-share.com/help/en/help/mvr/index>

Import MVR

MVR files can be imported into a show file. This is done from the **Patch menu**. The MVR file needs to be in the correct folder for the software to find it. The folder is `../gma3_library/mvr`. It is the same location if the files are on a USB stick.

There is a button at the bottom of the patch menu called Import MVR. Tapping this opens an **Import MVR** pop-up. This is used to browse and select the MVR file. The source drive can be selected in the title bar by tapping the drive selection button (next to the )

Select the desired file in the list and tap **Import MVR** to import the MVR file.

The content of the MVR file is added to the patch and the information in the MVR files means that there might be added **Layers, Classes**, stage elements, or fixtures.

It is a good idea to store the show file **before** importing the MVR. In case it adds unwanted elements or in worst case corrupts the show.

Export MVR

Exporting the patch to an MVR file is done from the Patch menu. The file is created in the [folder with grandMA3]\shared\resource\lib_mvr folder on the selected drive.

At the bottom of the menu, there is a button called **Export MVR**. Tapping this opens an **Export MVR** pop-up. This can be used to select the desired drive and give the file a name. It also lists already existing MVR files on the selected drive.

The MVR file contains the entire patch including fixture files, stage elements, and organizational elements like the **Classes and Layers**.

It is a good idea to store the show file **before** exporting it to the MVR file.



Important:

In order to export an MVR file or GDTF file that contains user meshes and gobos, save the show file to the local hard drive before exporting.



13.5. Live Patch

The **Live Patch** menu is a version of the **Patch menu** with limited functionality. It looks like the patch menu, but some buttons are missing and some columns have a dark background - these columns cannot be edited.

The idea behind Live Patch is that it is only possible to change the elements that do not require a new show upload to the system.

This means that fixtures/objects cannot be added or removed.

In the live patch it is possible to change the following columns:

- Name
- Patch
- Pan Offset
- Tilt Offset
- Pan DMX Invert
- Tilt DMX Invert
- Pan Enc Invert
- Tilt Enc invert
- Position and Rotation
- Gel Color and Gel
- Note
- Beam Angle
- Cast Shadow and Follow Target
- Selectable 3D and Visible 3D
- Target Space and Movement Space
- Master React
- Values for automatic position calculation
- Appearance

The automatic position calculation columns are for the four points used for automatic fixture position calculation. Read more about this in the **Position Fixtures in the 3D Space** topic.

Follow the link to the Patch menu (above) to read more about the other columns.

Changes made in the live patch are changed immediately.

The lowest button on the left side gives access to an additional RDM Fixtures menu. This lists fixtures discovered using RDM. Learn more about RDM in the **RDM section**.



13.6. DMX Sheet

The DMX sheet displays the actual DMX output from the system. It shows the result from the sequences, programmer, incoming merged DMX, and any masters that might limit the output.

The sheet can be created as a window on any empty space on the screens. The minimum size is 1.5 squares wide and 2 high. It is created like any other window using the **add windows**.

MA		DMX																		Tester Encoder Bar	Only Selection	Address 1.001	←	Universe 1: Profile	→	Readout Percent
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20						
Attrib.	Sh1	Dim	Dim	R	G	B	C1	G1	G1 <-> G1 <->	G2	Prism	Iris	Zoom	Zoom	Focus	Focus	P	P	T							
1:001	12	50	0	0	0	0	0	0	50	0	0	0	0	50	0	50	0	50	0	35						
Attrib.	T	Ctrl1	FX1	FX1 R	FX2	FX2 R	FX Sy	Sh1	Dim	Dim	R	G	B	C1	G1	G1 <-> G1 <->	G2	Prism	Iris							
1:021	7	0	0	50	0	50	0	12	50	0	0	0	0	0	0	50	0	0	0	0						
Attrib.	Zoom	Zoom	Focus	Focus	P	P	T	T	Ctrl1	FX1	FX1 R	FX2	FX2 R	FX Sy	Sh1	Dim	Dim	R	G	B						
1:041	50	0	50	0	50	0	35	7	0	0	50	0	50	0	12	0	0	0	0	0						
Attrib.	C1	G1	G1 <-> G1 <->	G2	Prism	Iris	Zoom	Zoom	Focus	Focus	P	P	T	T	Ctrl1	FX1	FX1 R	FX2	FX2 R							
1:061	0	0	50	0	0	0	50	0	50	0	50	0	35	7	0	0	50	0	50							
Attrib.	FX Sy	Sh1	Dim	Dim	R	G	B	C1	G1	G1 <-> G1 <->	G2	Prism	Iris	Zoom	Zoom	Focus	Focus	P	P							
1:081	0	12	50	0	0	0	0	0	0	50	0	0	0	0	50	0	50	0	50	0						
Attrib.	T	T	Ctrl1	FX1	FX1 R	FX2	FX2 R	FX Sy	Sh1	Dim	Dim	R	G	B	C1	G1	G1 <-> G1 <->	G2	Prism							
1:101	35	7	0	0	50	0	50	0	12	50	0	0	0	0	0	0	50	0	0	0						

DMX 1.001 Fixture 1:- "QuantPro 1" Attribute = Sh1 Coarse

DMX sheet with patched fixtures



The main part of this window displays a big grid with every DMX address represented by its own square. This makes it a very long list (all 1024 universes are there).

The first column on the left side is a label for the second column.

There are two ways to see the DMX address: Absolute DMX address and the normal Universe divided. This can be changed in the **Sheet Settings**. The absolute address mode displays the DMX addresses as a continuous number. This means that the first address in the second universe will not be written as "2:001" but as "513" (512 addresses from the first universe + 1 from the second).

The top row displays the column number. The grid is a matrix, so the column number should be added to the address displayed in the vertical bar (except the first column).

Each square can display up to three different types of information:

- **Value:**
The DMX channels value.
- **Attribute:**
The attribute the DMX channel is controlling.
- **ID:**
The FID and CID assigned to the fixture patched at the DMX channel.

The three elements can be turned On or Off individually in the mask settings (read the **Sheets Settings** below for details).

If a fixture uses more than one DMX channel, the rounded corners are on the first and last attribute/DMX channel. The background color alternates between two shades of gray to indicate different fixtures.



Detail of the DMX sheet



In the example above, there are some RGB LED fixtures patched. The selected fixture (FID 201) gets a yellow text color in attributes and FID/CID.

If fixtures in the leftmost column have a red text color then the channels/universes are not granted. Read more about getting more parameters in the [Expand the Amount of Parameters topic](#).

If the cursor is moved to a square then the information about the square is displayed in a text line at the bottom of the window (see the image above as an example).

The two images show the two different ways the value can be visualized in the DMX sheet. The first is called **Background**, and the second is **Bar**. They can be changed or turned Off in the [Sheet Settings](#). The setting is called **Level Bar**.

The title bar of the window has some buttons.

Tester Encoder Bar can be toggled to show or hide the DMX tester encoder bar. Read more [below](#).

Only Selection jumps to the universe of the first selected fixture. The one called **Address** displays the number of the highlighted square. Tap this button to get an input box. Typing a new address in this box jumps to the typed address.

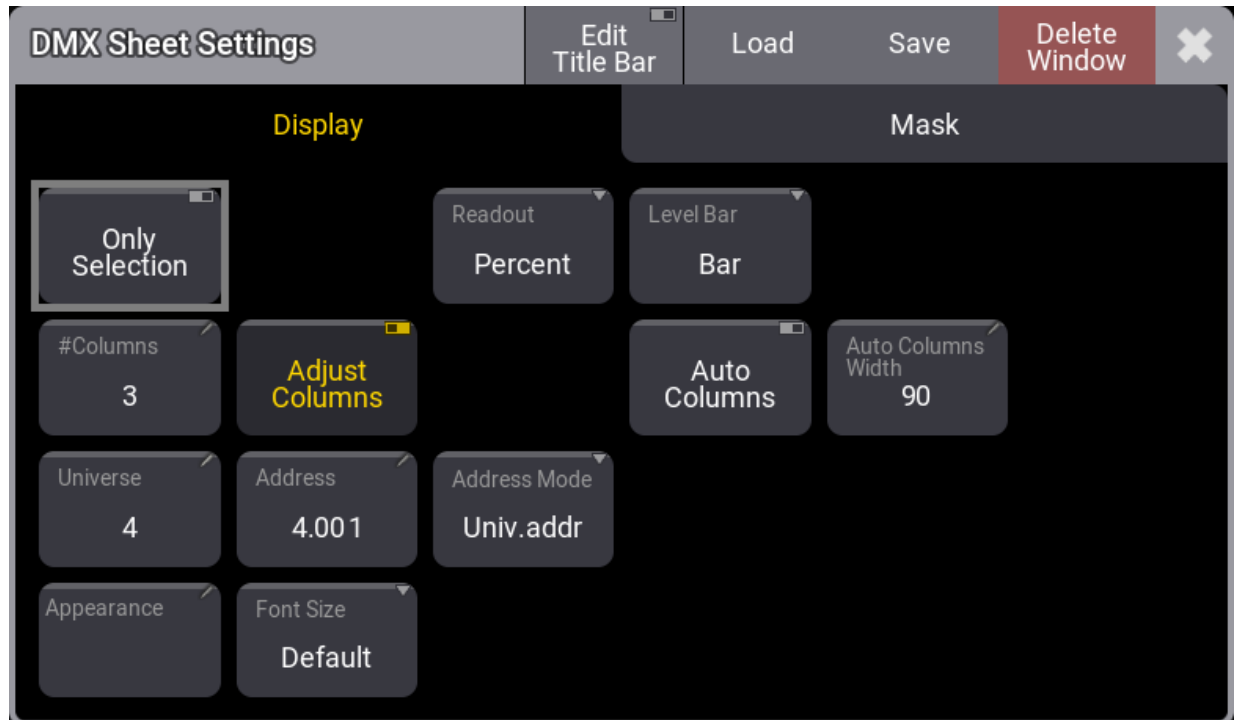
The button called **Universe** displays the current universe number. Tap this button to get an input box. Typing a new number in this box jumps to the universe. The two arrows jump one universe back or forward.

The final button in the title bar is the **Readout**. This can be used to change how the value readout is displayed. The options are Percent, Decimal, or Hex.

Sheet Settings

The DMX sheet has some settings that define how the sheet shows the data.

Tap the MA logo in the upper left corner of the window to open the settings. It opens the Display tab as a default.



DMX Sheet Settings - Display tab

Some of these settings are general settings shared with other windows. They are described in the [Window Settings topic](#).

Some settings are only for the DMX Sheet. This is a list of them with a short explanation:

- **Address:**
This input button opens a calculator where a DMX address can be typed. This address defines the address the sheet should begin with.
- **Address Mode:**
This toggle button changes how the DMX address is displayed. They can be split up into universes and a range DMX address from 1 through 512 - this option is called **Univ.addr**. The other option is called **Absolute**, which shows the addresses continuously starting from number 1 and upwards.
- **Auto Columns:**
This On/Off button activates a function that automatically adjusts the number of columns in the DMX Sheet. **Adjust Columns** need to be active for this to work. **#Columns** is automatically adjusted by this function.
- **Auto Columns Width:**
The **Auto Columns** function uses this value to determine how many columns the sheet can show.

- **Level Bar:**
This input field changes the level bar setting in the sheet. There are three options:
 - **Off:**
The value only visualizes the level.
 - **Background:**
The background changes from dark green to bright green as the value rises.
 - **Bar:**
The value is visualized as a bar that moves from left to right as the value rises.
- **Only Selection:**
This On/Off button changes the mode of the DMX sheet. This will limit the DMX sheet to only display the universes of the selected fixtures.
- **Universe:**
This input button can change the universe a DMX sheet should display. There is a special setting called **Selected**. This makes the DMX sheet scroll to the selected universe. This setting is also in the title bar.

The mask tab contains the three On/Off toggle buttons called **Value**, **Attribute**, and **ID**. They are described in the text above.

DMX Tester

The DMX channels can have output from the DMX tester.

This can be done using the **DMXAddress** and **DMXUniverse** keywords.

It can also be done using the **DMX Tester** encoder bar. It can be shown by turning on the **Tester Encoder Bar** in the title bar of the DMX Sheet.



Encoder bar - DMX Tester

This encoder bar allows fast access to DMX addresses, a test value, and the option to patch the selected address to an existing fixture. There is a row with buttons that gives fast access to some useful functions.

On the left side of the bottom row, there is a button that changes the address mode for the encoders. It can be **Uni** for **Universe** and **Address** separation on the first and second encoder or **Abs** for absolute DMX addressing on the second encoder.

The third encoder (Tester Output) defines the DMX value used for the test. Changing the readout for the DMX sheet also changes the readout for this encoder.

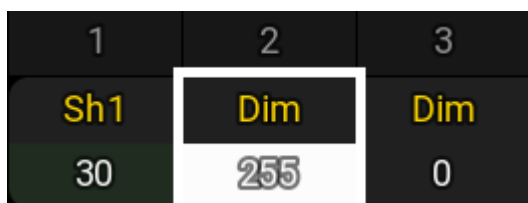
The fourth encoder can be tapped to open a **Fixture(s) to Patch** pop-up. This can be used to select an existing fixture that should be patched to the selected DMX address.

DMX addresses can also be selected by tapping on the DMX sheet.



The buttons above the encoder control have the following functions:

- **Retain:**
Turning this On will keep the value from the tester active in output for any DMX address selected using the tester. Retain Off releases the previous addresses from the tester.
- **DMX Channels:**
This button toggles between three different values:
 - **All:**
Any DMX address can be selected using the encoders.
 - **Patched:**
This allows only patched DMX addresses to be selected using the encoders.
 - **Unpatched:**
This allows only unpatched DMX addresses to be selected using the encoders.
- **Select All:**
Tapping this selects all the DMX addresses affected by the DMX tester.
- **Release Selected:**
This will release the currently selected DMX address from being affected by the DMX tester.
- **Release Unselected:**
This will release all other DMX addresses except the currently selected one from being affected by the DMX tester.
- **Release All:**
This will release all DMX addresses from being affected by the DMX tester.
- **Park Selected:**
This can park the currently selected DMX address at the current DMX value.
- **Unpark Selected:**
This can be used to unpark the currently selected DMX address.

When a DMX tester affects a DMX channel, the background color for the value is white.



DMX address 2 is being tested

The **DMX Universe pool** shows when a universe has parked DMX addresses with a small  icon. Universes, where DMX addresses are affected by the tester, have a small  icon.

It could look like this in the pool:



DMX universe pool with parked addresses and actively testing addresses



13.7. DMX Universes

Universe Menu

All the 1024 DMX universes are listed in the DMX Universes menu. This can be accessed by **Menu - Patch - DMX Universes**.

Patch	Lock	No	Name	Scribble	Appearance	Note	Merge Mode	Request	Granted	Coarse Params	Used
Fixture Types	S	1 (5 12)	Profile				Prio	Auto	Yes	273	13
	S	2 (5 12)	DmxUniverse 2			This is disabled	Prio	Off	Yes	192	16
	S	3 (5 12)	DmxUniverse 3				Prio	Auto	Yes	47	11
Attribute Definitions	S	4 (5 12)	DmxUniverse 4				Prio	Auto	Yes	300	100
	S	5 (5 12)	DmxUniverse 5				Prio	Auto	Yes	5	5
Parameter List	S	6 (5 12)	DmxUniverse 6				Prio	Auto	Yes	5	5
	S	7 (5 12)	DmxUniverse 7				Prio	Auto	Yes	5	5
DMX Universes	S	8 (5 12)	DmxUniverse 8				Prio	Auto	Yes	6	1
	S	9 (5 12)	DmxUniverse 9				Prio	Auto	Yes	5	5
Stages	S	10 (5 12)	DmxUniverse 10				Prio	Auto	Yes	0	0
	S	11 (5 12)	DmxUniverse 11				Prio	Auto	Yes	0	0
DMX Curves	S	12 (5 12)	DmxUniverse 12				Prio	Auto	Yes	0	0
	S	13 (5 12)	DmxUniverse 13				Prio	Auto	Yes	0	0
	S	14 (5 12)	DmxUniverse 14				Prio	Auto	Yes	0	0
	S	15 (5 12)	DmxUniverse 15				Prio	Auto	Yes	0	0
	S	16 (5 12)	DmxUniverse 16				Prio	Auto	Yes	0	0
	S	17 (5 12)	DmxUniverse 17				Prio	Auto	Yes	0	0
	S	18 (5 12)	DmxUniverse 18				Prio	Auto	Yes	0	0

DMX Universes menu

The menu has several columns showing information about each universe.

- **Lock:**
This shows if the row is locked.
- **No:**
This is the universe number.
- **Name:**
This is the name of the universe. It can be a good idea to give the universes a name for a better overview.
- **Scribble:**
A scribble can be assigned to the universe.
- **Appearance:**
An appearance can be assigned to the universe.
- **Note:**
A note can be added to the universe. Universes with a note have the note icon (📝) in the pool object - see below. Learn more in the [Notes topic](#).
- **Merge Mode:**
This is the merge mode used with incoming DMX data. These are described in the [DMX Port Configuration topic](#).

- **Request:**

A universe can be requested or not. Requesting a universe means a desire to send DMX data to the universe. This request might not be granted - generally because there are not enough unlocked parameters (read more in [Expand the Amount of Parameters topic](#)). The setting has three options:

 - **Auto** (default): This automatically requests the universe if something is patched.
 - **On**: This requests the universe.
 - **Off**: This does not request the universe. It can be helpful to turn off the request if there are not enough unlocked parameters, but a test is needed for a specific universe, for instance, during preprogramming.
- **Granted:**

This shows if the output of the universe is granted. It says "Yes" if the universe is requested and granted. This field is information only.
- **Coarse Params:**

This is the number of parameters patched to the universe.
- **Used:**

This is the number of fixtures patched to the universe.

Universe Pool

The universe pool has a pool object for all the 1024 universes that the system can handle.

The pool is a window created like any other window. Learn how in the [Add Windows topic](#).



DMX universe pool

Each pool object represents a DMX universe.

The pool object shows some information about the universe.

There is a colored bar at the top of the pool object. This is green if the universe is requested and granted. It flashes bright green when the DMX data is outputting. It is red if it is not granted.

If a universe is assigned to a DMX port, there is a small port icon. This can have different colors depending on whether the port is set to output (green), input (orange), output and input (bright cyan), if it is output with RDM enabled (dark cyan), or if it does not output DMX (black).

There are indicators to show if the universe is output using sACN (a small "S") or Art-Net (a small "A"). These flash green when the DMX output values are updated. See the difference on universe 4 and 6 in the example image above.

Indicators also show if DMX data is merged into the universe from Art-Net or sACN. This is indicated with an orange text: **Art-Net In** and **sACN In**. This is a dark orange text to show the assignment. It flashes orange when there is an incoming signal. See universe 5 and 7 in the example image above.

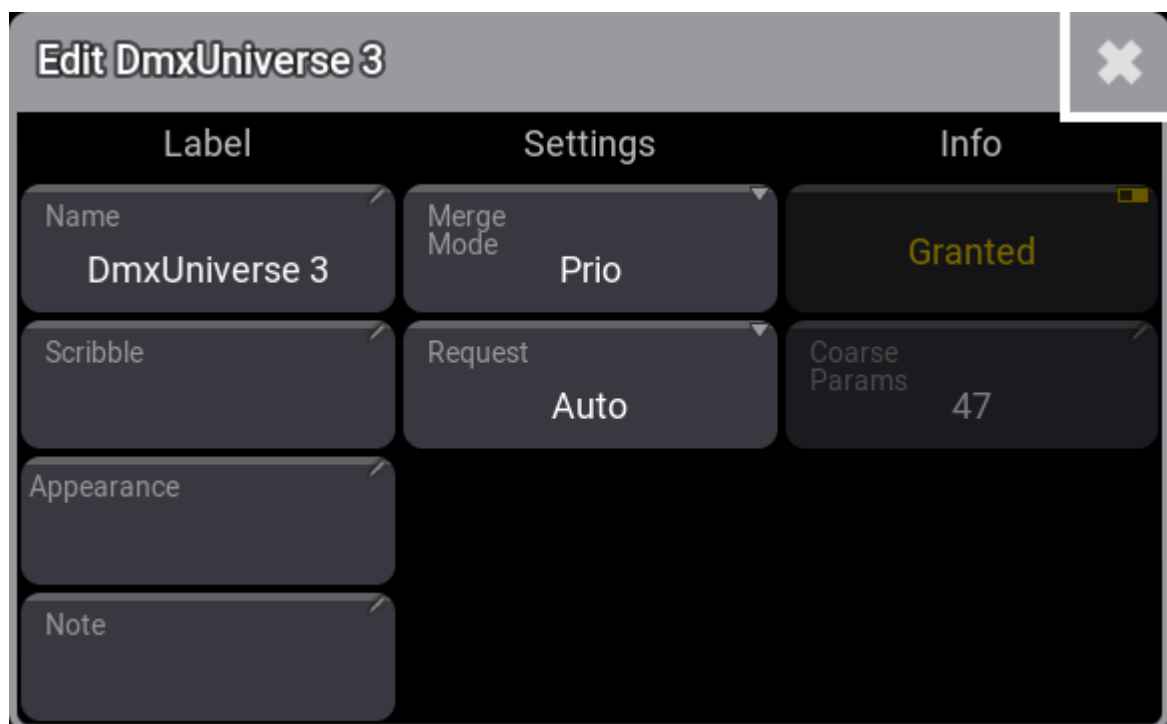
The final indicator is the PSN In. This appears on universes with MArker fixtures mapped to a PSN source. It also flashes orange when a PSN signal is received. Learn more about mapping PSN in the [PSN topic](#).

Read the topics in the [DMX In and Out section](#) to learn how to set up the DMX ports and how to set up Art-Net and sACN.

Moving a universe in the pool also changes the universe for the fixtures patch in the universe but keeps the respective DMX start addresses.

A universe can be selected by tapping it in the pool or using the [Select keyword](#) together with the [DMXuniverse keyword](#). The [DMX Sheet](#) can be set up to only show the selected universe in the sheet settings - using the universe setting.

Editing a pool object opens the Edit DMX Universe pop-up:



Edit DMX Universe pop-up

Here it is possible to edit the same settings as in the DMX Universe menu.



13.8. Remove Fixtures from the Show

Removing or deleting a fixture row in the **Patch menu** removes all programmed information from the show.

It can not be done from the Live Patch menu.

Removing the DMX address or the ID number does not remove the programmed value, but the fixture can no longer be edited and it does not produce any output. It is also not visible in the **Fixture Sheet**.

Remove a Fixture from the Patch

Requirement: Patched fixtures.

1. Open the Patch Menu.
 1. Press **Menu**
 2. Tap **Patch**
2. Select the fixtures in the patch.
3. Tap **Delete**
4. Close the Patch menu and tap **Ok** to accept the changes.

The fixtures are deleted and all programming with those fixtures is lost. It cannot be oopsed.



13.9. Position Fixtures in the 3D Space

The **3D Window** shows a 3D visualization of the fixtures and objects in a 3D space.

The fixtures need to be positioned in the 3D space for this to truly be a powerful tool. The virtual fixtures should be positioned and rotated to match the real-world fixtures.

There are three primary ways to change the fixture position: **Using the patch**, **using the 3D window**, or **position calibration**.

Position Fixtures Using the Patch

The best way to position the fixtures from the patch is from the **Live Patch**:

1. Press **Menu**.
2. Tap **Live Patch**.
3. Make sure the menu is in **Full** column mode.

In the patch, there are rows for each fixture and there are columns with position and rotation values:

Pos			Rot			
X	Y	Z	X	Y	Z	Z
0.000	0.000	0.000	0.00	0.00	0.00	0.00
0.000	0.000	0.000	0.00	0.00	0.00	0.00
0.000	0.000	0.000	0.00	0.00	0.00	0.00
0.000	0.000	0.000	0.00	0.00	0.00	0.00
0.000	0.000	0.000	0.00	0.00	0.00	0.00
0.000	0.000	0.000	0.00	0.00	0.00	0.00

Position and Rotation values in the Patch.

New fixtures are always added in the zero point location and with the zero rotation.

The zero points are 0.000 meters for all three position axis (X, Y, and Z) and 0.00° for all three rotation axis.

- X position is usually regarded as the Stage Left and Stage Right indication. A positive value is in the stage left direction.
- Y position is usually the Downstage and Upstage direction. A positive value would move the fixture more upstage.
- Z position is the height. A positive value would move the fixture above the stage.
- X rotation is rotating the fixture around the fixtures' own X-axis. A positive value is rotating the top of the fixture towards downstage.
- Y rotation is rotating around the fixtures' own Y-axis. A positive value rotates the top towards stage left.
- Z rotation is rotating around the fixtures' own Z-axis. A positive value rotates the fixture counter-clockwise seen from the top.

The grandMA3 software currently only works with meters and degrees.

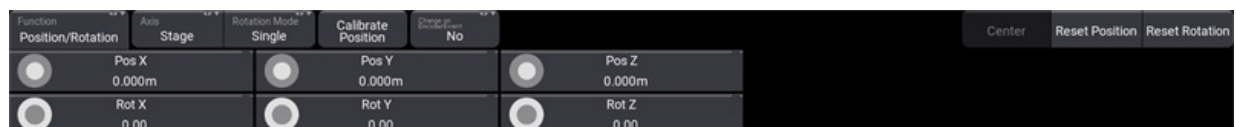
Do the following steps to edit the fixtures position and rotation:

1. Locate the rows for the fixtures that need to be positioned.
2. Edit the needed fields.
3. Type the new position.

Position Fixtures Using the 3D Window

The 3D window can be used to position and rotate the fixtures. The window needs to be in **Setup** mode. This can be changed by tapping **Mode** in the title bar.

This mode changes the encoder toolbar into this:



Use the encoders to position and rotate fixtures

There are different buttons in this toolbar that changes how the fixtures are affected by the encoders. Some are standard functions that do not change, others change depending on selected functions.

The general workflow is this:

1. Select the fixtures that need to be moved.
2. Change the settings in the toolbar to match the wanted action.
3. Use the encoders to adjust the values.



The values on the encoders show the relevant values depending on the selected fixtures. Turning the encoders changes the values and thus the position or rotation of the selected fixtures.

These are the different standard buttons for position and rotation:

- **Function:**
This button toggles between two different modes: **Position/Rotation** and **Arrangement**. This changes the function of the encoders to either change the position and rotation of the selected fixtures or to different arrangement types. Read about the Arrangement tool [below](#).
- **Axis:**
The axis is used when a fixture is moved or rotated. The two options here are **Stage** and **Object**. This means that the selected fixtures can be positioned and rotated using their own axis (Object) or the stage or world axis (Stage).
- **Rotation Mode:**
When multiple fixtures are selected, then rotation mode can be changed between **Single** or **Group**. This defines if the rotation is done for each fixture or if the selection of fixtures is treated as a group, and thus rotated as one object.
- **Calibrate Position:**
This opens the Position Calibration menu. Read more about it [below](#).
- **Change on EncoderEvent:**
This determines when changes are distributed through the network to other stations. When it is set to **Yes**, then the new position is sent immediately. When it is **No**, then the new information is sent two seconds after the encoders stop turning.
- **Reset Position:**
Tap this button to reset the fixture position to 0 in all positions.
- **Reset Rotation:**
Tap this button to reset the rotation to 0 on all axis.



Position Arrangement Tool

The position arrangement tool is a set of three different arrangement or layout types. It can be used to arrange the selected objects in a **Line**, **Grid**, or **Circle**.

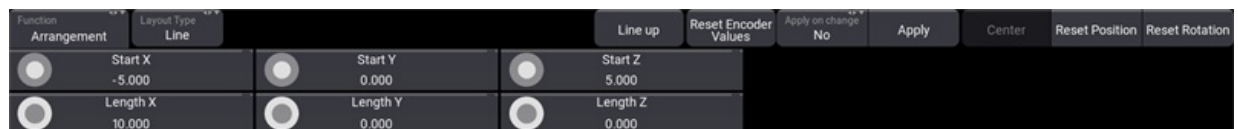
The tool is accessed by changing the Function in the encoder toolbar to **Arrangement**. Each of the three layout types has its own encoder functions and their own buttons. There are some common arrangement buttons:

- Layout Type:**
 There are three types of arrangement types. **Line**, **Grid**, and **Circle**. These different types change the available buttons in the encoder bar and the functions of the encoders.
- Reset Encoder Values:**
 Resets the values on the encoder to the default values.
- Apply on Change:**
 This controls if arrangement changes are applied and distributed immediately (Yes) or if they are only marked by a purple indicator of the would-be location (No).
- Apply:**
 If **Apply on Change** is set to No, then this button needs to be tapped to confirm the new arrangement location.

When an arrangement is being adjusted, then there are purple fixtures in the 3D window with ID numbers to indicate where the fixtures would end up if the arrangement settings are applied.

Line

This is used to position the fixture on a single row.

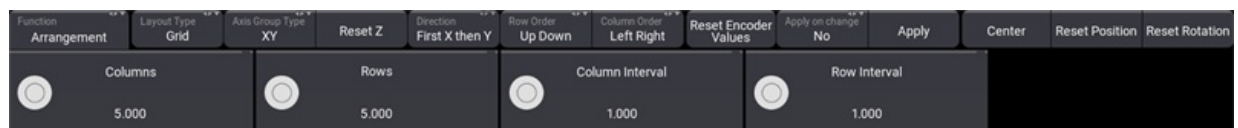


Line Arrangement Encoder Toolbar

The encoders change to set a start and a length for all three axes. There is a special button called **Line up**. Tap this to align the base of the fixture to match the line.

Grid

The grid arrangement moves the fixtures into rows and columns. It is a 2D grid.



Grid Arrangement Encoder Toolbar

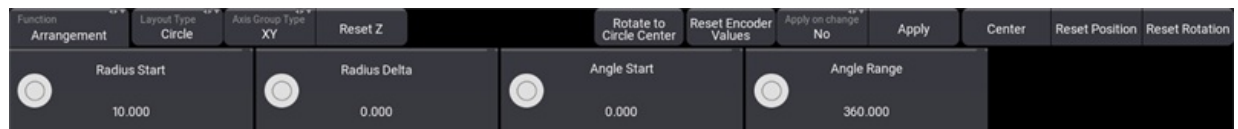
The encoders change to set the number of columns and rows and the interval spacing between the columns and rows. There are some special buttons:



- **Axis Group Type:**
This sets the orientation of the grid. It shows the two axes that are used as columns and rows.
- **Reset Z / Reset X / Reset Y:**
This button resets the position on the third axis. The one not affected by the grid.
- **Direction:**
This changes the direction of the grid. It changes the order of the two axes.
- **Row Order:**
Tap this button to reverse the direction of the row.
- **Column Order:**
Tap this button to reverse the direction of the row.

Circle

The circle arrangement tool is used to position fixtures in circles and spirals.



Circle Arrangement Encoder Toolbar

The encoders change to set the Radius Start size, Radius Delta (sets if the radius changes size), Angle Start, and Angle Range. Angle Start sets where the first fixture is on the circle. Angle Range is used when creating semi-circles (value below 360) or even multiple circles (value above 360).

There are some special buttons:

- **Axis Group Type:**
This sets the orientation of the circle. It shows the two axes that are used.
- **Reset Z / Reset X / Reset Y:**
This button resets the position on the third axis. The one not affected by the circle.
- **Rotate to Circle Center:**
This rotates the fixture bases to match the circle. Tapping this button several times rotates the fixtures each time to a different rotation.



Position Fixtures Using Position Calibration

The fixture position calibration system calculates the fixture position based on pan and tilt values needed to hit three or four known points on the real-world space. Using only points one, two, and three can be enough but the best result is achieved using all four points.

The four points do not need to be the same for all fixtures. Each fixture can have its own four points. The points are visible in the 3D window as a red, green, blue, and yellow octahedron when the window is in setup mode and the **Position Calibration** pop-up is open.

The pop-up is opened by tapping the **Calibrate Position** on the encoder bar - read [above](#).

Position Calibration																		
FID	Name	P1					P2					P3					P4	
		X	Y	Z	Pan	Tilt	X	Y	Z	Pan	Tilt	X	Y	Z	Pan	Tilt	X	
1	AS QWO 1	-5.000	-5.000	0.000	47	29	-5.000	5.000	0.000	22	34	5.000	5.000	0.000	4	26	5.000	-5
2	AS QWO 2	-5.000	-5.000	0.000	45	32	-5.000	5.000	0.000	20	30	5.000	5.000	0.000	6	25	5.000	-5
3	AS QWO 3	-5.000	-5.000	0.000	43	37	-5.000	5.000	0.000	20	27	5.000	5.000	0.000	8	24	5.000	-5
4	AS QWO 4	-5.000	-5.000	0.000	44	27	-5.000	5.000	0.000	26	31	5.000	5.000	0.000	5	28	5.000	-5
5	AS QWO 5	-5.000	-5.000	0.000	42	30	-5.000	5.000	0.000	23	28	5.000	5.000	0.000	8	26	5.000	-5
6	AS QWO 6	-5.000	-5.000	0.000	39	33	-5.000	5.000	0.000	22	26	5.000	5.000	0.000	9	25	5.000	-5
7	AS QWO 7	-5.000	-5.000	0.000	42	26	-5.000	5.000	0.000	28	28	5.000	5.000	0.000	7	31	5.000	-5
8	AS QWO 8	-5.000	-5.000	0.000	40	28	-5.000	5.000	0.000	26	26	5.000	5.000	0.000	10	28	5.000	-5
9	AS QWO 9	-5.000	-5.000	0.000	37	29	-5.000	5.000	0.000	24	25	5.000	5.000	0.000	11	26	5.000	-5
10	AS QWO 10	-5.000	-5.000	0.000	41	24	-5.000	5.000	0.000	29	26	5.000	5.000	0.000	11	34	5.000	-5
11	AS QWO 11	-5.000	-5.000	0.000	39	26	-5.000	5.000	0.000	27	25	5.000	5.000	0.000	13	30	5.000	-5
12	AS QWO 12	-5.000	-5.000	0.000	36	26	-5.000	5.000	0.000	25	23	5.000	5.000	0.000	14	27	5.000	-5

Store P1
Call P1
Store P2
Call P2
Store P3
Call P3
Store P4
Call P4
Solve

Position calibration pop-up with some selected fixtures and their values

The pop-up shows the selected fixtures as rows and four colored sections of columns.

Each color section has X, Y, and Z values for the calibration point and Pan and Tilt values needed to hit the point.

At the bottom, there are buttons to store and recall the pan and tilt values for each point. There is also the **Solve** needed to start the position calculation.

To record the pan/tilt position that matches the calibration point, tap **Store Px**. **Call Px** can be used to recall a stored position to refine it.

Hint:
 It is recommended to use four calibration points. Using three calibration points deteriorates the quality of the calibration.



To calibrate fixtures, use this workflow:

1. Open the **Position Calibration** pop-up.
2. Select the fixtures to calibrate.
3. Define the real-world coordinates of the calibration points (P1 - P4) in the position calibration pop-up for each fixture.
4. Position each fixture using pan and tilt to each of the calibration points, and store these positions.
5. Tap **Solve**.
6. Close the pop-up.

Now the fixtures should move and rotate in the 3D window to match the real-world values.

Using the Command Line

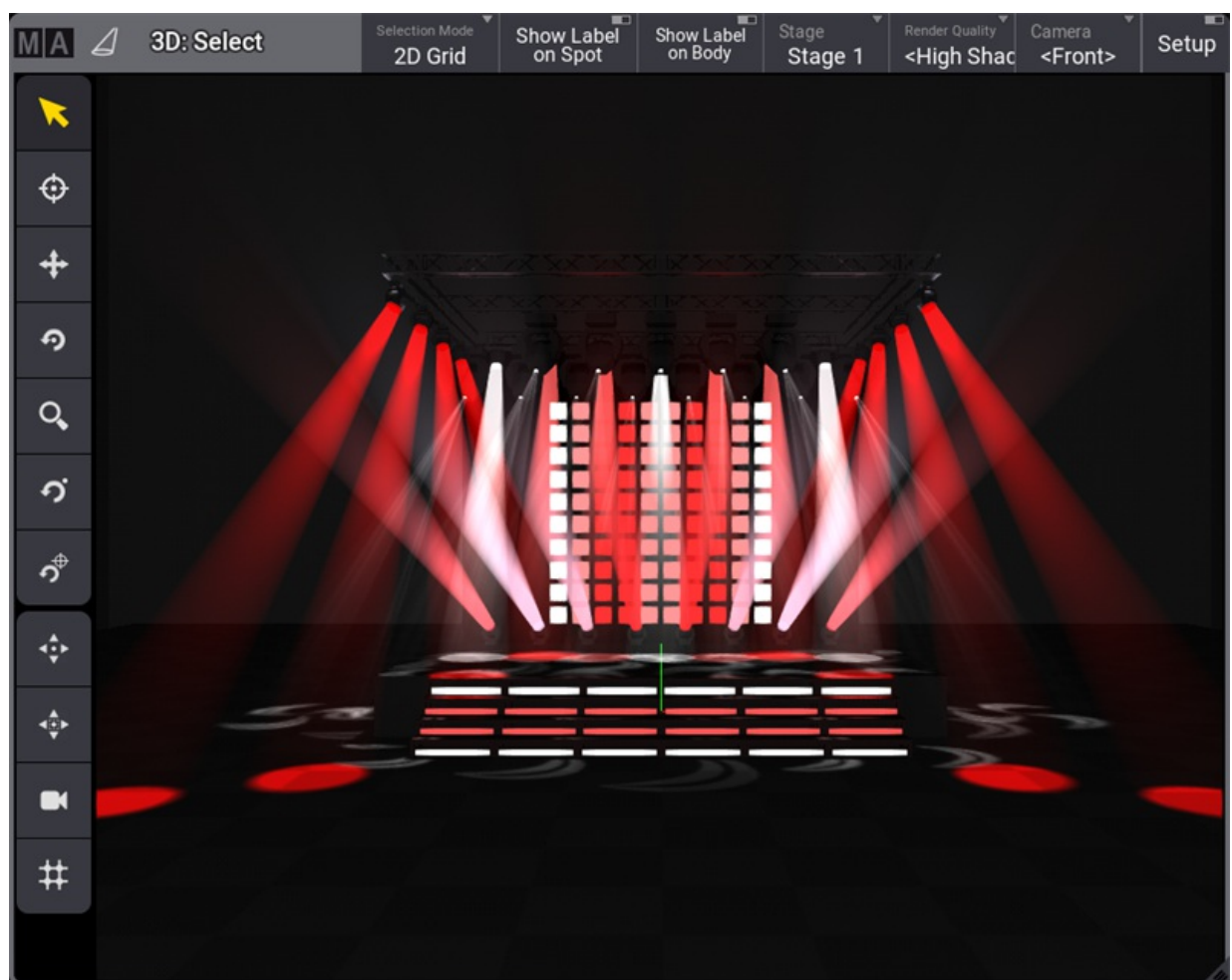
Storing calibration points, calling calibration points, and solving the calibration can be done using the **Action keyword**.

13.10.3D Viewer

The 3D Viewer shows a 3D visualization of the virtual space where fixtures and 3D elements can be positioned and rotated.

The fixtures can project a light beam that moves and changes color when the values for the fixtures are changed.

All the fixtures and other stage objects can be **positioned and rotated** in the **Patch**, **Live Patch**, or using this window.



The 3D Viewer with some fixtures

The **Stage** object can be visualized as a box or just a floor. The box can be looked into, but not out of. Read more **below**.

Fixtures can be interacted with in the 3D Viewer like the **Fixture Sheet**. For instance, it can be used in combinations with commands.

There are virtual cameras that are used to see the 3D space. These define the position, direction, and other settings from where and how the 3D space is viewed. The cameras are stored in the **Camera Pool**.



Restriction:

When on an onPC station, the driver of the GPU has issues with OpenGL, the 3D Viewer informs the user about the driver issue instead. In this case, the 3D Viewer does not try to render 3D scenes anymore.

Title Buttons

There are several default buttons in the title bar. The available buttons in the title bar can be edited using the window settings. Learn more in the **Title Bar Configuration topic**.

Most of the default buttons are swipe buttons that open a list of the available options.

Three toggle buttons turn On or Off labels and turn On or Off the Setup mode.

The others give fast access to select Selection Mode, Stage object, Render Quality, and Camera selection.

There can be several stages in the patch. Each 3D Viewer can show one of these stages or all stages. Read more about stages in the **Stages topic**.

Read about all the settings **below**.



Left Side Tool Buttons











There can be a toolbar on the left-hand side of the window. These tools control what happens when the window is touched and change camera positions (read more about the cameras below).

The ones that change the touch mode are tapped to select the mode, and when the window is touched, the mode dictates what happens.

The selected mode is displayed with a yellow color in the window title bar. Most tools are grayed out and disabled if the selected camera is locked.

This is the explanation of the different tool buttons:

-  **(Select):**
Sets the touch mode to **Select**. This is used to select fixtures in the window. The fixtures can be tapped or selected using a selection lasso.
-  **(Follow):**
Sets the touch mode to **Follow**. This makes all the selected moving lights point to the touched stage area. The follow function obeys the **Align** settings.

-  **(Move):**
The mode is set to **Camera Move**. Move means changing the camera's position without changing the camera's pan and tilt values. The scroll wheel on a mouse moves the camera forward and backward in the view based on the location of the pointer (not necessarily the center).
-  **(Rotate Center):**
Sets the mode to **Camera Orbit**. This mode orbits the camera around the center of the window, keeping it pointed toward the center. The scroll wheel on a mouse moves the camera closer to or away from the location of the pointer (not necessarily the center).
-  **(Zoom):**
Changes the mode to **Camera Zoom**. This mode moves the camera in and out of the 3D Viewer. This can also be done in the other modes using a scroll wheel on a mouse.
-  **(Rotate Pivot):**
Changes the mode to **Camera Pivot**. This pivots the camera around a pivot point that does not have to be the center. The pivot point can be set (read about the next button) and is remembered until a new point is set. The scroll wheel functions just like the rotate center mode.
-  **(Set Pivot):**
This button is used to set a new pivot point in the window. The rotate pivot mode is selected as soon as the point has been set (by clicking or touching the window).
-  **(Focus)** - only available when **Setup** is On:
This tool can focus static fixtures (fixtures without position attributes). Select the desired fixture, tap this tool, and tap the location in the 3D Viewer where the fixture should be pointed. This tool affects the rotation values of the fixture.
-  **(Fit):**
Tapping this button moves the camera to fit the entire stage area into the view.
-  **(Fit Selected):**
Tapping this button moves the camera to fit the selected fixtures into the view without orbiting the camera.
-  **(Camera):**
This button reloads the selected camera to the settings set in the **camera pool object**.
-  **(3D view to Selection Grid):**
Tapping this tool creates grid values for the **Selection Grid**, using the current view of the fixtures.

Moving the Camera

The cameras are pool objects in the **Camera Pool**. A camera can be moved to different locations and pointed in different directions.

This information can be edited in the pool, but the position and direction are easier to change in the 3D Viewer - read about the left-hand side tool buttons above.

A scroll wheel on a mouse moves the camera closer to or away from the location of the pointer.



Editing the camera in the pool allows changing the camera mode and type. Read more in the camera pool topic (link above).

Moving the Fixtures

The fixtures' position and rotation can be set in the **patch** or **live patch**. But they can also be positioned live in the 3D Viewer.

Read more about this in the **Position Fixtures in the 3D Space** topic.

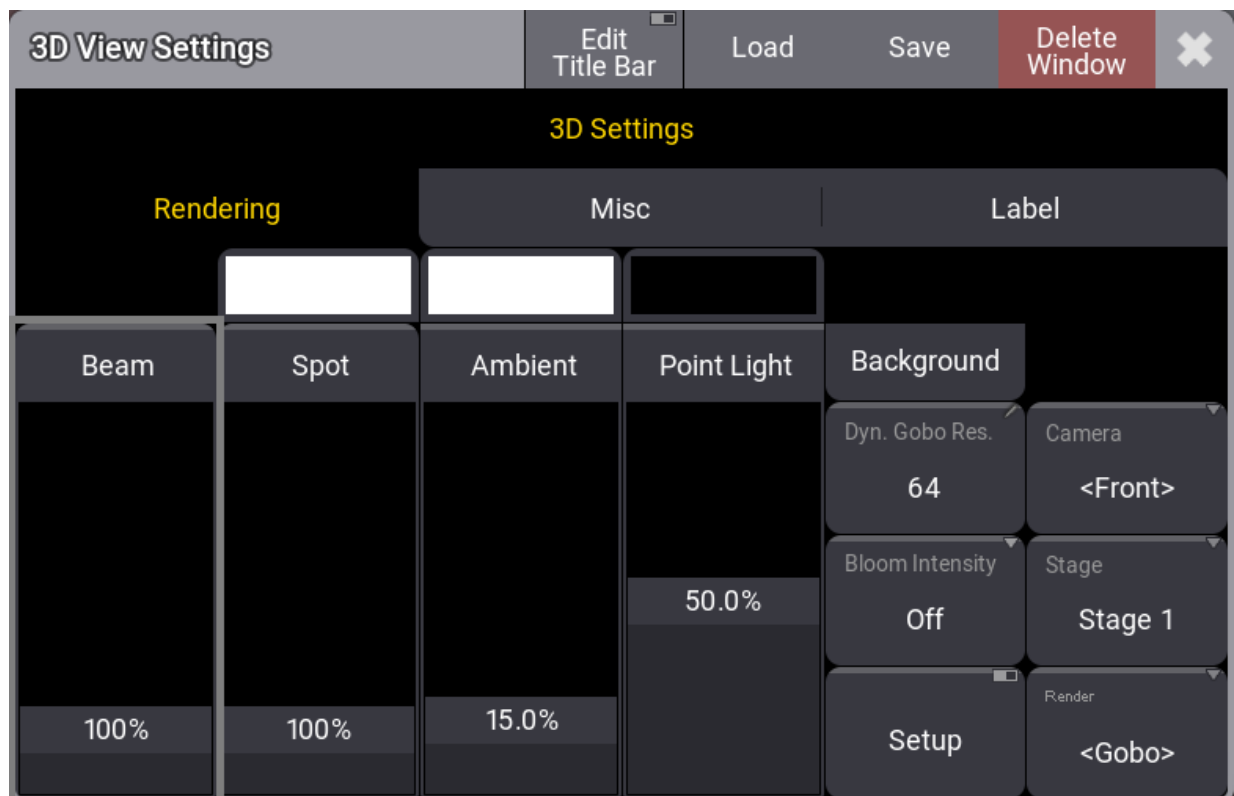
3D Viewer Settings

The settings can be opened by tapping the MA logo in the upper left corner of the 3D Viewer.

This opens a settings pop-up. This pop-up has tabs called **Rendering**, **Misc**, and **Label**.

Rendering Settings

The rendering settings are about light levels, colors, and rendering quality.



Rendering tab in 3D View Settings

There are four on-screen faders:

- **Beam:**
This is the visibility of the light beam from all fixtures.
- **Spot:**
This is the general intensity of the visualization of the light beam reflection where it hits a surface.
- **Ambient:**
This is the ambient light level inside and outside the stage box. It is a very diffuse light that removes some of the contrast in the 3D Viewer.
- **Point Light:**
This is a light source from the direction of the camera. It is used to light up the elements in the 3D space.



Hint:

All fixtures' intensities are set following the fixture type with the most lumen, the brightest lamp visualized in the 3D Viewer. As a result, all other fixture types are displayed a bit darker to demonstrate the output related to the brightest fixture type.

The ambient and point light can be colored, similar to putting a gel in front of the light. The area above the faders can be tapped to open a color selector pop-up.

The **Background** can also be colored. The ambient light must be turned up for this color to be visible.

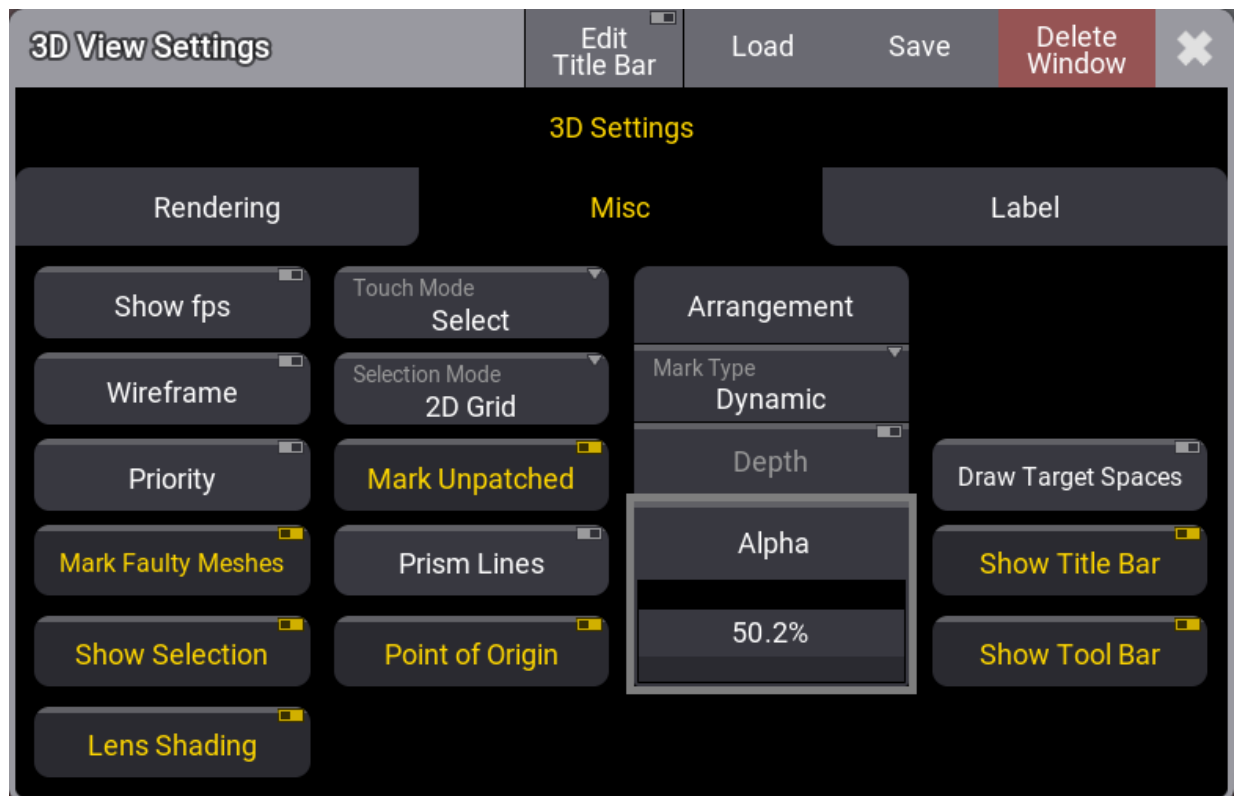
There are some buttons:

- **Dynamic Gobo Resolution (Dyn. Gobo Res.):**
This is the resolution of dynamically created gobos.
- **Bloom Intensity:**
The blooming effect can be turned On or Off. Tapping this button toggles between the two.
- **Setup:**
Turning the setup On makes it possible to move the fixtures and 3D objects using the 3D Viewer and encoder bar. Read more about this in the [Position Fixtures in the 3D Space topic](#).
- **Camera:**
This is used to select one of the cameras from the camera pool. If it is set to follow the selected camera, the name is inside angled brackets. Read more about cameras in the [Camera Pool topic](#).
- **Stage:**
This selects what stage the 3D Viewer display. A single stage can be selected, or all stages can be displayed. Read more about stages in the [Stages topic](#).
- **Render:**
This is used to select one of the render qualities. Read more in the [Render Quality topic](#).

Some of these are swipe buttons, so remember that the options can be reached easily by swiping out of the button.

Misc Settings

The second tab is called **Misc**.



Misc tab in the 3D View Settings

The Misc. tab has the following settings:

- **Show fps:**
Turning this On displays frames per second information in the upper right corner of the window.
- **Wireframe:**
Turning this On shows the 3D Viewer as a wireframe instead of a shaded view.
- **Priority:**
Turning this On gives the 3D Viewer a high priority. It is only recommended to turn this On when used on a computer with a high-quality graphics card. It takes away resources from a console interface, making it react slower to user input.
- **Mark Faulty Meshes:**
This marks meshes that are unloaded or faulty.
- **Show Selection:**
Selected fixtures are marked with a yellow body color for the primary fixtures and a light red color for multipatch fixtures when this is On. Learn more about multipatch in the [Add Multipatch Fixtures topic](#).

- **Lens Shading:**

This defines whether the selection is shown on the lens of a geometry type “Beam”. The shaded selection is drawn when it is On.

- **Touch Mode:**

Touching and swiping in the 3D Viewer can interact with the window differently. This setting defines how.

Tapping it toggles through the following options: **Select**, **Follow**, **Focus**(Only valid when Setup is On), **Camera Orbit**, **Camera Zoom**, **Camera Pivot**, **Camera Move**, and **Camera Set Pivot**.

The Camera options move the camera - Read more [above](#). The **Select** option is used to select fixtures in the view.

- **Selection Mode:**

This defines how the fixtures are selected and positioned in the **Selection Grid**. The selection mode has two different options:

- **2D Grid:**

The 2D grid selection offers two different modes, **Planar** and **Perspective**. Using the lasso selection to select fixtures in the 3D Viewer adds them as a two-dimensional selection to the selection grid. In planar mode, the camera position does not affect the selection order. Only the real position of the fixtures is significant. In perspective mode, the camera's orientation also influences the selection's order in the selection grid.

To select fixtures in the 3D Viewer using the planar selection, draw a lasso starting with a horizontal or vertical mouse movement. The color of the lasso changes to green when the selection is locked to planar mode. To select fixtures in perspective mode, draw a lasso starting with a diagonal mouse movement. The color of the lasso changes to cyan when the selection is locked to perspective mode.

- **Linearize:**

The selection is linearized to only the X-axis of the selection grid depending on which direction the selection lasso was created (top/bottom - left/right). This selection mode is indicated with a yellow lasso.



Hint:

The projection distortion of the camera of the 3D Viewer may affect the position of the fixtures in the selection grid if the perspective selection is used. To prevent this, use a 2D camera in the 3D Viewer.

- **Mark Unpatched:**

This marks the body of unpatched fixtures with a dark red color when On.

- **Prism Lines:**

This shows a line for each prism facet when the **Beam Quality** is set to **Line**.

- **Point of Origin:**

This shows a point of origin axis marker. It is typically at the zero point of the stage.



- **Arrangement:**

This is a collection of settings for visualizing the purple arrangement "ghosts" markers. They are seen when using the arrangement tool to position fixtures.

 - **Mark Type:**

There are two options: **Dynamic** or **Small**. Dynamic shows a box matching the size of the fixture. Small is just a small square marker.
 - **Depth:**

Set if the depth of the 3D space is considered when showing the marks. This could lead to marks being hidden behind objects. If the depth is switched off, the marks are always drawn in front of other objects.
 - **Alpha:**

Define the transparency of the marker objects.
- **Draw Target Spaces:**

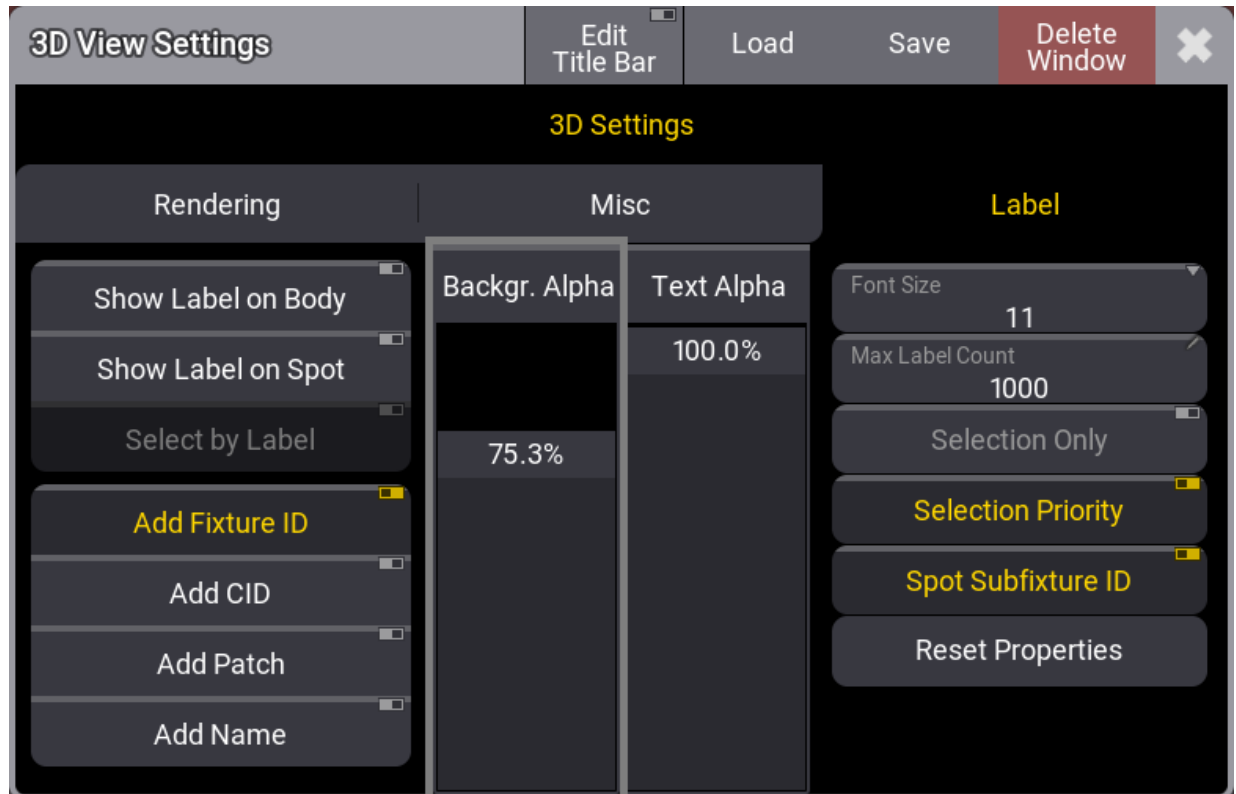
This setting show or hide a square wireframe for MArker fixtures' target spaces.
- **Show Title Bar:**

This setting shows or hides the title bar for the window.
- **Show Tool Bar:**

This shows or hides the toolbar on the left-hand side. The toolbar is always visible when **Setup** is On.

Label Settings

The tab is called Label. It is settings about the labels that can be turned On or Off for the spots and the fixture bodies.



Label tab in the 3D View Settings

- **Show Label on Body:**
This On / Off button enables labels to be drawn on the fixture's body.
- **Show Label on Spot:**
This On / Off button enables labels to be drawn in the center of the fixture's spot. A fixture can also be selected using the rubber band selection on a label.
- **Select by Label:**
This setting is possible when **Show Label on Spot** is active. It makes it possible to select fixtures by selecting the spot labels in the 3D Viewer when this setting is On.
- **Add Fixture ID / Add CID / Add Patch / Add Name:**
These On/ Off buttons define the information displayed on the label.
- **Background Alpha:**
This on-screen fader sets the transparency of the label's background.
- **Text Alpha:**
This on-screen fader sets the transparency of the text displayed on the label.
- **Font Size:**
This button sets the font size of the text displayed on the label. Tapping this toggles through the available Font Sizes. It can be swiped to open a small Select Label Font Size pop-up with all the size options.
- **Max Label Count:**
This input field sets the maximum count of labels that are displayed at the same time. If the number of labels exceeds the maximum label count, the labels closest to the camera will be displayed.



- **Selection Only:**
This On / Off button defines whether labels are displayed for all fixtures or only for selected or partly selected fixtures.
- **Selection Priority:**
This On / Off button defines whether labels of selected fixtures are displayed on top of not selected fixtures.
- **Spot Subfixture ID:**
This On / Off button defines if the label of the spots shows the Fixture ID of the corresponding sub fixture (On). Otherwise, all spots of a fixture show the Fixture ID of the main fixture (Off).
- **Reset Properties:**
Tap this button to reset the label properties to their factory defaults.

Setting Up a 3D Computer

The grandMA3 consoles can show a 3D Viewer with the fixtures and the stage setup. But the consoles are optimized to be the human interface used to program the light.

If a high-quality and high-framerate 3D render machine is needed, the best solution is to have a high-performing computer with good graphics cards. This computer must have the grandMA3 onPC installed and optimized for 3D graphics.

The grandMA3 onPC needs to be in a session with the console. The grandMA3 onPC could be logged in as the default 3D user who uses a different **Screen Configuration** (read about screen configurations in the [User Settings](#) topic) but is the same user profile as the default admin user. The grandMA3 onPC should be logged in with a user with the same profile as the user looking at the 3D computer. This ensures that the 3D computer follows the user into preview mode.

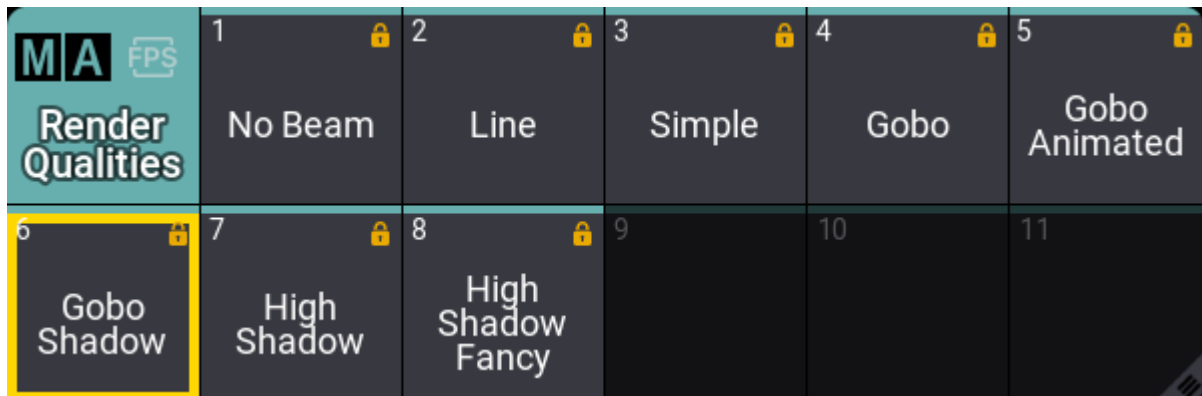
The 3D computer could have one big 3D Viewer with all the render settings set to the best quality, and the 3D Viewer setting called **Priority** should be On (see settings above).

The window can also be set to hide the title bar to maximize 3D viewing space using the [Window Settings](#). Other elements, like the view buttons, encoder bar, etc., can also be hidden using the [Configure Display pop-up](#).

13.11. Render Quality

The render quality defines how the light and fixture bodies are rendered in the 3D Viewer.

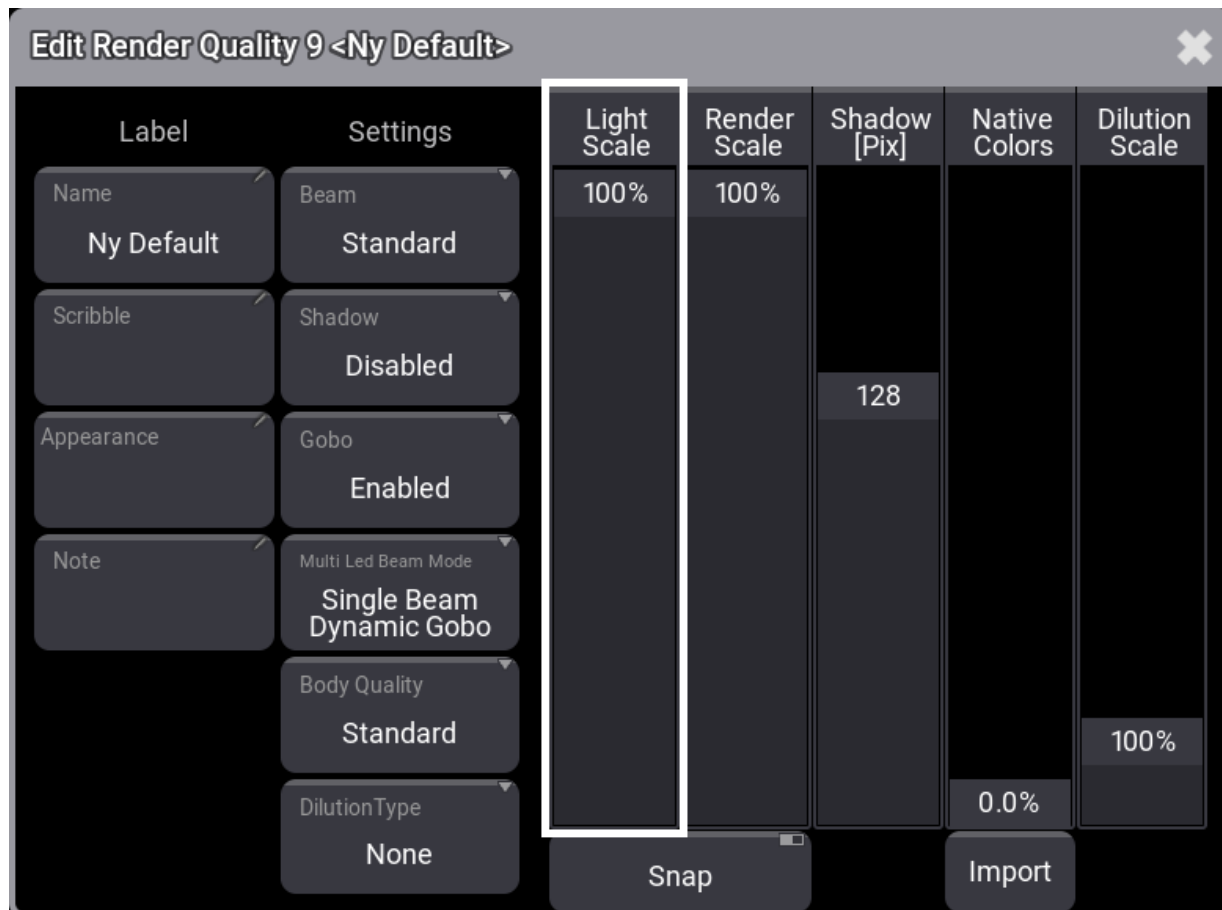
All the render settings are stored in a Render Quality pool object and can be easily selected in the pool.



RenderQualities pool

There are some predefined render qualities in the pool. There is always a selected object. The 3D Viewer can use any render qualities or be linked to the selected pool object.

New pool objects can be made and edited to change the render settings.



Render Quality Editor

The editor has the following settings:

- **Name:**
This is the name of the pool object. Tapping it allows changing the name.
- **Scribble:**
This is the assigned scribble, and tapping it allows select a scribble or create a new one.
- **Appearance:**
This is the assigned appearance. Tapping it allows select a scribble or create a new one.
- **Note:**
A note can be added to a render quality.



- **Beam:**

This toggles through a list of different simulation qualities that can be selected for the light beam. It goes from low to high quality. The higher quality uses more computer resources.

 - **No Beam:**

This does not render any beams. This mode does not show any gobos or spot rendering.
 - **Line:**

This renders a simple line as the beam. This mode does not show any gobos or spot rendering.
 - **Standard:**

This is a simple light beam from the fixture.
 - **High:**

This mode adds more reality to the light beam.
 - **High Fancy:**

This is a more precise calculation of light dissipation.
- **Shadow:**

Turns On or Off if the beam shall throw shadows when hitting another object. Therefore the 3D object needs the setting **Cast Shadow** enabled within the patch.
- **Gobo:**

This setting defines whether gobos shall be rendered within the beam and spot. It has three different options:

 - **Disabled:**

Gobos are not rendered.
 - **Enabled:**

Gobos are rendered.
 - **Animated:**

This means that in addition to the gobo, gobo animations like gobo shake or gobo wheel spin are rendered. When gobo is only set to enabled, these animations will not be rendered.
- **Multi LED Beam Mode:**

This defines how multi-emitter fixtures render the light beam. The options are:

 - **Separated Beams:**

This renders a beam for each emitter.
 - **Single Beam Mean Color:**

This renders a single beam using the mean color.
 - **Single Beam Dynamic Gobo:**

This takes the output of the emitters and creates a "virtual dynamic gobo". This is then rendered as a single beam.



- **Body Quality:**

This defines the quality of the fixture body rendering. The options are

- **None:**

No meshes of fixtures are visualized. Nevertheless, environmental objects are displayed.

- **Box:**

A whole fixture's body is visualized as one box. This box has the dimensions of the fixture.

- **Low:**

Every geometry of a fixture is visualized as a box. This is not changed if one of the fixture's meshes is a default cylinder.

- **Simple:**

Fixtures with a common geometrical structure of a moving head (Base - Yoke - Head - Lens with or without Pigtail) are visualized with grandMA3 default meshes. All other fixtures are visualized in "Low" mode.

- **Standard:**

Fixtures with a vertex count above 1 200 are visualized in "Simple" quality mode. The meshes of fixtures with a vertex count lower than 1 200 are not changed.

- **High:**

Fixtures with a vertex count above 10 000 are visualized in "Standard" quality mode. The meshes of fixtures with a vertex count lower than 10 000 are not changed.

- **Ultra:**

The fixtures are visualized with their original meshes, no matter the vertex count.

- **Dilution Type:**

This defines the dilution intensity of the beam and the spot regarding the distance. These are the options:

- **None:**

The beam and spot keep their brightness over the distance.

- **Linear:**

Corresponds to a linear attenuation of the intensity over the distance.

- **Correct:**

Corresponds to a more realistic attenuation.

- **Light Scale:**

This changes the resolution of the light rendering in the 3D Viewer continuously. 100% means that the 3D Viewer is rendered with its native resolution. 10% means that the resolution of the light in the 3D Viewer is divided by 5 in width and height.

- **Render Scale:**

This changes the resolution of the whole rendering in the 3D Viewer continuously. 100% means that the 3D Viewer is rendered with its native resolution. 10% means that the resolution of the 3D Viewer is divided by 5 in width and height.

- **Shadow [Pix]:**

This changes the resolution of the rendered shadows. The higher this value is, the higher the resolution of the shadows. This only affects the rendering if the property "Shadow" is enabled, for example, in the Beam Quality "Gobo Shadow".



- **Native Colors:**

This fader interpolates between the native representation of colors and the intensities of the beams and spots. An automatic method that changes the colors and intensities so that the rendered beams are easier to separate. A higher value for native colors can make distinguishing between individual overlapping beams impossible but provides stronger colors and higher contrasts.

- **Dilution Scale:**

This defines the beam intensity dilution of the beam dilution. The higher the value, the less the beam is attenuated, resulting in a beam that remains visible over further distance.

- **Snap:**

If "Snap" is enabled, the Render Scale and Light Scale are divided with integer values. The fader then indicates the scale (1/1 to 1/5).

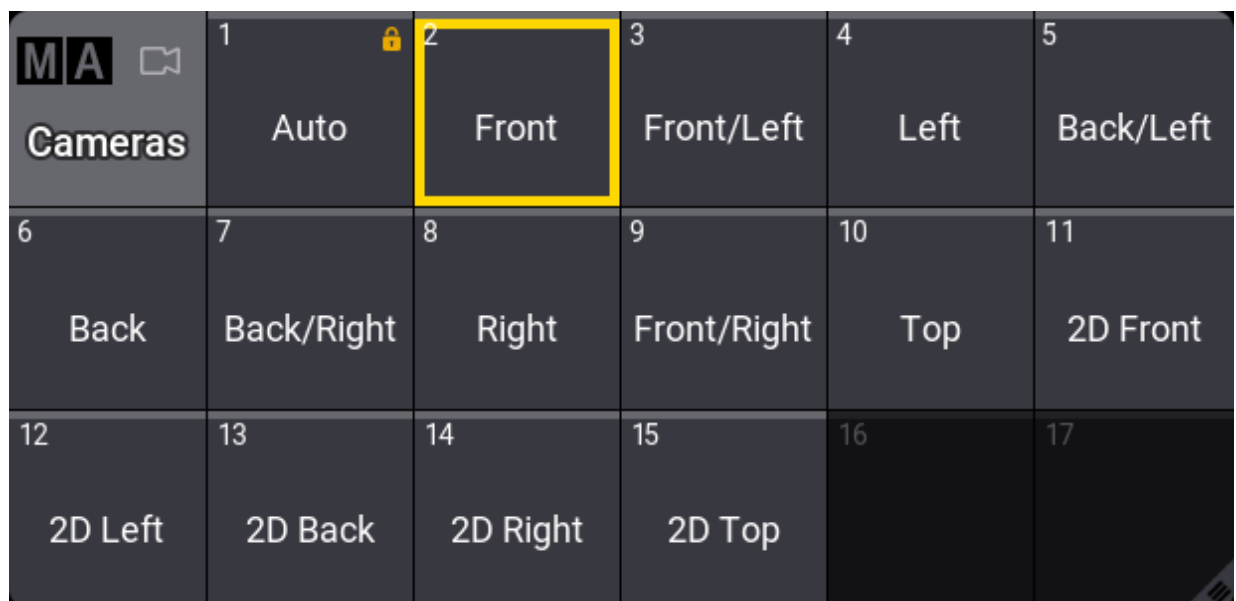
Render Scale and Light Scale can be used in combination. Setting both to 1/2 means that the whole 3D Viewer is rendered with half of its resolution, while the light is rendered with a quarter of its resolution.

13.12. Camera Pool

The cameras are used in the **3D window** to see the fixtures in the 3D virtual stage.

Multiple cameras can be arranged to see fixtures and 3D objects from different angles and with different camera settings.

The cameras are all in the **Camera Pool**.

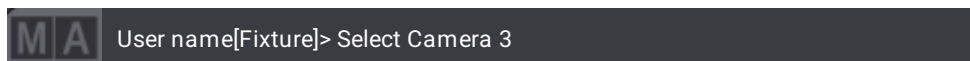


Camera pool with some cameras and a selected camera

There are some default cameras in the pool in a new show. Most of these can be edited, but the "Auto" camera is locked.

The "Auto" camera is a special camera that zooms and pans to put the selected fixtures in the center of the 3D window. When no fixtures are selected, it shows the entire stage.

A camera can be selected by tapping it in the pool or using the command line. For example, selecting camera number 3 can be done with this command:



The selected camera has a yellow frame.

Editing an empty pool object creates a new camera that copies the currently selected camera.

Editing an existing camera object opens the **Edit Camera** pop-up:



Label	Settings	
Name Front	CameraMode 3D	Pos Z 2.633m
Scribble	FOV 45.00	Pan 0.00
Appearance	Pos X -0.014m	Tilt 0.00
Note	Pos Y - 11.466m	Zoom 0.50

Edit Camera pop-up for the front camera.

This pop-up contains all the values needed for the camera.

Changing the values here changes the camera.

- **Name:**
This input box can be used to change the name of the camera.
- **Scribble:**
A scribble can be assigned to the camera pool object.
- **Appearance:**
An appearance can be assigned to the camera pool object.
- **Note:**
A note can be added to the camera.
- **Camera Mode:**
The cameras can have a 3D perspective or a flat 2D projection. Editing this field allows selecting one of the predefined 2D camera angles. Selecting a 2D camera disables some of the other values in this pop-up. The 3D camera mode gives access to all values in this pop-up.
- **FOV (Field Of View):**
The Field of view or Field of vision describes wide or narrow the camera looks at the 3D stage.
- **Pos X:**
This is the camera position on the X-axis.
- **Pos Y:**
The camera's position on the Y-axis.



- **Pos Z:**
The camera's position on the Z-axis.
- **Pan:**
This value pans the camera. Positive values turn the camera clockwise.
- **Tilt:**
This tilts the camera up and down. A positive value points the camera down.
- **Zoom:**
The zoom value is very useful with 2D cameras. The zoom value goes from 0.01 to 1. It goes from narrow to wide.

Moving the camera and changing the pan and tilt values directly in the 3D window might be easier. There are nice tools for manipulating the camera. Read more about this in the [3D window](#) topic.

The values are applied to the camera when **Reload Camera** is tapped.

Close the pop-up by tapping the  in the upper right corner of the pop-up.



13.13. Stages in grandMA3

In grandMA3 there are virtual stages. The fixtures **added to the patch** are placed in a stage. The fixtures can be **positioned** in this 3D virtual stage.

It is possible to create more stages. This could be useful in a house with several physical stages, a television station with several studios, a theme park with several different areas, or for adding a festival rig to a touring show.

The stages have location and dimension information.

Add a Stage in the Patch

1. Press **Menu**.
2. Tap **Patch** in the menu pop-up.
3. Tap **Stages** on the left side.
4. Tap **Insert New Stage** at the bottom.

A new stage is now added above the line that had focus (yellow frame).

Patch	Lock	No	Name	Appearance Right	Appearance Left	Appearance Front	Appearance Back	Appearance Top	Appearance Bottom	Pos X	Pos Y	Pos Z	Rot X	Rot Y	Rot Z	Min X	Max X
S		1 (2)	Stage 1							0.000m	0.000m	0.000m	0.00	0.00	0.00	-15.000	15.000
S		2 (2)	Stage 2							20.000m	0.000m	0.000m	0.00	0.00	0.00	-5.000	5.000
			New Stage														

Stages menu with two stages and a global stage

The stages can be named for organizational purposes.

The standard size for a new stage is 30 meters wide (X) and deep (Y), with a zero in the middle. The height (Z) is from zero to 15 meters high.


The stage displayed in the 3D window will automatically expand the visible box but will not change the dimension defined for the stage object.

The stages' position and rotation can be changed in the stage setup. It is possible for a 3D window to show one or all stages.

The buttons at the bottom make it possible to insert new stages.



The stages can also be deleted. Be careful with this!

 **Important:**
 Fixtures are patched inside stages. Deleting a stage also deletes all the fixtures in that stage, including all the information programmed about those fixtures.

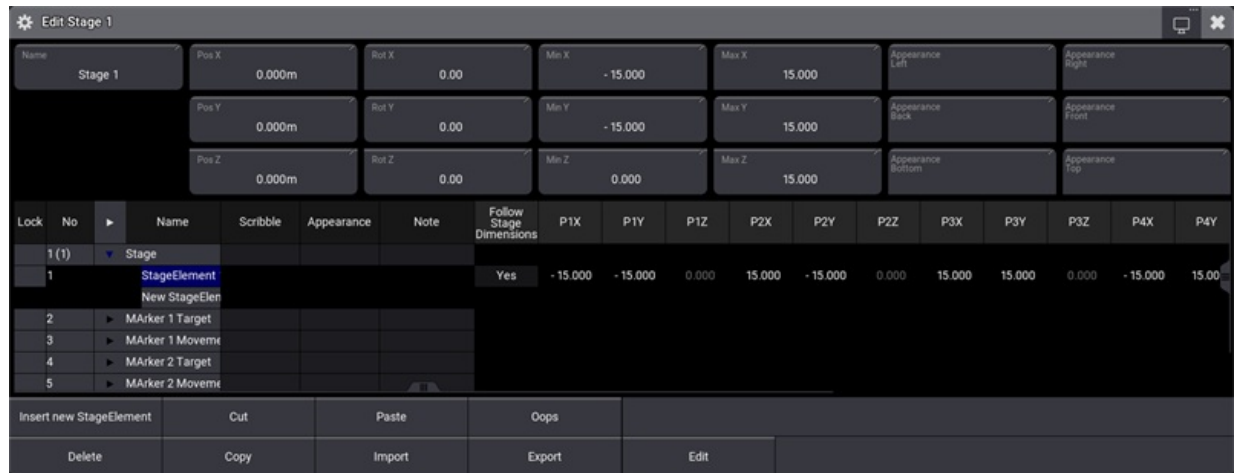
If the stages need to be reorganized, they can be copied, cut, and pasted to move them in the list.

A stage can also be exported and imported. Remember that the stage contains the fixtures. The exported stage object contains the patched fixtures and any other stage objects inside the stage.

For more information about adding fixtures to the stage, please follow the first link at the top.

Edit a Stage

A selected stage in the stage menu can be edited by tapping **Edit** at the bottom. This opens the **Edit Stage** menu.



Edit stage menu for stage 1

The top part of this edit has input fields for the same settings as the **Stage** menu.

The lower part is used to add and edit a stage **Space** object and stage **Elements**. The elements are flat surfaces that can be combined to create boxes or areas that the light can hit.

A stage space defines a volume in the stage. This space is relevant when using XYZ programming. It defines the size of the space the fixtures can point to. Read more about this in the [XYZ section](#).

A stage space can have multiple stage elements inside. These elements are defined by four corners, each with an X, Y, and Z coordinate.



Stage spaces and elements can be created, deleted, copied, cut, pasted, oopsed, imported, and exported like many other objects.

Export and Import are especially nice if a lot of time has been spent on creating a 3D set.



Hint:

More complex 3D objects can be added to the Patch. They are found in the "Set" manufacturer.

A new stage always has a Stage Space and a default Stage Element (the "floor").

Adding MArker fixtures automatically adds stage spaces for Target and, if needed, for the Movement. Read more about this in the [MArker Fixture topic](#).



13.14. Classes and Layers

Fixtures can be organized and grouped in both **Class** and **Layer**.

It is inherited from the MVR (My Virtual Rig) import and from the CAD drawing programs where fixtures can be organized using classes and layers.

In grandMA3 there is no real difference between a class or a layer.

Each fixture can have one of each assigned.

The class and/or layer can be used when applying filters in the patch.

Classes and layers can be created manually.

Create a Class or Layer

The following description is for a layer, but the process is the same for classes. Just exchange all the places it says "Layer" or "FixtureLayer" with "Class".

1. Press **Menu**.
2. Tap **Patch** in the menu pop-up -> this opens the patch menu.
3. Activate **Full** columns in the title bar.
4. Activate **Split View** in the title bar.
5. Tap **Layers**.
6. Tap the location in the list where the new layer should be or tap **New FixtureLayer** at the bottom of the list.
7. Tap **Insert new FixtureLayer**.

A new class or layer is now added.

The name can be changed.

Classes and layers can also be created by editing the class or layer field for the fixture. See the description below, but instead of selecting an existing class or layer, tap **New**. This creates a new class or layer, and the name pop-up appears to give the new class or layer a name.

Assign a Class or Layer to a Fixture

Classes and layers can be assigned to fixtures in the patch menu when it is in **Full** column mode or when the **Split View** is active in layers or classes.

When the column mode is full, they also appear as options in the Fixture Wizard (read about it following the link above).

Editing the class or layer cells opens small select pop-ups where the existing classes or layers can be selected. There is also the option to select **None**. This can be used to remove a class or layer from a fixture.



13.15. Attribute Definitions

Attributes are the building blocks of **fixture types**. The same building blocks are used throughout the console and they are what is controlled using the **Encoder bar** when **operating fixtures**.

Attributes definitions describe the relation between Main Attributes and sub-attributes. Attributes are organized in **Activation Groups**, **Feature Groups**, and **Deactivation Groups**.

There is a set of 366 predefined attributes with a complete setup and relations. They can be edited but it is also possible to create custom attributes.

It is all done from the sub menus in the patch menu.

Lock	No	Name	Pretty	Main Attribute	Activation Group	Feature	Special	Special Index	Physical Unit	Geometry Type	Color	Intensity	Natural Readout	Encoder Resolution
	1	Dimmer	Dim			1 'Dimmer', 1 'Dimmer'	Dimmer	0	LuminousIn	None	0.000000,0.000	1.00000000	Percent	Coarse
	2	Pan	P		1 'PanTilt'	2 'Position', 1 'PanTilt'	PanTilt	0	Angle	None	0.000000,0.000	1.00000000	Physical	Coarse
	3	Tilt	T		1 'PanTilt'	2 'Position', 1 'PanTilt'	PanTilt	1	Angle	None	0.000000,0.000	1.00000000	Physical	Coarse
	4	PanTiltDistance	Dist		1 'PanTilt'	2 'Position', 1 'PanTilt'	PanTilt	2	Length	None	0.000000,0.000	1.00000000	Physical	Coarse
	5	PanRotate	P Rotate			2 'Position', 1 'PanTilt'	None	0	AngularSpe	None	0.000000,0.000	1.00000000	Physical	Coarse
	6	TiltRotate	T Rotate			2 'Position', 1 'PanTilt'	None	0	AngularSpe	None	0.000000,0.000	1.00000000	Physical	Coarse
	7	PositionEffect	Pos FX			2 'Position', 1 'PanTilt'	None	0	None	None	0.000000,0.000	1.00000000	Percent	Coarse
	8	PositionEffectRate	Pos FX Rate			2 'Position', 1 'PanTilt'	None	0	None	None	0.000000,0.000	1.00000000	Percent	Coarse
	9	PositionEffectFade	Pos FX Fade			2 'Position', 1 'PanTilt'	None	0	None	None	0.000000,0.000	1.00000000	Percent	Coarse
	10	XYZ_X	X		2 'XYZ'	2 'Position', 2 'XYZ'	XYZ_Pos	0	None	None	0.000000,0.000	1.00000000	Percent	Coarse
	11	XYZ_Y	Y		2 'XYZ'	2 'Position', 2 'XYZ'	XYZ_Pos	1	None	None	0.000000,0.000	1.00000000	Percent	Coarse
	12	XYZ_Z	Z		2 'XYZ'	2 'Position', 2 'XYZ'	XYZ_Pos	2	None	None	0.000000,0.000	1.00000000	Percent	Coarse
	13	XYZ_Flip	Flip		2 'XYZ'	2 'Position', 2 'XYZ'	XYZ_Pos	3	None	None	0.000000,0.000	1.00000000	Physical	Coarse

Attribute Definitions

The attribute definitions have the following columns:

- Lock:**
 This column can be used to lock the rows and protect them from editing.
- No:**
 This is an auto-generated row number.
- Name:**
 This name needs to be a unique identifier. When it is typed, it is locked and cannot be edited.
- Pretty:**
 This is the name displayed above the encoders. This input accepts local country characters if typed using grandMA3 onPC.
- Main Attribute:**
 This is used to define a hierarchy between the attributes. Editing a cell in this column opens a small **Select Main Attribute** pop-up. Selecting the main attribute makes this attribute a subattribute of the main attribute.
- Activation Group:**
 Activation groups are used to activate a group of attributes as soon as one of the members in the group is activated. Read more in the **Activation Group topic**.



- **Feature:**
Features are part of the hierarchy system. Attributes need to be a part of a feature. Read more in the [Feature Group topic](#).
- **Special:**
This is information only. It shows how the attribute is visualized in the [3D window](#).
- **Special Index:**
This is information about the index number for the special value.
- **Physical Unit:**
The physical units describe the physical properties of the attribute. Read more [below](#).
- **Geometry Type:**
This can be used as a standard geometry type when using the attribute in new Fixture Types. Read more about geometry types in the [Fixture Types topic](#).
- **Color:**
Any attribute can have a color but it is only relevant for the attributes that use "ColorComponent" as the physical unit. Here the color information defines the color of the LED emitter.
- **Intensity:**
The intensity value can be used to define the relationship between the Color Components. It is used by the color engine to compensate if, for instance, the blue emitters are much brighter than the green. This intensity information can also be defined in the fixture types. The information in the attribute definition is used if nothing is defined in the fixture type.
- **Natural Readout:**
This is the readout type used when **Natural** readout is selected.
- **Encoder Resolution:**
This defines the resolution of the encoder controlling the attribute.
- **Log Channels:**
This is information only. It shows how often the attribute is used in logical channels of fixture types added to the show.
- **Channel Functions:**
This is information only. It shows how often the attribute is used in channel functions of fixture types added to the show.
- **Hide:**
This can be used to hide the attribute on the interface. It will not be visible on encoder bars or on any sheets. If the attribute is matched with a DMX, channel then it will still be visible in the DMX sheet.

Add an Attribute Definition

It is recommended to use one of the factory-defined attributes, if possible.

Custom attribute definitions can be created following these steps:

1. Press **Menu**.
2. Tap **Patch**.
3. Tap **Attribute Definitions** on the left side.
4. Select the line in the list, where the new definition should be above.



5. Tap **Insert New Attribute** at the bottom.
6. Fill in all the relevant cells.
7. Close all the menus and tap **Ok** to accept the changes to the setup.

Now the definition exists and can be used to create new custom devices.

Physical Units

The physical units are used to describe the physical part of the attribute - if relevant.

For some attributes, this is not relevant. For instance, the selection of a Gobo is linked to an image of the gobo projected, but it is not defined as a physical unit (it is a wheel slot information). Physical unit information is relevant for the attributes that define the rotation of the selected gobo. So the attribute defining the gobo position (index) uses **Angle** as a physical unit. The attribute defining the continuous rotation of the gobo uses **AngularSpeed** as the physical unit.

Editing the cells for physical units (see the column descriptions above) opens the **Select Physical Unit** pop-up.

There are many physical units available on the list. Select the one matching the needed unit.



13.15.1. Activation Group

An activation group is used when there are several attributes where it makes sense that they are activated together.

For instance, storing pan and tilt values together often makes sense. Having them in the same activation group makes this possible. Activating one of them also activates the other.

There are factory-defined groups that are used with the factory-defined attributes. Custom groups can be made and then used in the **Attribute Definitions**. The group needs to be created before it can be used.

Attribute Definitions		Activation Groups	Feature Groups	Deactivation Groups
Lock	No	Name	Attrib Count	Deactivation Group
	1	PanTilt	3	1 'Position'
	2	XYZ	5	1 'Position'
	3	ColorRGB	37	2 'Color'
	4	ColorHSB	4	2 'Color'
	5	ColorCIE	3	2 'Color'
	6	Gobo1	9	
	7	Gobo1Pos	3	
	8	Gobo2	9	
	9	Gobo2Pos	3	
	10	Gobo3	9	
	11	Gobo3Pos	3	
	12	AnimationWheel1	7	
	13	AnimationWheel1Pos	3	

The standard Activation Groups

The menu has the following columns:

- **Lock:**
This can be used to lock the row from being edited.
- **No:**
This is an auto-generated row number.
- **Name:**
This is the name of the activation group.



- **Attrib Count:**
This is a counter showing how many attributes are part of the group.
- **Deactivation Group:**
This column selects a **Deactivation Group** for two or more activation groups. Read more about this in the **Deactivation Group topic**.

Create and Use a New Activation Group

To create a new activation group:

1. Press **Menu**.
2. Tap **Patch**.
3. Tap **Attribute Definitions** on the left side.
4. Tap **Activation Groups** on the top tabs.
5. Select the line in the list, where the new group should be above.
6. Tap **Insert New ActivationGroup** at the bottom.
7. Edit the name field to add a descriptive name.
8. Optionally edit the **Deactivation Group** cell to add a deactivation group.
9. Tap **Attribute Definitions** on the top tabs.
10. Assign the new group to the relevant attributes in the **Attribute Definitions** menu.
11. When finished, close the menus and tap **Save and Exit** in the pop-up asking if the changes should be kept.

For more information on how to use the **Activation Group** read the **Attribute Definition topic**.



13.15.2. Feature Group

Feature groups are part of the structure and hierarchy of the entire show.

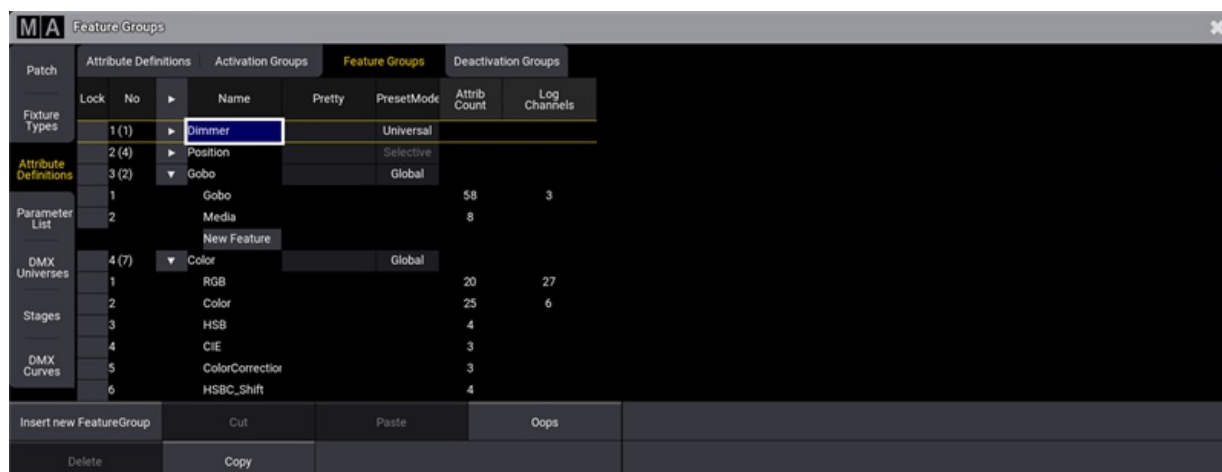
This structure is visible and used in the **Feature Group Control Bar** in the **Encoder Bar**.

A **Feature Group** contains at least one **Feature**.

The feature is assigned to attributes in the **Attribute Definitions**. All attributes need to have an assigned feature.

A feature group automatically has a **Preset Pool** where the attributes in the feature group can store values. Read more about presets in the **Presets topics**.

The feature group and feature need to be created before they can be used.



The standard Feature Groups with Features

The menu has 6 columns:

- **Lock:**
This can be used to lock the row from being edited.
- **No:**
This is an auto-generated row number for the feature group. The number in parentheses is the number of features inside the group.
- **Name:**
This is the name of the feature. This needs to be a unique name.
- **Pretty:**
This is a custom label for the feature group. It can have local characters or a different spelling. For instance "Color" could be labeled "Colour".
- **Preset Mode:**
This is the default type used when storing values in the preset pool.
- **Attrib Count:**
This is information only. This is a counter showing how many attributes use the feature.




- **Log Channels:**

This is information only. It shows how often the feature is used in logical channels of fixture types added to the show.

Create and Use a New Feature Group and Feature

Create a new feature group and feature following these steps.

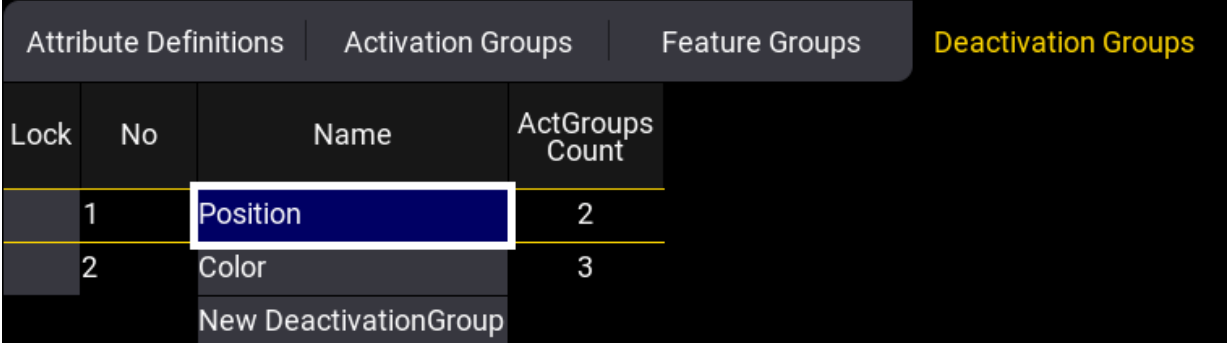
1. Press **Menu** on the console.
2. Tap **Patch** to select the patch menu.
3. Tap **Attribute Definitions** on the left side.
4. Tap **Feature Groups** on the top tab.
5. Select the row in the list, where the new group should be above.
6. Tap **Insert New FeatureGroup** at the bottom.
7. Fill in the name of the group.
8. Tap the triangle icon  next to the new group to expand the group.
9. Tap the **New Feature** area in the new group.
10. Tap **Insert** and write the name of the feature. Repeat until all the needed features have been created.
11. Tap **Attribute Definitions** on the top tab.
12. Assign the new features to the relevant attributes.
13. When finished close all the menus and tap **Save and Exit** in the pop-up asking if the changes should be kept.

For more information on how to use the **Feature Group** read the [Attribute Definition topic](#).

13.15.3. Deactivation Group

Deactivation Groups are groups of **Activation Groups**. A deactivation group makes sure that only one set of valid attribute values are active for the actual DMX channels being output. This is to avoid DMX channels getting conflicting information. The deactivation group knocks out other groups of attributes when a value is assigned to a different activation group than the one already active.

For example, when having active pan and/or tilt values in the programmer, the deactivation group "Position" takes care, that these attributes will be knocked out when activating XYZ attributes instead.



Lock	No	Name	ActGroups Count
1		Position	2
2		Color	3
New DeactivationGroup			

Default deactivation groups

By default, there are 2 automatically generated deactivation groups: Position and Color.

The menu has the following columns:

- **Lock:**
This can be used to lock the row from being edited.
- **No:**
This is an auto-generated row number.
- **Name:**
This is the name of the activation group.
- **ActGroups Count:**
This is the number of activation groups using the deactivation group.

The deactivation groups are assigned to activation groups in the **Activation Groups** tab.

Create and Use a New Deactivation Group

Create a new deactivation group following these steps.

1. Press **Menu**.
2. Tap **Patch**.
3. Tap **Attribute Definitions** on the left side.
4. Tap **Deactivation Groups** on the top tabs.
5. Select the line in the list, where the new group should be above.
6. Tap **Insert New DeactivationGroup** at the bottom.



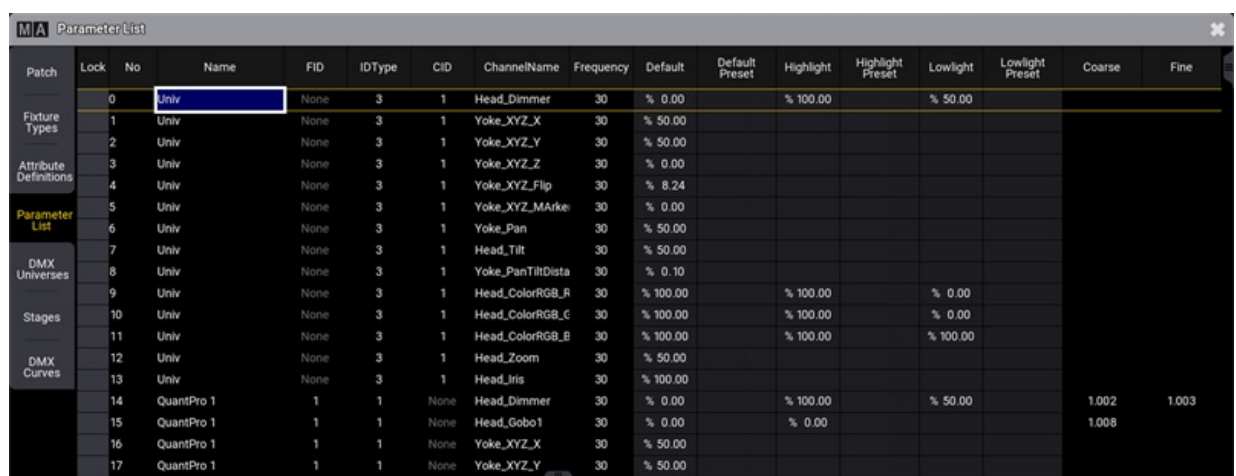
7. Edit the name field to add a descriptive name.
8. Tap **Activation Groups** on the top tabs.
9. Assign the new deactivation group to the relevant activation groups in the **Activation Groups** menu.
10. When finished, close the menus and tap **Save and Exit** in the pop-up asking if the changes should be kept.

13.16. Parameter List

The Patch menu gives access to the list of all the parameters by tapping the **Parameter List** button in the menu on the left of the **Patch Menu**.

These parameters are also called RTChannels (Realtime Channels).

This menu lists all the parameters in the show.



Patch	Lock	No	Name	FID	IDType	CID	ChannelName	Frequency	Default	Default Preset	Highlight	Highlight Preset	Lowlight	Lowlight Preset	Coarse	Fine
		0	Univ	None	3	1	Head_Dimmer	30	% 0.00		% 100.00		% 50.00			
Fixture Types		1	Univ	None	3	1	Yoke_XYZ_X	30	% 50.00							
		2	Univ	None	3	1	Yoke_XYZ_Y	30	% 50.00							
Attribute Definitions		3	Univ	None	3	1	Yoke_XYZ_Z	30	% 0.00							
		4	Univ	None	3	1	Yoke_XYZ_Flip	30	% 8.24							
Parameter List		5	Univ	None	3	1	Yoke_XYZ_MArke	30	% 0.00							
		6	Univ	None	3	1	Yoke_Pan	30	% 50.00							
DMX Universes		7	Univ	None	3	1	Head_Tilt	30	% 50.00							
		8	Univ	None	3	1	Yoke_PanTiltDista	30	% 0.10							
Stages		9	Univ	None	3	1	Head_ColorRGB_R	30	% 100.00		% 100.00		% 0.00			
		10	Univ	None	3	1	Head_ColorRGB_C	30	% 100.00		% 100.00		% 0.00			
DMX Curves		11	Univ	None	3	1	Head_ColorRGB_B	30	% 100.00		% 100.00		% 100.00			
		12	Univ	None	3	1	Head_Zoom	30	% 50.00							
		13	Univ	None	3	1	Head_Inr	30	% 100.00							
		14	QuantPro 1	1	1	None	Head_Dimmer	30	% 0.00		% 100.00		% 50.00		1.002	1.003
		15	QuantPro 1	1	1	None	Head_Gobo1	30	% 0.00		% 0.00				1.008	
		16	QuantPro 1	1	1	None	Yoke_XYZ_X	30	% 50.00							
		17	QuantPro 1	1	1	None	Yoke_XYZ_Y	30	% 50.00							

Parameter List menu

The menu is a big table with the parameters in rows. This is a short explanation of the different columns:

- **Lock:**
This indicates a "Yes" if the row is locked.
- **No:**
This is the parameter number.
- **Name:**
This is the name of the fixture.
- **FID:**
This is the FID of the fixture using the parameter.
- **IDType:**
This is the ID number of the fixture ID type.
- **CID:**
This is the CID for the fixture using the parameter.
- **ChannelName:**
This is the name of the channel in the fixture definition.
- **Frequency:**
This is the DMX output frequency.
- **Default:**
This is the default value for the parameter.



- **Default Preset:**
If the default value is referencing a preset, then the preset number and name are shown here.
- **Highlight:**
This is the highlight value for the parameter.
- **Highlight Preset:**
If the highlight value is referencing a preset, then the preset number and name are shown here.
- **Lowligh:**
This is the lowlight value for the parameter.
- **Lowligh Preset:**
If the lowlight value is referencing a preset, then the preset number and name are shown here.
- **Coarse:**
This is the DMX address for coarse control of the parameter.
- **Fine:**
This is the DMX address for fine control of the parameter.
- **Ultra:**
This is the DMX address for ultra-fine control of the parameter.
- **DMX Curve:**
This field can be used to select one of the existing DMX curves. Learn about DMX curves in the [DMX Curve topic](#).

Only the default, highlight, lowlight values and presets, and the DMX curves can be changed in this list.

A filtered version of this list is also shown when a fixture is edited. It is filtered to only show the attributes (or RTChannels) for the fixture being edited.

Change the Default, Highlight, and Lowlight Values

The values in the Default, Default Preset, Highlight, Highlight Preset, Lowlight, and Lowlight Preset columns can be edited in this menu or when editing the fixture.

It is also possible to store the programmer content to these elements. It automatically assigns presets that might be in the programmer into the preset column.

To reset any of the special values, set the corresponding attributes to remove values in the programmer and Store /Merge them to the desired type of special values. Or set any value for the corresponding attributes and do a Store /Remove.

It is done using the **Default**, **Highlight**, and **Lowligh** keywords and the **/Remove** and **/Merge** option keywords.

Examples:

Store a new default position for a group of fixtures.

1. Clear the programmer for any existing values and selection.
2. Select the desired fixtures.

3. Give them the desired position for instance from a preset.
4. Be careful to not have undesired values in the programmer (for instance, intensity values).
5. Execute this command: **Store Default**.
6. Clear the programmer and verify the new default position.

Remove the currently stored highlight values for a selection of pan and tilt attributes.

1. Clear the programmer for any existing values and selection.
2. Select the desired fixtures.
3. Give the pan and tilt attributes some value - the actual value does not matter it only matters that there are active programmer values for the desired attributes.
4. Be careful to not have undesired values in the programmer (for instance, intensity values).
5. Execute this command: **Store Highlight /Remove**.
6. Clear the programmer and verify the new highlight position.

For resetting the special values of default and lowlight, follow the example above but replace Highlight with Default, or Lowlight.

In addition, storing with the **/Release** option keyword to the special values will do the same as Remove.



Important:

For which fixtures the special values will be changed when using the command line approach, the Use Selection setting and the If not empty setting of the Store Options are important.

To learn more about the Use Selection setting and the If not empty setting, read in the **Store Options topic**.



Restrictions:

Presets with more than 1 step ("Phaser presets") will only use step 1 when using them as special values.

13.17. DMX Curves

Attributes use DMX curves. As a default, they use a linear transition from 0% to 100%. Other DMX curves can be created and assigned to parameters.

There are three types of DMX curves: MinMax, Switch, and Custom.

 **Restriction:**
 A show file can contain 9 999 DMX curves.

Importing fixture types that use special DMX curves adds the curves to the list.

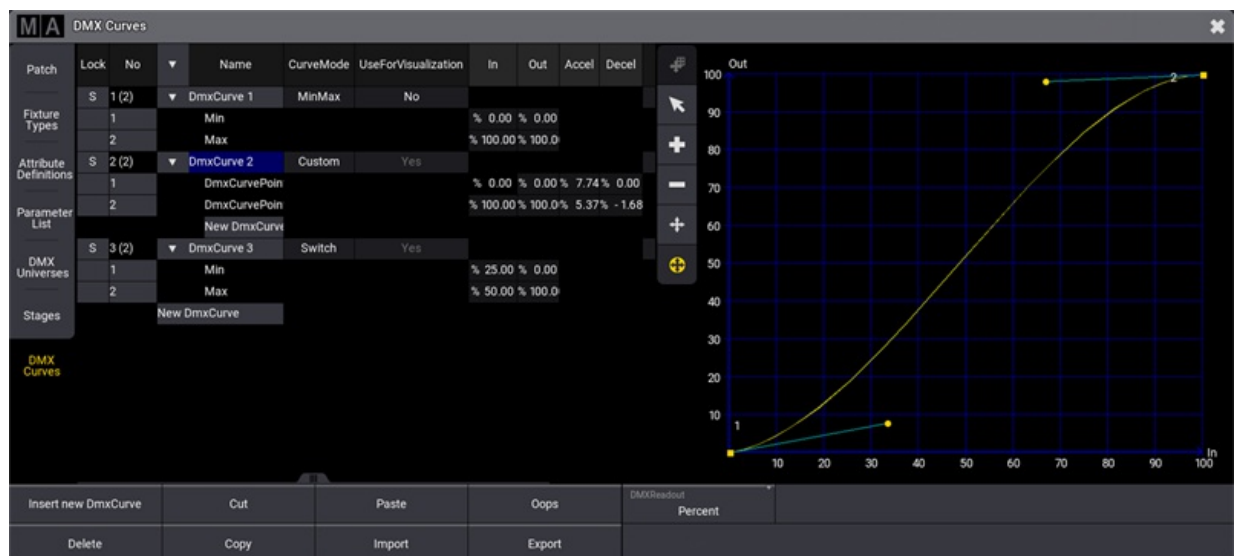
Curve values can be seen and edited in different readouts. These can be changed in the menu at the bottom by tapping the **DMXReadout** button.

Other buttons at the bottom give access to Cut, Copy and Paste the curves in the list. There is also a button that deletes a curve. If the curve is used, it might be impossible to delete it.

The curves are displayed, created, and edited in the DMX Curves menu found in the patch.

Open the DMX Curves Menu






1. Press **Menu**.
2. Tap **Patch**.
3. Tap **DMX Curves** on the left side.



DMX Curve menu

The DMX Curve menu is split into left and right sides. The left side lists the curves, and the right displays the selected curve.

The curve settings are listed on the left side list. The curves can be unfolded (tap the white right-pointing triangle), and the different points on the curves can be seen and edited in the table mode. In and Out show the relationship between the incoming value and what it is translated to in the curve.

The right side can also be used to edit curves. It has a standard toolset for selection () , addition of point () , removal of point () , move point () , and move point handle () . Not all tools not available for all curve modes.

The column called **UseForVisualization** defines whether the DMX output, as defined by the DMXCurve, shall be visualized in the 3D Viewer.



Important:

When enabling this property for DMXCurves created by color-calibrated fixtures, the colors in the 3D Viewer will not match the actual output on stage!

All DMX curve changes are made inside the patch menu. This means that the menu needs to be closed and changes saved for changes to take effect. Finish all DMX curve changes by leaving and saving the patch changes. Or exit without saving to cancel any changes.

Create a New MinMax Curve

MinMax curves are linear DMX transitions from a minimum to a maximum value. These can be useful to limit output values in, for instance, dimmers or the pan and tilt movements of fixtures.

1. Open the DMX Curves menu.
2. Tap a location in the curves list on the left side where the new curve should be.
3. Tap **Insert new DmxCurve** at the bottom - the default curve mode for a new curve is **MinMax**.
4. Edit the name to give it a custom name.
5. Edit the Min and Max values to match the needs.

Create a New Switch Curve

Switch curves use two points to create a switch point where the output value instantly changes from one output value to the other.

1. Open the DMX Curves menu.
2. Tap a location in the curves list on the left side where the new curve should be.
3. Tap **Insert new DmxCurve** at the bottom.
4. Edit the CurveMode value to open the small **Select CurveMode** pop-up.
5. Tap **Switch** in the pop-up.
6. Edit the name to give it a custom name.
7. Edit the Min and Max values of the two points to match the needs.

Having two points in the switch allows for setting a value needed for switching on and another value for switching down. This can help with hysteresis.

Example:

1. Create a new switch curve.
2. Set the **Min** point to In = 70% Out = 0%
3. Set the **Max** point to In = 80% Out = 100%

This creates a switch that needs a value of 80% and above to turn On (output 100%) and a value of 0% to 70% to turn Off (0% output)

Create a New Custom Curve

Custom curves can be used for many different things, for instance, to match the emitter output of LED fixtures.

1. Open the DMX Curves menu.
2. Tap a location in the curves list on the left side where the new curve should be.
3. Tap **Insert new DmxCurve** at the bottom.
4. Edit the CurveMode value to open the small **Select CurveMode** pop-up.
5. Tap **Custom** in the pop-up.
6. Edit the name to give it a custom name.
7. Points can be added using the Add tool on the right side or by creating a **New DMXCurvePoint** in the table view on the left side.

To create a point using the tools:

1. Tap the Add tool (+)
 2. Tap in the curve where the point should be added
8. Edit the Min and Max values and the Accel (acceleration) and Decel (deceleration) of the points to match the needs. This can be done in the table or by using the graph editor on the right side.



Important:

A curve might give a warning if the output at some point is lower than a previous output value. Curves with a warning have the name displayed with a red text color.

Export and Import Curves

Curves can be exported to a drive.

1. Select the desired drive.
2. Tap **Export**.
3. Select the desired drive.
4. Write a name in the name input field.
5. Tap **Export** in the pop-up.

Curves can be imported to the show.

1. Select a location in the curve list.
2. Tap **Import**.
3. Select the desired source drive.



4. Select the desired curve.
5. Tap **Import** in the pop-up.

Assign Curve to a Parameter

The curves can be assigned to parameters in the **Parameter List**.

1. Open the **Parameter List** in the menu on the left side.
2. Locate the parameter that needs to use the custom curve.
3. Edit the DMXCurve field and select the custom curve in the small select pop-up.

Assign Curve to a Fixture Type

DMX Curves can be assigned to an attribute in a **Fixture Type**.

1. Open the **Fixture Types** in the menu on the left side.
2. Select the desired fixture in the list.
3. Tap **Edit** in the menu at the bottom.
4. Tap the **DMXModes** tab at the top.
5. Edit the DMX Curve field for the desired attribute.
6. Select the wanted curve in the pop-up.
7. Close the FixtureType editor pop-up by tapping the **X** in the upper right corner.

14. Operate Fixtures

When the fixtures are added and patched in the show, see [Patch and Fixture Setup](#), the next step will be to select them and operate them.

Different fixture types can do different things. Most fixtures have some kind of intensity control, but if it is a moving light, it will also have pan and tilt control. If the fixture can change color, it will have a color wheel, an RGB, a CMY, or a scroller. If it is a media server, there are many other settings to control, for instance, which clip to play. All these different control elements are called **Attributes**.



Hint:

It is important to know the patched fixtures and what they can do (what attributes they have). Do not waste a lot of time trying to mix the perfect color if it is a fixture with a fixed color wheel.

One of the primary ways to operate fixtures is via the **encoder bar**.



Encoder bar

This gives fast access to all the attributes available for the selected fixtures.

The **dual encoders** change function depending on the selected **feature group**, for instance, position.

Values that are changed get a red background color. This indicates that the values exist in the **programmer**.

There is a special window for color control - the **color picker**. This is a really nice tool for selecting a color independently of the fixture's actual color system.

The **smart view** gives quick access to the fixture defined values. For instance, most fixtures with a gobo wheel have information about what value is needed to select the gobo slots.

Dimmer

The dimmer is a little special because it is a universal attribute. Almost all fixtures have some intensity control.

There are many ways to set the dimmer value on selected fixtures. Here are some of them:

- Use the level wheel to set the value.
- Use the numeric keys to set a specific value, for example: **At 3 0 Please**
- Press **Full** to give the fixture 100% intensity.



- Select **Dimmer** in the encoder bar and use the dual encoders to set a value.
- Select **Dimmer** in the encoder bar and press on the inner ring of the dual encoder. This opens the **calculator**. Use this to set a specific value.
- Edit the value in the fixture sheet to get the calculator to set a value.
- Use a preset.



14.1. Select Fixtures

To operate fixtures and change the values of attributes (for example dimmer, pan, tilt, or zoom), it is necessary to select fixtures first.

To select fixtures, use one of the following ways:

- Type the fixture ID in the command line, using the numeric keys.
- Tap at the fixture in a **fixture sheet**.
- Draw a lasso around a fixture in the **3D window**.
- Tap at a **group** containing fixtures.
- Tap at a **preset** containing fixture information.

To create groups or presets, use one of the three first listed methods in the bullet list above.




Hint:

To learn more about the font colors used when selecting/deselecting fixtures in the fixture sheet, see **System Colors** topic.

Examples

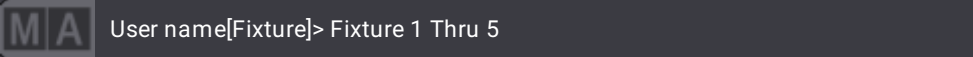
To recreate the examples, the following should be prepared:

- The grandMA3 demo show file is open. To load the grandMA3 demo show file, see **Show File Handling** topic.
- A fixture sheet window is open.
- The following fixture types are patched:
 - Ayrton Alienpix - RS Ex 16 Bit
 - Quantity: 3
 - FID: 301 - 303

To select fixtures 1 through 5, press the following hardkeys on the console or use the command section menu  :

Fixture 1 Thru 5 Please

This command is visible in the command line input:



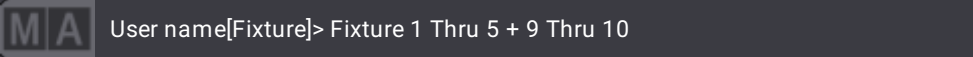
Fixture 1 through 5 is selected. The five selected fixtures are displayed in a yellow font in the Fixture Sheet window.

To deselect fixtures in one command, type:

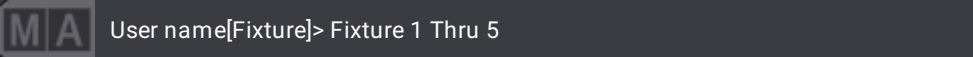


Fixtures 1 to 5 are selected, Fixtures 6 to 8 are deselected, and Fixtures 9 and 10 are selected.

To make the same selection of fixtures as shown in the example above using a different syntax, type:



To select fixtures in two commands, type:



To select the fixtures 9 and 10 in the example above, **+** or **Thru** can be used.



Important:

Some keys are used to navigate the sub-selection of fixtures. This function is a part of the **MAtricks** function. The keys have an on-screen version called **Selection Bar**. These can also be used to select fixtures.



Recursive Selection of Fixtures

Some fixtures have multiple sub-fixtures. The following examples show how to select fixtures and their sub-fixtures in different orders and hierarchies using an additional dot (.).

 **Hint:**
To learn more about sub-fixtures, see the **First Patch** topic in the Quick Start Guide.

To select fixture 301, type:

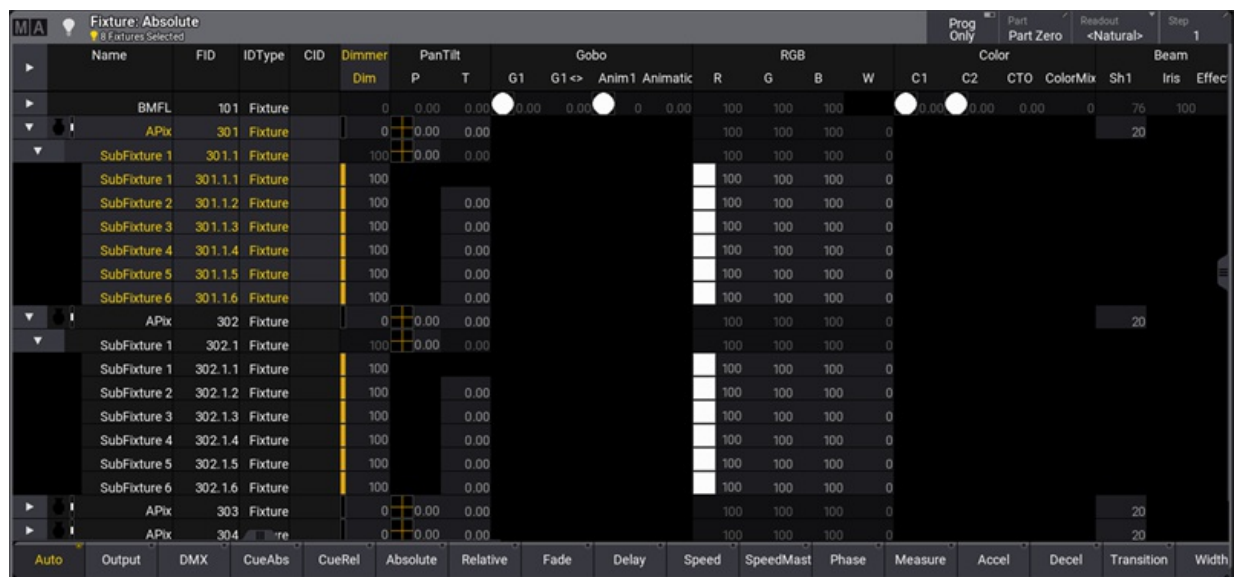
```
MA User name[Fixture]> Fixture 301
```

Only the main fixture is selected.

- For a fast selection of all sub-fixtures, a dot (.) can be added to the selection syntax of the main fixture.

To select fixture 301 and all its sub-fixtures, type:

```
MA User name[Fixture]> Fixture 301.
```



Name	FID	IDType	CID	Dimmer	PanTilt	G1	G1<>	Anim1	Animatic	R	G	B	W	C1	C2	CTO	ColorMix	Sh1	Iris	Effec
BMFL	101	Fixture		0	0.00	0.00	0.00	0.00	0.00	100	100	100	0	0.00	0.00	0	76	100		
APix	301	Fixture		0	0.00	0.00				100	100	100	0						20	
SubFixture 1	301.1	Fixture		100	0.00	0.00				100	100	100	0							
SubFixture 1	301.1.1	Fixture		100						100	100	100	0							
SubFixture 2	301.1.2	Fixture		100		0.00				100	100	100	0							
SubFixture 3	301.1.3	Fixture		100		0.00				100	100	100	0							
SubFixture 4	301.1.4	Fixture		100		0.00				100	100	100	0							
SubFixture 5	301.1.5	Fixture		100		0.00				100	100	100	0							
SubFixture 6	301.1.6	Fixture		100		0.00				100	100	100	0							
APix	302	Fixture		0	0.00	0.00				100	100	100	0						20	
SubFixture 1	302.1	Fixture		100	0.00	0.00				100	100	100	0							
SubFixture 1	302.1.1	Fixture		100						100	100	100	0							
SubFixture 2	302.1.2	Fixture		100		0.00				100	100	100	0							
SubFixture 3	302.1.3	Fixture		100		0.00				100	100	100	0							
SubFixture 4	302.1.4	Fixture		100		0.00				100	100	100	0							
SubFixture 5	302.1.5	Fixture		100		0.00				100	100	100	0							
SubFixture 6	302.1.6	Fixture		100		0.00				100	100	100	0							
APix	303	Fixture		0	0.00	0.00				100	100	100	0						20	
APix	304	Fixture		0	0.00	0.00				100	100	100	0						20	

Fixture 301 and the sub-fixtures are selected.



To select the main fixtures and all sub-fixtures of fixtures 301 thru 303, type:

```
MA User name[Fixture]> Fixture 301 Thru 303.
```

Name	FID	IDType	CID	Dimmer	PanTilt	Gobo	RGB	Color	Beam													
				Dim	P	T	G1	G1<>	Anim1	Animatic	R	G	B	W	C1	C2	CTO	ColorMtx	Sh1	Iris	Effec	
BMFL	101	Fixture		0	0.00	0.00	0.00	0.00	0	0.00	0	100	100	100	0	0.00	0.00	0	76	100		
APix	301	Fixture		0	0.00	0.00						100	100	100	0							
SubFixture 1	301.1	Fixture		100	0.00	0.00						100	100	100	0							
SubFixture 1	301.1.1	Fixture		100								100	100	100	0							
SubFixture 2	301.1.2	Fixture		100		0.00						100	100	100	0							
SubFixture 3	301.1.3	Fixture		100		0.00						100	100	100	0							
SubFixture 4	301.1.4	Fixture		100		0.00						100	100	100	0							
SubFixture 5	301.1.5	Fixture		100		0.00						100	100	100	0							
SubFixture 6	301.1.6	Fixture		100		0.00						100	100	100	0							
APix	302	Fixture		0	0.00	0.00						100	100	100	0							
SubFixture 1	302.1	Fixture		100	0.00	0.00						100	100	100	0							
SubFixture 1	302.1.1	Fixture		100								100	100	100	0							
SubFixture 2	302.1.2	Fixture		100		0.00						100	100	100	0							
SubFixture 3	302.1.3	Fixture		100		0.00						100	100	100	0							
SubFixture 4	302.1.4	Fixture		100		0.00						100	100	100	0							
SubFixture 5	302.1.5	Fixture		100		0.00						100	100	100	0							
SubFixture 6	302.1.6	Fixture		100		0.00						100	100	100	0							
APix	303	Fixture		0	0.00	0.00						100	100	100	0							
SubFixture 1	303.1	Fixture		100	0.00	0.00						100	100	100	0							

All AlienPix (fixtures and sub-fixtures) are selected.

- The dot selects the sub-fixtures downwards from the specified level. Therefore, for example, it is also possible to select only the pixels of an AlienPix.

To select all pixels of an AlienPix, type:

```
MA User name[Fixture]> Fixture 301.1. Thru
```



Name	FID	IDType	CID	Dimmer	PanTilt	G1	G1<->	Anim1	Animatic	R	G	B	W	C1	C2	CTO	ColorMx	Sh1	Iris	Effec
BMFL	101	Fixture		0	0.00 0.00	0.00	0.00	0	0.00	100	100	100	0	0.00	0.00	0.00	0	76	100	
APix	301	Fixture		0	0.00 0.00					100	100	100	0							
SubFixture 1	301.1	Fixture		100	0.00 0.00					100	100	100	0							
SubFixture 1	301.1.1	Fixture		100						100	100	100	0							
SubFixture 2	301.1.2	Fixture		100	0.00					100	100	100	0							
SubFixture 3	301.1.3	Fixture		100	0.00					100	100	100	0							
SubFixture 4	301.1.4	Fixture		100	0.00					100	100	100	0							
SubFixture 5	301.1.5	Fixture		100	0.00					100	100	100	0							
SubFixture 6	301.1.6	Fixture		100	0.00					100	100	100	0							
APix	302	Fixture		0	0.00 0.00					100	100	100	0							
SubFixture 1	302.1	Fixture		100	0.00 0.00					100	100	100	0							
SubFixture 1	302.1.1	Fixture		100						100	100	100	0							
SubFixture 2	302.1.2	Fixture		100	0.00					100	100	100	0							
SubFixture 3	302.1.3	Fixture		100	0.00					100	100	100	0							
SubFixture 4	302.1.4	Fixture		100	0.00					100	100	100	0							
SubFixture 5	302.1.5	Fixture		100	0.00					100	100	100	0							
SubFixture 6	302.1.6	Fixture		100	0.00					100	100	100	0							
APix	303	Fixture		0	0.00 0.00					100	100	100	0							
SubFixture 1	303.1	Fixture		100	0.00 0.00					100	100	100	0							

All pixels of an AlienPix are selected.

To select pixels of an AlienPix in particular, type:

```
MA User name[Fixture]> Fixture 301.1. Thru 4 + 6
```


Name	FID	IDType	CID	Dimmer	PanTilt	G1	G1<->	Anim1	Animatic	R	G	B	W	C1	C2	CTO	ColorMx	Sh1	Iris	Effec
BMFL	101	Fixture		0	0.00 0.00	0.00	0.00	0	0.00	100	100	100	0	0.00	0.00	0.00	0	76	100	
APix	301	Fixture		0	0.00 0.00					100	100	100	0							
SubFixture 1	301.1	Fixture		100	0.00 0.00					100	100	100	0							
SubFixture 1	301.1.1	Fixture		100						100	100	100	0							
SubFixture 2	301.1.2	Fixture		100	0.00					100	100	100	0							
SubFixture 3	301.1.3	Fixture		100	0.00					100	100	100	0							
SubFixture 4	301.1.4	Fixture		100	0.00					100	100	100	0							
SubFixture 5	301.1.5	Fixture		100	0.00					100	100	100	0							
SubFixture 6	301.1.6	Fixture		100	0.00					100	100	100	0							
APix	302	Fixture		0	0.00 0.00					100	100	100	0							
SubFixture 1	302.1	Fixture		100	0.00 0.00					100	100	100	0							
SubFixture 1	302.1.1	Fixture		100						100	100	100	0							
SubFixture 2	302.1.2	Fixture		100	0.00					100	100	100	0							
SubFixture 3	302.1.3	Fixture		100	0.00					100	100	100	0							
SubFixture 4	302.1.4	Fixture		100	0.00					100	100	100	0							
SubFixture 5	302.1.5	Fixture		100	0.00					100	100	100	0							
SubFixture 6	302.1.6	Fixture		100	0.00					100	100	100	0							
APix	303	Fixture		0	0.00 0.00					100	100	100	0							
SubFixture 1	303.1	Fixture		100	0.00 0.00					100	100	100	0							

Particular selection of pixels.


General Fixture Selection

To select all fixtures and sub-fixtures, type:



 User name[Fixture]> Fixture Thru.

To select all parents of all fixtures, type:

 User name[Fixture]> Fixture Thru



14.2. What Is the Programmer

The programmer is a temporary memory, where the edited values are placed. The values can then be stored or released.

Every user has their own programmer.

The programmer has three levels:

- Selected fixture
- Active programmer values
- Inactive programmer values

The programmer's values usually affect the output from the system. But there is a **Blind** function that allows hiding the programmer values from the output.

Selected fixtures are the ones that will be affected by encoder input and any intensity changes. Selected fixtures can be identified with a yellow name and ID text color.

Both active and inactive programmer values can affect the output. The difference is that the active values will be stored using the default **store options** (Use Selection = Active).

Active values have a red background, a white value text color, and a red marker in the fixture sheet.

Inactive values have red value text color and a white marker in the fixture sheet.

These are the default system colors. Read more about all the different system colors in the **system (color) topic**.

Name	FID	IDType	CID	Dimmer	PanTilt	Gobo	RGB	Color													
				Dim	P	T	G1	G1 <-	Anim1	Animatic	R	G	B	W	C1	C2	CTO	ColorMix	Sh1	Iris	Eff
BMFL 101	101	Fixture		76	0.00	0.00	1.87	0.00	0	0.00	100	100	100		0.00	0.00	0.00	0	76	100	
BMFL 102	102	Fixture		76	0.00	0.00	1.87	0.00	0	0.00	100	100	100		0.00	0.00	0.00	0	76	100	
BMFL 103	103	Fixture		76	0.00	0.00	1.87	0.00	0	0.00	100	100	100		0.00	0.00	0.00	0	76	100	
BMFL 104	104	Fixture		76	0.00	0.00	1.87	0.00	0	0.00	100	100	100		0.00	0.00	0.00	0	76	100	
BMFL 105	105	Fixture		76	0.00	0.00	1.87	0.00	0	0.00	100	100	100		0.00	0.00	0.00	0	76	100	
BMFL 106	106	Fixture		76	0.00	0.00	1.87	0.00	0	0.00	100	100	100		0.00	0.00	0.00	0	76	100	
BMFL 107	107	Fixture		76	0.00	0.00	1.87	0.00	0	0.00	100	100	100		0.00	0.00	0.00	0	76	100	
BMFL 108	108	Fixture		76	0.00	0.00	1.87	0.00	0	0.00	100	100	100		0.00	0.00	0.00	0	76	100	
BMFL 109	109	Fixture		76	0.00	0.00	1.87	0.00	0	0.00	100	100	100		0.00	0.00	0.00	0	76	100	
BMFL 110	110	Fixture		76	0.00	0.00	1.87	0.00	0	0.00	100	100	100		0.00	0.00	0.00	0	76	100	
BMFL 111	111	Fixture		76	0.00	0.00	1.87	0.00	0	0.00	100	100	100		0.00	0.00	0.00	0	76	100	
BMFL 112	112	Fixture		76	0.00	0.00	1.87	0.00	0	0.00	100	100	100		0.00	0.00	0.00	0	76	100	
BMFL 113	113	Fixture		0	0.00	0.00	0.00	0.00	0	0.00	100	100	100		0.00	0.00	0.00	0	76	100	
BMFL 114	114	Fixture		0	0.00	0.00	0.00	0.00	0	0.00	100	100	100		0.00	0.00	0.00	0	76	100	
BMFL 115	115	Fixture		0	0.00	0.00	0.00	0.00	0	0.00	100	100	100		0.00	0.00	0.00	0	76	100	
BMFL 116	116	Fixture		0	0.00	0.00	0.00	0.00	0	0.00	100	100	100		0.00	0.00	0.00	0	76	100	

Values in programmer

When a fixture is selected, it is possible to press **Please** twice to activate all attributes for the selected fixture. This puts all the current values for all the attributes of the fixture in the programmer. Press **Please** once more,



and the attributes will be inactive in the programmer.

To release values from the programmer, press **Off** and then tap the value to deactivate it. Feature groups can also be released from the programmer using **Off** and then tapping the encoder toolbar.

When there are values in the programmer, and a fixture is selected, it is possible to use the clear button to clear each of the three levels:

- Press **Clear** to deselect the fixture but keep the active and inactive values in the programmer.
- Press **Clear** again to keep the values in the programmer but as inactive values.
- Press **Clear** a final time to clear the programmer and release all the values.

Completely clearing the programmer can also be done by holding **Clear** for more than one second.

The associated keywords are **Clear**, **ClearSelection**, **ClearActive**, and **ClearAll**.

The **Clear** button in the AT overlay executes the Clear command. For more information, see **At overlay**.

Layers

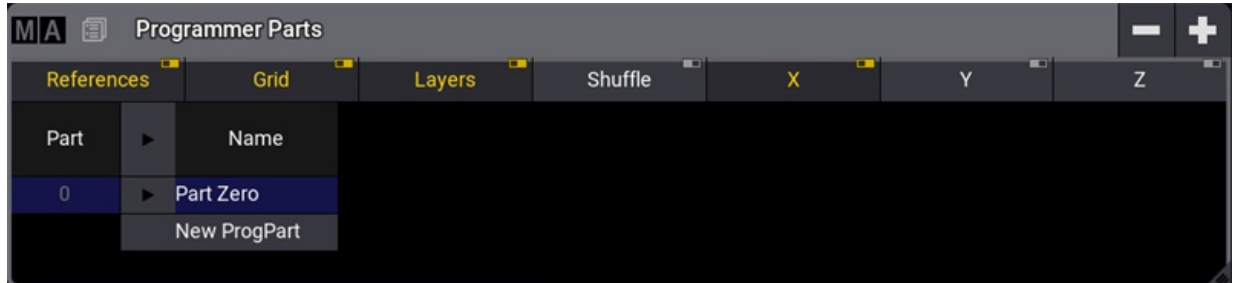
The programmer has several layers. Normal values - for instance, a selected static position - are in the **Absolute** value layer. An absolute value can be affected by a value in the **Relative** layer. For more information, see **Encoder bar**. A dark purple marker indicates relative values. The **Fade** layer can be used to adjust and see individual fade times. Individual fade times are indicated with a green marker. The **Delay** layer can be used to adjust and see individual delay times. An orange marker indicates them. These values are best looked at in the **Sequence Sheet** in tracking mode. Learn more about **Individual Attribute Timing** in Cues and sequences topic.

Programmer Parts

The programmer can have several parts (Maximum: Part Zero + 239 individual parts = 240 parts). The ProgrammerParts window shows all parts of the current programmer. Parts can be labeled here directly.

Using cue parts is a good way to organize your show better. The primary information for every cue is in part 0. Part numbering starts at Part 0. Presets can be called into the programmer parts automatically when calling the preset. This then creates the respective cue parts when storing the programmer's content into a cue. Learn more about pool settings at **Preset Pool – Preset Pool Settings**.

To see all active programmer parts, open a Programmer Parts window. For more information, see **Add Windows**.



Programmer Parts

Programmer parts can be deleted or created using the buttons in the title bar. A part can be selected by tapping the part in the window. A new programmer part can also be created using the command line and the **Programmer keyword**.

To create programmer part 3, which is labeled "Color", type:

```
MA User name[Fixture]> Store Programmer 3 "Color"
```

Select the programmer part simply by writing the keyword and the number or name.

```
MA User name[Fixture]> Programmer "Color"
```

When a programmer part is empty, it will be deleted when entering a different cue part.

For example, to store cue 12 with part cue 3, type:

```
Store Cue 1 2 Cue 3
```

```
MA User name[Fixtures]>Store Cue 12 Part 3
```

Freeze

By default, the executors have a higher priority than the programmer. That means that the programmer's values can be changed if a running executor has stored values for the exact attributes. To give the programmer a higher priority than the executors, press **Freeze**. This keeps the adjusted values in the programmer even when the executor is executed.

Blind

Blind is a function that hides programmer values from the output. The blind mode can be toggled On and Off by pressing **Blind** or using the **Blind Keyword**.

Turning on the blind mode without any values in the programmer does not change the output. The changing values while in blind do not affect output. If the programmer is cleared before leaving the blind mode, then the output is not affected. Leaving blind with programmer values will take these values to the output. Entering blind

mode with values in the programmer removes them from the output, meaning the output is changed.

Program Time

The value changes in the programmer are usually done immediately. But the value change can be done with a timed or manual fade using the **Program Time** function. Read more about it in the [Time Control topic](#).

Preview

In preview mode, the user can have values entered in a separate programmer and will not generate output on a stage.

The preview programmer behaves the same as the live programmer. There is one preview programmer in a session.



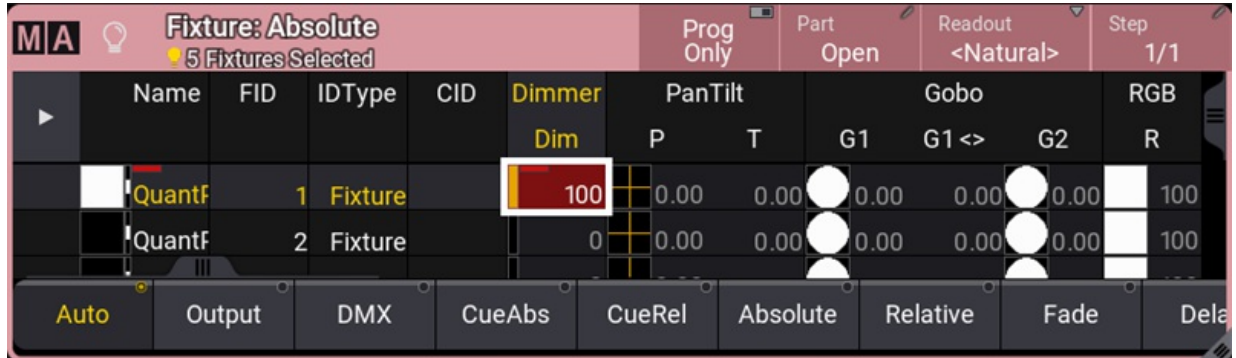
Warning:

When multi users are working in a session, be aware that everyone is working in one shared preview programmer. As a result, one user can modify other users' values in the preview programmer.

To enter the preview mode, press **Prvw**.

When Preview is active, the **Prvw** key is lit up, and the borders of the following windows will turn red:

- 3D Viewer
- Align Bar
- At Filter, window, and overlay
- Color Picker
- DMX Sheet
- Fixture Sheet
- Layout Viewer
- MAtricks Editor, window, and overlay
- Phaser Editor, window, and overlay
- Programmer Parts
- Selection Grid
- sMArt
- Step Bar



Fixture Sheet in Preview Mode

Editing cues and presets in preview mode will turn the window frame yellow.



Fixture Sheet in Edit Preview Mode


To exit the preview mode, press **Prvw** again. The border of the windows listed above will turn gray.

The output of the preview programmer will be displayed in the corresponding windows. If the values of the preview programmer were not manually cleared, leaving and entering the preview mode will bring back the previous values.

14.3. Fixture Sheet

The fixture sheet shows all the patched objects in the showfile with an FID or a CID. This is usually every object that needs to be controlled in the show. Read the [Patch and Fixture Setup topics](#) to learn how to add objects to the patch.

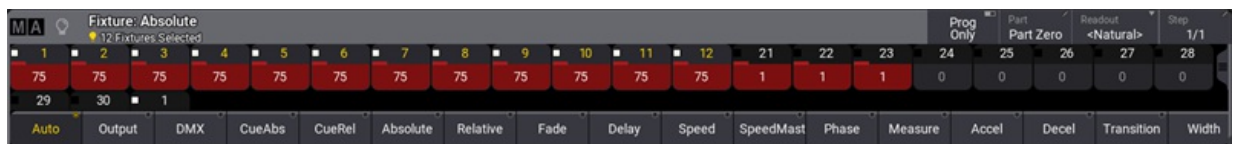
The fixture sheet is a window that can be created like any other window. For more information, see [Add windows](#).



Name	FID	IDType	CID	Dimmer	PanTilt	Gobo	RGB	Color											
Dim	Presets	P	T	G1	G2	G2<->	Anim1	Anim1 F	Anim Po	R	G	B	W	Color	MaChase	PaChase	SpChase	Fd	C1
BMFL	1	Fixture		75	0.00 -0.00	0.00 0.00	100 100 100	0.00											
BMFL	2	Fixture		75	0.00 -0.00	0.00 0.00	100 100 100	0.00											
BMFL	3	Fixture		75	0.00 -0.00	0.00 0.00	100 100 100	0.00											
BMFL	4	Fixture		75	0.00 -0.00	0.00 0.00	100 100 100	0.00											
BMFL	5	Fixture		75	0.00 -0.00	0.00 0.00	100 100 100	0.00											
BMFL	6	Fixture		75	0.00 -0.00	0.00 0.00	100 100 100	0.00											
BMFL	7	Fixture		75	0.00 -0.00	0.00 0.00	100 100 100	0.00											
BMFL	8	Fixture		75	0.00 -0.00	0.00 0.00	100 100 100	0.00											
BMFL	9	Fixture		75	0.00 -0.00	0.00 0.00	100 100 100	0.00											
BMFL	10	Fixture		75	0.00 -0.00	0.00 0.00	100 100 100	0.00											
BMFL	11	Fixture		75	0.00 -0.00	0.00 0.00	100 100 100	0.00											
BMFL	12	Fixture		75	0.00 -0.00	0.00 0.00	100 100 100	0.00											
Ayr Black	21	Fixture		0	-1.0x -1.0x		0 0 0	100 0 0 0											
Ayr Black	22	Fixture		0	-1.0x -1.0x		0 0 0	100 0 0 0											
Ayr Black	23	Fixture		0	-1.0x -1.0x		0 0 0	100 0 0 0											
Ayr Black	24	Fixture		0	-1.0x -1.0x		0 0 0	100 0 0 0											


Fixture Sheet in fixture mode

The sheet has two modes. It can be **Fixture**, or it can be **Channel**. This can be changed in the [window settings](#).



ID	Intensity
1	75
2	75
3	75
4	75
5	75
6	75
7	75
8	75
9	75
10	75
11	75
12	75
21	1
22	1
23	1
24	0
25	0
26	0
27	0
28	0

Fixture Sheet in channel mode



Hint:
The sheet uses the system colors - read more about them [here](#).

The **Channel** mode shows all objects with an ID and an intensity attribute. It shows only the ID and its intensity values.

The **Fixture** mode shows the same objects, but it shows the values of every available attribute.

The active world filters the sheet. If this world is different than world one, then the world name and number are displayed in the title bar.

The fixture sheet is divided into rows and columns. In the **Fixture** mode, each row represents a fixture or a sub-fixture.



The first column is the fixture name. To see the sub-fixtures for the current fixture, tap on the arrow on the left side of the fixture name column.

The second column is the FID (Fixture ID). Every fixture patched in the showfile must have a unique FID, CID, or both. If a fixture contains sub-fixtures, these IDs will be the fixture number, a dot, and a sub-number.

The next column shows the IDtype of the fixture. Learn more about the ID types in the [What are Fixtures topic](#).

The next column is the CID (Channel ID / Custom ID depending on the ID type). This is the other ID a fixture can have.

The next columns display the values for the fixture's different attributes, depending on the mask and display settings. Some attributes might have symbols next to them, showing the selected gobo or the result of the color attributes.

Channel mode only shows the ID (it prioritizes the CID if available), the intensity value, and a small square showing the combination of current color and intensity.

In the title, there are four buttons.









- **Prog Only:**

Displays only the programmer data.

- **Part:**

This is the Programmer Part - read more about it in the [What is the Programmer topic](#).

When the programmer has values from different parts, all cells of attributes that are not part of the selected programmer part are displayed in the fixture sheet with a darker background color.

M A		Fixture: Absolute		Prog Only	Part	Readout	Step
		8 Fixtures Selected			Part Zero	<Natural>	1/1
Name	FID	IDType	CID	Dimmer			
				Dim			
 BMFL	1	Fixture		100			
 BMFL	2	Fixture		100			
 BMFL	3	Fixture		100			
 BMFL	4	Fixture		100			
 BMFL	5	Fixture		100			
 BMFL	6	Fixture		100			
 BMFL	7	Fixture		100			
 BMFL	8	Fixture		100			

Auto Output DMX CueAbs CueRel Absolute Relative

Fixture sheet displaying values for part 1 (Dimmer) with a stronger and brighter red background color.

- **Readout:**

The readout defines how the attribute values are displayed. The options are:

- **Auto** - This makes the sheet follow the readout set in the encoder toolbar. Notice that the button in the title bar does not say Auto; it displays the selected readout in angle brackets.
- **Natural** - For more information, see Patch and [Fixture Setup - Attribute definitions](#).
- **Percent** - This displays the value as a percentage from 0% to 100%. This is a whole number. **Percent** Keyword.
- **PercentFine** - This is also displaying the value in percent. The range is the same, but the resolution is higher. The value has two decimal numbers. **PercentFine** Keyword.
- **Physical** - This is the physical values defined in the fixture type definition. **Physical** Keyword.
- **Decimal8 / Decimal16 / Decimal24** - This displays the value in a decimal number in the three available resolutions (256, 65 536, or 16 777 216). **Decimal8**, **Decimal16**, and **Decimal24** keywords.
- **Hex8 / Hex16 / Hex24** - This displays the value in a hexadecimal number in the three available resolutions (FF, FFFF, or FFFFFFFF). **Hex8**, **Hex16**, and **Hex24** keywords.

- **Step:**

This is the step number - learn more about steps in the [Phaser section](#).



Hint:

There is a **user profile setting** called **Value Readout**. This is the one being used by the **Auto** option, and it is also the one actually being changed by the encoder toolbar.

To read more about all the other different settings and masks, read the **Window settings** topic.



Hint:

It is possible to select fixtures by tapping them in the Fixture Sheet.



14.3.1. Graphics

Feature Graphic

The attribute values of the **Dimmer**, **Position**, **Gobo**, and **Color** features can be displayed with a small symbol graphic representing the attribute values.

Fixture: Absolute																	Prog Only	Part Part Zero	Readout <Natural>	Step 1/1
Name	FID	IDType	CID	Dimmer	PanTilt		Gobo				RGB				Color					
				Dim	P	T	G1	G1<->	Anim1	Animatic	R	G	B	W	C1	C2	CTO			
BMFL	101	Fixture		100	90.0	-47.25	3.44	0.00	0	0.00	0	100	100		0.00	0.00	0.00			
BMFL	102	Fixture		100	90.0	-47.25	3.44	0.00	0	0.00	0	100	100		0.00	0.00	0.00			
BMFL	103	Fixture		100	90.0	-47.25	3.44	0.00	0	0.00	0	100	100		0.00	0.00	0.00			
BMFL	104	Fixture		100	90.0	-47.25	3.44	0.00	0	0.00	0	100	100		0.00	0.00	0.00			
BMFL	105	Fixture		100	90.0	-47.25	3.44	0.00	0	0.00	0	100	100		0.00	0.00	0.00			
BMFL	106	Fixture		100	90.0	-47.25	3.44	0.00	0	0.00	0	100	100		0.00	0.00	0.00			
BMFL	107	Fixture		100	90.0	-47.25	3.44	0.00	0	0.00	0	100	100		0.00	0.00	0.00			
BMFL	108	Fixture		100	90.0	-47.25	3.44	0.00	0	0.00	0	100	100		0.00	0.00	0.00			
BMFL	109	Fixture		100	90.0	-47.25	3.44	0.00	0	0.00	0	100	100		0.00	0.00	0.00			
BMFL	110	Fixture		100	90.0	-47.25	3.44	0.00	0	0.00	0	100	100		0.00	0.00	0.00			
BMFL	111	Fixture		100	90.0	-47.25	3.44	0.00	0	0.00	0	100	100		0.00	0.00	0.00			
BMFL	11	Fixture		100	90.0	-47.25	3.44	0.00	0	0.00	0	100	100		0.00	0.00	0.00			

- Tapping **Feature Graphic** in the Fixture Sheet, the Content Sheet, and the Sequence Sheet settings window will show or hide these symbols in the corresponding sheet.

Fixture Sheet Settings
Edit Title Bar
Load
Save
Delete Window
✕

Display

Mask

Layer
Auto

Step
1/1

Readout
Auto

Speed
Auto

Preset
ID+Name

Sheet Mode
Fixture

#Columns
20

Adjust Columns

Transpose

Merge Cells

Fixture Sort

Feature Sort

Layer Toolbar

Color Mode
<RGB>

Time Format
<10d11h23m

Frame Readout
<Seconds>

Appearance

Font Size
Default

Fixture Appearance
None

Fixture Graphic
Gobo

Fixture Graphic Source
Value

Feature Graphic



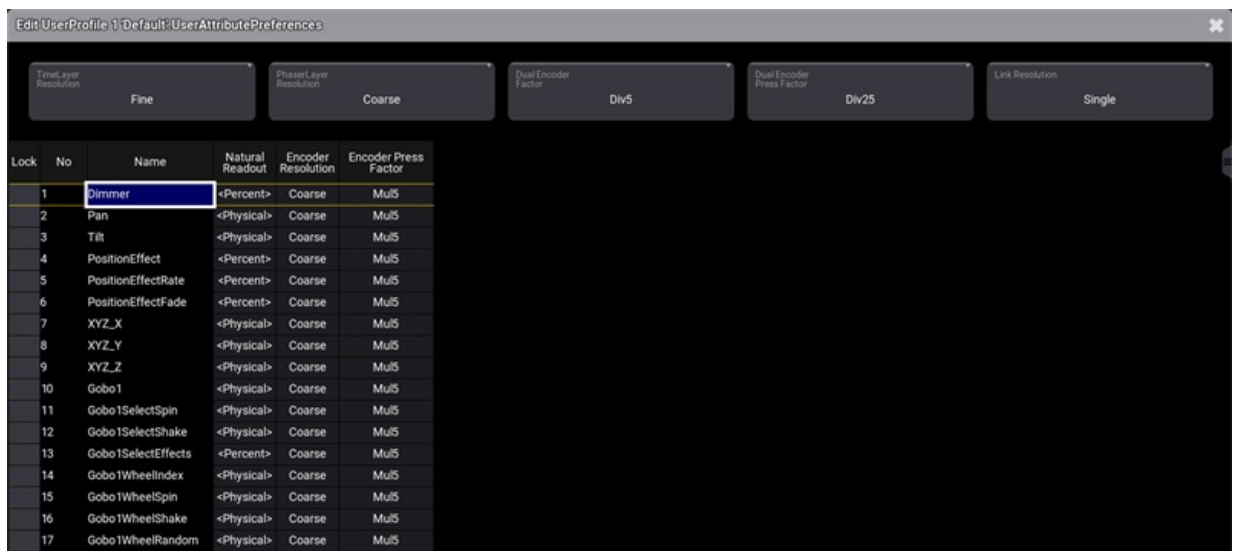


14.4. Encoder Resolution

Sometimes precise handling is required, and you may want to change the resolution of an encoder.

Default encoder resolution of attributes can be defined from the user configuration menu:

1. Press **Menu**, tap **Settings**, and tap **User Configuration**. The user configuration menu opens.
2. On the left side of the window, tap **Profiles**.
3. Tap **Edit Encoder Bar**. The Default User Attribute Preferences menu opens.



It is also possible to change the encoder resolution directly on an encoder.

To do so:

1. Press and hold **MA**, this will display the possible resolutions in the channel function area of each encoder.
2. While holding **MA**, tap **Link Resolution** and select **Single** or **FeatureGroup**, then tap the resolution area of an encoder, the selected resolution of the encoder will change to the next resolution.
3. When the desired resolution is selected, release **MA**.





Hint:

- An encoder has 24 clicks for one turnaround.
- 5 turns of an encoder are needed to cross the whole range of the attribute from its minimum to maximum.

The possible resolutions are:

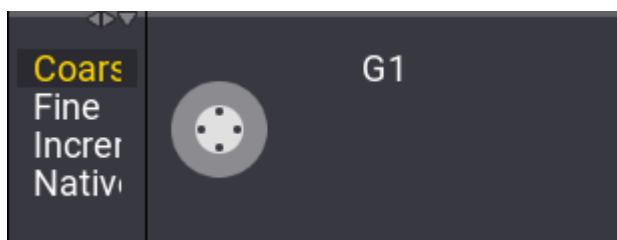
1. **Coarse:** One encoder turn click will change the value depending on the readout:
 - Percent: 1
 - PercentFine: 1
 - Physical: $(\text{MaxValue} - \text{MinValue}) / (24 * 5)$
When multiple fixture types are selected, the smallest physical range of a fixture is taken to determine the size of one click. This allows having the same value change when for example, turning the tilt encoder.
 - Dec8: $255 / (24 * 5)$, therefore 1 click equals 2.125
 - Dec16: $65\ 535 / (24 * 5)$, therefore 1 click equals 546.125
 - Dec24: $1\ 677\ 216 / (24 * 5)$, therefore 1 click equals 13\ 976.8
 - Hex8: $255 (=FF) / (24 * 5)$, therefore 1 click equals 2.125
 - Hex16: $65\ 535 (FFFF) / (24 * 5)$, therefore 1 click equals 546.125
 - Hex24: $1\ 677\ 216 (=FFFFFF) / (24 * 5)$, therefore 1 click equals 13\ 976.
2. **Fine:** Fine has a 10x finer resolution than coarse.
3. **Increment:** When the resolution is set to Increment, one encoder turn click will change the lowest digit of the displayed readout.
4. **Native:** The value Layers absolute and relative offer this mode to have direct access to the smallest possible value change of the parameter resolution.



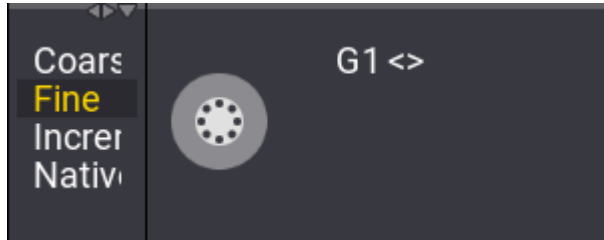
Hint:

When Native mode is selected, if for example, a dimmer channel is based on 8/16/24 bit, one encoder turn always results in one digit change of the DMX output.

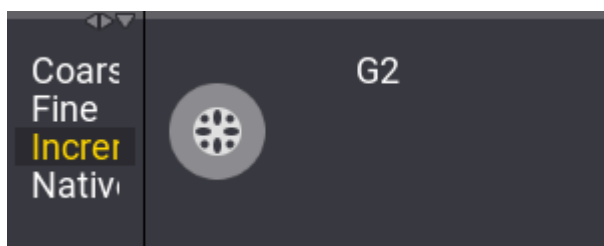
An encoder displays the current resolution in the center of the encoder symbol within the encoder bar:



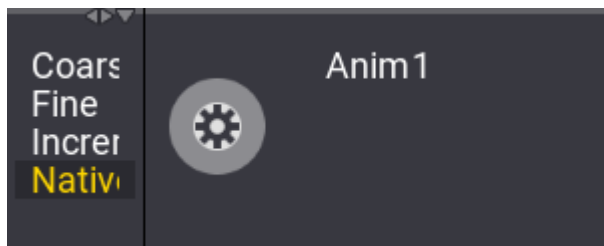
Encoder resolution coarse



Encoder resolution fine



Encoder resolution increment



Encoder resolution native

Furthermore, the factors of value change between turning the inner encoder and all other encoder actions can be defined. See [User settings](#) for more information.



All factors can be selected from a list of predefined factors:

Predefined Factors	Background Used Factors
Div50	0.02
Div25	0.04
Div10	0.1
Div5	0.2
Div2.55	0.39
Div2	0.5
One	1
Mul2	2
Mul2.55	2.55
Mul5	5
Mul10	10
Mul25	25
Mul50	50
Disable	0

14.5. Using the Color Picker

The **Color Picker** is a window that can be used to select a color in fixtures with a mix color system.

The color picker provides convenient access to mixing the desired color using several color mixing and selection options. The method is independent of the fixture's color mix system (LED emitters or color subtraction).

Open the Color Picker

To open the color picker window, see [Add windows](#). It is found in the **Common** tab.

RGB and HSB color space:

Tap **RGB/HSB Space** in the title bar of the color picker to switch the color picker's color space. There are four options:

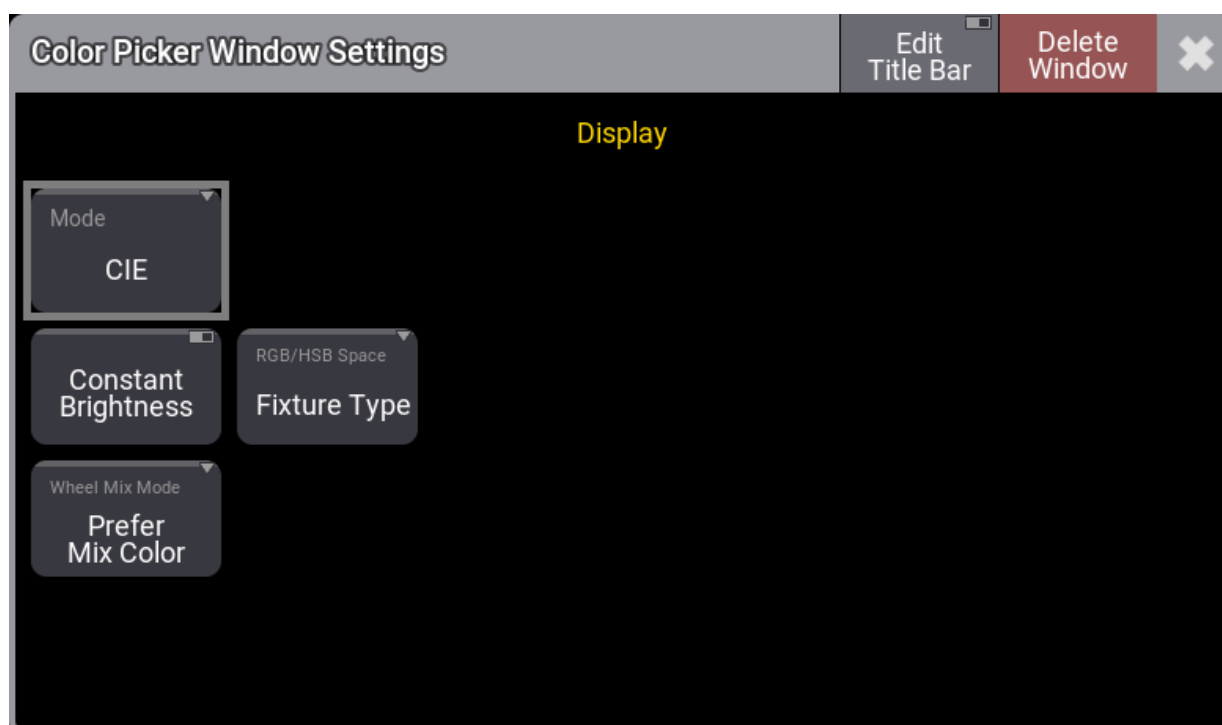
Fixture Type: The color space is defined by the emitters of the respective fixture type.

Standard: Plasa Standard E1.54 for Color Communication in Entertainment Lighting.

Rec.2020: ITU-R BT.2020 or Rec. 2020 is an audiovisual industry standard for ultra-high definition (UHDTV).

Rec.709: ITU-R BT 709 or Rec. 709 is an audiovisual industry standard for high definition (HDTV).

Tap **MA** in the top left corner of the color picker to open the settings.



The mode can be selected using the buttons in the title bar. This is a short description of the different modes.

CIE : A CIE color space area picker with Brightness, Quality, x, and y on-screen faders.

HSB : An HSB area with Brightness and Quality on-screen faders.

Fader : On-screen faders to adjust RGB, CMY, HSB, Brightness, and Quality.

Book : This is a swatch book with colors from different filter manufacturers.

Tap **Edit Title Bar** to define what buttons will be displayed in the title bar of the color picker window.



Quality

The Q fader or quality fader is available when the fixtures have a color mix system of more than three colors. It controls how the colors are mixed.

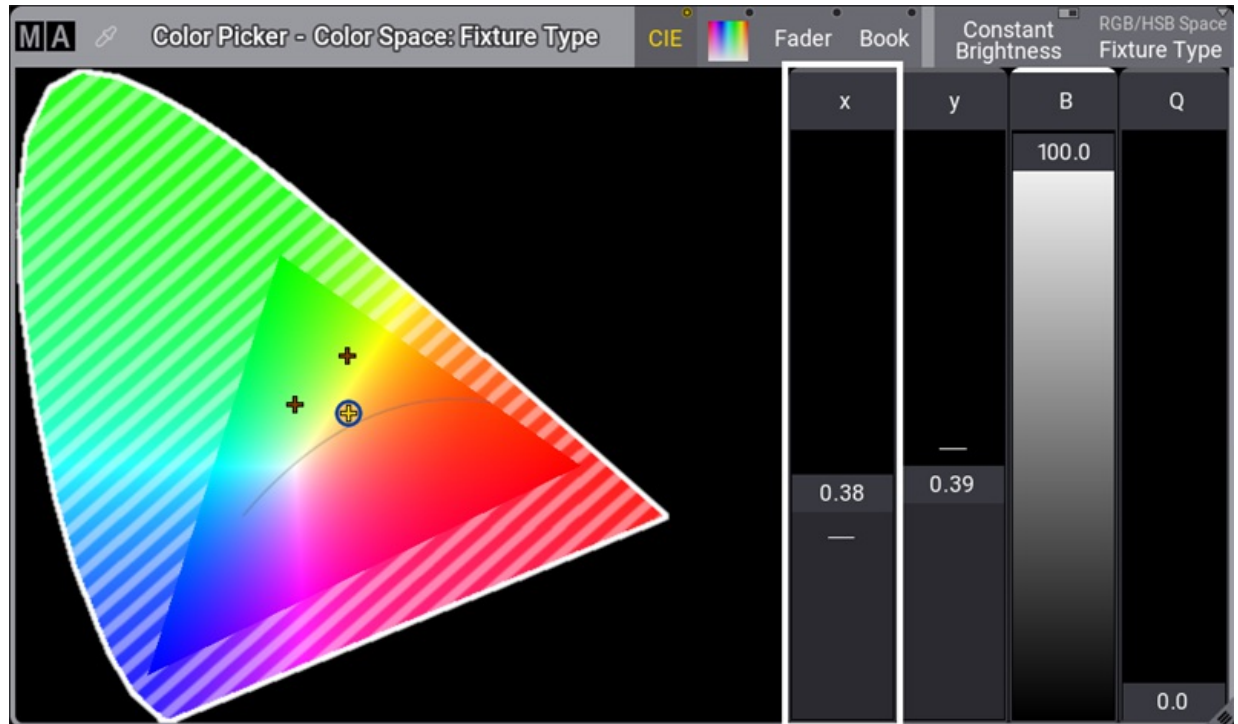
Q at 100 results in a kind of small-band mixing (the specialized emitters are used). 0% results in a broadband mix. That uses as many emitters as possible to mix the color.



Hint:

The **Align** function can be used together with the color picker.

CIE



The CIE (Commission Internationale de l'éclairage) standard uses a figure that indicates the visible light spectrum.

The RGB triangle shows the colors that the specific fixture can mix.

- To select a color, tap inside this area.

If the fixture has more than 3 mixing colors, the shaded part will become smaller.

The color picker offers the Constant Brightness Mode. The Constant Brightness mode can be enabled by tapping **Constant Brightness** in the title bar. The default setting is off.

If the constant brightness is disabled, the selected color is mixed with maximum brightness when the brightness fader is set to 100%. In this case, the fixture's output intensity is not kept constant but changes with the color. If the constant brightness mode is enabled, the maximum overall brightness is limited to the brightness of the fixture's darkest emitter. Changing the color in constant brightness mode does not change the output intensity of the fixture.



Hint:

When enabling the constant brightness mode while the brightness fader is currently positioned above the constant brightness color mixing range, the CONST B fader will become red, showing a value of > 100 %. To ensure constant brightness color mixing, the fader needs to be moved to ≤ 100 %.



Hint:

The color mixing and the constant brightness mode work better the more precise the fixture type's emitter data is.

Except for the Fixture Type color space, the gamut of the selected color space is displayed in the CIE color picker with a white line. The shaded area only depends on the emitters of the fixture. It does not change with the Color Space (only the small white triangle changes with the selected color space). Color mixing in the RGB tab and the HSB Color Picker depends on the color coordinates of the RGB primaries of the selected color space.



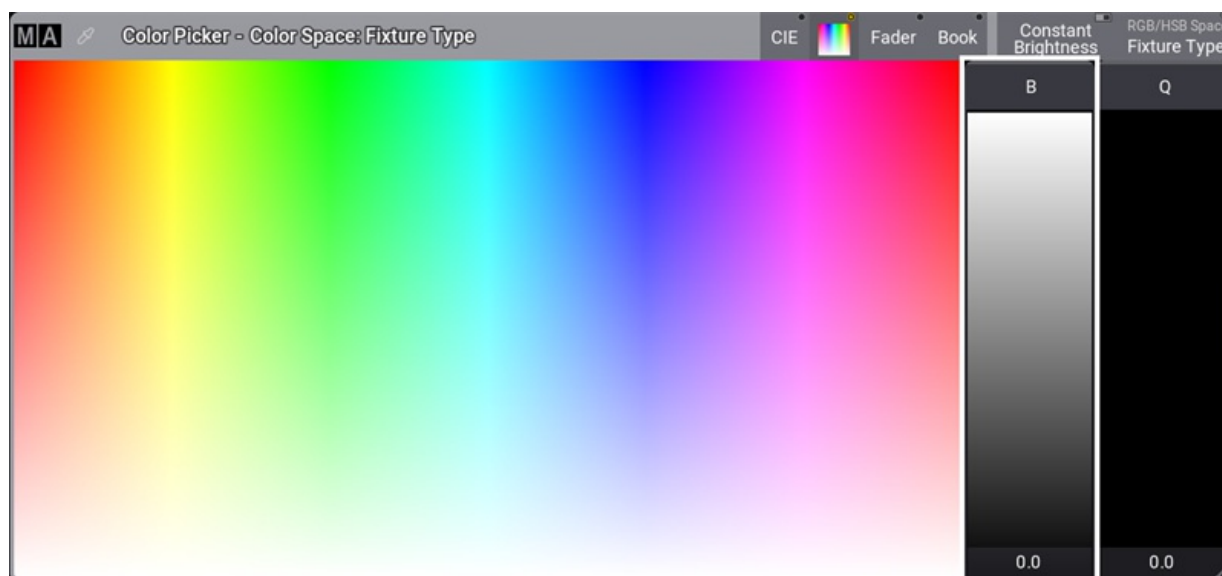
Hint:

If a color is picked in the CIE Color picker outside of the gamut of the selected color space, the faders in the RGB tab will show values below 0% or above 100%.

The CIE Color Picker displays the spectral profile (or **curve**) at a specific temperature corresponding to a specific peak wavelength and vice versa. As the temperature of the black body increases, the peak wavelength decreases (Wien's Law). The intensity (or flux) at all wavelengths increases as the temperature of the blackbody increases. That is what we call the **black body curve**.

Color Picker

Tap the HSB field symbol in the title bar, also known as a Color Picker, to adjust the color mix.

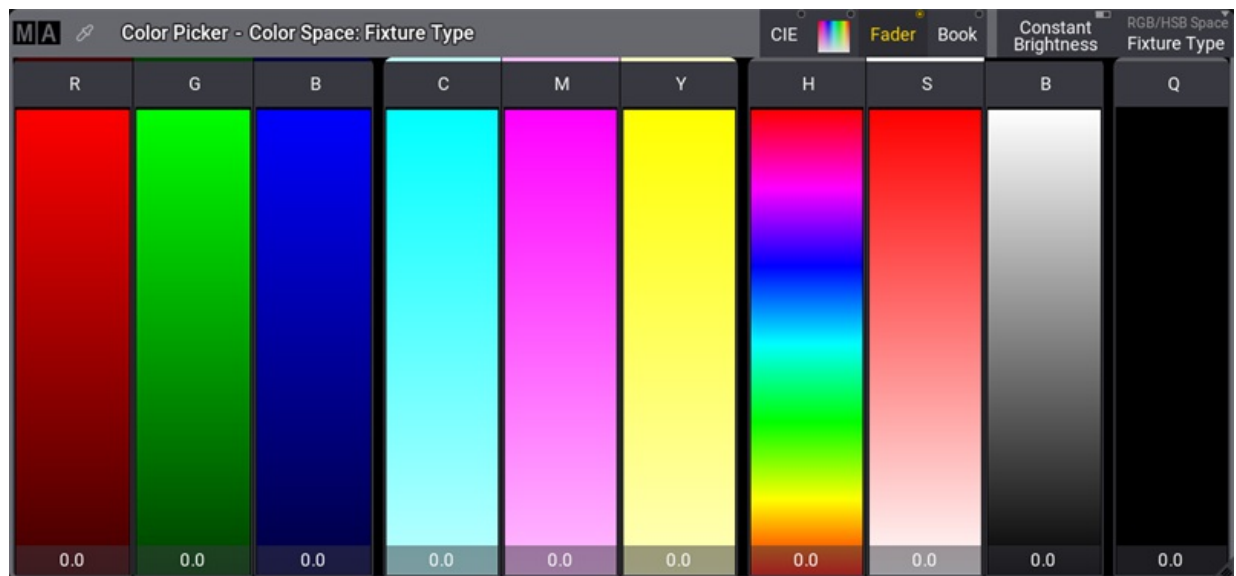


Color picker HSB mode

Here it is possible to tap a color in the HSB field. The x-axis (left/right) is the Hue value. The y-axis (up/down) is the Saturation value, and Brightness is the B-fader on the right side.

Fader

On-screen RGB, CMY, HSB, Brightness, and Quality faders.



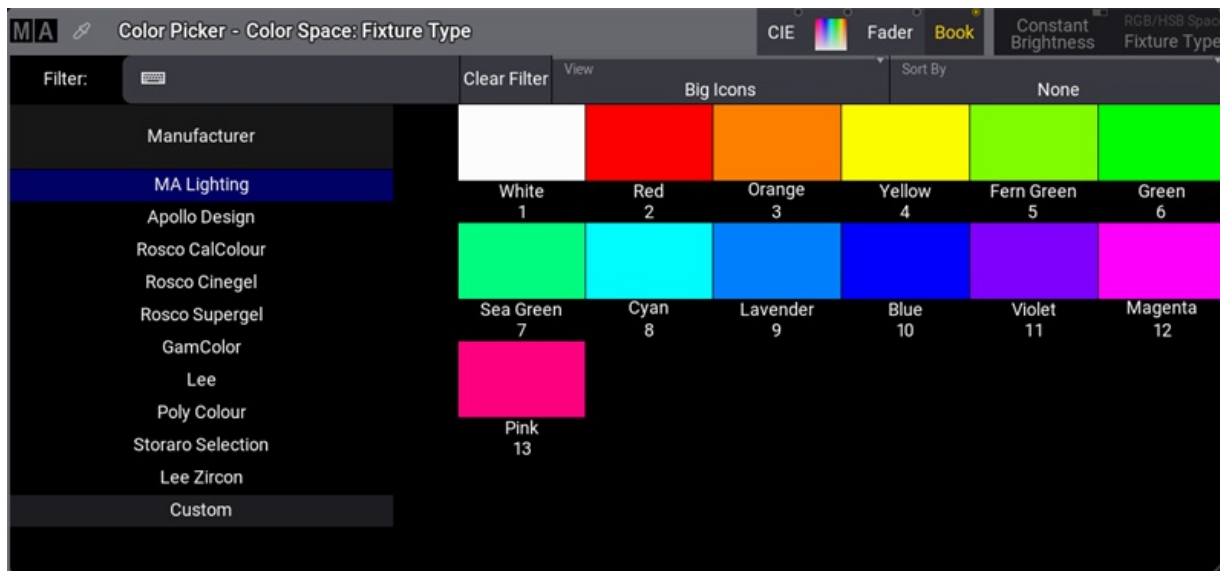
Color picker fader control

Here it is possible to adjust the colors using RGB, CMY, or HSB. The Three-color systems are interlinked. This means adjusting the colors in RGB also moves the CMY and HSB faders.

Book

The swatch book is a library of filter colors from different manufacturers. This can also be accessed using the **Gel keyword**.

The manufacturers are listed on the left side, and the right side displays the manufacturer's colors.



Color picker book mode

There are three buttons at the top of the window.

- **Filter** is to enter a name or a key of the color to limit the list of colors.
- **View** defines how the color list is displayed. It is a swipe button with the following options:
 - **Big Icons** - This displays a big color example. Below the color is the name and color key.
 - **List** - This displays a list with Name, Key, and Color columns.
 - **Small Icons** - displays a small color field only.
- **Sort by** has three different ways to sort the displayed colors. It has the following options:
 - **None** - the list is not sorted and is displayed in the order in the library.
 - **Key** - Sorted by the color key number.
 - **Name** - The colors are sorted by the color name.

The manufacturer list includes many of the manufacturer's filters. It also includes MA Lighting. The MA Lighting library contains all the primary and most used colors.

Color Wheels

When working with fixtures with color wheels, the software chooses the color as close as possible to the picked color from the color picker. When the fixture has two or more color wheels, the software will use all wheels (also combined) to get a color as close as possible.

To change the wheel mix mode, tap **MA** in the title bar. The Color Picker Window Settings opens.

There are three different **Wheel Mix Modes** available:



- **Mix Color Only:**
Uses only color-mixing engine attributes, for example, RGB.
- **Color Wheel Only:**
Uses only color wheel attributes.
- **Prefer Mix Color:**
Uses mainly the color mixing engine attributes. If needed, the color wheels will be used as well.



Restriction:

When a fixture type has more than one color wheel, the color picker can only handle up to 255 color combinations across the color wheels.

The **Fixture Sheet** shows the selected color for the individual wheels. The combined color output is displayed in the sheet.

14.6. Gel Pool



Important:

New users for Pools, read **Pool Windows topics** first.

The Gel Pool displays manufacturers and their corresponding gel series. Each gel series has different amounts of gel in its pool. Manufacturer gel pools are locked.

Each Gel in the manufacturer pool displays:



Gel Pool

1. Color name
 2. Key catalogue number
 3. Color example
- To open the Gel Pool, in the **Add Window pop-up** tap **Pools** and **Gels**.
 - To select a manufacturer catalogue, tap **<MA>** under the MA Logo in the top left corner. The Select Display Pool pop-up opens.



Custom Gel Pool

A Custom Gel Pool is able to save gels at personal preferences.

It is possible to change the name, the key and the color for each gel.

To copy gels in the custom gel pool:

1. Open a manufacturer gel pool and a custom pool next to each other.
2. Use the copy function in the swipecy command. For more information, see Swipecy Commands in **Pool Windows**.

The gel is copied into the custom gel pool.

To edit a gel in the custom pool:

1. Do the Swipecy Command for the gel you want to edit.
2. Enable **Settings** option in Edit GelPool Window.
3. Edit **Name**, **Key**, and/or **Color**.
4. To exit the GelPool Editor tap **X** in the top right corner.

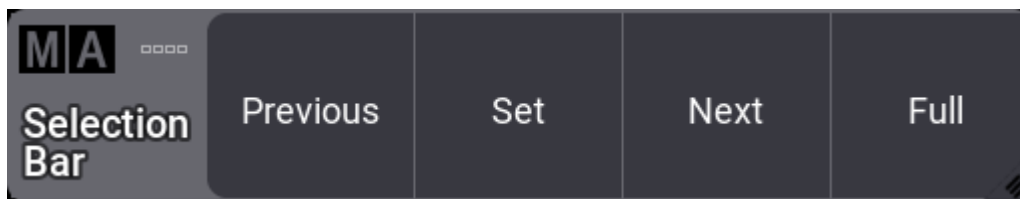
The custom gel is edited.

For more information see **Gel keyword** and Book in **Using the color picker**.

14.7. Selection Bar

The selection bar is a quick toolbar that gives access to some of the most used hard keys used for selecting fixtures.

It is a window that can be created like any other window. Learn about how in the [Add Windows topic](#). It is found in the **More** tab.



Selection Bar

This window can be half field height.

There is only one setting for this window. The background can have an [appearance](#).

There are four buttons in the bar:

- **Previous** - Selects the previous fixture.
- **Set** - Resets the MAticks.
- **Next** - Selects the next fixture.
- **Full** - Gives the selected fixtures a full intensity value on the dimmer.

They behave just like the hard keys. Follow the links to learn about them.

They are also handy when working with [MAticks](#).

14.8. Align

The Align function appears in different places in the software. There is a dedicated **Align key**. There is an **Align keyword**. **Align** is accessible on the left side of the encoder bar. There is an **Align Bar** that can be created as a window.

Align is used to distribute attribute values between two or more values. There are five different align modes and Off. Read about them below.

The default is a linear transition between the values, which can also be adjusted. There are four different **Align Transition** options:

- **Linear** (default):
Spreads the values with the same spacing.
- **Sinus**:
Spreads the values as if the fixtures were placed on a sinus curve. The values themselves will not represent the sinus form. Depending on the Align mode, this results in smaller value gaps at the beginning and end of the range and bigger gaps towards the center of the range, or vice versa.
- **Slow**:
The gaps between the values will be small at the beginning of the range and increase toward the end of the range.
- **Fast**:
The gaps between the values will be big at the beginning of the range and decrease toward the end of the range.

These can be accessed using the **AlignTransition keyword**, the **Align key**, or the **Align Bar**.

The Align function can be used for different attributes. Dimmer, position, and color are the most common. The examples below use tilt or dimmer attributes.

By default, the align mode is set to Off, and the transition is Linear. The result is the encoder will adjust all the selected fixtures equally.



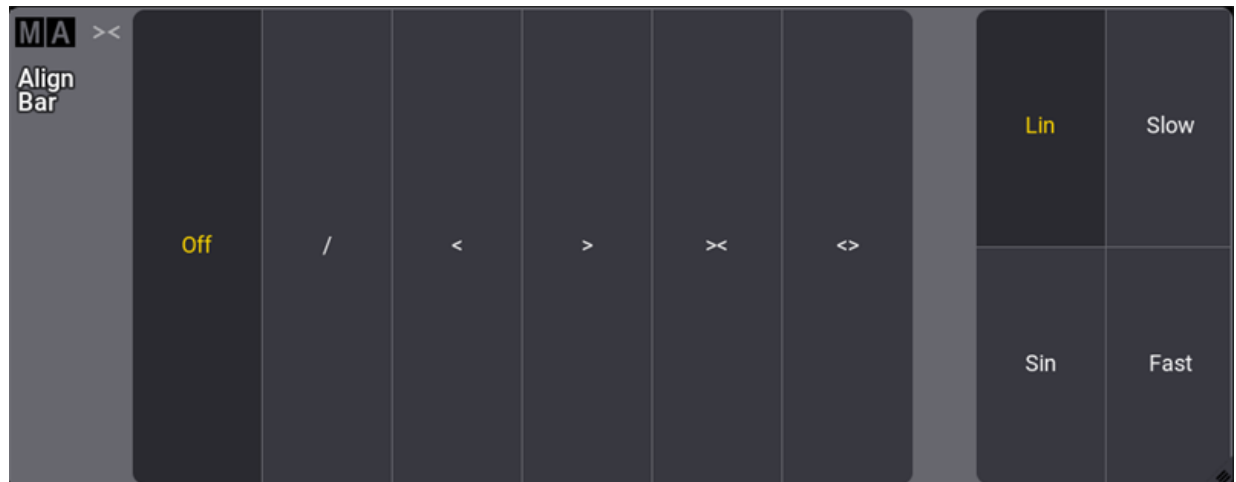
Important:

The selected order of the fixtures is important. The attribute will be adjusted proportionally to the selected order.

The align mode is active until a new attribute is adjusted.

Align Bar

The **Align Bar Window** gives fast access to all the align functions. It has both the align modes and the align transitions.



Align bar with default settings

It can be created like any other window using the **Add Window pop-up**. It can be found in the **More** tab.

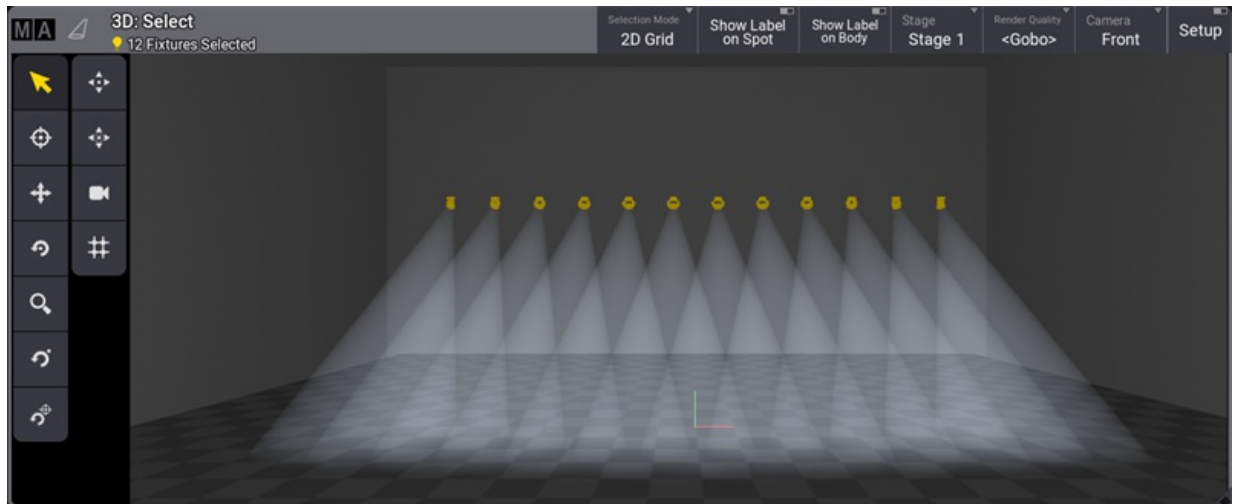
If align is used a lot, then this is a convenient window. Read the descriptions above and below to understand the different modes.

For the following examples, fixtures 1 thru 12 were patched from left to right. For more information, see **Patch and Fixtures Setup**.

Align /

This mode is great for fanning fixtures. Follow the steps below to create a linear fan out.

1. Select fixtures 1 thru 12.
2. Set the dimmer to full.
3. Tilt the fixtures up using the encoder wheel.
4. From the Encoder bar, tap **Position**.
5. Tap and hold **Align** on the left side of the encoder bar. This will open the align dropdown menu, then select **/**.
6. Turn the **Pan** encoder wheel clockwise to fan the fixtures out and counterclockwise to fan the fixtures in.

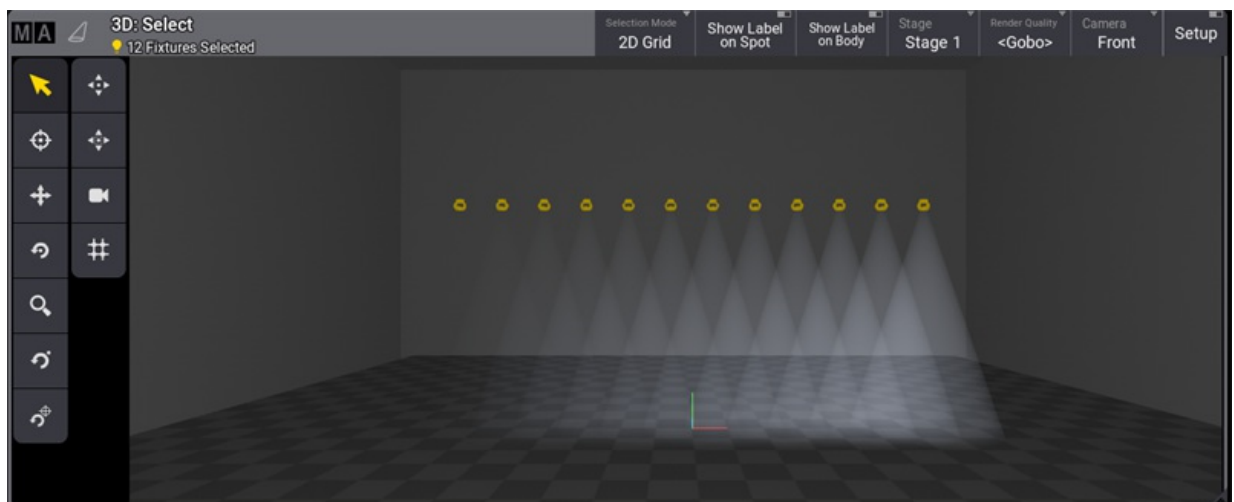


Align mode / with pan

Align <

This align mode increases the values from the first selected fixture.

1. Select fixtures 1 thru 12.
2. From the Encoder bar, tap **Dimmer**.
3. Tap and hold **Align** on the left side of the encoder bar. This will open the align dropdown menu, then select **<**.
4. Turn the **Dimmer** encoder wheel clockwise.

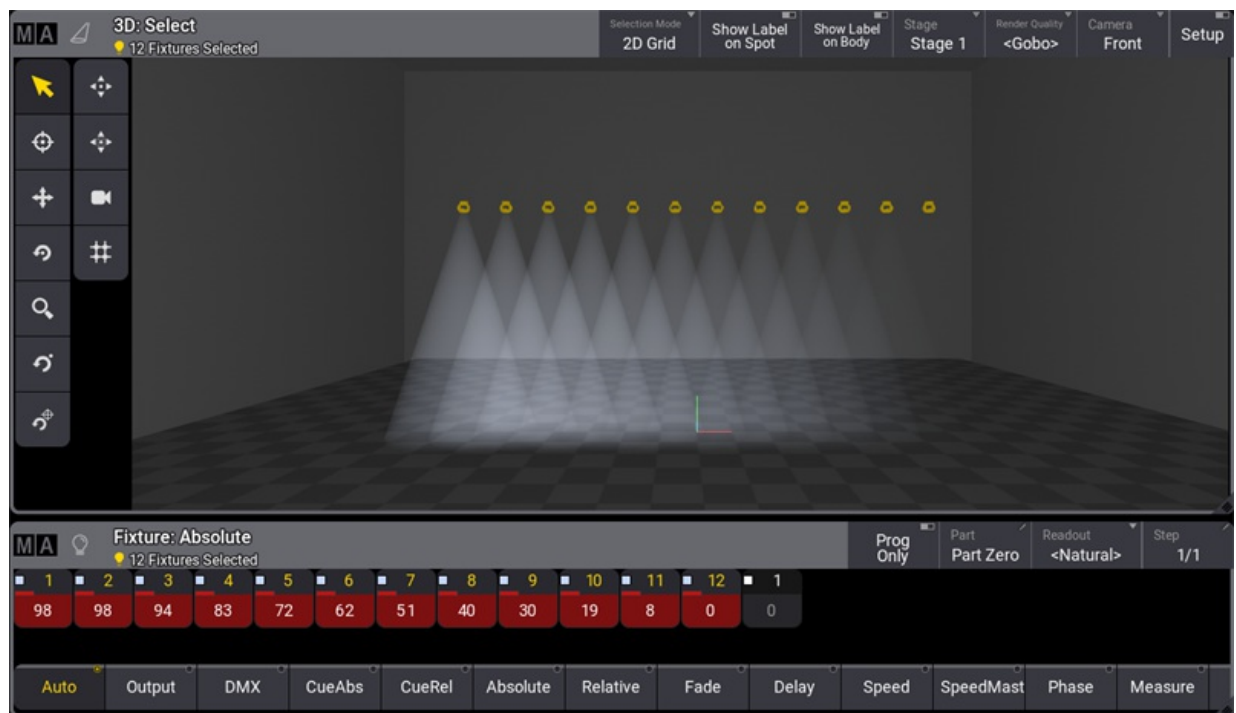


Align mode <

Align >

This align mode increases the values from the last selected fixture.

1. Select fixtures 1 thru 12.
2. From the Encoder bar, tap **Dimmer**.
3. Tap and hold **Align** on the left side of the encoder bar. This will open the align dropdown menu, then select **>**.
4. Turn the **Dimmer** encoder wheel clockwise.

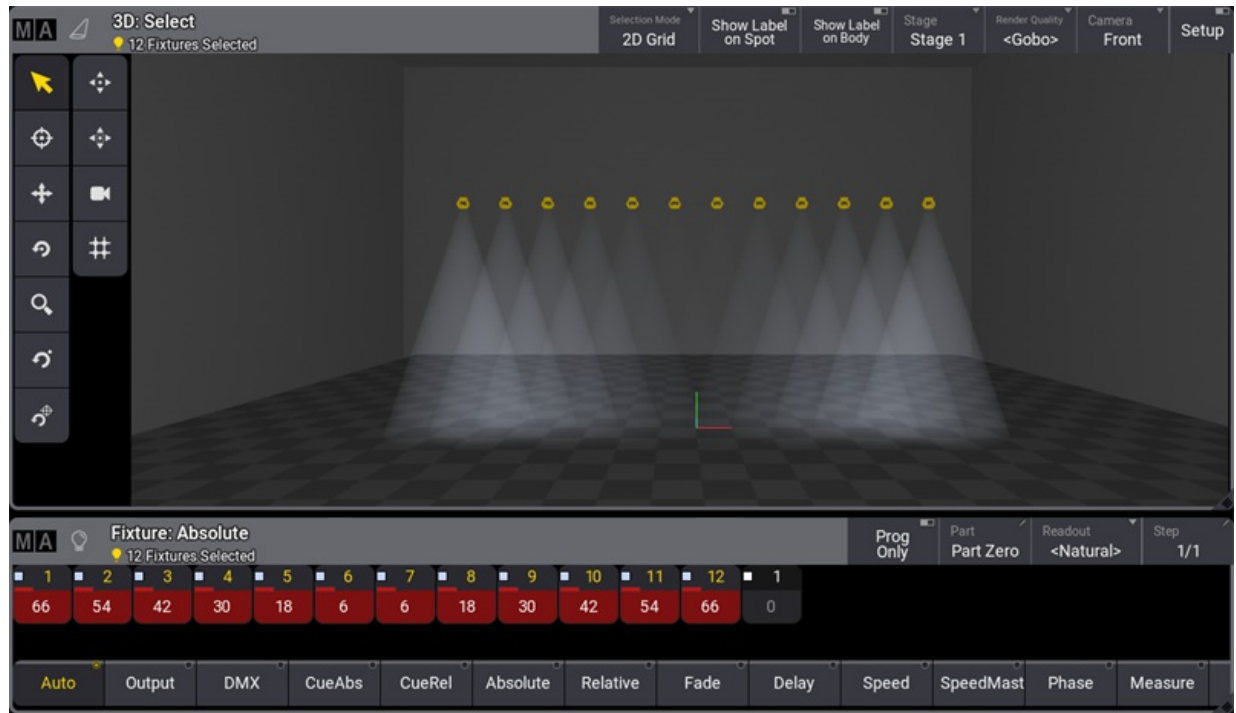


Align mode >

Align ><

This align mode increases the values from the center of the selection.

1. Select fixtures 1 thru 12.
2. From the Encoder bar, tap **Dimmer**.
3. Tap and hold **Align** on the left side of the encoder bar. This will open the align dropdown menu, then select **><**.
4. Turn the **Dimmer** encoder wheel clockwise.

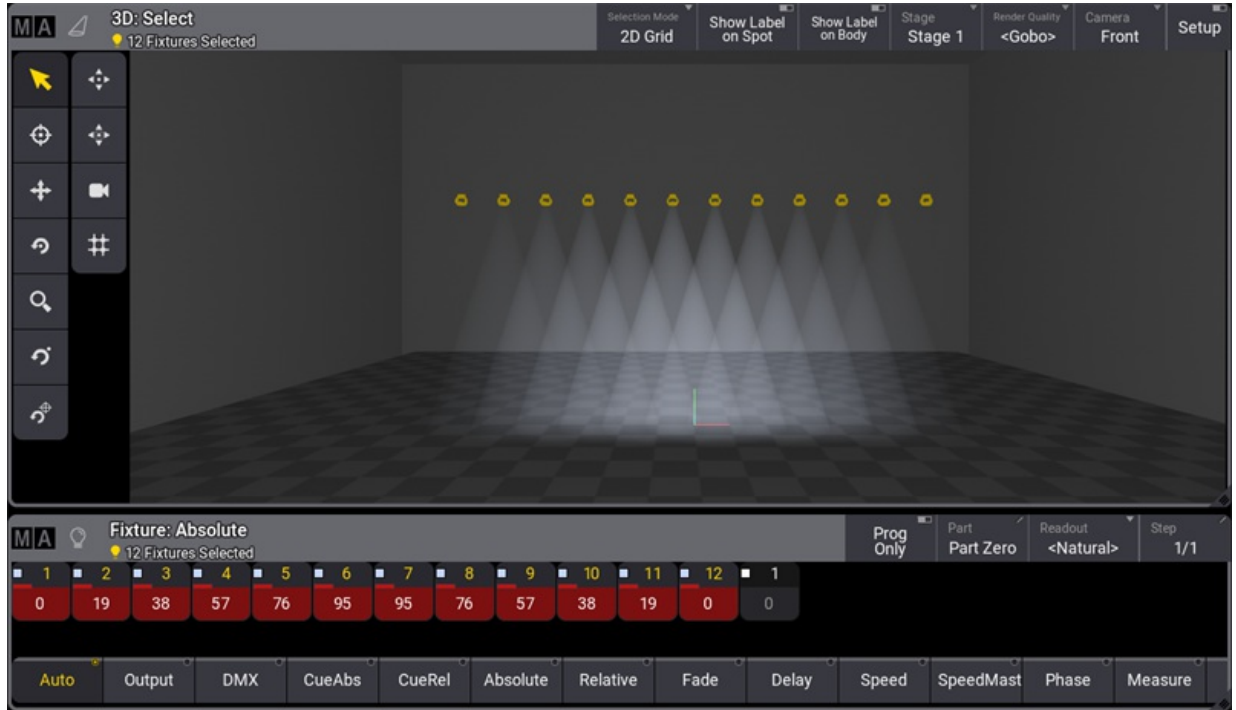


Align mode ><

Align <>

This align mode increases the values from the first end last fixture of the selection.

1. Select fixtures 1 thru 12.
2. From the Encoder bar, tap **Dimmer**.
3. Tap and hold **Align** on the left side of the encoder bar. This will open the align dropdown menu, then select ><.
4. Turn the **Dimmer** encoder wheel clockwise.

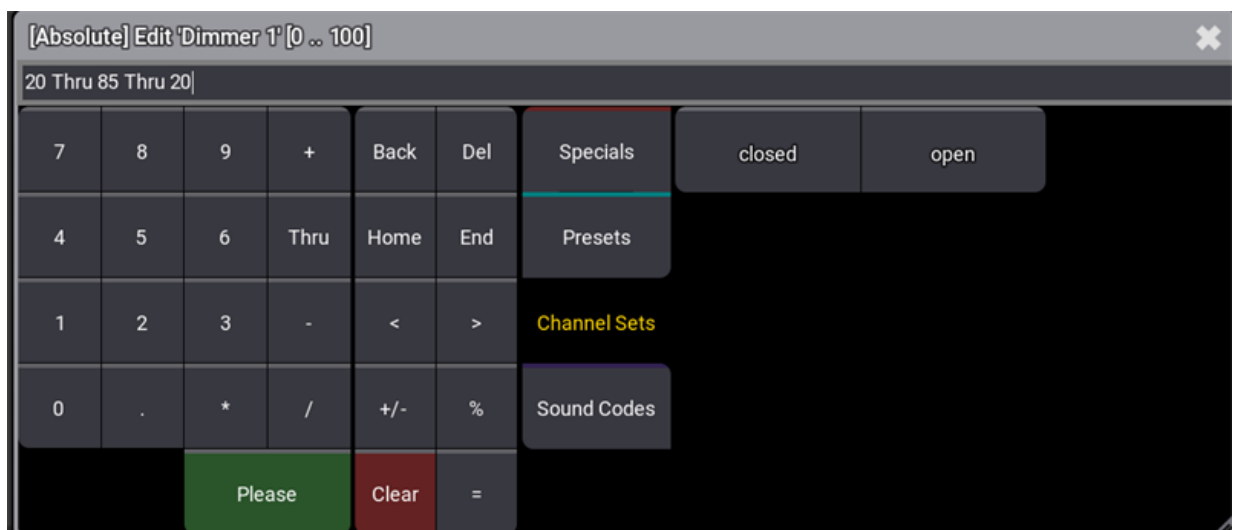


Align mode <>

Align in Other Ways

It is possible to get the same align functions with the **Calculator** or by the command line.

Tap the encoder belonging to the attribute that shall be aligned, and the calculator appears. It is now possible to enter the align values like the example below.



Calculator



Another way to align attributes is by the command line. Select the fixtures and set the dimmer value like the example below.

```
MA User name[Fixture]> At 75 Thru 10 Thru 75
```

14.9. Selection Grid



Fixtures can have information about their position in a 3D selection grid.

Each fixture is symbolized as a box using a space in the grid.

The grid is a way to organize the fixtures in relation to each other but not necessarily their position in the 3D Viewer. A useful tool for positioning fixtures in the selection grid according to their position in 3D space is called 3D view to Selection Grid. For more information about this tool, see [3D Viewer](#).

The grid is used when a range of values is applied. Fixtures positioned at the same X and Y axes but at different Z axis will apply the same value.

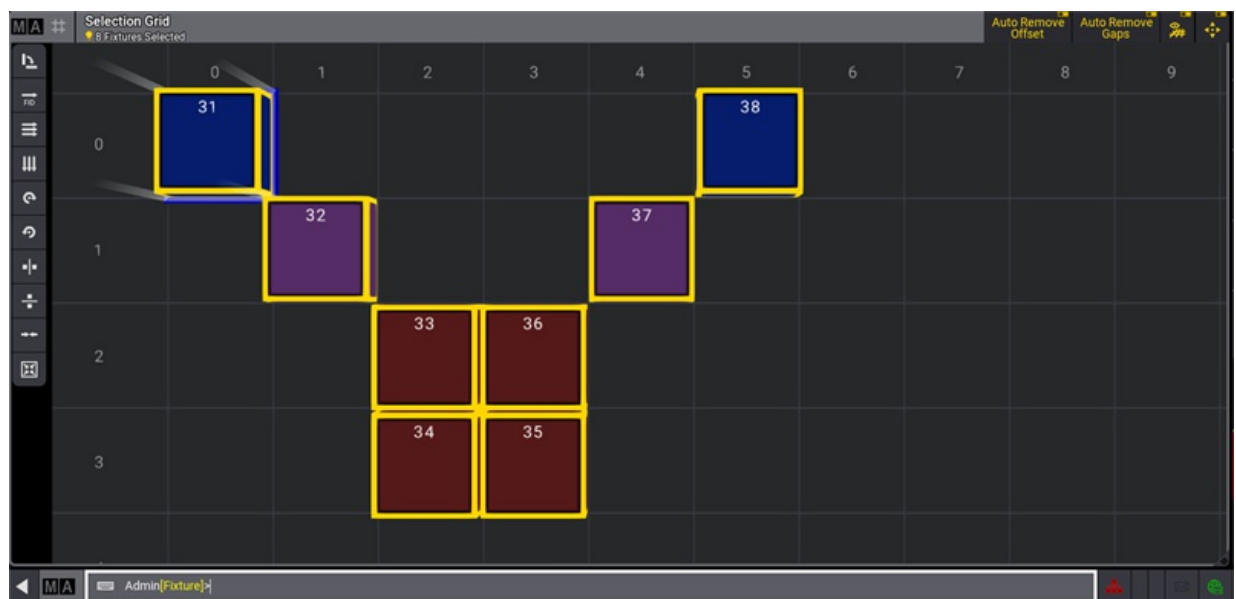
The following toggle buttons are located in the title bar:

- **Auto Remove Offset**: Removes the blank space between the beginning of the grid and the first fixture when it is enabled.
- **Auto Remove Gaps**: When enabled, removes the gaps in the fixture selection.
-  **(Show Grid)**: Toggles the visibility of the grid.
-  **(Zoom to Fit)**: Resets the window to fit all the fixtures in the grid.

In the first example image below, the fixtures were selected using a group with the grid information, and then colors were assigned using the **Align** function in the color picker. The value range is applied from the lowest column to the highest column number in the selection grid.

At the top center of each box, the Fixture ID and custom Fixture IDs are displayed.

The number of selected fixtures is displayed in the title bar.



Selection grid

Grid Cursor

Fixtures are positioned based on the position of the grid cursor. The grid cursor is the blue cells in the grid.

The grid position 0/0/0 is marked with a white frame.

Various commands can move the grid cursor:

- Tapping the arrow keys of the keyboard.
- Pressing **MA** + **2** / **4** / **6** / **8** on the number block of grandMA3 hardware
- Tapping in the selection window (only selects at the 2D XY plane).
- Using the **Grid keyword**.

For example, moving the cursor to X position 3, Y position 2, and Z position 1, press the following keys:

MA + **X3 | Grid** **1** **Thru** **3** / **2** / **1** **Please**

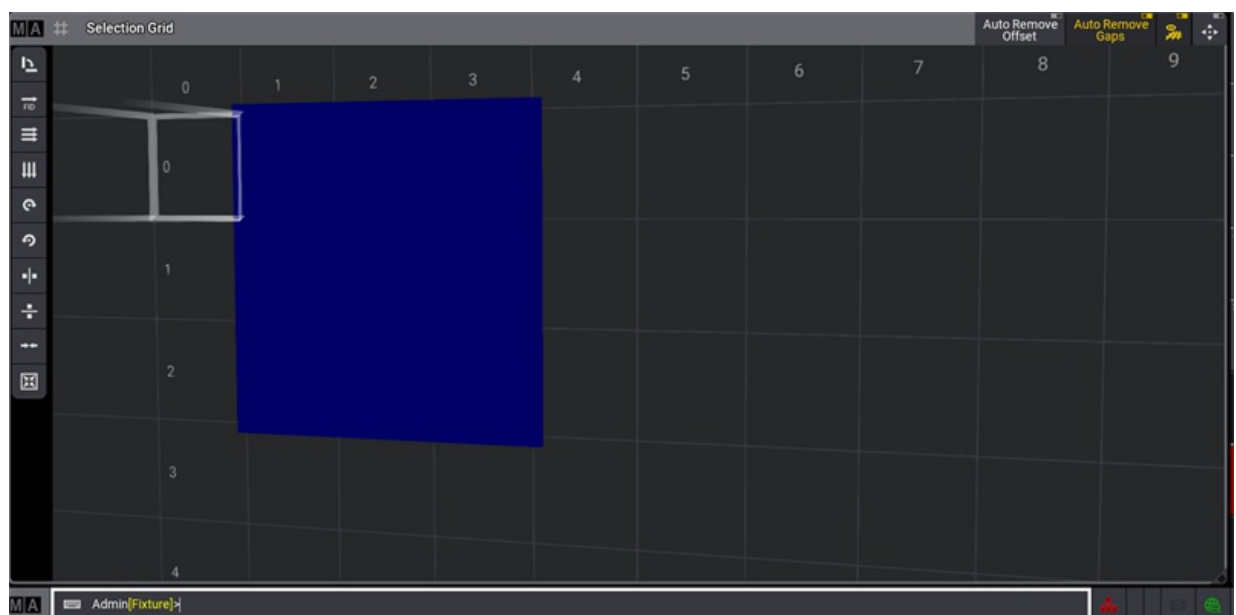
The Z-axis can be omitted if it is 1.

Multiple Cells

The grid cursor can be more than one cell. A rectangular area of grid cells can be selected.

To select a range of cells:

- Use the **Thru keyword** and specifying the beginning cell and the ending cell.
For example, press the following keys: **MA** + **X3 | Grid** **1** **Thru** **3** / **2** **Please**
- Press and hold **Shift** and move the grid cursor with the arrow keys to define the size of the rectangle.
- Press and hold **Shift** and tap the desired end point of the rectangle.



Multiple cells are selected in the selection grid.

Selecting fixtures after defining the grid puts the fixtures in the area. If only one cell is selected, then the grid cursor moves to the following free cell in the X-axis and positions horizontally the fixtures from this location. Moving the grid cursor to one field "right" (positive X-axis) with every placed fixture.

The grid numbers cannot currently be negative.

Store the Grid

When the fixtures are positioned in a grid, it can be stored in groups, presets, or other objects using the grid information. Use any available store methods (buttons, screens, or any combination) to store the group, preset, or another object.

Read more about storing groups in the [Create Groups topic](#) or generally about groups in the [Groups topic](#).

Read about storing presets in the [Create Presets topic](#) or, more generally, about presets in the [Presets topic](#).

Read more about storing cues in the [Store Cues topic](#) or the [cues and sequences section](#).

Adjust the Selection Grid




The grid can be rotated by pressing the window with a single finger and moving the finger around the screen. The same can be achieved by moving the mouse while having the left button pressed.








The grid can be zoomed using a pinch motion with two fingers on the touch screens. A scroll wheel on a mouse also zooms.

The grid can be moved around by touching the screen with two fingers and moving them around the screen. This can be done with a mouse by keeping the right mouse button pressed while moving the mouse.

Grid Tools

The Grid Tools are available on the left side of the Selection Grid window. They can be used to modify the selection of the current fixtures in the Selection Grid window.

-  **Transpose:** Interchanges the X and Y grid coordinates of every selected fixture.
-  **Linearize Numerical:** The currently selected fixtures are rearranged into a line starting at Grid 0/0/0. The new order of the fixtures is in ascending order by their Fixture IDs (and secondarily by the CIDs).
-  **Linearize Left To Right, Top to Bottom:** The currently selected fixtures are rearranged into a line starting at Grid 0/0/0. The new order of the fixtures is determined to work from left to right first and then from top to bottom, based on the current grid arrangement.

-  **Linearize Top to Bottom, Left to Right:** The currently selected fixtures are rearranged into a line starting at Grid 0/0/0. The new order of the fixtures is determined to work from top to bottom first and then from left to right, based on the current grid arrangement.
-  **Rotate 90° clockwise:** Rotates the currently selected fixtures by 90° in a clockwise direction along the XY plane.
-  **Rotate 90° counterclockwise:** Rotates the currently selected fixtures by 90° in a counterclockwise direction along the XY plane.
-  **Mirror Grid at X-axis:** Mirrors the fixtures in the grid along a vertical mirror line.
-  **Mirror Grid at Y-axis:** Mirrors the fixtures in the grid along a horizontal mirror line.
-  **Center Alignment:** When different numbers of fixtures are selected on different Y coordinates, they will be aligned together to their center on the X-axis.
-  **Compress:** Removes all gaps between fixtures and moves the selection to the left and up so that the selection starts at 0/0/0.

Use the Grid Keyword to trigger the grid tools from the command line.

Pressing and holding **MA** + **X3** enters the Grid keyword into the command line.

```
MA User name[Fixtures]> Grid 'Transpose'
```

These are the commands for each tool:

- Transpose: **Grid 'Transpose'**
- Linearize Numerical: **Grid 'Linearize' 'Numerical'**
- Linearize Left To Right, Top to Bottom: **Grid 'Linearize' 'LeftToRight'**
- Linearize Top to Bottom, Left to Right: **Grid 'Linearize' 'TopToBottom'**
- Rotate clockwise: **Grid 'Rotate' 'Right'**
- Rotate counterclockwise: **Grid 'Rotate' 'Left'**
- Mirror Grid at X-axis: **Grid 'Flip' 'X'**
- Mirror Grid at Y-axis: **Grid 'Flip' 'Y'**
- Center Alignment: **Grid 'Align'**
- Compress: **Grid 'Compress'**



14.10. SMART View

The **sMArt View** is a dynamic pool that gives quick access to fixture defined channel sets.

The channel sets are a part of the fixture types.

It can be created like any other window using the **Add Windows** pop-up. It is found in the **Tools** tab, where it is called **sMArt**.



Smart View showing a gobo wheel

Each pool object represents a channel set defined in the fixture profile.

The channel sets cannot be edited using the sMArt view.

Tapping one of the objects puts the respective value into the programmer.

The bottom of the sMArt view shows an attribute selection bar. This makes it easy to select the desired attribute in the currently selected feature group. The example above shows the **Gobo** feature group and the **Gobo Wheel 1 (G1)** attribute.

This bar can be enabled or disabled in the **Settings**. This is the only special setting for this window.

15. Scribbles

The grandMA3 has a scribble pool where it is possible to save the scribbles created within the scribble pool itself or when labeling an object.



Important:

Scribbles created during labeling of a pool or any other object are automatically saved in the scribble pool.



Important:

It is not possible to assign a scribble to the appearance pool and vice versa.

15.1. Create Scribbles



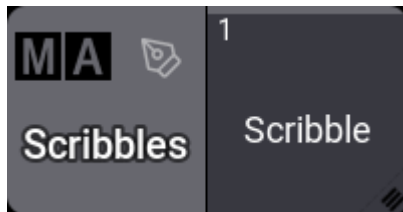
Hint:

It is possible to create scribbles in the scribble pool or when labeling objects.

Create Scribbles in the Scribble Pool

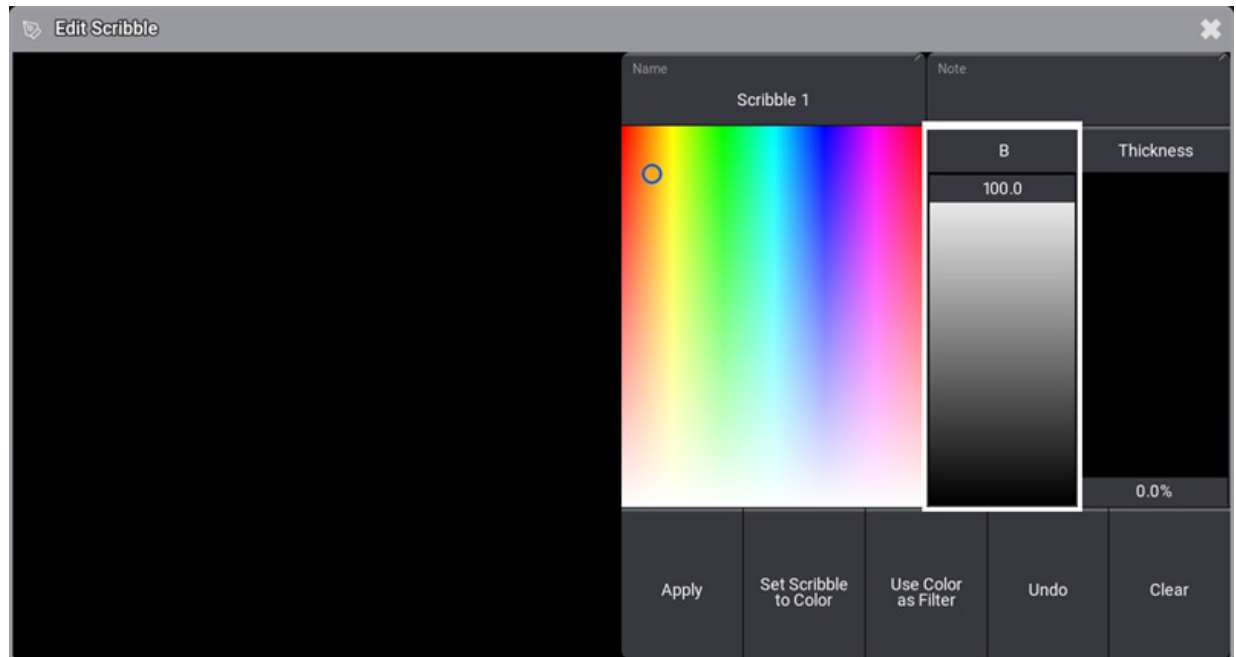
1. Access the scribble pool.
 - Open the **Add Window**.
 - Open **Pools**.
 - Open **Scribbles**.

For more information, see **Add Windows**.



Scribble Pool

2. Store a new scribble.
 - Tap and hold an empty pool object.
 - The new scribble is saved.
3. Create a scribble.
 - Press **Edit** and tap the scribble you saved.
 - The **pop-up Edit Scribble** opens.



Pop-up Edit Scribble

4. Label the scribble:
 - Tap **Name**.
 - The **pop-up Edit Name** opens.
 - Give the scribble a name.
5. Select the color of the scribble.
 - Tap the color picker.
6. Set the brightness of the color.
 - Move the **B** fader.
7. Set the thickness of the scribble.
 - Move the **Thickness** fader.
8. To scribble, move your finger across the scribble pad on the left of the pop-up.
9. To delete the last step made in the scribble, tap **Undo**.
10. To apply the scribble, tap **Apply**.



Hint:


The color and thickness settings of a scribble are saved using a grandMA3 console. Once the console is powered down, these settings are reset.

The scribble is applied to the scribble pool.

Create Scribbles in Other Pool Objects

This example is based on the object of a group pool.

Requirement:

- The group pool is open
 1. Save a new pool object in the group pool.
 2. Label the new pool object.
 - Press **Assign** **Assign** and tap the pool object.
 - The virtual keyboard opens.
 3. Tap  in the title bar of the keyboard.
 - The virtual keyboard expands and opens the scribble pad and its controls.
 4. Follow steps **5 to 8** described in the scribble pool.

The scribble is automatically saved on the next free position in the scribble pool and applied to the group pool object.



Important:

Creating scribbles outside the scribble pool spams the pool because every scribble created with the label function within other pools is saved as a new scribble in the scribble pool.
To learn how to avoid spamming, see [Assign Scribbles](#).

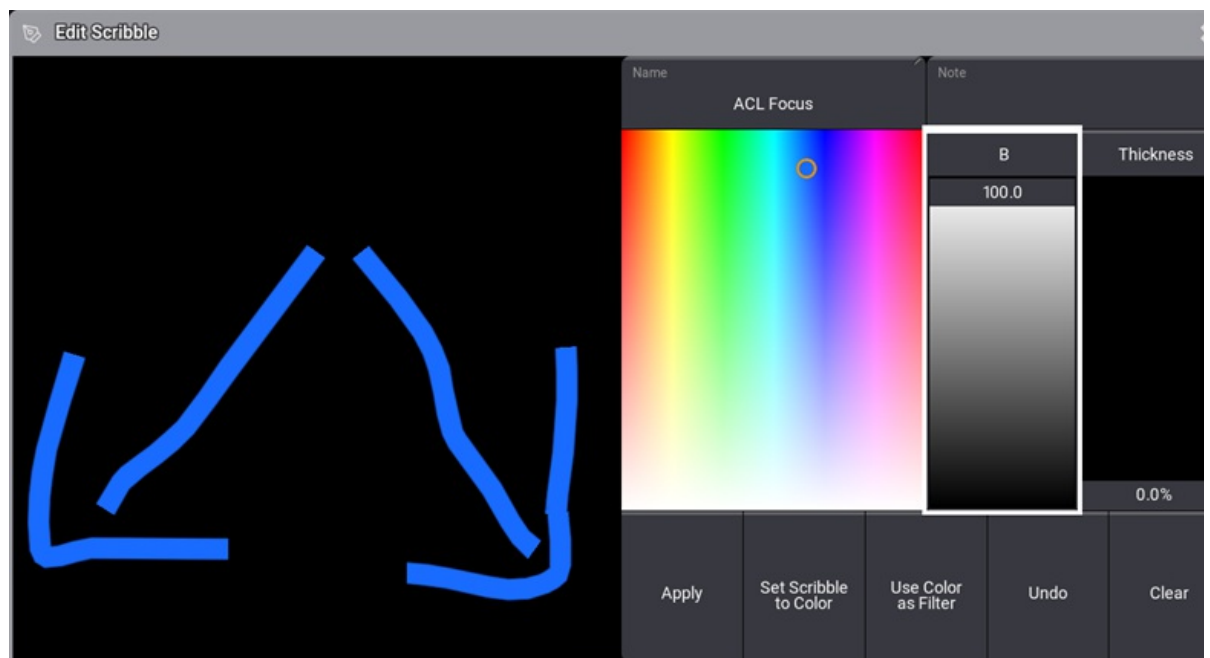
15.2. Edit Scribbles

It is possible to edit an existing **Scribble**.

Edit Scribbles in the Scribble Pool

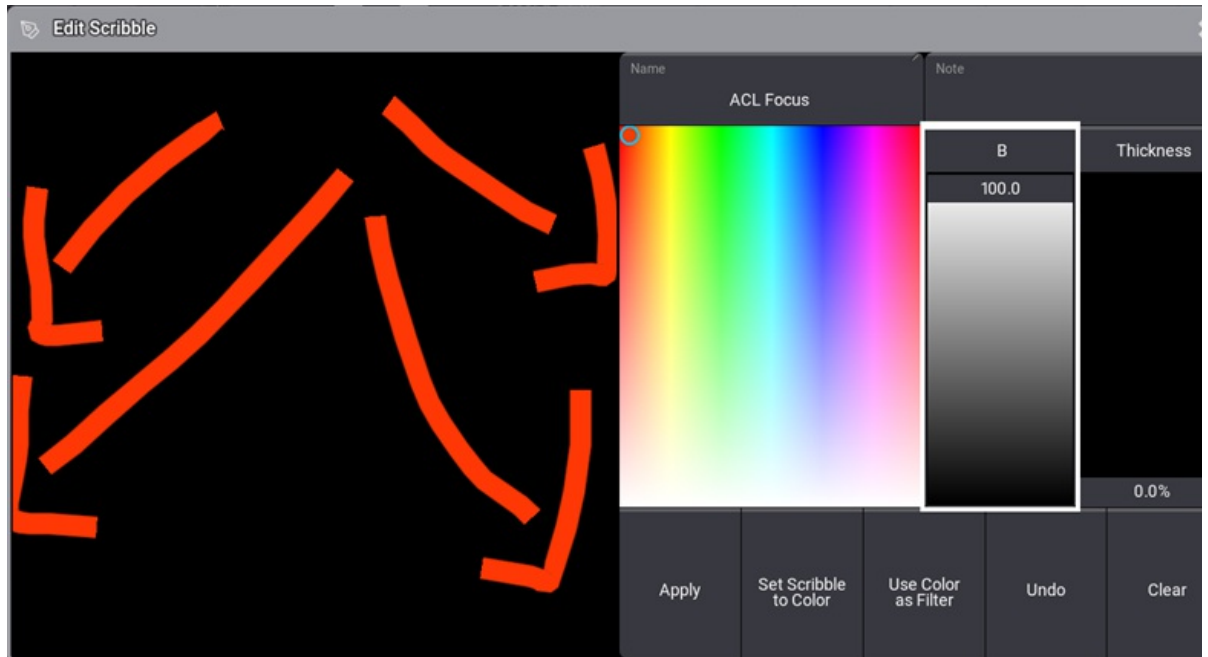
Requirement:

- The scribble is applied in the scribble pool
 1. To edit an existing scribble, press **Edit** and tap it.
-The window **Edit Scribble** opens.



Edit an existing scribble

2. Tap **Undo** to start from scratch or adjust the scribble.



Adjustments in the existing scribble


3. Tap **Apply**.

The edits are saved in the scribble pool.

Edit Scribbles in Other Pools

This example is based on the group pool.

Requirement:

- The scribble is assigned to a group pool object
 1. To edit an existing scribble in the group pool, press **Assign** **Assign** and tap the pool object.
-The Window **Edit Name** opens. Tap  in the title bar to open the scribble area if needed.
 2. Adjust the scribble.
 3. Tap **Apply**.

The edits are automatically saved in the scribble pool.

15.3. Assign Scribbles

You can assign existing scribbles in the scribble pool to other pool objects.



Hint:

To avoid spamming the scribble pool, create the scribbles in the scribble pool and then assign them to other pool objects.

This example is based on a group pool object.

Requirement:

- A scribble was created in the scribble pool.

To assign a scribble to a group pool object:

1. Press **Assign**.
2. Tap the scribble in the scribble pool.
3. Tap the pool object you would like to assign the scribble to.

The scribble is assigned to the group pool object.

15.4. Delete Scribbles

Delete Scribbles in the Scribble Pool

It is possible to delete scribbles using one of the options:

1. To delete single scribbles:

- Press **Delete** and tap the scribble you would like to delete.

-or-

- Use the Swipecy.
For information, see [Pool Windows – Swipecy](#).



Important:

If you assigned scribbles to other pool objects, deleting single scribbles in the scribble pool deletes the links in all pools.

2. To delete several scribbles, type in the command line:

```
MA User name[Fixture]> Delete Scribble 1 Thru 5
```

Deletes scribbles 1 to 5.

3. To delete all scribbles of the scribble pool, type in the command line:

```
MA User name[Fixture]> Delete Scribble 1 Thru
```

16. Images

Imported images are in the **Image Pool**. For more information, see [Pool windows](#).



Important:

The overall size of the media pools has a maximum of 200 MB.

We advise that you keep the image pool as small as possible. For example, keep the maximum image size below 64MB. Do not exceed an image resolution higher than 1920 x 1080.

Images can be used for appearances. Read more in the [Create appearances](#) topic.



Example of the images pool

Edit an image pool object using one of two options:

- Press **Edit** and then tap the desired pool object.
- Open the swipecommands on the pool object and choose **Edit**.

Import Predefined Images

Some predefined test pattern images can be used. Custom images can also be imported - read [below](#).

Workflow:

Images can be imported using the command line.

1. Navigate to the image pool:

```
MA User name[Fixture]> ChangeDestination Image 3
```

2. Now import the predefined library images:

```
MA User name@ShowData/MediaPools/Images> Import Library "*.xml"
```

This command imports the image files from MA into the image pool.

3. Return to the command line root:

```
MA User name@ShowData/ImagePools/Images> ChangeDestination Root
```

For more information, see [ChangeDestination keyword](#).

Import Images Using the Image Pool

Workflow:

Images can be imported using the Image pool.

1. Edit an empty pool object.
2. Tap the **Import** button.
3. Tap **Internal** in the top-right of the title bar to change the drive to the desired source.
4. Select the desired image.
5. Tap **Import**.
6. Close the **Edit Image** pop-up.
7. Before closing the Edit Image pop-up, it is possible to edit the name.

By default, the grandMA3 software uses the **gma3_library/media/images** folder to import and export images. For more information, see [Folder Structure](#).

Import Images from the Import/Export Menu

Workflow:

1. Press **Menu**.
2. Tap **Import/Export**

3. Tap **Import** on the top left corner of the window.
4. Tap **Images**.
5. Select the drive, then tap an empty pool object on the right side of the window.
6. Select the image to be imported.
7. Tap **Import** at the bottom of the window.

Delete an Image from the Pool

Deleting images is like deleting any other pool object. The image disappears in any appearance where it might be used.

There are three common ways to delete images.

Delete an Image Using the Command Line

The important keyword for this is: **Delete**.

This is the syntax for deleting a single image:

Delete Image 3.x

Deleting a range of images using the standard range syntax (Thru, +, and -) is also possible.

For example, if image 4 needs to be deleted:

```
 User name[Fixture]> Delete Image 3.4
```

Or if images 5 to 10 need to be deleted:

```
 User name[Fixture]> Delete Image 3.5 Thru 10
```

Delete an Image Using the Image Pool on a Screen

Requirement:

A visible image pool on one of the screens.

1. Press **Delete**.
2. Tap the image in the pool.

The image is deleted.

Delete an Image Using the Swipey Commands

Requirement:

A visible image pool on one of the screens.



1. Tap and hold the image you wish to delete.
2. Swipe out of the pool object without releasing the screen.
3. Swipe to the **Delete** swipecy and release the screen.

The image is deleted.

16.1. Screenshots

Screenshots can be created by pressing the **Print Screen** key or shortcut **F11** on a built-in or external keyboard. A green flash on all screens indicates that the screenshot has been taken.

An image from each screen is stored as individual png files in the image folder. The file names are generated using this template: YYYYMMDD_hhmmss_displayX.png

The images can be imported into the **image pool** or copied from a console using an SFTP connection. For more information, see [Networking – SFTP connection](#).



Restriction:

There is no keyword or command to create screenshots.

Storing Screenshots on a USB Stick

Screenshots can be stored on a USB memory stick. They are stored as png files.

The USB needs to be the selected drive when **Print Screen** is pressed.

A list of the drives can be seen in the **Command Line History window** by running the following command:

```
MA User name[Fixture]>List Drive
```

It is drive number two if there are no other USB devices connected.

Select the drive using this syntax: **Select Drive [drive_number]**

Now press **Print Screen**.

Bring the USB to a computer. Screenshots are saved to the **/gma3_library/media/images** folder. On USB drives, the **gma3_library** folder is located in the **grandMA3** folder.

Copy Screenshots using SFTP

When accessing the internal drive of the console using SFTP, the **gma3_library** folder is located at the root of the visible folder structure.

Open an **SFTP connection**, navigate to the images folder, and copy the relevant files from the console.

For more information on folder structure, see the **Folder Structure** topic.

17. Video

Videos can be used as a source for an appearance.

Videos can be imported into the Video Pool.





Video pool

The video pool holds the imported videos but can also have streaming video sources using the NDI format.

The video objects can be used as an image source in Appearances. Learn more about appearances in the [Appearances section](#).

The video will then play back in a loop everywhere the appearance is used.

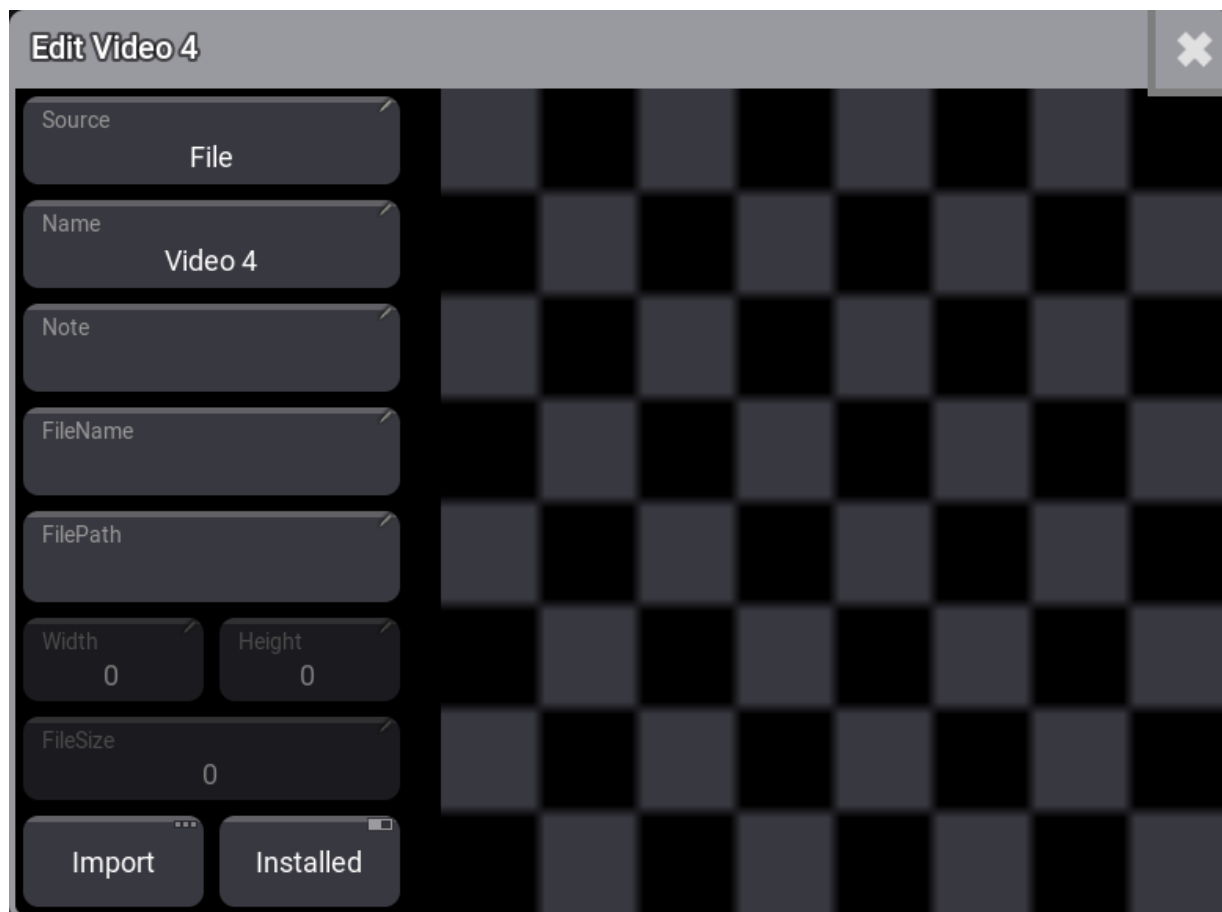
 **Restriction:**
Third-Party Software needs to be activated and agreed to in **Menu** / **Settings** / **Software Update** / **EULA** for videos to be played back. The button is in the lower right corner. Turning it On opens a pop-up that needs to be agreed to.

 **Restriction:**
The overall size of the media pools is limited to a maximum of 200 MB.

Import a Video

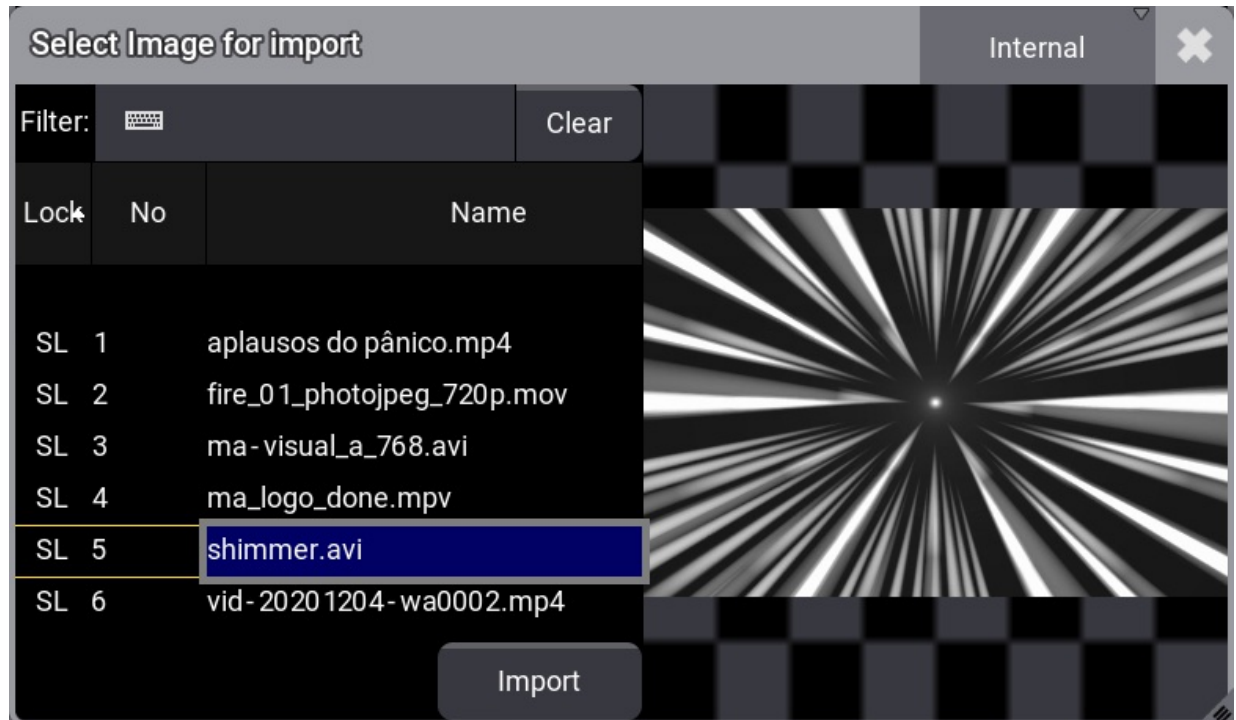
The video file needs to be in the correct folder to be able to be imported. The folder is `/gma3_library/media/videos`. Learn more about this in the [Folder Structure topic](#).

1. Edit an empty video pool object. This opens the **Edit Video** pop-up:



Edit video pop-up

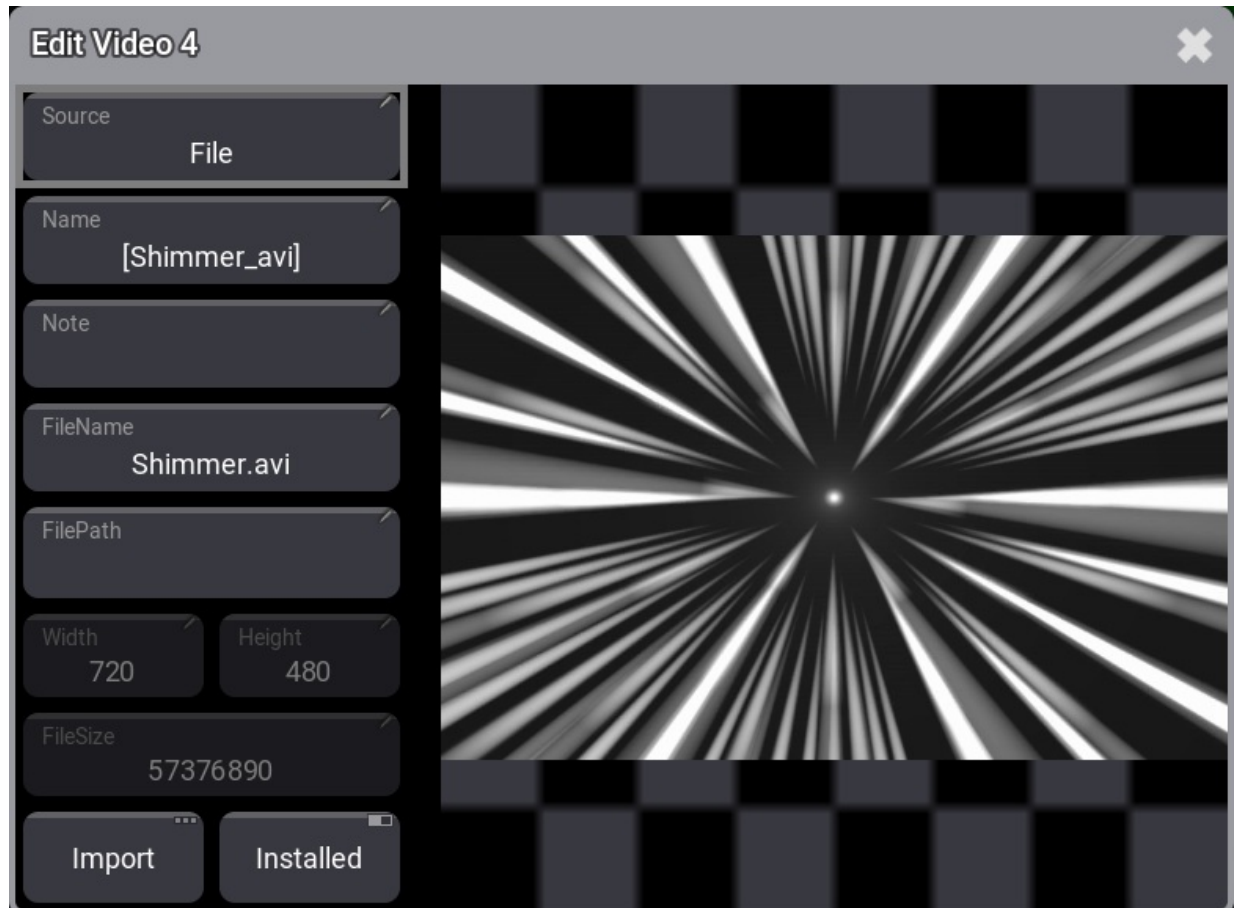
2. Tap **Import** to start the import process. This opens a new pop-up:



Select image for import pop-up


This lists all the video files in the video folder.

3. If needed then tap **Internal** in the title bar of the pop-up to change the selected drive.
4. Tap the desired video file. The video should play back in the preview area. If it does not appear then it is not a supported format or third-party software is not activated.
5. Tap **Import**. This returns to the **Edit Video** pop-up:



Edit video pop-up with a file selected

The editor now shows relevant data like video width, height, and file size.

6. Edit the name if desired.
7. The file is imported and the editor can be closed by tapping the .

Use an NDI Source


An NDI streaming video source can be used instead of importing a video file.

Requirement:

There needs to be an NDI stream in the network.

1. Edit an empty video pool object. This opens the **Edit Video** pop-up.
2. Tap **Source**.
3. Tap **NDI** in the **Select Source** pop-up.
4. Tap **Select Source**. This opens the **Select NDI Source** pop-up.
5. Tap the desired NDI source. If the list is empty then there are no NDI sources in the network or third-party software is not activated.



6. Close the editor by tapping the .

Create an Appearance with Video

Adding a video to an appearance is almost the same as adding an image from the image pool.

The difference is that **ImageSource** needs to change to "Videos" in the title bar of the **Select Image** pop-up.

Learn how in the [Create Appearances](#) topic.

18. Meshes

Meshes are stored in the Mesh **Pool**. Meshes 1 thru 9 are locked and can not be edited.



Important:

For more information on **Meshes**, see **Build fixture types**.

Open a **Mesh Pool** window. For information, see the **Add window** topic.



Mesh pool

In the **Add Window** pop-up, tap **Pools**, then in the **Media** column, tap **Meshes**.

Edit a mesh pool object using one of two options:

1. Press **Edit** and then tap the desired pool object.
2. Open the swipecommands on the pool object and choose **Edit**.

The Edit **UserMesh** pop-up opens.

Import Meshes

To import meshes:

1. Repeat option 1 or 2 above.
2. In the **Edit UserMesh** pop-up, tap **Import**. The **Import mesh** pop-up opens.
3. Select the source from the title bar, for example, Internal.
4. Select the mesh to import.
5. Tap **Import**.

The new mesh pool object is imported.

19. Gobos

Gobos are stored in the Gobo **Pool**. Gobos 1 thru 4 are locked and can not be edited.

When fixtures are patched in the show file, the images of the gobos used by these fixtures are stored in the **Gobo Pool**.

Open a **Gobo Pool** window. For information, see the [Add window](#) topic.

In the **Add Window** pop-up, tap **Pools**, then in the **Media** column, tap **Gobos**.



Gobo pool

Edit a gobo pool object using one of two options:

- Press **Edit** and then tap the desired pool object.
- Open the swipecommands on the pool object and choose **Edit**.

Assign Gobo to Appearances

To assign a gobo to an appearance:

1. Open the swipecommand on the pool object, select **Edit** or press **Edit**, and tap an appearance pool object. The **Edit Appearance** pop-up opens.
2. Tap **Image**. The **Select Image** pop-up opens.
3. From the title bar, tap and hold **ImageSource** and select **Gobos**. The **Select Image** pop-up opens.
4. Select an image and close the pop-up.
5. Continue editing the new appearance if needed and close the **Edit Appearance** pop-up.

The new appearance is created.

20. Symbols

Symbols are stored in the **Symbol Pool**. Symbols 1 thru 14 are locked and can not be edited.

Open a **Symbol Pool** window. For information, see the [Add window](#) topic.

In the **Add Window** pop-up, tap **Pools**, then in the **Media** column, tap **Symbols**.



Symbol pool

Edit a symbol pool object using one of two options:

- Press **Edit** and then tap the desired pool object.
- Open the swipecy commands on the pool object and choose **Edit**.

Assign Symbols to Appearances

To assign a symbol to an appearance:

1. Open the swipecy command on the pool object, select **Edit** or press **Edit**, and tap an appearance pool object. The **Edit Appearance** pop-up opens.
2. Tap **Image**. The **Select Image** pop-up opens.
3. From the title bar, tap and hold **ImageSource** and select **Symbols**. The Select Image pop-up opens.
4. Select an image and close the pop-up.
5. Edit the new appearance if needed and close the **Edit Appearance** pop-up.

The new appearance is created.

20.1. Import Symbols

Import Predefined Symbols

Symbols can be imported using the command line.

1. Navigate to the symbol pool:

```
MA User name[Fixture]> ChangeDestination Image 2
```

2. Now import the predefined library images:

```
MA User name@ShowData/MediaPools/Symbols> Import Library "*.xml"
```

This command imports the symbol files from MA into the symbols pool.

3. Return to the command line root:

```
MA User name@ShowData/ImagePools/Symbols> ChangeDestination Root
```

For more information, see [ChangeDestination keyword](#).

Using the Symbol Pool

Workflow:

1. Tap **Edit**, then tap an empty pool object or open the swipecy commands and choose **Edit**.
2. From the **Edit Symbols** pop-up, tap **Import**. This opens the **Select Image** pop-up.
3. Select the drive, then select the symbol to import.
4. Tap **Import**.
5. If needed, label the new imported symbol, then close the **Edit Symbols** pop-up.

By default, the grandMA3 software uses the **gma3_library/media/symbols** folder to import and export symbols.

For more information, see [Folder Structure](#).

From the Import/Export Menu

Workflow:

1. Press **Menu**.
2. Tap **Import/Export**.
3. Tap **Import** on the top left corner of the window.
4. Tap **Symbols**.
5. Select the drive, then tap an empty pool object on the right side of the window.
6. Select the symbol to be imported.
7. Tap **Import** at the bottom of the window.



20.2. Delete Symbols

Delete Symbols

Deleting symbols is like deleting any other pool object. The symbol disappears in any appearance where it might be used.

There are three common ways to delete symbols.

Using the Command Line

The important keyword for this is: **Delete**.

This is the syntax for deleting a single symbol:

Delete Image 2.x

For example, if symbol 4 needs to be deleted:

```
MA User name[Fixture]> Delete Image 2.4
```

Deleting a range of symbols using the standard range syntax (Thru, +, and -) is also possible.

For example, if symbols 15 to 20 need to be deleted:

```
MA User name[Fixture]> Delete Image 2.15 Thru Image 2.20
```

Using the Symbol Pool on a Screen

Requirement:

A visible symbol pool on one of the screens.

1. Press **Delete**.
2. Tap the symbol in the pool.

The symbol is deleted.

Using the Swipey Commands

Requirement:

A visible symbol pool on one of the screens.

1. Tap and hold the symbol you wish to delete.
2. Swipe out of the pool object without releasing the screen.



3. Swipe to the **Delete** swipecy and release the screen.

The symbol is deleted.

21. Appearances

Appearances are sets of looks that can be assigned to pool objects, presets, view buttons, or windows.

All appearances are stored in the **Appearance Pool**. This can be created like any other window - learn how in the [Add Windows topic](#).



Example of the Appearance pool

Images can be a big part of an appearance. To use images, they need to be imported into the **Image Pool**. Read about the images in the [Images topic](#).

21.1. Create Appearances

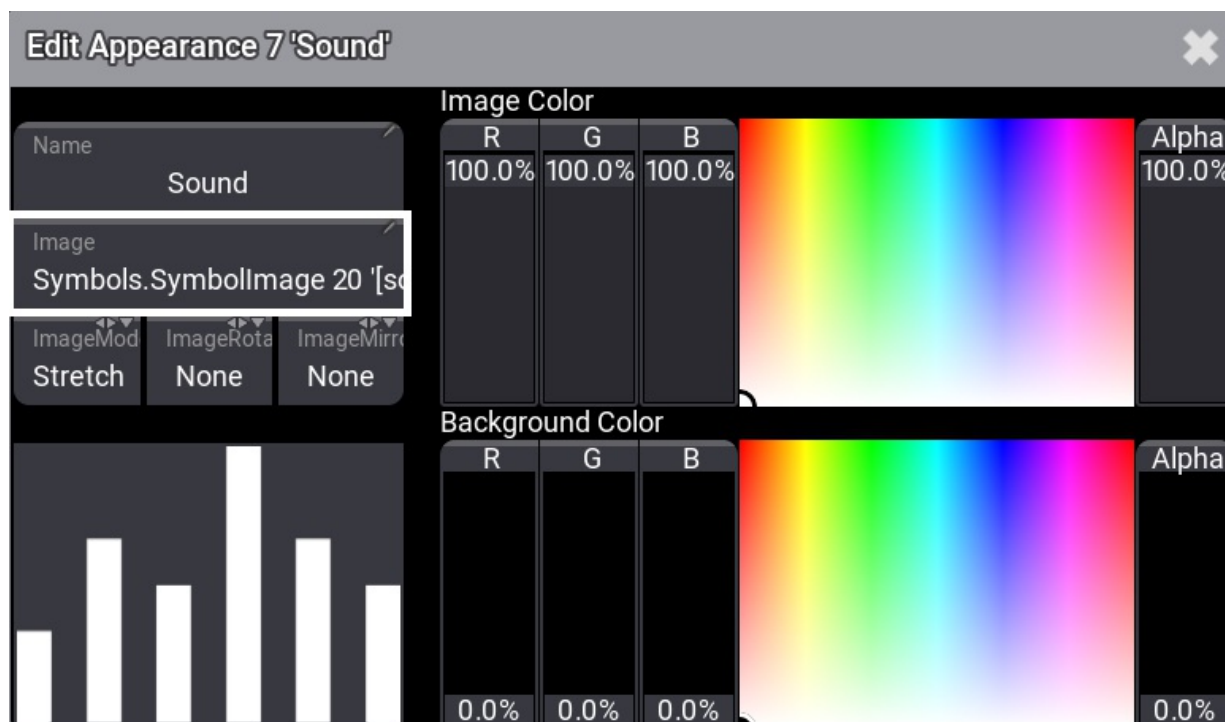
Edit an empty pool object in the **Appearance Pool** to start to create a new appearance.

Edit an appearance pool object using one of these 3 options:

- Press **Edit** and then tap the desired appearance object.
- Open the swipecy commands on the pool object and choose **Edit**.
- Using the command line: **Edit Appearance [ID]**.

If an existing pool object is edited, then the editor opens with the current state of the appearance.

This is the **Appearance Editor**:



There are five input boxes, two sets of faders with a color picker to adjust color, and finally a preview area that displays a preview of the appearance.

Setting a Background Color

The **Background Color** fader and color picker can be used to change the background of the appearance. The default background is transparent (**Alpha** fader at 0%).

Turning up the alpha and using the R (red), G (green), and B (blue) faders make it possible to mix any color in the RGB range. The color picker can also be tapped to select a color.

Giving the Appearance a Name

Many editors and pop-ups have a **Name** input field. A name can be set using the **Label keyword**.

Appearances can be labeled like any **other pool object**.

It is also possible to give the appearance a name in the editor:

1. Tap the name input box.
2. Write a new name using the Edit Name pop-up.



Restriction:

Appearances cannot have a Scribble in the label.

Adding an Image to the Appearance

Appearances can use an image. The image is placed in front of the background but behind a label.

1. Tap the **Image** input box to add an image. This opens the **Select Image pop-up** where all the images are listed—Tap **ImageSource** in the title bar, to select a new source.
2. Choose the desired image source.
3. Tap the desired image in the pop-up.

The images have different modes. Their modes define how the image is adapted to the aspect ratio.

These are the different modes:

- **Stretch**
The image is stretched to fit the appearance area.
- **Bar**
The image is fitted to the appearance area without changing the aspect of the image.
- **Crop**
The image is fitted to fill the entire appearance area. The aspect of the image is kept, but it is chopped.
- **Tile**
The image is tiled and chopped to fit the entire appearance area. The aspect of the image is kept.

Do the following to change the mode:

1. Tap the **Mode** swipe button. This toggles through different modes. Or swipe the button to open the **Select Image Mode pop-up**.
2. Tap the desired mode in the pop-up.

The color of the image can be adjusted using the top set of fader and color picker.



This means that the same image can be used on different appearances but with different color hues and transparency.



Restriction:

Color can only be added to an image. This means that a black image cannot be changed, but a white image can be changed.



21.2. Use Appearances

Appearances can be used in many places.

Almost all pool objects can have an appearance. Many windows can also have an appearance.

The appearance needs to be **created** before it can be assigned. And the object that it is assigned to needs to exist before anything can be assigned to it.


Using Appearances on Objects

Appearances can be assigned to objects using the following syntax:

Assign Appearance appear_number At object_type object_number

This can be used on pool objects.

For example, assigning appearance 1 at group 5 could be done like this in the command line:

```
 User name[Fixture]> Assign Appearance 1 At Group 5
```

This can also be done using the GUI.

The Appearance pool and the Groups pools should be visible on the screens for this.

1. Tap, hold, and slide out the finger on appearance 1.
2. From the swipecy commands, move the finger to the **Assign** button and release.
3. Tap group number 5.

Editing many objects also gives access to an appearance button that can be tapped to select one of the existing appearances.

Using Appearance on Windows

Many windows can have an appearance assigned.

The best way is to open the **settings for the window** and edit the **Appearance** setting in the **Display** tab.



21.3. Delete Appearances

Deleting appearances can be done like **any other pool object**.

If the appearance is used somewhere, then a warning pop-up opens; press **Ok** to delete.

The object that used the appearance is reset to **No Appearance**.

Oopsing the deletion will reassign the appearance.



22. Notes

The Notes feature allows to add useful information to objects. Each object that has Notes available, has a **Note** button in its editor or a Note cell. For example:

- Presets
- Groups
- Macros
- Cues

Notes can be added to pool objects via the label command. For more information on how to label pool objects, see [Label pool objects](#).

Notes can be added via the note keyword, too. For more information, see [Note keyword](#).

Notes are part of the object information. For more information, see [Info window](#).

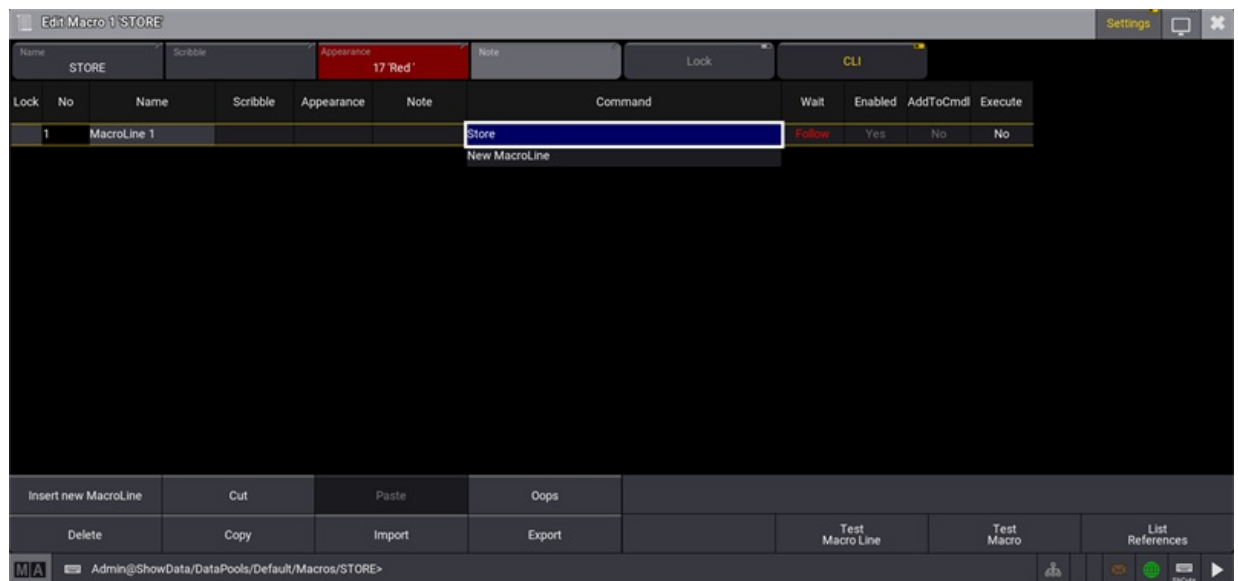
In the following examples, notes are added in different objects.

- **Requirement:** grandMA3 demo show file is open.

Notes in Pool Objects

To add a note to a pool object, for example a macro, follow these steps:

1. Open the Add Window dialog, tap the **Data Pools** tab and then tap **Macros**.
The Macro Pool window opens.
2. Press **Edit** and then tap the first macro in the pool, **STORE**. The editor opens:




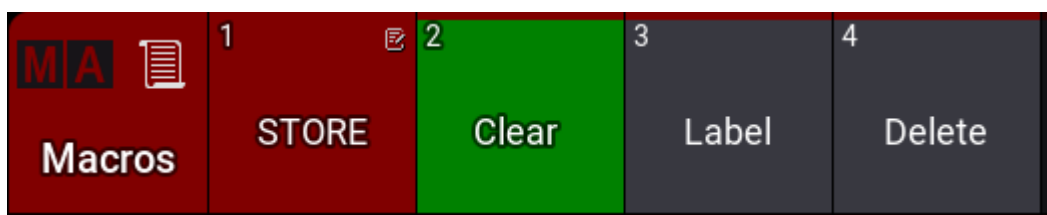
Text input editor.



Make sure **Settings** is enabled in the top right corner in the macro editor.

3. Tap **Note** in the Settings bar at the top in the macro editor , to open the Text input editor.
4. Write "Stores everything" in the text field and then tap **Save** in the top right corner. The Text input editor closes and the text is saved.
5. Tap **X**, to close the macro editor.

The pool object displays a note symbol  in the top right corner:

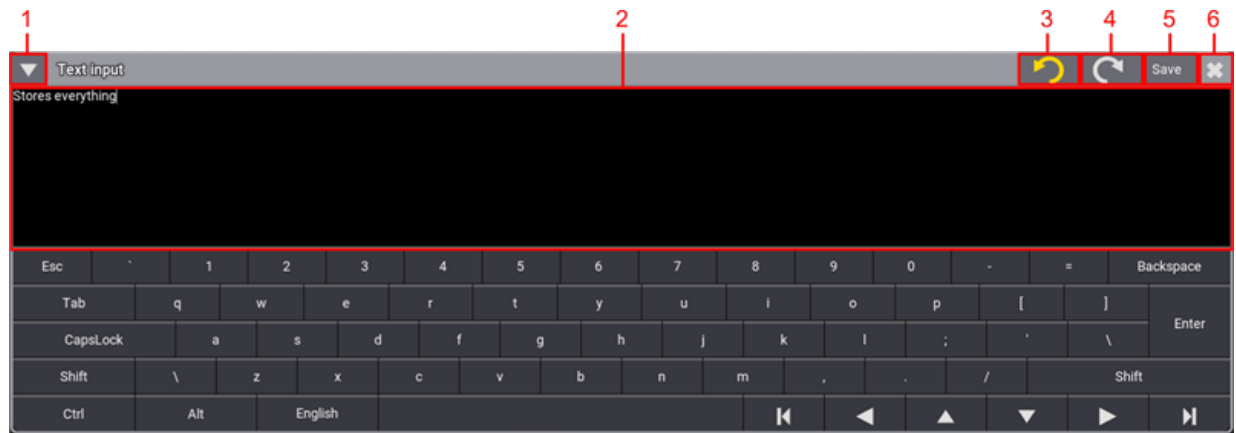


A note symbol is added to a pool.

To edit the note in a Note cell, for example a macro line, follow these steps:

1. Type **Edit Macro 1** into the command line and press **Please** . The editor opens.
2. Right-click the **Note** cell of MacroLine 1.
The Text input editor opens.
3. Follow the steps 4 and 5 from above.

Text Input Editor



Caption

To open the text editor, tap a **Note** button or 2 finger edit a **Note** cell in the grid.

The Text input editor has the following features:

1. Shows or hides the virtual keyboard.
2. Text field.
3. Undoes last changes.
4. Redoes last changes.
5. Saves the changes and closes the input dialog. Alternatively press **Please** on the console.
6. Closes the input dialog. A pop-up opens, if changes are unsaved.



Hint:

A note can have up to 8 192 characters.

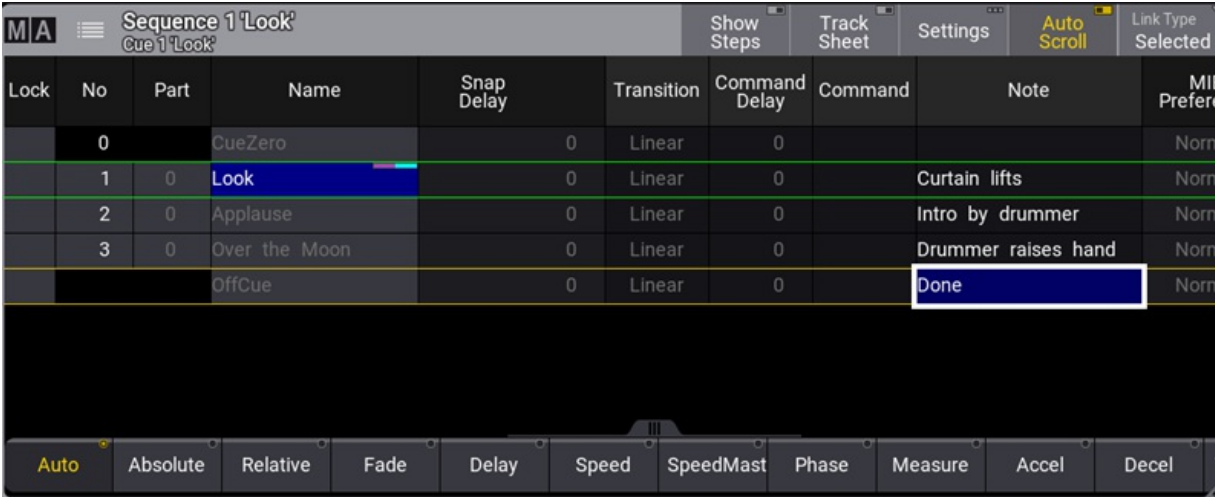
Notes in Cues

Notes inform the operator, for example, about primary actions on stage to trigger the next cue.

- For information about cues, see [Cues and Sequences](#).

To add a note to a cue:

1. Open the Add window dialogue, tap **Common** and then **Sequence Sheet**.
2. Tap into the Note column in a cue and start typing. The Text input editor opens. After finishing typing the note, tap **Save**.
3. The text is displayed in the Note column.



Lock	No	Part	Name	Snap Delay	Transition	Command Delay	Command	Note	MIB Prefere
	0		CueZero	0	Linear	0			Norm
	1	0	Look	0	Linear	0		Curtain lifts	Norm
	2	0	Applause	0	Linear	0		Intro by drummer	Norm
	3	0	Over the Moon	0	Linear	0		Drummer raises hand	Norm
			OffCue	0	Linear	0		Done	Norm

Sequence Sheet with different Notes added.

Multi-Line Editor

- The text editor in notes is a multi-line editor. This means, multiple lines can be added by pressing **Shift** + **Please** when the text editor is open.
- In the Mask-tab in the sequence sheet settings, tap **Line Height** to adjust the line height. When set to **Auto**, the row automatically adjusts its height based on the number of lines. For more information about the sequence sheet settings, see [Sequence Sheet](#).



Lock	No	Part	Name	Morph	Transition	Command Delay	Command	Note	MIB Preference	MIB Mode	MIB Target	MIB MultiStep	MIB Fade	MIB Delay
	0		CueZero		Linear	0			Normal					
	1	0	Look		Linear	0		Note 1 Note 2 Note 3 Note 4 Note 5 Note 6	Normal	<None>		Running	Default	Default
			OffCue		Linear	0			Normal					

The height of the rows is adjusted dynamically when Line Height is set to Auto.

Show Notes

Another way of displaying notes in the Sequence Sheet or Content Sheet is possible via **Show Notes** -tab. Show Notes can be enabled/disabled in the dedicated window settings. For more information about window settings, see **Window Settings**.

To enable Show Notes in the Sequence Sheet:

1. Tap **MA** in the Sequence window. The Sequence Sheet setting opens.
2. Tap the **Mask** -tab and then enable **Show Notes**. Close the window.
3. A text field with information about notes is added to the Sequence Sheet as shown below:

Lock	No	Part	Name	Snap Delay	Transition	Command Delay	Command	Note	MIB Preference
	0		CueZero	0	Linear	0			Normal
	1	0	Look	0	Linear	0		Curtain Lifts	Normal
	2	0	Cue 2	0	Linear	0		Intro by drumme	Normal
	3	0	Cue 3	0	Linear	0		Drummer raises	Normal
			OffCue	0	Linear	0		Done	Normal

Note for Cue 1 : Look

Curtain Lifts

Show Notes enabled in the Sequence Sheet



Hint:

When Show Notes is enabled, Show Recipes is disabled.

Important features of Show Notes:

- The text field can be used, for example, to show notes of a currently active cue, a preloaded cue, or a currently selected cue.
- The text field allows editing notes live.
- The text field shows information about the currently selected note. For example, "Note for Cue 1: Look" in the image above.



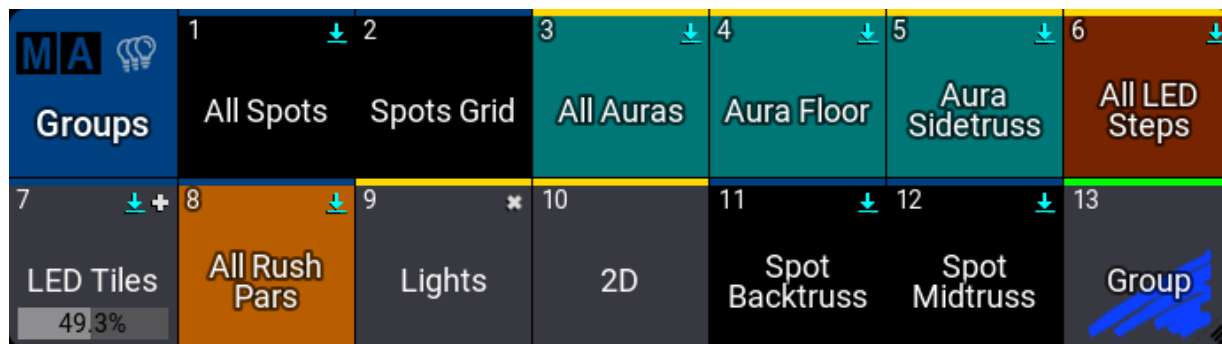
23. Groups

Groups contain a selection of fixtures. This includes the order and grid position of the selection.

This can be used as a programming tool, where it is a fast way to select the fixtures. Grid information is stored in the groups - read more about grids in the [Selection Grid Window topic](#).

Groups can also be used as masters. There are four different types of group masters. Read more in the [Group Master topic](#).

Groups are organized in a pool - read more about pools in the [Pool Windows topic](#).



Groups pool example

Information about the groups is not stored in cues or presets. Only the values applied to the fixtures are stored in cues and presets.

The next topics explore the creation of groups.

23.1. Create Groups

Groups are created by storing the group while having a selection of fixtures. Groups are stored in the **Group Pool**.

The groups store the selection of fixtures, the grid information, and the fixtures' selection order (this is also a grid).

Grid information is 3D position information indicating the position relationship between the fixtures. It is not the location on the 3D stage. For more information, see [Selection Grid topic](#).

The order and grid information is important for ranged value input or when creating [Phasers](#).



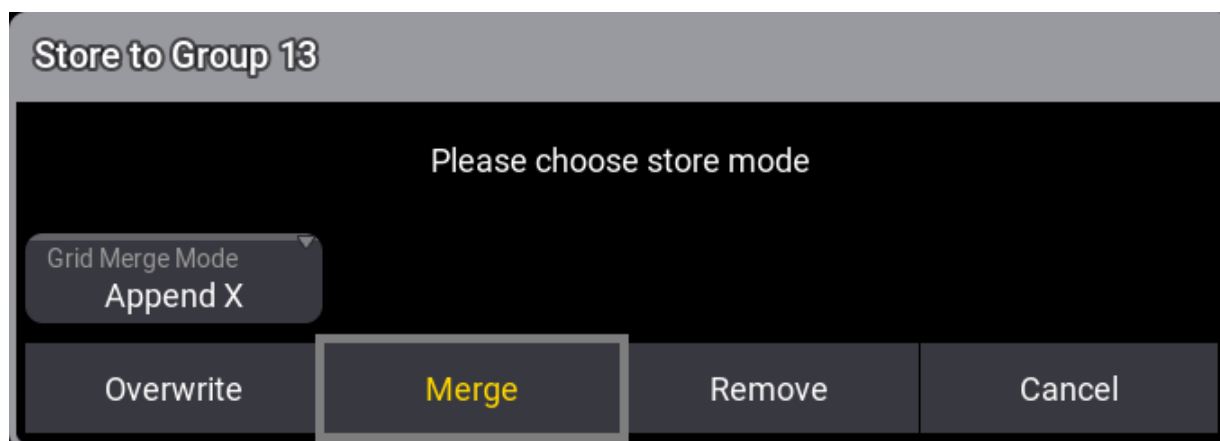
Important:

Groups do not store values! - Only the fixture selection, order, and grid.

These are the steps needed to create a new group:

1. Please make sure the correct fixtures are selected, and that they are in the correct order.
2. Store the group using a syntax like this: **Store Group [my_group_number]**. The two needed keywords are **Store** and **Group**.

When storing to an existing group, the store mode pop-up opens:



Store mode pop-up.

When **Grid Merge Mode** is set to **Append X** (this is the default setting), the selected fixtures will be added to the next available X coordinate.

Read the [Selection Grid topic](#) for information about positioning the fixtures in a grid before storing the group.

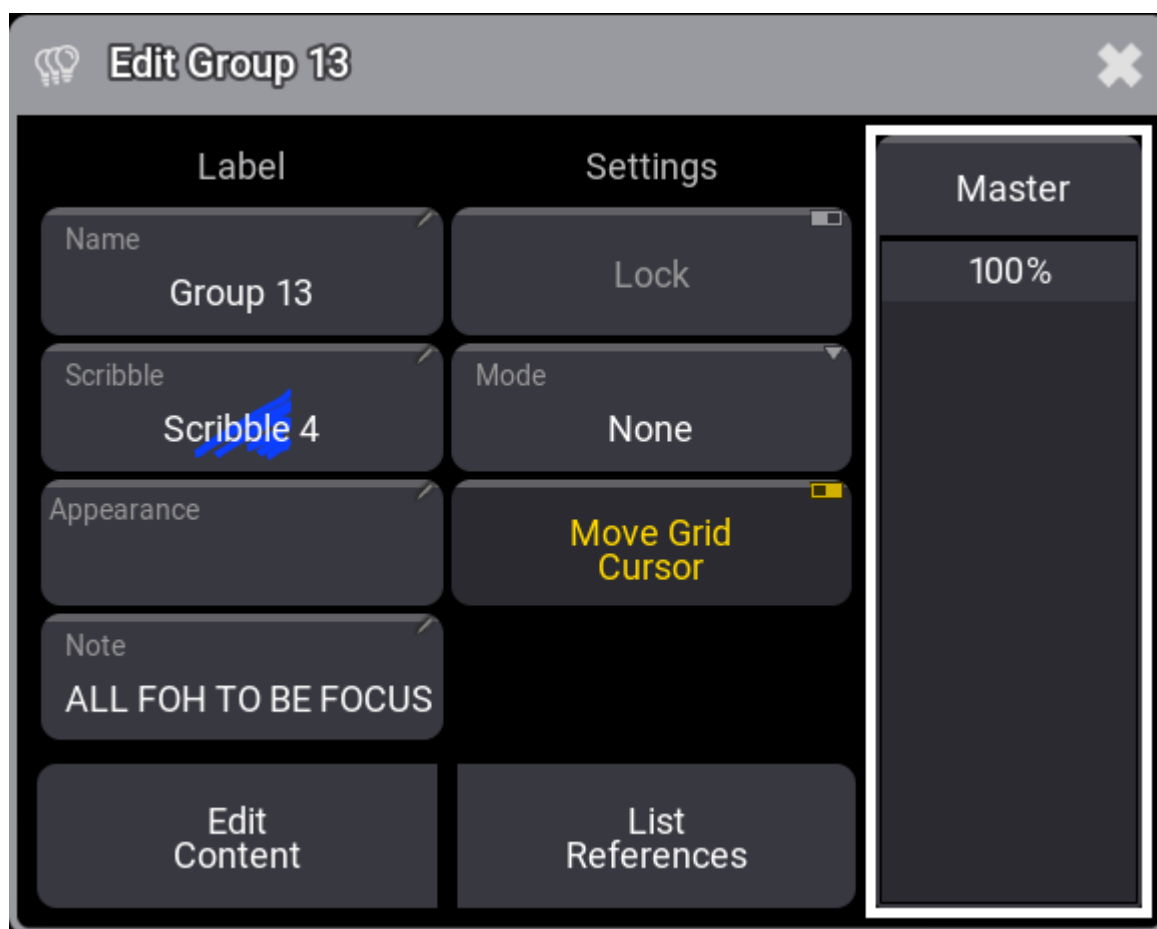
Existing groups can be edited using the [Edit Group pop-up](#).

It is not required, but it is a good idea to label the group. It can be done using any methods described in [Label pool objects](#), or the Edit Group pop-up.

23.2. Edit Groups

To edit a group:

- The **EditSetting keyword** or
- Press **Edit** twice and then tap the at the group in the group pool.



Edit group pop-up.

The editor has two columns, **Label** and **Settings**.

There is a **Master** fader on the right. This is used when the **Mode** is one of the four group master types. Read more in the [Group Masters](#) topic.

Here is a description of the left buttons under the **Label** column:

Name:

The group name. To change a group name, use the **Label** command. For more information, see [Label keyword](#).

Scribble and Appearance:

Can be assigned using these two buttons. For more information, see [Scribble](#) and [Appearances](#).



Note:

For more information, see [Notes](#).

Lock:

When toggled on, **Name**, **Scribble**, **Appearance**, **Mode**, and **Move Grid Cursor** are disabled.

Mode:

Used to select one of the four group master modes. For more information, see [group masters](#).

Move Grid Cursor:

Off: The cursor in the **selection grid window** stays at its grid position when the group is called in the programmer.

On: The cursor moves in the **selection grid window** to the next X grid position when the group is called in the programmer.

Tap **List References** to open an **Info pop-up** that shows a list of the group's references and dependencies.

Tap **Edit Content** to open the **Edit** mode (Fixture sheet frame turns green). Now it is possible to change the fixture selection. Press the **Off** key, then select the fixtures you want to remove from the group. From the swipecy commands, move your finger to the store command. A pop-up menu will ask you to Overwrite, Merge, Remove, or cancel. Use **Esc** to exit the **Edit** mode without changes.



23.3. Delete Groups

Groups can be deleted like **any other pool object**.




It does not have any effect on programmed cues or any other element. The only exception is if the group is assigned to an executor directly and used as the base for a **Group Master**, then the executor will be empty after deleting the source group. Read more in the **Group Master topic**.

23.4. Group Masters

Groups can be masters of the fixtures in the group.

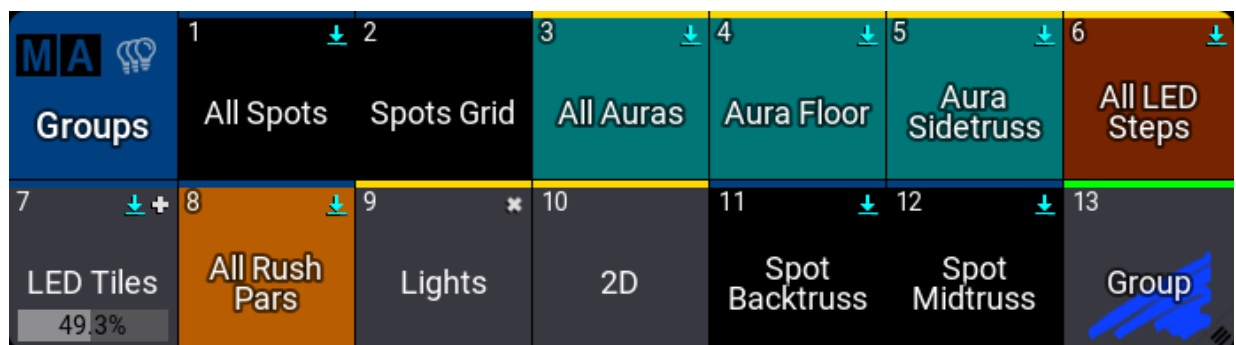
If the value is adjusted often, it can be advantageous to assign the group to an executor. Learn how in the [Assign Object to an Executor topic](#).

There are four different kinds of group masters:

- **Positive** - This is an HTP group master, which is indicated by  in the upper right corner of the pool object. The output of the fixtures is reduced depending on the value of the master of the group.
- **Negative** - This is a LoTP group master, which is indicated by  icon in the upper right corner of the pool object. The output of the fixtures is reduced depending on the values of the master of the group.
- **Scaling** - This scales the intensity output. Scaling is indicated by  icon in the upper right corner of the pool object. For example, if the fixture is at 50% and the group master is at 50%, then the output is 25%.
- **Additive** - This master does not limit output but adds output as HTP merges with values from the programmer and playbacks. It is indicated by a small circle icon in the upper right corner of the pool object.

Difference Between Positive, Negative, and Scaling

The difference between the positive and negative masters is relevant when several groups contain the same or overlapping fixtures. A negative master has priority over a positive. For example, if two groups contain a fixture with 100% output, and if one group is a positive master at 80% and the other group is a negative master at 60%, then the output is 60% (limited by the negative master). If the negative master is turned up, the output stops at 80% when the positive master becomes valid. If you then switch both groups to Scaling, the output will be 48% since the fixture output of 100% is multiplied by 60% and by 80% of the second group master.



Group pool with masters

Groups with a mode assigned have a horizontal bar that displays the master level if it is relevant.

The master mode can be selected using the **Edit Group pop-up**. Read more about the pop-up in the [Edit Groups topic](#).



The mode can also be assigned using the **Set keyword**.

Example

Using the command line to set the master mode of group 4 to positive:

```
MA User name[Fixture]> Set Group 4 "Mode" "Positive"
```

The mode type is capital sensitive. This means that writing "positive" fails, but "Positive" works.



Hint:

If the mode is not specified, a pop-up appears, allowing you to choose a mode from a list.

Delete a Group Master

Deleting a group assigned to an executor does not delete the group from the pool.

If a group is deleted from the group pool, it is deleted from the show.



Restriction:

If the group is limited by the master, when it is deleted, then the fixtures stay limited. This means that a new group needs to be created with the fixtures to remove the limitation!

If the group is removed from an executor, then it still exists in the group pool, and any master level set using the executor is still valid in the pool.

24. Presets

A preset can hold information about attribute and timing values for a selection of fixtures and may be referenced and re-used in cues or in the programmer.

The principle of presets is to store a labeled reference in a cue, rather than the actual value itself. Or use presets for busking, where presets are called into the programmer or played back on the fly.

Updating the preset means that the cues do not need to be updated since they reference the preset and not the actual preset content.

Presets are marked with a cyan marker. This marker is visible on cues that use preset and on the values themselves, for instance, in the tracking sheet or the fixture sheet. Read more in the [Marker topic](#).



Hint:

Using presets when programming facilitates the work, especially when working with a show which is used in different locations and for various customers.

Preset Pools

Presets are stored in pools. There are pools for each feature group. These pools have a default input filter that only allows the values for that feature group to be stored in the preset pool. There are also five preset pools called "All 1" to "All 5". These do not have a default input filter and can be used to store any attribute values. Learn more about the pools in the [Preset Pool topic](#).

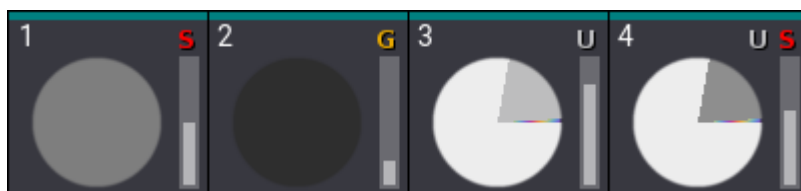
Preset Modes

Preset modes are used when the preset is stored (or updated) and when the preset is called.

There are three different preset modes Selective, Global, and Universal.

Each preset got letters showing the preset mode.

Example:



Preset 4 has all three modes stored but only shows U and S.

The three modes are:

- **Selective (S):**
The data will be added as selective data for each fixture that has active data in the programmer.
- **Global (G):**
The data will be added mainly as global data. If there are several fixtures of the same fixture type but with different values, then the global data will be determined by average, and selective data will be added for the other fixtures which have divergent data.
- **Universal (U):**
Data will be added as global data, and the preset mode will be set to Universal.

When a preset is stored or updated, then there are two more options:

- **Auto:**
When updating or storing into an existing preset, the preset mode of the preset will be respected.
In the case of global preset mode, selective data will be added to the preset when at least one fixture that can use the preset is active with new values.
When creating a new preset Auto mode will take the mode defined by the pool and use the rules for each mode.
- **Force Global:**
Data will be added as global data, and untouched existing selective data will be discarded within the preset for the fixtures of the same fixture type.
Force Global will discard the selective data when updating a preset or when storing with the merge option into an existing preset.

Each preset can store all three modes or some combinations of modes.

Each preset pool has a setting for a default mode. This mode is indicated by one of the letters in the upper right corner of the pool title field.



Position preset pool title field with Selective as the default mode for the pool

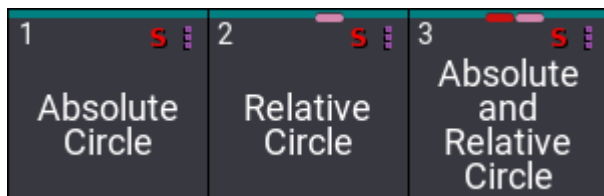
This default can be used when storing the preset or a different mode can be used.

Learn more in the [Create New Presets topic](#) and the [Use Preset topic](#).

Absolute and Relative Values

Presets can store absolute and/or relative values. Relative values are often used with multistep presets but it is not limited to this use.

Example:



Preset with absolute, relative, and both types of values

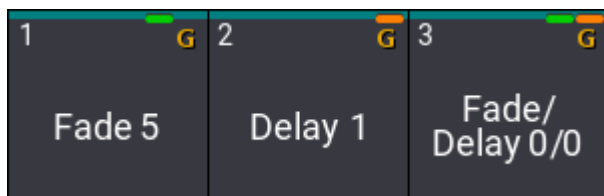
Presets with only absolute values do not have a marker. The absolute marker is a square dark red marker with rounded corners. Relative values are indicated by a dark pink square with rounded corners. Presets can contain both absolute and relative values - they will have both markers in the pool object.

Learn more about the different programmer layers in the [What is the Programmer topic](#). Learn more about the markers in the [Markers topic](#).

Timing Values

Timing values can be stored in presets. The timing layers are Fade and Delay.

Example:



Preset with fade, delay, and both types of values

The Fade marker is green. The Delay marker is orange.

Timing values can be combined with absolute and relative values.

Default Action

A preset pool has a default action for the pool objects. This action is performed when the preset is tapped in the pool without a (relevant) keyword in the command line.

A small icon in the upper right corner of the pool title object indicates the selected default action.

The action is set in the [Preset Pool Settings](#). The function is described in the [Window Settings topic](#). Follow the links to learn more.

Feature Group Indicator Bar

This bar is visible when a preset has attributes of two or more feature groups stored.

The bar is always visible on All presets.



Feature Group Indicator Bar at the bottom of the preset

The feature group indicator bar has a square for each feature group, displayed in white, gray, or red color.

- **White:**
Indicates that values of attributes of the corresponding feature group are stored in the preset.
- **Gray:**
Indicates, that no values of attributes of this feature group are stored in the preset.
- **Red:**
Indicates that at least one attribute of this feature group is not accessible at the moment. For example, when having a world with a limited set of attributes active.
The square turns also red when the currently selected world does not contain fixtures that can make use of the attribute data in the preset.

The order of the squares is similar to the order of feature groups in the encoder bar, from left to right:

- Dimmer
- Position
- Gobo
- Color
- Beam
- Focus
- Control
- Shaper
- Video

If a show file contains additional feature groups, more squares will be added automatically to the bar.

The bar can also be visible in [Worlds and Filters](#).

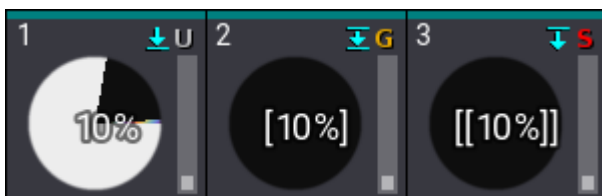
Embedded Presets

A preset can contain a link to a different preset. It is the same as cues storing a reference to a preset - just for other presets. This can be useful for instance when a show is built using a set of general presets and other presets are created using these as building blocks.

Example:

In this example, there is a universal dimmer preset that has values for the universal fixture. This is used to create a new global preset for fixture types. This is again used to create a selective preset for only some fixtures. If the

fixture type change, then the global preset needs to be updated. The selective preset links to the global preset values and do not need to be updated. The universal fixture is unaffected by the fixture change.



Preset 1 is used in an embedded preset. Preset 2 contains embedded data and is also used in a different embedded preset. Preset 3 contains embedded data.

The "source" preset has a downward pointing arrow that points to a line. This indicates that this preset is referenced by other presets (or cues). The preset with embedded data has an icon with an arrow and a line above to indicate that this preset uses referenced data. Preset with embedded data that is also referenced has the arrow and a line above and below the arrow.

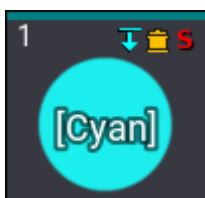
If a new name is not defined then the name is also referenced (referenced names are in square brackets). This means that changing the name of the source preset also updates the name of the new preset. Learn more in the [Create New Presets](#) topic.

Recipe Presets

Presets can have recipe information. A recipe is one or more lines with information about a group, preset, MAticks, individual fade, delay, speed, and phase values.

This information can be used to "cook" values into the preset or programmer. If the source information changes then the preset can be cooked again to reflect the changes. For instance, if a group is used in a recipe and it changes after the initial cooking, then the preset can be cooked again and it will now reflect the changes to the group. Learn more in the [Recipe Presets](#) topic.

Example:



Preset with recipe information

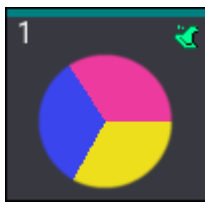
A small pot icon indicates a preset with recipes.

MAGic Presets

MAGic presets are presets where value points in a range are defined. This range can then be applied to a dynamic selection of fixtures. For instance, two fixtures are stored in a MAGic preset with dimmer values of 0% and 100%. MAGic presets should be stored as a selective preset, but that does not mean that only the selected fixtures can use it - it only means that they contain values for a selection of fixtures. The MAGic preset uses the values to define the points in a range. A different fixture selection can be made and the range of values between the points is assigned to the fixtures when the MAGic preset is called into the programmer. These values are calculated based on the points and then taken into the programmer as hard values. There is no reference back to the MAGic preset.

There can be up to five defined points in the range (on each axis in the selection grid).

MAGic presets have a small wizard hat icon in the preset pool object

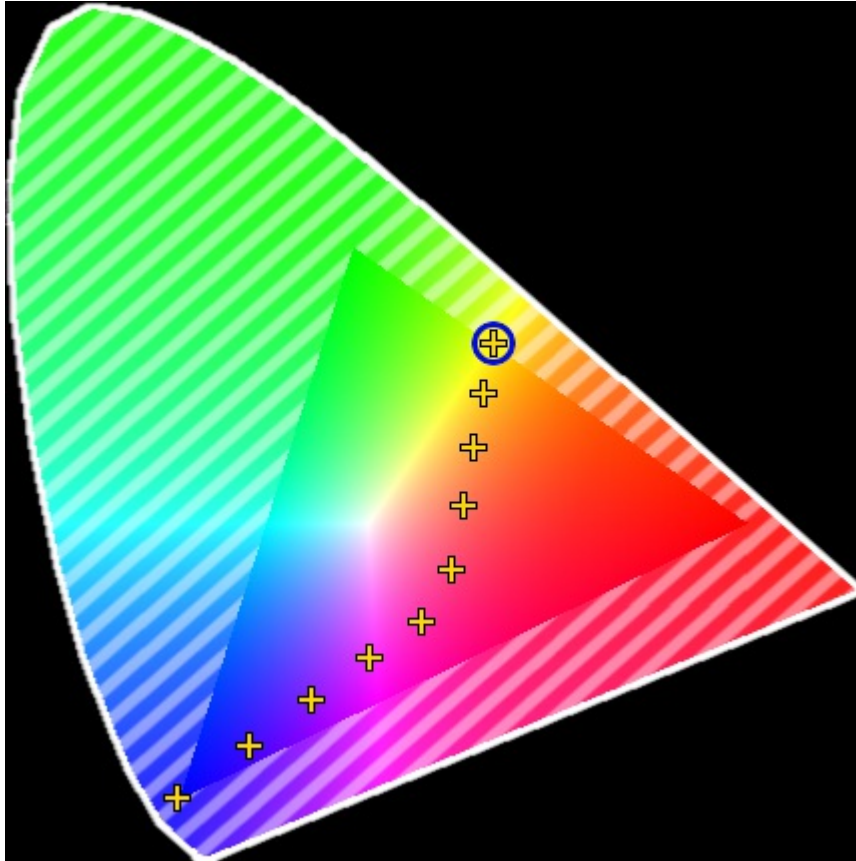


Position preset with MAGic information

Example:

A path needs to be defined to control a color range from blue to yellow. The wish is to have the range go through the red area in a CIE color picker - instead of going through white. Three fixtures are needed to define the range.

The first fixture is blue, the second is magenta, and the third is yellow. This is then stored as a MAGic preset. Now this range through three points can be used by multiple fixtures to create the desired path.



Result of a three-point MAGIC color preset applied to multiple fixtures

Learn the details on how to create this MAGIC preset in the [Create New Presets](#) topic.

Multistep Presets

Presets can contain values in multiple steps (Phasers). Presets with only one step usually have the values stored in step one. If a preset has multiple steps then they are called multistep or Phaser presets.

Example:



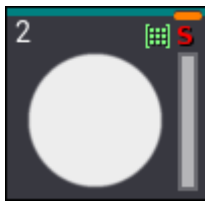
Multistep preset

They have a three-dot icon to indicate multistep values.

MATricks Presets

MATricks information can be stored in presets. Having MATricks information in the programmer and storing a preset will add the MATricks information into the preset (if the **Store Settings** allow it).

Example:



Preset with MATricks information

The MATricks icon is a green icon with nine dots in a grid.

Filtered Presets

The entire preset pool has a default input filter - except the All preset pools. The default filter is on feature group preset pools. They automatically filter the attributes in the feature group.

The input filter can be changed to a custom filter for any preset pool, including the All presets. If the input filter is different than the default, then there is an input filter icon on the preset pool title field.

Besides an entire preset pool having an input filter, it is also possible to have an input filter on individual presets. Learn more about input filters in the **Edit Presets** topic.

Example:



Preset with input filter

The input filter icon is a gray filter icon with a small right-pointed arrow.

Extract Preset Data

The values stored in presets can be extracted from the preset. This will take the values stored in the preset and put them into the programmer. There will not be a link to a preset when the values are extracted.

This can be done using the **Extract keyword**.

Different sources can be specified with the extraction.

If there is a selection of fixtures with references to presets in the programmer, then the entire selection can be extracted. The command for this would be: **Extract Selection**

Specific fixtures can be specified instead of extracting the entire selection. It can be single fixtures or a selection of fixtures: **Extract Fixture [Fixtures_Numbers or "Fixture_Names"]**

Specific presets can also be extracted into the current selection of fixtures: **Extract Preset [FeatureGroup_Number or "FeatureGroup_Name"].[Preset_Number or "Preset_Name"]**

All presets from a cue can also be extracted to a selection of fixtures: **Extract Cue [Cue_Number or "Cue_Name"]**



Hint:

When extracting by specifying a fixture, selection, cue, feature group, or attribute, presets must be active in the programmer for the desired fixtures.

When extracting embedded presets or phaser presets which have presets integrated into their steps, extract will call directly the values of the source presets.

Using the **/Single** option together with extract makes it possible to extract one level down in the hierarchy of the presets.

Example

Create 2 color presets ("red" and "blue" presets) that are embedded into a different color preset ("odd/even red/blue"). The second color preset ("odd/even red/blue") is embedded into All preset 21.1 ("cool look").

Select some fixtures and apply the All preset ("cool look") to them.

Executing **Extract Selection /Single** calls the second color preset ("odd/even red/blue") into the programmer.

Executing **Extract Selection /Single** again calls the first color presets ("red" and "blue" presets) into the programmer.

When executing **Extract Selection /Single** a third time, the result is the same as using **Extract Selection** in the first step: The hard red and blue values without a preset link.

24.1. Preset Pools



Important:

If pools are new to you, please read the [Pool Windows topics](#) first.

There is a preset pool for each **Feature Group** in the show. A show with no custom feature groups will have nine preset pools named after the default feature groups. For instance, Dimmer. These pools can have a feature group filter besides the standard input filter (read more below).



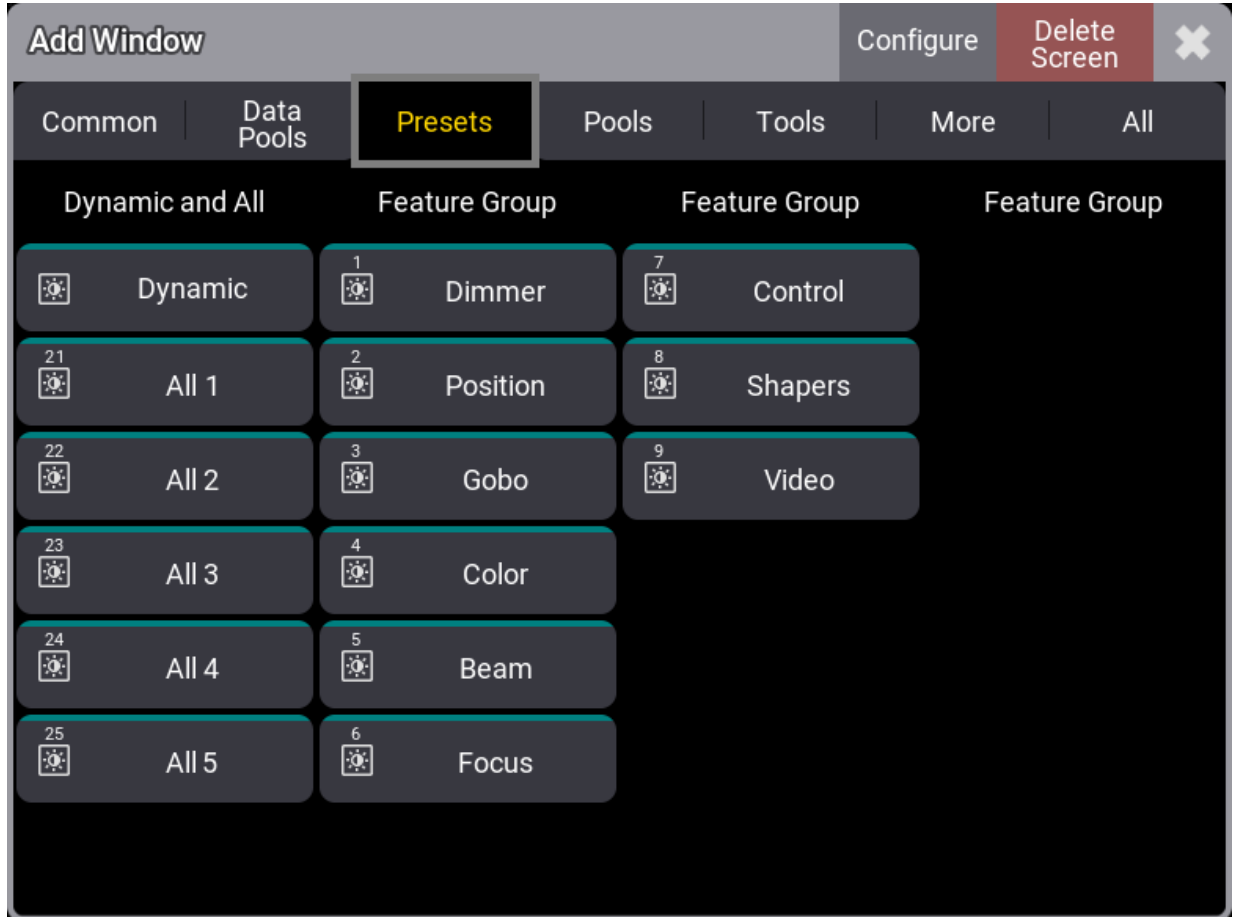
Hint:

Create custom preset pools using Feature Groups. For more information, see [Feature Group](#).

Besides the feature group preset pools, there are five "All" preset pools. These do not filter based on feature groups. The "All" preset pools can be labeled to match the needs. For instance, one of the pools could be used for storing multistep phaser presets or different parts of the show.

There is a unique preset pool called "Dynamic". This is not a preset pool in itself. It automatically changes between the feature group preset pools based on the selected feature group in the [Feature Group Control Bar](#).

The preset pools are created like any other window in the user-defined areas using the [Add window pop-up](#). They are all under the **Presets** tab:

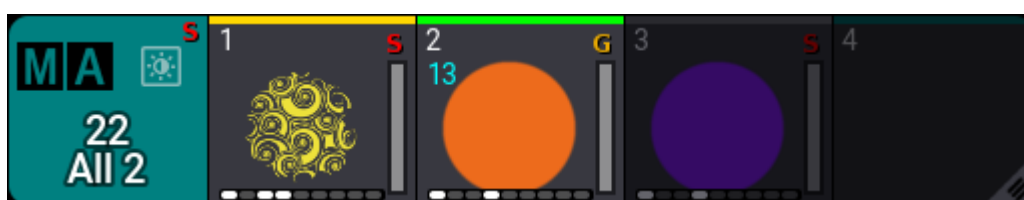


Open the Presets

The preset pools look and behave like many other pools, but small differences exist. Read more about pools in general in the [Pool Windows topics](#).

The default preset mode for the preset pool is indicated by a letter in the upper right corner of the pool title field. Read more about the preset modes in the [Preset topic](#) and below for information about setting the default.

There is a colored indicator bar at the top of each pool object. If nothing is selected, it is colored like the pool color. The pool object is dimmed if there is a selection, but none of the currently selected fixtures can use a preset. If a preset can be used by all the fixtures currently selected, then the colored bar is green. It is yellow if the preset is only valid for some of the selected fixtures. These are the default colors. They can be edited in the pool settings (see below).



Presets in the All 2 pool

The light blue number below the pool number is the number of fixtures currently using the preset. See preset 2 in the example image above.

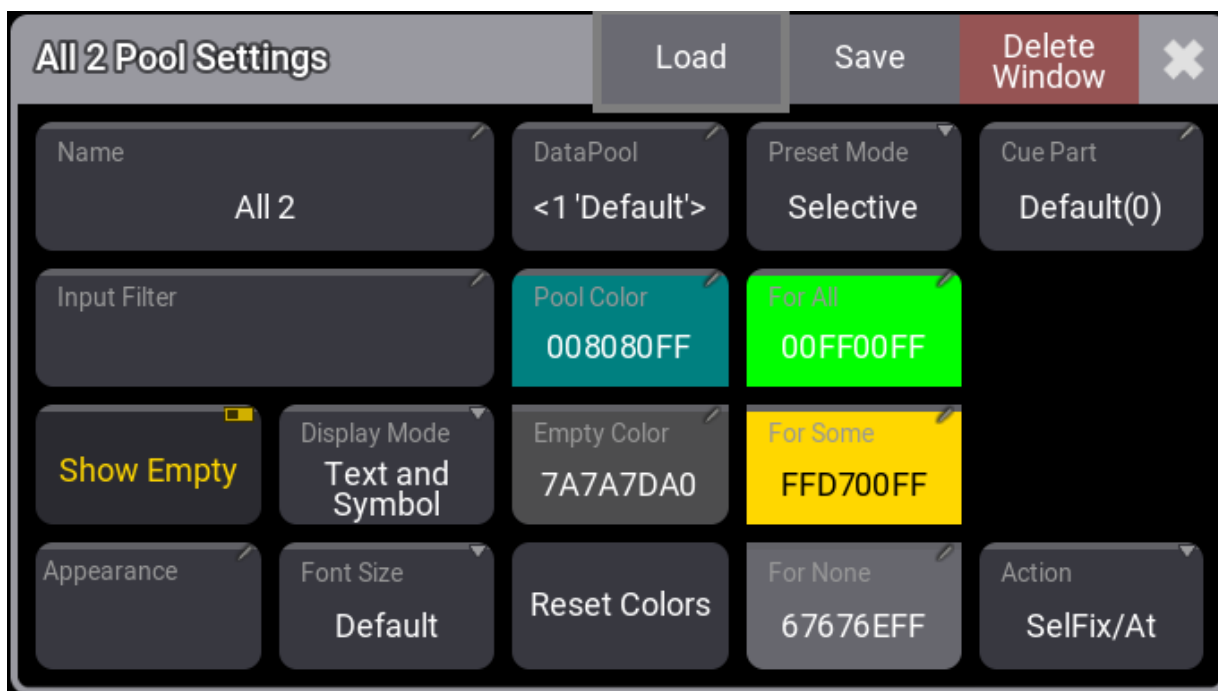
Preset Pool Settings

The general [Window Settings topic](#) describes many of the preset pool settings.

There are, however, some settings that are special for the preset pools.

Enter the settings by tapping the MA logo in the pool title field.

This is an example of the preset pool settings.



Settings for the All 2 preset pool window

These are the special settings for presets:

- **Name:**
The five All preset pools can be named. The other pools are named from the feature group.
- **Input Filter:**
This can be used to select an input filter for the entire pool. [Worlds and Filters](#) can be used. An input filter blocks some elements. The blocked elements cannot be stored in the pool. Tap this to open an **Assignment Editor**. Here filters or worlds can be selected, or it can be set to **Empty**.



- **Preset Mode:**
This sets the default preset mode when storing into the entire pool. A different mode can always be specified when storing. Learn more about preset modes in the [Presets topic](#).
- **Cue Part:**
By default, presets call their values into programmer part 0. Use this setting to specify a different programmer part for presets for this pool. This setting can also be changed for individual presets in the pool. Learn more about programmer parts in the [What is the Programmer topic](#).
- **Display Mode:**
This is used to define what the pool object displays. The pool objects can display the **Text** (name) of the preset or a graphic **Symbol** representing the stored content. It can also show both **Text and Symbol**.

The settings described above (except the preset mode) can also be set for individual presets. The individual preset settings have a higher priority than the preset pool setting.

24.2. Create New Presets

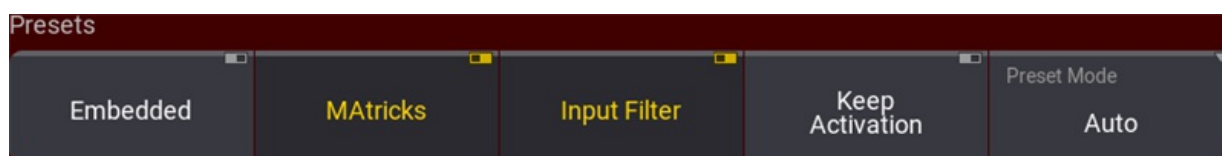
Please read the general [Preset](#) topic and the [Preset Pool](#) topic before this topic to get an understanding of the presets and how they are organized.

Creating and using presets requires some fixtures patched in a show file or the universal fixture in the patch for general universal presets.

For more information on patching fixtures, see [Add Fixtures to the Show](#).

When presets are stored, some defaults are used unless other choices are actively made.

Some of these choices are accessible in the **Store Options** pop-up. This is opened by keeping **Store** pressed for approximately 2 seconds. The pop-up has a section regarding the presets:



The preset section in Store Options pop-up



Hint:

Changing a setting here without saving this as the new preference only uses the changed setting for the next store action. Learn more about it in the [Store Options and Preferences](#) topic.

The preset mode is one of the options. The preset pool has a default setting, but a different mode can be chosen by tapping **Preset Mode** until it has the desired mode. Learn about the presets modes in the [Presets](#) topic and the [Preset Pools](#) topic.

Keep Activation means that when the preset is stored, the newly created preset is active in the programmer, ready to be stored in, for instance, a cue. Turning this setting Off stores the preset but leaves the values as inactive values in the programmer.

MAtricks allows storing MAtricks settings in the preset.

Read about [Embedded](#) and [Input Filter](#) below.

Input Filter

The **Input Filter** setting in the store options defines if the input filtering in preset pools are used. If this option is active, then it is only possible to store attribute values inside the preset pools if it is allowed by the filter. If the input filter setting is empty, a Feature Group filter is applied. This means that only attributes of the corresponding feature group can be stored in the preset pool. For instance, pan attributes can be stored in the Position preset pool but not in the Color preset pool.



The All preset pools do not have any feature group filtering. Any attribute can be stored, regardless of the Input Filter setting.

Custom filters and worlds can be assigned to the preset pool, or individual presets using the preset settings or individual preset options. Filters and worlds can also be assigned using the following syntax: **Assign Filter [filter_ID] At Preset [preset_ID]**. Substitute filter with world if a world needs to be assigned as an input filter.

Learn more about filters and worlds in the **Worlds and Filters** section. Learn about the preset pool settings in the **Preset Pools** topic.

Simple Static Preset

Storing a preset with simple static values is very easy:

1. Select some fixtures.
2. Give some of the fixture attributes a value in the programmer.
3. Store this in the relevant empty pool object.

This is the general workflow.

The programmer values are stored in the preset if they are allowed to pass the input filter (read above).

The best way to work with presets is to have the desired preset pool available on a touch screen. Then the pool can be tapped to store and call the presets.

It is considered a static preset if it only contains values in **step** one. This means that the values are not changing. Read the **Phaser** topic to learn what steps are.

Global Preset Data

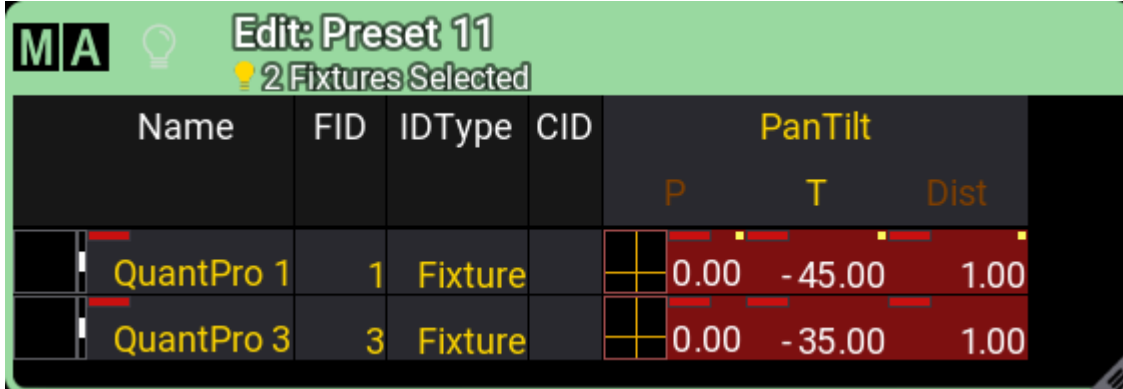
The global data is stored on real fixtures within the patch.

When storing new global presets, the global data will be created within the new preset automatically. When all data for the fixture type is the same, the first fixture from the stored data will be used to hold the global data within the preset.

Each fixture type gets its own global data value so that the global value can be adjusted per fixture type.

When adding selective data to the preset for one of the fixtures that hold the global data, or when deleting the fixture from the show, the global data will be moved to the first patched fixture for the fixture type. If the global data was held before by the first fixture, the data will be moved to the next patched fixture of this fixture type.

When editing a preset with global data, the fixtures with the global data in the programmer will display a yellow square marker in the top right corner of the attribute cell within the fixture sheet.



Name	FID	IDType	CID	PanTilt		
				P	T	Dist
QuantPro 1	1	Fixture		0.00	-45.00	1.00
QuantPro 3	3	Fixture		0.00	-35.00	1.00

Example editing a preset with fixture 1 holding global data and fixture 3 holding selective data.



Important:

When converting old show files from grandMA3 v1.5 or prior, to grandMA3 v1.6 or later, the data from the global fixture type object will be migrated to the first patched fixture of the same fixture type that is not already holding selective data.

The global value is determined by the average across all values of attributes within the same activation group:

$$\text{Average value} = \frac{\text{Value of fixture 1} + \text{Value of fixture 2} + \text{Value of fixture 3} + \dots + \text{value of fixture n}}{\text{Number of used fixtures}}$$

The value closest to the calculated average value will be the global value. The fixture with this value will then hold the global value. In the case of color, all color attributes are handled together when choosing the global value.

Furthermore, the method to determine the global data across several attributes is now considering all attributes of the same activation group. This will result in taking all attributes of the same activation group of one fixture and not taking the different attributes from different fixtures.

Universal Preset Data

Storing a universal preset using a real patched fixture will store global data for active fixture types when the preset was stored.

Universal presets are more versatile and can use the data from a real fixture with global values when the **Universal Fixture** has no data inside the preset.

This has the added benefit of creating universal presets using a real fixture from the stage if the universal fixture has these attributes.

When calling a universal preset, the software first uses global fixture type data and then uses the data from the universal fixture. When no data exist for the universal fixture, the first fixture with global data is used instead.



Hint:

When calling universal color, it will be transformed through the color engine. If for instance, a 7-color LED is used to create a color and call that data universal, a similar color will be called on an RGBA fixture.

Embedded Preset

The concept of embedded preset is explained in the [Presets](#) topic.

This is the workflow:

1. Select the desired fixtures.
2. Recall an existing preset so the preset is in the programmer.
3. Store a new preset with the Embedded store option active.

A recipe preset could be an alternative to an embedded preset. Learn more about them in the [Recipe Presets](#) topic.

Store Selective Preset

1. Select one or more fixtures.
2. Give the fixture attributes some values in the programmer.
3. Press and hold **Store** until the **Store Settings** open.
4. Tap **Preset Mode** until the mode is **Selective**.
5. Tap the desired preset pool object - ensure it is valid for the attribute values.

Store Global Preset

Attribute values need to be tagged with a global flag to be stored as global data.

1. Select one fixture from one or more fixture types.
2. Give the fixture attributes some values in the programmer.
3. Press and hold **Store** until the **Store Settings** open.
4. Tap **Preset Mode** until the mode is **Global**.
5. Tap the desired preset pool object - ensure it is valid for the attribute values.

Store Universal Preset

1. Select one or more fixtures or the universal fixture.
2. Give the fixture attributes some values in the programmer.
3. Press and hold **Store** until the **Store Settings** open.
4. Tap **Preset Mode** until the mode is **Universal**.
5. Tap the desired preset pool object - ensure it is valid for the attribute values.

Values that are a part of the universal fixture type are stored in this fixture. Attributes outside the scope of the universal fixture type are stored as global values.

MAGic Presets

The concept of MAGic presets is explained in the [Presets](#) topic.



Important:

The MAGic presets only work if the stored fixtures have different grid positions on the relevant axes, and the best result is if they start at grid position 0 and the fixtures are next to each other. This does not need to match the real-world position; it is only for the individual positions between the fixtures.



Important:

Calling a MAGic preset in the programmer does not create a link to the preset. The fixture values are calculated based on the selection and stored as hard values. The only way to keep a reference to the MAGic preset is if the preset is used in a recipe. Learn more about recipes in the [Recipe Preset topic](#).

This is the general workflow for creating a MAGic preset:

1. Select all the needed fixtures.
2. Use **Next** to give the first fixture values to match the first point in the range.
3. Press **Next** to select the next fixture and give it values for the next point in the range.
4. Repeat step 3, if needed, to have a maximum of five fixtures (on each axis in the selection grid).
5. Select all the used fixtures using **Set**.
6. Store the preset in an appropriate preset pool.
7. Edit the preset settings and tap **MAGic** to turn the MAGic function On.

The new preset can now be used by as many fixtures as needed.

Learn more about editing the [Edit Presets topic](#).



Known Limitation:

Storing MAGic preset only works with selective data as you cannot define multiple points across universal or global fixtures.

Example:

A path must be defined to control a color range from blue to yellow. The wish is to have the range go through the magenta/red area in a CIE color picker - instead of going through white or the blue/green area. Three fixtures are needed to define the range.

Requirement:

- Patch several fixtures (10 or more) with color mixing possibilities.
- Create a view with the color picker (in CIE mode) and the All1 preset pool.

Follow these steps:

1. Select three of the fixtures.
2. Turn the fixture intensity to 100%.
3. Press **Next** to select the first of the three fixtures.
4. Tap the blue area in the color picker.
5. Press **Next** to select the second fixture.
6. Tap the magenta area in the color picker.
7. Press **Next** to select the third fixture.
8. Tap the yellow area in the color picker.
9. Press **Set** to select all three fixtures again.
10. Press **Store**.
11. Tap an empty pool object in the All1 preset pool.
12. Use the **swipey menu** to select **Edit Settings**.
13. Tap **MAGic** to turn On the option and close the settings.

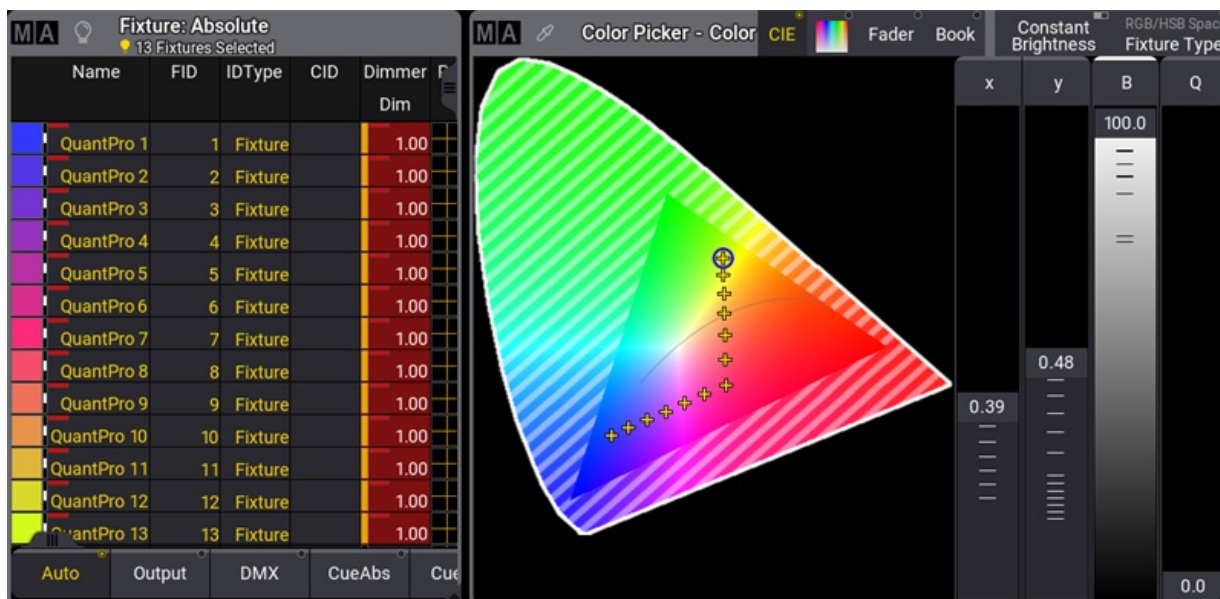
Now there is a preset that looks something like this:



MAGic preset in the preset pool

14. Clear the programmer.
15. Select all the fixtures.
16. Tap the MAGic preset.

The result should look similar to this in the color picker:





Result of the MAagic preset used on multiple fixtures

The dimmer values are not needed for this example; it just makes it easier to see the result.

Multistep Preset

Multistep presets are presets that contain values in more than one step. It is explained in a little bit more detail in the [Presets](#) topic. For an explanation of steps, please read the [Phasers](#) section.

With Phaser information in the programmer, a preset can be stored. This will then be a multistep preset.



Hint:

Presets can be used for more advanced functions with **Recipes**. Please read the [Recipe Presets](#) topic for more about recipes.



24.3. Recipe Presets

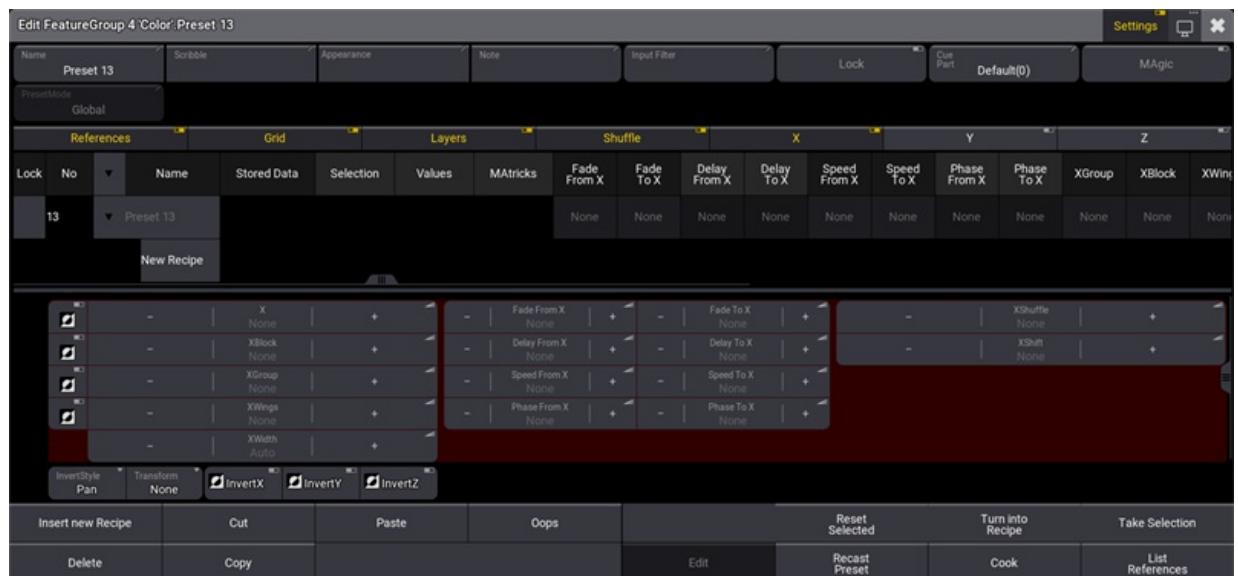
Please read the [Recipe topic](#) before this topic.

This topic explores using recipes in presets.

Recipes can only be added to presets that do not contain attribute values when using the user interface. Presets with attribute data offer to turn the existing attribute data into a recipe (by tapping **Turn into Recipe** - read more below).

Adding Recipe Lines

Recipe lines are added to presets using the **Edit Preset Object** pop-up. This can be accessed using the [Swipey](#) on a preset pool object.



Edit options for a preset

The middle part of this pop-up is about the recipes and, for instance, grid values.

Besides the main recipe area (described in the [Recipe topic](#)), the recipe editor area has a colored background. This is just a different tool to edit and see the values in the selected recipe line.

The bottom of the editor has different buttons:

- **Insert New Recipe** (available when the preset is turned into a recipe preset):
Creates a new recipe line. Adding additional recipe lines will reference the values from the first line when values are stored in the first line. When the first line contains a preset, the reference to this preset is copied. Additional lines also automatically reference the MAtricks values from the first line.
- **Cut:**
Cuts the selected recipe line.



- **Paste:**
Paste the copied or cut recipe line.
- **Oops:**
Oops the last action.
- **Delete:**
Delete the selected recipe line.
- **Copy:**
Copies the selected recipe line.
- **Edit/Edit Preset:**
Closes the editor and takes the selected recipe line into the programmer in edit mode.
- **Reset Selected:**
Resets the MAtricks for the selected recipe line.
- **Turn Into Recipe:**
Tap this button to turn a preset containing attribute value into a recipe preset. A new empty preset must also be turned into a recipe preset before adding recipe lines.
- **Take Selection:**
This allows using a selection of fixtures instead of a group. Tap this button to make the current programmer fixture selection the new selection for the selected recipe line.
- **Recast Preset:**
This will recast the preset where it is referenced. This means that if attributes are added or deleted after the preset is used in cues, then the preset might need to be recast for the cues to reflect the new content. Read about this in the [Cue Recipes topic](#).
- **Cook:**
This will cook the preset recipes. This might be necessary for the preset to reflect the recipe content.
- **List References:**
Tapping this opens a pop-up listing all elements that reference and depends on

Create a New Recipe Preset

Open the editor for an empty preset.

Tap **Turn into Recipe** in the menu at the bottom.

This turns the preset into a recipe preset and creates the first recipe line.

It is almost always relevant to at least add a preset reference value. To do this, tap and hold the field in the recipe row in the values column.

This opens a **Preset Pool** selection pop-up. Here, navigating the existing presets and selecting the desired preset is possible. It does not have to be in the same feature group as the recipe preset.

Add the desired values in the other columns.



Add a selection if desired - from a group or by tapping **Take Selection** .

If the recipe contains a group, then the preset is automatically cooked when the edit pop-up is closed.

Example

We want a preset that can take the current selection of fixtures to a new position in two wings and with a ranged time.

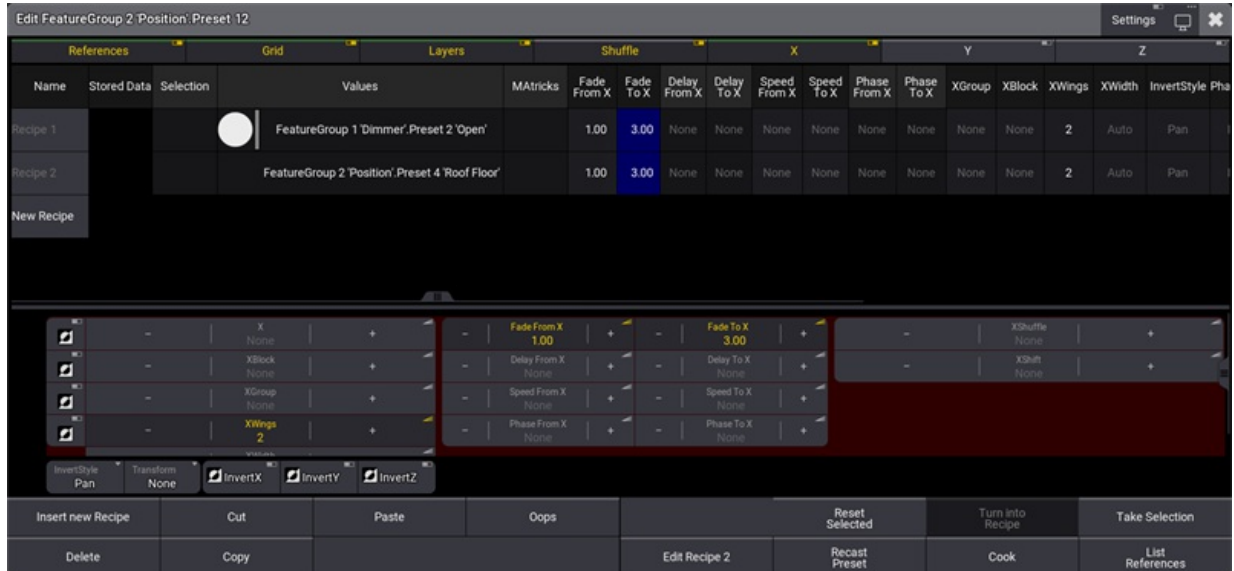
Requirement:

Have a show with some moving heads patched and placed in a row. The show also needs a position preset and dimmer preset where the fixtures are at full. This example uses the Demoshow.

Follow these steps to create the recipe preset:

1. Tap an empty position preset pool object and swipe the finger outside the preset to open the Swipecy menu.
2. Select **Edit Setting** in the Swipecy menu.
3. Tap **Turn into Recipe** to be able to create recipe lines.
4. Tap and hold the first field in the "Values" column.
5. Tap **Dimmer** and then the preset where the intensity is at full (Open).
6. Tap and hold **New Recipe** to create a second line.
7. Press **Assign** .
8. Tap the position preset (Roof Floor) in the position preset pool.
9. Tap the second recipe line to assign the preset to the line.
10. Select both recipe lines in the XWings column (if the column is not visible, then make sure both "X" and "Grid" are active in the row with toggle buttons).
11. Edit this value so it says "2" in the XWings.
12. Select both rows in the "Fade From X" column and set the value to "1".
13. Select both rows in the "Fade To X" column and set the value to "3".

It should look like this ("Settings" is turned Off in the title bar):



The finished recipe

14. Close the editor.
15. Select some fixtures (for instance, group 2).
16. Tap the new recipe preset and see the fixture move while they turn on.

If the move and fade should start from the center then open the edit preset object pop-up again and set the InvertStyle to "All" and the InvertX to "Yes" for both recipe lines.


Working with Recipe Presets

There are two suggested workflows with presets using recipes:

- Preset containing recipe with a selection.
- Preset containing recipe without a selection.

Preset Containing Recipe With a Selection

The recipe will be loaded into fixtures selected in the programmer (**Programmer Parts window**); therefore, the values are cooked into the programmer. These values can be stored or removed/overwritten in the programmer.

 **Hint:**
Using Presets this way is a quick way to build recipes into cues. After the recipes are in the cues, the preset is no longer needed; the recipes should only be worked with inside the cues. Think of it as a "Recipe Template Preset".

Recipe presets with a selection can be played back from executors. Learn more about this in the [Use Presets topic](#).



Preset Containing Recipe Without a Selection

This recipe works with the current programmer selection of fixtures. The recipe is not loaded into the programmer. Instead, the recipe value links and grid values are used with the current selection of fixtures to cook values directly to the programmer.



Important:

If values are stored into the recipe and then such a recipe preset with no selection stored is called, the values will be called into the programmer, but without a preset link.

Recipe presets with no selection stored will call values with preset links into the programmer when a preset is defined in the preset instead of storing values in the recipe.

This is a great way to have a bunch of template objects in the preset pools that allow quickly calling complex looks based on grid values and MAtricks ranges.

24.4. Use Preset

Presets are often used for live playback or as building blocks in cues. Tapping a preset to get the values uses the **At** keyword.

Calling Preset Into the Programmer

The first step is to select the desired fixtures and then call the preset into the programmer.

The workflow is the same for every type of preset (MAgic, Recipe, Multistep, standard single step, etc.)

If the programmer does not have a fixture selection and a preset is tapped, then the first tap selects all the fixtures that can use the preset. In this case, it works like a group.

Tapping a preset with a fixture selection in the programmer calls the preset if it is valid for the selected fixtures.



Hint:

Please be aware that the **Default Pool Action** could be changed. This text assumes the factory default **SelfFix/At** action. Learn more about default pool action in the [Windows Settings topic](#).

Timing values can be stored together with attribute values in the preset, and calling a preset that includes timing information calls the values into the programmer using the preset timing.

Calling presets that only contain timing values call the timing values into the timing layers of the programmer.

Another way to use timing with presets is the **Programmer Time** master. Learn more about the master in the [Time Control](#) topic.

Presets with stored timing values have a higher priority than the **Programmer Time** master, and the stored timing will be used when the preset is called.



Known Limitation:

Calling a MAgic preset into the programmer will extract the data and not reference the preset. It is recommended to use MAgic presets in combination with recipes to maintain referenceable data.

Presets Running from Executors

Presets can be assigned to executors using the [Assign keyword](#) or the Assign menu. Learn more about this in the [Assign Object to an Executor topic](#).

If the preset uses global or universal data, the executor button should use the **At** command to call the preset values.



Important:

Using the At command on presets assigned to executors does not playback the values from the executor. The values are called into the selected fixtures in the programmer just like described above - if the preset is valid for the fixture selection.

Other playback commands (like Go+, Toggle, Flash, etc.) are relevant for presets with Selective data. This also means that presets with selective data can be played back from the preset pools using, for instance, a Go+ command.

Opening the **Edit Setting** tab of an executor's assign menu with a preset assigned offers some of the same settings as for sequences. For more information on playback settings, see the [Sequence Settings topics](#).

Edit the global preset playback settings in **Menu** - **Preferences and Timing** - **Preset** .

The OffFade property of preset playbacks determines the fade time when switching off a selective preset playback.

Store Presets in Cues

When a preset is in the programmer, it can be stored in cues. Learn more about storing cues in the [Store Cues](#) topic.

A reference to the preset is stored for the specific fixture attributes. This means that the value stored in the preset is not stored in the cue, but a reference to the preset is stored. So if the value stored in the preset is changed after the cue is stored, then the cue still looks in the preset to get the value when the cue is played back, and the values in the preset will be used.

Be aware that the **Absolute** and **Relative** value layers are connected to the Fade and the Delay layers. Individual fade and delay times for an attribute use the Fade and Delay layers. Storing a value in an attribute's absolute or relative layer in a preset can also add values from the fade and delay layers.

This means that if a preset contains absolute or relative values, it also connects to possible values in the fade and delay layers. If the fade or delay values are later added, changed, or updated, this change will be reflected in the cue referencing the absolute or relative value.

Fade and Delay values connected to an absolute value has a higher priority than fade and delay values connected to a relative value. For instance, if two presets are called, each with fade and delay values, but one is for absolute values, and the other is for relative values.

The phaser layers (**Speed, Phase, Measure, Accel, Decel, Transition, and Width**) are also linked. Adding one of these layers in a preset effectively adds all the phaser layers.

This is very important to keep in mind when updating the preset information.

If attributes are added to or deleted from the preset after it is used in a cue. The stored cues referencing the preset



need to reflect this new change, then the preset needs to be recast. This can be done using the **Recast** keyword or the **Edit Preset Object pop-up**. Recasting a preset removes or adds attributes to the fixtures in the cue.

The recast function only recasts presets to cues where a preset reference exists for the absolute layer.

When presets are used in cues and later deleted, the preset values are transferred to the cue (hardcoded values in the cue).

Assign Presets to Attributes in the Tracking Sheet

When a value is edited in the **Track Sheet**, the available presets can be chosen in the **Calculator**.

Extract Preset Values

Preset values can be extracted to the selected fixtures. The preset needs to be valid for the selected fixtures. The **Extract** keyword is used for this.

The values stored in the preset will be pulled into the programmer without a reference to the preset. The values are then like any other typical programmer values.

24.5. Edit or Update Presets

Edit Existing Preset

The existing presets can be edited. The values in the preset can be seen in a **Fixture Sheet** in **Fixture** mode.

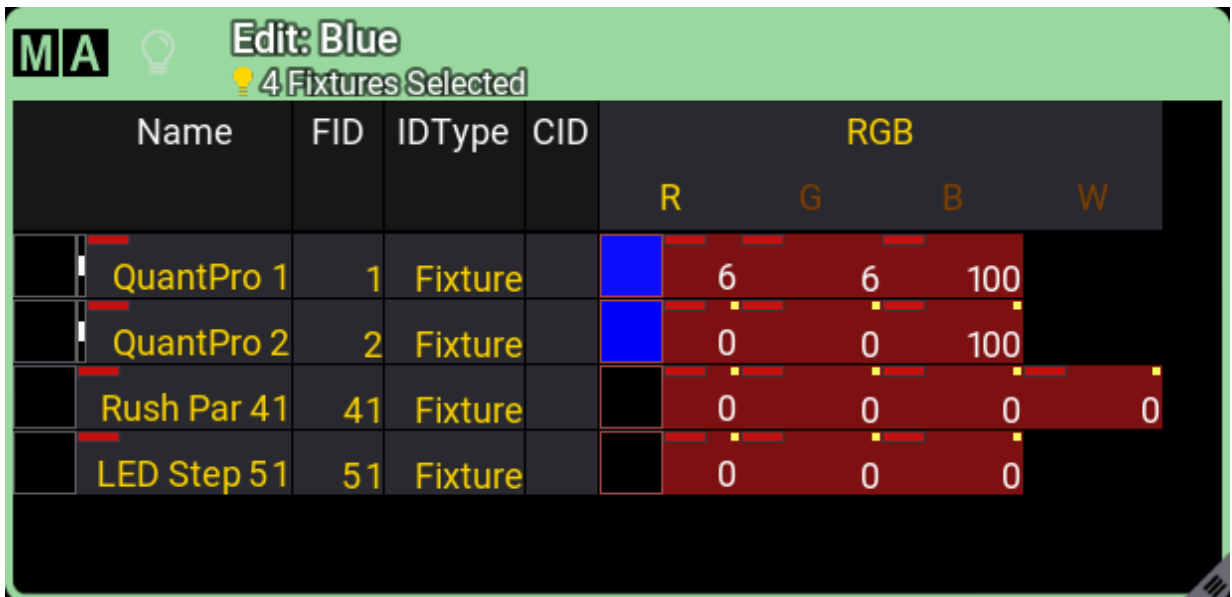


Important:

Editing a preset call the values into the programmer and this is sent to output unless **Blind** mode is activated first.

1. Press **Edit**.
2. Tap the preset to be edited - this changes the frame around the fixture sheet to a green color (yellow frame on other stations with the same user), the fixtures are selected, and the values are pulled into the programmer as active values. **Edit** starts blinking.

The sheet shows the fixtures that can use the preset and the attribute values.



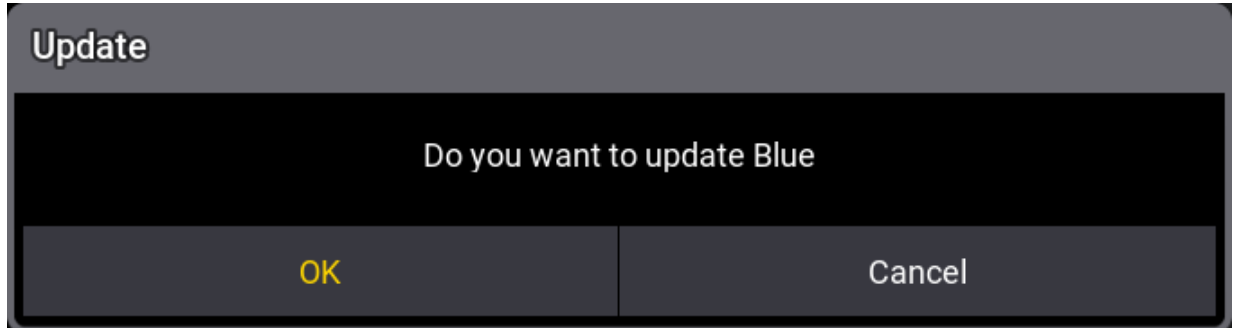
Name	FID	IDType	CID	RGB			
				R	G	B	W
QuantPro 1	1	Fixture		6	6	100	
QuantPro 2	2	Fixture		0	0	100	
Rush Par 41	41	Fixture		0	0	0	0
LED Step 51	51	Fixture		0	0	0	

Fixture sheet in edit mode

The example above is a **Global** and **Selective** preset. The global data is marked with a yellow marker in the upper right corner of the attribute values. Any fixture with that fixture type gets the values from the global fixture type. The selective data are values without the yellow marker.

3. Edit the values in the programmer. **Update** lights up.
4. When the values are correct, press **Update**.

This opens a pop-up asking for confirmation to update the preset.



Pop-up asking for update confirmation

5. Tap **OK** to confirm the update. **Cancel** will return to the update menu and not update the preset.



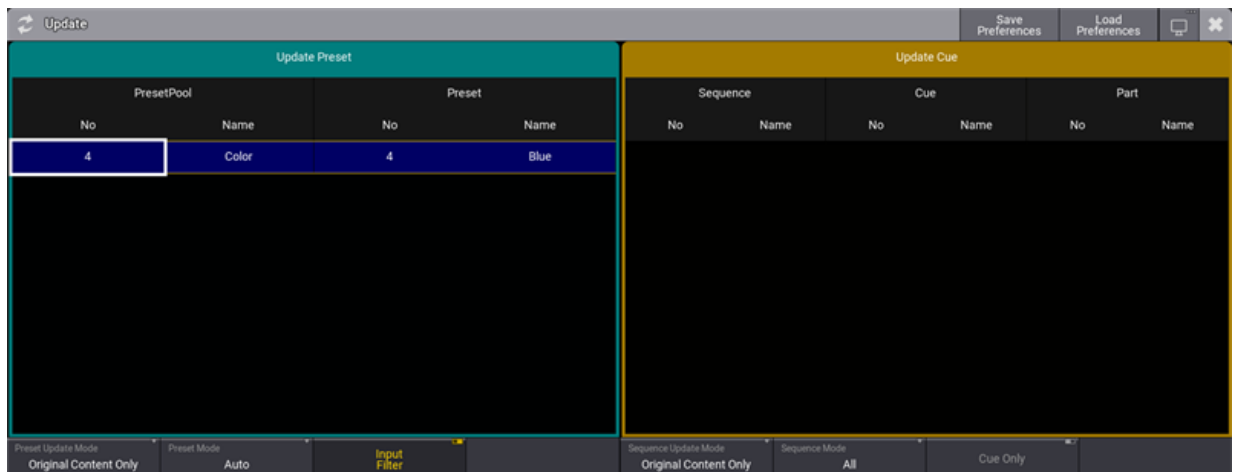
Hint:

To deactivate the edit mode without saving anything, press **Esc**. Remember that the values are still in the programmer.

Update Existing Preset

A preset called into the programmer, where the values changed after it was called, can be updated.

Update flashes when there are values that can be updated. Tap the key to open the **Update Menu**.



Update Menu

The presets that can be updated are listed on the left side of the menu.

The **Preset Update Mode** is relevant when updating presets. There are two modes:

- **Original Content Only:**
Only existing attribute values are updated if a preset is updated with this mode.



- **Add New Content:**

This mode update existing attribute values and adds any new attribute values to the preset.

The mode can be toggled by tapping **Preset Update Mode**.

The **Preset Mode** defines how the preset values are updated.

This setting sets what mode will be used when updating the preset. The options are **Auto**, **Selective**, **Global**, **ForceGlobal**, and **Universal**. These modes are described in the [Preset topic](#).

The mode set here is not linked to the mode selected in the store settings.

The **Input Filter** setting is also used when updating presets. It can be toggled to turn On or Off the input filters when updating preset values. Learn more about input filters in the [Create new Preset topic](#).

When the correct settings are selected, then the desired preset can be updated by tapping it in the menu.

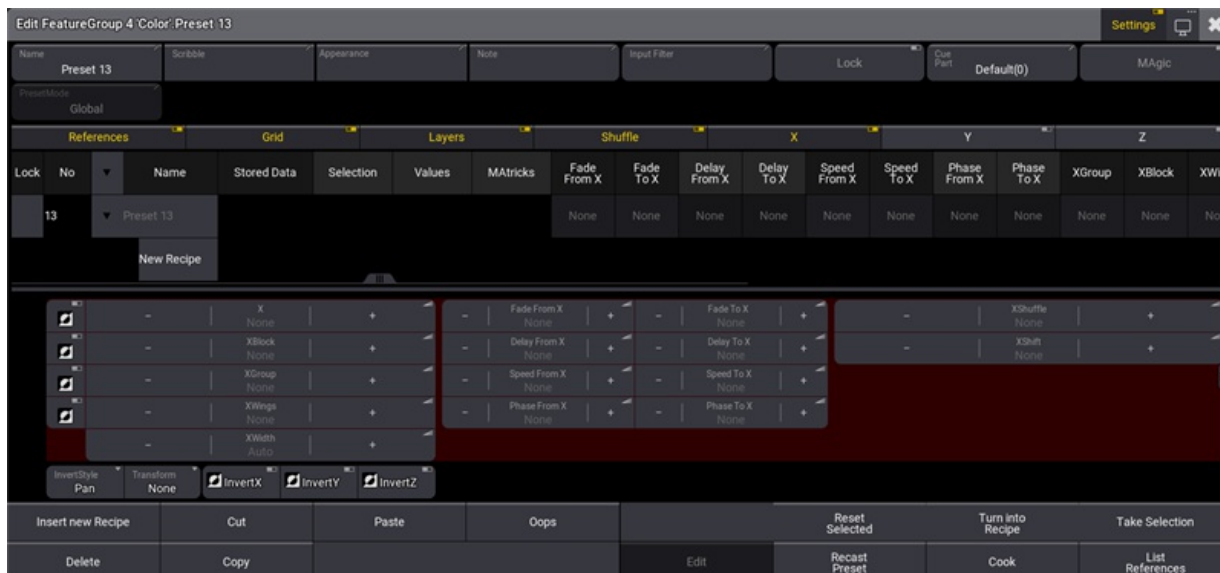
Absolute and relative layers are now handled separately during preset updates.

When adding a relative layer value to a preset with an existing absolute layer value, the relative value will not automatically be added to the cue where the preset is already used. Vice versa, when adding an absolute layer value to a preset with an existing relative layer value, the absolute value will not automatically be added to the cue where the preset is already used. Recasting the preset adds information from all layers to the relevant cues. Recast can be done from the preset editor or using the command line. Learn more in the [Recast keyword](#).

Recipe presets that reference the updated preset might need to be recooked to look as expected, and cues might need to be recast for the expected result.

Edit the Preset Object

The preset object can be edited using the edit pop-up. It can be accessed using the [Swipecy](#) menu or the [EditSetting keyword](#).



Edit preset object including the settings

The settings in the top line are the same options as the ones for the entire pool, but here they can be set for the specific preset. Settings on a preset have a higher priority than the pool settings. Learn about the settings in the [Preset Pool](#) topic.

There is an extra setting here called MAagic. It is a toggle button that changes the behavior when the preset is called into the programmer.

Turning on **MAagic** is used for MAagic presets. Learn more about creating and using MAagic preset in the [Create New Presets](#) topic.

The settings on the top can be shown or hidden using the **Settings** toggle button in the title bar.

The rest of the editor is for recipe lines. Learn about these in the [Recipe Presets](#) topic.

25. Worlds and Filters

Worlds and filters can be used as tools for programming, playback, or filtering information in some windows.

Worlds are used to limit access to fixtures and attributes.

Filters are used to prevent attributes to pass a filter. Typically in store, update, and recall actions but also to filter what is displayed in some windows.

The selected world and filter always dictate what is possible using the programmer.

Worlds and filters can be assigned to some objects. For instance, a sequence or a preset. The world or filter then dictates what can be played back from or stored into the object.

Worlds

Worlds contain information about fixtures and attributes.

Worlds are used to prevent access. Fixtures and attributes not in the active world are removed in some windows and cannot be used in programmer actions.

They are stored in the Worlds pool. This can be created like any window using the **Add Window** pop-up.



World pool with world 1 selected

There is always a selected world. The selected world has a yellow frame around it.

There is a default world from the factory. It is always world number 1 and in a new show it is called "All". This automatically contains all fixtures and all attributes. This world is locked and cannot be edited.

Filters

Filters contain information about attributes and patch information about the fixture (Name, ID Type, Fixture Type, Layer, and Class).

Filters are used to block values from being stored or recalled. For instance, assigning a filter that blocks dimmer values as a playback filter on a sequence prevents the dimmer values from being played back from that sequence. Assigning a filter that blocks certain ID Types to a **Layout** will hide these fixtures in the layout view.

Filters are stored in the Filters pool. This can be created like any window using the **Add Window** pop-up.



Filter pool with filter 1 selected

There is always a selected filter. The selected filter has a yellow frame around it.

A new show will have some default filters from the factory. The first one is called "All". It contains all attributes and layers. If the attribute structure changes, then this filter is automatically updated. The filter is locked and cannot be edited.

If a different filter than number 1 is the selected filter, then the **At key** flashes to indicate that there is an active filter.

Feature Group Indicator Bar

This bar is visible at the bottom of the world or filter pool object when a world or filter does not have all feature groups stored.

Learn more about the bar in the [Preset topic](#).

Read more about worlds and filters in the subtopics.

25.1. At Filter

The **At Filter window** can be used to see and change the current filter settings.

The principle is that turned Off elements are blocked and do not pass the filter.

Attributes with yellow text and a yellow bar above the text are turned On, and they will allow information to pass through the filter.

There are three column groups. The left column displays the **Feature Groups**. The next column is the **Features** in the feature groups. The last group is the **Attributes** inside the features. Read more about the feature and attribute structure in the [Feature Group topic](#).

The elements in this list change with the fixture setup. If fixtures are added with more attributes, then these attributes are added to the list.

The right-hand side of the window has another column with filters for different **Layers**.

The window can be created like any other window using the [Add Window pop-up](#). It is in the **Tools** tab.



At Filter Window with all attributes selected

The image above shows all attributes and layers selected (yellow text).

The title bar in the **At Filter window** has some buttons:

- **Line Height:**
This swipe button changes the spacing above and below the text in the main window. There is a **Default** value. It corresponds to size 50.



- **Select All:**
Tapping this button select all attributes and layers.
- **Select None:**
Tapping this deselects all attributes but does not change the layer selection.

The main part of the window displays all the attributes and the structure.

The structure dictates that if a feature group is Off, then all features inside are also Off. If a feature is Off then all attributes inside are Off.

If a single attribute is On, then both the feature and feature group is On.

The **At Filter** window follows the selected filter in the **Filter Pool**.

Filter 1 in the filter pool is called **Open**. This contains all attributes. It is automatically updated if the structure changes. It is locked and cannot be changed.

Temporary Version

The **At Filter** window exist as a temporary pop-up version that can be opened by pressing and holding the **At key**, by tapping **At Filter** in the **At overlay**, or by executing the following command:

```
MA User name[Fixture]> Menu "AtFilterOverlay"
```

The functions are the same in the pop-up as in the window.



25.2. Create a World

Worlds are created in the programmer and stored in the world pool.

Worlds may be thought of as matrix with rows (fixtures) and columns (attributes), and being in a world may block programmer access to rows and/or columns in sheets.

The created world will contain the rows of the current fixture selection and the columns of any active attribute. All columns/attributes are included in the created world if no attributes were active. If no fixtures were selected, then all fixtures are a part of the world.

Examples

Requirement:

Have a world pool and **fixture sheet** visible on a screen.

Create a World with Fixture 1 to 5

1. Clear the programmer.
2. Select fixtures 1 thru 5 (or any 5 fixtures in your show).
3. Press **Store** followed by an empty world in the **World Pool**.
4. Tap the newly created world.

Now there is only programming access to fixtures 1 thru 5, reflected by the fixture sheet. All attributes are available.

Create a World with Fixtures and Attributes

Worlds do not contain any values, but active attribute values are needed to store the attribute information for filtering in the world.

Example with five fixtures and only one attribute:

1. Tap World 1 (Full) in the **World Pool**.
2. Clear the programmer.
3. Select fixtures 1 thru 5 (or any 5 fixtures in your show).
4. Give the fixtures a dimmer value - the values do not matter.
5. Press **Store** followed by tapping an empty world in the **World Pool**.
6. Tap the new world.

Look in a fixture sheet. It only displays the selected fixtures and the dimmer attribute.

Store a World with All Fixtures but Only Some Attributes

If the world is stored without selected fixtures, all fixtures can be used.

Example with only position attributes:



1. Tap World 1 (Full) in the **World Pool**.
2. Clear the programmer.
3. Select fixtures 1 thru 5 (or any 5 fixtures in your show).
4. Give the fixtures pan and tilt values - the values do not matter.
5. Press **Clear** once to clear the fixture selection, but keep the pan and tilt values active.
6. Press **Store** followed by tapping an empty world in the **World Pool**.
7. Tap the new world.

The fixture sheet lists all the fixtures but only the pan and tilt columns.

As usual, it is really a good idea to name the worlds as soon as they are created.

Edit a World

The content of the world can be edited using the following steps:

1. Press **Edit** followed by the desired world. This takes it into edit mode.
2. Change the programmer selection to match the desired fixtures and attributes.
3. Press **Update** followed by the world.
4. Confirm the update by tapping **OK** in the update pop-up.

The world settings can also be edited. This is done using the following steps:

1. Use the swipecy menu on the desired world.
2. Select the **Edit Settings** button in the swipecy. This opens an **Edit World** pop-up.
3. Make the desired changes.
4. Close the pop-up.

The edit world pop-up gives access to the following options:

- **Name:**
This is the name of the world.
- **Appearance:**
This can be used to select an appearance to the world.
- **Scribble:**
This can be used to select or create a scribble for the world.
- **Note:**
A note can be added to a world.
- **Lock:**
The world can be locked to avoid changes.

- **Selection:**

This defines if sub-fixtures should be part of the world or if it is only the actually selected fixtures that are part of the world. There are two options:

 - **Normal:**

This is the default option. Sub-fixtures are also part of the world when they were not selected when the world was stored. This includes fixtures in a selected Grouping fixture.
 - **Strict:**

Only the fixtures and sub-fixtures selected when storing the world are accessible.
- **Master:**

This can limit the intensity output of the fixtures in the world.



Important:

Changes to the **Selection** are not visible immediately when changing **Selection** values for the currently selected world. A different world has to be selected before selecting the desired world again.



Hint:

When storing a world, the Selection property can be specified within the command: **Store World**
["World_Name" or World_Number] Property "Selection" ["Option"]

There are also two buttons. **Edit Content** sets the world in edit mode, as described above. **List Reference** opens another pop-up that lists the different elements that use the world.

25.3. Create a Filter

Filters are stored in the filter pool.



Filter pool with some filters

There are two main ways to create filters:

- Create a filter using the pool by editing one of the empty pool objects. This opens the **Edit Filter pop-up**.
- Store a new filter pool object using the current filter settings.

The **At Filter window** is a nice tool for seeing the current filter settings.

Create Filter from the Pool

Editing a filter pool object opens a menu that looks almost the same as the **At Filter** window.



Edit Filter pop-up




The only difference between them is some extra input fields and the fixture patch information filter at the bottom.


The extra fields at the top can be used to edit the name, scribble, appearance, and note.

At the bottom, there is also a **List References** button that opens a list of elements that relates to this filter. For instance, if the filter is assigned to a preset.

The extra patch information filters at the bottom offer to add information from the patch as filters. The different information is Name, ID Type, Fixture Type, Layer, and Class. Read more about these elements in the **Patch and Fixture Setup section**.


Values can be added to each element type.

These values can be used as a positive or negative filter. Meaning that fixtures with the value can pass the filter or be blocked by the filter. Next to each of the elements is a  button. Tapping this toggles it On or Off. If this is white, then it is positive, and fixtures with the values can pass. If it is yellow, then it is negative (On), and fixtures with the values are blocked.

For example, if the **ID Type** has **Fixture** and **Channel** values and the  is white, then the filter allows fixtures with these two ID types to pass the filter. Other ID types are blocked. These filter options make it possible to have very precise filters.

The title bar has an extra button that allows this pop-up to be moved to a different display.

Follow these steps to create a new filter by editing the pool object:

1. Edit an empty filter pool object using swipecs, typing a command (**Filter keyword**), or using the keys (**Group key**).
2. Make sure the desired attributes and layers are active.
3. Add the desired patch information filter values.
4. Optionally give it a name by editing the **Name** input field.
5. Optionally assign an **Appearance** and **Scribble** by editing the input fields.
6. Optionally add a **Note** about the filter.
7. Close the editor by tapping the  in the upper right corner.

The edit function is the easiest way to update or change an existing filter.

Create Filter from Active Filter

The **At Filter** window is needed for this. Also, the **Filter** pool is nice to have visible.

1. Change the filter settings using the **At Filter** window.
2. Store a new filter by pressing **Store** followed by tapping an empty pool object or specify the filter using keys or commands (use links above).



25.4. Use a World or Filter

There are several scenarios where worlds and filters can be useful.

When Programming

Many actions obey the world and filtering. For instance store and delete.

When any store or delete action is used, then the action is filtered through the selected world and filter. Only the fixtures and attributes passing the filtering can be affected.

For instance, having filters for each feature might help to program different values into different cues or cue parts.

In Windows

Some windows can have a filter assigned as a mask for the window. This will hide the blocked elements from the view.

Windows also obey the selected world and hide fixtures and attributes that are not in the world.

The Fixture Sheet and Sequence Sheet also have the option to turn on the **Filter Toolbar**. This toolbar is the patch elements filter - learn more about it in the [Create a Filter topic](#). These filter options can be part of a filter from the pool, but they can also be a local sheet option in this toolbar.

The filters from the pool need to be assigned using the Settings pop-up. Read more in the [Window Settings topic](#).

On Sequences

Sequences can use worlds and filters as input and output filters.

A world or filter can be assigned as an input filter. This limits what can be stored in the sequence.

Worlds and filters can also be assigned as an output filter. This limits which fixtures and attributes the sequence sends values to during playback.

Read more about assigning filters in the [Sequence Settings topic](#).

In Preset Pools

Preset pools and single presets can use worlds and filters as input filters. This limits what can be stored in the preset pool or the single presets. This can be especially interesting for the All preset pools.

Read more about assigning the filters to presets in the [Preset section](#).



25.5. Delete a World

Worlds can be deleted like most other pool objects.

Deleting the world does not delete the fixtures or attributes from the show. It is only the world that is deleted.

If a world is assigned as an input or output filter to a sequence or preset pool, then this is also removed when the world is deleted.

Opsing the deletion brings back the world and also the filter settings for the sequence and preset pool.

The general syntax for deleting a world is:

Delete World ["World_Name" or World_Number]

It can also be a range of worlds.

World number 1 is locked from the factory and cannot be deleted.



25.6. Delete a Filter

Filters can be deleted like most other pool objects.

Deleting the filter does not delete the attributes from the show. It is only the filter that is deleted.

If a filter is assigned as an input or output filter to a sequence or preset, then this is also removed when the filter is deleted.

Opsing the deletion brings back the filter and also the filter settings for the sequence and preset.

The general syntax for deleting a filter is:

Delete Filter ["Filter_Name" or Filter_Number]

It can also be a range of filters.

Filter number 1 is locked from the factory and cannot be deleted.



26. MAtricks and Shuffle

MAtricks is a tool that can be used to divide a selection of fixtures into sub-selections.

The general workflow is that a selection of fixtures is selected, then different MAtricks settings are applied to select fixtures inside the primary selection.

For example, ten fixtures are selected, and you want to step through these ten fixtures one at a time to do a position correction. MAtricks is the tool used to do this.

The selection can be shuffled using a set of shuffle tools. Read more [below](#).

MAtricks Tools

One of the ways to work with MAtricks is the MAtricks toolbar or window.

The MAtricks window is separated into 3 sections. One for each axis. The X-axis section has a red background, the Y-axis section has a blue background, and the Z-axis section has a green background.

Tap **X**, **Y**, or **Z** in the toolbar on the left side to display or hide the corresponding axis.

Each axis section has its properties that are grouped.

Here are the groups and their properties:

- **Grid:** Axis (X, Y, Z), Block, Group, Wings, and Width.
- **Layers:** Fade From/To, Delay From/To, Speed From/To, and Phase From/To.
- **Shuffle:** Shuffle and Shift.

Tap **Grid**, **Layers**, or **Shuffle** to display or hide the corresponding group in the title bar.



MAtricks tools in a window

The above is the MAtricks tools available in a window that can be created like any other window.



An overlay version of the window can be opened by tapping **MAtricks** in the standard **Encoder bar**.

The two versions have the same buttons and options. Read the next topics for details about the different options.

Dimensions and Selection Grid

The MAtricks toolbar shows a lot of settings that can be applied to X, Y, and Z. These are the three dimensions used by the **Selection Grid**.

If the fixtures are in a 3-dimensional grid selection, then the MAtricks tool can be used in all three dimensions.

MAtricks Pool

The different settings that can be made in the MAtricks tool can be stored in the **MAtricks Pool**.

This pool can be created like any other window. You can find it under the **Data Pools** tab.

A screenshot of the MAtricks pool interface. It features a dark grey background with a grid of 10 buttons. The top-left button is green and contains the 'MA' logo and a grid icon. Below it is a button labeled 'MAtricks'. The remaining 9 buttons are arranged in a 2x5 grid and are labeled with numbers 1 through 9 and their corresponding XGroup settings: 1 NO XGroup, 2 2 XGroup, 3 3 XGroup, 4 4 XGroup, 5 5 XGroup, 6 6 XGroup, 7 8 XGroup, 8 10 XGroup, and 9 12 XGroup. A small menu icon is visible in the bottom right corner of the grid.

MAtricks pool with some store MAtricks

This pool works just like most pools in the grandMA3. The two most used functions are storing the MAtricks settings and recalling them.

Store a New Pool Object

This is the process for storing a MAtricks pool object.

The easiest way to do this is by long-pressing an empty pool object. This stores the current MAtricks settings (even if no settings are stored).

The general syntax uses the **Store keyword** and the **MAtricks keyword**: **Store MAtricks [MAtricks_Number]**

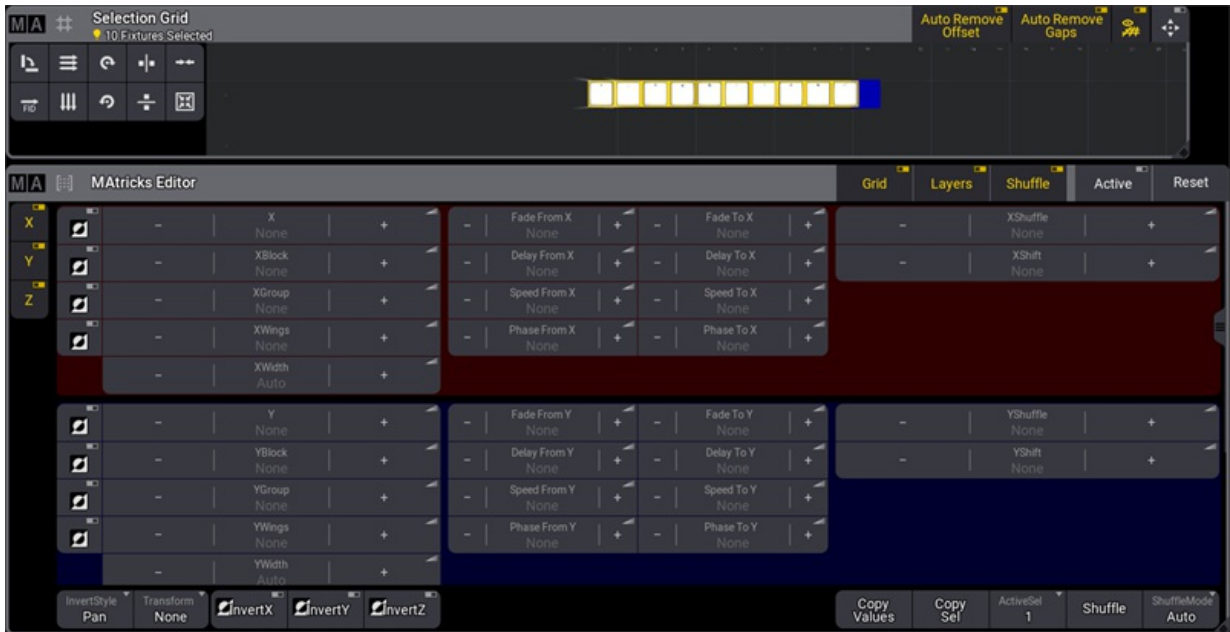
Simple Example

This simple example uses ten fixtures.

Select the ten fixtures without any specific grid information. Press **Full** to turn the intensity on.




This is how it looks in the **Selection Grid** and the **MAtricks** window:



Ten fixtures without any MAtricks settings

These ten fixtures are now in one row on the X-axis.

Press **Next**. This is the same as tapping the **+** in the X setting.

 **Hint:**
It is also possible to tap, hold, and slide your finger across any value area to change it.

This is the result:



Ten fixtures with X 0

Notice that the MAtricks is active - the **Active** button in the title bar is On. This can be used to toggle the MAtricks tool On or Off without resetting the MAtricks settings.

The selection frame around the first fixture is the usual yellow color. The rest have a darker yellow selection color. This indicates that they are part of the bigger selection.

Try pressing **Next** to step through the selection. Pressing **Prev** (previous) goes through the selection in the opposite direction.

Notice how the X number in the MAtricks tool updates with the key presses. There is no difference between using the **-** and **+** buttons in the X-axis in the MAtricks tool or the keys.

A specific X number can be reached using the command line. For instance, if X needs to be 6, then the following command can be used:

```
MA User name[Fixture]>Set Selection MAtricks 'X' 6
```

When you are comfortable with **Next** and **Prev**, then press **Set**.

When **Set** is pressed, the MAtricks toggles between active and inactive.

Multiple Selection

Each user profile has two different fixture selections called selection 1 and selection 2.



Tap and hold **ActiveSel** at the button off the MAtricks tool; this opens the dropdown menu, and select selection 1 or 2.

The selection can also be changed using the **Selection keyword**.

For example, the command needed to change to selection 2 is:

```
MA User name[Fixture]> Select Selection 2
```

A selection of fixtures can be copied from one to the other by tapping **Copy Sel**. It copies the currently active selection to the other.

Values can be cloned from one selection to another by tapping **Copy Values**. This applies the values from the currently active selections to the other selection.

A **ClearAll** command clears both selections. **ClearSelection** only clears the active selection.

Shuffle Selection

The **Shuffle keyword** allows shuffling the selection order in a random sort.

By default, Shuffle will randomize the selection order on all 3 axes in the **Selection Grid**.

Each shuffle setting per axis can be set to 0 (=None) up to 32 767. Each value represents a different shuffled selection order. When selecting the same amount of fixtures repeatedly, the same shuffle value will result in the same shuffled selection order. This can be useful when a specific nice shuffled selection order is desired for the same number of fixtures. In this case, apply the same shuffle value in the MAtricks when having the same number of fixtures selected. The fixtures are then shuffled the same way.

Tapping **Shuffle** in the MAtricks tool shuffles all three axes in the Selection Grid by adding a random number in XShuffle, YShuffle, and ZShuffle.

To shuffle on a single axis, enter a shuffle value for the desired axis in the MAtricks tool or tap the **+** or **-** in the axis until there is a wanted shuffle result.

On the right side of the MAtricks tool, there is a Shuffle Mode with three available modes:

- **Auto:**
When doing shuffle only for one dimension, this behaves like **Linked**. Shuffling on two or three axes behaves like **Unlinked**.
- **Linked:**
All fixtures placed on the same position along the axis that will be shuffled but have a different position on the other axis will keep their alignment along the other axis.
- **Unlinked:**
The fixtures placed on axes other than the axis that will be shuffled but have the same position on the shuffled axis will be shuffled independently.

Tap **ShuffleMode** to switch between these three modes.



Hint:

When deactivating or resetting the MATricks, the original selection order will be restored.

Since the shuffle is part of the MATricks, they are also stored in a MATricks pool object like any other MATricks setting.

See shuffle examples in the [Shuffle topic](#).

Shift Selection

The shift setting in the MATricks tool allows for shifting the current selection within the selection grid positions.

This can be done per axis in the grid. Therefore, change the values for **XShift**, **YShift**, or **ZShift**.

Positive values shift to the right (x-axis), to the bottom (y-axis), and to the front (z-axis). According to this, negative values shift in the opposite direction.


Invert Options

Invert will define which axis in the selection grid the values should be inverted when turning the encoder or applying a range of values.

The inverted fixtures are displayed with a green font in the fixture sheet, a green body color in the 3D window, and a green border in the layout window and the selection grid window.

The following buttons can be found at the bottom left of the screen:

- **InvertStyle**: Defines if Invert shall be applied to Pan, Tilt, Pan and Tilt, or All attributes.
- **InvertX**: Inverts the overall invert of the current individual inverts per MATrick property on the X axis.
- **InvertY**: Inverts the overall invert of the current individual inverts per MATrick property on the Y axis.
- **InvertZ**: Inverts the overall invert of the current individual inverts per MATrick property on the Z axis.

Grid properties can also be inverted by tapping .



Hint:

When using Align in combination with Invert, the alignment is still based on the arrangement of fixtures inside the selection grid; however, the aligned values will be inverted.



Restriction:

At the moment, Align only works with the X axis.

26.1. MAticks Blocks

The blocks function in the MAticks creates blocks of fixtures of the specified size.

This treats blocks of fixtures as one fixture.

It is better explained with examples.

Example - Ten Fixtures One Axis

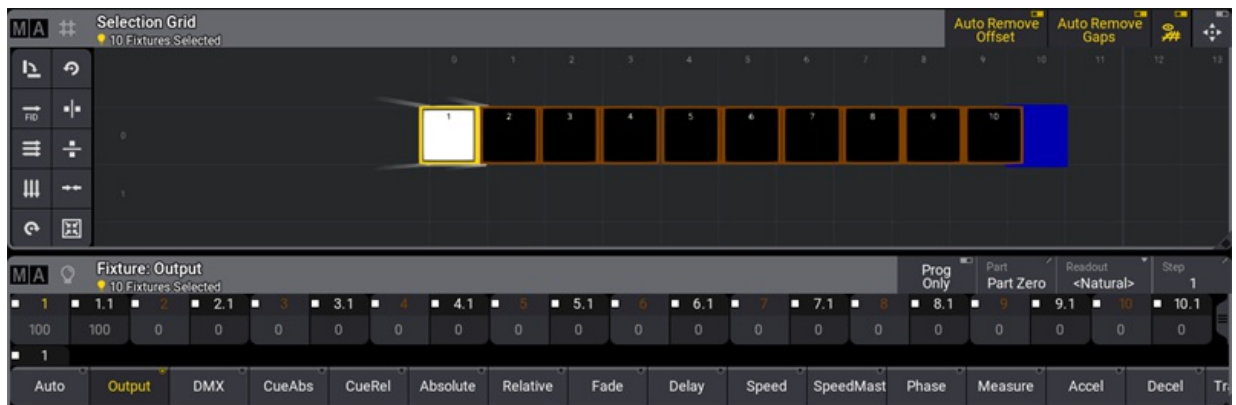
In this example, there are ten fixtures (1 thru 10).

They are selected from 1 to 10 without any specific grid information. Highlight is activated.

Since there is no grid information, the fixtures are only on one axis - the X-axis.

Press **Next** to make the MAticks X value 0.

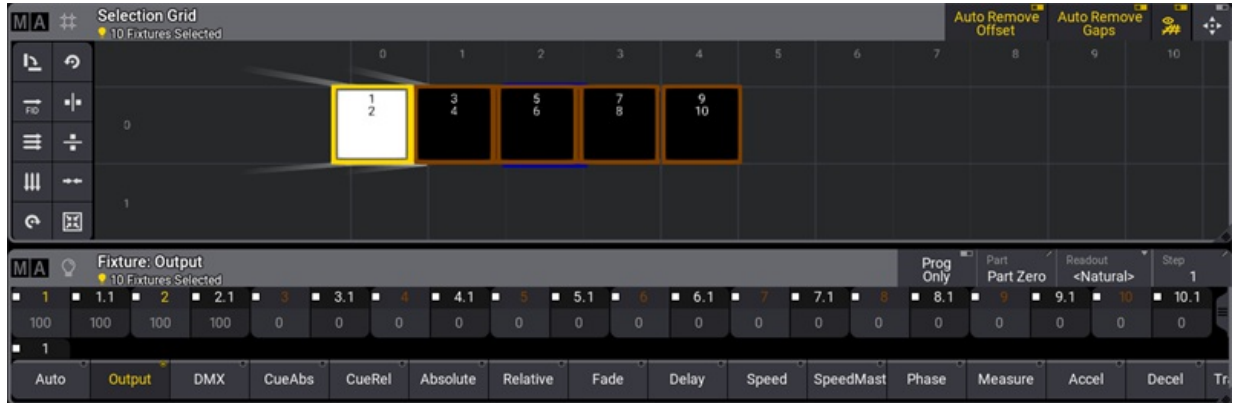
It looks like this in the Selection Grid and Fixture Sheet windows. (Fixture Sheet Mode set to Channel and Output layer selected). For more information, see [Fixture sheet](#).



Ten fixtures with MAticks X at 0 - No Block

Now tap **+** in the XBlock in the MAticks window to set the value to two.

Now it looks like this:



Ten fixtures with MAtricks X at 0 and XBlock at 2

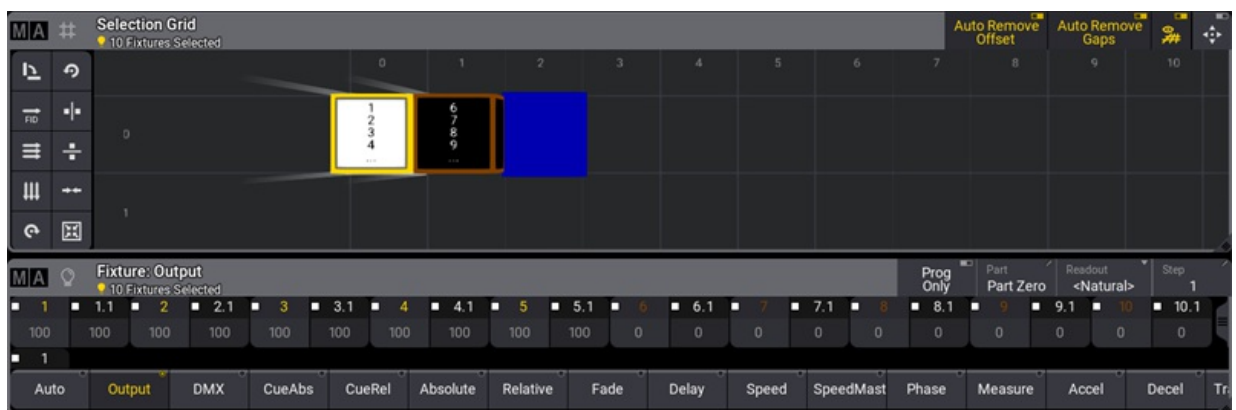
The selection grid now only shows five boxes. This is because fixtures 101 and 102 are blocked together and are in the first position. They are the ones currently outputting light. This can be seen in the fixture sheet.

Press **Next** and **Prev** to jump through the ten fixtures blocked together two by two.

Increasing the block size increases the number of fixtures together in a block. It decreases the number of visible boxes in the selection grid and the useful MAtricks X values.

For instance, if the block is increased to five, there are only two boxes in the grid, X = 0 and XBlock = 5.

It looks like this:



Ten fixtures with MAtricks X at 0 and XBlock at 5



26.2. MAtricks Groups

Groups in the MAtricks are used to separate the selection into the number of groups set.

It alternates through the selection putting fixtures into each group.

It is best explained with an example.

Example - Ten Fixtures One Axis

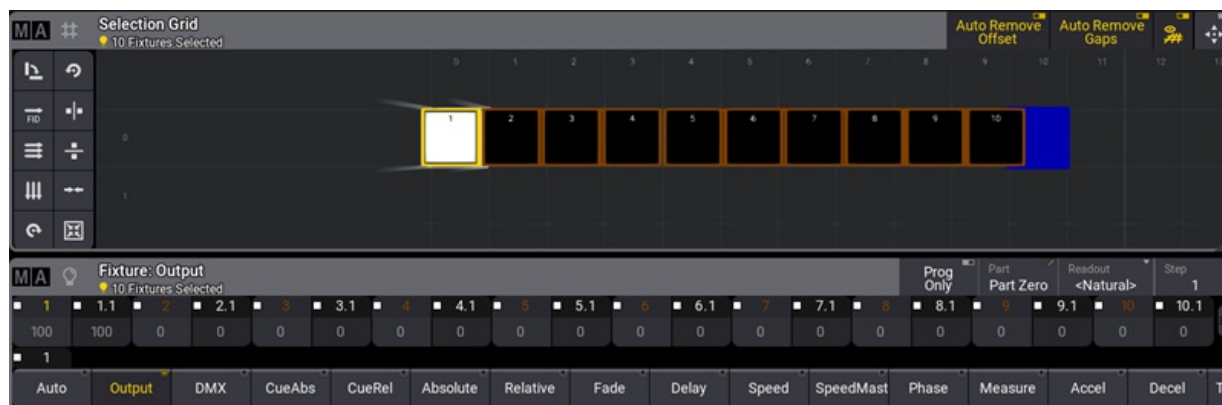
In this example, there are ten fixtures (1 through 10).

They are selected from 1 to 10 without any specific grid information. Highlight is activated.

Since there is no grid information, the fixtures are only on one axis - the X-axis.

Press **Next** to make the MAtricks X value 0.

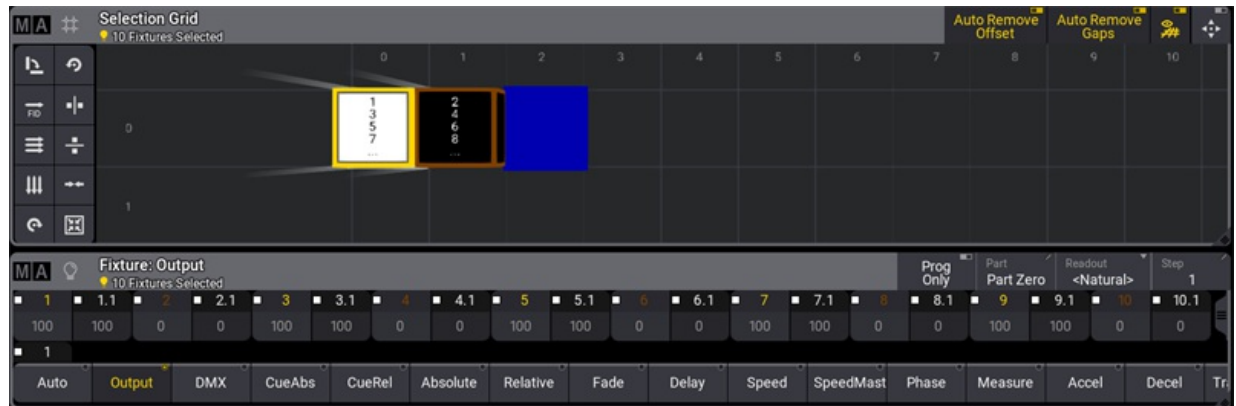
It looks like this in the Selection Grid and Fixture Sheet (Channel SheetMode and Output layer selected) window:



Ten fixtures with MAtricks X at 0 - No Group

Now tap **+** in the XGroup in the MAtricks window to set the value to two.

Now it looks like this:



Ten fixtures with MAtricks X at 0 - XGroup at 2

Notice how the ten fixtures are moved into two boxes in the selection grid.

The fixture sheet shows how the fixtures are interleaved into the two groups. The first fixture goes into the first group, the second fixture into the next group, the third fixture in the first group, and so on until there are no more fixtures.

The **Next** and **Prev** can be used to change the MAtricks X values. Values 0 and 1 are used to select the two groups.

26.3. MAtricks Wings

Wings in the MAtricks are used to separate the selection into the number of wings set and select devices from each wing from opposite directions.

It is better explained with examples.

Example - Ten Fixtures One Axis

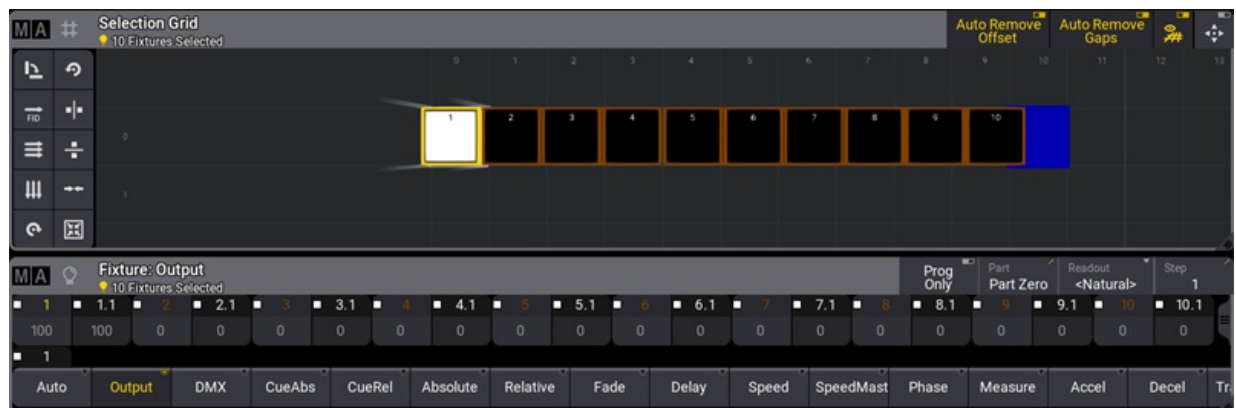
In this example, there are ten fixtures (1 through 10).

They are selected from 1 to 10 without any specific grid information. Highlight is activated.

Since there is no grid information, the fixtures are only on one axis - the X-axis.

Press **Next** to make the MAtricks X value 0.

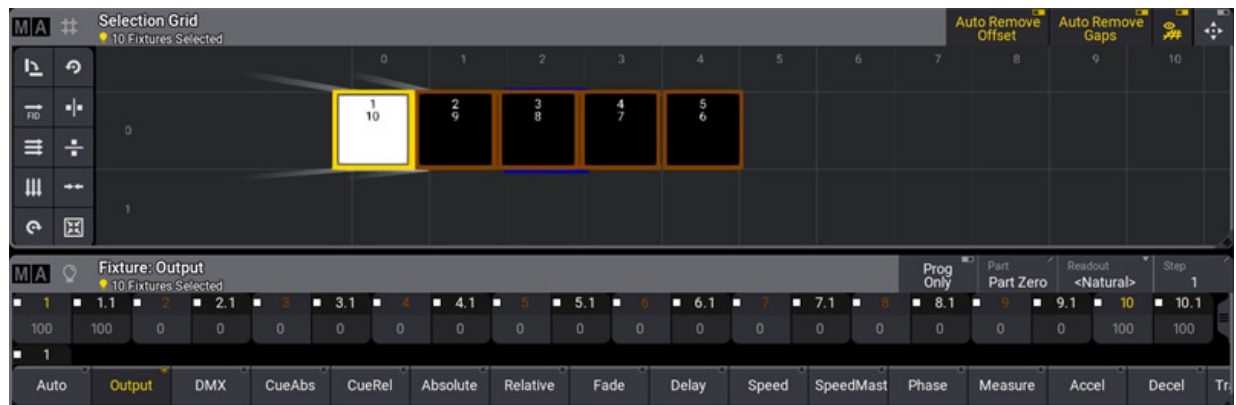
It looks like this in the Selection Grid and Fixture Sheet (Channel SheetMode and Output layer selected) window:



Ten fixtures with MAtricks X at 0 - No Wings

Now tap **+** in the XWings in the MAtricks window to set the value to two.

Now it looks like this:





Ten fixtures with MAtricks X at 0 and XWings at 2

The selection is split into two wings, and each end of the selection is currently On.

Use **Next** to change the X number up. This moves the highlighted selection towards the "center" of the selection. **Prev** moves out again.



26.4. MAtricks Width

Width can appear to give the same result as groups. The difference is that groups place the grouped fixtures in the same grid positions where width moves the fixtures out on the next axis.

This makes it possible to combine several axes in the MAtricks selection.

It can be a great tool for fixtures arranged in grids.

Having the fixtures in MAtricks with a width and storing a group in the **Group pool** also stores the current selection grid setup.

The MAtricks width is best explained with an example.

Example - Ten Fixtures One Axis

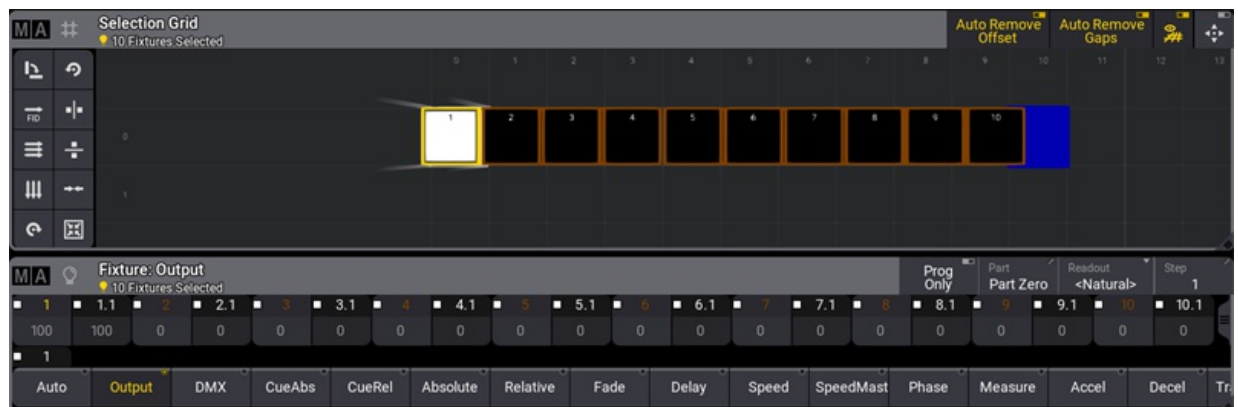
In this example, there are ten fixtures (1 through 10).

They are selected from 1 to 10 without any specific grid information. Highlight is activated.

Since there is no grid information, the fixtures are only on one axis - the X-axis.

Press **Next** to make the MAtricks X value 0.

It looks like this in the Selection Grid and Fixture Sheet (Channel SheetMode and Output layer selected) window:



Ten fixtures with MAtricks X at 0 - No Width

Now tap **+** three times in the XWidth in the MAtricks window to set the value to three.

Now it looks like this:

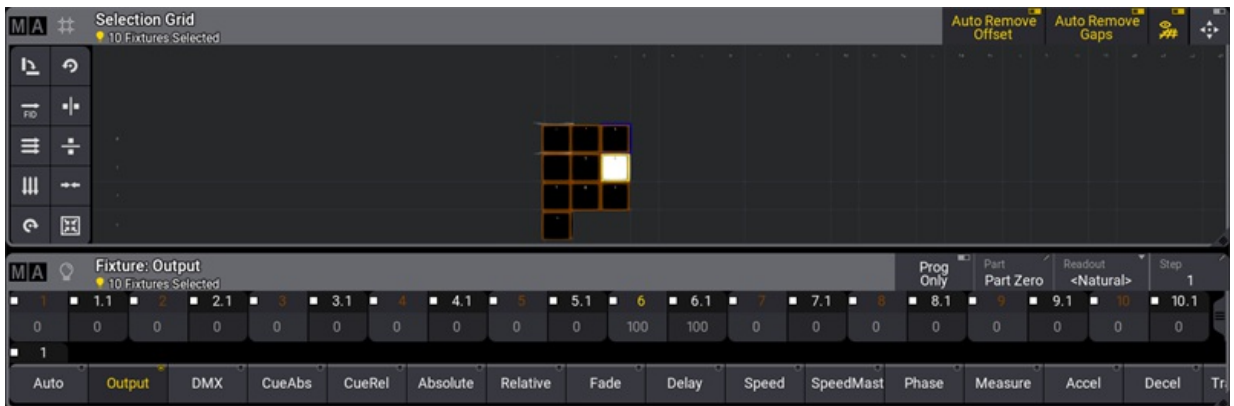


Ten fixtures with MAtricks X at 0 - XWidth at 3

All fixtures in the first X column are now highlighted.

This arrangement makes it possible to select a specific fixture or columns and rows of fixtures using a combination of X and Y values.

Highlighting fixture 6 in the current arrangement can be achieved by having X = 2 and Y = 1. It looks like this:



Ten fixtures with MAtricks X at 2, Y at 1, and XWidth at 3

Try to play around with different X and Y values.

26.5. MAtricks Shuffle

Shuffle is used to shuffle the selection order of the current fixture selection.

The shuffle function is described in more detail in the [MAtricks and Shuffle topic](#).

This example uses the **MAtricks window**, the **Selection Grid window**, the **Fixture Sheet window**, the **Group Pool**, the **Color Preset Pool**, and the **All 1 Preset Pool**. It is useful to have them visible on the screens.

This is an example of a shuffle on two axes.

The main setup is 20 fixtures with color mix, and four global color presets (Red, Green, Blue, and Magenta).

1. Fixture 101 to 105 is given the red color.
2. Fixture 106 to 110 is green.
3. Fixture 111 to 115 is blue.
4. Fixture 116 to 120 is magenta.
5. They are selected from 101 to 120.
6. Use the MAtricks tool to set XWidth to 5. This arranges the fixtures nicely in a 5 x 4 grid.
7. Store a group with this grid arrangement.
8. A dimmer range of 25% to 100% is applied.
9. This is stored in an All preset.

It looks like this:



Initial shuffle setup

Imagine this is how the fixtures are arranged in the real world, and this now needs to be randomly applied in both X and Y axes.

Tap **Shuffle** in the MAtricks tool until a desired pattern is achieved.



Notice in the fixture sheet that this does not change the actual output of the fixtures. It only changes how they are selected.

The values need to be reapplied to get the output to look like the random selection.

To apply the dimmer, press **At 2 5 Thru 1 0 0 Please**.

The Y value makes it easy to reapply the colors.

Tap **+** in the MATricks Y, so the value is 0. Now apply the red color preset. Tap **+** again to select the next row and apply the next color preset. Continue doing this for the last two rows.

Press **Clear** once and tap the group stored at point seven in the initial setup.

The result could look something like this:



Fixture grid after the shuffle

26.6. MAtricks Transform

Transform is a setting in the MAtricks window. It is found in the **Invert Options** at the bottom of the MAtricks window.

If Transform is set to Mirror, the values will be mirrored depending on the other MAtricks settings. For example, Blocks, Groups, and Wings.

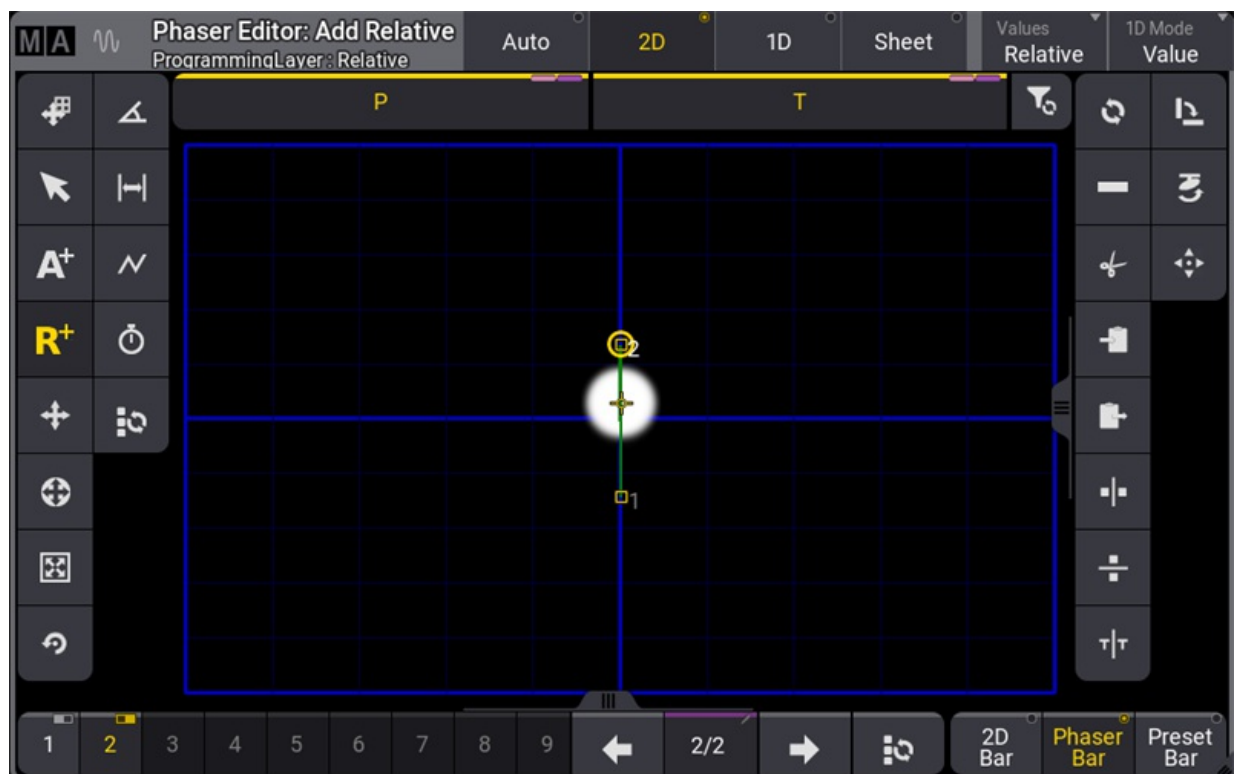
Transform allows to:

- Quickly create symmetrical objects.
- Keep a symmetrical impression on an odd fixture selection.



Example

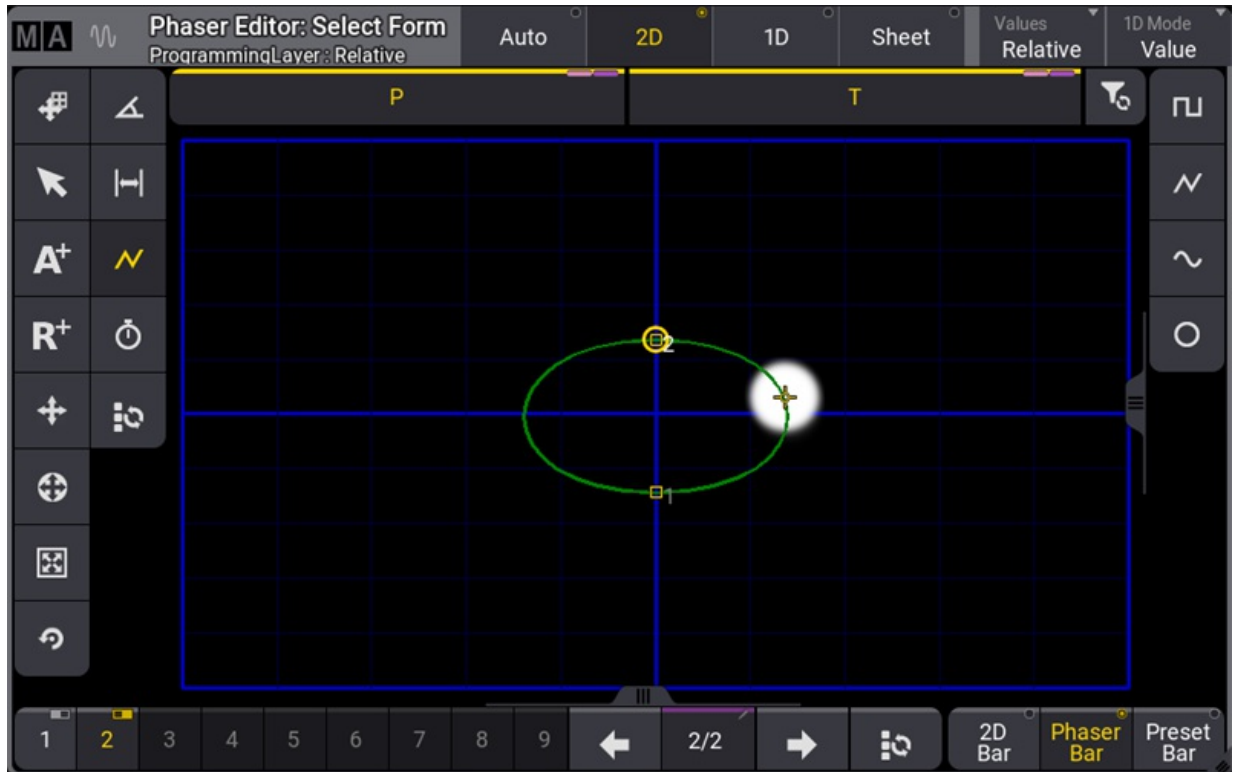
How to set Transform in a simple example:

1. Select some fixtures and set the dimmer to full.
2. Open a **Phaser Editor** window using the Add Window pop-up. Or tap the **Phaser** button in the Encoder Bar.
3. Tap the **R⁺** button in the left tool button bar of the phaser window.
4. Set two points, "1" and "2," like in the image below:



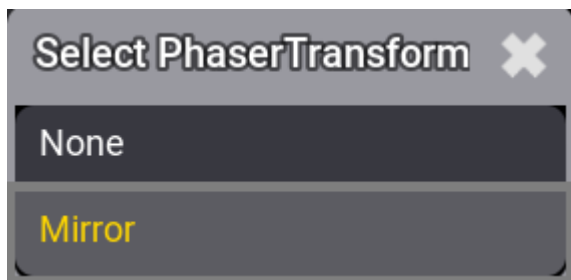
Phaser with relative position.

4. To change the phaser to a circle form, select  in the left button toolbar and then select  on the right button toolbar in the phaser window.



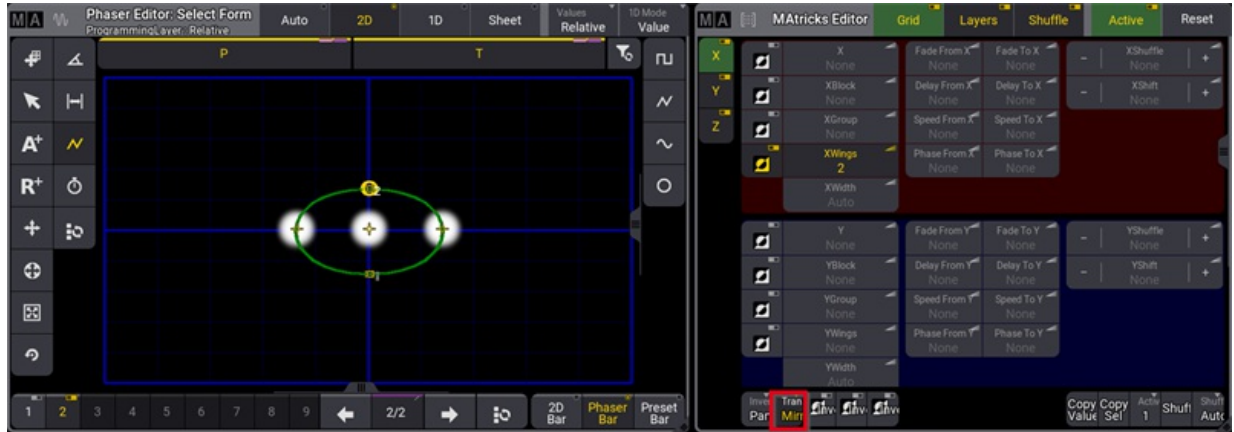
Phaser with circle form

5. Open the **MAticks** window and set **XWings** to 2.
6. To set Transform to Mirror, tap and hold **Transform**. A pop-up opens. Tap **Mirror**.



Transform Mirror pop-up

7. The result is shown in the image below:



A summary of the steps above

If **Transform** is set to Mirror:

- The font color of the **Transform** button is yellow.
- **InvertStyle** changes to Pan.

If **Transform** is set to None:

- **Inverts** are switched off.



Important:

Setting the MAtricks, for example, **XWings**, to None and **Transform** to Mirror, disables **Invert** for this setting.



If there are only values for one step active, the values will be mirrored in this step.



As soon as there are attributes with two or more steps, Mirror will only mirror the values of the multi-step phasers.

Cues and Preset along with Transform Mirror

Transform - Mirror works by transforming values of the inverted fixtures to create symmetry. This requires changing the original values in the Programmer to hard values.

Restriction:
Storing the transformed values directly to a Cue will not store the preset reference, if the transformed values are not part of the used phaser preset.

Hint:
To avoid losing references, store the transformations with MAticks to a new Preset object.

With the combination of transformed values with MAticks, the reference links will be automatically saved to the preset object. Calling this Preset with transformation MAticks included will keep the links when storing it to a Cue.

Odd Fixture Selection along with Transform Mirror

The image below shows the 3D window when working with Wings and an odd number of fixtures. Fixtures are split into three groups, shown in the image below:

- 1. Selected fixtures are generally colored in yellow.
- 2. The center fixture is marked as an edge fixture. It is colored light green.
- 3. Selected Fixtures with enabled Transform Mirror setting are colored in green.
- 4. The MAticks XWings is set to 2.



Eleven selected fixtures in 3D with XWings active and transition set to mirror.



Important:

An edge fixture in an odd fixture selection does not follow all values to keep the symmetry.

Example: When creating a mirrored circle with the eleven selected fixtures, the edge fixture in the center will only tilt but not pan.



Hint:

For more information on system colors, see [System](#) topic.

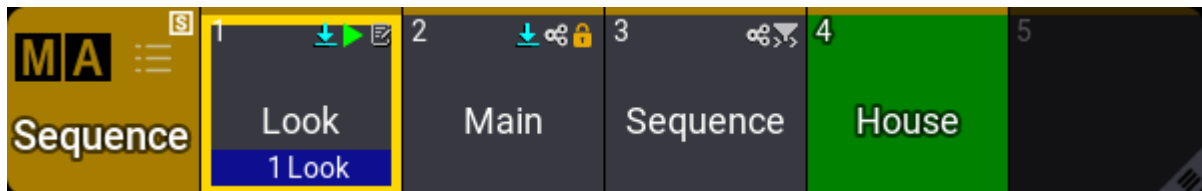
27. Cues and Sequences

Fixture values can be stored in **Presets** or **Cues**.

Cues are organized in **Sequences**.

Executors can control and playback sequences. The sequence is played back from the sequence pool. The executors are handles for the sequence.

The sequences are all in the **Sequence Pool**. This is created like any other window - see more in the [Add Window topic](#).



Sequence pool with sequence 1 running a cue

The sequence contains information about the cues and how to transition from one cue to another.

Cues can contain fixture values (sometimes called "hard values"), references to presets, and recipes. Read the [Store Cues topic](#) to learn about making cues.

All cues have a minimum of one step. If there is more than one step, then it is a **Phaser**. Follow the phaser link to read about multi-step cues.

Cues have **Cue Parts**. All values are actually stored in the parts. All cues automatically have part 0. Other parts can be created, but it is not necessary. Creating several parts allows separating values into different sub-cues. The parts can have different timing or properties, but parts are connected to the primary (parent) cue. This means that all the cue parts in a cue trigger together with the primary cue.

The cues in a sequence can be seen in a **Sequence Sheet**. Read more about looking at the sequence in the [Look at Cues and Sequences topic](#).

Sequences have a lot of different settings. Read the [Sequence Settings topic](#) for more information.

Default Action

A sequence pool has a default action for the pool objects. This action is performed when the sequence is tapped in the pool without a (relevant) keyword in the command line.

Learn more in the [Window Settings topic](#).

Reference Marker

If a sequence is referenced somewhere, it gets a small reference marker (↓). Referenced means that it is used or assigned somewhere. The [Info window](#) can show where an object is referenced.

Filters

The sequence can have a world or a filter assigned as an input filter and an output filter.

If there is an input filter, there is a small icon (↵) in the pool object. The input filter only allows the data in the world or filter to be stored in the sequence.

If an output filter is assigned, the pool object has a small icon (↵). The output filter allows only the fixtures and attributes in the world or filter to be played back from the sequence.

See the example image above. Sequence four is called "BlueLights". It has both an input and output filter assigned, and this is a combined filter icon (↵↵).

Read more about filtering sequences in the [Worlds and Filters section](#).

Shared Data

Two sequences can be linked and share the cue data. If a sequence share data with another sequence, then it has a small icon (↔) in the pool object. See sequences two and five in the example image above.

Shared data means that the cue content and the cue settings (for example, fade times) are the same. Changing one of the two sequences will also change the other sequence. For example, creating cues, changing cue content, or deleting cues.

The sequence settings can be different. Different executors can control the different linked sequences and playback different cues in the two (or more) linked sequences.

This can be useful if, during rehearsal, a video programmer and a lighting programmer want to work with one sequence but needs to be able to run different cues. The relevant input and output filters can be assigned to the two shared sequences. Two different executors can be assigned to control the two linked sequences. Now one sequence can playback and store video data, and the other can run and store light data. All values are stored in the same linked sequence. So when the show is ready, an operator can remove the filters and just playback one sequence with all the data.



27.1. What is Tracking

Tracking is the principle of storing only the **changes** in the cues.

If a fixture is turned On in blue color, it will stay like this until it is told to change. It does not matter how many cues there are between being told to turn On and the cue where it is told to turn Off.

Tracking is the principle that once a parameter has a value, it stays there until it is told to go somewhere else or released from a sequence.

The following example explores the basic tracking principles. Some of the other basic related functions are described after the example. Two tracking principles have their own topics: **Tracking Distance** and **Break**.

Example

Fixture number 1 is stored in blue color, at 100% dimmer value in cue number 1. If there are 6 cues and fixture number 1 does not get any new information, it will stay at 100% in blue color in all 6 cues.

The best way to see this information is by turning the **Sequence Sheet** into tracking sheet mode. This can be activating the **Track Sheet** setting either in the title bar or the window settings.

M A Sequence 2							Cue Only	Track Sheet		
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	R	G	B
	0		CueZero							
	1	0	Blue	<Yes>			100	0	0	100
	2	0	Cue 2				100	0	0	100
	3	0	Cue 3				100	0	0	100
	4	0	Cue 4				100	0	0	100
	5	0	Cue 5				100	0	0	100
	6	0	Cue 6				100	0	0	100
			OffCue	Yes						

A sequence with 6 cues - fixture 1 stored values in cue 1

Notice the difference in text color between the values in cue 1 and the others. The magenta text color indicates that the value is not stored in the cues, but is a tracked value from a previous cue.

Now change the fixture's color to red and set the dimmer to 100% again.

Store this in cues 4 **and** 5 (using the Merge option). Now it looks like this:

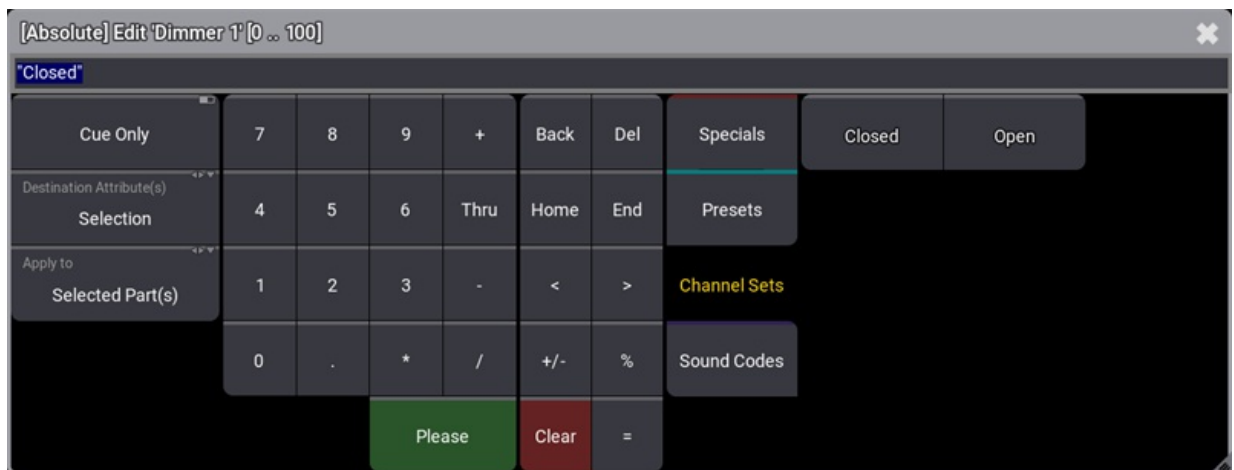


Sequence 2							Cue Only	Track Sheet		
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	R	G	B
	0		CueZero							
	1	0	Blue	<Yes>			100	0	0	100
	2	0	Cue 2				100	0	0	100
	3	0	Cue 3				100	0	0	100
	4	0	Red				100	100	0	0
	5	0	Cue 5				100	100	0	0
	6	0	Cue 6				100	100	0	0

A sequence with 6 cues - fixture 1 has a new color in cues 4 and 5 - blocked values

The text color of the dimmer value is now white in cues 4 and 5. All the color values are also white in cue 5. This indicates that the values are stored here but are the same as the tracked value. These are called "Blocked" values. This means that if any value is changed in any previous cue, it would still be 100% in cue number 4. It blocks the tracked value from cue 1 (the same value), and the value stored in cue 5 is tracked to the end.

Values can be edited directly in the tracking sheet. Edit the dimmer value in cue 3 by right-clicking with a mouse or the two-finger edit gesture on a touchscreen. This opens the **calculator**.



Calculator editor

Select **Channel Sets** and tap **Closed**.

This closes the calculator and changes the dimmer value to 0 in cue 3 in the sequence.



M A Sequence 2							Cue Only	Track Sheet		
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	R	G	B
	0		CueZero							
	1	0	Blue	<Yes>			100	0	0	100
	2	0	Cue 2				100	0	0	100
	3	0	BO				0	0	0	100
	4	0	Red				100	100	0	0
	5	0	Cue 5				100	100	0	0
	6	0	Cue 6				100	100	0	0

A sequence with 6 cues - fixture 1 has a new dimmer value in cue 3

The dimmer value is now green in cue 3. This indicates that the dimmer value is now stored with a lower value than the previous cue. Only dimmer values show this green indication for a lower value.

The dimmer value in cue 4 is now cyan, indicating a higher value than the previous cue.

The redundantly stored values in cue 5 can be removed from the cue by removing the stored values - also called "unlocking".

M A Sequence 2							Cue Only	Track Sheet		
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	R	G	B
	0		CueZero							
	1	0	Blue	<Yes>			100	0	0	100
	2	0	Cue 2				100	0	0	100
	3	0	BO				0	0	0	100
	4	0	Red				100	100	0	0
	5	0	Cue 5				100	100	0	0
	6	0	Cue 6				100	100	0	0

A sequence with 6 cues - unblocked sequence

There are several ways to do this.

One option is to have the values for fixture number 1 as active values in the programmer (it does not matter what the values are) and store cue 5 with the "Remove" option. This would remove the values from the cue.

A second option is using a command like this:

```
MA User name[Fixture]> Unblock Sequence 2
```

This removes all redundant blocked values from the entire sequence number 2.

A third option is to unblock cue 5. This can be done by adding "cue 5" to the command above or editing a value in cue 5. It does not matter what value; all that is needed is the calculator opened with a value from cue 5.

On the left side of the calculator, change the **Destination Attributes(s)** value to **All Fixtures** and the **Apply to** to **Selected Part(s)**. Now tap **Specials** and then **Unblock**. This unblocks all attributes for all fixtures in the selected cue parts. The selection here is the selected cue when the calculator was opened.



Hint:

The Move In Black (MIB) function is visible in the tracking sheet if the MIB function is activated. It appears as values with deep-sea green background. Learn more about it in the [Move In Black topic](#).

Cue Zero

Cue Zero is an automatically created empty cue. Cue Zero can be modified or stored into using the cue name. Notice the example picture above where MIB is active. It has the MIB values automatically in CueZero.

Off Cue

The Off cue has timing information that is used when the sequence is turned Off. Release is active for the OffCue as a default.

Release

The sequence sheet has a **Release** column. Editing a field in this column changes the cell to **Yes**. This means that tracked values will be released from this cue and forward.

M A Sequence 2							Cue Only	Track Sheet		
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	R	G	B
	0		CueZero							
	1	0	Blue	<Yes>			100	0	0	100
	2	0	Cue 2				100	0	0	100
	3	0	BO				0	0	0	100
	4	0	Red				100	100	0	0
	5	0	Cue 5	Yes						
	6	0	Cue 6							
			OffCue	Yes						

A sequence with 6 cues - cues 5 and OffCue with release active

It makes a white horizontal line above the released cue. This indicates that tracked values are stopped, and if there are no other values in the sequence, then the attribute is released from the sequence - empty black cells.

When an attribute is released in the sequence, then it is the same as the sequence no longer sending any information to the attribute.

If a different sequence sends values to the attribute, these values are used. Sequence priority can be important in this case - read about priorities in the [Play back Cues topic](#).

If the attributes do not receive values from any sequence or the programmer, then values go back to the default values for the attributes.

It is also possible to store values using the **Release** option. This does not actually store the values but empties the relevant fields in the sequence tracking sheet.

Turning Tracking On or Off

In the [sequence settings](#), there is the option to turn On or Off the tracking function.

When tracking is Off, the tracked values are gone, and where the value was tracked, it will be released instead.

Looking at the sequence used in the example above, it would look like this without tracking:

M A Sequence 2							Cue Only	Track Sheet		
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	R	G	B
	0		CueZero							
	1	0	Blue	<Yes>			100	0	0	100
	2	0	Cue 2							
	3	0	BO				0			
	4	0	Red				100	100	0	0
	5	0	Cue 5							
	6	0	Cue 6							
			OffCue	Yes						

Sequence with Tracking turned Off

Fixtures 1 would only be On in cue 1 in blue. In cue 2 it would return the default values. In cue 4 it would turn On in red. In cue 5 the color returns to the default. The zero value stored in cue 3 is possibly redundant since it is often the default dimmer value.



27.1.1. Tracking Distance

Tracking and common tracking functions are described in the [What is Tracking topic](#).

Tracking values can have a distance. This can be changed in the **Tracking Distance** column in a [Sequence Sheet](#).

There are two types of tracking distances: Normal tracking tracks until a specific cue number. Delta tracks values to a cue number calculated by the cue number, where it is set, plus a number.

Both types release the attribute when the tracking distance ends. This will return the attribute to a previously tracked value or release it from the sequence. See the examples below.

A vertical white line next to the value displays the attributes affected and the number of cues the tracking distance is valid.

A tracking distance affects all values stored in the cue where the distance is set. The tracking distance value does not consider cue parts where the distance ends.

Tracking Using Cue Number

The tracking distance can be set to track to and include a specific cue number using the tracking distance with a normal number input. See the following example for an explanation.

Example

The starting point for this example is a simple sequence with nine cues and two fixtures dimmer value set to 50% in the first cue:



M A Sequence 3 'Tracking Distance'							Cue Only	Track Sheet
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	2 Dim
	0		CueZero					
	1	0	Cue 1	<Yes>			50	50
	2	0	Cue 2				50	50
	3	0	Cue 3				50	50
	4	0	Cue 4				50	50
	5	0	Cue 5				50	50
	6	0	Cue 6				50	50
	7	0	Cue 7				50	50
	8	0	Cue 8				50	50
	9	0	Cue 9				50	50
			OffCue	Yes				

Base sequence for Tracking Distance using cue number

Normal tracking rules dictate that the values in cue number 1 are tracked to the end where the **OffCue** releases the value - because **Release** is set to "Yes"

A dimmer value of 70% for fixture 1 is added to cue number 3.

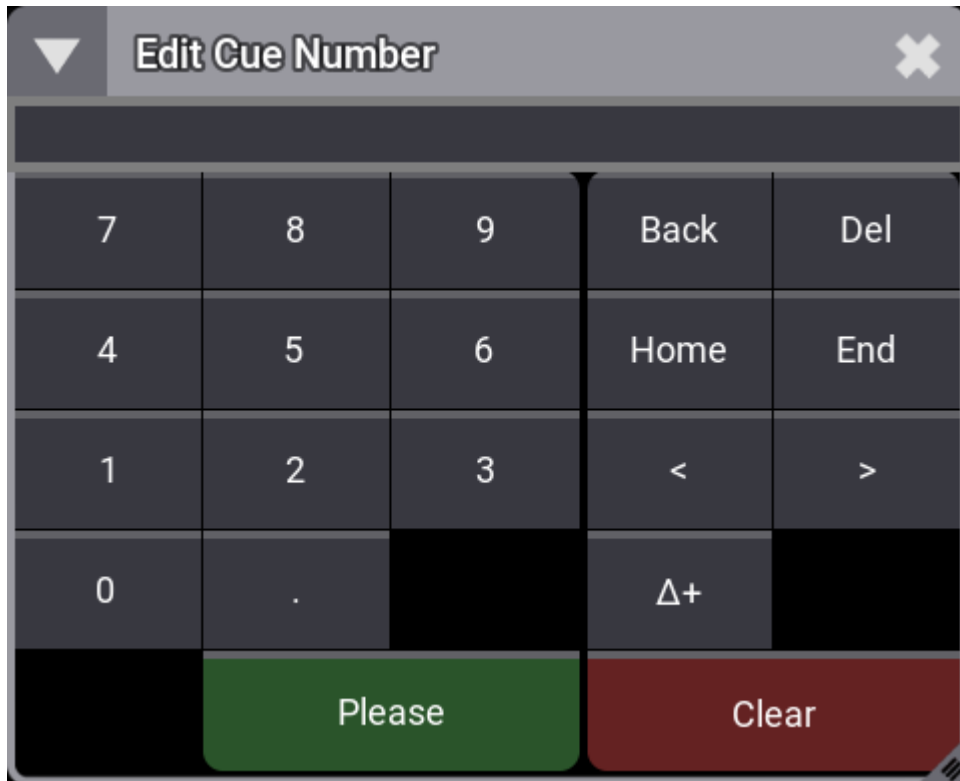


M A Sequence 3 'Tracking Distance'							Cue Only	Track Sheet
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	2 Dim
	0		CueZero					
	1	0	Cue 1	<Yes>			50	50
	2	0	Cue 2				50	50
	3	0	Cue 3				70	50
	4	0	Cue 4				70	50
	5	0	Cue 5				70	50
	6	0	Cue 6				70	50
	7	0	Cue 7				70	50
	8	0	Cue 8				70	50
	9	0	Cue 9				70	50
			OffCue	Yes				

Value added to cue number 3

A tracking distance is set by editing the Tracking Distance cell for cue number 3.

This opens the **Edit Cue Number** pop-up, where a number can be typed.



Edit Cue Number pop-up

The number typed is the cue number, after which the tracking should stop. The cue does not need to exist.

This is the result of the Tracking Distance set to a value of 5:



M A Sequence 3 'Tracking Distance'							Cue Only	Track Sheet
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	2 Dim
	0		CueZero					
	1	0	Cue 1	<Yes>			50	50
	2	0	Cue 2				50	50
	3	0	Cue 3			5	70	50
	4	0	Cue 4				70	50
	5	0	Cue 5				70	50
	6	0	Cue 6				50	50
	7	0	Cue 7				50	50
	8	0	Cue 8				50	50
	9	0	Cue 9				50	50
			OffCue	Yes				

Tracking Distance set using a cue number

The tracking stops between the existing cues numbers 5 and 6 - notice the white line indicating the tracking distance. It will track to and include cue number 5. The previous tracked value of 50% continues tracking after the distance.

Adding or removing cues in between does not change that it tracks to and including cue 5.

Renumbering cue number 3 also does not change the distance.



M A Sequence 3 'Tracking Distance'							Cue Only	Track Sheet
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	2 Dim
	0		CueZero					
	1	0	Cue 1	<Yes>			50	50
	2	0	Cue 2				50	50
	2.5	0	Cue 2.5			5	70	50
	4	0	Cue 4				70	50
	4.5	0	Cue 4.5				70	50
	5	0	Cue 5				70	50
	6	0	Cue 6				50	50
	7	0	Cue 7				50	50
	8	0	Cue 8				50	50
	9	0	Cue 9				50	50

Caption

Here cue number 3 is renumbered to 2.5, and cue 4.5 is added.

The tracking still ends after cue number 5.

Tracking Delta Distance

Setting a tracking distance with a delta number will track the value to a cue number mathematically higher than the cue where it is set. See the following example for a better understanding.

Example

The starting point for this example is the same sequence used in the previous example but cleaned up a bit:



M A Sequence 3 'Tracking Distance'							Cue Only	Track Sheet
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	2 Dim
	0		CueZero					
	1	0	Cue 1	<Yes>			50	50
	2	0	Cue 2				50	50
	3	0	Cue 3			5	70	50
	4	0	Cue 4				70	50
	5	0	Cue 5				70	50
	6	0	Cue 6				50	50
	7	0	Cue 7				50	50
	8	0	Cue 8				50	50
	9	0	Cue 9				50	50
			OffCue	Yes				

Base sequence

Now cue number 4 is updated with a dimmer value of 80% for fixture 2:



M A Sequence 3 'Tracking Distance'							Cue Only	Track Sheet
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	2 Dim
	0		CueZero					
	1	0	Cue 1	<Yes>			50	50
	2	0	Cue 2				50	50
	3	0	Cue 3			5	70	50
	4	0	Cue 4				70	80
	5	0	Cue 5				70	80
	6	0	Cue 6				50	80
	7	0	Cue 7				50	80
	8	0	Cue 8				50	80
	9	0	Cue 9				50	80
			OffCue	Yes				

Updated cue number 4

When the Tracking Distance cell for cue number 4 is edited, the **Edit Cue Number** pop-up appears again.

Notice the $\Delta+$ button. Use this to tap $\Delta+$ 2 . 5 Please .

The result is that the value of 80% is tracked from cue number 4 and forward to and including cue number 6.5 ($4 + 2.5 = 6.5$).



M A Sequence 3 'Tracking Distance'							Cue Only	Track Sheet
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	2 Dim
	0		CueZero					
	1	0	Cue 1	<Yes>			50	50
	2	0	Cue 2				50	50
	3	0	Cue 3			5	70	50
	4	0	Cue 4			$\Delta +2.5$	70	80
	5	0	Cue 5				70	80
	6	0	Cue 6				50	80
	7	0	Cue 7				50	50
	8	0	Cue 8				50	50
	9	0	Cue 9				50	50
			OffCue	Yes				

Delta Tracking Distance

Cue number 6.5 does not exist, so the current last cue for the distance is cue number 6.

Any cue added between cue number 3 and cue number 5.5 will have the 80% tracked value, but a cue number higher than 5.5 will have the 50% value.

Make sure the programmer is empty and store empty cues number 6.5 and 6.6 in the sequence.



M A Sequence 3 'Tracking Distance'							Cue Only	Track Sheet
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	2 Dim
	0		CueZero					
	1	0	Cue 1	<Yes>			50	50
	2	0	Cue 2				50	50
	3	0	Cue 3			5	70	50
	4	0	Cue 4			Δ +2.5	70	80
	5	0	Cue 5				70	80
	6	0	Cue 6				50	80
	6.5	0	Cue 6.5				50	80
	6.6	0	Cue 6.6				50	50
	7	0	Cue 7				50	50
	8	0	Cue 8				50	50

Added cues number 6.5 and 6.6

Now it can be seen that the tracking stops between cues number 6.5 and 6.6.

If cue number 4 is renumbered, this affects the distance. Renumber cue number 4 to number 3.1 by editing the cue number cell.



M A Sequence 3 'Tracking Distance'							Cue Only	Track Sheet
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	2 Dim
	0		CueZero					
	1	0	Cue 1	<Yes>			50	50
	2	0	Cue 2				50	50
	3	0	Cue 3			5	70	50
	3.1	0	Cue 3.1			$\Delta +2.5$	70	80
	5	0	Cue 5				70	80
	6	0	Cue 6				50	50
	6.5	0	Cue 6.5				50	50
	6.6	0	Cue 6.6				50	50
	7	0	Cue 7				50	50
	8	0	Cue 8				50	50

Updated cue number

Now the tracking stops between cues number 5 and 6. It tracks from cue number 3.1 and up to number 5.6 (3.1 + 2.5 = 5.6).

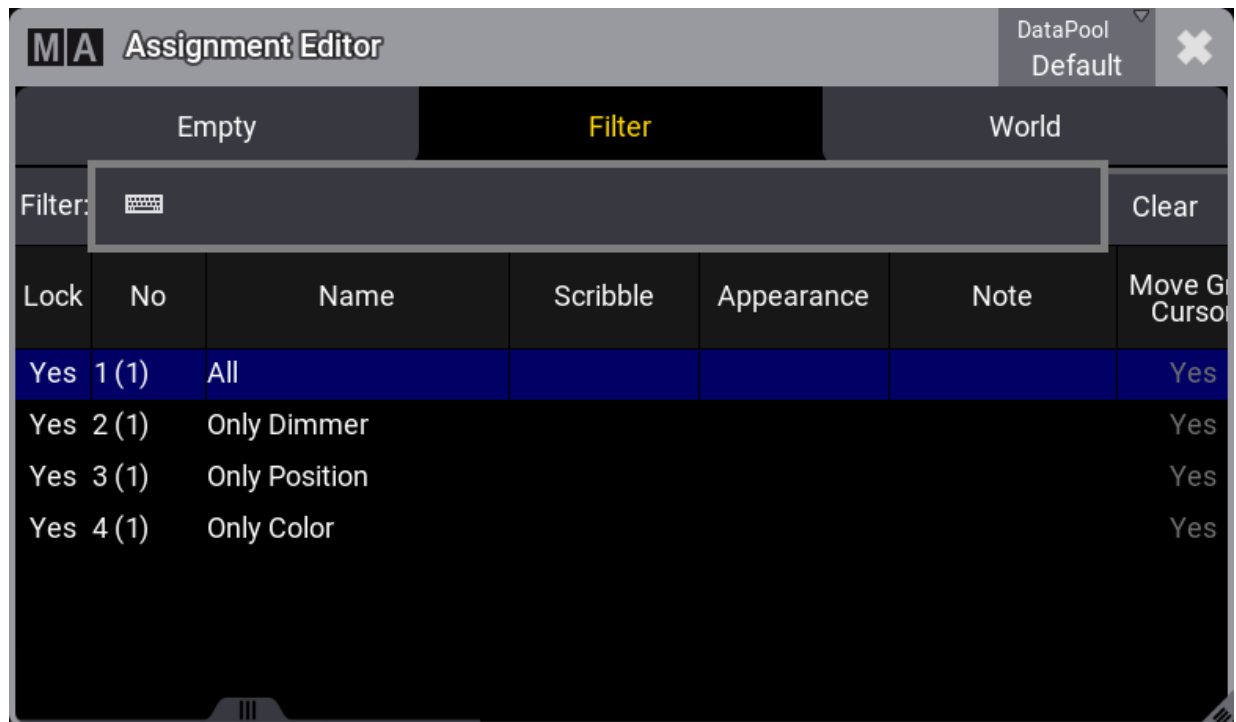
27.1.2. Break

Tracking and common tracking functions are described in the [What is Tracking topic](#).

Breaks in cues are a filtered version of the **Release** function. Where the release stops all tracking values, the break will stop tracking of attributes based on a selected filter or world.

The main difference between the release function and setting a break, filtering all attributes, is that the release will release currently tracked attributes from the sequence immediately when the release is activated. Setting a break does not affect existing stored and tracking values. An existing break can only affect tracking values added or changed that pass the cues where the break is already set - if the selected filter or world includes the attribute.

Editing the **Break** cell in a cue opens an **Assignment Editor** pop-up.



Assignment Editor pop-up

This has three tabs: Empty, Filter, and World.

Empty will remove any break setting. Filter and World allow the selection of an existing filter or world. Selecting one of these will stop tracking changes or additions, passing the cue where the break is set based on the content of the selected filter or world. Learn more about filters and worlds in the [Worlds and Filter section](#).

Follow the example below for a better understanding of the break function.



Example

The base sequence for this example looks like this:

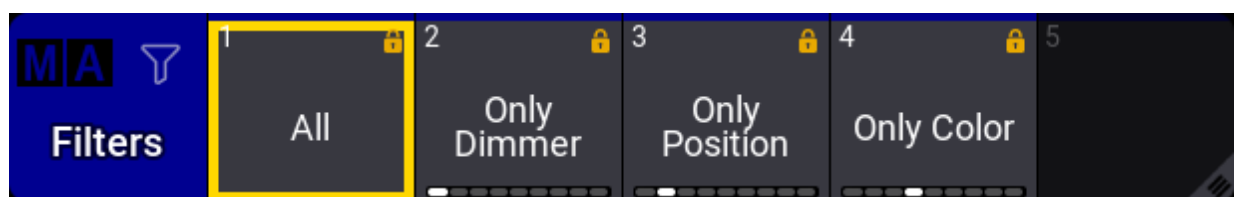
Sequence 4 'Tracking Break'							Cue Only	Track Sheet				
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	1 P	1 T	1 R	1 G	1 B
	0		CueZero									
	1	0	Preset Scene 1	<Yes>			0	2.3 Center			4.1 White	
	2	0	Cue 2				.	2.3 Center			4.1 White	
	3	0	Cue 3				.	2.3 Center			4.1 White	
	4	0	Fade Out				.	2.3 Center			4.1 White	
	5	0	Cue 5				.	2.3 Center			4.1 White	
	6	0	Cue 6				.	2.3 Center			4.1 White	
	7	0	Preset Scene 2				.	2.3 Center			4.1 White	
	8	0	Cue 8				.	2.3 Center			4.1 White	
	9	0	Cue 9				.	2.3 Center			4.1 White	
			OffCue	Yes								

Base sequence for Break example

Here we have nine cues. Cue number 1 has some stored values (Dimmer, Position, and Color) for fixture 1.

The values stored in cue number 1 are tracked from 1 to the end. The values are released in the **OffCue** because there is a **Yes** in the **Release** column.

Two filters in the **Filters** pool will be used: "All" and "Only Dimmer".



Filter pool with filter objects

The filters are one filter that contains all attributes and layers and one that only contains the dimmer attributes and all layers.

In this scenario, it is known that cue number 4 is a fade out to black, and cue number 7 is used to pre-set fixtures for the second scene in the show.

The break function can protect these two cues from tracking value changes.

Before moving any further with the example, let us explore the release function by editing the **Release** cell for cue 7, so it changes to "Yes".



Sequence 4 'Tracking Break'							Cue Only			Track Sheet		
Lock	No	Part	Name	Release	Break	Tracking Distance	1	1	1	1	1	1
							Dim	P	T	R	G	B
	0		CueZero									
	1	0	Preset Scene 1	<Yes>			0	2.3 Center			4.1 White	
	2	0	Cue 2				.	2.3 Center			4.1 White	
	3	0	Cue 3				.	2.3 Center			4.1 White	
	4	0	Fade Out				.	2.3 Center			4.1 White	
	5	0	Cue 5				.	2.3 Center			4.1 White	
	6	0	Cue 6				.	2.3 Center			4.1 White	
	7	0	Preset Scene 2	Yes								
	8	0	Cue 8				.	2.3 Center			4.1 White	
	9	0	Cue 9				.	2.3 Center			4.1 White	
			OffCue	Yes								

Release is out of the scope of this topic (It is explained more in the **What is Tracking topic**), but it is included here to show that a release stops all future and existing tracking. The tracking values are released from the sequence.

Release is not the goal for cue 7. Remove the "Yes" in the **Release** column again.

Editing the **Break** cell for cue number 7 opens the **Assignment Editor**. Select **Filter** and then tap **All** to select it.

Now the sequence looks like this:

Sequence 4 'Tracking Break'							Cue Only			Track Sheet		
Lock	No	Part	Name	Release	Break	Tracking Distance	1	1	1	1	1	1
							Dim	P	T	R	G	B
	0		CueZero									
	1	0	Preset Scene 1	<Yes>			0	2.3 Center			4.1 White	
	2	0	Cue 2				.	2.3 Center			4.1 White	
	3	0	Cue 3				.	2.3 Center			4.1 White	
	4	0	Fade Out				.	2.3 Center			4.1 White	
	5	0	Cue 5				.	2.3 Center			4.1 White	
	6	0	Cue 6				.	2.3 Center			4.1 White	
	7	0	Preset Scene 2		'All' Filter 1		.	2.3 Center			4.1 White	
	8	0	Cue 8				.	2.3 Center			4.1 White	
	9	0	Cue 9				.	2.3 Center			4.1 White	
			OffCue	Yes								

Break is set for cue 7

Notice the difference between this and the release. The existing tracking values are not released but are protected from future changes.



A break adds a white line to indicate the tracking break. If the break affects all attributes - typically by using Filter 1 or world 1 - it will extend across the entire row (just like the release).

Cue 4 is a fade out. The current dimmer value of 0% can be protected by adding a second break in cue 4, but this time using the "Only Dimmer" filter:

Sequence 4 'Tracking Break'										Cue Only		Track Sheet		
Lock	No	Part	Name	Release	Break	Tracking Distance	1			1				
							Dim	P	T	R	G	B		
	0		CueZero											
	1	0	Preset Scene 1	<Yes>			0		2.3 Center			4.1 White		
	2	0	Cue 2				.		2.3 Center			4.1 White		
	3	0	Cue 3				.		2.3 Center			4.1 White		
	4	0	Fade Out		'Only Dimmer' Filter		.		2.3 Center			4.1 White		
	5	0	Cue 5				.		2.3 Center			4.1 White		
	6	0	Cue 6				.		2.3 Center			4.1 White		
	7	0	Preset Scene 2		'All' Filter 1		.		2.3 Center			4.1 White		
	8	0	Cue 8				.		2.3 Center			4.1 White		
	9	0	Cue 9				.		2.3 Center			4.1 White		
			OffCue	Yes										

A break is set for cue 4

The break in cue 4 looks a bit different. The white line is black in the attribute columns. This indicates that some attributes are filtered. The Feature Group Indicator Bar is visible at the bottom of the break cell. It shows the features active in the filter.



Now all the breaks are set, and the values can be changed. A new dimmer, position, and color values for fixture 1 are stored in cue number 2.

Sequence 4 'Tracking Break'							Cue Only	Track Sheet				
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	1 P	1 T	1 R	1 G	1 B
	0		CueZero									
	1	0	Preset Scene 1	<Yes>			0	2.3 Center			4.1 White	
	2	0	[Fan Out /Blue]				75	2.2 Fan Out			4.4 Blue	
	3	0	Cue 3				75	2.2 Fan Out			4.4 Blue	
	4	0	Fade Out		'Only Dimmer' Filter		0	2.2 Fan Out			4.4 Blue	
	5	0	Cue 5				.	2.2 Fan Out			4.4 Blue	
	6	0	Cue 6				.	2.2 Fan Out			4.4 Blue	
	7	0	Preset Scene 2		'All' Filter 1		.	2.3 Center			4.1 White	
	8	0	Cue 8				.	2.3 Center			4.1 White	
	9	0	Cue 9				.	2.3 Center			4.1 White	
			OffCue	Yes								

Result

The result is the dimmer, and only the dimmer, value being returned to 0% (the previous tracked value) in cue number 4 because the filter in the break stopped the newly added values tracking past the cue.

All the values in cue 7 were protected from changes by the filter in the break, and the old tracking values are added (if needed) in cue number 7.



27.2. Sequence Sheet

The different sequences are best seen in the [Sequence Pool](#).

The best way to see the cues inside a sequence is the [Sequence Sheet](#):

Lock	No	Part	Name	Trig	Duration	Cue	Snap Delay	Release	Break	Assert	Allow Duplicates	Tracking Distance	Sync Morph Transition	Command Delay	Command	Note
	0		CueZero		0	0	0	0								
	1	0	Intro	Go	0	1	1	0						Linear	0	
	2	0	Limo Pickup	Go	0	3	3	0		Only Dimm	Yes			Linear	0	
	2	0	[Straight Front			3	0	3						Linear	0	
	2.5	0	Limo Ride	Follow	+2	2	2	0						Linear	0	
	4	0	Nakatomi Plaza	Go	0	5	5	0	Yes					Linear	0	
	4.5	0	Christmas Party	Go	0	1	1	0		Yes			Slow	0		
	1	0	G1 Indiv Dim			4	0	0						Linear	0	
	5	0	Hollys Office	Go	0	3	3	0						Linear	0	
	6	0	End of First Part	Go	0	2	2	0	Yes					Linear	0	
			OffCue			0	0	0	Yes	Yes				Linear	0	

Sequence sheet for a sequence called "Main" - Cue 2 is active

The sheet shows the cues and cue parts in rows. The different cue settings are in columns. Read below for a description of the different columns.

The purpose of this sheet is to see the cues in a sequence. It also shows the active cue with a green frame. Cue fades and delays are visualized with moving bars while the fades are running.

The sheet can be created as a window in a view using the [Add Window pop-up](#), or displayed as a temporary pop-up by editing a sequence pool object.

The sheet can show a lot of different markers and colors - read more about all these in the [Markers topic](#).

The sheet can display a value tracking section. In this section, all the stored attribute values are displayed. This can be a very handy tool to see the flow of an attribute through the cues. These values can also be edited directly in the sheet.

Most other fields in the sheet can be edited directly. This can affect the look of the show. For instance, the cue fade and delay times are stored when the cue is created. The default timing is used if nothing else is defined. The cue timing can be edited in this sheet.

Tapping this sheet to give it focus changes the encoder toolbar to make it easy to edit the cue timings. Read more about the toolbar [below](#).

When a trigger type is set for the OffCue of a sequence, Wrap Around will be disabled. Enabling Wrap Around for a sequence removes the trigger type for the OffCue.

Title Bar

The left side of the title bar has the MA logo. Tap this to open the settings for the sheet. Read about them [below](#).



Next to the logo are the sequence number and name. Information about the active world (if different than world 1) is displayed below the Sequence number and name. Next to the world information is the number and name of the currently active cue.

Some buttons can be on the right side of the title bar. This can be defined by editing the title bar in the window settings. Learn more in the [Title Bar Configuration topic](#).

Main Part of the Sheet

The main part of the sheet is below the title bar. Here is the sheet with rows and columns.

There is a sequence sheet setting that defines how the cue timing is displayed in the sheet. The option is called **Condensed Timing**. The cue fade and delay are actually four different times because there are both fade and delay for values fading in (or up) and for dimmer values fading down. The condensed view shows this in two columns where the time can be separated by a slash (/). The value on the left is the InFade, the value on the right is the OutFade. The list below shows the uncondensed timing columns. Read more in the [Cue Timing topic](#).

This is a short description of each of the possible columns in the sheet:

- **Lock:**
Changing the value to **Yes** in this cell locks the cue from being edited.
- **No:**
This is the cue number.
- **Part:**
This shows the cue part number.
- **Name:**
This is the cue (part) name. If the cue contains part cues, an arrow allows them to fold and unfold the parts. In the example above, there are cue parts in cues 2 and 4.5. It is unfolded so the parts can be seen.
- **Release:**
Changing the value to **Yes** in this cell makes the cue release tracked values. Learn more in the [What is Tracking topic](#).
- **Break:**
A break blocks new values of attributes being tracked. Editing the cell opens an **Assignment Editor** pop-up where a filter or world can be selected. The selected filter or world defines what is blocked by the break. Learn more in the [Break topic](#).
- **Assert:**
Changing the value to **Yes** in this cell makes the cue assert tracked values.
- **Allow Duplicates:**
If several parts of the same cue are to contain values for the same attributes, use **Allow Duplicates** to enable this function. Absolute and relative values in multiple parts will use the value with the highest cue part number.



- **Trig Type:**

There are five different trigger types. Editing this cell opens a small select pop-up with the five different options:

 - **Go:**

The cue needs a Go command to be triggered.
 - **Time:**

The cue is triggered a set time after the previous cue is triggered. The time is set in the **Trig Time** column.
 - **Follow:**

A follow cue is triggered when the previous is completely done with the cue **Duration** (which includes all individual timing).
 - **Sound:**

This will trigger the cue using a sound as the trigger. Choosing one of 22 different frequency areas in the Trig Sound column is possible. Learn more about sound input in the [Sound Window topic](#).
 - **BPM:**

This will trigger the cue using the beats in the sound input. This can become useful with several cues triggered by the BPM (beats per minute).
- **Trig Time:**

The values stored here are only used if the trigger is **Time**.
If the trigger is **Time**, then the time in the cell will be used. The time starts counting down when the previous cue is triggered.
- **Trig Sound:**

This setting defines the sound used to trigger the cue when the **Trig Type** is sound.
- **Tracking Distance:**

The tracking distance sets how many cues a value should track. If the cell is empty, it tracks until changed. Read more about tracking distance in the [What is Tracking topic](#).
- **Duration:**

This is the overall cue time transition time. It is a combination of the longest fade time and any delays. This is the time used with the Follow trigger. The cell cannot be edited. It always shows the complete transition time.
- **Sync:**

Synchronizes the fixtures of the phaser. For example, if fixtures join the already running phaser, they will be synchronized with the fixtures already running. Learn more about sync in the [Phaser topic](#).
- **Morph:**

If the property is enabled and the phase of the fixtures changes from one cue to the next, they will stay on track and morph into their new phase value. If the option is disabled, the fixtures will take the direct way to their new position in the phase.
- **CueIn Fade:**

This is the fade time for all non-snap attributes and dimmer values that increase in value.
- **CueIn Delay:**

This is the delay or wait time between the trigger, and the actual cue in fade begins.
- **CueOut Fade:**

This is the fade time for dimmer values that goes down in value. The default values for this are the same as the CueIn Fade time - it is linked to the cue in fade with the **None** value.



- **CueOut Delay:**
This is the delay for the **Out Fade** (only dimmer values). Its default value is the same as the CueIn Delay value.
- **Snap Delay:**
This can be used to control when "snap" attributes change values.
- **Transition:**
This can be used to select a transition path for the fade. The different options are described in the [Cue Timing topic](#).
- **"Preset type" Fade:**
Each preset type has columns called the name of the preset type followed by "Fade". This uses the cue in fade as a default. It can be changed to give all values in the preset type a different fade time. All fixtures with new values in this preset type will use this timing for the attributes in the preset type.
- **"Preset type" Delay:**
Each preset type has columns called the name of the preset type followed by "Delay". This uses the cue in delay as a default. It can be changed to give all values in the preset type a different delay time. All fixtures with new values in this preset type will use this timing.
- **Command Delay:**
This will add a delay between the triggering of the cue and the execution of the command. See **Command** just below.
- **Command:**
Commands (like the ones written in the command line input) can be written in the cues. When the cue is triggered, they are executed on the GlobalMaster, IdleMaster, or Standalone station.
- **Note:**
This is a multiline text field where a note can be added to the cue. Learn more in the [Notes topic](#).
- **MIB Preference:**
This defines whether the cue is good for the MIB function. Read more in the [Move In Black topic](#).
- **MIB Mode:**
This sets the MIB mode. This can only be edited if MIB is possible. Read more in the [Move In Black topic](#).
- **MIB Target:**
Defines a cue where the MIB should be performed if possible. This can only be edited if MIB is possible. Read more in the [Move In Black topic](#).
- **MIB MultiStep:**
Defines what should happen with phasers in the MIB. This can only be edited if MIB is possible. Read more in the [Move In Black topic](#).
- **MIB Fade:**
Sets the MIB fade time. This can only be edited if MIB is possible. Read more in the [Move In Black topic](#).
- **MIB Delay:**
Sets the MIB delay time. This can only be edited if MIB is possible. Read more in the [Move In Black topic](#).
- **Indiv Fade:**
This is "Individual Fade". It displays the time for attributes that have individual stored fade times. This cell cannot be edited.
- **Indiv Delay:**
This is "Individual Delay". It displays the time for attributes that have individual stored delay times. This cell cannot be edited.



- **Indiv Duration:**

This is "Individual Duration". It displays the overall time for attributes that have individual stored fade and delay times. This cell cannot be edited.

- **Speed Master:**

A Speed Master can be assigned to the cue or cue part by editing this cell. The speed of the cue or cue part is only controlled by the assigned speed master. A speed master assigned to the sequence has a lower priority and does not influence a cue or cue part with a different assigned speed master.

- **Speed Scale:**

This can be used to scale the speed of the cue or cue part. A speed scale assigned to a cue or cue part has a higher priority than a speed scale assigned to the sequence.

- **Appearance:**

An Appearance can be assigned to the cue or cue part. It is connected with a sequence sheet setting called **CuePart Appearance**, which defines how the appearance is displayed. Read more [below](#).

All cells with a light or dark gray background color can be edited, and the field's value can be changed. Fields with a black background cannot be edited.

Track Sheet Mode

The sequence sheet can be in **Track Sheet** mode. This can be changed in the window settings - and the setting can be a button in the title bar.

Lock	No	Part	Name	Release	Break	Tracking Distance	Dim	Show Steps	Track Sheet	Settings	Auto Scroll	Link Type
	0		CueZero				3	3			3	
	1	0	Intro	<Yes>			20	P	T	R	G	B
	2	0	Limo Pickup		'Only Dimm		50					
		2	[Straight Front				50		2.1 Stra2.1 Straight			
	2.5	0	Limo Ride				0		2.1 Stra2.1 Straight			
	4	0	Nakatomi Plaza	Yes					2.2 Fan2.2 Fan Out			
	4.5	0	Christmas Party						2.2 Fan2.2 Fan Out	4.4 Blue	4.4 Blue	4.4 Blue
		1	G1 Indiv Dim				50		2.2 Fan2.2 Fan Out	4.4 Blue	4.4 Blue	4.4 Blue
	5	0	Hollis Office				0		2.2 Fan2.2 Fan Out	4.4 Blue	4.4 Blue	4.4 Blue
	6	0	End of First Part	Yes								
			OffCue	Yes								

Sequence Sheet in Track Sheet mode

The default for this mode is to hide most of the normal columns and show attribute data in columns instead.

The attribute values and markers have different colors indicating different statuses like the tracking status. Read about the colors and markers in the [Colors topics](#). Read about tracking in the [What is Tracking topic](#).



The values can be edited in the sheet. The **Cue Only** setting can be activated in the settings, and it can be a button in the title bar that appears when **Track Sheet** is On. This makes edited values follow the **cue only rules**.

When cues with multiple parts are expanded to show all the parts, it is easy to see exactly what parts have stored which values. When the cues are collapsed only to show one row, the values from the parts are shown in this row with a small text in the lower right corner telling what part the values come from.

No	Part	▶	Name	Release	Break	Tracking Distance	3	3	
							Dim	P	T
0			CueZero						
1	0		Intro	<Yes>			20		
2	0	▶	Limo Pickup		'Only Dimm'		50	2.1	Straight Front <small>Part 2</small>

Collapsed cue with multiple parts

This example is the same cue 2 as the image above. The only difference is that the cue is collapsed to only show one row. Notice the small text showing that the pan and tilt values are from part 2.

The **Layer Toolbar** can be turned On in the settings. This can be useful when there is a desire to edit or look at values in other layers.

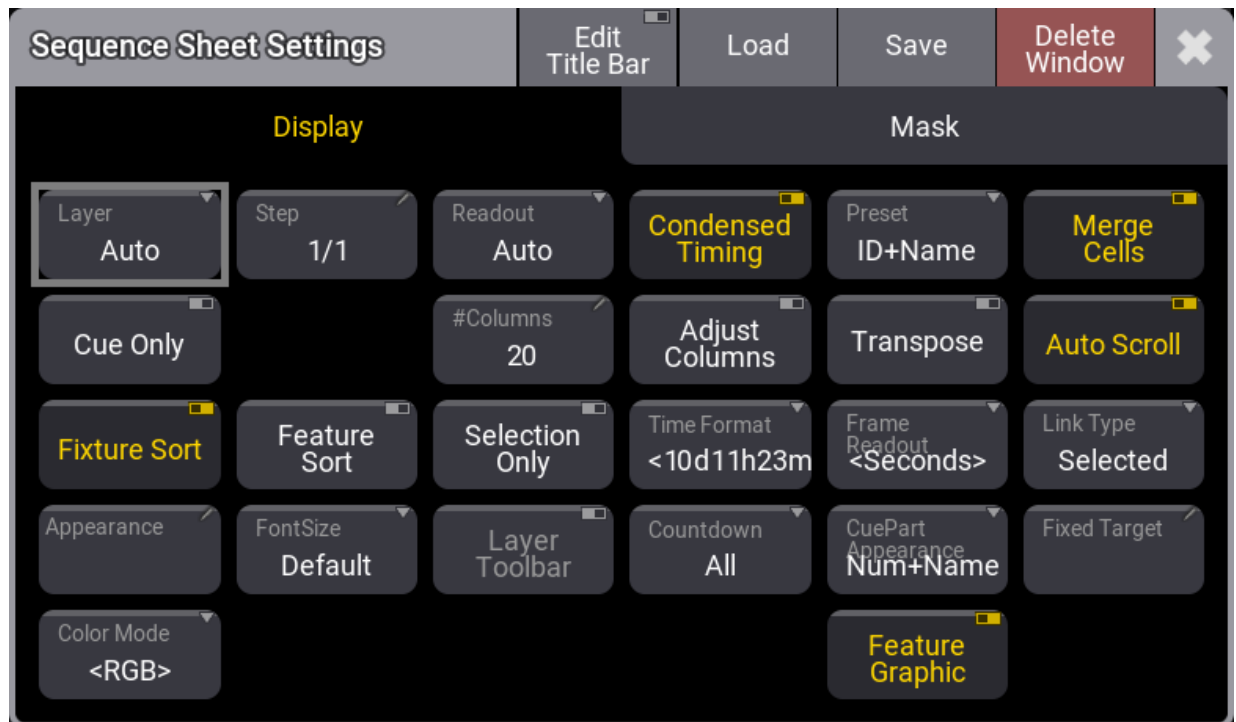
Editing a value opens the Calculator, where a new value can be selected from Presets, Channel Sets, Specials, or simply typed.

The **calculator** has some special functions in the track sheet that defines what is changed, and there are special buttons that give access to block, unblock, and extract presets.

Extract presets will remove the link to a preset and store the current preset values directly in the cue.

Sequence Sheet Settings

The sheets have a lot of settings. They are accessed by tapping the MA logo in the upper left corner of the window.



Sequence Sheet Settings - Display tab

Some of them are general settings that are shared with other windows. Read about them in the [Window Settings topic](#).

There are two tabs in the settings: **Display** and **Mask**.

This is a description of the display settings unique to the sequence sheet.

- **Auto Scroll:**
This On/Off button activates the auto-scrolling function. This is the same setting as the button in the title bar. Read more in the description above.
- **Condensed Timing:**
This toggles if the timing columns are displayed condensed or if all four timing columns are visible. Read about the main part of the sheet above.
- **Countdown:**
A cue timing countdown can be displayed while the fade is running. This setting has three options:
 - **Off:**
There is no countdown in any of the timing columns. They always display the set times.
 - **Duration:**
The duration column displays a countdown while the cue transition is running.
 - **All:**
The duration and cue timing columns display a countdown while the cue transition is running.



- **CuePart Appearance:**

This defines how the cue part appearance is displayed in the sheet. The options are:

- **Off:**
Cue part appearance is not displayed.
- **Number:**
The appearance is only shown on the cue number column.
- **Num+Name:**
The appearance is displayed in the number and name columns.
- **All:**
The appearance is displayed on all columns.

- **Selection Only:**

This On/Off button is a mask function that hides fixtures not currently selected in the programmer. This is valid when the **Track Sheet** mode is On.

- **Settings:**

This is only visible when editing the Title Bar. This opens the sequence settings. Here it is possible to change different settings about the sequence. Read more in the [Sequence Settings topic](#).

The mask settings unique for the sequence sheets are:

- **Command:**

This On/Off button shows or hides the group of Command columns.

- **Cue Settings:**

This On/Off button shows or hides the cue settings columns.

- **Cue Timing:**

This On/Off button shows or hides the cue timing columns.

- **Filter:**

An existing filter can be chosen to filter the content of the sheet.

- **Filter Toolbar:**

This setting show or hide a filter toolbar that can be used to filter the content of the sheet.

- **MIB Settings:**

This On/Off button shows or hides the MIB columns.

- **Note:**

This On/Off button shows or hides the Note column.

- **Preset Timing:**

This On/Off button shows or hides the preset timing columns.

- **Show Notes:**

This On/Off button shows or hides the notes for the selected cue at the bottom of the Sequence sheet. Read more about it in the [Notes topic](#).

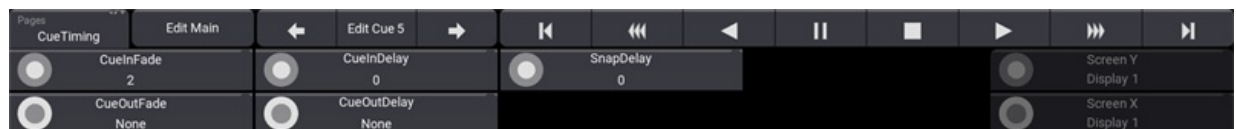
- **Show Recipes:**

This On/Off button shows or hides the cue recipes at the bottom of the Sequence sheet. Read more about it in the [Cue Recipes topic](#).

- **Show Steps:**
This On/Off button shows or hides the cue steps in the Sequence sheet. This is useful with **Track Sheet On**.
- **Track Sheet:**
This On/Off button shows or hides the values and tracking information for each attribute in the sheet.

Sequence Edit Toolbar

The encoder toolbar changes when the sequence sheet has focus. The sheet can get focus by tapping the sheet.



Sequence Edit Toolbar - Basic Timing page

There are several pages with many different settings for the cues. The pages can be changed using the swipe button in the upper left corner of the toolbar (see the image above).

The top row in the toolbar gives access to select a cue. There are also playback controls that can be used to run cues. Read more in the [Play Back Cues topic](#).

Turning the two rings on the encoders changes the respective values for the selected cue in the sheet.

The lower row is the outer ring of the dual encoder. The middle row is the inner ring of the dual encoders.



27.3. Content Sheet

The Content Sheet is used to see the fixtures and values stored in cues. It looks much like the **Fixture Sheet** but has a masking function that only displays what is stored in cues and cue parts. It is like a combined Sequence Sheet in tracking mode and a Fixture Sheet.

It could look like this:

Name	FID	IDType	CID	Dimmer	PanTilt	
				Dim	P	T
QuantPro 1	1	Fixture		50	2.1 Straight Front	2.1 Straight Front
QuantPro 2	2	Fixture		50	2.1 Straight Front	2.1 Straight Front
QuantPro 3	3	Fixture		50	2.1 Straight Front	2.1 Straight Front
QuantPro 4	4	Fixture		50	2.1 Straight Front	2.1 Straight Front
QuantPro 5	5	Fixture		50	2.1 Straight Front	2.1 Straight Front
QuantPro 6	6	Fixture		50	2.1 Straight Front	2.1 Straight Front
QuantPro 7	7	Fixture		50	2.1 Straight Front	2.1 Straight Front
QuantPro 8	8	Fixture		50	2.1 Straight Front	2.1 Straight Front
QuantPro 9	9	Fixture		50	2.1 Straight Front	2.1 Straight Front
QuantPro 10	10	Fixture		50	2.1 Straight Front	2.1 Straight Front
QuantPro 11	11	Fixture		50	2.1 Straight Front	2.1 Straight Front
QuantPro 12	12	Fixture		50	2.1 Straight Front	2.1 Straight Front
QuantPro 13	13	Fixture		50	2.1 Straight Front	2.1 Straight Front

Content Sheet

The Title Bar shows the sequence ID and the cue ID of the cue being shown. If a world other than the default Full world is selected, it is also displayed in the title bar (Small World icon with a name and number next to it).

The values displayed in the sheet can be edited directly in the sheet, just like in the Sequence Sheet in **Track sheet mode**.

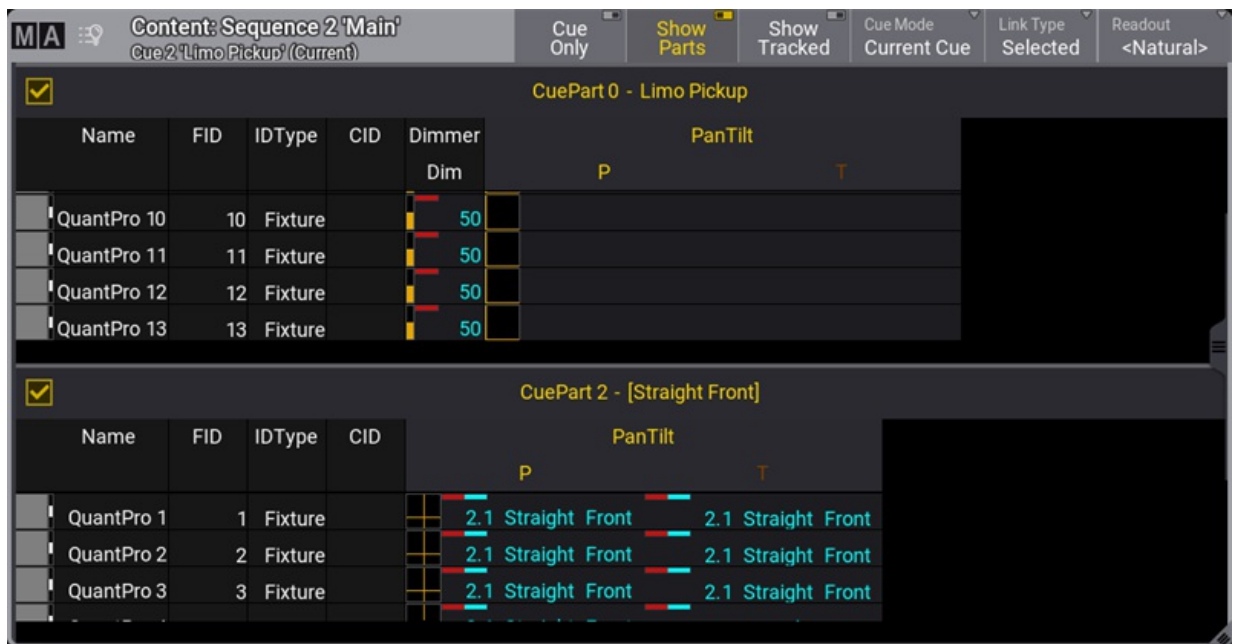
The title bar can have multiple buttons. These are settings that can also be changed in the window settings. The settings can be opened by tapping the **MA** logo in the upper left corner of the window.

Some settings are shared with other windows, and the description of these settings can be found in the **Windows Settings topic**.

Turning On **Cue Only** will make the editions using the cue only option. Learn more about Cue Only in the **Store Cues topic**.

The example image above shows most of the pan and tilt values with a small note saying "Part 2". This indicates that the values are stored in cue part 2. A red "DUP" in the lower right corner means duplicate data is stored in a different part of the cue.

If **Show Parts** is turned On, the parts are separated into different frames. The same example cue would look like this:



Content Sheet with multiple parts

Each frame with a cue part can be toggled On or Off by tapping the checkmark in the frame.

Turning **Show Tracked** On shows all the values tracked from previous cues. Turning it Off hides the tracked values and only shows the values stored in the cue.

Tapping the **Cue Mode** toggles between three different options for the sheet:

- **Current Cue:**
This will make the sheet display the information related to the current active cue.
- **Previous Cue:**
This will display the values from the previous cue. This is the last active cue, even if the sequence order is jumped.
- **Next Cue:**
This displays the values for the next cue if a Go+ is performed on the sequence. If a cue is "Loaded", the loaded cue will be displayed.
- **Manual:**
This mode allows selecting a specific cue. When this mode is selected, a list of the cues appears on the left side. Tap the desired cue in the list to see the content.



Content: Sequence 2 'Main'			Cue Only	Show Parts	Show Tracked	Cue Mode	Link Type	Readout				
Cue 4.5 'Christmas Party' (Manual)						Manual	Selected	<Natural>				
No	Part	▶ Name	Name	FID	IDType	CID	Dimmer	PanTilt		RGB		
								Dim	P	T	R	G
			QuantPro 1	1	Fixture		50 Part 1	2.2	Fan Out	4.4	Blue	
0		CueZero	QuantPro 2	2	Fixture		50 Part 1	2.2	Fan Out	4.4	Blue	
1	0	Intro	QuantPro 3	3	Fixture		50 Part 1	2.2	Fan Out	4.4	Blue	
2	0	▶ Limo Pickup	QuantPro 4	4	Fixture		50 Part 1	2.2	Fan Out	4.4	Blue	
2.5	0	Limo Ride	QuantPro 5	5	Fixture		50 Part 1	2.2	Fan Out	4.4	Blue	
4	0	Nakatomi Plaza	QuantPro 6	6	Fixture		50 Part 1	2.2	Fan Out	4.4	Blue	
4.5	0	▶ Christmas Party	QuantPro 7	7	Fixture		50 Part 1	2.2	Fan Out	4.4	Blue	
5	0	Hollys Office	QuantPro 8	8	Fixture		50 Part 1	2.2	Fan Out	4.4	Blue	
6	0	End of First Part	QuantPro 9	9	Fixture		50 Part 1	2.2	Fan Out	4.4	Blue	
		OffCue	QuantPro 10	10	Fixture		50 Part 1	2.2	Fan Out	4.4	Blue	
			QuantPro 11	11	Fixture		50 Part 1	2.2	Fan Out	4.4	Blue	
			QuantPro 12	12	Fixture		50 Part 1	2.2	Fan Out	4.4	Blue	
			QuantPro 13	13	Fixture		50 Part 1	2.2	Fan Out	4.4	Blue	

Content Sheet in Manual mode

The green frame around a cue indicates the active cue being output. The blue background indicates the selected cue being displayed in the content sheet.

Tapping the part number in the cue list automatically deselects the other parts if **Show Parts** is active.

The **Link Type** defines what sequence the sheet is showing. It is possible to select a specific sequence. Read about the Link Type and Fixed Target setting in the Window Settings topic (link above).

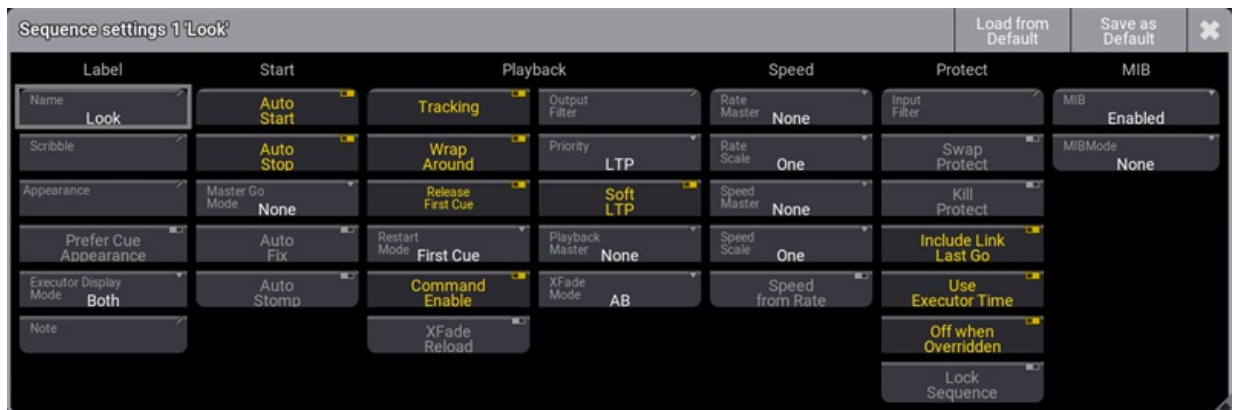
Another setting that is unique for the content sheet is called **Fixture Select**. When this is On, fixtures can be selected by tapping the name or ID in the content sheet.

The setting called "Show Notes", found in the **Mask** tab, toggles if an area is visible at the bottom of the sheet. This area shows the cue notes related to the displayed cue.



27.4. Sequence Settings

Each sequence has a lot of different settings. The settings can be accessed from the top bar of each **sequence sheet** or the **executor assign menu**. Each of these two locations has a button called **Settings**. Tap this to open the settings.



Sequence settings pop-up

The settings are divided into different sections. Label, Start, Playback, Speed, Protect, and MIB. Read about the different settings in each section below.

Each user profile has default sequence settings. These settings are used when a new sequence is created. The default can be loaded and stored from this pop-up.

The title bar has two relevant buttons. The left one is **Load from default**. Tapping this loads the stored sequence defaults. Next to this is **Save as default**. Tapping this stores the current settings as the new default.

Label

This group is about the name and look of the sequence.

Name

Edit this input field to change the name of the sequence.

Scribble

This scribble field can be used to select or create a scribble for the sequence. This scribble is visible in the sequence pool and on the executor label.

Appearance

This appearance field can be used to select or create an appearance for the sequence. This appearance is visible in the sequence pool and on the executor label.

Prefer Cue Appearance



When this option is enabled, and the current cue has an appearance, the cue appearance will be displayed on the executor or in the layout instead of the sequence appearance.

Executor Display Mode

The executor display mode defines how the sequence will be displayed on an executor:

- **Data only:**
Only the cues with their appearances will be displayed. The cue appearance is only displayed in the line of the cue and not in the background.
- **Appearance only:**
Only the sequence or cue appearance will be displayed. No cue names, fade bar, etc., will be displayed. This can be handy for sequences with only one cue, color cues, or gobo cues.
- **Both:**
Each cue line displays its cue appearance, and the sequence appearance or the appearance of the current cue will be displayed in the background of the executor.

Start

The settings in this section are about starting and stopping the sequence.

Auto Start

The **Auto Start** feature switches the executor 'On' when the master is moved from zero.

Auto Stop

The **Auto Stop** feature switches the executor 'Off' when the master is moved down to zero.

Master Go Mode

The **Master Go** function is active if auto stop is turned Off. Tapping this button opens the **Select Master Go Mode** pop-up.

There are four options here. They all take effect when the master fader is moved from zero and up. The options are:

- **None:**
The cue is still running.
- **Go:**
It executes a Go.
- **On:**
The current cue is reloaded (fading in again).
- **Top:**
The first cue is activated.

Auto Fix



The **Auto Fix** feature can be activated for each executor. It will automatically **Fix** active executors and keep them visible even when pages are changed. The executor is automatically unfixes when it is switched off.

Auto Stomp

If auto stomp is On, an **absolute** value from a cue will stomp a phaser running from a different playback.

Playback

The playback settings are about running the cues. Running or playing back cues is described in the **Play Back Cues topic**.

Tracking

This turns On or Off value tracking in the sequence. Read more in the **What is Tracking topic**.

Wrap Around

Wrap around allows the sequence to return to the top/first cue if a Go (forward) command is performed after the last cue in the sequence is reached.

Release First Cue

This setting defines if the first cue releases tracking values. These tracking values can come from the last cue if **Wrap Around** is active. If **Release First Cue** is On then it adds a **<Yes>** to the **Release** column in the first cue of the sequence. Learn more about the different columns in the **Look at Cue and Sequences topic**. The first cue can manually be set to release by editing the field in the sequence.

Restart Mode

There are three different restart modes:

- **First Cue:**
This always restarts the sequence with the first cue.
- **Current Cue:**
This restarts the sequence with the cue where it was when the sequence was turned Off.
- **Next Cue:**
This restarts the sequence with the next cue based on where it was when the executor was turned Off.

Cmd Enable

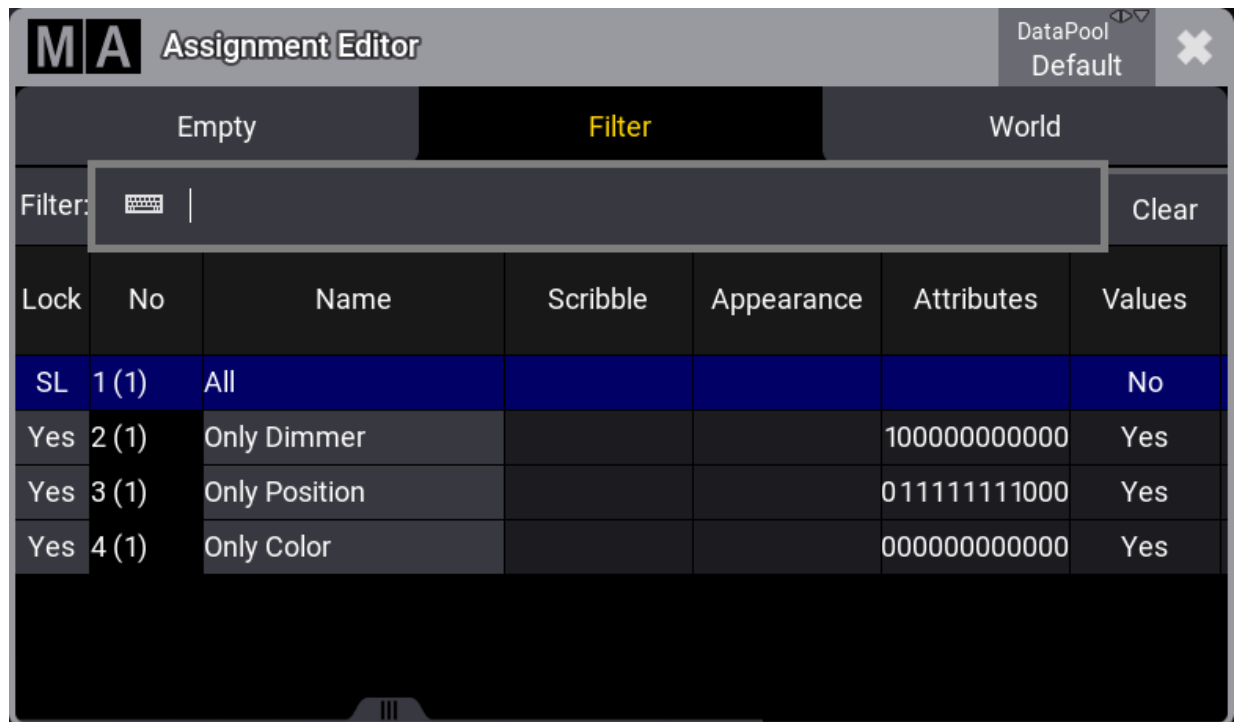
This option disables the execution of the commands in the CMD columns in a sequence. The stored commands are not deleted or removed. They are just disabled.

XFade Reload

When this option is enabled, the Xfader needs to be pulled back to 0 after completing a crossfade to do the next crossfade.

Output Filter

The output filter can have a **Filter or World** assigned. Tapping the button opens the **Assignment Editor** pop-up:



Tap a world or filter in the list to apply an output filter

The editor has three tabs. **Empty** can be used to select no filtering. **Filter** and **World** are lists of each type. Each lists the possible choices of their type. Tap the desired filter or world to apply it.

The fixtures and attributes in the world or filter are allowed to pass the output filter and be output from the sequence. The same sequence can be played back from several executors, and each executor shares the output filter settings. If different output filters are needed, the sequence should be linked or shared with another. The other sequence can have different output filter settings. Read about linked sequences in the **Cues and Sequences topic**.

If the sequence has an output filter applied, there is a small output filter icon (🔒) in the sequence pool button.

Priority

This is the priority of the sequence. The priorities are described in the **Play Back Cue topic**.

Soft LTP

The Soft LTP function is described in the **Play Back Cue topic**.

Playback Master



Here it is possible to select a **Playback Master**. It functions as a sub-master that multiple sequences can share.

XFade Mode

This is used to set how the two CrossfadeA/XFadeA and CrossfadeB/XFadeB faders work. There are two crossfade modes:

- **Split:**
The dual crossfaders work as masters for the current/next cue.
- **AB:**
The dual crossfaders work as crossfaders for increasing/decreasing values.

Speed

Speed and rate can be used to adjust the stored times without reprogramming the show. For instance, a cue can have a fade time of 5 seconds. A rate master can adjust the fade time live while playing it back. All sequences have their own individual rate and speed, but they can be linked to a global master. This master can then adjust the timing for multiple sequences simultaneously. Read more details in the **Speed Masters topic**.

Rate Master

The sequence has a rate master. It can be linked to a shared global speed master, or it can have an individual rate master. Tapping this button will open the **Select Rate Master pop-up**.

In the pop-up, it is possible to select the **None** option for having an individual rate master for the sequence or select one of the global speed masters.

Rate Scale

Enabling this binds the rate to defined steps instead of a variable value. Tapping this button opens the **Select Rate Scale pop-up**.

Choosing one of the steps in the pop-up, selects the multiplier or divider. This multiplies or divides the rate by the selected factor.

Speed Master

The sequence has a speed master. It can be linked to a shared global speed master, or it can have an individual speed master. Tapping this button will open the **Select Speed Master pop-up**.

In this pop-up, it is possible to select the **None** option for having an individual speed master for the sequence or select one of the global speed masters.

Speed Scale

If a sequence is assigned to a global speed master (read above), it can be useful to adjust a speed scale. Tapping this button will open the **Select Speed Scale pop-up**.



In the pop-up, selecting one of the multipliers or dividers is possible. This multiplies or divides the speed by the selected factor.

Speed from Rate

This links the speed to follow the rate.

Protect

This group of settings is used to protect the sequence from different actions.

Input Filter

The playback filter is described in a little bit more detail in the [Cues and Sequences topic](#). Tapping this button opens the **Assignment Editor pop-up**: See above about the output filter.

The fixtures and attributes in the world or filter are allowed to pass the filter and can be stored in the sequence.

Swap Protect

Activating this option protects this sequence from the Swap playback action. Learn more about this action in the [Swap Keyword topic](#).

Kill Protect

Activating this option protects this sequence from the Kill playback action. Learn more about this action in the [Kill Keyword topic](#).

Include Link Last Go

This setting is On as default. When it is set to Off, playing back a sequence will not trigger the LinkLastGo functionality in the [sequence sheet](#).

Use Executor Time

This makes the executor playback cues using the stored timing. If this is turned on, it is affected by the **Exec Time** master fader, who overwrites the timing.

Off when Overridden

The Off when Overridden function allows a sequence to be automatically turned Off if another sequence has taken control with all the attributes in the sequence = this executor does not control any attributes.

Lock Sequence

The sequence is locked against changes when this is On. It can still be played back.



MIB

This group is about MIB settings for the sequence. Read more about MIB in the [Move In Black topic](#).

MIB

Enable, disable, or force MIB for the sequence. The options are:

- **Enabled:**
MIB will be performed according to the cue and cue part MIB settings.
- **Never:**
MIB will never be performed for this sequence. All cue and cue part-specific MIB settings will be ignored.
- **Force Early:**
Forces the early MIB for all cues that can perform MIB as soon as the dimmer is closed. Further MIB settings specified per cue or cue part will be ignored.
- **Force UponGo:**
MIB is forced to be executed with the next cue transition after the dimmer is closed. For all cues that can perform MIB. Further MIB settings specified per cue or cue part will be ignored.
- **Force Late:**
Forces the MIB latest in the cue before the dimmer opens again. For all cues that can perform MIB. Further MIB settings specified per cue or cue part will be ignored.

MIB Mode

This setting defines which MIB mode will be executed when doing a MIB. The **MIB Mode** per cue or cue part needs to be set to **Default**. The options are:

- **None**
- **Early**
- **UponGo**
- **Late**

Learn about the mode types in the MIB topic (link above).

27.5. Store Cues

Storing a cue is the default **Store** action in grandMA3.

This means that if nothing else is defined and **Please** is pressed right after **Store**, then a new cue is stored in the selected sequence.

Or press **Store**, followed by pressing the executor button where the sequence, with the cue, should be stored.

If active values are in the programmer, they are stored in the cue, but programmer values are not needed to store cues.

Of course, there are more details about storing cues, so please keep reading.

Store a New Cue on an Empty Executor

If a cue is stored on an empty executor, the grandMA3 software automatically stores the cue in a new sequence and assigns this executor to control the sequence using the default settings.

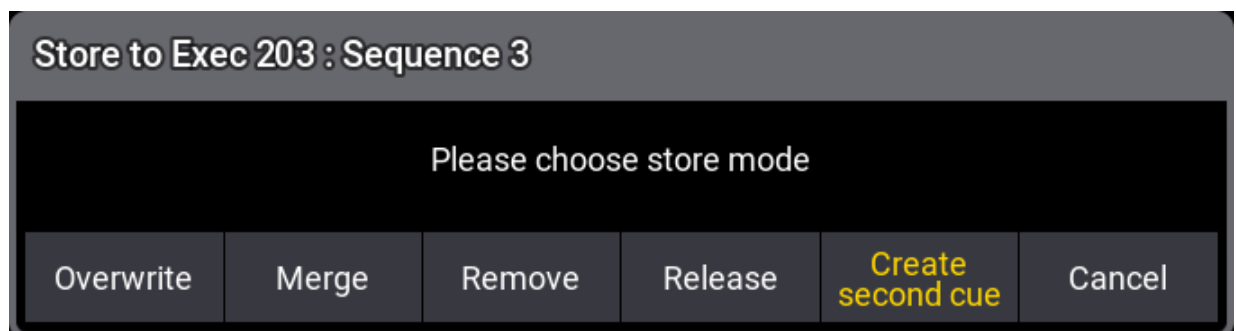
Returning to the second example at the top of this topic: just pressing the **Store** button and then an executor button on the empty executor is enough. The software assumes the desired action is storing a cue.

This will automatically be cue number 1 - nothing else was specified.

Storing cues obey **Worlds and Filters**. This enables control of what is stored. Worlds and filters can also be assigned to the sequence both as an input and also as an output filter - independently of each other. This will function as an input or output filter, allowing only the elements in the world or filter to be stored in or played back from the sequence.

Store the Second Cue

If the store function is used again on the same sequence - without adding cue number details - then the grandMA3 does not know what should happen, and a pop-up appears, giving different choices.



Please choose store mode pop-up with "Create second cue" option

Tapping **Create Second Cue** will store a cue with the next whole number.



Overwrite and **Merge** options are explained below. **Remove** and **Release** are described in the [What is Tracking](#) topic. **Cancel** does not store anything.

Cue Numbers

When a cue is stored, it is possible to specify a cue number. This is done by using the following syntax: **Store Cue [Cue_Number]**. It is also possible to specify a sequence or an executor, using the keys in the command section, while storing:

Store Cue [Cue_Number] Sequence ["Sequence_Name" or Sequence_Number] or
Store Cue [Cue_Number] Executor [Executor_Number].

Notice that the cues are stored in the sequence. Using the executor number will store the cue into the sequence the executor is controlling.

Cue numbers have three decimal numbers. If all are zero then they are not displayed. But cue number "42" is the same as cue number "42.000" - it is not "42 thousand", it is "42 point 0 0 0". The currently highest cue number that can be stored is "999 999.999". The lowest number that can be stored is "0.001".



Restriction:

Storing nearly 1 billion cues will completely fill the memory and make the show file VERY big. The software will cancel the store process before the system crashes, but almost any operation after this will make the software shut down!!

The software hides the trailing decimal zeros, but they are still there. This means that cue "5.2" is after cue "5.11" because they are actually cues "5.200" and "5.110".

It is not limited to only storing a single cue number at a time. It can just as easily be a range of numbers - this means it is possible to use **Thru**, **+**, and **-** keys to create number ranges to store.

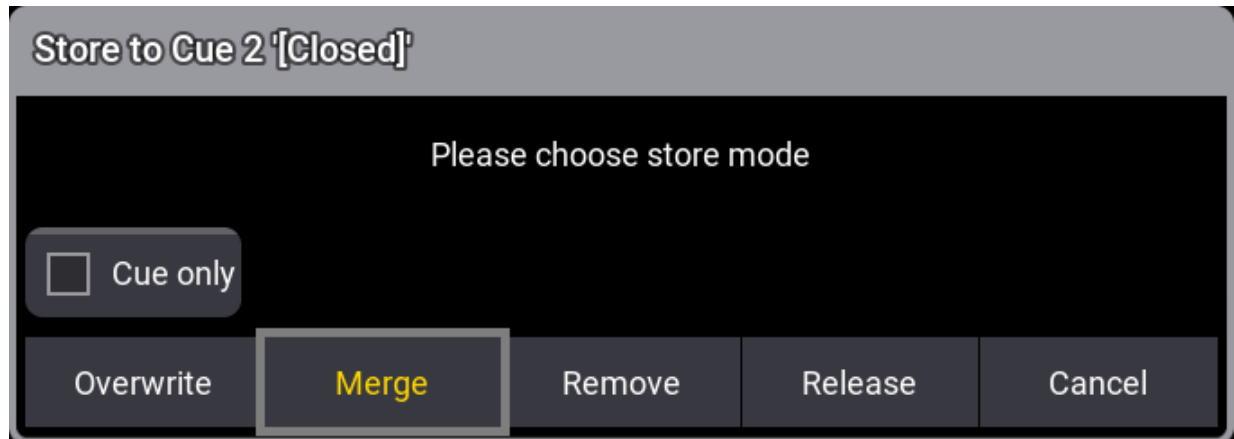
If you are working with or adding a lot of cues to the same sequence, then selecting the sequence can be a good idea.

This can be done easily by pressing **Select** and then one of the buttons associated with the executor controlling the sequence or tapping the sequence in the sequence pool.

The selected sequence is used if no sequence or executor is defined in the store command.

Store Into Cues That are Not Empty

If the store operation is used to store into already existing cues, then a pop-up like this appears:



Please choose store mode pop-up

It only appears if the store options do not specify what should happen.

Look at the [Store Options and Defaults topic](#) for information about specifying this while storing.

Remove and **Release** are described in detail in the [What is Tracking topic](#). **Cue Only** is described [below](#).

The two relevant options are:

- **Overwrite:**
This will remove what is already stored in the cue and only store the new values.
- **Merge:**
This will merge the new values into the existing values. New values have a higher priority and will overwrite existing values.



Examples

In the following examples, we have a sequence with the following two cues:

M A Sequence 3							Cue Only	Track Sheet	Settings
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	2 Dim	
	0		CueZero						
	1	0	Cue 1	<Yes>			50		
	2	0	Cue 2				50	60	
			OffCue	Yes					

The magenta value for fixture 1 in cue 2 is a tracked value.

Now we turn on fixture 3 at 100 % and store this into cue 2.

This is the result if **Overwrite** is chosen:

M A Sequence 3							Cue Only	Track Sheet	Settings
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	3 Dim	
	0		CueZero						
	1	0	Cue 1	<Yes>			50		
	2	0	Cue 2				50	100	
			OffCue	Yes					



Now fixture 2 is gone. This is because it only had values stored in cue number 2. The dimmer value of fixture 1 is not affected because it is a tracked value.

If we had chosen **Merge** instead, it would have looked like this:

Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	2 Dim	3 Dim
	0		CueZero						
	1	0	Cue 1	<Yes>			50		
	2	0	Cue 2				50	60	100
			OffCue	Yes					

Now the value from fixture 3 is added to the existing values.

Store Cues with Timings

When storing a cue, it is possible also to store the different cue timings. This is described in detail in the [Cue Timings topic](#), but here is the short version.

The **Time** key will add different timing keywords to the command when storing.

For instance, storing cue 4 with a fade time of 6 seconds and a delay of 1 second, the following keys can be pressed:

Store Cue 4 Time 6 Time 1 Please

This is the result in the command line feedback:

```
OK : Store Cue 4 CueFade "6" CueDelay "1"
```

Pressing the **Time** button repeatedly will change what timing keyword it adds.



Adding and Using Cue Labels

A cue can be given a name - using the **label keyword** - while it is stored. This is the syntax: **Store Cue [Cue_Number] "My Cue Name"**.

The keyboard is needed for writing this. The quotation marks are needed to tell the software that this is text - then, it is not interpreted as a command.

Labels can also automatically be enumerated while storing. Have a look at this command:

```
MA User name[Fixture]> Store Cue 2 + 4 "BO Scene 1"
```

This will not label both cues 2 and 4 the same - it will add 1 to the number for each cue. The result is that cue 2 is called "BO Scene 1" and cue 4 is "BO Scene 2". This enumeration only works if the number is the last part of the label and if there is a space between the last word and the number.

Cue labels can be used when storing. This means that if there are several cues whose labels start with "BO", it is possible to store into all these cues in one operation using BO plus an asterisk. See the following example.

Example

This is the cue sequence and content before storing:

Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim
	0		CueZero				
	1	0	Cue 1	<Yes>			100
	2	0	BO Scene 1				100
	3	0	Cue 3				100
	4	0	BO Scene 2				100
	5	0	Cue 5				100
			OffCue	Yes			

Notice that cues 3 and 5 are **blocked**. This means that 100% is stored in the cues even though it is currently not necessary.

With an active value of 0% for fixture 1, use the keyboard to type the following command:



MA User name[Fixture]> Store Cue "BO*" /Merge

This is the result:

MA		Sequence 4				Cue Only	Track Sheet	Settings
Lock	No	Part	Name	Release	Break	Tracking Distance	1 Dim	
	0		CueZero					
	1	0	Cue 1	<Yes>			100	
	2	0	BO Scene 1				0	
	3	0	Cue 3				100	
	4	0	BO Scene 2				0	
	5	0	Cue 5				100	
			OffCue	Yes				

The two cues whose label begins with BO now got the new value.

Store Cue Parts

Besides the many cues that could be stored in each sequence, it is also possible to store 256 **Cue Parts** to each cue.

Values are always stored in a cue part, and often when we talk about a cue, we are actually talking about all the parts in a cue.

Cue parts are a division of the cue. Cue part 0 is always created with the cue, and this part has all the values unless a different part is created or specified when storing.



Have a look at this example:

M A Sequence 3							Cue Only	Track Sheet	Settings		
Lock	No	Part	▼	Name	Release	Break	Tracking Distance	1 Dim	2 Dim	3 Dim	
	0			CueZero							
	1	0		Cue 1	<Yes>			50	0	0	
	2	0	▼	Cue 2				50	60	.	
		10		Part 10				50	60	100	
				OffCue	Yes						

Notice that the value for fixture 3 is stored in cue 2, part 10. It does not have a value in cue 2, part 0. And the values from fixtures 1 and 2 are in part 0 and therefore do not have any value in cue 2, part 10. It appears as tracked values.

Storing something in a part is almost as easy as storing the main cue. Using the example above, the keypresses would be:

Store Cue 2 Cue 1 0 Please

The second press on the cue key will result in the **Part keyword**, and the command line feedback looks like this:

OK: Store Cue 2 Part 10

As a default, it is only possible for an attribute to be stored in one cue part per cue. But this can be changed so an attribute can have values in multiple cue parts in the same cue. If **Allow Duplicates** is turned On for a cue then attributes can be stored in all parts of the cue.

In this example, **Allow Duplicates** is set to Yes for cue 2, and fixture 1 has values in both part 0 and part 10:

M A		Sequence 3					Cue Only	Track Sheet	Settings		
Lock	No	Part	▼	Name	Release	Break	Tracking Distance	1 Dim	2 Dim	3 Dim	
	0			CueZero							
	1	0		Cue 1	<Yes>			50	0	0	
	2	0	▼	Cue 2				50	60	.	
		10		Part 10				40	60	100	
				OffCue	Yes						

The cue part with the highest number takes precedence, and in the example above, fixture 1 will end at 40% output when cue 2 is triggered.

Allow Duplicate is a column in the Sequence Sheet.

Cue Only

Cue Only is a **store option** for cues. When storing into a cue with the Cue Only, the tracked values will be blocked in the next cue or cue part to preserve the previous look on stage. The actual values of the programmer will only be stored in the target cue or the cue part.

Cue Only can be used when storing into existing cues - except the last cue. There is no hard key for cue only, so it needs to be activated otherwise. These are the following options:

- As command line option keyword **/CueOnly** or /CO after the normal **Store keyword**.
- In the store options: To open the store options, press and hold **Store** for at least 2 seconds. Then turn On cue only by tapping **Cue Only**.
- In the **Store Cue pop-up**. When storing onto an existing cue, the Store Cue pop-up will appear and ask whether to Overwrite, Merge, Remove, Release, or Cancel the current store operation. It is also possible to decide if the cue is to be stored with active cue only within the pop-up. Remove and Release can be used for Cue Only. The pop-up only appears if the desired cue is not the last one in the sequence.

When storing cue only, the grandMA3 software decides, based on the 3 following rules, which cue part will block the original values:

1. The default cue part is cue part 0.



2. If the next cue already contains a part with the same name as the cue part from which the original value is coming, the blocked value will be used in this part.
3. If the cue already contains attributes of the same feature group, the previous values will be blocked in this cue.

If none of the rules are true for phaser values, a new cue part will be created where the previous values will be blocked.

Using Command Line Input to Add More While Storing

In the example using the cue labels to store, a command showed some of the other possibilities while storing cues.

The command line gives access to all the store options in the GUI Store Options - read about them in the [Store Options and Defaults topic](#).

All the different elements are described in the [Store Keyword topic](#).

Examples

The following are just a few extra command line examples showing some of the possibilities while storing.

```
MA User name[Fixture]> Store Cue 1.2 Sequence 4
```

Stores cue number 1.2 in sequence 4.

It does not matter if you write sequence or cue first. So this could also have been:

```
MA User name[Fixture]> Store Sequence 4 Cue 1.2
```

The commands can often be written shorter in the command line input. See some examples in the [General Syntax Rules topic](#).

Read the topics about each keyword to see the short version of the keyword.

```
MA User name[Fixture]> Store Cue 42 "Al Powell arrives at the Plaza" CueFade 6/3 /merge
```

This will store the cue with a name, merged, and stored with an in-fade of six seconds and an out-fade of three seconds. Read more about the store options in the [Store options and defaults topic](#).

Store Remove

A version of storing is the **Store Remove**, where **Remove** is selected in the store pop-up (described above).

This will remove the stored values for the attributes that currently have active values in the programmer.



The actual values in the programmer are irrelevant in this case. They simply indicate what attributes should be removed from the cue.

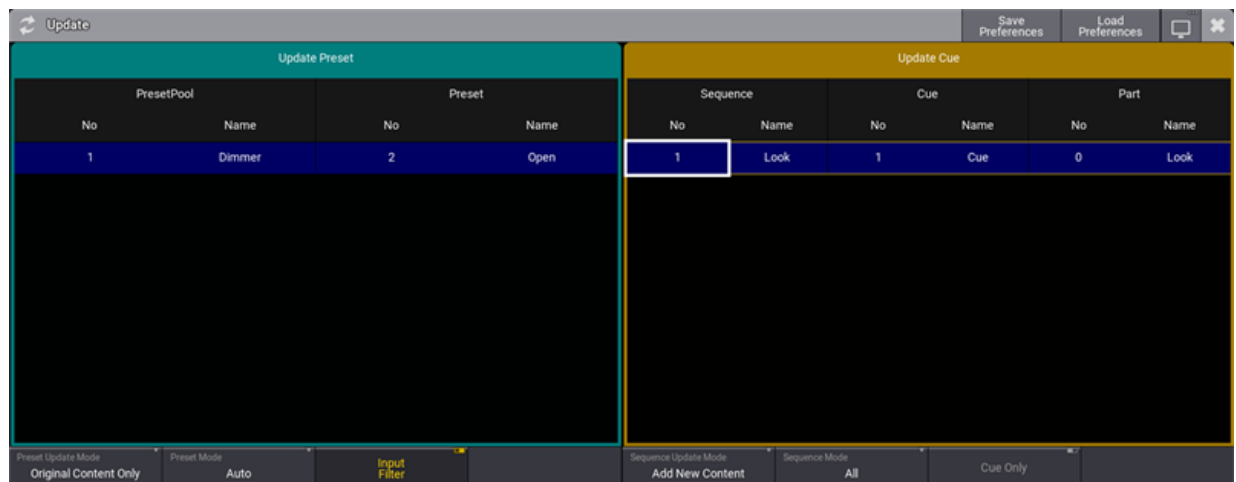
27.6. Update Cues

The stored values in a cue can always be changed by storing new values in them.

But if there are active cues and active values in the programmer, then it is also possible to update the cue or cue part.

The **Update key** flashes when this scenario is valid.

Pressing **Update** opens the **Update Menu**.



The Update Menu with possible update locations

To update the cue part with the current settings, just tap the desired cue part on the list on the right side. This immediately updates the stored values in the selected location.

There are two options that are relevant to the cue update process. **Sequence Update Mode** and **Cue Only**.

Sequence Update Mode:

The mode toggles between two options. **Original Content Only** updates only the already existing values at their original location - possibly a previous cue. **Add New Content** updates the active cue with all the programmer values (filter and worlds are still respected), possibly adding not only new values but also adding new content to the cue.

Cue Only:

Cue only can be toggled On or Off. If it is Off, then the values are updated using normal tracking behavior. If it is On, then the updated values are updated as cue only changes. Read more about cue only in the [Store Cues topic](#).

Sequence Mode:

The sequence mode is a filter that can be used to filter the list of possible cue parts. The options are:



- **All:**
This shows all possible cue parts from all active sequences.
 - **Selected:**
This only shows the cue part from the selected sequence.
 - **Last Go:**
This shows the latest cue to receive one of the trigger commands (<<<, >>>, Go+, Go-, Goto, Load, On, Select, Top, Temp, Flash, Toggle On, Pause). This includes if the cue is triggered from a running timecode recording. A sequence (and its cues) can be excluded from LastGo - read about the [Sequence Settings](#). Last Go only shows cues triggered by the same [user profile](#).
-

The left side of the **Update Menu** is about presets. Learn more about presets in the [Preset section](#).

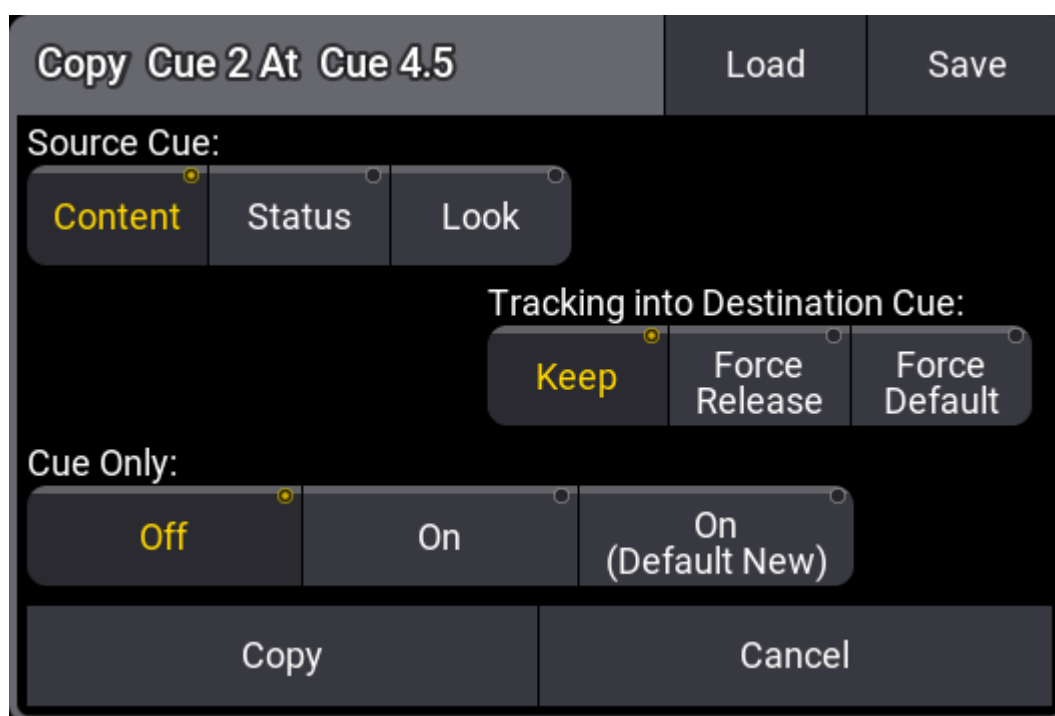
27.7. Copy Cues

Copy to an Empty Destination

Cues can be copied to a new cue.

The **Copy keyword** is needed for this operation. The general syntax is **Copy Cue ["Cue_Name" or Cue_Number] At ["New_Cue_Name" or New_Cue_Number].**

Different options define what and how the cue is copied. These options appear in a pop-up when the copy command is executed.



Copy cue to a new empty location - with cues after the new copy location

Source Cue has three different radio buttons that are mutually exclusive:

- **Content:**
Copies only the values stored in the source cue.
- **Status:**
Copies the status of the source cue. Status includes the values stored in the cue and all tracked values from former cues.
- **Look:**
Copies all values of attributes for fixtures where the dimmer is open in the source cue, no matter if the open dimmer value is set in this cue or if it tracks into the source cue. All other fixtures used in the source sequence will get only their dimmer copied at 0.



Tracking into Destination Cue also has three mutually exclusive radio buttons. This setting defines what should happen with values that track into the new copied cue from previous cues.

- **Keep:**
The tracking values are kept and tracked into the new cue.
- **Force Release:**
Tracking values are forced to the "Release" special value.
- **Force Default:**
Tracking values are forced to the default value.

Cue Only is another three radio buttons. Read more about cue only in the [Store Cues topic](#). This is only visible if there are cues following the copied-to location.

- **Off:**
The **Cue Only** function is Off, and values will track onward.
- **On:**
Cue Only is on.
- **On (Default New):**
Cue only is on. Attributes that do not have a previous value to return to will be set to the default value.

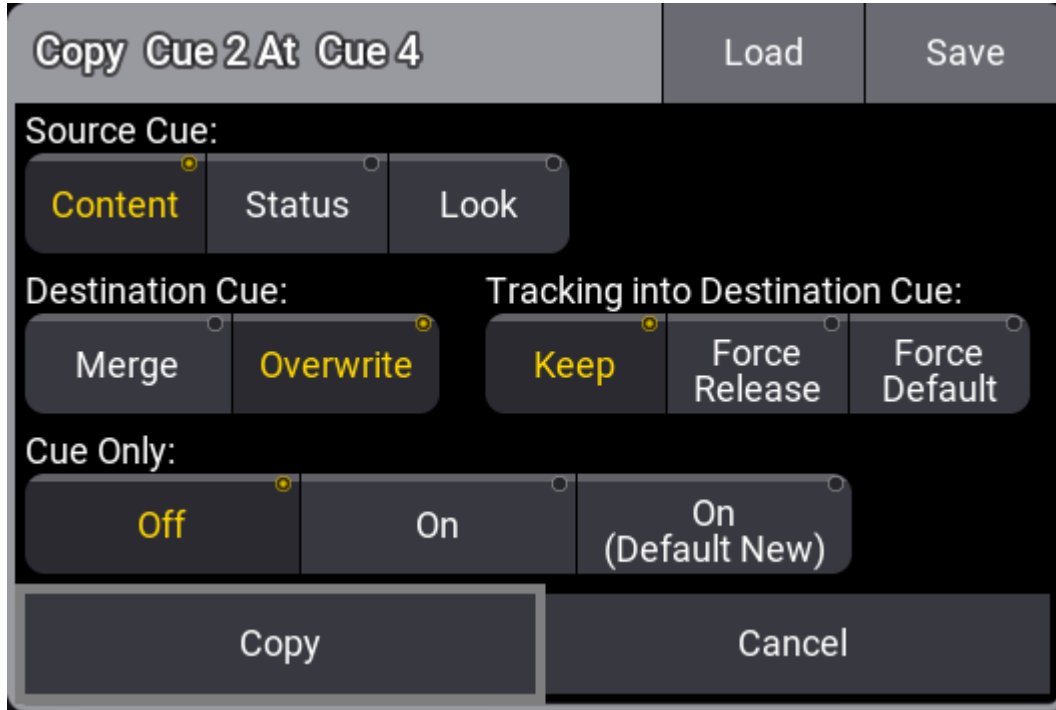
Finally, there are two execution buttons:

- **Copy:**
Copies the source to the destination using the settings above.
- **Cancel:**
Cancels the cue copy operation and closes the pop-up.

Copy to an Existing Destination

Copying a cue to an existing cue uses the same syntax except the destination is an existing cue.

The pop-up look s a little different:



Copy cue to an existing cue

There is another option called Destination Cue. This defines what should happen with values existing in the destination cue. Values from the source cue will copy into the cue. This option defines what happens with values in the destination cue that is not defined by the source cue.

- **Merge:**
Merges the data of the source to the destination. Existing data in the destination will be kept as long as both the source and destination do not have data of the same attribute. If existing data in the destination is also part of the source, the data from the source wins and overwrites the destination.
- **Overwrite:**
Overwrites the cue with the source cue data and deletes values that are not defined by the source cue.

Copy of Ranges

When copying ranges of cues, these rules apply:

- When a natural order of cues is selected using **Thru**, the software attempts to maintain all gaps from the source range at the destination.

Example:

Precondition: Cues 1, 2, and 4 exist.





Result: New cues 11, 12, and 14.

- If a cue exists already in the range of the destination spot (even if it would fit into a gap), the software suppresses the range at the destination so that the original order of cues is uninterrupted. The software tries to set whole cue numbers. If this is not possible, it appends dotted cue numbers. The software tries to use as few decimals as possible (1 -> .1 -> .01 -> .001).

Examples:

Precondition: Cues 1, 2, 4, and 12 exist.

```
MA User name[Fixture]> Copy Cue 1 Thru 4 At Cue 11
```

Result: New cues 11, 11.1, and 11.3. The gap between cues 2 and 4 is maintained but shifted, as the start of the destination is 11 and not 11.1.

Precondition: Cues 1, 2, 4, and 11.1 exist.

```
MA User name[Fixture]> Copy Cue 1 Thru 4 At Cue 11
```

Result: New cues 11, 11.01, and 11.03. The gap between cues 2 and 4 is maintained but shifted, as the start of the destination is 11 and not 11.1.

- When the source range of cues is selected in a reversed order by using **Thru**, the software ignores the gaps from the source range when creating the destination range. The same rule for the resulting cue numbers at the destination from above applies here as well.

Example:

Precondition: Cues 1, 2, and 4 exist.

```
MA User name[Fixture]> Copy Cue 4 Thru 1 At Cue 11
```

Result:

The content, status, or look of cue 4 is copied to cue 11.

The content, status, or look of cue 2 is copied to cue 12.

The content, status, or look of cue 1 is copied to cue 13.

- In the case of defining the source range by using **+**, the arising gaps between the single cues can be maintained if the cues are selected in ascending order. If the order is reversed or mixed, then the gaps are suppressed. The same rule for the resulting cue numbers at the destination from above applies here as well.

Examples:

Precondition: Cues 1, 2, and 4 exist.



```
MA User name[Fixture]> Copy Cue 1 + 4 At Cue 11
```

Result: New cues 11, and 14

Precondition: Cues 1, 2, and 4 exist.

```
MA User name[Fixture]> Copy Cue 2 + 1 + 4 At Cue 11
```

Result: New cues 11, 12, and 13.

- When the source list is generated by using a combination of **Thru** and **+**, the above rules apply. The thru-part of the source list uses the rules of thru, while the +-part uses the +-rules.

Copy Paste

The copy function places data in temporary memory. This can be pasted to a destination using the **Paste keyword**.

The principle is that a cue is copied into the memory and pasted to a destination.

This process does not provide the same options as described above in the normal copy syntax. Paste uses content as the source and will do a standard copy or merge.

Cut

Cues can be cut and pasted as a means to move a cue. Use the **Cut keyword** instead of copy. This does not give the same options as the copy function.

A cue needs to be cut first and then pasted into a destination.



27.8. Cue Recipes

Recipes can be used in cues and presets. See the [Recipe](#) topic to learn the basics of recipes and the [Recipe Preset](#) topic to learn about using recipes in presets. It is a good idea to read the recipe topic before this one.

This topic is about recipes in cues.

A recipe can contain multiple lines describing what should happen based on a set of information. The recipe "cooks" values into the cue.

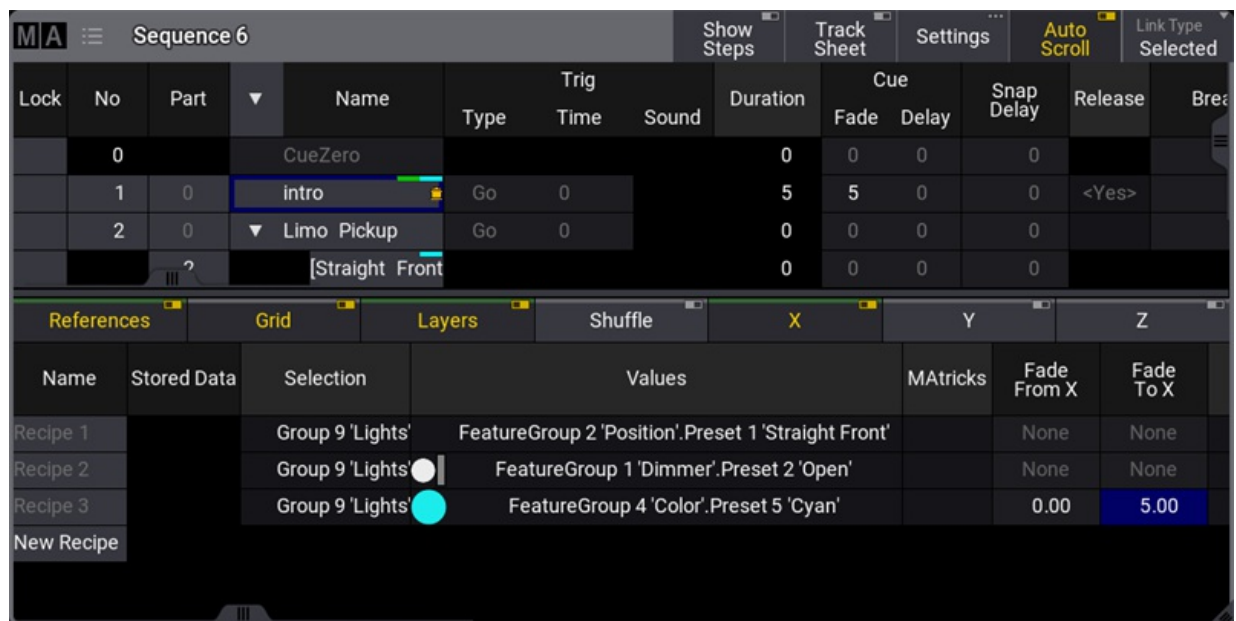
A recipe line can contain information about a selection, values, MAtricks reference, individual fade, delay, speed, phase values, and grid values.

Values from recipes can be combined with conventionally stored values.

Adding Cue Recipes

Recipes are added to each cue part in a sequence.

The best way to access the recipes is by turning On the **Show Recipes** mask in a [Sequence Sheet](#):



Sequence sheet with cue recipes showing

This gives access to adding, editing, and deleting recipe lines.

Showing the recipe lines in the sequence sheet adds a filter line that allows filtering of the different elements in the recipe. This line can be moved up or down by tapping and holding the line and sliding it up and down.

Release the screen at the desired location.

Learn more about this line and the different columns in the [Recipes](#) topic.



The cells in the lines can be edited and more lines can be added by tapping and holding the **New Recipe** .

Each line can contain a set of information.

The MAtricks reference column and the individual MAtricks columns only take effect when there is ranged data from, for instance, a **MAGic preset** or a timing range.

Making changes to the recipe line automatically cooks the line using the merge option. Recipe lines without a group do not auto-cook.

Deleting an already cooked recipe line does not auto-cook. The recipe needs to be manually cooked to reflect the new result.

Cooking can be done using the **Cook keyword**. An entire sequence can be cooked in one command.

The general syntax for cook is: **Cook [object] (/option)**

There are three options:

- **Merge** - Default if nothing else is specified. Replace existing cooked data and add new data based on the recipe ingredients, but do not change non-cooked data.
- **Overwrite** - Delete all contents of target object and cook new data based on the recipe ingredients.
- **Remove** - Delete any data from the target object that has been cooked but never updated.

Executing a cook command without an option keyword opens a pop-up prompting for which option to use. It also includes a cancel option that cancels the cook command.

A cue with recipe information gets the small pot icon in the name column. It does not indicate whether there are cooked values or not.

27.9. Store Settings and Store Preferences

There are some settings used when cues or presets are stored.

Each user has their own preferred settings.

During the store, it is also possible to add option commands that will use a specific set of store settings.

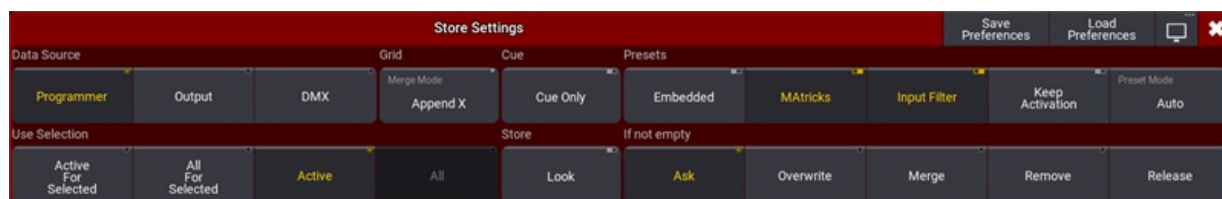
In this topic, the settings regarding storing cues are described. Some of these settings will impact how other cues will respond. Elements regarding tracking are described in the [What is Tracking topic](#).

Temporary Store Settings

The store settings can be opened as a temporary version.

This is done by pressing and holding the **Store** key for approximately one second.

It appears as a pop-up on screen 1 (default location).



Store Settings pop-up showing the current settings

There are different areas with different settings: Data Source, Grid, Cue, Presets, Use Selection, Store, and If not empty.

In the title bar, there are three buttons. Tap **Save Preferences** to store the current settings as the new default user store preferences. Tap **Load Preferences** to load the stored user store preferences values into the store settings pop-up. The last button is used to [change the menu location](#).

Data Source

This defines the data source for what will be stored.

There are three different sources:

- **Programmer** (Default):
This will store the values in the programmer. It takes the Use Selection and Store setting into account (read below).
- **Output**:
Stores the status of the current output from the console. The output can be affected by masters and DMX profiles. The [DMX sheet](#) shows what the output actually is. The **Use Selection** setting is limited to **All For Selected** or **All** (Default).



- **DMX:**

This stores incoming DMX values for DMX channels that have fixtures patched. The **Use Selection** setting is limited to **All For Selected** or **All** (Default).

When storing incoming DMX, only the values are stored that are not at the default of the fixture type. In addition, values that are at fixture type default will be stored into the sequence when the sequence contains already this attribute with a different value.

Grid

There is one setting in this area. It is Merge Mode. This is relevant for all elements that store the fixture grid position - Learn more about the grid in the [Selection Grid topic](#).

The **Merge Mode** is used when an object (including fixture grid position) is merged into another object of the same type. For instance, a group merged into another group or a preset merged into another preset.

There are two options that the button toggles between:

- **Append X:**

The fixtures are merged into an offset X position in the selection grid. The merged fixtures (source fixtures) are positioned after the last existing fixture (destination fixtures). They are appended to the existing fixtures X position.

- **Off:**

The fixtures are merged into the grid positions where they are originally stored.

Cue

There is one toggle button here: Cue Only. This can be turned On or Off.

The principle behind cue only is that changes are stored in the specified cues without tracking values into the next cues.

Learn more about **Cue Only** in the [Store Cues topic](#).

Presets

These are the settings used when storing presets.

They are described in the [Create Presets topic](#) or in the other [Preset topics](#).

Use Selection

This decides what values, from the source, will be used when storing.

- **Active For Selected:**

Stores all active attributes but only for the fixtures selected in the programmer. This is only available to the **Programmer** data source.



- **All For Selected:**
Stores all attributes of the fixtures selected in the programmer, with respect to the data source selected above.
- **Active** - (default for programmer source):
Stores the values that are active in the programmer. This is only available to the **Programmer** data source.
- **All** - (default for Output and DMX source):
Stores all attributes for all fixtures.

Store

This area only has one toggle button: **Look**. This can be turned On or Off.

When this is On, then all attributes for fixtures with a dimmer value above 0, are stored. If **Active For Selected** is selected and there are fixtures that have a dimmer level actively at 0% in the programmer, then only the dimmer value will be stored for these fixtures even if other attributes are active in the programmer.

This can be combined with various settings for **Data Source** and **Use Selection**.

If Not Empty

This defines the store method used if values are stored into an existing cue.

- **Ask** - This is the factory default setting:
This means that none of the other options below are used and a pop-up asks the user what to do.
- **Overwrite:**
All existing data is deleted and the current source and selection are used to store new values.
- **Merge:**
This will merge the new values into the existing values. New values have a higher priority and will overwrite existing values.
- **Remove:**
This will remove the stored values for the attributes using the current **Use Selection** setting.
- **Release:**
A special release value is stored. The actual values in the source are not relevant, but the selection is used to define where the release value is stored.

Cue Preferences

There is a set of timing preferences used when cues are created. They can be seen and edited in the **Menu** -> **Preferences and Timings**.

Here are the general cue timings and the preset timings. Read more about these different timings and what they do in the **Cue Timing topic**.



27.10. Play Back Cues

Cues are stored in a Sequence. All sequences are stored in a sequence pool.

When a cue is active, then the sequence is active in the pool.

It is not possible to run a cue without playing back the sequence.

Executors are controls that allow for easy hands-on playback of the sequences. Executors can playback or trigger other objects than sequences as well.

Sequences do not need to be associated with an executor to playback cues.

Relevant Playback Commands

Many keywords can be used for playback operations. They are all listed using the [Help](#) command.

For sequence playback, there are some very common keywords (read details about them by following the links):

- **Go+**
Use this to trigger the next cue with a "Go" trigger or to specify a cue (**Go+ Cue [Cue_Number]**).
- **Go-**
Use this to trigger the previous cue.
- **Goto**
Use this command to go to a specific cue. The specified cue is triggered when the command is executed. Please note that this command also asserts tracked values so that the visual result can be different than a normal Go command.
- **Load**
This is used to preload cues. The cue is then ready to be triggered via a Go+ command. Several cues in different sequences can be loaded and triggered together with the **Go+ Loaded** syntax.
- **Pause**
Executing this command will pause all running fades, delays, and phasers. Effectively halting all values where they currently are. The fade and delay are resumed also using the Pause command.
- **Top**
This keyword is used to trigger the first cue

But there are many more that can be useful.

These functions can be assigned to executor keys for easy access (See how in the [Assign Object to an Executor topic](#)), but they can also always be sent directly to the sequence. For instance, the following command can be used if you want to go to cue 4 in sequence 6:

```
MA User name[Fixture]> Goto Cue 4 Sequence 6
```

The order of sequence and cue does not matter. So it could also be:



```
MA User name[Fixture]> Goto Sequence 6 Cue 4
```

The system interprets it as in the last example. The Command Line History window shows this response:

```
OK: Goto Sequence 6 Cue 4
```

If an executor handles a sequence, then the playback commands can also be sent to the executor.

```
MA User name[Fixture]> Go Executor 101
```

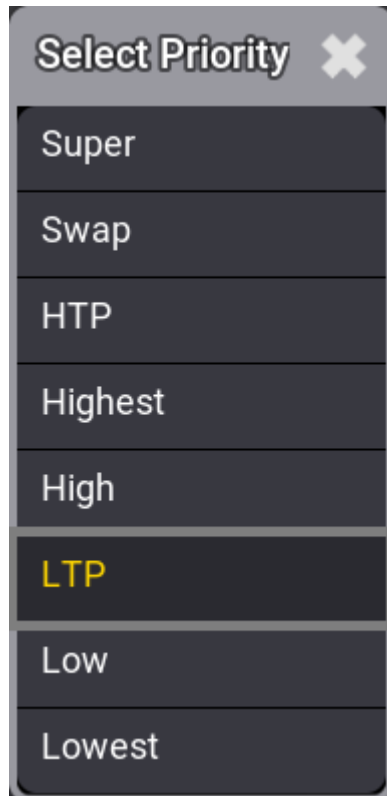
This will send the go command to the object assigned to Executor 101. If this object is a sequence, then the next cue will be triggered.

Sequence Priorities

When several sequences affect the same fixture attributes, then priorities become important. The priority setting is a property of the sequence.

These settings can be opened by pressing the **Assign** key and then one of the executors keys where the sequence is assigned. This opens the **Assign Menu**. On the right side of the menu, there are some buttons. Please tap **Edit Settings**. These are all the settings available for a sequence. The settings can also be opened by tapping **Settings** in the title bar of a sequence sheet. All the settings are discussed in the **Sequence Settings topic**.

The priority setting can be found in the **Playback** group of settings. Tapping it toggles through the different properties, swiping it opens a small select pop-up like this:



Select Priority pop-up

This is the list of possible priority properties. This is a short explanation of the priorities:

- **Super:**
This priority is the LTP priority above any other playbacks and even above the programmer.
- **Swap:**
Intensity is working as LTP with higher priority than HTP.
- **HTP (Highest Takes Precedence):**
The highest intensity value will be used. Other parameters will use LTP.
- **Highest:**
Highest LTP priority - Like LTP but with the highest possible LTP priority.
- **High:**
High LTP priority - like LTP but a higher priority than normal LTP.
- **LTP (Latest Takes Precedence):**
This is the normal LTP priority. The newest attribute value is prioritized over the old value.
- **Low:**
Low LTP - This is a lower LTP priority.
- **Lowest:**
Lowest LTP - This is the lowest possible LTP priority.

LTP is one of the most used priority settings. There are five different levels of LTP priority to be able to give different sequences different priorities.

The list in the pop-up is also prioritized, where Super at the top has the absolute highest priority. This also means that HTP sequences have a higher priority than LTP sequences.

Soft LTP

This option relates to how intensities change from one sequence to another when the values transfer from the original sequence to the new one.

When an attribute has a value from a cue and another cue in a different sequence with the same LTP priority is beginning to send new values to the same attribute, then Soft LTP might influence how the transition from the old value to the new value happens. This is only relevant when the new value is applied by moving the master of the new sequence. In this case, the attribute can jump from the old value to the new value and immediately have the value it should have according to the master position (Soft LTP Off), or it can start to fade from the old value to the new value using the master position as a crossfader (Soft LTP On).



Important:

Both sequences need to be active, and both need to send value to the same attributes. **Auto Start** and **Auto Stop** are turned On by default for sequences. If these properties are turned Off, the sequences must be turned On manually for the SoftLTP function to affect the transition.

If the Master faders are both at 100%, then SoftLTP does not have a function.

Example

1. Patch a fixture with a dimmer attribute.
2. Give the dimmer a 50% value and store this in a new sequence - call this sequence "Original".
3. Give the dimmer a value of 100% and store this as a new cue in a new sequence. Call this sequence "New".
4. Assign an executor fader as the master for sequence "New".
5. Ensure **Auto Start** and **Auto Stop** are active for sequence "New".
6. Open a fixture sheet and make sure it is set to show the **Output** layer.
7. Run the cue on sequence "Original" and have the master at zero for sequence "New". The dimmer value should now be 50%.
8. Now move the master slowly up from zero and see the output value change.

Soft LTP On: The dimmer will fade from 50% to 100% while the master moves up to 100 (fade from the "Original" value to the "New" value).

Soft LTP Off: The dimmer will jump down to the value dictated by the master position and fade up together with the master (jump from the "Original" value to the "New" value based on fader position).

9. Now try to move the fader slowly down to zero.



Soft LTP On: The dimmer fades from 100% to 50% (from the "New" value to the "Original" value).

Soft LTP Off: The dimmer value will fade down to 0% with the master, and when the sequence turns Off then the dimmer will jump to 50% (Fade down with "New" master position - jump to "Original" value when "New" is turning Off).

27.11. Move In Black

Move in black (MIB) is a function where tracking sequences look ahead and preposition attributes of fixtures that are fading the dimmer in from zero to automatically prevent transitions where the fixture would move the attributes into position while the fixture is fading in.

MIB is enabled on a cue part basis by giving the MIB property of the cue part a value that tells the console when and how it should do the prepositioning.



Important:

MIB works with absolute values only.

Several options and properties modify MIB behavior. Two of them are **MIB Fade** and **MIB Delay** timing properties.

MIB Fade is the fade time of attributes that will be positioned by MIB. It is available in different places. There is an order of importance in which MIB fade time to apply:

1. Per cue part
2. Per attribute in a fixture type
3. Global in the show file

When a MIB fade time is specified in the cue part, the individual attribute MIB fade time will be ignored. When the MIB fade time of the cue is set to default, the global MIB fade time will be applied unless an attribute has an individual MIB fade time set.

MIB Delay is the delay time the attributes wait from having the dimmer closed until the MIB fade is performed. The MIB delay time can be specified per cue part or global in the show file. The same ruleset as for MIB fade applies: When the MIB delay time of the cue is set to default, the global MIB delay time will be applied.

When the MIB fade and/or MIB delay is performed between cues, the MIB times specified in the (future) cue part where the dimmer opens again will be applied. For example, a fixture is moving in cue 3 to be ready for cue 5. The MIB times specified in cue 5 is used for the MIB.

Cue MIB Settings

The sequence sheet can display several columns for the different MIB settings. All MIB-related columns can be displayed or hidden by the **MIB Settings** mask within the **Sequence Sheet Settings**. Read more about the sequence sheet in the [Look at Cues and Preferences](#).



- **MIB Preference:**

This is a specification of the suitability of a cue for MIB. The MIB modes Early, UponGo, and Late will prioritize the cue with the highest rated suitability and will choose this cue for executing the MIB. MIB mode Defined does not respect the MIB preference. The preference is a percentage number from 0(never) to 100(best). Edit the cell to type a number or select one of the following preference options:

- **Never(0):**
An MIB will never be performed.
- **Worst(1):**
If there are no other options, then this cue will be used.
- **Bad(25):**
It is not optimal, but better than the two others.
- **Normal(50):**
This is the default value.
- **Good(75):**
This is a better cue than normal.
- **Best(100):**
This is the optimal cue to perform the MIB.

- **MIB Mode:**

Defines how early or late the MIB shall be performed per cue part.

- **Default:**
Performs MIB corresponding to the MIB mode setting of the sequence setting **MIB Mode**.
- **None:**
MIB will not be performed for this cue.
- **Defined:**
A specific cue can be defined in the **MIB Target** column where the MIB is to be performed. The MIB is performed when the specified cue is active.
- **Early:**
Performs the MIB as soon as the dimmer is closed. Typically after the cue transition has finished.
- **UponGo:**
Performs the MIB with the next cue transition after the dimmer has closed. The MIB executes with the cue after **Early** would have triggered the MIB.
- **Late:**
Performs the MIB latest in the cue before the dimmer opens again.

- **MIB Target:**

A specific cue where MIB is performed for this cue part, see MIB Mode above. When setting an MIB target, the **MIB Mode** will be changed to **Defined**, and vice versa when changing an MIB Mode that is not **Defined** will remove the **MIB Target**.



- **MIB MultiStep:**

It is possible to decide whether a phaser where the fixtures are already prepositioned shall keep running with the closed dimmer or if they shall be paused. This can prevent unwanted noise and movement of stepper motors for prepositioned fixtures running a phaser, especially when it would disturb the audience. The two options are:

- **Running:**

A phaser is running with a closed dimmer.

- **Paused:**

A phaser will be prepositioned but does not start running until opening the dimmer.

- **MIB Fade:**

The MIB fade time per cue part. It can be a set time or default. Default takes the global show file MIB fade time, or the attribute MIB fade time.

- **MIB Delay:**

The MIB delay time per cue part. It can be a set time or default. Default takes the global show file MIB delay time or the attribute MIB delay time.

The sequence sheet hides those MIB cells that are not considered for the different combinations of MIB settings or if a cue or cue part is not suitable for MIB.

Global MIB Settings

To change the global MIB Preferences go to **Menu - Preferences and Timings**.

In the **Timings** tab, there is a section called MIB Timings.

This defines the default **MIB Fade** and **MIB Delay** times. This value is input as time. For more information about MIB fade and MIB delay, please read above.

The property **MIB Transition** defines which transition type will be applied to the fade of MIB. Read more about the different types of transitions in the **Cue Timing topic**.

The MIB Transition can only be defined for all MIB fades globally in the show file. It is not possible to define a different transition type for a single MIB fade per cue or cue part.

In the **Cues** tab, there is a section called **MIB Preferences**.

Here it is possible to change the defaults that will be set to a new cue when it is stored.

- **MIB Mode:**

The MIB Mode used when storing a new cue that can execute MIB.

- **MIB Fade:**

The MIB fade time used when storing a new cue that can execute MIB. Default uses the time set in the timing tab.



- **MIB Delay:**
 The MIB delay time used when storing a new cue that can execute MIB. Default uses the time set in the timing tab.
- **MIB Dynamic:**
 The MIB dynamic mode used when storing a new cue that can execute MIB.

For more information about these settings, read above.

Sequence MIB Settings

There are sequence-wide MIB settings within the sequence settings. Read more about the settings in the [Sequence Settings topic](#).

The purpose of the sequence MIB settings is to have the option to overwrite the cue-based MIB settings with the MIB settings.

When setting the MIB to any option except Enabled, the sequence sheet displays the corresponding setting in the MIB Mode column extended by an exclamation mark, for instance, Never!

The MIB Mode setting selects the default MIB mode used when doing a MIB. The MIB Mode per cue or cue part must be set to Default.

Please read the above for more information about the different MIB Mode types per sequence.

MIB Color Indicators

Attribute values will be displayed with special colors within the fixture sheet and the sequence sheet in track mode to show when the MIB is performed.

The tracking sheet view shows when the MIB is performed in this example.

Two versions of the sequence sheet showing MIB data and the Track Sheet

The fixtures need to be ready for cue 5. In cue 5, the MIB Mode is set to "UponGo". This means that when the fixtures fade to 0% in cue 2, they are ready to MIB with the next cue trigger. When cue 3 is activated, they will perform the MIB for cue 5.

The default color indicators are a deep-sea green background and black text color. Read more about the MIB colors in the fixture sheet and other grandMA3 colors in the [Colors topics](#).



27.12. Cue Timing

Each cue part has a lot of timing information.

The default times are used if nothing is defined on cue creation. The default can be changed. Read more about defaults in the [Store Options and Defaults topic](#).

Any cue timings can be changed at any point using the command line or the GUI ([Sequence Sheet](#)).

There are a lot of elements that affect how a fixture changes values, and they can be divided into different groups:

- **General cue times**
- **Preset Timing**
- **Individual Attribute times**
- **Executor Time at point of cue execution**
- **Dynamic changed Rate**

This is also the priority list from lowest to highest priority.

There is also the [cue transition](#) that defines how the values change from one to another.

General Cue Times

There are six different general cue timings. Each has its own column in the Sequence Sheet when condensed timing is turned off - read about it in the [Look at Cues and Sequences topic](#) :

- **Cue In Fade**
This is the fade time used for all intensity values changing from a lower value and going to a higher and any other attribute changing value. It starts when the cue is triggered and after the Cue In Delay has counted down.
- **Cue In Delay**
This is the delay before the fade. This defines a countdown time between the cue is triggered and when the fade should begin. The In Delay affects the In Fade. The default value is 0, meaning that there is no delay.
- **Cue Out Fade**
The out fade is used by intensity values fading from a higher value to a lower. It is executed after the Cue Out Delay. The default value is the in fade, meaning that it is the same as whatever the cue in fade is.
- **Cue Out Delay**
This is the delay time for the Out Fade. It can be used to delay when intensity values should start to fade down in value. The default value is in delay, meaning it is the same as the Cue In Delay value.
- **Snap Delay**
Some attributes are defined to Snap. This means that they do not fade from one value to another. They change values as fast as possible. This can make sense for attributes like gobos. This delay is used to delay when the snap is performed. It makes it possible to have the fixture fade out before snapping to a new gobo. This delay affects all snap attributes stored in the cue. It can be overwritten by individual attribute timing.



- **Command Delay**

This delays the execution of a command. This is the only place where this delay can be defined. There is no individual timing for this.

These values can be changed using this syntax:

Cue ["Cue_Name" or Cue_Number] CueInFade [New_Cue_Time]

Just use the relevant keyword: **CueInFade**, **CueOutFade**, **CueInDelay**, **CueOutDelay**, **SnapDelay**, or **CommandDelay**.

There are two more options for setting the cue fade and delay. They are called **CueFade** and **CueDelay**. They can be used to set the four cue fade and delay times mentioned above. They are used to set the in and out time using just one keyword.

The in and out time is separated by a slash. The time before the slash is the **In** time. The time after the slash is the **Out** time. It is not necessary to specify both in and out. The slash can be used to set just one of them. The desired time has to be on the correct side of the slash.

If only one time is set without any slash, then the in time gets the actual value, and the out time is linked to the in time - it is technically set to time "None". The result is that, for instance, the fade in and fade out will be the same.

For example, setting the in fade to 5 seconds and the out fade to 8 seconds in cue 3 could be done with the following keystrokes:

Cue 3 Time 5 / 8 Please

The command line would read:

```
MA User name[Fixture]> Cue 3 CueFade 5/8
```

If only one time is given, both in and out will use the time. For instance,

```
MA User name[Fixture]> Cue 4 CueFade 7
```

This makes cue 4 use 7 seconds to both fade in and out.

They can also be used to address only the in or out time by adding the slash and a number to the relevant side. For instance, setting the CueOutFade to 3 and the CueInDelay to 1 on the currently active cue:

```
MA User name[Fixture]> CueFade /3 CueDelay 1/
```

Both CueFade and CueDelay can be addressed using the **Time key**. Read more about delays by following the links above to the keywords.

Preset Timings

Each preset type has its own fade and delay time. There are columns for each of these in the Sequence Sheet.

These times are used by all fixtures changing the values of that preset type. Overwriting the general cue timing for that preset type.



Important:

Intensity values changing to a lower value are not affected by the "Fade Dimmer" time. They are controlled by the "Cue Out Fade".

The preset type fade and delay default is general cue timing. This means they are the same as the time for the Cue In Fade and Cue In Delay.

Individual Attribute Timing

Each individual attribute can have an individual fade and/or delay time. This is called individual time. These individual times are selected using the **programmer** and added to the cue when it is **stored**. They can also be edited in the sequence sheet while in **tracking mode**.

There are two columns in the sequence sheet called **Indiv Fade** and **Indiv Delay**. They display the highest of the individual times. The **Indiv Duration** column shows the complete individual time (delay + fade).

Executor Time

There is an **Executor Time** master fader. It can be seen and changed in the **Master Controls pop-up**. It can also be **assigned to physical controls or executors**.

If this function is turned On, the set time will be used instead of the stored fade and delay timing. The **Executor Time** faders position is registered when the cue is triggered. The fader can be moved afterward without this affecting the running cue fade. Read more in the **Time Control topic**.

Sequences can be protected from this function by turning Off the **Use Executor Time** setting in **Sequence Setting**.

Rate Time

The **Rate** allows the timings to be dynamically adjusted while the cue fade runs. It does not change the stored times. It simply adjusts the time to be faster or slower.

Moving the fader up makes the fade faster, making the stored times appear smaller. Pulling the fader down makes it go slower, extending the stored times.

Times affected by the rate has an asterisk (*) in front of the time. If the rate is all the way down, the times will show ***Stopped**. If the rate is all the way up, the times will show ***0**.

An executor with the rate function shows the rate as a percentage in the executor label. If the rate is exactly 100%, then the label shows **1:1**. If the rate is at maximum, the label show **1:∞**. If the rate is at minimum, the label show ***Stopped**.

The rate can be reset using the **Rate1 keyword**.

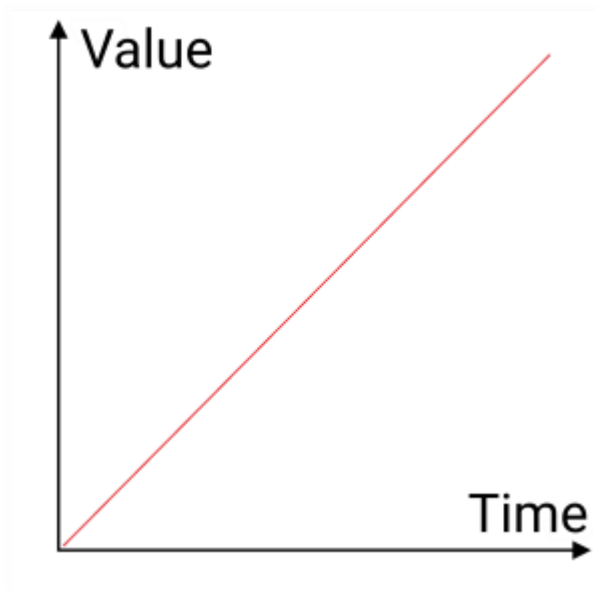
Cue Transition

The cue part property **Transition** allows modifying the path values used during the fade from one cue to another. The transition can be set per cue part.

There are nine different transition forms available. They are visualized with these images.

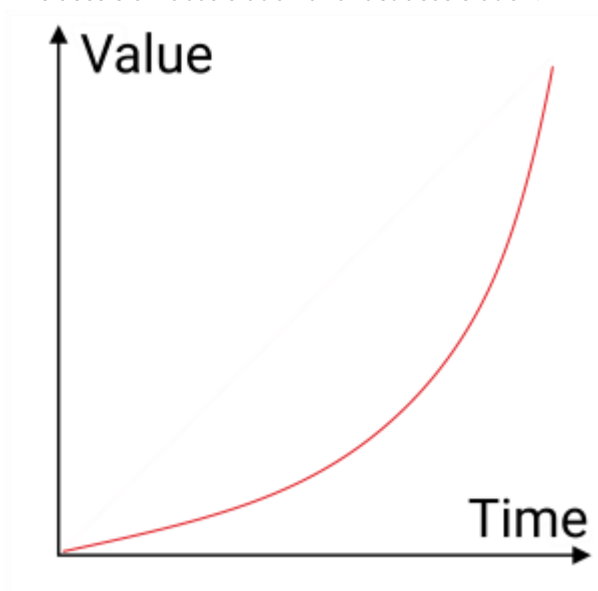
- **Linear:**

This is the default transition. There is no acceleration or deceleration of the values.



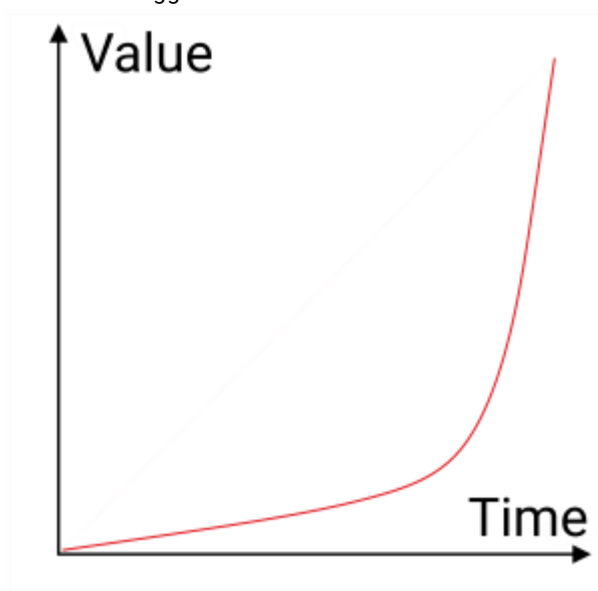
- **Slow:**

This uses slow acceleration and fast deceleration.



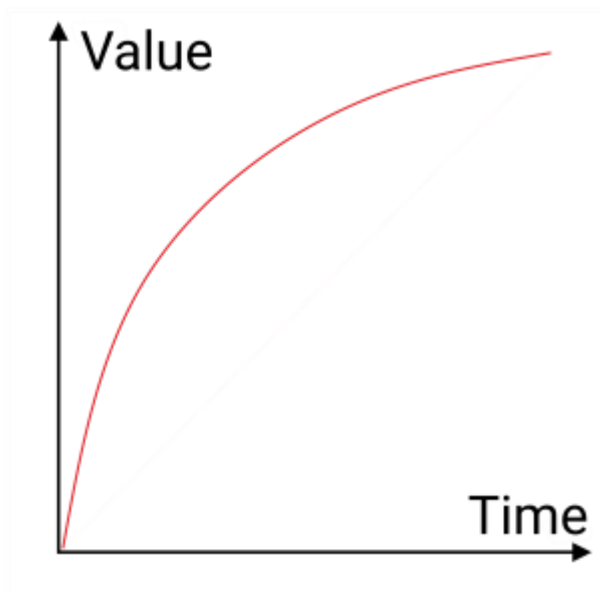
- **Slow+:**

This is an exaggerated version of **Slow**.



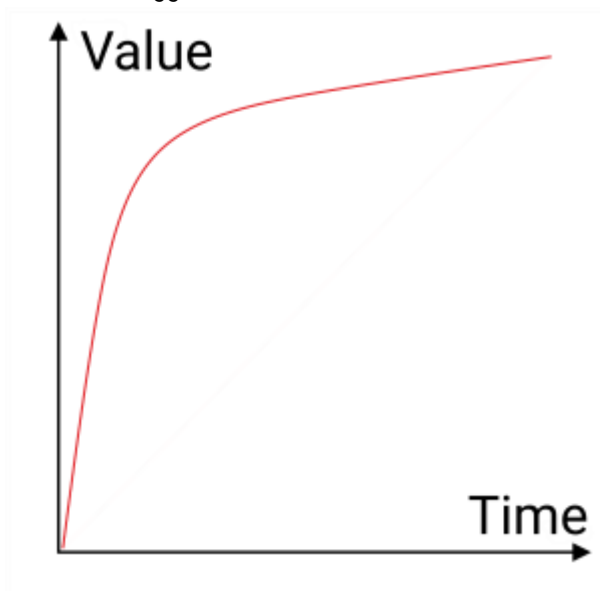
- **Fast:**

This is a fast acceleration and slow deceleration.



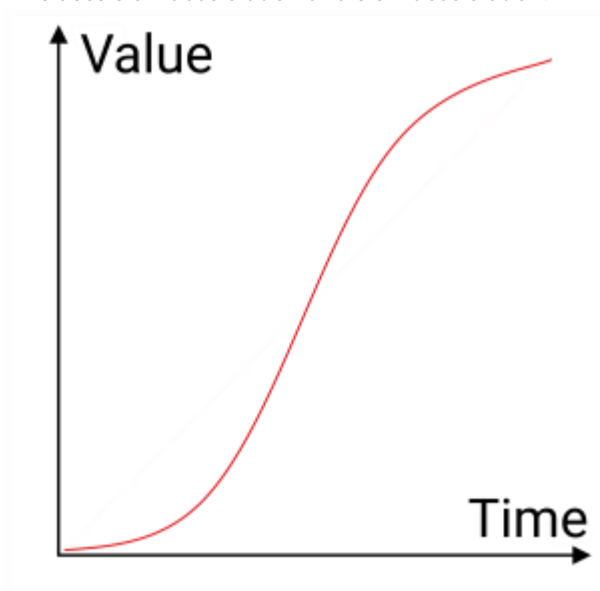
- **Fast+:**

This is an exaggerated version of **Fast**.



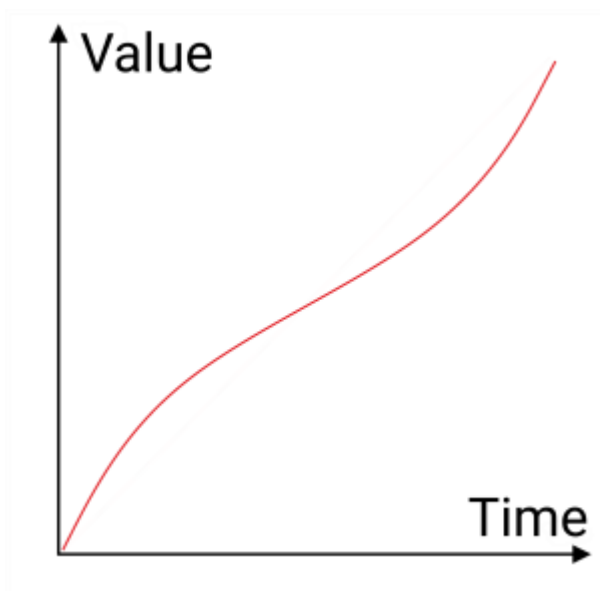
- **SCurve:**

This uses slow acceleration and slow deceleration.



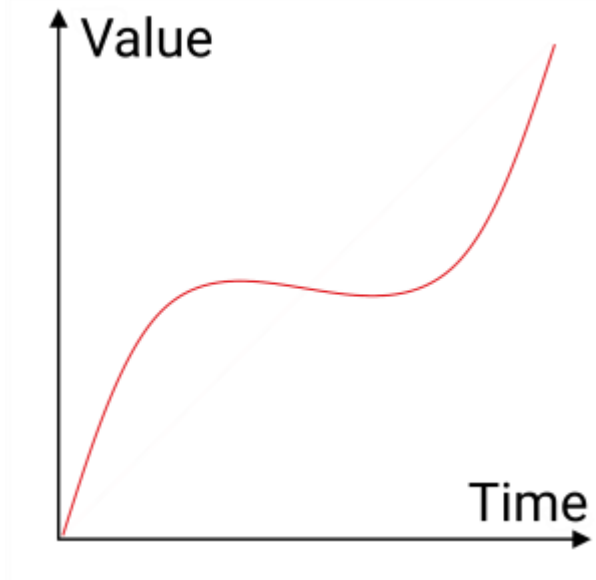
- **Swing-:**

This uses fast acceleration and fast deceleration.



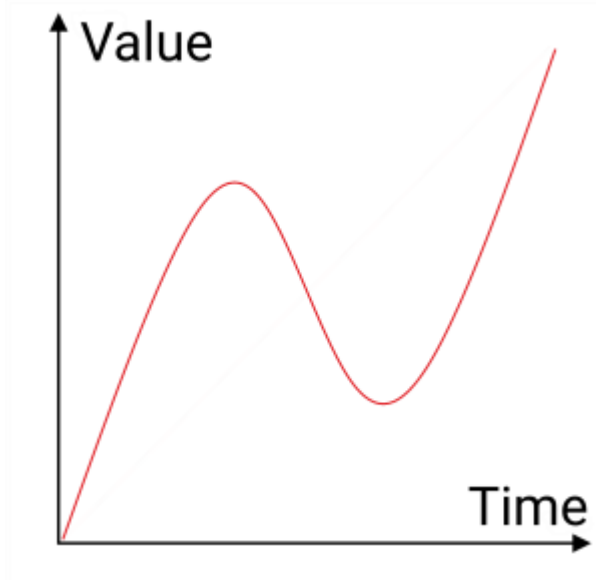
- **Swing:**

This is a slight exaggeration of **Swing-**. The values change slows down in the middle of the transition.



- **Swing+:**

This is an exaggeration of **Swing**. The values go backward in the middle before accelerating forward again.



27.13. Renumber Cues

Cues can be renumbered. This cannot be used to change the order of the cues.

Single cues can be renumbered but it can also be a range of cues.

It is not possible to give the same cue number to two or more cues - they must be unique.

Setting the cue number to a previous number that is already used is prevented by the system.

Entering a cue number that is used later in the sequence, all following cues will be renumbered until no collision with existing cue numbers occurs.

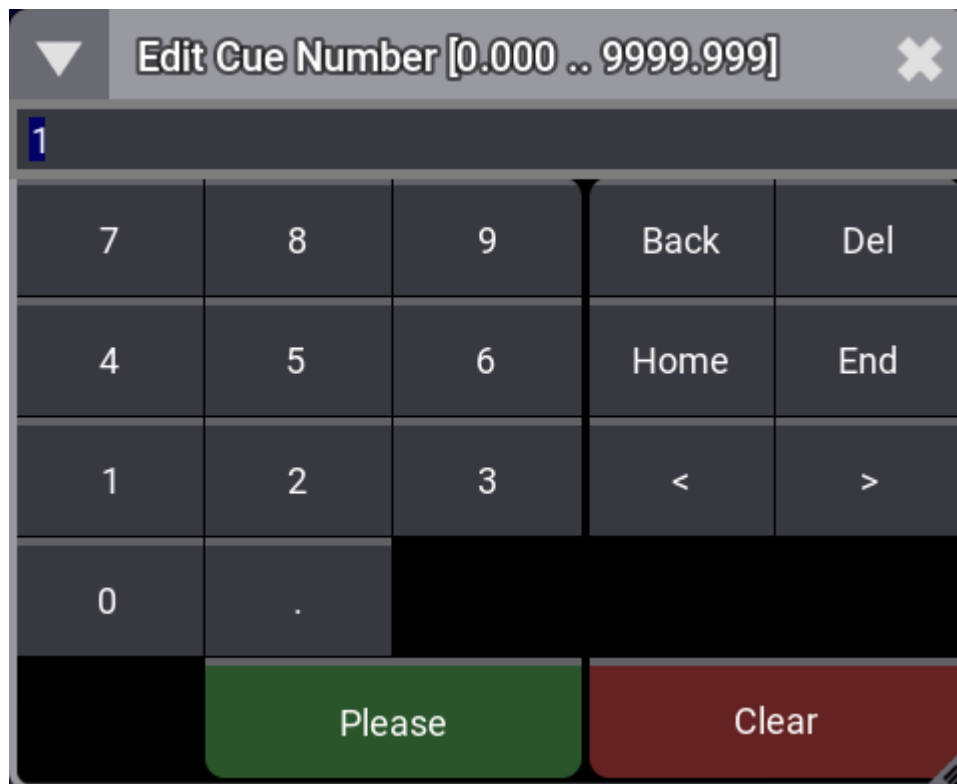
Requirement:

Have a sequence with cues.

Procedure:

1. Open **sequence sheet window**.
2. Select the cues numbers that are to be renumbered using the number column in the sheet.
3. Press **Edit** and tap inside the cue selection.

This opens an Edit Cue Number pop-up.



Edit cue number pop-up



4. Type the new cue number.

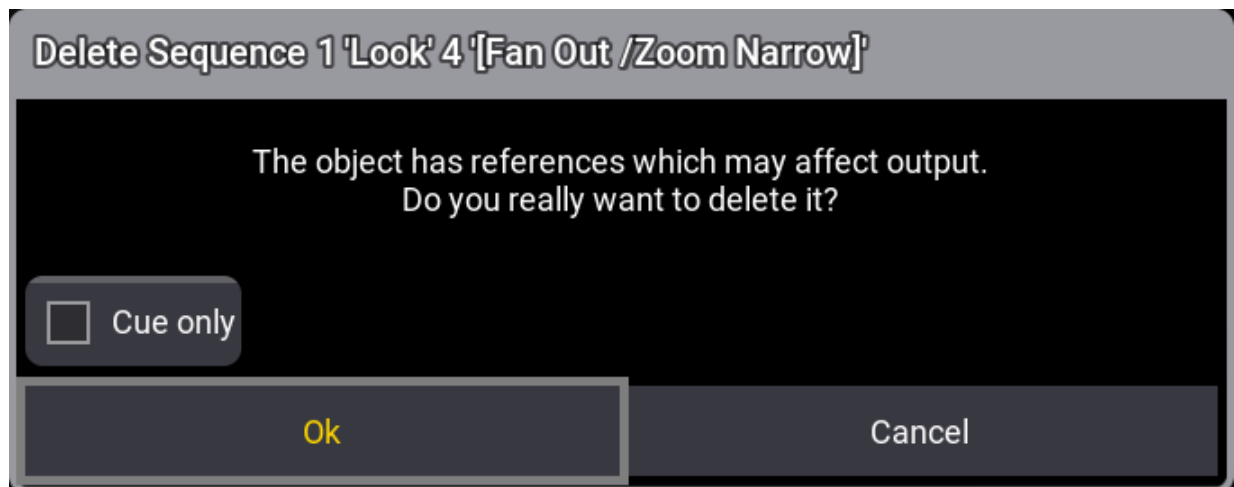
If several cues are to be renumbered then use the lasso selection when tapping the cue numbers.

27.14. Delete Cues

Cues can be deleted using the **Delete keyword**.

This can be accessed using the command line input or the **Delete key**.

Deleting a cue opens a pop-up asking for confirmation.



Delete Cue pop-up

There is a small **Cue Only** checkbox. Checking this deletes the cue using the **cue only** principle.

Tap **Ok** to delete the cue and close the pop-up.

Tapping **Cancel** closes the pop-up and does not delete the cue.

Deleting the last cue in a sequence does not open the pop-up; it simply deletes the cue.



28. What are Executors

The executors are handles and controls to other objects.

They are often used to control sequences, but they can control other objects.

The idea is that an object (for instance, a sequence) is assigned to the executor. The executor can then control the object. Several executors can control the same object.

A sequence assigned to an executor is running cues from the sequence pool. In essence, the executor is manipulating or sending commands to the sequence in the pool.

Executors are physical keys (executor buttons), knobs (executor knobs), and faders (executor faders) on the grandMA3 hardware. They can also be represented as on-screen virtual executors - these can be viewed and operated in the **Playback Window** (see below).

Read more about the physical executor hardware in the **Executor Elements topic**.

There are 4 rows of executors. The bottom row is numbered from 101 up to 190. This row only has a single executor button. The row above is numbered from 201 up to 290. This row has an executor button and an executor fader. The next row is from 301 up to 390. This row has an executor button and a rotating executor knob. The top row is from 401 up to 490. This row also has an executor button and a rotating executor knob.

The Xkeys are also executors. The executor buttons labeled X1 to X8 are executor 291 to 298. The executor buttons labeled X9 to X16 are executors 191 to 198.

Executors are organized in wings. There are vertically 15 columns of executors on a wing. The columns are organized in sections of 5. There are 60 (15 columns x 4 rows) executors on a wing. There are 6 wings for a total of 360 (6 wings x 60 executors) executors plus the 16 from the Xkeys - the total is 376.

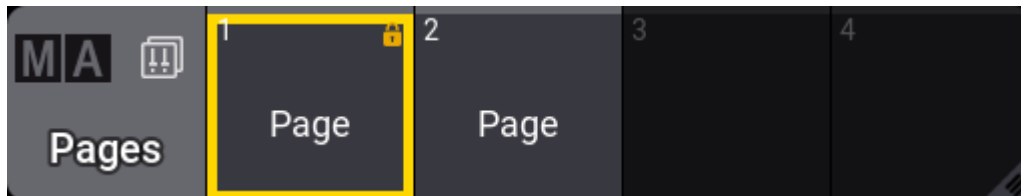
The physical device might have fewer executors than the software's amount. For instance, the grandMA3 compact console has 2 x 5 columns of executors.

The executors can also be used from the command line using the **Executor keyword**.

Executor Pages

The 376 executors are just on one page. It is possible to create up to 9 999 executor pages in each **Data Pool**. Each page has its own setup of executors. Executors that are active on one page are still active when the selected page is changed to another.

The executor pages can be seen and labeled in the **Page Pool** - it can be **created as a window**.



Page pool with page 1 selected.

The page pool is also available as a pop-up that can be opened by tapping the middle part of the page selector.



Tap the middle part of the page selector to open the page pool pop-up

There are no functional differences between the pool as a pop-up or window.

Pages are automatically created when the **Page+** and **Page-** keys are used to change through the pages. The up and down arrows in the page selector can also be used to change the page, just like the physical keys.

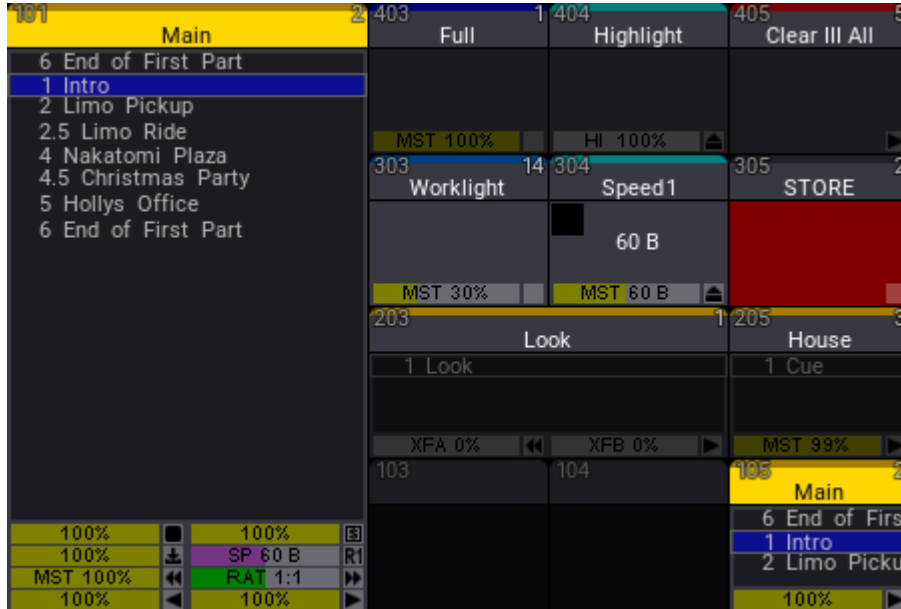
A page can also be created using the following syntax:

Store Page [Page_Number] ("Page_Name")

The page name is optional in the syntax above. Read more in the [Page keyword](#) topic.

Executor Labels

The letterbox screens above the executors show the labels for the executors. The bottom of the left command screen shows the labels for the Xkeys.



Example of the playback bar for the first section of 5 x 4 executors

The executor labels above the executors are a part of the Playback Bar. Read more about them in the [Playback bar topic](#).

The labels display how the encoders are grouped, what object they control, and what handle function each executor has.

Read more about how to change all these things in the [Assign Object to an Executor topic](#).

The labels take appearance and scribbles from the object assigned. For example, if a sequence has a special appearance, this appearance is also displayed in the playback bar.

Playback Window

The playback window is a virtual representation of the physical executors.



Playback window showing all four rows and both labels and executor handles

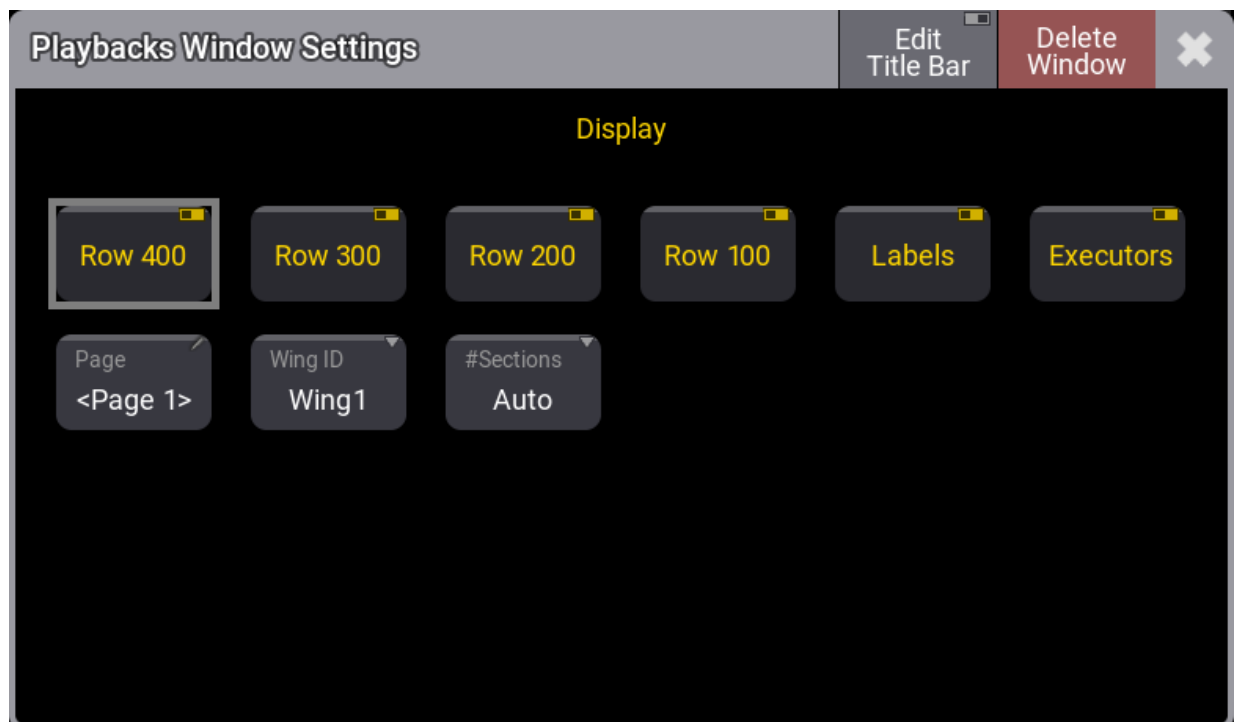
The window can automatically adjust the number of visible executors based on the window's width and the selected number of sections. This means that a window can minimum have 5 x 4 executors and a maximum of 15 x 4 (one wing). The window's minimum size to display a section of 15 executors is 10 squares wide on a screen.

The virtual executors can be operated on the screen.

Playback Window Settings

The settings allow for customization of the window.

The settings can be accessed by tapping the MA logo in the upper left corner.



Playback window settings

There are toggle buttons that can show or hide the four executor rows. There are also settings to show or hide the labels and the executor handles. This makes it very flexible. For instance, it could just show the labels for the 200 row of executors or just the executor handles for the 100 row executors.

The **Page** and **WingID** settings are also described in the [Window Settings](#) topic.

The **Executors** setting toggles the displayed on-screen copies of the relevant faders, knobs, and keys.

In addition, **Labels** shows or hides the executor or Xkeys labels.

The **#Sections** setting selects how many sections of five columns the window should display. The properties are:

- **Auto:**
The window displays as many sections as possible based on the width of the window and a minimum width of each column.



- **1:**
The window only shows the first section of executors - 5 columns.
- **2:**
The window shows the first and second sections of executors - 10 columns.
- **3:**
The window shows all three sections of executors in the wing - 15 columns.

The properties with a specified number of sections will always show the selected amount, even when the window width makes it hard to use.

28.1. Assign Object to an Executor

Many objects can be assigned to an executor. The executor is a physical key, fader, or knob that controls the assigned object. The physical devices can also be represented as on-screen controllers.



Quick Steps:

1. Press **Assign**.
2. Tap what should be assigned or press the relevant keys.
3. Press the executor where it should be.

These are the simplest steps to assign something to an executor - read below for details.

Assigning Objects Using Keys and Pools

It is easy to assign something to an executor.

Press **Assign** followed by the desired object and then the executor to which it should be assigned.

Here are three variations on how it works. The examples use sequences, but it can be any of the allowed types:

Example 1

Using only the keys to assign sequence 3 at executor number 105 on the current page, type:

Assign Sequ 3 At MA + X16 | Exec 1 0 5 Please

Example 2

It is also possible to use a combination of keys and pools.

Having a pool visible on one of the screens makes it possible to combine key presses with pool selection.

1. Tap and swipe out of the sequence pool object that should be assigned.
2. Swipe to the **Assign** option and release the screen.
3. Press one of the keys associated with the desired executor.

Example 3

Pressing the keys puts keywords into the command line. This means it can also be typed as a command line input.



MA User name[Fixture]> Assign Sequence 4 At Page 2.301

This command will assign sequence 4 to executor 301 on executor page 2. The page keyword needs to be used when addressing executors on specific pages. The page needs to exist before it can be addressed.

Assign Objects Using the Assign Menu

The **Assign** menu offers a visual approach to assigning something to the executors. Use the assign menu by selecting the executor first and then selecting the object.

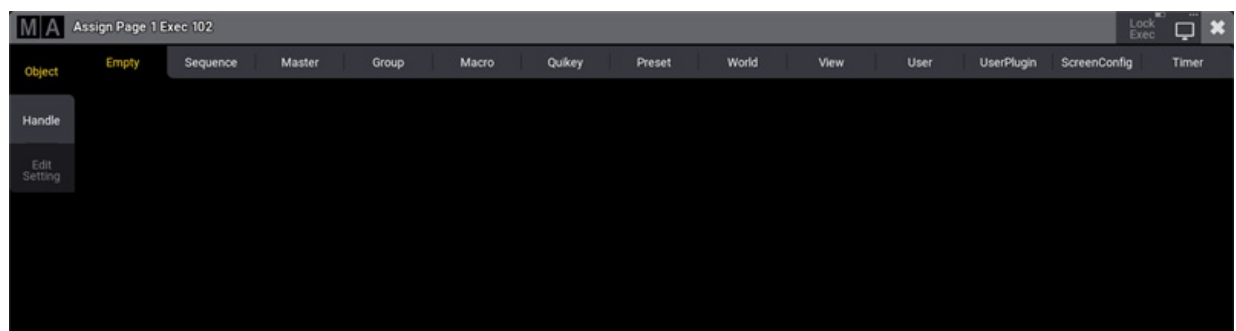
There are two main ways to open the Assign Menu.

- Press **Assign** followed by any of the keys associated with the desired executor.

-Or-

- Tap the executor label in the **Mini Executor Bar** (do not do a long press - it opens the editor instead).

This is the Assign menu:



Assign Menu - Object page

The title bar has a toggle button called **Lock Exec**. This can be used to lock the executor from changes. It does not lock the executor from playing back or performing other functions normally; the lock only prevents making changes to the executor.

This is the **Object** page of the Assign menu. This page is selected by tapping **Object** on the right side.

The top has several tabs. One for each object type that can be assigned to the executor and a special one used to select an **Empty** object. The other tabs open a selection list. The list will contain the possible objects of that type.

The different types are:

- **Sequence**
- **Master**
- **Group**
- **Macro**
- **Quickey**
- **Preset**
- **World**
- **View**
- **User**



- **UserPlugin**
- **ScreenConfig**
- **Timer**

Tap **Sequence** to open the list of possible sequences. It could look like this:

Object	Empty	Sequence	Master	Group	Macro	Quikey	Preset	World	View	User	UserPlugin	ScreenConfig	Timer				
Handle	Lock	No	Name	Scribble	Appearance	Note	Auto Start	Auto Stop	Auto Fix	Auto Stomp	Release First Cue	Soft LTP	Command Enable	XFade Mode	XFade Reload	Swap Protect	Kill Protect
1 (3)			Look				Yes	Yes	No	No	Yes	Yes	Yes	AB	No	No	No
2 (3)			Main				Yes	Yes	No	No	Yes	Yes	Yes	AB	No	No	No
3 (3)			House				Yes	Yes	No	No	Yes	Yes	Yes	AB	No	No	No
4 (3)			BlueLights				Yes	Yes	No	No	Yes	Yes	Yes	AB	No	No	No
5 (3)			Sequence 5				Yes	Yes	No	No	Yes	Yes	Yes	AB	No	No	No
			New Sequence														

Assign menu in the sequence object tab

Each allowed type that can be assigned to an executor provides a list of the available objects. The **DataPool** in the title bar makes selecting an object from a different data pool easy.

Select the desired object by tapping it.

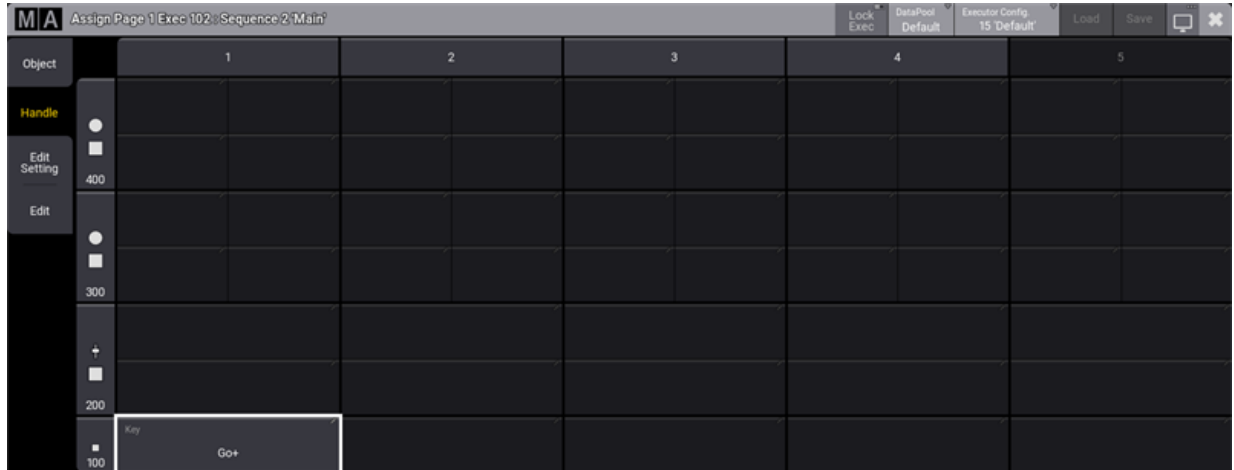
The default handle configuration is applied to the executor. This, including executor expansion, can be defined on the handle page.

Change Key Function and Executor Size

When the executor has something assigned, changing the functions assigned to the executor keys, faders, and knobs is possible.

If continuing to work in the Assign menu, tap **Handle** on the left side.

It could look like this:



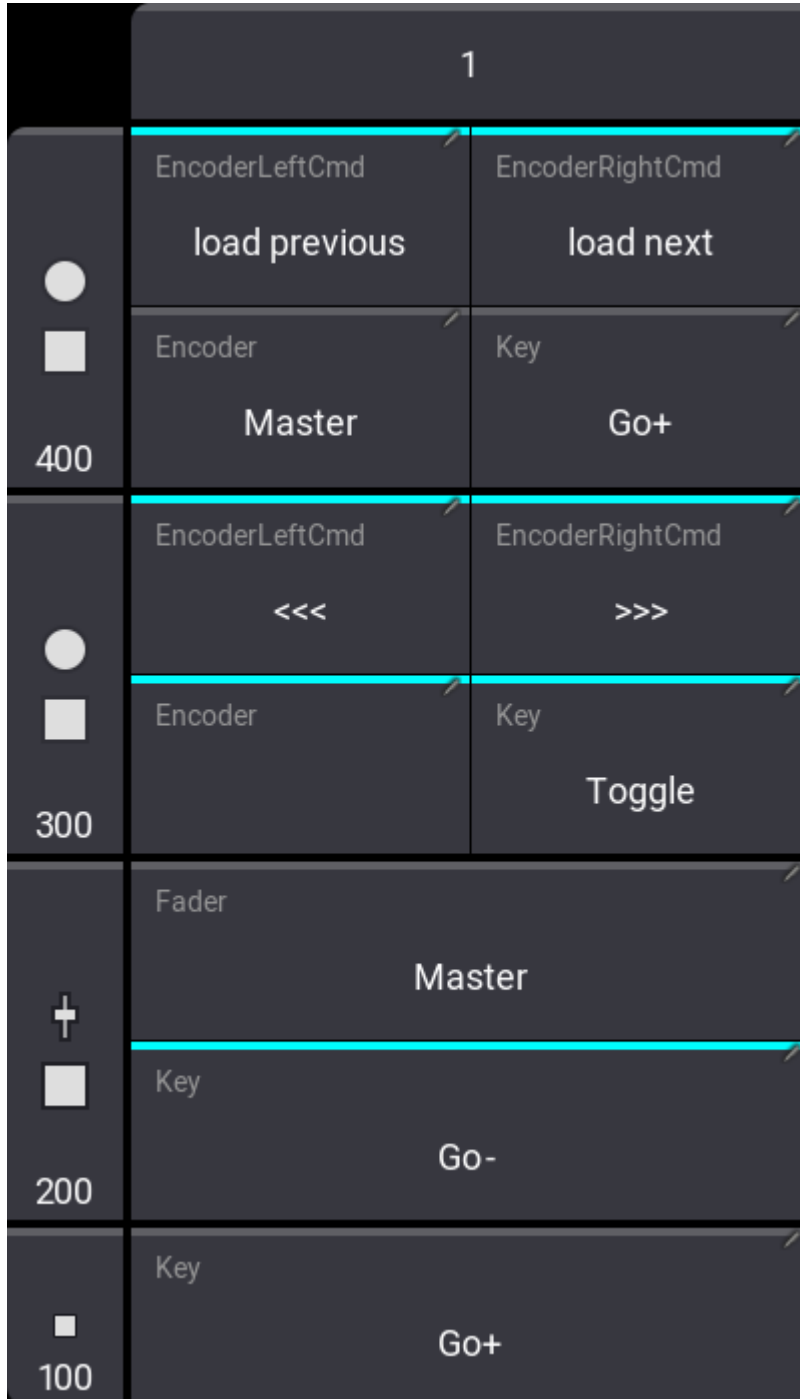
Assign menu on the Handle page

Executors can be expanded if the executors above and to the right are empty. They are grouped into sections of 5 in width, and this grouping cannot be crossed. For instance, executors 104 and 105 cannot be extended into executor 106.

They can also be extended upwards if there is available space. The example image above shows Executor 102 with space above and to the right. This executor can be extended to a width of 4 columns and a height covering all four executor rows.

It is expanded by tapping the column numbers on the top and the row numbers on the left. When it is expanded, then more buttons get available for the executor.

The 100-row only offers a single executor button. The 200-row has a button and a fader. Both the 300- and 400-rows have a button and an encoder knob. If an executor is extended to cover all four rows, then it could look like this:



Executor with all four rows in height

The keys, faders, and encoders can have different functions. The options might vary depending on the object assigned to the executor.

Tap any available button to get a small select pop-up with the available options.

The **Encoder Left Cmd** and **Encoder Right Cmd** open a pop-up that allows any command to be typed. There is even an option for different commands if the MA key is pressed while the encoder knob is turned.



Hint:

The encoder commands are mutually exclusive from the normal encoder function. In the example image above, there is both an encoder function and a command function. The encoder commands have a higher priority and overwrite the normal encoder function.

The normal key options are:

- **Empty:**
The key has no function.
- **At:**
Used with presets. Performs an **At command** on the assigned preset. Learn more about presets on executors in the **Use Presets topic**.
- **<<< (GoFastBackward):**
This jumps one cue back without cue timing.
- **>>> (GoFastForward):**
This jumps one cue forward without cue timing and will not trigger other cues.
- **Black:**
This turns off the intensity of the executor as long as the button is pressed.
- **DoubleSpeed:**
Pressing this doubles the resulting speed of the speed master.
- **Call:**
Used with screen configurations. Performs a **Call command** on the assigned screen configuration.
- **Flash:**
This turns the executor on and sets a virtual intensity master at 100, as long as you have the button pressed. When the button is released, it will return to the status before it was pressed.
- **Go+:**
This executes a "Go". It uses fade and delay times.
- **Go-:**
This fades backward using fade and delay times.
- **Goto:**
This opens a pop-up where a cue can be selected. Tapping a cue in the pop-up immediately performs a **Goto command** on that cue. This means that the selected cue is now active.
- **HalfSpeed:**
Pressing this halves the resulting speed of the speed master.
- **Kill:**
This sequence playback action turns this sequence On and all other sequences Off. Sequences can be protected from the Kill action in the **Sequence Settings**.
- **LearnSpeed:**
This sets the speed. By pressing it at least two times, it automatically adjusts the speed.



- **Load:**
This opens a pop-up that lists the cues in the sequence. Tapping a cue performs a **Load command** on the cue, which will perform a go to the selected cue when the next Go command.
- **LogIn:**
Used when a user is assigned. Performs the **LogIn command** for the assigned user.
- **Off:**
This turns the executor Off.
- **On:**
This turns the executor On.
- **Pause:**
This holds an active fade and/or delay.
- **Rate1:**
This resets rate fader.
- **Select:**
This selects the executor.
- **SelectFixtures:**
This selects the fixtures used in the object assigned to the executor.
- **Speed1:**
This resets the speed fader to the default speed.
- **Swap:**
Swap temporarily overrides the master level of executors to full and sets all other master levels to zero. Executors can be protected against the Swap in the **Sequence Settings**.
- **Time:**
This toggles the time function for the executor. It overwrites the stored cue part times when it is On.
- **Temp:**
This turns the executor on as long as it is pressed. The temp function uses the fade times from the cues and the level set by the intensity master.
- **Toggle:**
This turns an active executor Off and an inactive executor On.
- **Top:**
This fades to the first cue in an assigned sequence.

These main functions are keywords - they can be found in the **All keywords topics** with detailed descriptions of the keywords.

The functions can also be assigned using the keys or commands. Not all functions have a physical key that can be used; it can be done by writing the command using the keyboard.

Key Example

If the desired function has a physical key, assigning it to an executor key is easy. Press **Assign**, the function key you want, and the executor key where the function should be.



Example 1

For instance, to assign the **Off** function to the key associated with executor number 101, you need to press the following keys:

Assign **Off** **-** (the executor key in the lower-left corner on consoles)

Example 2

Assign the **Fix** function to an executor key

Assign **MA** + **Pause** [desired executor key]

Command Example

Functions can be assigned using the command line.

This is the general syntax:

Assign [Function] at [Location]

The location must be a specific physical key associated with an **Executor Page**.

To assign the Pause function to the top key associated with executor 5 on executor page 8, the following command could be typed:

```
MA User name[Fixture]> Assign Pause At Page 8.405
```

Change Fader Function

It might be possible to change the function of the fader. This depends on what type of object an executor has. They are changed just as the executor keys - by tapping the onscreen representation in the assign menu (see above).

The **Select Fader** pop-up lists the possible options.

- **Empty:**
Fader has no function.
- **Master:**
Controls the intensity. The keyword is **FaderMaster**.
- **X (CrossFade):**
Crossfades between two cues. Current cue and next cue. The current cue will change when the fader reaches the other end position from where it started. The keyword is **FaderCrossFade**.



- **XA (CrossFadeA):**
Is the first of a two-fader manual crossfade between two cues. Works along with XB. The crossfade function can be changed in the [Sequence settings](#). The keyword is [FaderCrossFadeA](#).
- **XB (CrossFadeB):**
Is the second of a two-fader manual crossfade between two cues. Works along with XA. The crossfade function can be changed in the [Sequence settings](#). The keyword is [FaderCrossFadeB](#).
- **Temp:**
Crossfades the cue on when pulled up, and off when pulled down. The keyword is [FaderTemp](#).
- **Rate:**
Divides the fade and delay time in a sequence by the value of the fader. If **Speed from Rate** is on, it is also valid for phaser speed stored in cues - see [Sequence Settings topic](#). The keyword is [FaderRate](#).
- **Speed:**
Controls the phaser speed in a cue. The keyword is [FaderSpeed](#).
- **Time:**
Sets the time for the executor time overwrite. The keyword is [FaderTime](#).

The above functions are described in more detail in each of their keyword topics.

The functions can also be assigned using the command line.

The syntax is the same as the key functions. Except for "empty" - see examples below.

Examples

Example 1

To assign the rate function to executor number 209 on the current page, you will need to type the following command:

```
MA User name[Fixture]> Assign FaderRate At Executor 209
```

Example 2

If you want to assign the "Empty" function to the executor fader, then it is necessary to use a different syntax. This example assigns empty to the left-most fader on page 1:

```
MA User name[Fixture]> Set Page 1.201 Property "Fader" "Empty"
```

Save the Key and Fader Assignment


The current assignment of key and fader functions can be saved into a pool of different configurations. Read more in the [Executor Configurations topic](#).



28.2. Executor Configurations

Executor key and fader configurations can be saved and reused. Read the **Assign Object to an Executor topic** for information on changing the assignment.

The configurations are stored in an Executor Configuration pool. This can be created as any other window. Read the **Add Window topic** to learn how.

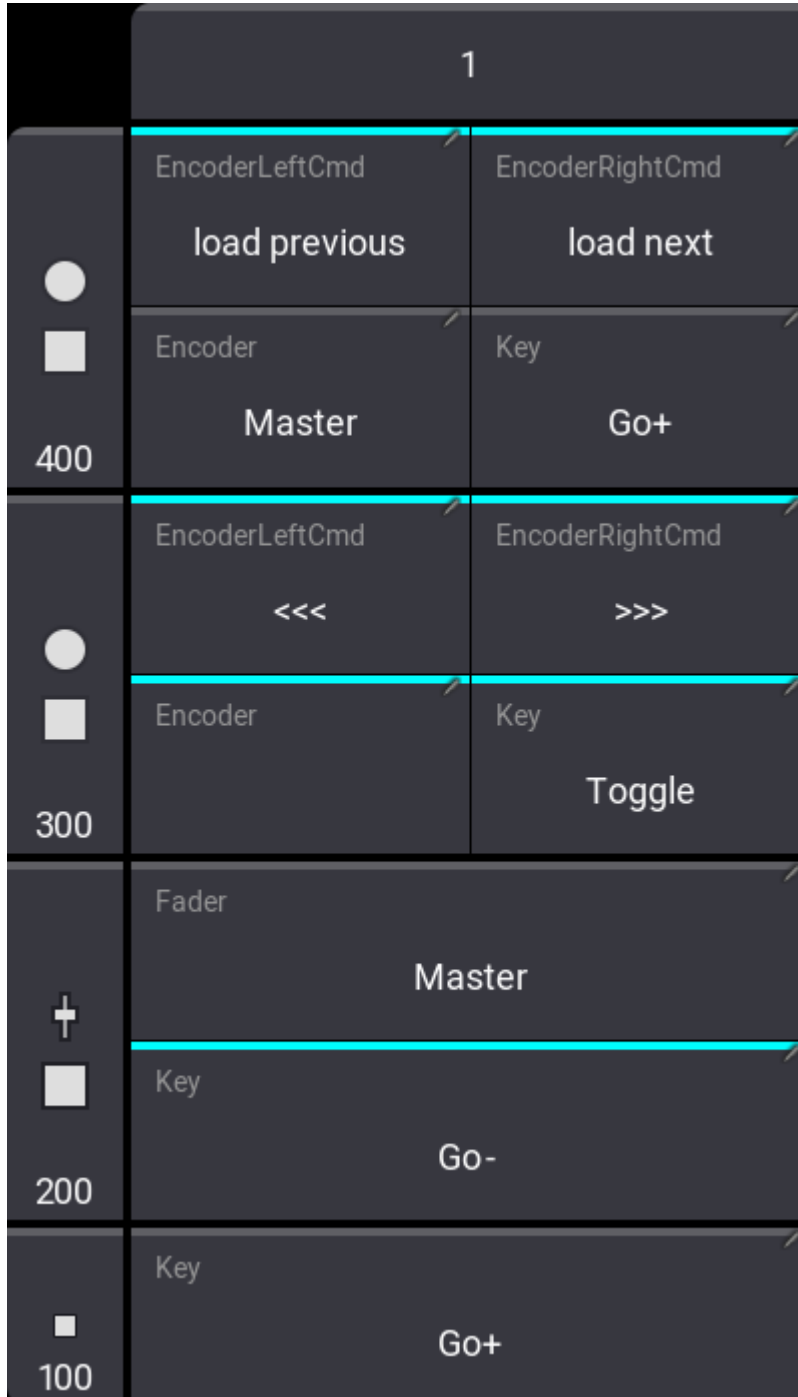
 Executor Configs	1 Default Sequence	2 Default Macro	3 Default View	4 Default World	5 Default Group
6 Default Preset	7 Default Plugin	8 Default User	9 Default Sound	10 Default Screen Config	11 Default Timer
12 Default Master	13 Default Speed Master	14 Default Playback Master	15 Default	16 Extended Fader	17 Toggle Button
18 Temp Button	19 Flash Button	20 Rotary Knob	21	22	23

Executor Configuration pool with a selected configuration

Each object type that can be assigned to an executor has a default executor configuration used if nothing changes. This is described below.

The selected pool object is used as the default for sequences.

This configuration can be changed in the Assign menu.



Key and Fader configuration with unsaved changes

In the example above, there are several Keys and Encoders that have a different assignment than the original saved configuration. The different assignments are marked with a cyan color bar.



Restriction:

Be aware that the image above shows an encoder with both an encoder function and encoder turn commands. These are mutually exclusive functions, and the commands have higher priority. The normal encoder function is ignored.

In the title bar of the assign menu, there are three buttons: Executor **Config.**, **Load**, and **Save**.



Title bar of the assign menu

Executor Config. shows the currently selected configuration, and tapping it opens a small select pop-up where any existing executor configurations can be selected or a new one can be created.

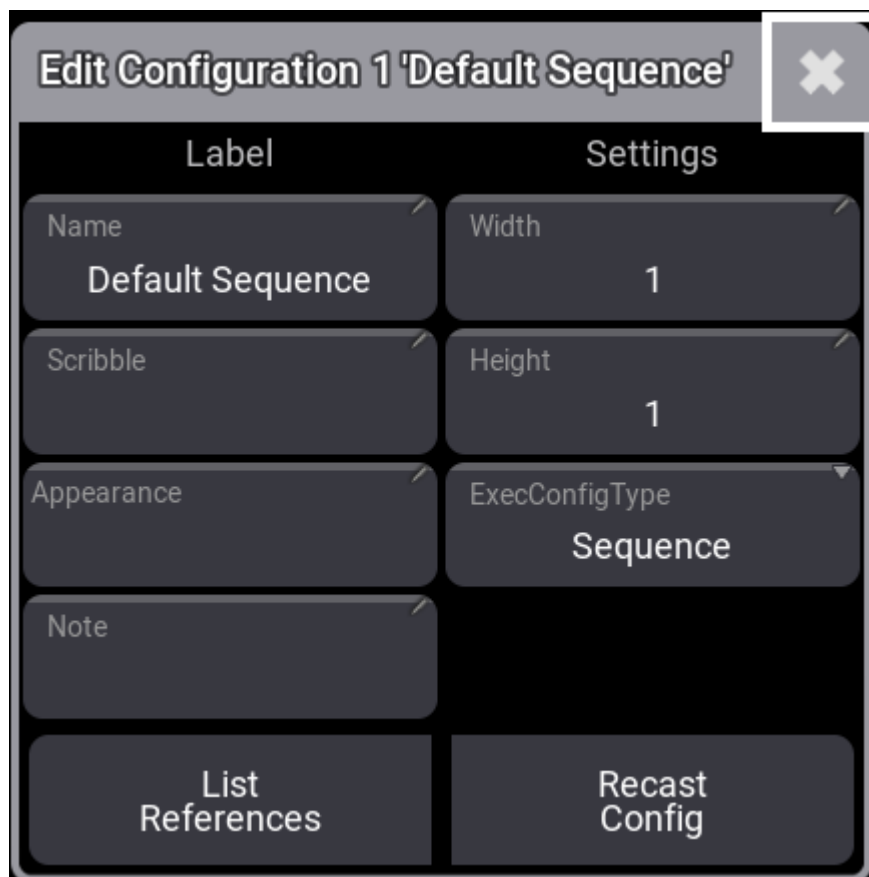


Hint:

Creating a new executor configuration this way will discard all changes that were made until then.

Load and **Save** are used to load the selected configuration onto the keys and faders. **Save** can be used to save the currently assigned functions to the selected configuration.

Editing the configuration pool object opens an editor like this:



Edit configuration pop-up for the first pool object

Here it is possible to change the name of the executor configuration.

This is also where the executor configuration's width and height are set. It is not shown here, but the configuration also knows the starting row for the configuration. This means that a configuration of 1x1 can be different for the rows.

The `ExecConfigType` defines what type of object the setting is relevant for. Tapping the button opens a small pop-up with all the possible objects.

The executor configuration can also have an appearance and scribble assigned. This is only visible in the pool.

A note can be added to the executor configuration.

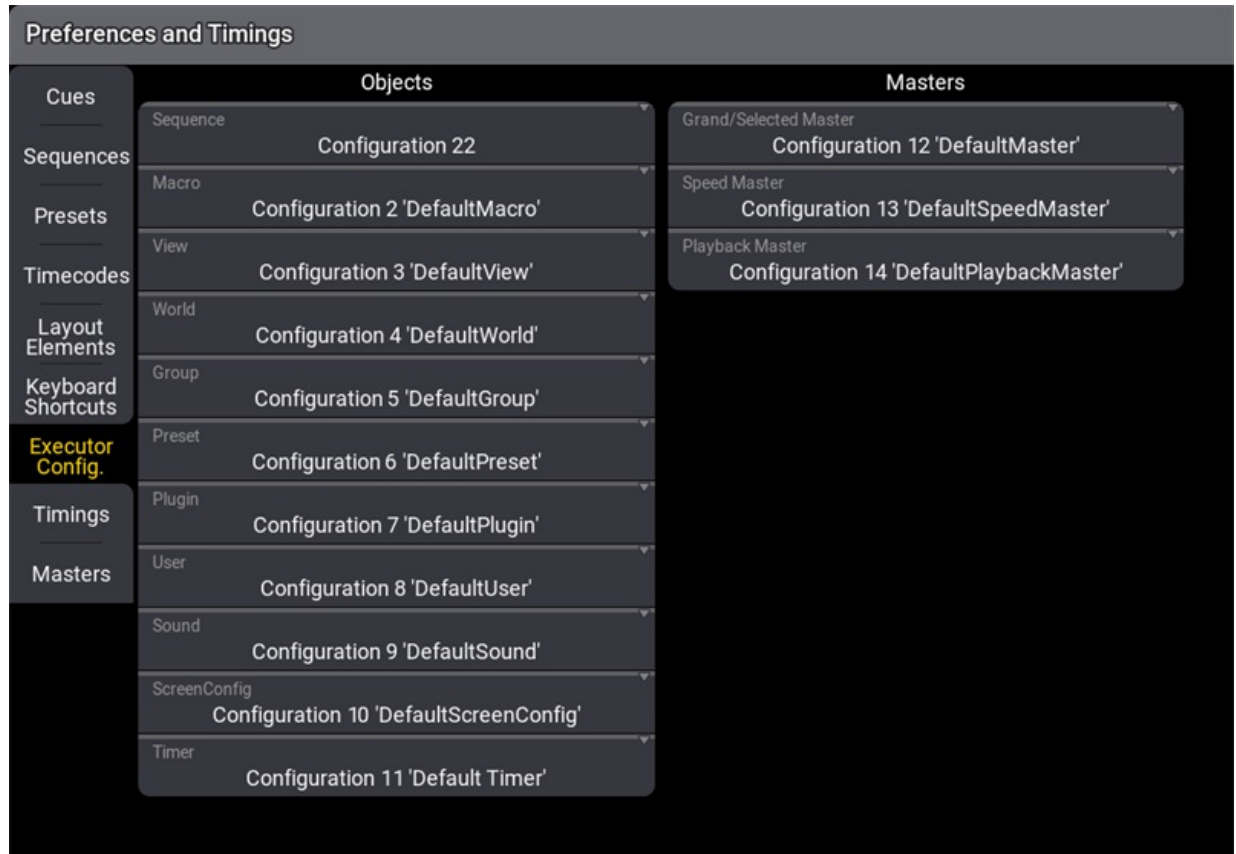
Tapping `List References` opens an info pop-up showing the objects that reference the configuration and what other objects this configuration might depend on.

`Recast Config` is used when a configuration has been saved with changes, and these changes should also be applied to the other executors that use this configuration. Saving a change to a configuration does not automatically apply the changes to other executors. Pressing `Recast Config` applies the changes to all the executors using the configuration. The recast can also be applied using the **Recast keyword**.

Executor Configuration Preferences

The default executor configuration for each object is stored in the Preference and Timing.

This is accessed by pressing **Menu** /  and then **Preference and Timings**, and finally **Executor Config**.



Preferences and Timings - Executor Config menu

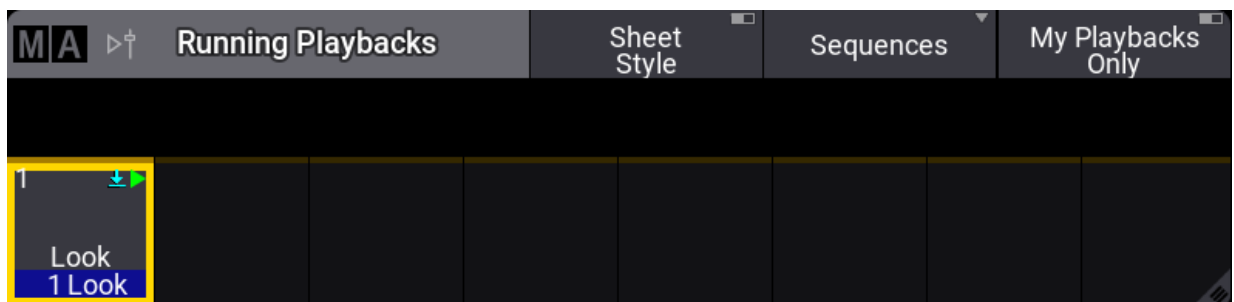
This menu has a setting for each object type that can be assigned to an executor. Tapping each of these settings opens a small select pop-up where any of the executor configurations can be selected as the default settings for the object type.

28.3. Running Playbacks

There are two ways to see the different running playbacks. This can be useful when trying to find out what is running and also to quickly turn off different playbacks.

Running Playback Window

It can be a window on a screen. This is called **Running Playbacks** and can be created like any other window. It is in the "More" tab.



Running playback window with a sequence running

Each running object is displayed as a pool object and can be interacted with like any other pool object. For instance, it is possible to use the Off keyword to turn off an object by pressing **Off** and then tapping the object in the window.

This window can show different types of objects: Sequences, Macros, Timecodes, Presets, and Timers.

Toggle between them by tapping **Sequence**, **Macros**, **Timecodes**, **Presets**, or **Timers** in the title bar. This is a swipe button, so it can also be pressed and swiped out to select one of the options.

It is possible to mask the objects in the window to only display the playbacks triggered by the currently logged-in user. This can be turned On or Off by tapping **My Playbacks Only** in the title bar.

The last button in the title bar is **Sheet Style**. This turns On or Off the sheet style for this window. The default style is pool style, like the image above. Sheet style could look like this:



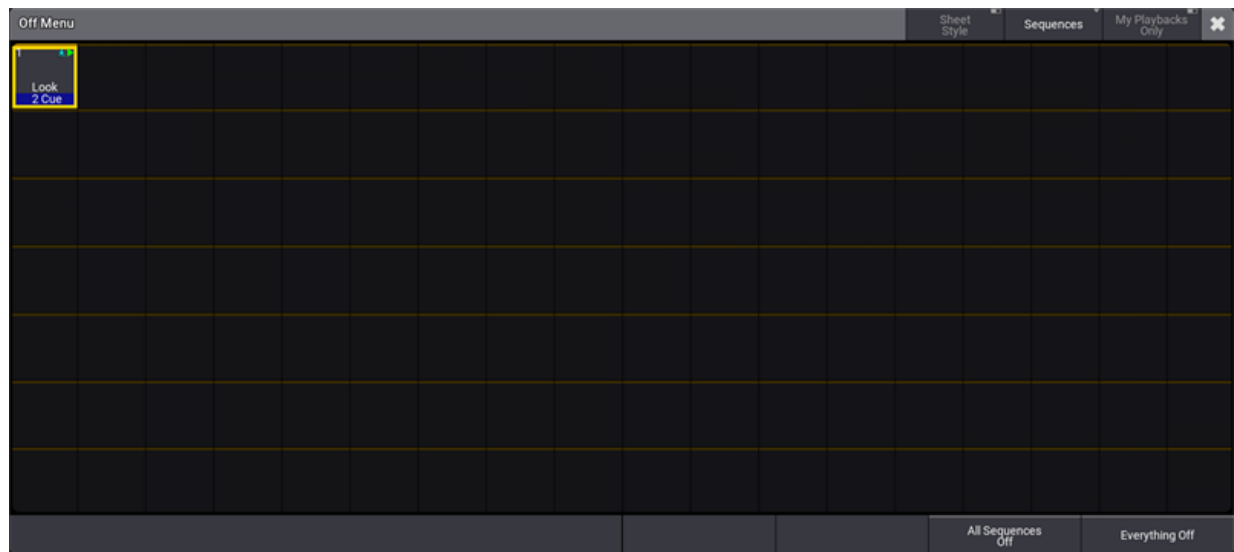
Running playbacks window in sheet style - sequence 1 running

This mode shows fewer details, not displaying the extra information available on pool objects.

Changing the font size is the only other option for the **Running Playbacks** window. These settings can also be found in the **Running Playback Settings**, opened by tapping the MA logo in the upper left corner of the title bar.

Off Menu

The other way to see running playbacks is to open a temporary version of the window. This is called the **Off Menu**. It is opened by pressing **Off** twice:



Off Menu with a sequence running cue 1


This is the same as the **Running Playbacks** window, with a few exceptions.

When the menu is in **Pool Style** there are two buttons at the bottom, and the pool style has a permanent **Off** action, meaning that tapping any object instantly turns the object off.

The button called **All Sequences Off** in the image above changes depending on the settings. This means it says **All Macros Off** if the pool displays the macros and similar for the other objects. It also changes from **All** to **My** if the pool only displays your sequences or macros. Tapping it does what the button says.

Tap **Everything Off** to immediately turn **all playbacks** off.


In **Sheet Style**, there is an extra button, and **Off** is not the default action. This means selecting multiple rows in the sheet is possible, and then tapping the extra **Off Selected** button turns off the selection.

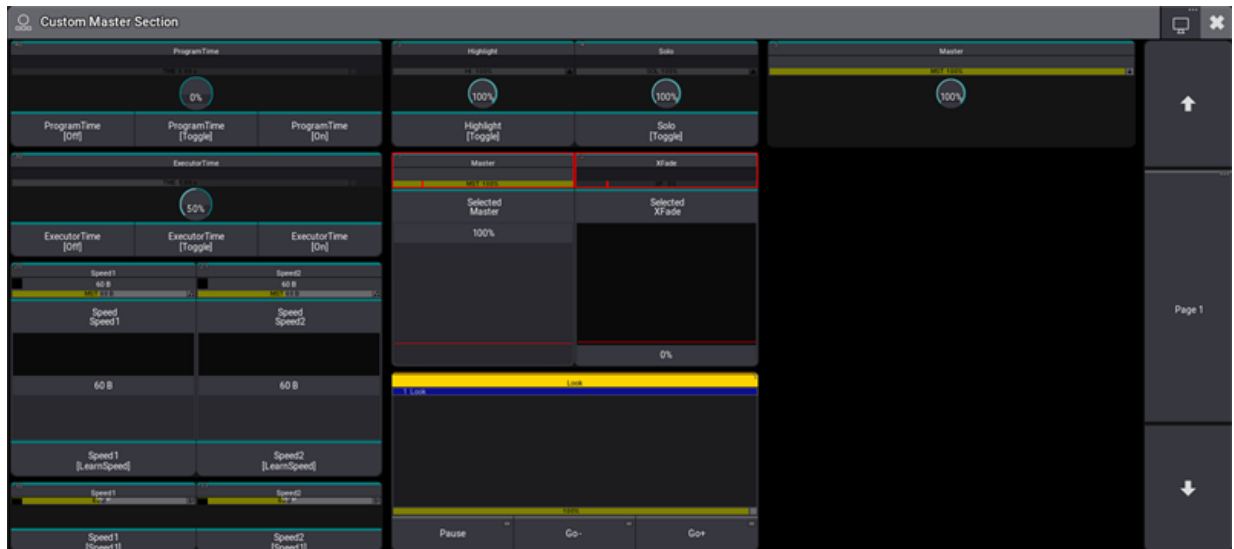
The menu can be closed like any other pop-up by tapping the  in the upper right corner.

28.4. Special Executors

Special executors include the grand master knob as well as the keys, knobs, faders, encoders, and wheels found in the custom area and master area. By default, they are assigned to control a useful assortment of masters. As with other executors, changing the assignment of special executors allows control of many types of objects including sequences, masters, plugins, groups, macros, worlds, and views.

The master area is always displayed to the right of the playback bar on screen 9. The master area is also displayed to the right of the optional playback bar on screen 2. The custom area is always displayed to the right of the playback bar on screens 10, 11, and 12. The custom area is also displayed to the right of the optional playback bar on screens 3, 4, and 5. For more information on screen allocation, see the [Screen Allocation](#) topic.

The **Custom/Master Section** window displays the current assignment and status of the special executors in the two custom areas, the master area, and the grand master knob. This window can also include on-screen copies of the special executor encoders, wheels, knobs, and keys; as well as the **Go+**, **Go-**, and **Pause** keys for the selected sequence. Tap the  icon in the control bar to open the **Custom/Master Section** overlay temporarily. The **Custom/Master Section** window is also available under the **More** and **All** tabs in the **Add Window** pop-up.



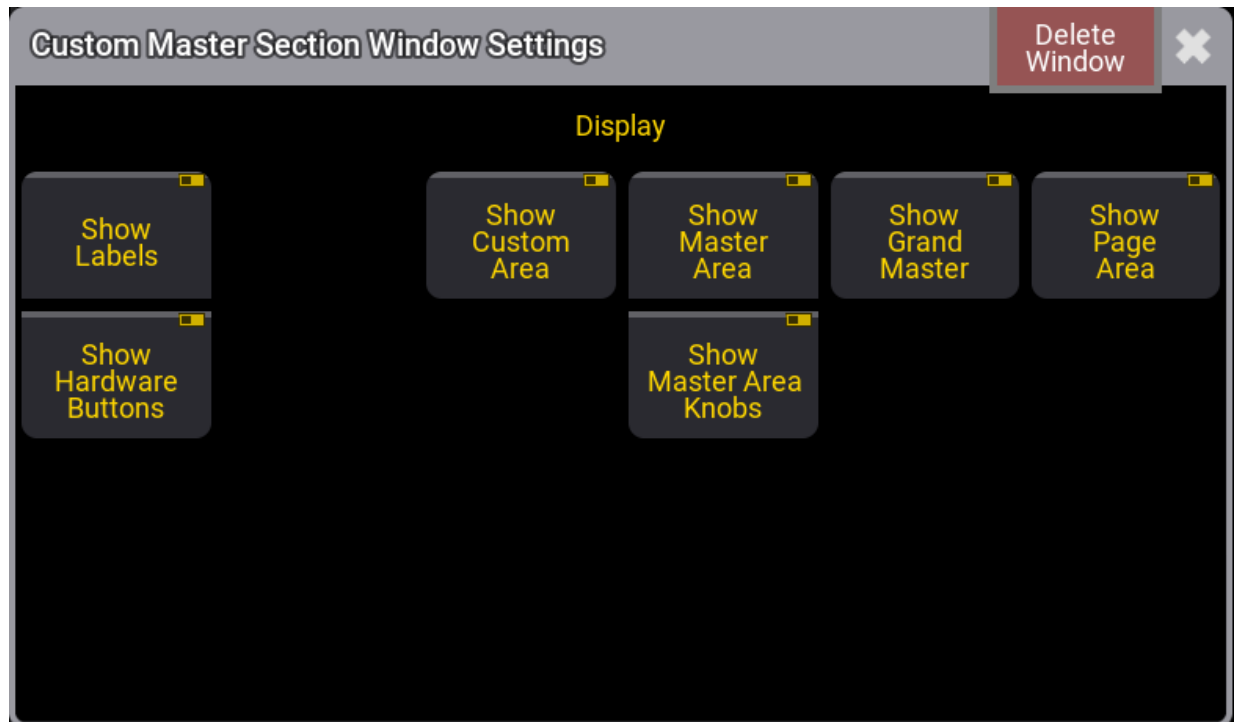
Custom/Master Section window

The Custom/Master Section window contains three main areas:

- Custom area encoders and wheels displayed on the left side of the window when all sections are visible.
- Master area, including the **Go+**, **Go-**, and **Pause** buttons for the selected executor, displayed center of the window when all areas are visible.
- Grand master, displayed on the right side of the window when all sections are visible.


These three main areas can include labels and a visualization of the associated hardware. When both labels and hardware are visible, the label for each special master appears directly above the associated hardware. In addition, page navigation buttons can appear along the right edge of the window.


Tap **MA** in the upper left corner of the window to open the Window Settings pop-up.



Custom/Master Section Window Settings pop-up

Tap **Show Labels** to show or hide the labels for all visible sections. Tap **Show Hardware Buttons** to show or hide the hardware visualizations for all visible sections. Tap **Show Custom Area**, **Show Master Area**, **Show Grand Master**, **Show Page Area**, and **Show Master Area Knobs** to show or hide the corresponding area of special executors or page navigation controls.

 **Hint:**
The **Show Hardware Buttons** and **Show Master Area** settings must all be enabled in order for the **Default Playback** buttons to be visible.

 **Hint:**
As elements are hidden, the remaining visible elements automatically rescale to fill the available space within the window.

Tap the label of the **Default Playback** area to open the **Edit Sequence** overlay for the selected sequence.

Tap the label of any special executor to open the **Assign Menu** for that special executor. The options available in the **Assign Menu** for a special executor are the same as the options for any other executor. For more information on the assignment process, see the [Assign Object to an Executor](#) topic.

Special Executor 5 and the Grand Master

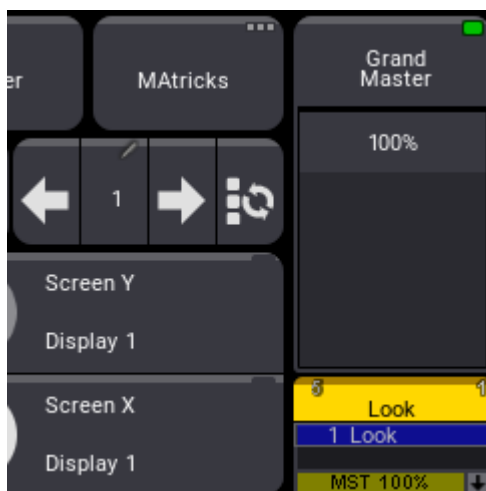
By default, the grand master is assigned to special executor 5. Press **Assign** then press the special executor 5 knob to open the **Assign Menu** for special executor 5. Use the Assign Menu to assign any desired object to special executor 5.



Hint:

Since the hardware associated with special executor 5 consists of only a knob with no additional key, it is possible to assign a normal key function to the press of the knob.

When any object other than the grand master is assigned to special executor 5, the area of the encoder bar that normally shows the grand master adjusts to show both the grand master and a label for the object assigned to special master 5.



Encoder bar showing both sequence 1, assigned to special master 5, and the grand master.



30. Masters

Masters are physical representations of various timing and level overrides that exist in the software.

Masters can be assigned to fader executors or any of the special executors in the [Master section](#), [Custom section](#), or the [Grand Master](#).

They can all be assigned using the Assign Menu. Tap **Object** in the top-left corner, then **Master** at the top of the menu to reveal a tree structure separated into the four groups of Masters.

They can also be assigned and adjusted using the [Master keyword](#).

For more information on assigning objects to executors, see the [Assign Object to an Executor topic](#).

Quick access to all of the selected masters and grand masters is available in the master controls menu. For more information on this menu, see the [Master Controls topic](#).

Masters have a limited selection of functions available as key assignments.

Please read about the available masters in the following subtopics.

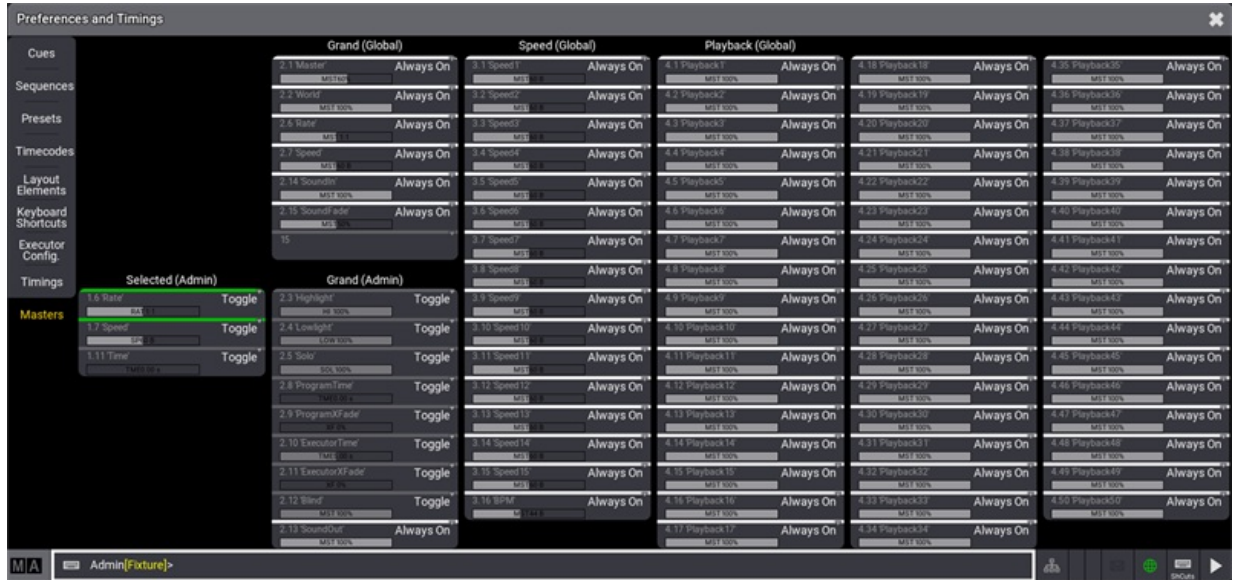
Master Modes

Masters have a mode that defines how they operate.

There are three available modes:

- **Toggle:**
Switch the master On or Off. The fader value of the master can always be changed and affects the output when the master is On. The indicator colors are gray (Off) and green (On).
- **Always On:**
The master is always On and cannot be switched Off. The fader value of the master can always be changed and affects the output. The indicator color for Always On is white.
- **Disabled:**
The master is switched Off and its fader value internally uses its default value. For example, the grand master uses 100% and the program time uses 0s. The indicator color for Disabled is red.

The modes of all masters can be changed in **Menu** - **Preferences and Timings** - **Masters**.



Masters settings in Preference and Timings menu

This menu displays the masters separated by their type and their membership in the show file.

Masters can have a global membership which means that these masters are controlled by all users of the show file. Masters with a user membership are located within the user profile which means that every user profile has its own individual set of these masters.

Changing one of these masters does not affect the same master in a different user profile.

The section of masters that are controlled by the user profile displays the name of the user behind their type, for example, Selected (Admin) for the masters that control the selected sequence of the Admin user.

Masters of the global membership display (Global) behind their type, for example Playback (Global).

The mode can be changed by tapping the current mode of the master. This toggles through the three modes.

This menu also shows the current value of each master.



30.1. Selected Masters

The selected masters replicate the 11 masters that each executor with a sequence can have assigned.

These masters can be assigned to executors and special executors. They will adjust the levels and timing of the selected sequence.

Read about the functions of the masters in the [Assign Object to an Executor topic](#).

The 11 masters are:

- **Master 1.1 'Master'**
- **Master 1.2 'XFade'** - this is also called Crossfade
- **Master 1.3 'XFadeA'** - this is also called CrossfadeA
- **Master 1.4 'XFadeB'** - this is also called CrossfadeB
- **Master 1.5 'Temp'**
- **Master 1.6 'Rate'**
- **Master 1.7 'Speed'**
- **Master 1.8 'Highlight'**
- **Master 1.9 'Lowlight'**
- **Master 1.10 'Solo'**
- **Master 1.11 'Time'**



30.2. Grand Masters

The Grand Masters are a selection of 14 different masters that can be assigned to executors and special executors.

They are:

- **Master 2.1 'Master'**
- **Master 2.2 'World'**
- **Master 2.3 'Highlight'**
- **Master 2.4 'Lowlight'**
- **Master 2.5 'Solo'**
- **Master 2.6 'Rate'**
- **Master 2.8 'ProgramTime'**
- **Master 2.9 'ProgramXFade'**
- **Master 2.10 'ExecutorTime'**
- **Master 2.11 'ExecutorXFade'**
- **Master 2.12 'Blind'**
- **Master 2.13 'SoundOut'**
- **Master 2.14 'SoundIn'**
- **Master 2.15 'SoundFade'**

Master is the grand master. This has a dedicated virtual fader, but can also be assigned to any executor or special executor.

The **World** master functions like a grand master but only for fixtures included in the selected world. For more information on worlds, see the [Worlds and Filters topic](#).

The **Highlight** master inhibits the effect of the highlight function on selected fixtures. It offers proportional control over dimmer as well as all other highlighted attributes. It also works as an inhibitor for the lowlight function. For more information on highlight, see the [Highlight keyword topic](#).

The **Lowlight** master inhibits the effect of the lowlight function on highlighted fixtures. It offers proportional control over all lowlighted attributes. For more information on lowlight, see the [Lowlight keyword topic](#).

The **Solo** master inhibits the effect of the solo function on selected fixtures. It offers proportional control over dimmer attributes. For more information on solo, see the [Solo keyword topic](#).

The **ProgramTime** master modifies the fade time used by the programmer. For more information on program time, see the [Time Control topic](#).

The **ProgramXFade** master crossfades new programmer values to the output. It affects all programmer values changed after ProgramXFade is activated.



The **ExecutorTime** master overrides stored cue timing. For more information on exec time, see the [Time Control topic](#).

The **ExecutorXFade** master crossfades one or multiple executors. It is like a crossfade for a single executor but affects all executors triggered after the ExecutorXFade is activated.

The **Blind** master inhibits the effect of the blind function. It offers proportional control over dimmer as well as all other attributes. For more information on blind, see the [Blind keyword topic](#).

The **SoundOut** is the volume of the console's audio output.

The **SoundIn** master inhibits incoming audio signals. For more information about using sound input, see the [Sound Window topic](#).

The **SoundFade** master adjusts how quickly the console responds to changes in incoming audio signals. For more information about using sound input, see the [Sound Window topic](#).




30.2.1. Time Control

When enabled, the Program Time and Executor Time masters can override playback timing within the programmer and executors, respectively, with a time range of 0 seconds to 10 seconds. They can be assigned to executors as well as special executors in the **Master area**, **Custom area**, or the **Grand Master**. This gives physical control of their activation and level. Quick access to these masters is available in the master controls menu. For more information on this menu, see the **Master Controls topic**.

Program Time

Program Time adds a fade time to changes of attribute values in the programmer. Set the master to the desired value and activate Program Time by pressing **On** or **Toggle** on an executor assigned to the program time master, or tap **Prog Time** in the master controls menu so there is a green marker. The set time will be used for all value changes in the programmer except for those made by turning an encoder or the dimmer wheel.

To deactivate program time, press **Off** or **Toggle** on an executor assigned to the program time master, or tap **Prog Time** again in the master controls menu. The green marker will disappear.



Hint:
Recalling presets that include individual fade time and delay time will continue to use their stored times.
Presets with no individual fade time will respect the program time setting.

Executor Time

Executor Time overrides cue fade times during sequence playback, regardless of whether the sequence is running from an executor or directly from the sequence pool. Set the master to the desired value and activate Executor Time by pressing **On** or **Toggle** on an executor assigned to the executor time master, or tap **Executor Time** in the master controls menu so there is a green marker. Cues stored with only basic timing will change values using the specified executor time. Individual fade and delay and times will continue to playback as stored. Cues with follow or time triggers will continue to use the stored triggers.

To deactivate executor time, press **Off** or **Toggle** on an executor assigned to the executor time master, or tap **Executor Time** again in the master controls menu. The green marker will disappear.

If the **Use Executor Time** option in the sequence settings menu is enabled, the sequence will follow the executor time master. Disabling **Use Executor Time** allows all of the cues in the sequence playback using their stored timing. For more information on the sequence settings menu, see the **Sequence Settings topic**.



30.3. Speed Masters

15 different Speed Masters can be assigned to executors and special executors.

This group of masters also includes the BPM Master. This master adjusts the expected speed of the incoming audio. The console will adjust to the actual detected Beats Per Minute closest to this setting. For more information about using sound input, see the [Sound Window](#) topic.

Speed masters can be assigned to sequences. This makes it possible to sync multiple sequences to the same speed.

Read more about assigning speed masters to executors in the [Assign Object to an Executor](#) topic. Read about assigning a Speed Master to a sequence in the [Sequence Settings](#) topic.

The colored indicator bar above the speed master label blinks in accordance with the speed. This gives visual feedback about the current speed.



30.4. Playback Masters

50 different Playback Masters can be assigned to executors and special executors.

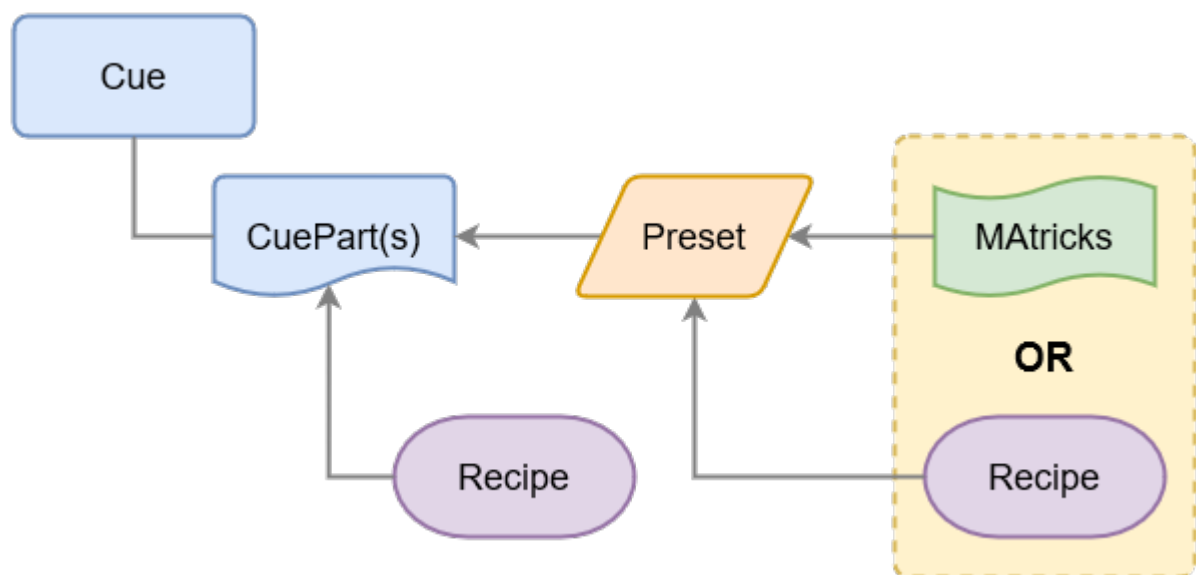
Playback Masters can be assigned as playback settings of sequences. The masters function as an inhibitive dimmer control for the assigned sequences. All sequences with the same playback master assigned can be simultaneously inhibited by that master.

Read more about assigning playback masters to executors in the [Assign Object to an Executor](#) topic. Read about assigning a playback master to a sequence in the [Sequence Settings](#) topic.

31. Recipes

Recipes can be a very useful tool for touring shows or when the changes are expected.

Recipes can be stored in cue parts and presets.



Recipe data flowchart

There is a possibility to create some conflicting data using recipes and stored values in both cue parts and in presets.

Values can be stored in a cue part. Cue parts can also contain recipe data.

The cue part recipe needs to be cooked into the cue part. If this is cooked with the "Merge" option then existing values in the cue part are not overwritten by the recipe. If the "Overwrite" option is selected, then all existing values are overwritten by the recipe - values that are not affected by the recipe are removed from the overwritten cue.

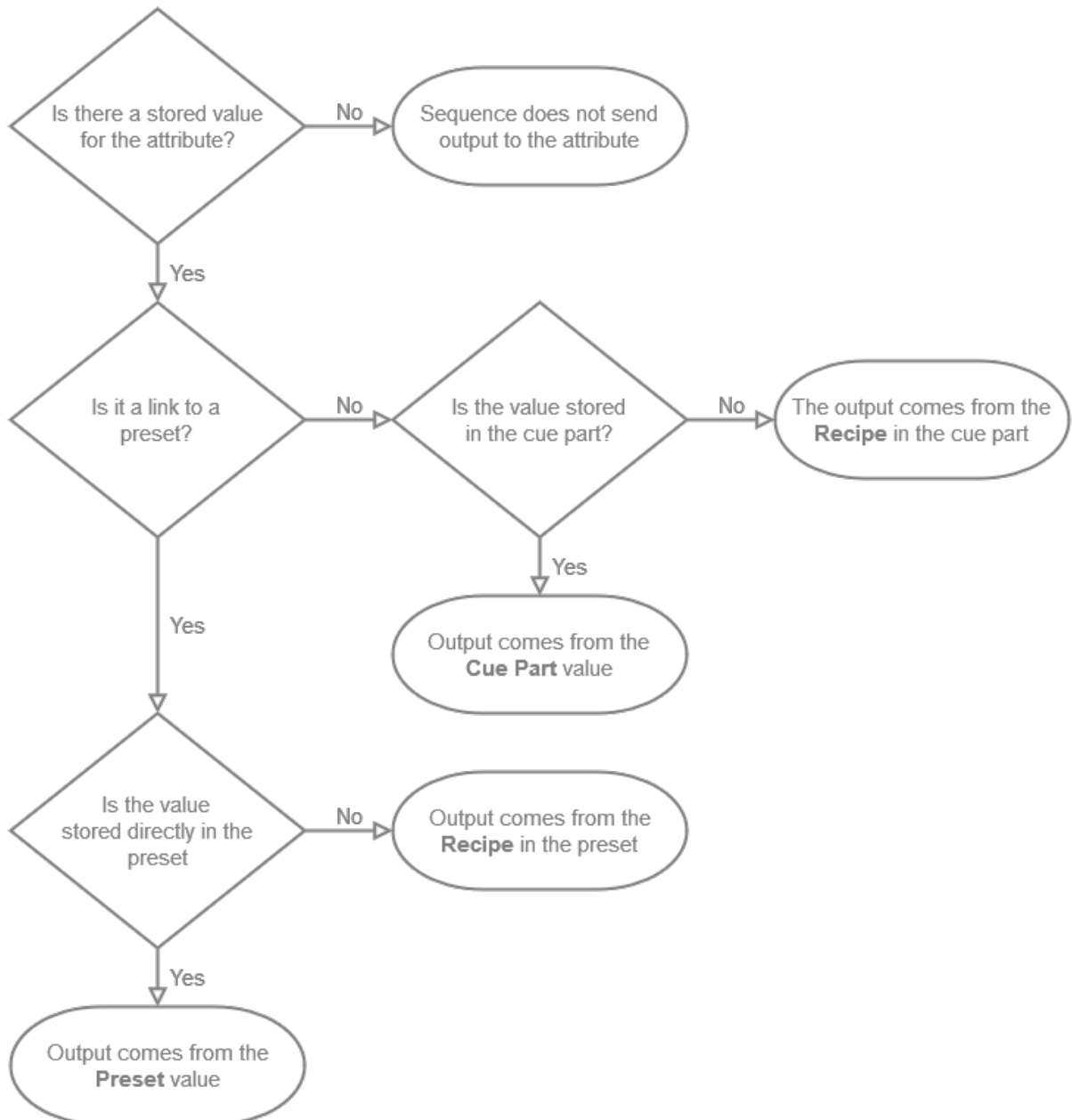
Preset recipes are automatically cooked into the preset and a preset can be used in a cue part.

All this means that an attribute can potentially get a value from four different locations, but only one of these values can be output:


- The cue part
- The cue part recipe
- The preset
- The preset recipe

Values stored in the cue part have higher priority than the cue part recipe value. Values stored in the preset have a higher priority than the preset recipe values. Storing a preset to a cue part creates the preset reference to the preset so in that case the output is decided by the preset priority.

This flow chart can be used to determine where the output comes from in a sequence:



Output decision flowchart

A recipe contains one or multiple recipe lines describing what should happen based on a set of information. The recipe can "cook" values into the cue part, preset, or programmer. This cooked data is marked by a small pot icon  and the cooked data can easily be removed again if needed.



A recipe line can contain information about a selection of fixtures, group, preset, MATricks, individual fade, delay, speed, and phase values.

Values from recipes can be combined with conventionally stored values.

The flexibility in the recipe system allows for a variety of uses. The recipes could be used to:

- Create recipe presets for groups referencing other presets for a flexible fixture setup.
- Create "template" presets with ranged values that can be applied to a flexible selection of fixtures.
- Cues that contain a recipe on how different elements create the desired look.

This is a recipe line:

References		Grid			Layers				Shuffle		X		Y			Z			
Name	Stored Data	Selection	Values	MATricks	Fade From X	Fade To X	Delay From X	Delay To X	Speed From X	Speed To X	Phase From X	Phase To X	XGroup	XBlock	XWings	XWidth	InvertStyle	PhaserTransform	InvertX
Recipe 1					None	None	None	None	None	None	None	None	None	None	None	Auto	Pan	None	No
New Recipe																			

Empty recipe line

A top bar with toggle buttons can show or hide different elements of the recipe.

- **References:**

These are the columns for referenced data.

- o **Selection:**

This is the selection of fixtures using this recipe line. If the selection is a group, then the Group ID is displayed. If the selection is from the programmer, then it says **<Recipe>**.

- o **Values:**

This is the value reference used in the recipe line. If the value is a preset, then the preset ID is shown. If the value is from programmer values, then it says **<Recipe>**.

- o **MATricks:**

This is a reference to an existing MATricks pool object. Having a reference to an existing MATricks adds referenced values in the Grid columns.



- **Grid:**

These columns are the same as known from **MAtricks**. There is a set of columns for X, Y, and Z axes in the grid. The columns can be filtered by activating one or several of the **X**, **Y**, and **Z** in the top bar.

 - **Group:**

The number of groups the selection is split into.
 - **Block:**

The number of fixtures blocked together.
 - **Wings:**

The number of wings the selection is split into.
 - **Width:**

This changes the width of the selection in the **Selection Grid**.

This section also has columns for invert. Some of these are linked to X, Y, and Z. Some are not linked to an axis but a common setting:

 - **InvertStyle:**

This defines if Invert is applied to Pan, Tilt, Pan and Tilt, or All attributes.
 - **PhaserTransform:**

Transform can be set to Mirror. This mirrors values depending on the other grid settings, for example, Blocks, Groups, Wings. Learn more in the **Transform topic**.
 - **Invert, Inv, InvB, InvG, InvW:**

These are Yes/No inverting settings for each of the grid axes.
- **Layers:**

These are the timing layers. X, Y, or Z also needs to be active for any of the layer columns to be shown.

 - **Fade From / Fade To:**

The two fade values allow spreading the fade time over a range.
 - **Delay From / Delay To:**

The two delay values allow spreading the delay time over a range.
 - **Speed From / Speed To:**

The two speed values allow spreading the speed values over a range.
 - **Phase From / Phase To:**

The two phase values allow spreading the phase values over a range.
- **Shuffle:**

These are the shuffle columns. X, Y, or Z also needs to be active for any of the layer columns to be shown.

 - **Shuffle:**

This value can be set to shuffle the selection order.
 - **Shift:**

This value can be set to shift the selection in the selection grid.
- **X, Y, and Z:**

Each of the grid axes has columns. If the recipe only uses one or two axes, then the columns can be limited to only show the relevant columns.



Besides the columns mentioned above, the **Name** column can also be used to give the recipe line a name. Finally, the **Stored Data** column shows the type of values (Universal, Global, or Selective) stored if it is from the programmer.



Hint:

The MAtricks and Shuffle settings are described in detail in the **MAtricks and Shuffle section**.

Recipes do not need information in all columns, but they can have information in all. They often only have information in a few columns.

Read the **Cue Recipe** and **Preset Recipe** topics for details on how to create and use each of the two.

32. Phasers

Phasers are the effects and chasers of grandMA3, creating dynamic output from a single preset or cue.

Phasers change the output for attributes using a set of information in two or more steps. A normal, static cue or preset only contains one step of information. Adding additional steps creates a phaser.

Example

A phaser has two steps with an absolute dimmer value of 100% in step 1 and an absolute dimmer value of 0% in step 2. With all other phaser layers at their default values, the resulting output is a smooth, repeating fade of the dimmer output from 100% to 0% and back to 100%.

Layers

Value Layers

Each phaser step can include attribute value information. These values can be an absolute value (for instance, a dimmer value of 50%) or a relative value (for instance, a dimmer value of -20%). Steps can contain both absolute and relative values simultaneously.



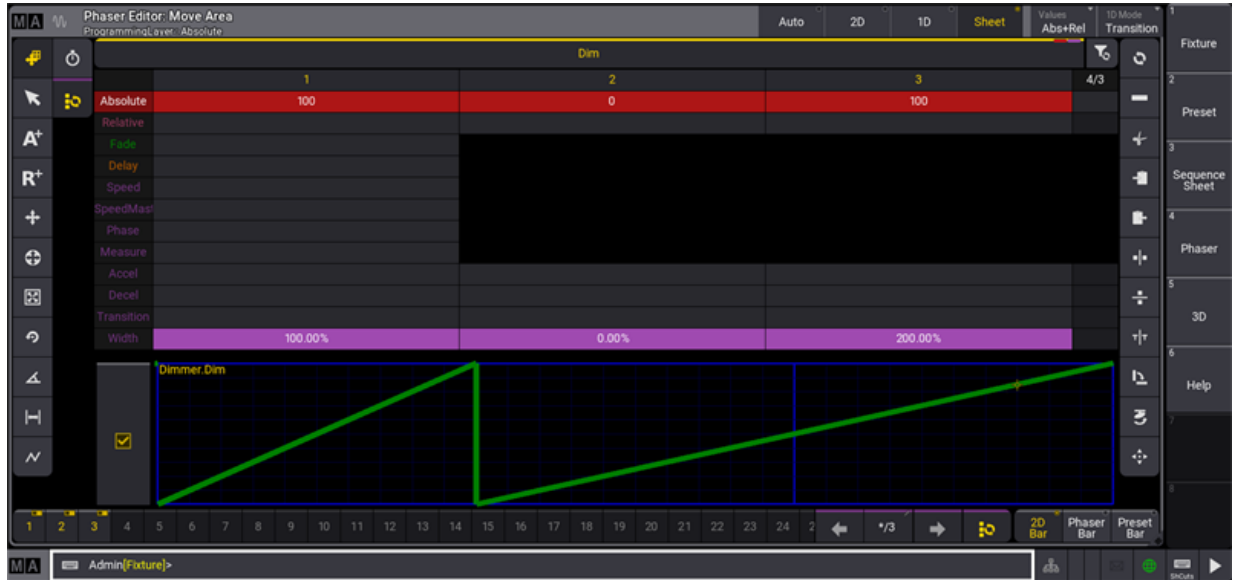
Important:

The grandMA3 software handles storage, editing, and playback of absolute values and relative values separately. This separation allows the user to combine different absolute and relative values as desired.

Step Layers

Each phaser step also includes additional layers, which define the overall width of the step, the percentage of the step dedicated to transitioning to the new value, and the amount of acceleration and deceleration used when changing to the new value.

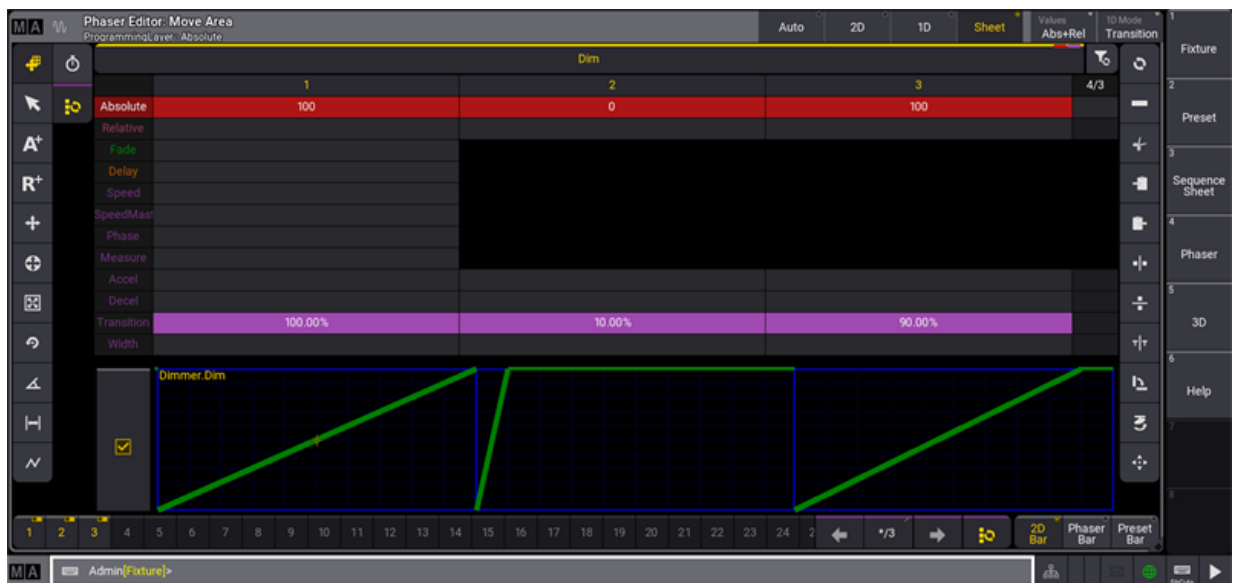
- **Width**



Phaser Editor showing steps with various widths.

The Width layer defines the total amount of time from the beginning of a step to the beginning of the next step. Width is shown as a percentage of one beat as defined by the Speed layer. The image above shows the first step with 100% width, so the change in value takes place over the time of one beat. The second step has a width of 0%, so the values change instantaneously and the next step begins immediately. The third step has a width of 200%, so the values change over the time of two beats.

- **Transition**

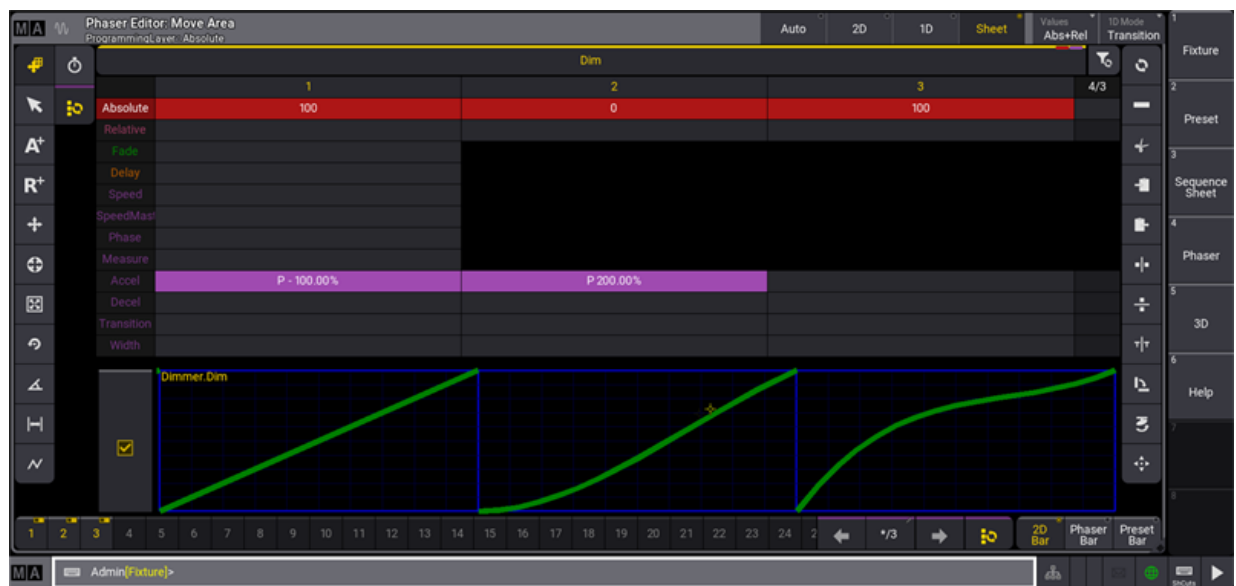


Phaser Editor showing steps with various transitions.

The Transition layer defines how much of the step width is used to adjust values to their new levels. The

transition of a step is shown as a percentage of the width of that step. The image above shows the first step with a transition of 100%, so the values change during the entire time of the step. The second step has a transition of 10%, so the values change during the first 10% of the width of the step, then remain at the new level until the end of the step. The third step has a transition of 90%, so the values change during the first 90% of the width of the step, then remain at the new level until the end of the step.

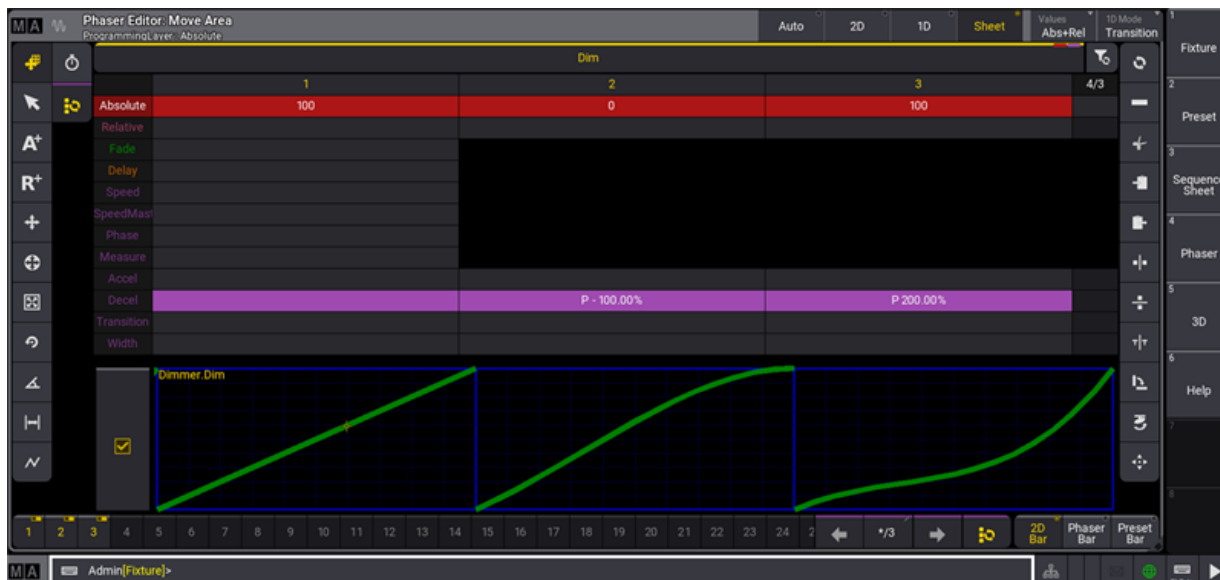
- **Accel**



Phaser Editor showing steps with various acceleration values.

The Accel layer defines how abruptly or gently the attributes begin to change from the value defined by the current step toward the value of the next step (acceleration). In the image above, the first step shows an acceleration of -100%, resulting in a smooth start to the transition away from the value at the end of the step. The second step shows an acceleration of 200%, resulting in an abrupt start to the transition away from the value at the end of the step.

- **Decel**



Phaser Editor showing steps with various deceleration values.

The Decel layer defines how abruptly or gently the attributes reach the value defined by the current step (deceleration). In the image above, the second step shows a deceleration of -100%, resulting in a smooth end to the transition into the value for the step. The third step shows a deceleration of 200%, resulting in an abrupt end to the transition into the value for the step.



Hint:

For linear transitions, keep Accel and Decel values at 0%.

For smooth transitions, closely resembling sinus or cosinus waves, Accel and Decel values of -100%.

Phaser Layers

Each phaser additionally includes layers, which affect all steps of the phaser. Each attribute of each fixture can have its own speed, speed master, phase, and measure value in a phaser, but these values will be the same for all steps in a phaser.

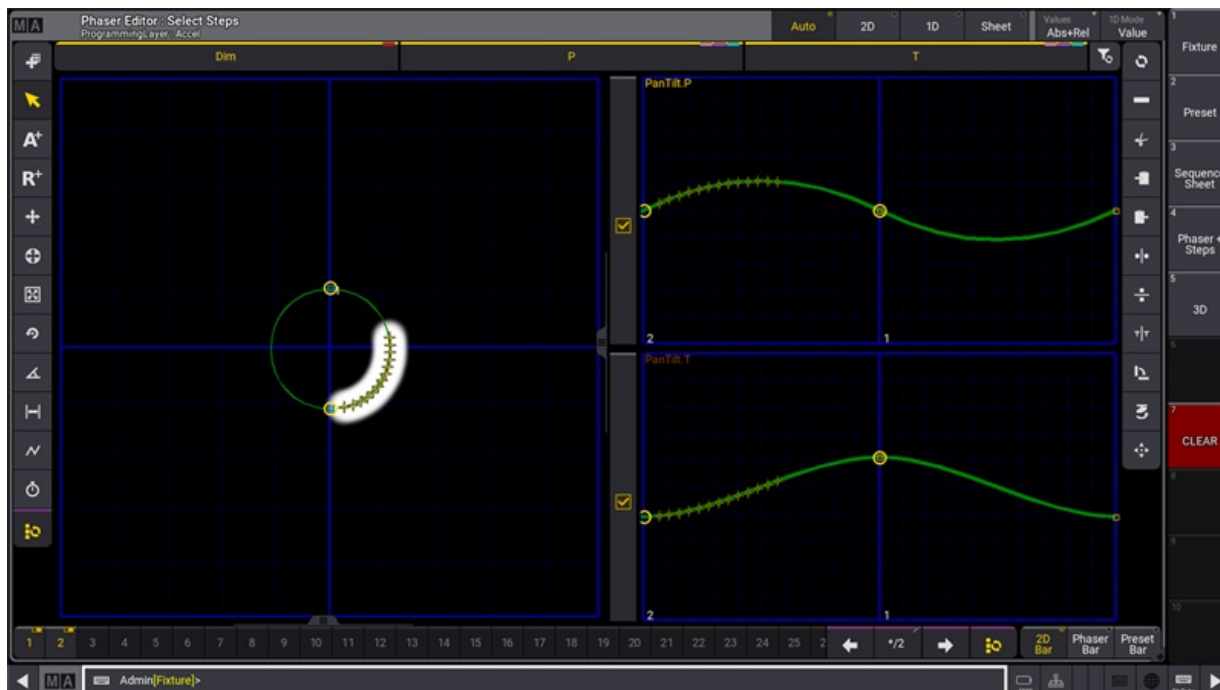
- **Speed**

Speed defines the overall rate at which the console plays back the steps of a phaser. The grandMA3 displays speed in units of either BPM (beats per minute), Hz (beats per second), or Seconds (seconds per beat).

- **SpeedMaster**

SpeedMaster defines per attribute to which master of the type speed the phaser adapts its speed to. Phasers speeds can be assigned to executors by using speed masters. For more information, see **SpeedMaster keyword**.

- **Phase**



Phaser editor showing a range of phase offsets.

Phase defines the timing offset of each fixture's attributes within a phaser. Phase is displayed in degrees (°). The image above shows a selection of fixtures in a circle phaser with phase offsets ranging from 0 to 90 degrees. The Phaser Editor displays this offset in both the 2D and 1D layouts.

● Measure

The optional Measure layer defines the number of beats in the repeating phaser loop. For example, a phaser with a measure of 4 beats and a speed of 120BPM will repeat every 2 seconds.



Hint:

When using the Measure layer, multiple combinations of step widths can produce the exact same result. In this case, it may be easiest to think of the width as a percentage of one beat in the measure.

All phaser layers, as well as a step selection tool, appear in the **Encoder Toolbar** any time attributes are mapped to the encoders. Phaser layers are indicated by a dark purple color. Phaser layers behave just like other attribute layers when activating, deactivating, storing, and clearing. For more information about attribute encoders and layers, see the **Operate Fixtures** topic. The Phaser Editor provides a graphical view and powerful manipulation of phaser layer information.

Steps

The grandMA3 offers multiple tools and methods for creating, selecting, and deleting phaser steps.



One intuitive method to quickly create steps based on presets involves the **Step** key. To create steps using this method, press and hold the **Step** key, then tap a preset for each desired step.

Example

To create a 3-step color phaser from white to red to blue:

1. Select the desired fixtures.
2. Press and hold the **Step** key.
3. Tap the white color preset.
4. Tap the red color preset.
5. Tap the blue color preset.
6. Release the **Step** key.

When using this method, the software will only create the next step when tapping a new preset, which contains any of the same attributes as the current step. This behavior allows each step to reference multiple presets as long as those presets apply to different attributes.

Example

To create a 2-step phaser, where each step contains both color and position presets:

1. Select the desired fixtures.
2. Press and hold the **Step** key.
3. Tap the desired color preset for step 1.
4. Tap the desired position preset for step 1. This preset is added to step 1.
5. Tap the desired color preset for step 2. Step 2 is automatically created.
6. Tap the desired position preset for step 2. This preset is added to step 2.
7. Release the **Step** key.

The **Step** key can also quickly change which preset is referenced by a step.

Example

Start with the 3-step color example above, but afterward, change the blue step to yellow:

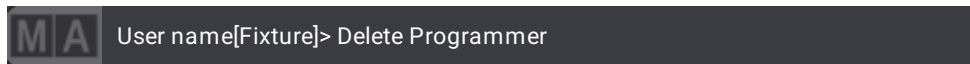
1. Follow all directions in the 3-step color example above.
2. Press and release **Step**.
3. Press and release **3**.
4. Tap the yellow color preset.

Delete specific steps from the programmer using standard syntax with the Delete keyword. For example, to delete step 3, type:

```
MA User name[Fixture]> Delete Step 3
```



Alternatively, delete specific steps from the programmer by selecting the steps in the Step Bar and deleting the **programmer**.



An easy way to access this command (especially on grandMA3 onPC) is to tap **Delete Steps** in the **At Overlay**.

Stomp

The stomp function stops a phaser, running either in the programmer or from a playback, and resolves it to a single, static step. The console calls and activates the last static output value for each stomped attribute into step 1 of the programmer. All other steps of the stomped phaser are discarded.

Sync

The Sync option ensures that phasers recall with predictable timing and phase offsets.

When calling phasers into the programmer, the Sync option is available in the **Encoder Bar**.

When recalling phasers stored in cues, the Sync option is available as a setting for each cue part in the Sequence Sheet. For more information about settings in cues and cue parts, see the **Look at Cues and Sequences** topic.

DelayToPhase

The DelayToPhase setting links the phase calculation of a phaser to any active individual delay time for the attribute.

When enabled, the phase calculation for each attribute begins as the values start to change at the end of the delay time. Because this delay timing may be different for different attributes or fixtures within a single cue, the resulting output may appear to include different phase offsets than originally stored within the phaser.

When disabled, the phase calculation for all attributes starting a phaser in the same cue starts when the cue is triggered, regardless of any individual delay times. This allows for more predictable, synchronized phaser playback, even with different individual delay times.

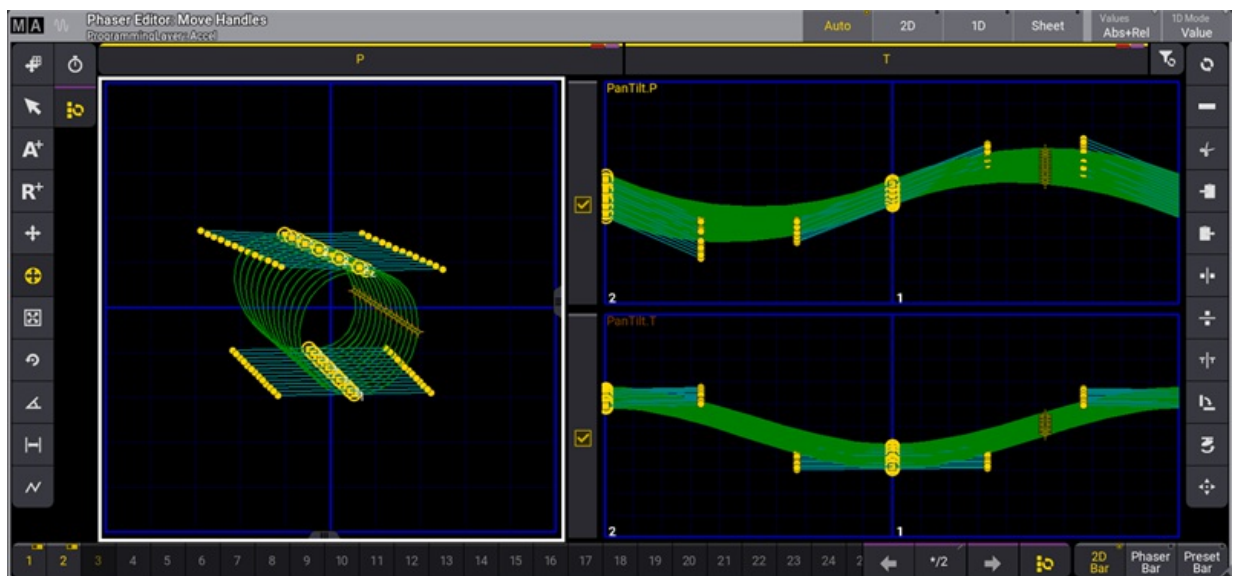
The DelayToPhase setting applies to all phasers in the show and appears in the Timing tab of the Preferences and Timing menu.

32.1. Phaser Editor

The **Phaser Editor** is a diverse tool for manipulating phasers.

It offers multiple means of visualizing running phasers as well as tools for dynamically creating and editing phasers. It can also be used as a simple trackpad to adjust the position attributes of fixtures.

Open a storable **Phaser Editor** window using the **Add Window pop-up**. Alternatively, open a temporary version of the **Phaser Editor** by tapping the **Phaser** button in the **Encoder Bar**.



Create and manipulate phaser steps in the phaser editor

The image above shows the Phaser Editor in the Auto view mode. Choose between multiple available view modes using the radio buttons in the title bar or the drop-down button in the Window Settings pop-up. Read below for images and descriptions of the other available view modes.

The blue grid on the left represents the entire pan range (horizontally) and the entire tilt range (vertically). This is called the 2D layout.

The smaller blue grids on the right are 1D layouts. Each attribute with more than one step in the phaser is displayed in its own 1D layout. The bold, vertical, blue lines in the 1D layouts represent beats, based on the speed of the phaser. The example above shows pan and tilt attributes, each with two steps.

When fixtures are selected, they are visualized by small yellow cross-hairs. The beams can be represented in the 2D layout by toggling the **Show Beams** button in the Window Settings pop-up. When **Show Beams** is enabled, the intensity and color of each fixture are visualized as a larger circle behind the corresponding yellow cross-hair.

Adding points in the 2D layout adds steps to the programmer. The first point moves the fixtures to the specified position. Each subsequent point adds another step to the programmer.

When a step contains both an Absolute point (shown as a yellow or cyan-filled square) and a Relative point (shown as a hollow, yellow or cyan square), a thin red line connects the relative point and the related absolute point. The cyan-colored points show that the value is from a preset. Yellow-colored points are hard values from the programmer.

Selected points have a yellow circle around them.

A green line describes the path of the fixtures.

A row of buttons immediately below the title bar includes one button for each attribute included in the active phaser. These buttons function as a direct link to the **At Filter**. Phaser adjustments apply to any attributes with a yellow bar and yellow text. Attributes with a grey bar and grey text will ignore adjustments. Tap the button immediately to the right of the last attribute button in this bar to enable or disable all attributes in the bar.



Hint:

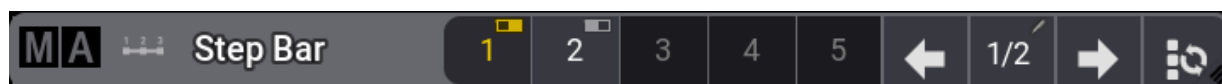
If any or all attributes in the bar are disabled, the first tap of the button to the right of the attributes enables all displayed attributes.

If all displayed attributes are already enabled, tapping this button disables them all.

The tool buttons on the left side are mouse or touch tools, and the tool buttons on the right are operational functions and shortcuts - read more below. The right side tool buttons might change depending on the selected tool on the left side.

Step Bar

The Step Bar appears across the bottom of the Phaser Editor. The Step Bar is a quick step selection tool. This element is optional in the Phaser Editor window, and it is enabled by default. Tap the **Step Bar** button in the Window Settings pop-up to show or hide the Step Bar. A storable window version of the Step Bar is also available in the **More** tab in the **Add Windows pop-up**.



Step Bar window with step one selected

Each step has a small square, which displays a status of empty, deselected, or selected.


Empty steps are steps that do not have any values. They have a dark background color.

Deselected steps have values in them, but they are not selected and are not affected by step-specific adjustments. Deselected steps have a light gray background color and an On/Off toggle icon in the upper-right corner.


Selected steps have values in them and are affected by step-specific adjustments. Selected steps have yellow text and a yellow On/Off toggle icon in the upper-right corner.

Tapping any of the steps toggles the selected status.

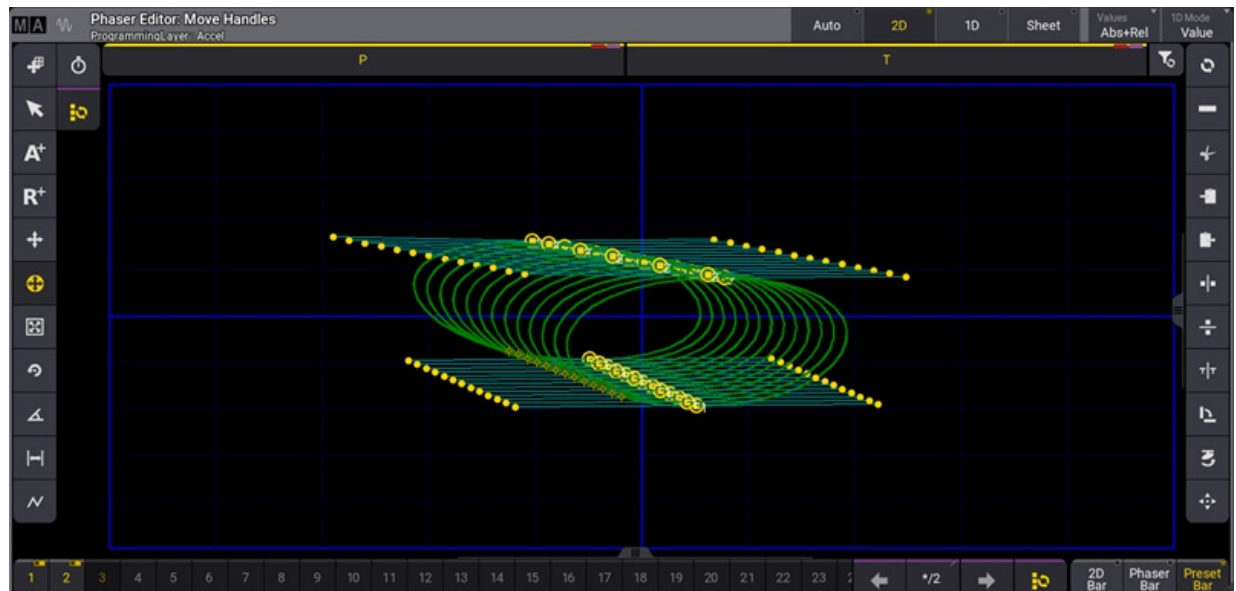
Additional controls appear on the right side of the Step Bar. These controls include left and right arrows used to change between single steps. An informative field appears between the arrows, showing which step is currently selected and the number of steps that contain values in the active phaser. If multiple steps are selected, the first selected step is shown with two dots after it (for example, 1.../3). If all steps are selected, the first number is replaced with an asterisk (for example, */3).

Tap the  button to the right of the right arrow in the Step Bar to select all steps. Tapping this selects all not-empty steps if only one step is currently selected. If more than one step is already selected, then only step 1 is selected. Tapping the button executes this command: **Step Toggle Executor**.

The Step Bar in the Phaser Editor window includes three radio buttons. Tap one of these buttons to call up the desired encoder toolbar. The Preset Bar opens the standard attribute **Encoder Toolbar** with access to all layers and all attributes. See below for Images and descriptions of the 2D Bar and Phaser Bar.

 **Restriction:**
The storable window version of the Step Bar does not include the encoder toolbar radio buttons.

2D View Mode



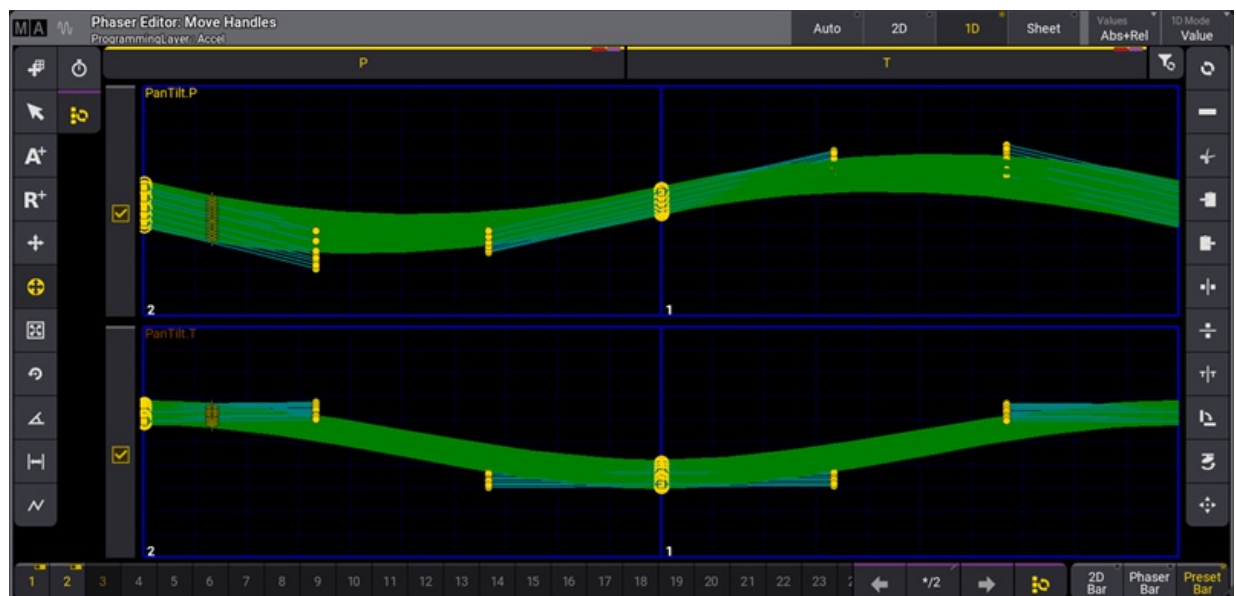
Phaser Editor in 2D View Mode

The 2D view mode removes other layout elements and expands the 2D layout to cover the main area of the window. This view mode is helpful when working with position phasers.

The Values button in the title bar allows or prevents the display and editing of absolute points or relative points. Tap to cycle through the available options or tap and swipe to open a list of available options. Options include:

- **None** - Displays and allows editing only of handles.
- **Absolute** - Displays and allows editing of absolute points and their handles.
- **Relative** - Displays and allows editing of relative points and their handles.
- **Abs+Rel** - Displays and allows editing of absolute and relative points, as well as their handles.

1D View Mode



Phaser Editor in 1D View Mode showing Values

The 1D view mode removes other layout elements and expands the 1D layouts to cover the main area of the window. This view mode displays a layout for every attribute included in the active phaser.

To change the height of the attribute lines, tap the **MA** logo in the upper-left corner of the window, then tap **LineHeight** in the **Window Settings** menu, enter the desired height, and press **Please**.

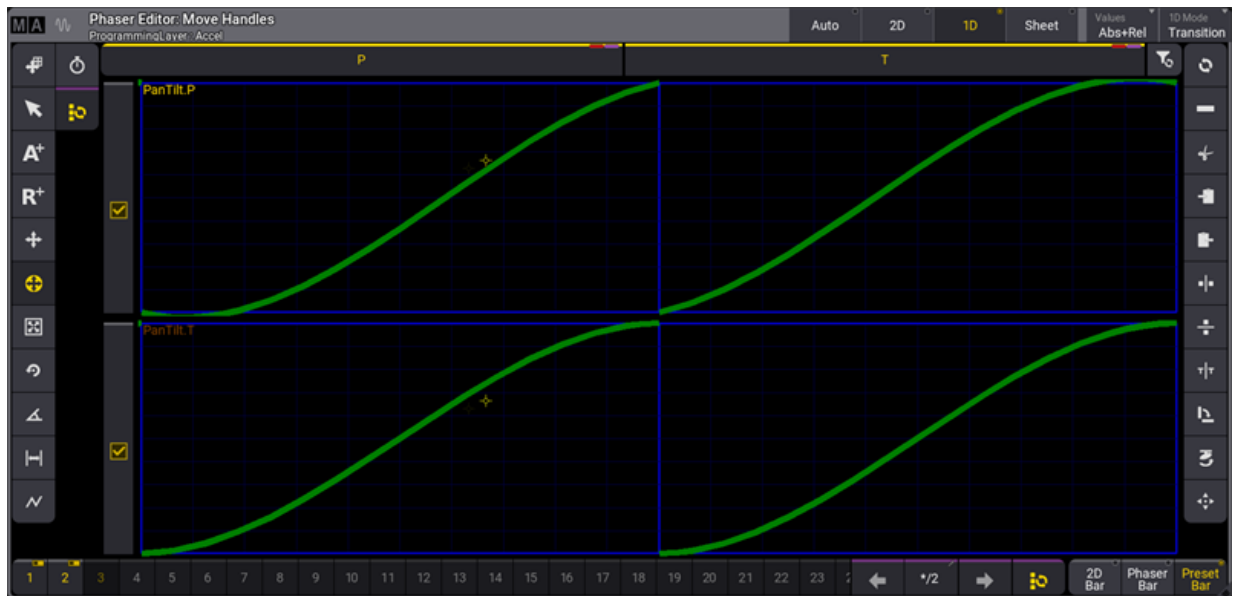
The checkboxes to the left of the attribute lines represent the status of those attributes in the **At Filter**. These checkboxes relate directly to the attribute buttons across the top of the Phaser Editor. A yellow checkmark indicates that adjustments are allowed on the corresponding attribute. An empty, grey box indicates that adjustments are not allowed.

Tap the **1D Mode** button at the right end of the title bar to toggle between the two available 1D drawing modes. These modes change the vertical scale of the 1D layouts.

- **Value** - In this mode, the vertical scale of each 1D layout equals the full range of values available for the displayed attribute.



- **Transition** - In this mode, the bottom of the vertical axis represents the beginning value for each step, and the top represents the ending value for the step.



Phaser Editor in 1D View Mode showing Transitions

Sheet View Mode

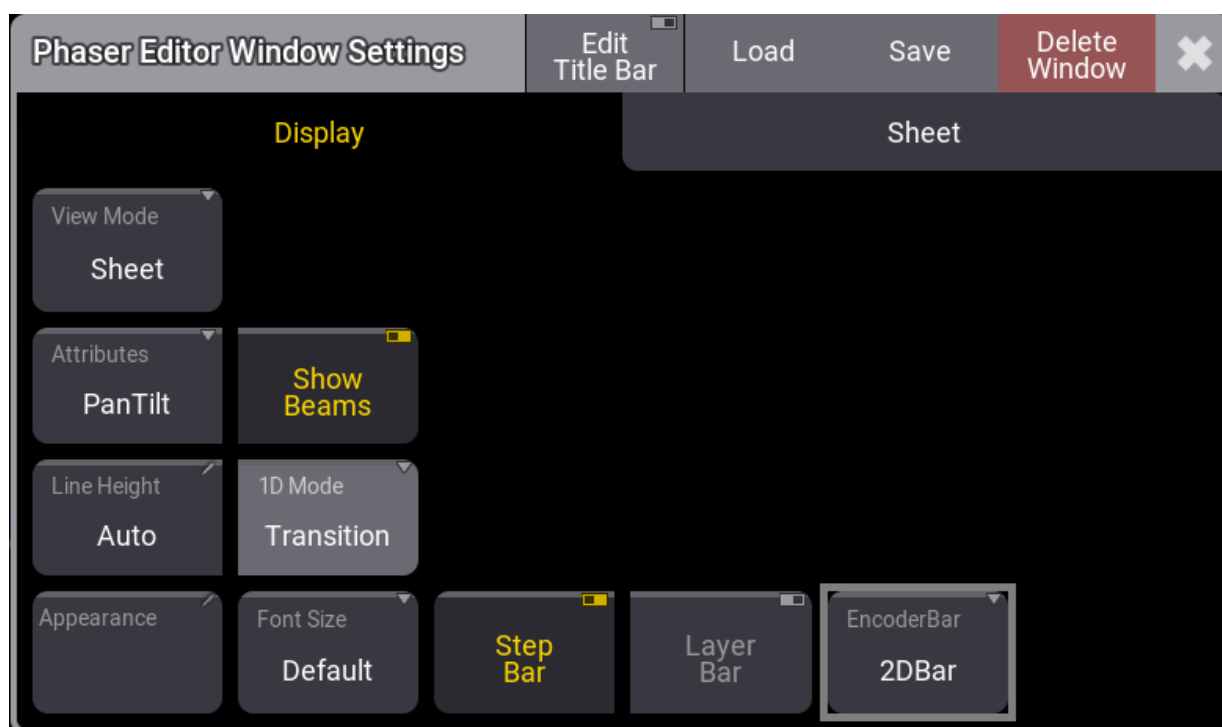


Phaser Editor displayed using Step view mode

The step view mode displays phaser steps using a configurable spreadsheet format similar to the fixture sheet. Beneath all of the relevant spreadsheet data, this mode also includes 1D layouts for each attribute included in the phaser.

The default configuration of the sheet view mode arranges phaser steps as columns and layers as rows. Layers that affect all steps of the phaser appear only under step 1. Each cell in the sheet displays an overview of all values for the layer in that step. In cases where the cell includes multiple values or presets, the sheet displays the lowest and highest values in the cell separated by two dots.

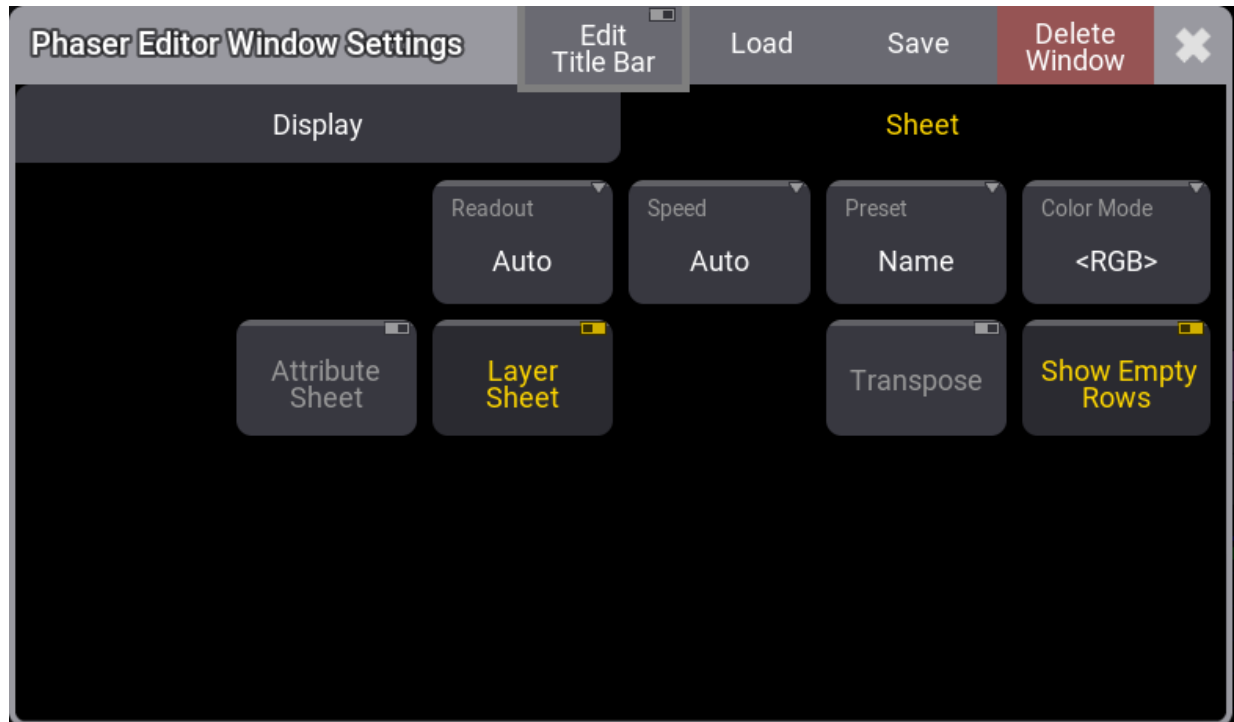
The **Display** tab of the Window Settings pop-up includes some common **Window Settings** as well as some that are specific to the Phaser Editor.



Display tab of the Phaser Window Settings Pop-up

- **Attributes** - Defines which position attributes are used when creating a position phaser. Tap to toggle between **PanTilt**, **XY**, **XZ**, and **YZ**.
- **Step Bar** - Tap to show or hide the Step Bar.
- **Layer Bar** - Tap to show or hide the Layer Bar.
- **EncoderBar** - Defines which encoder bar is displayed in addition to the step bar. Tap to toggle between **2D Bar**, **Phaser Bar** and **Preset Bar**.

The **Sheet** tab of the Window Settings pop-up includes some common **Window Settings** as well as some that are specific to the Sheet view mode of the Phaser Editor.



Sheet Tab of the Phaser Window Settings Pop-up

- **Attribute Sheet** - Tap to show or hide individual rows for each attribute included in the phaser. If **Layer Sheet** is enabled, each layer will include all of these individual layers.
- **Layer Sheet** - Tap either show all layers simultaneously or hide most layers, showing only the layer that is currently accessible on the attribute encoders.
- **Show Empty Rows** - Tap to show or hide empty rows.

Tools

The tool buttons on the left side are:



Phaser Editor left side toolbar



- **1 - Move Area:**
Moves the entire blue pan/tilt square to show relative values that can be outside the position range.
- **2 - Select:**
Selects single steps or uses lasso selection for multiple steps. The selection can be filtered to **Absolute** and **Relative** steps by toggling the respective buttons in the title bar.
- **3 - Add Absolute:**
Adds a step using absolute value. The encoder bar follows the absolute layer. A filled rectangle represents absolute values. The **Single Step Mode** is automatically activated as long as you hold a finger or the mouse cursor in the editor while adding an absolute point.
- **4 - Add Relative:**
Adds a step using a relative value. The encoder bar follows the relative layer. An empty rectangle represents relative values. The **Single Step Mode** is automatically activated as long as you hold a finger or the mouse cursor in the editor while adding a relative point.
- **5 - Move Point:**
Moves selected steps. It is also possible to use the **Align** function. The **Single Step Mode** is automatically activated as long as you hold a finger or the mouse cursor in the editor while moving a single point.
- **6 - Move Handles (The yellow highlight shows this is the current tool.):**
Move handles of selected steps. It is also possible to use the **Align** function. Each point/step has two handles to influence acceleration and deceleration at the same time. Each handle has a cyan line connected to the point it controls. Move both handles of a point by touching the point and dragging in the desired direction. A handle can be moved individually by touching and moving the handle (small yellow circle).
- **7 - Change Size:**
Enlarges/reduces all selected steps evenly from the center point of the selection. It is also possible to use the **Align** function.
Pressing **MA** while changing the size enlarges/reduces all selected steps from the center point of the selection on the horizontal or vertical axis.
- **8 - Change Rotation:**
Rotates the selected steps around the center of the selection. It is also possible to use the **Align** function.
Pressing **MA** while changing the rotation, rotates the selected steps around the center of the selection in 5-degree increments.
- **9 - Change Phase:**
Opens an overlay with yellow dots symbolize the value of the phase for each fixture. Here the phase value can be shifted independently from the selected steps for the selected fixtures. The **At** filter can be used to move phases for individual attributes. It is also possible to use the **Align** and the **Transition** mode functions. If several attributes are displayed along with different phases, the selected feature groups are displayed in a brighter color. If the fixtures are allocated to grid positions, the values of the phases can be vertically or diagonally aligned. Selecting this also changes the functions on the right side to quick selections for alignment. This changes the tool buttons on the right side to give quick access to standard phase values and a **Reset** and **Invert** button.

- **10 - Change Width:**

The width can be changed with this tool. It changes the functions on the right side to shortcuts of different percentages. If all steps are selected and the **Speed** shortcut is tapped, then the movement speed is equalized between the steps. Default is that each step has the same width with no regard to the distance the fixtures are moving in the step. Equalizing the step width makes the fixture move at the same speed by adjusting the width. There is a **Reset** shortcut that resets the width. Default is a width of 100%. This means that each step has the same width with no regard to the travel distance of the fixtures in each step.

- **11 - Select Form:**

Selecting a form overwrites the values of the transition layer, accel layer and, decel layer to create the desired effect. Available forms are listed in the toolbar on the right side and include Rectangle, Sawtooth, Sine and Cosine.

- **12 - Change Speed:**

Tap to display various speed tools in the right side toolbar. Available speed tools include functions which multiply or divide the current speed of the phaser by a given amount. Special speed options include:

- **Loop:** Tap **Loop** to multiply the current phaser speed by the number of steps currently in the phaser.
- **Fixture:** Tap **Fixture** to divide the current phaser speed by the number of fixtures in the current selection.

- **13 - Select all Steps:**

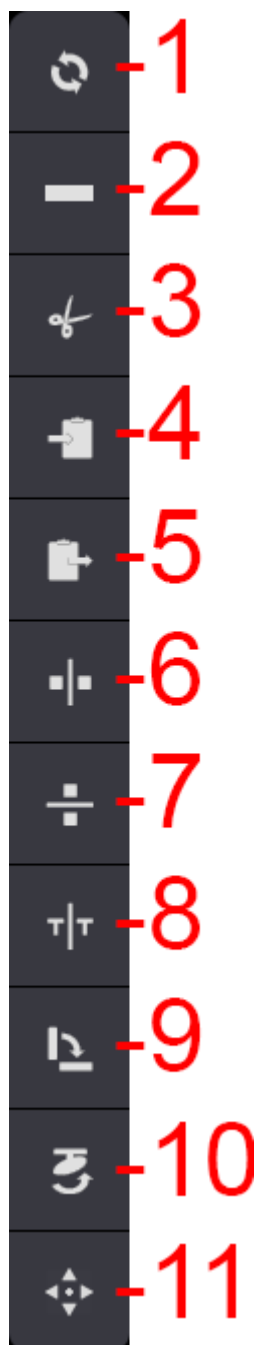
This is not a tool like the others. Tapping this toggles between all steps selected or the selection of single steps. It keeps the previously selected tool active.



Hint:

Some tools don't require the user to precisely touch points in the 2D and 1D layouts in order to adjust them. Instead, make sure the desired steps are enabled in the **Step Bar**, and use the blue grid areas in the **Phaser Editor** as trackpads to make the desired adjustments.

The standard operational functions on the right side are:



- **1 - Reset:**
Reset spline of the selected steps.
- **2 - Delete**
Deletes the selected steps.

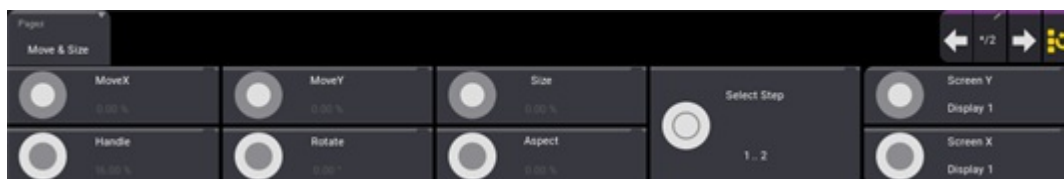
- **3 - Cut Programmer:**
Removes all programmer values in the currently selected fixtures and moves them to the clipboard.
- **4 - Copy Programmer:**
Copies all programmer values in the currently selected fixtures and moves them to the clipboard.
- **5 - Paste Programmer:**
Inserts all values from the clipboard to the currently selected fixtures in the programmer.
- **6 - Mirror X:**
Mirrors all selected points on the x-axis.
- **7 - Mirror Y:**
Mirrors all selected points on the y-axis.
- **8 - Mirror Time:**
Swaps the order of the steps. Select all steps first. This is the same as reversing the direction of the movement.
- **9 - Swap XY:**
Swaps the X and Y axes of the selected steps.
- **10 - Flip:**
Flips the position of the selected fixtures.
- **11 - Reset Zoom:**
Resets the area with the blue square.

Phaser Encoder Toolbar

Depending upon the type of encoder toolbar selected with the radio buttons in the bottom-right corner of the **Phaser Editor**, the encoder toolbar changes to display helpful tools, which are specific to the manipulation of Phaser data.

2D Bar

To enable the 2D Bar, tap **2D Bar** in the bottom-right corner of the Phaser Editor.



2D Phaser Encoder Bar

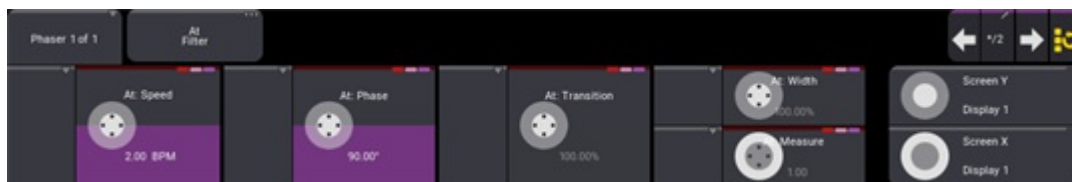
This encoder toolbar includes the following encoder controls:

- **MoveX:** Turn the inner encoder to adjust the X position in the 2D Phaser grid of all steps currently enabled in the **Step Bar**.
- **MoveY:** Turn the inner encoder to adjust the Y position in the 2D Phaser grid of all steps currently enabled in the **Step Bar**.
- **Size:** Turn the inner encoder to adjust size of all steps currently enabled in the **Step Bar**. This allows encoder access to the **Change Size** tool mentioned above.

- **Handle:** Turn the outer encoder to adjust the handle length of all steps currently enabled in the **Step Bar**. This allows encoder access to the **Move Handles** tool mentioned above.
- **Rotate:** Turn the outer encoder to rotate all steps currently enabled in the **Step Bar**. This allows encoder access to the **Change Rotation** tool mentioned above.
- **Aspect:** Turn the outer encoder to adjust the aspect ratio of all steps currently enabled in the **Step Bar**. This tool simultaneously adjusts positions of points as well as length and direction of handles in order to create a wider or taller version of the current 2D form.
- **Select Step:** Turn the inner or outer encoder to select individual steps. Press the inner encoder or the outer encoder key to toggle between selecting all steps and a single step.

Phaser Bar

To enable the Phaser Bar, tap **Phaser Bar** in the bottom-right corner of the Phaser Editor.



Phaser Bar

This encoder toolbar allows for adjustment of **Speed**, **Phase**, **Transition**, **Width**, and **Measure**; based upon the current fixture and step selection, as well as the current status of the **At Filter**.



32.2. Create a Sinus Dimmer Phaser

This topic details a couple of examples of how to create a sinus dimmer phaser using keys and buttons as well as the phaser editor and encoder bar. There are many ways to reach the same goal in grandMA3. Many of the steps in the examples below are interchangeable while producing identical results. These examples use a combination of keys and the calculator interface. If you are working with the onPC software, it might be easier to type commands into the command line rather than use the on-screen keys.

It is recommended to read the [Phasers topic](#) and the [Phaser Editor topic](#) before this topic.

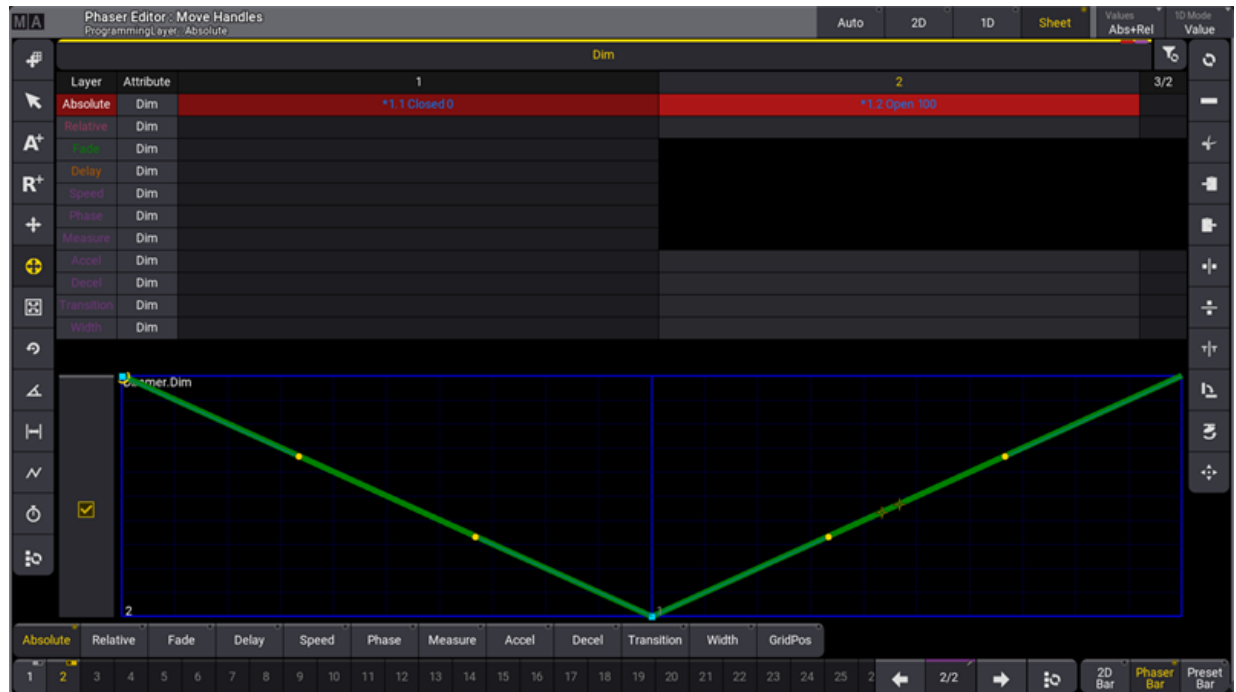
Requirements:

- Have a show with some lights patched.
- An open [Phaser Editor](#) window with the Step Bar enabled is recommended, though it is also possible to use the temporary Phaser Editor.
- Arranging the fixtures in the 3D window can be useful, but it is not a requirement.
- As it is highly recommended to use presets while programming, an open Dimmer preset pool is recommended with presets for dimmer values of 100% and 0%.

Quickly Create Steps and Apply Adjustments with the Phaser Editor




1. Select the desired fixtures. For example: **Fixture** **Thru** **Please**
2. Choose a dimmer preset for step one: Press and hold **Step** and tap the preset with the 0% dimmer value.
3. Choose a dimmer preset for step two: While still holding **Step**, tap the preset with the 100% dimmer value.

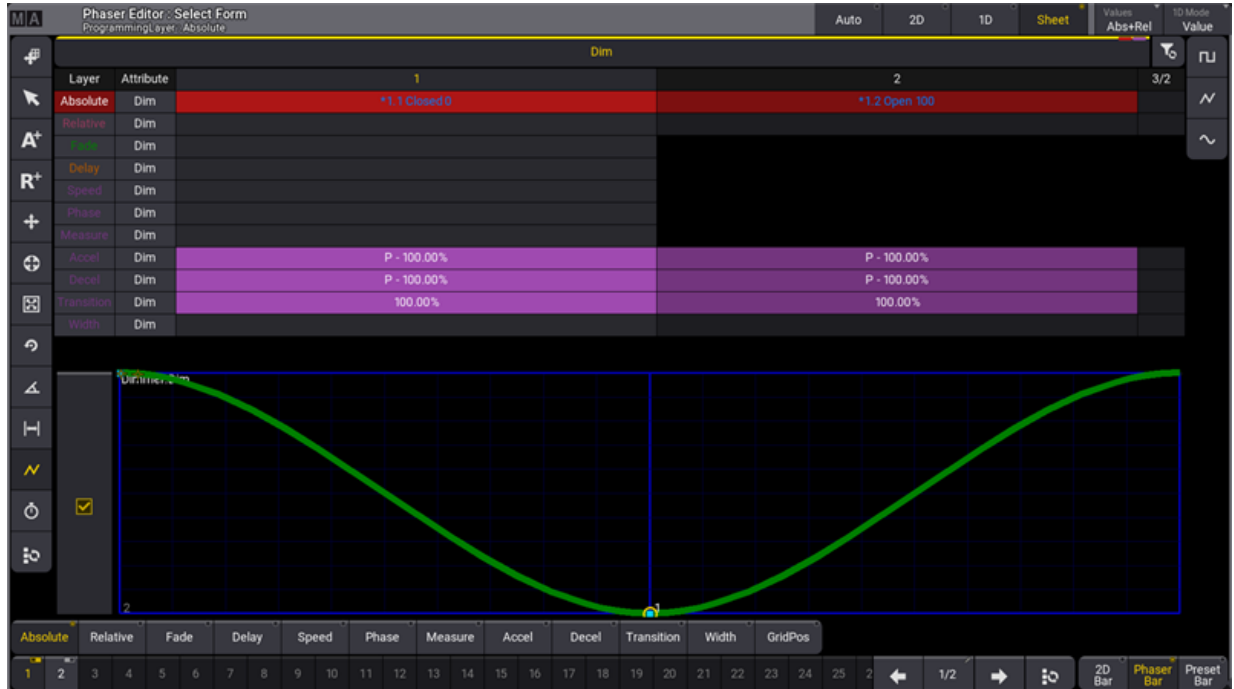
This is the base for the phaser. There are now two steps with a dimmer preset of 0% in the first step and a preset of 100% in the second step. All of the fixtures are changing together from step one to two with a linear fade (not yet using a sine curve). The Sheet view mode of the [Phaser Editor](#) shows:



Linear Dimmer Phaser with No Phase Distribution


Adjust the curve of both steps using tools in the Phaser Editor:

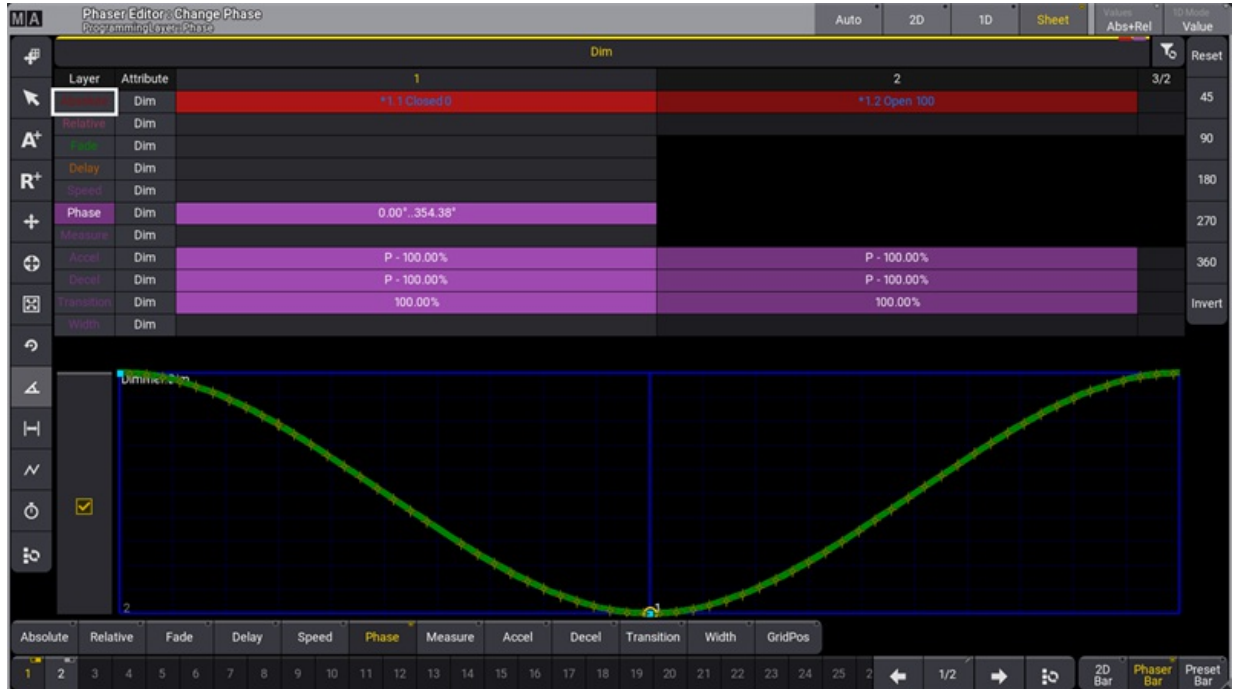
4. Select all steps by tapping the  button at the right end of the Step Bar or at the bottom of the left-hand tool bar.
5. Tap to select the Select Form tool () in the left-hand tool bar.
6. In the right-hand tool bar, tap the sine wave button () to select the sine wave form.



Sinus Dimmer Phaser with No Phase Distribution

Adjust the phase of all fixtures using tools in the Phaser Editor:

7. Tap to select the Edit Phase () tool in the left-hand tool bar.
8. In the right-hand tool bar, tap the **360** button to evenly distribute the selected fixtures over the wave form.



Sinus Dimmer Phaser with Full Phase Distribution

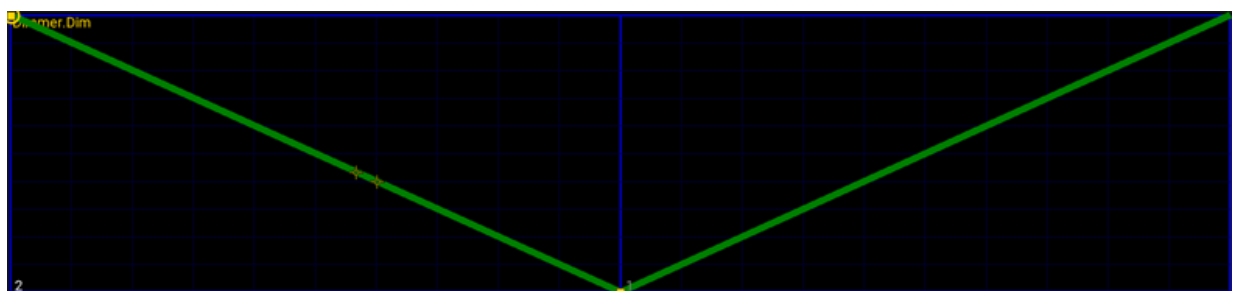
Store the phaser into a preset or a cue.

Alternate Methods to Create a Sinus Dimmer Phaser

Follow these steps to create a sinus dimmer phaser:

1. Select the desired fixtures. For example: **Fixture Thru Please**.
2. Set them at a dimmer value of 0: **At 0 Please**.
3. Create step 2 and select it: **MA + Next** (Next Step).
4. Set the fixtures at full: **Full**.

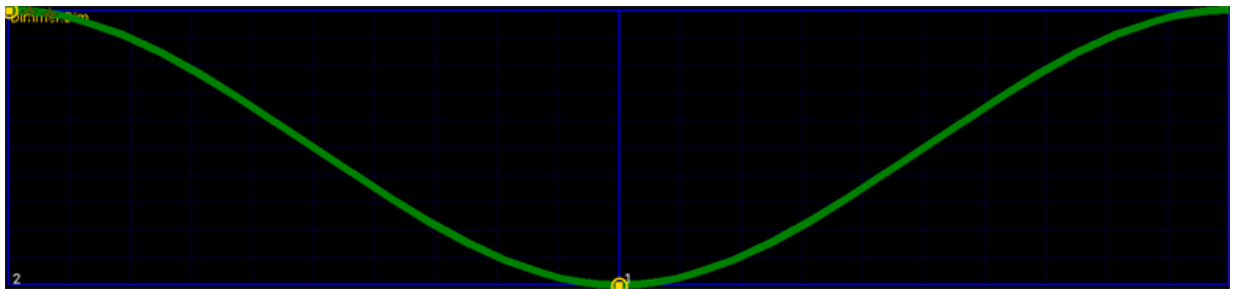
This is the base for the phaser. There are now two steps with a dimmer value of 0 in the first step and a value of 100 in the second step. All the fixtures are changing together from step one to two in a linear direction and do not fade using a sine curve. It looks like this in the **Phaser Editor**:



The curve is corrected by adjusting the acceleration and deceleration in both steps.

5. Select both steps: **MA** + **Set** (Step Toggle).
6. The steps need to have an accelerate value of -100. This is done using the **Encoder Toolbar**.
 1. Tap **Dimmer** to make sure it is selected.
 2. Tap **Accel** to select the acceleration layer.
 3. Tap the left attribute encoder to open the calculator.
 4. Tap **+/- 1 0 0 Please**. This gives a known and precise value. If **Handles** are used then it might not be as precise as typing a value.
7. Do the same for the **Decel** layer.

Now the attribute accelerates out of and decelerates into each step. The curve now looks like this:



8. The final thing to do is to spread out the fixtures using the phase value. This is also done from the encoder toolbar.
 1. Tap **Phase** to select the phase layer.
 2. Tap the left attribute encoder to open the calculator.
 3. Tap **0 Thru 3 6 0 Please**.

Now the fixtures are spread out equally over the entire phase of the phaser loop.



This is a sinus phaser that can be stored in a cue or preset.

It is created using a specific selection of fixtures so remember to select the desired **Preset Mode** when **storing a preset**.

32.3. Create a Simple Circle Phaser

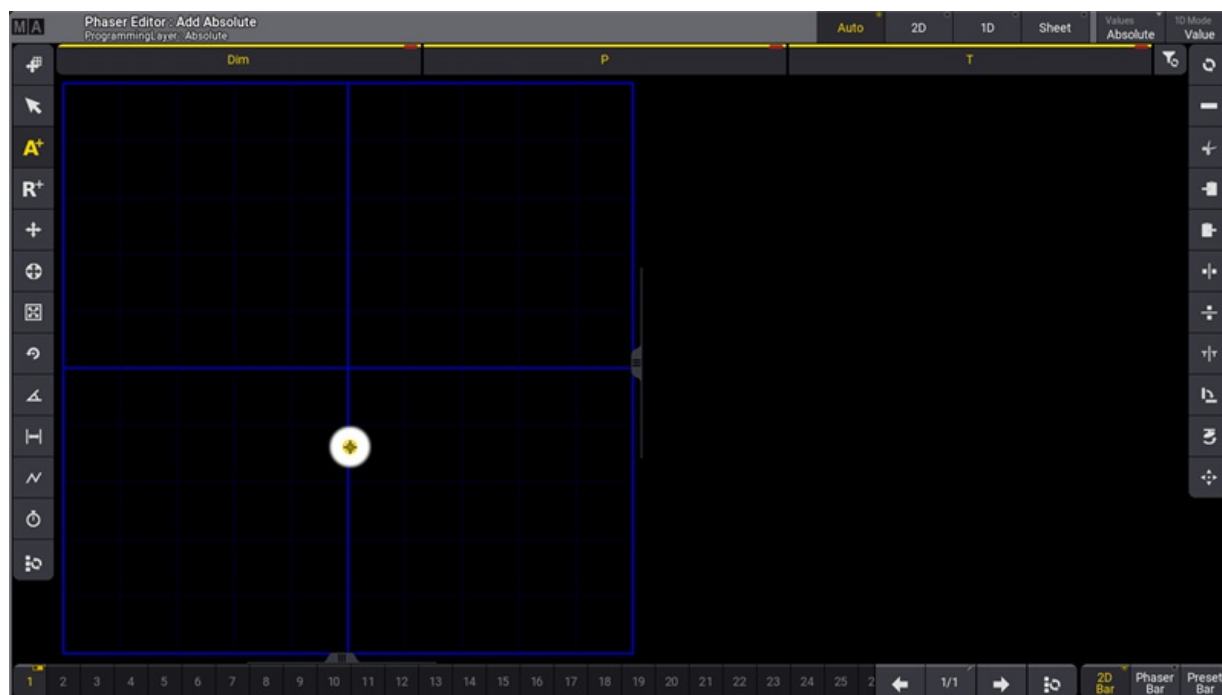
This topic presents an example of one method for creating a simple circle phaser. It uses absolute position values and will always move the lights in the specified circle. See the [Create Circle Phaser Around Position](#) topic for an example that covers creating a relative phaser circle, which can be used with different absolute base positions.

Requirements:

- Have a show with some moving lights patched - for instance, the demo show.
- An open **Phaser Editor** window is needed.
- Arranging the fixtures in the **3D window** can be useful, but it is not a requirement.

Follow these steps to create a circle:

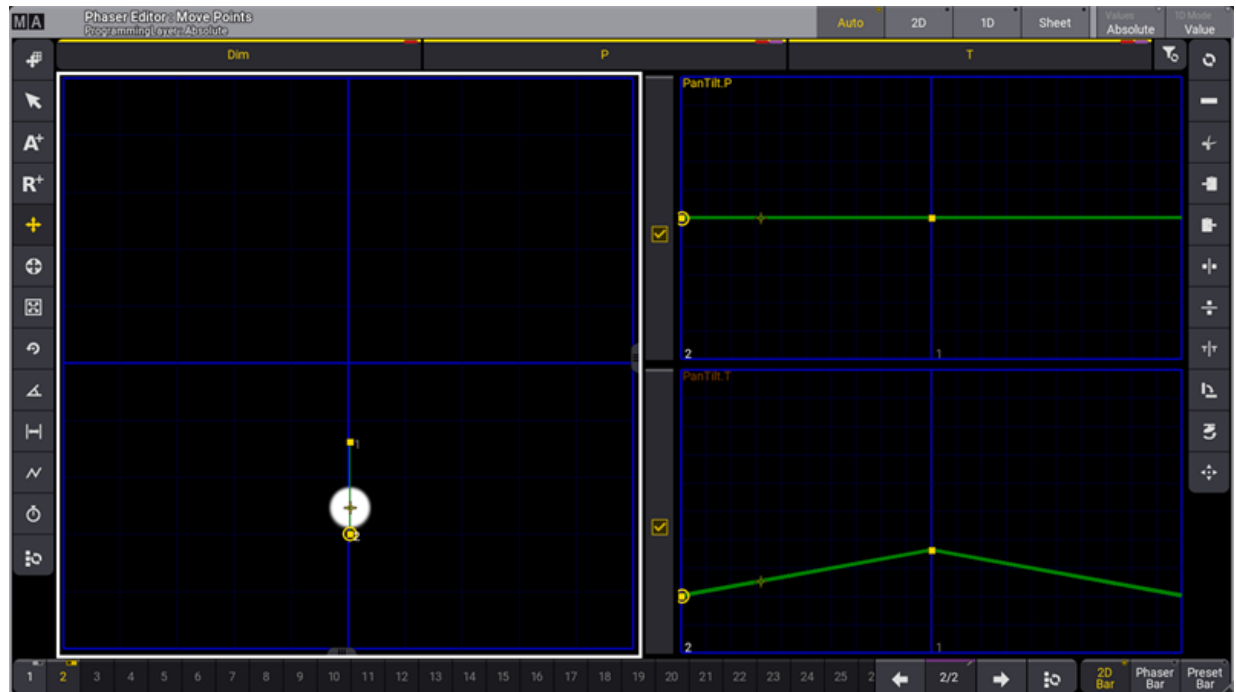
1. Select the desired fixtures and bring their dimmers to full.
The Phaser Editor shows a white beam in the center of the blue 2D grid.
2. Tap the **A+** button on the left of the editor.
3. Tap somewhere in the blue grid; for instance, just below the center.



Phaser Editor with fixtures turned on and one absolute position

This creates the first absolute point. The selected fixtures move to this static position. This is an absolute position value in step one.

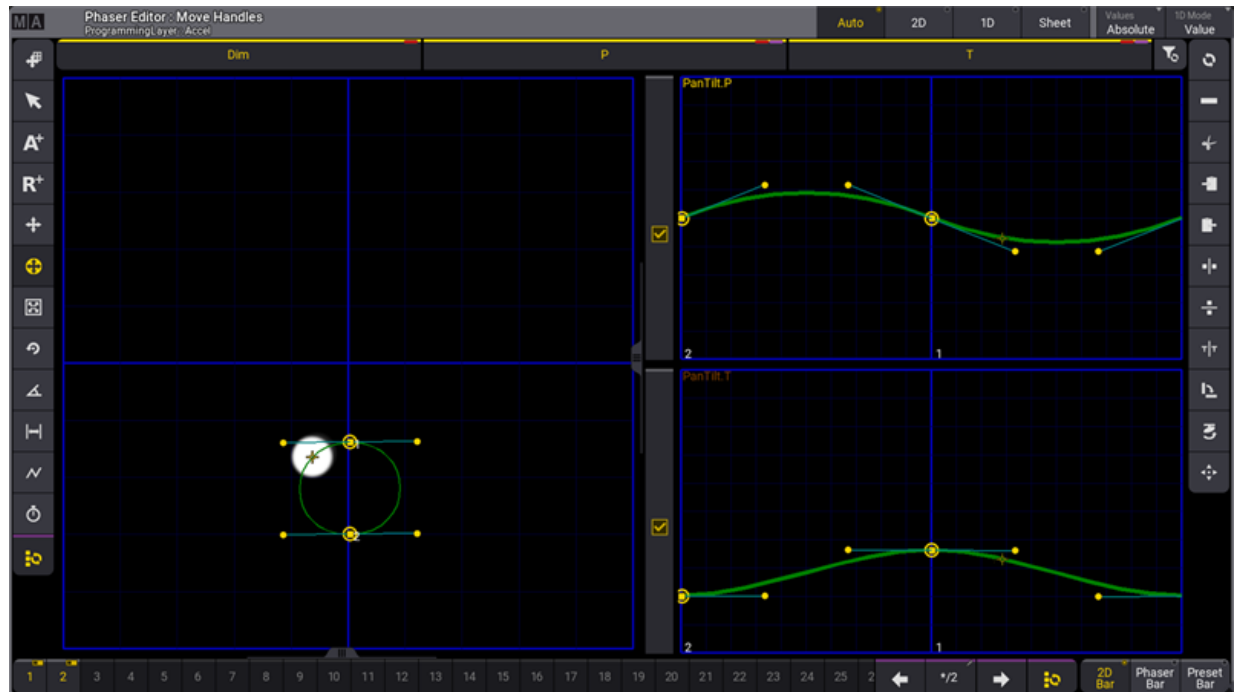
4. Tap somewhere else in the blue grid to create a second absolute position. Please tap 90 degrees vertically to the first point and do not cross the horizontal blue line - otherwise, the circle becomes a figure 8.



Two absolute positions - one in each step

Now, the fixtures move between the two absolute points (two steps).

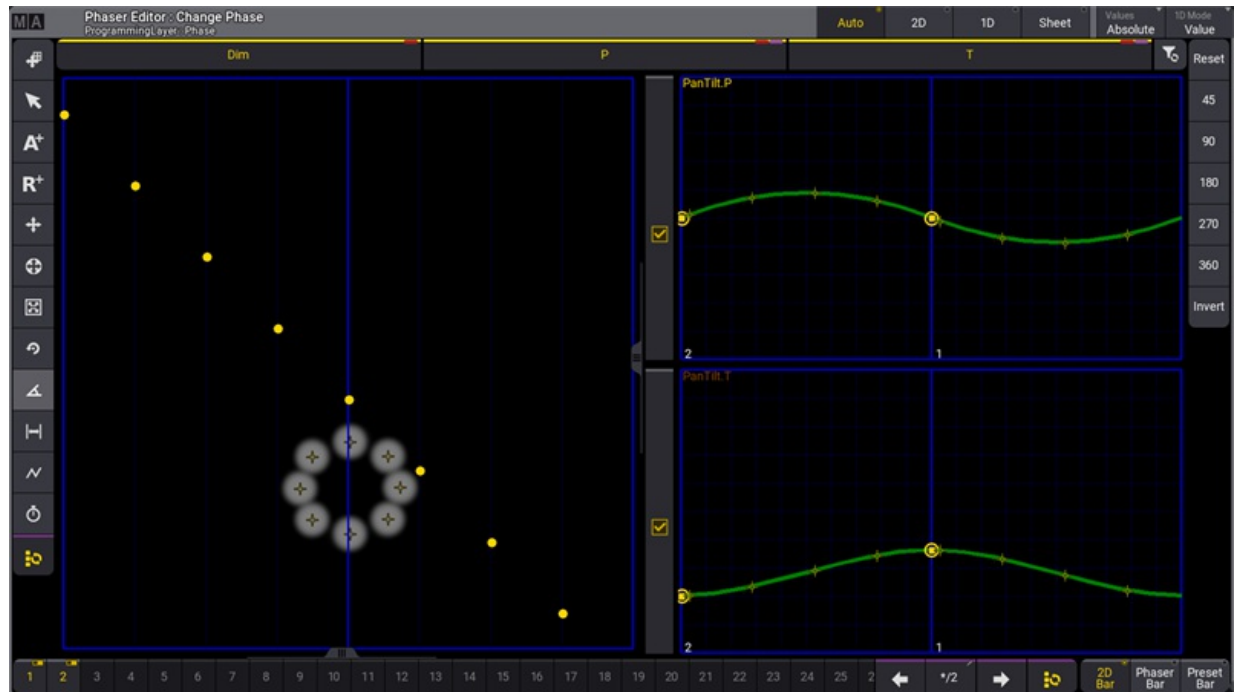
5. Tap the **Select All Steps** button (🔄).
This selects both steps.
6. Tap the **Move Handle** button (⊕).
7. Tap one of the two points and drag horizontally.
8. Release the screen when there is a nice circular movement.



The fixtures now move in a circle

9. Tap the **Edit phase** button (⏏) on the left menu.
10. Tap **360** in the menu on the right
This distributes the fixtures evenly along with the form.

The finished result could look like this:



Finished circle phaser

The **Change Phase** part of the phaser editor represents each fixture as a yellow dot in a grid where the horizontal axis is the phase value.

The circle phaser is now finished and it can be stored into a cue or a preset.



Important:

Following the steps above exactly results in an active dimmer value in the first step of the phaser. If you prefer that your circle phaser only contain position values, deactivate the dimmer attribute before storing.

32.4. Create a Circle Phaser Around a Position Preset

This topic presents an example of one method for creating a simple relative circle phaser.



Hint:

Combining relative phasers with separate absolute presets provides greater flexibility for creating dynamic looks.

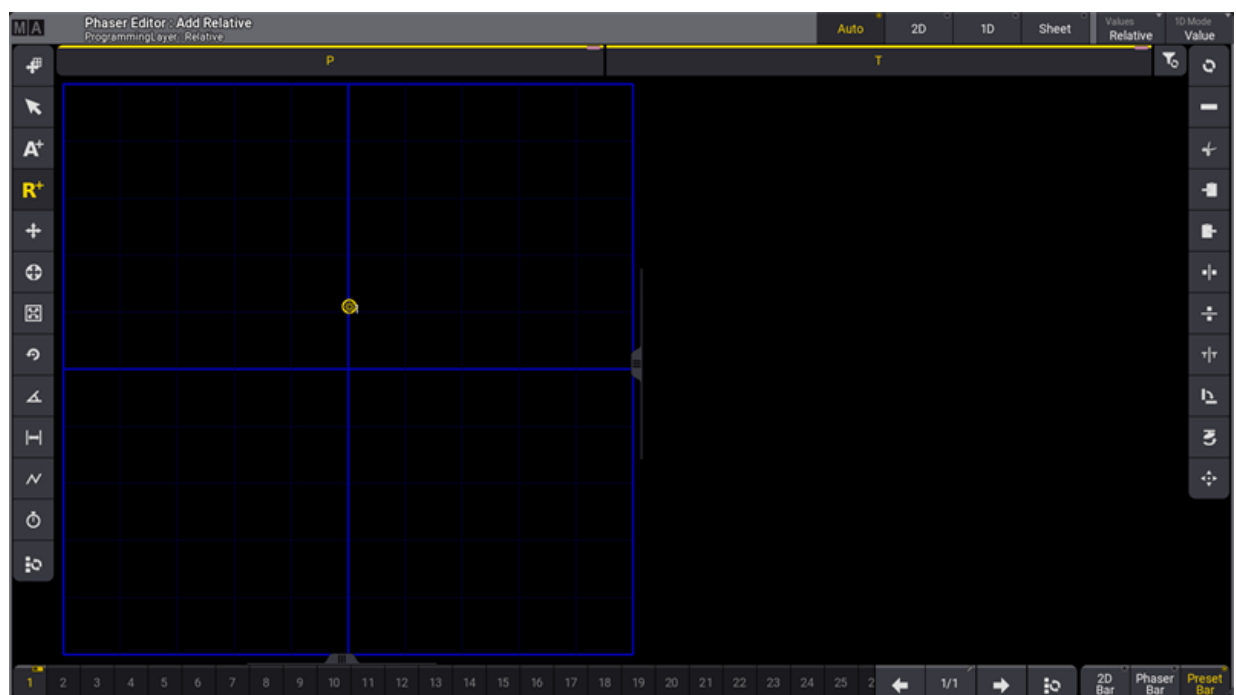
Requirements:

- Have a show with some moving lights patched. The following examples reference presets included in the demo show.
- An open **Phaser Editor** window is needed.
- Arranging the fixtures in the **3D window** can be useful, but it is not a requirement.

This example shows one possible workflow for creating a relative phaser. Alternate workflows can also produce the same results.

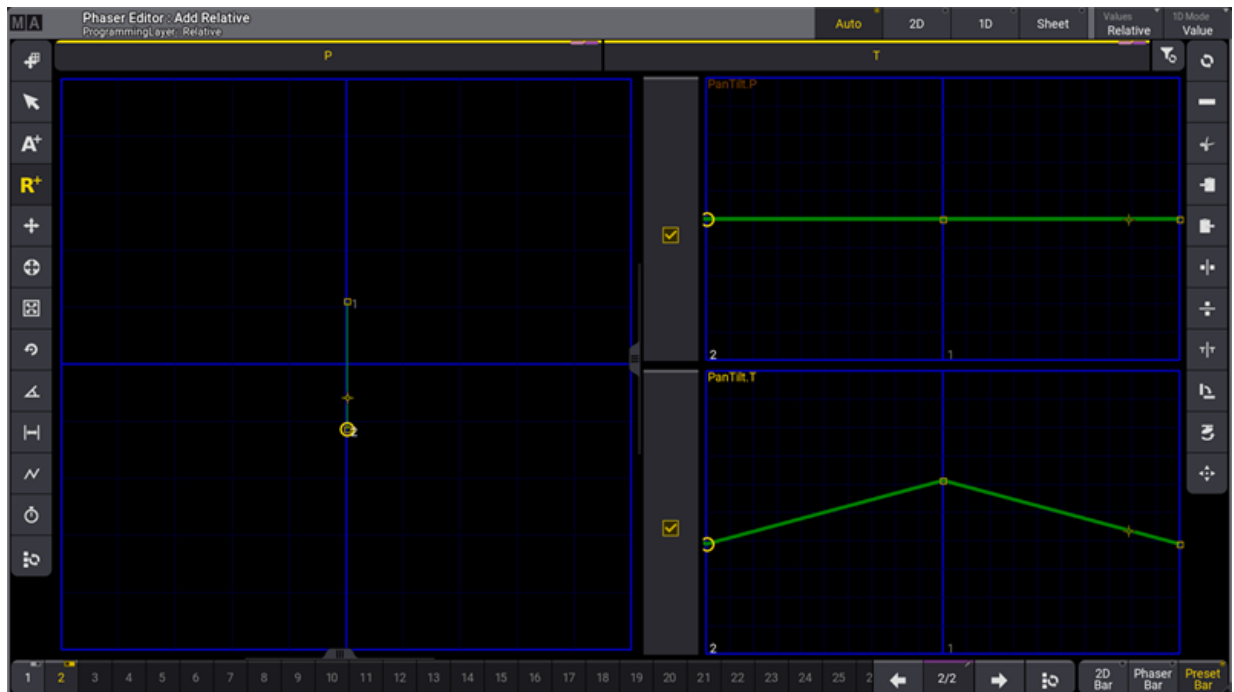
Follow these steps to create a relative circle preset:

1. Select all of the desired fixtures.
2. Tap **R+** on the left side menu in the **Phaser Editor**.
3. Tap just above the center of the blue 2D grid.





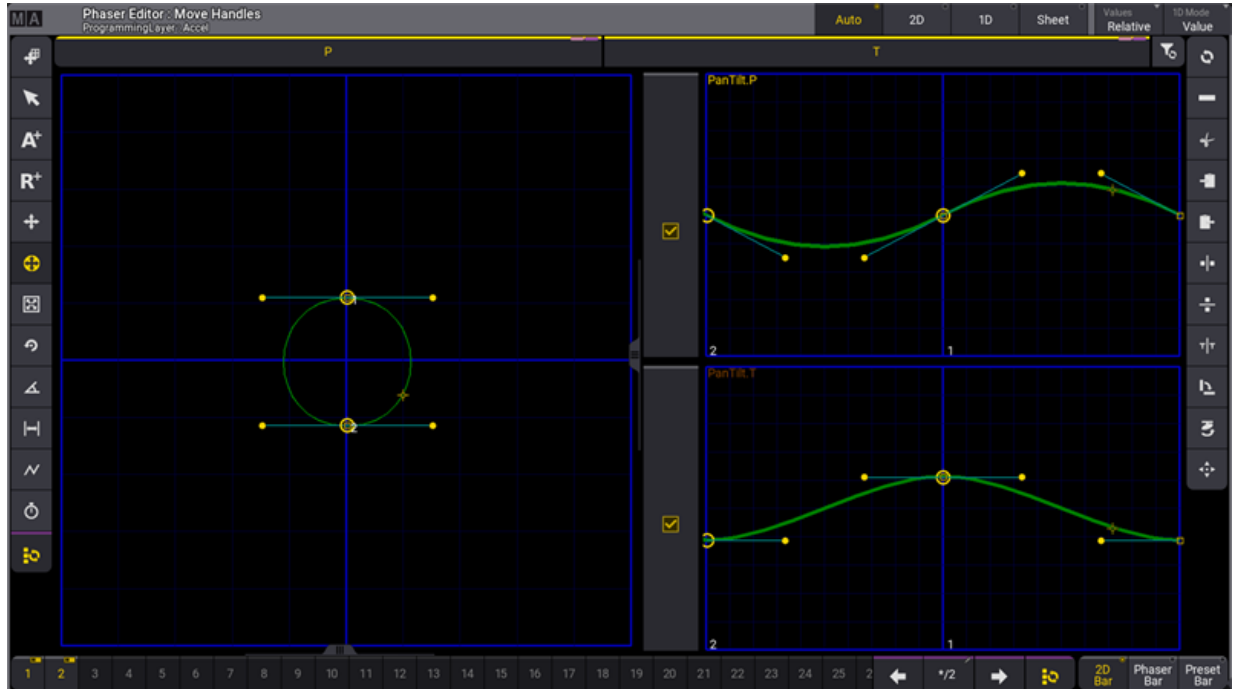
The first point in a relative circle

4. Tap just below the center of the blue 2D grid.
This creates two relative position points.




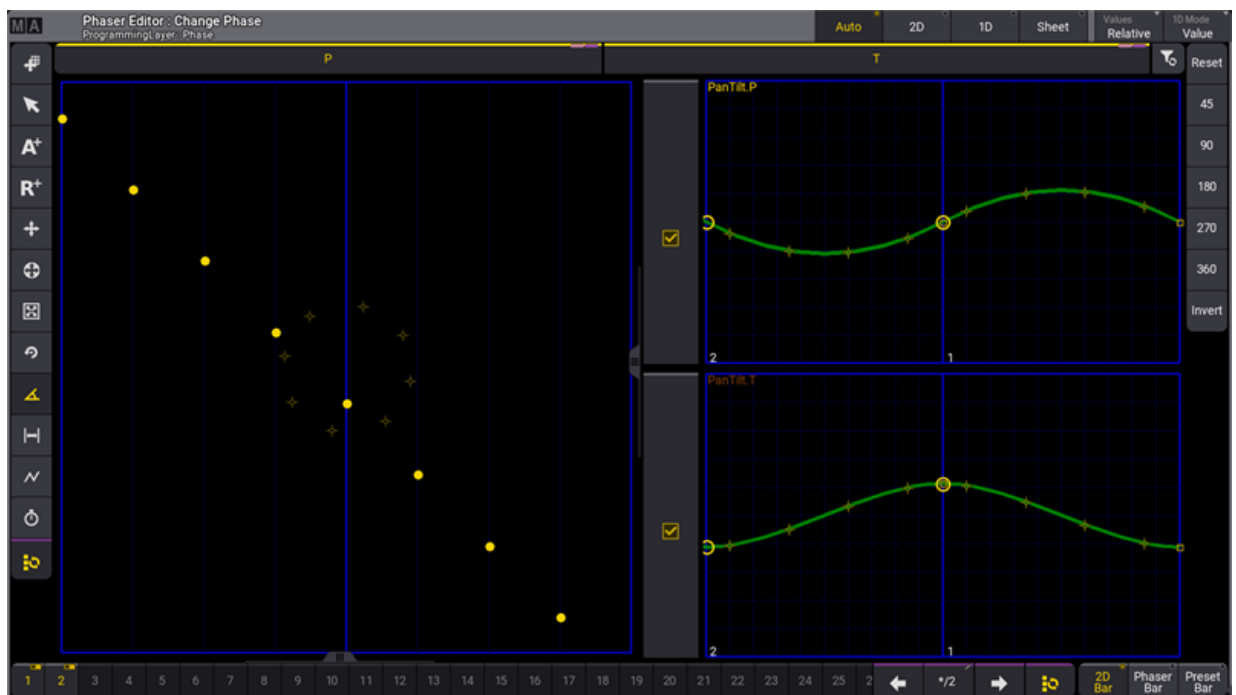
Two steps /points are the basis for the circle

5. Tap the Select All Steps () button (the bottom button on the left menu or the last button in the Step Bar).
6. Tap the Move Handle () button.
7. Tap one of the steps and drag horizontally.
8. Release the screen when the desired circular movement is displayed.



The relative circle with all selected fixtures moving simultaneously.

9. Tap the Change Phase () button on the left menu.
10. Tap **360** in the menu on the right.



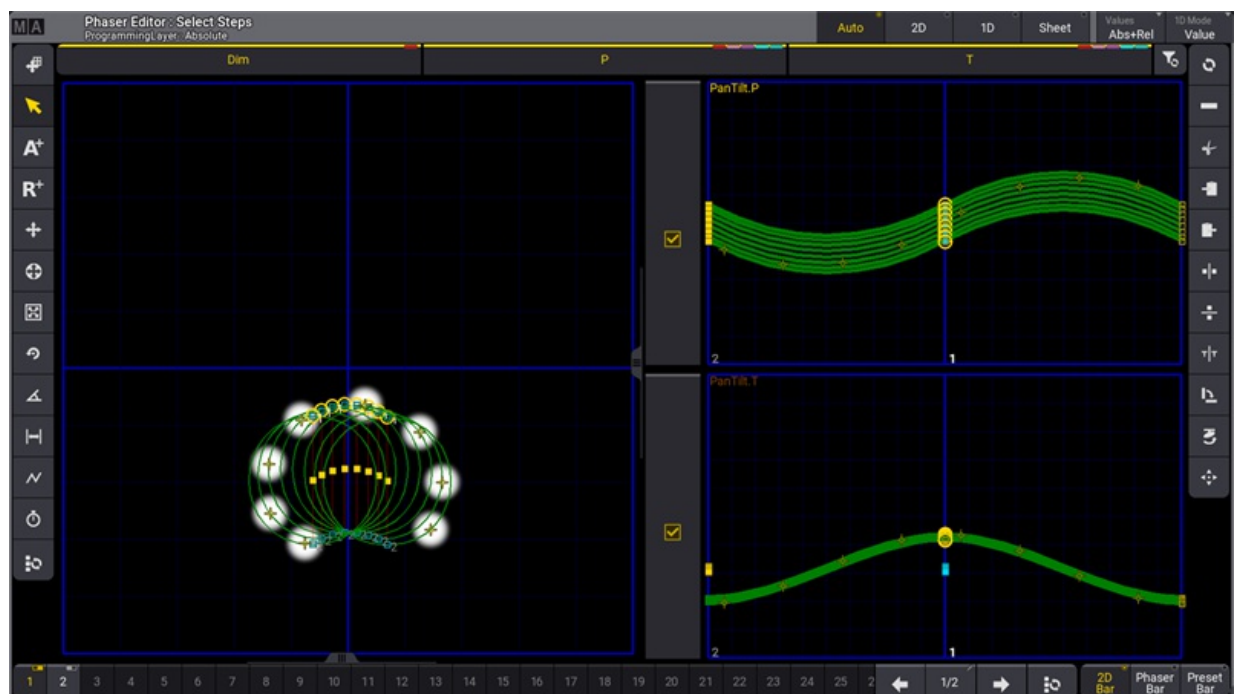
The relative circle with selected fixtures aligned evenly around the form.

11. Store this as a preset.

Follow these steps to create a cue combining this relative circle phaser with a static absolute position:

1. Select the desired fixtures.
2. Tap dimmer preset 2 "Open" to bring the fixtures to full.
3. Tap position preset 3 "Center" to move the lights to the downstage-center position.
4. Tap the new preset with the relative circle.

The resulting output shows the fixtures moving in a circle around the downstage-center position.



Fixtures moving in a circle around the downstage-center position.

Continue editing as desired or store this as a new preset or store as a cue.

32.5. Create Color Rainbow Phaser

This topic presents an example of one possible workflow for creating a rainbow color phaser. This example creates the rainbow chase using three steps with a primary color in each step. An additional example at the end shows an easy way to change the color used in one of the steps.

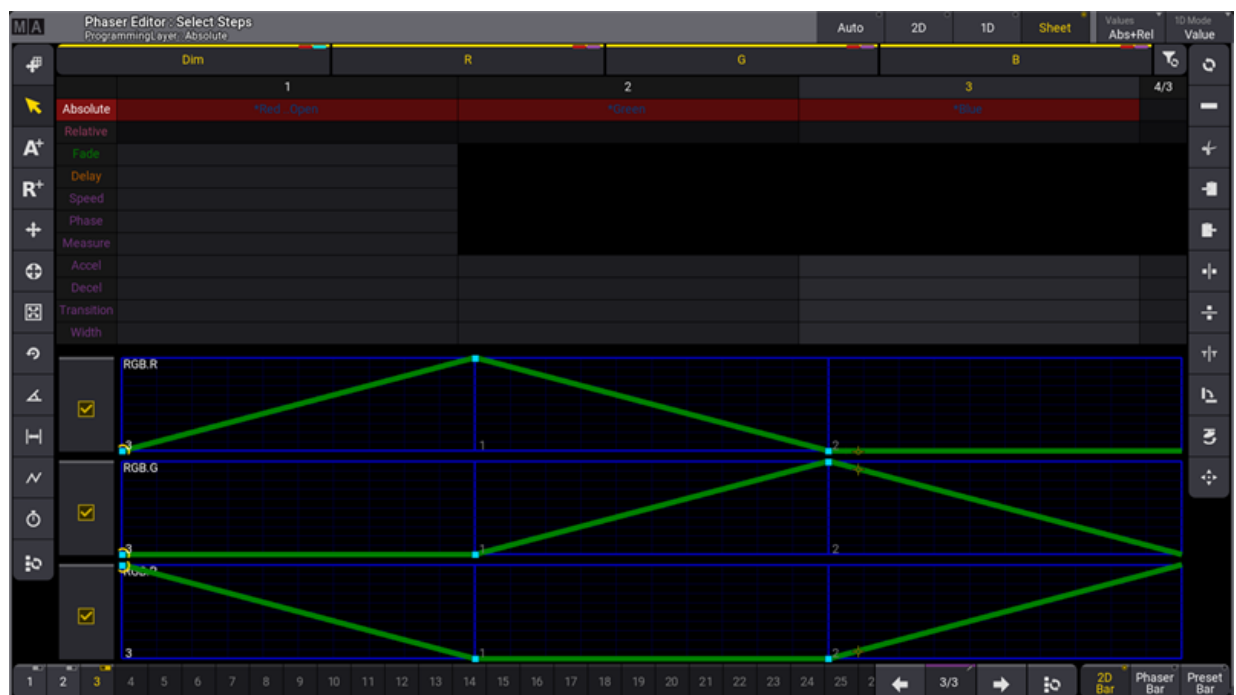
Requirements:

- Have a show with some fixtures that have a color mixing system (not color wheels with static colors). The steps below reference fixtures and presets included in the demo show.
- Have three color presets with Red, Green, and Blue colors stored using the color mixing system. The additional example also uses a White preset. These presets exist in the demo show file.
- An open **Phaser Editor** window is needed.
- An open Color preset pool for quickly calling the desired presets.
- Arranging the fixtures in the **3D window** can be useful, but it is not a requirement.

Follow these steps needed to create the phaser.

1. Select the desired fixtures. The following screenshots use fixtures 1 thru 8 in the demo show.
2. Bring the dimmers to full so the beams are visible.
3. Press and hold the **X5 | Step** key, and tap the red color preset.
4. While still holding the **X5 | Step** key, create the second step by tapping the green color preset.
5. While still holding the **X5 | Step** key, create the third step by tapping the blue preset.

The phaser editor shows the three steps and all referenced presets.



Phaser Editor with a three-step, three-color phaser

All the fixtures now change color together through the rainbow colors. If this is the desired result then store this in a cue or preset.



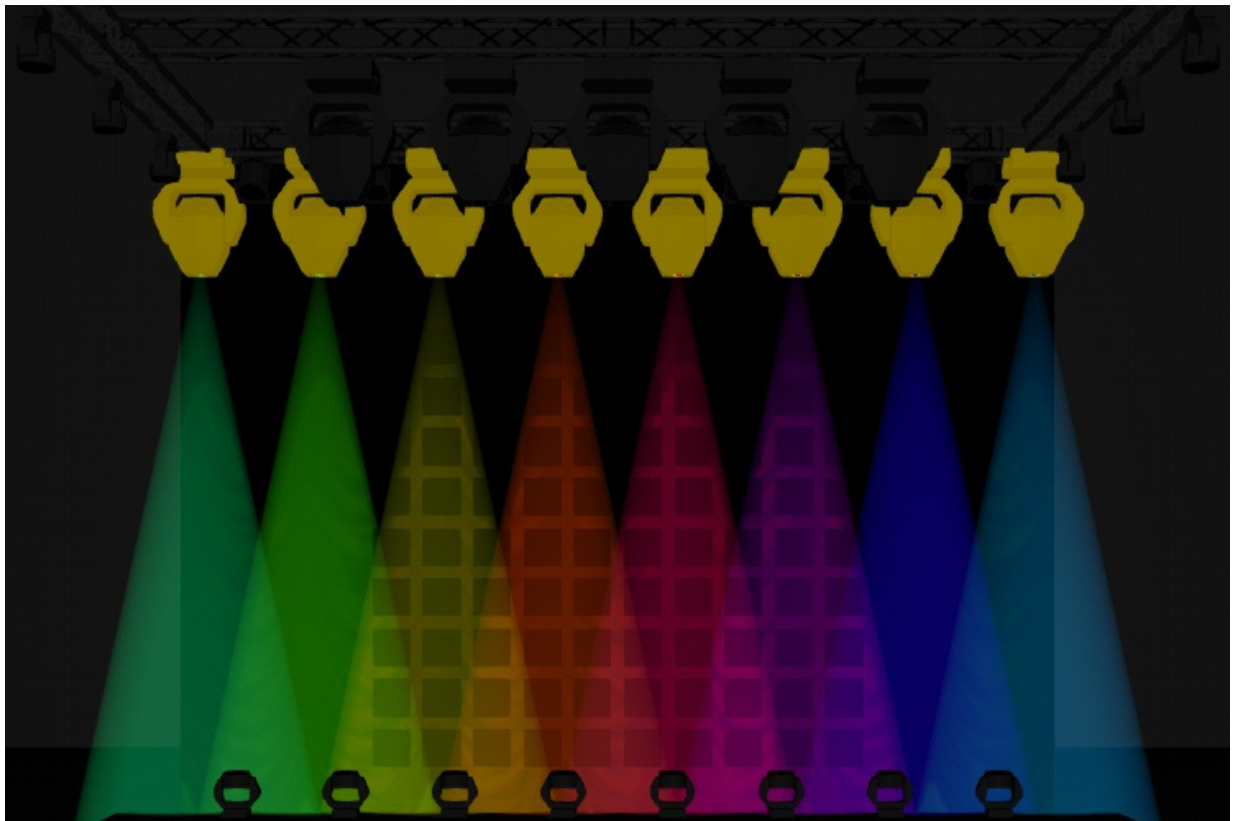
Important:

Following the steps above exactly results in an active dimmer value in the first step of the phaser. If you prefer that your rainbow phaser only contain color values, deactivate the dimmer attribute before storing

To spread the rainbow across the selected fixtures, adjust the Phase:

1. Tap the Change Phase (↔) button on the left menu in the **Phaser Editor**.
2. Tap **360** on the right menu in the **Phaser Editor** to spread the fixtures evenly.

The 3D window shows the result:

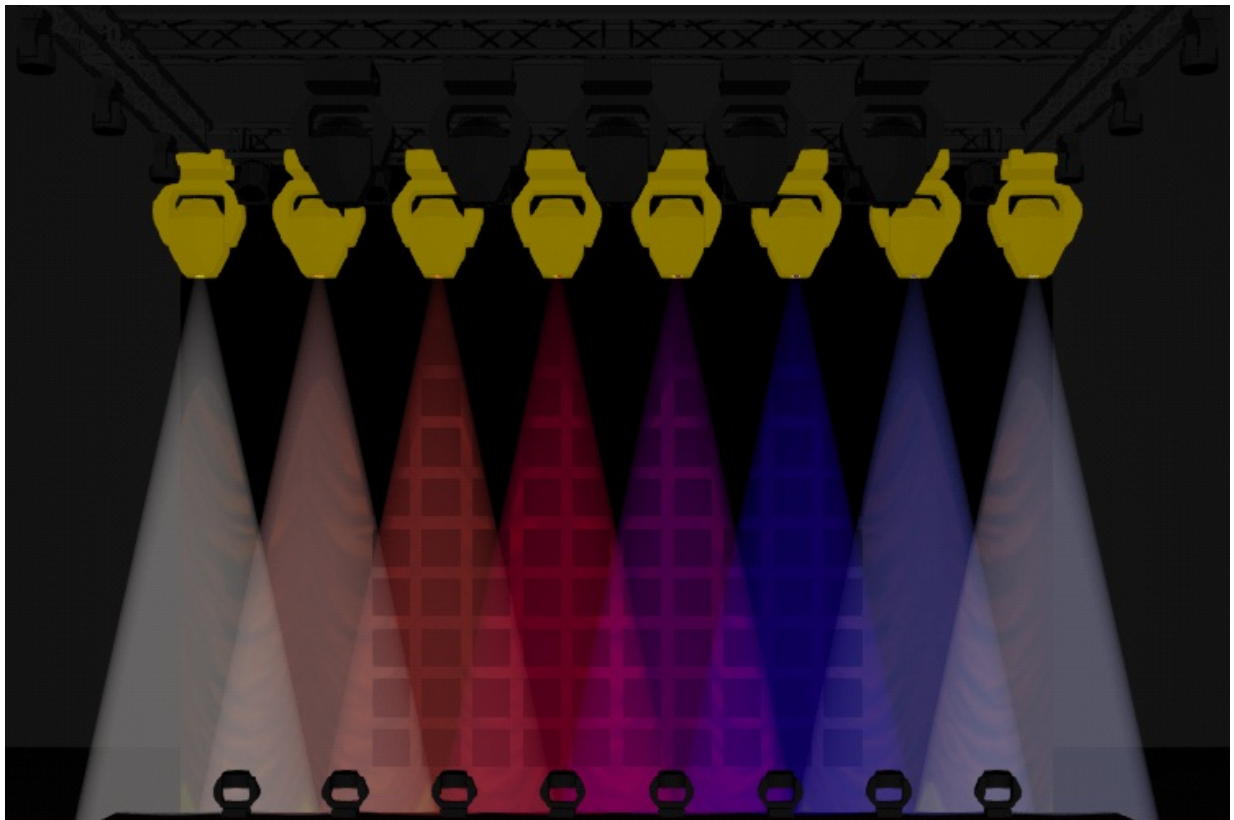


Rainbow chase across a range of fixtures

This can be stored as a preset or in a cue.

Quick Workflow for Changing the Preset Referenced in a Step

In the case that a different combination of colors is desired, the following workflow allows for such adjustments to be made quickly. With the rainbow from the above example still active in the programmer, use the following command to change the preset used in step two from green to white, so that the fixtures transition from red to white, then to blue: Press **X5 | Step** then **2** and tap the white preset.



Red, white, and blue phaser

The resulting phaser now references the red, white, and blue presets.

33. XYZ

Fixtures can be programmed using XYZ values instead of PanTilt values.

This offers some advantages over standard PanTilt programming. For instance, when fixtures move from one position to another, they do it in a straight line from position A to B when using XYZ values. When using PanTilt, the fixtures fade from DMX value A to B, this means that two fixtures often do not follow the same path.

XYZ values can, just like PanTilt, be relative and absolute.

PanTilt values can be mixed with XYZ values in a sequence.

All combinations of PanTilt and XYZ, and absolute and relative, are supported.



Important:

When mixing absolute PanTilt with relative XYZ, the value of the Distance attribute in the PanTilt feature is important. This value defines where the relative XYZ values are added.

Calling XYZ presets with absolute values or hard values in the absolute layer knocks out absolute PanTilt values. The same is true with relative values. Absolute PanTilt values can be combined with relative XYZ values and vice versa.

When knocking in attributes of one position type (PanTilt or XYZ) while the other one has active values in the programmer or is played back, the values will be converted to the knocked in position type and converted to match the same position on stage.

If several playbacks with values for both positioning systems run, Pan, Tilt, and Distance will be knocked in.

If the position feature group is knocked in while a transition from one system to the other is happening, Pan, Tilt, and Distance will be knocked in.

On FeatureGroup "Position".XYZ knocks in all XYZ attributes, while **On EncoderPage "Position".2** will knock in only the attributes that are currently on this encoder page (in most cases, this will be X, Y, Z, and Flip).

The **OffWhenOverriden** setting of sequences turns off playbacks with relative PanTilt when relative XYZ is played back, and vice versa.

XYZ Attribute Range

The absolute XYZ attributes range from 0 to 100 % of the space they use.

The relative XYZ attributes range from -100 to 100% of the space they use.

The default space for the absolute values is the default stage space volume. Read about stage spaces in the **Stages topic**.



The physical readout of the XYZ attributes displays the values in meters of the stage (when MArker is set to 0) or of the size of the target space of the selected MArker fixture.

Turning the encoders of the XYZ attributes by one click in the encoder resolution Coarse changes the value in physical readout and natural readout by 1 meter. The other encoder resolutions and encoder factors are also based on this size.

When a MArker fixture is set up on the relative layer for the MArker attribute, the relative XYZ values are relative to the movement space of the MArker fixture.

When a fixture does not have XYZ values active in the programmer or played back, activating the MArker attribute will set the values for XYZ in the programmer to 0.

When the value of the MArker attribute on the relative layer is 0, the absolute MArker value will be used.



Important:

XYZ needs to be activated for fixtures that want to use MArker fixtures. See the [MArker Fixture topic](#) for more information about this special fixture.

33.1. Activating XYZ for Fixture Types

XYZ needs to be activated for a fixture type to get access to the XYZ attributes.



Important:

Turning On XYZ adds extra virtual parameters for each fixture using the fixture type mode. These extra virtual parameters do not count against the parameter limit but they are shown in the **System Info window**.

XYZ is turned on in the Fixture Type Editor.

1. Press **Menu**.
2. Tap **Patch**.
3. Tap **Fixture Types** in the menu on the left.
4. Select the fixture type that needs XYZ.
5. Tap **Edit** in the menu at the bottom.
6. Select the desired DMX Mode, if there are multiple.
7. Turn On **XYZ** in the menu at the bottom or edit the cell to say "Yes".
8. Close the editor and the patch and save the changes.

Now XYZ can be used for the selected fixture type mode.

XYZ is activated for the selected fixture type mode. So make sure it is activated for the mode that is used.



33.2. MArker Fixture

A fixture type called **MArker** from the manufacturer MA Lighting is part of the fixture library for the MA source.

A MArker is a virtual fixture, that allows operating (for instance, moving around or rotating) objects such as fixtures, actors, or stage props in the 3D stage environment. Learn more about this in the [3D window](#).

It can also function as a target for XYZ-enabled fixtures. Learn how to activate XYZ in the [Activating XYZ for Fixture Types topic](#).

MArker fixtures have a **Target Space** in the Stage. The **Space** defines the volume around the MArker. This space is also the boundary for X, Y, and Z attribute values for the fixtures pointing to the MArker. For instance, if the target space is 2 meters wide on the X-axis, then a fixture pointing to this maker can move inside this 2-meter wide space volume. Setting the X attribute to 0 will move the fixture to the minimum X position in the MArkers target space. The default size for a target space is 200 meters on the X and Z axes (very much bigger than the default stage space).

Moving MArker fixture also has a Movement Space. This defines the space volume in which the MArker fixture can be moved. The default values for this space are also 200 meters in X, Y, and Z.

Read more about the spaces in the [Stage topic](#).

MArker as an Object Mover

For each object or group of objects that shall be moved separately, a MArker fixture needs to be added to the patch in the **Moving** mode.

As soon as a MArker fixture is patched, other patched fixtures can be added as children of the MArker. After applying these changes to the patch, the MArker fixture can be operated like any other fixture in the show.

The MArker fixture provides these attributes:

- X, Y, and Z for moving the MArker and all its children together along the corresponding axes.
- Rot X, Rot Y, and Rot Z rotate the MArker and all its children together around the corresponding axes.

These attributes are located within the features **XYZ** and **Rotation** within the feature group **Position** in the encoder bar.

Select the MArker fixture and turn the encoders for the described attributes in order to see the MArker and its children moving around in the 3D window. The children of a MArker are always moved around relative to the position set up of the MArker itself.

This means that if the fixtures are already at a height of 5 meters and then attached to a MArker that is at a height of 0 meters, then the MArker needs to move below zero to move the fixtures below 5 meters.



Important:

A MArker fixture needs to have a DMX address patched in order to be able to see its changes within the 3D window.

MArker as a Target

The MArker fixtures can be a target for other fixtures that are XYZ enabled. Read about activation in the **Activate XYZ for Fixture Types topic**.

The MArker fixtures need to have the MArker IDType in the patch and a CID number.

The MArker fixtures can be of the Moving or Still mode. Still markers can be positioned using the patch or any other method described in the **Position Fixtures in the 3D Space** topic. Still MArkers are meant to be used for positions in the 3D space that do not move around. Moving MArkers are meant to be used with positions that can move in the 3D space.

The fixtures with XYZ enabled have a MArker attribute. Giving this attribute a number makes the fixture point to the MArker fixture with the matching number.

Moving the MArker, makes the light fixtures pointing at the MArker move the beams to match the MArker fixtures' position.

If the light fixtures pointing to MArker fixture, are be moved using a different MArker fixture while pointing to a MArker, then the light fixture will try to keep the light beam pointing to the MArker.

Moving the MArkers by a Tracking System

The input of PSN trackers can be linked to moving MArkers. This allows moving around the MArker and its children via an external tracking system.

To do so, set up the number of markers needed, and add the desired children to them within the grandMA3 software.

Then set up the PSN system and configure the **PSN input** within the grandMA3 software. Within the PSN menu, the columns called IDType and ID allows entering the MArker IDType and ID of the MArker fixture that shall be linked to the input of each tracker.

The column DMX Priority defines at which level of the grandMA3 playback priorities the PSN data shall be processed. This allows overwriting the input of the PSN system by using a sequence with a higher priority if needed.



Important:

The DMX universes the marker fixtures are patched to which are receiving data from PSN trackers need to be set to the **merge mode Prio**.



34. Macros

What are Macros

Macros are commands stored in a pool object.

The commands can be simple, very complex, and everything in between.

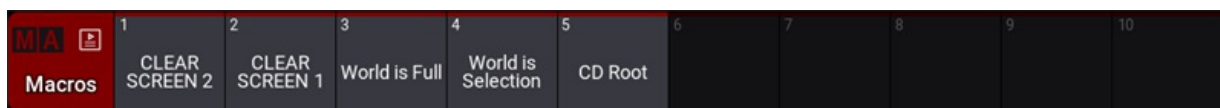
Macros can make programming faster and more convenient. Macros make it possible to do trivial or complex operations with a button push.

Macro Pool

Macros are stored in the **Macro Pool** but can be assigned to physical keys and view buttons. The macro pool is shared between all users in the show file. For example, User A stores a macro on pool object number 10, it will also be available for User B as macro number 10.

It can be created like any other window. [See Pools Window topic](#). You can find it under the **Data Pools** tab.

The Macro Pool could look like this:



Macro pool



Elements in a Macro

A macro consists of one or more rows.

Each row has information about a **Command** the row executes.

Each row has several cells that specify how the row is handled:

The **Wait** cell adds a wait time before moving to the next row in the macro. This wait time is added after the command in the row is executed. It is relevant if there is more than one row in a macro. The first row will execute, wait the specified time and then execute the next row. There are two special wait commands. They are not a time value but a special value:

- **Follow**: This is the same as having a wait time of 0. The next row will be executed as soon as this row has executed the command.
- **Go**: This special value pauses the macro at this row. The macro is paused until it receives a new Go+ command.

The **Enabled** field indicates if the row is enabled. This is a **Yes** or **No** field. If it is **Yes**, it is executed when the macro row is triggered.

The **AddToCmdline** function makes it possible to append the command from the macro line to existing content in the command line. This is a **Yes** or **No** field. If set to **Yes**, it is added to the command line.

The **Execute** field defines if the macro row is automatically executed. **Yes** means that the row is executed (automatic please at the end of the row). **No** will put the command in the command line ready for user interaction.

Importing Macros from Library

A factory library of macros can be imported using the import command.

For more information on how to import macros, read the [Import Macros](#) topic.

Read more about importing in the [Import Keyword](#) topic.

34.1. Create Macros

Macros are created using the GUI editor or the command line.

For a better understanding of macros, see [Macros topic](#).

Create a Macro

To create a macro:

1. To open the Macro pool, open the Add window dialog, tap **Data Pools** and then tap **Macros**.
2. Tap **Edit** an empty macro pool object. This automatically creates a new macro and opens the editor.
3. Tap **Insert New MacroLine** to add a row in the macro.
4. 2 Finger Edit the **Command** cell. The text input pop-up opens.
5. Write a command into the editor. See [Examples](#) for more information.
6. Repeat steps 3 through 5 to add several rows with commands to the macro.

There are several buttons in the editor. This is a short explanation of the specific buttons:

- **Settings**: Enable Settings in the title bar to display the Name, Scribble, Appearance, Note, Lock, and CLI buttons.
- **Test Macro Line**: Tests only the selected line.
- **Test Macro**: Tests the macro.
- **List References**: For more information, see [Info Window](#).



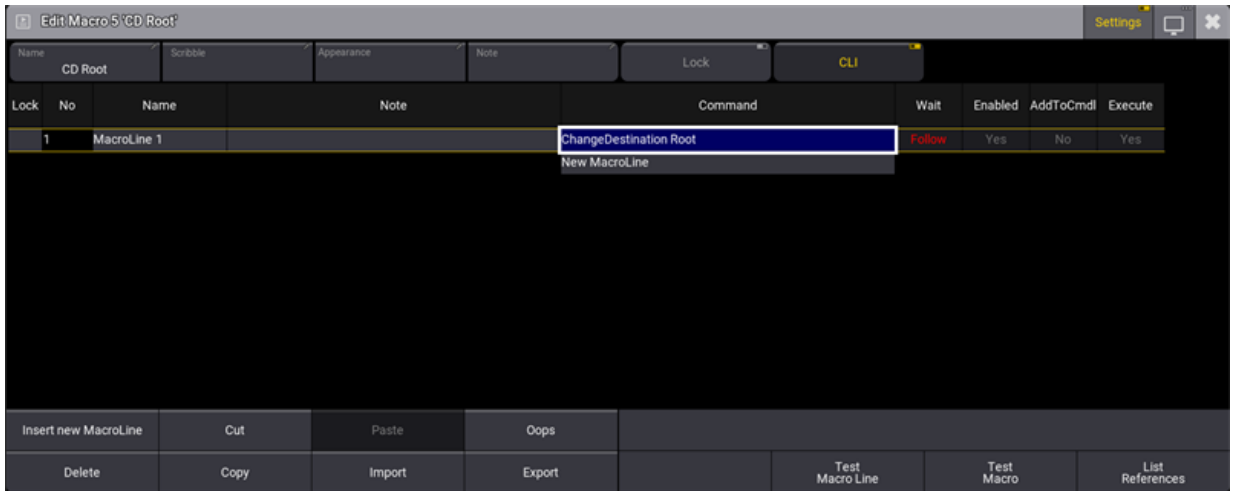
Hint:

The macro editor displays the value in the Wait cell of the last macro line in red to indicate that the value is not being used. For more information, see [Macros](#).

To execute the macro, close the editor and tap the macro.



The editor could look like this (the macro is the "ChangeDestination Root" macro from the library):



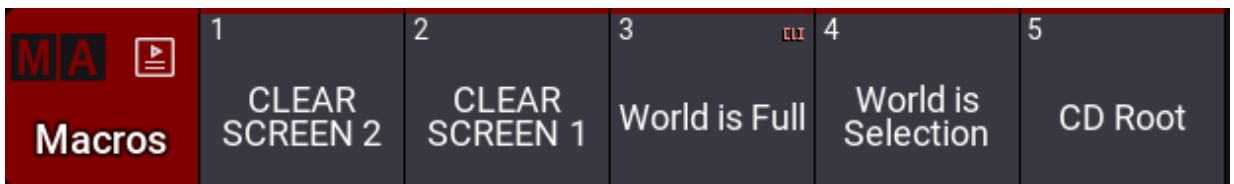
Macro editor

If **CLI** is disabled, tapping the macro pool will still execute it if nothing is entered into the command line. If there are commands entered in the command line, the macro is executed without affecting the command line.

When you press **Edit** and tap macro 5 in the macro pool, it will no longer interact with the edit command. To edit the macro, you would have to type:



Hint: When command-line interaction is disabled, CLI is displayed in red letters on the pool object. Now that commands no longer interact with the command line, use Swipecy Commands.



Macro 3 with CLI deactivated

Create a Macro by Using the Command Line

Macros can be added using the command line. This is relevant when the GUI is unavailable, for instance, when the software is accessed via the command line only, using the terminal window.

To see the result in the command line feedback, use the **List** command at any time during the creative process.

Requirement:

- The **command line feedback** needs to be visible.

This is the creation process:

1. Navigate to the macro part of the software:

```
MA User name[Fixture]>ChangeDestination Macro
```

2. List the existing macros to see empty macro objects:

```
MA User name@ShowData/DataPools/Default/Macros>List
```

3. Locate an available number (one that is not listed).

4. Store a macro with the number:

```
MA User name@ShowData/DataPools/Default/Macros>Store [Available_Number]
```

5. Navigate into the new macro:

```
MA User name@ShowData/DataPools/Default/Macros>ChangeDestination  
[Available_Number]
```

6. Insert a line into the macro:

```
MA User name@ShowData/DataPools/Default/Macros/Macro #>Insert
```

7. Add the command to the text field line number using the **Set** command:

```
MA User name@ShowData/DataPools/Default/Macros/Macro #>Set  
[Line_Number] Property "Command" ["Command Inside Quotations"]
```

8. The wait time can be set in the same manner as the text:

```
MA User name@ShowData/DataPools/Default/Macros/Macro #>Set  
[Line_Number] Property "Wait" ["Wait Time"]
```



Hint:

The wait time can be entered as a number in seconds without quotation marks or as text with quotation marks. The two special times called **Follow** and **Go** can be entered without quotation marks, but the software will add the marks. They are case-sensitive. Typing "go" gives an error. Typing "Go" works.

9. Add more macro lines by repeating steps 6 through 8, but remember to use the correct line number with the Set command.

10. Return to the root of the command line:

```
MA User name@ShowData/DataPools/Default/Macros/Macro  
#>ChangeDestination Root
```

A new macro is created.

Label a Macro

Macro pool objects are labeled like any pool object. Read more in the [Label Pool Objects topic](#).

A unique macro name makes it possible to run the macro by using their names.

34.2. Import Macros

Importing a Macro from the Macro Library

Requirement:

- Have a Macro Pool visible on one of the screens. For more information, see [Create Macros](#).

To import a macro by using the macro pool:

1. Edit an empty pool object. This automatically creates a new macro and opens the Editor.
2. Tap **Import** at the bottom of the editor. This opens the macro library.
3. Use the search function or scroll through the list to find a macro, select a macro and then tap **Import**.

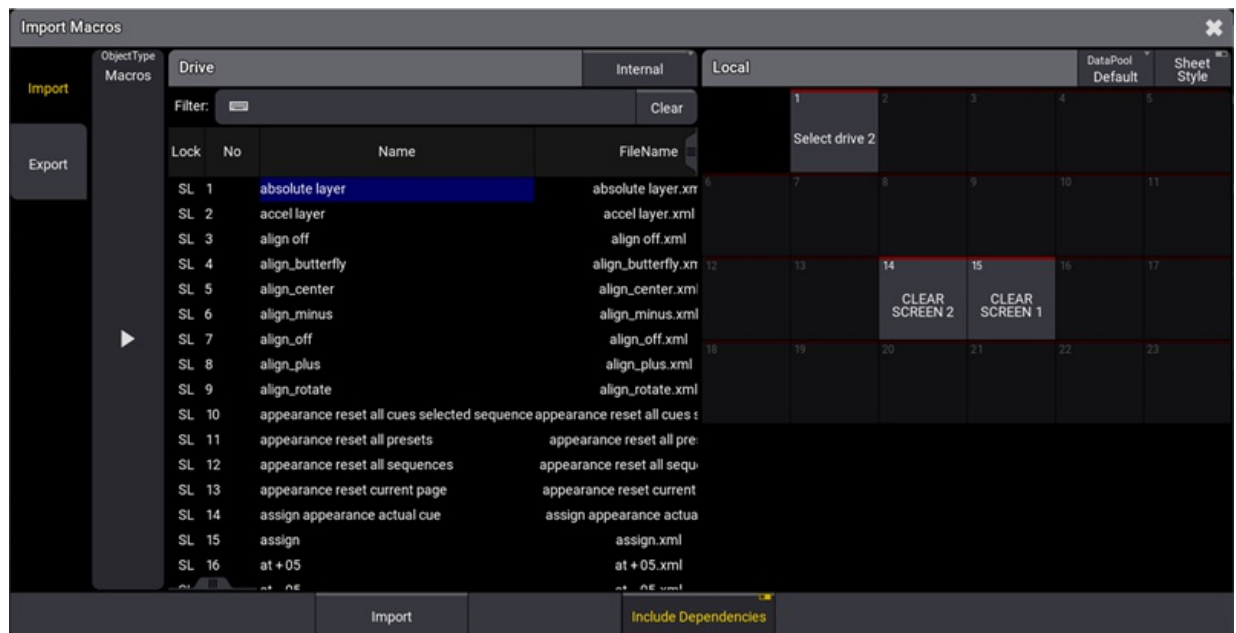
The macro is imported.

Importing Macros by Using the Import/Export Menu

For more information about the Import/Export menu, see [Import / Export menu](#).

Predefined macros can be imported from the library.

1. Press **Menu**.
2. Tap **Import/Export**. The Import menu opens.
3. Tap **Macros** in the Data Pools column. The import macros menu opens:



Import menu

4. To select a macro from the library, tap a macro from the list on the left side of the window.
5. Tap an empty pool object (on the right side of the window) where you want to import the macro.



6. Tap **Import** at the bottom of the window.
7. The macro is imported.



Hint:

If no pool object is selected, the macro will be imported at the next available empty pool object.



34.3. Edit Macros

There are two ways to edit macros: The GUI and the command line.

Edit a Macro by Using the GUI

Requirement:

- An open Macro Pool on one of the screens.

To edit a macro by using the GUI:

1. Tap, hold, and swipe out of the macro in the macro pool that is to be edited.
2. Swipe to the **Edit** swipecy command and release the screen.
The editor opens.
3. Edit the fields that are to be changed.
4. Close the editor when the macro is correct.

Read the [Create Macros](#) topic to learn about adding lines and labeling a macro.

Macro lines can be deleted by selecting the line and then tapping the **Delete** button in the editor.

Edit a Macro by Using the Command Line

Requirement:

- The **command line feedback** is very nice to have visible.

Remember that the [List](#) command displays the content at the current location and can be used anytime.

This is the editing process:

1. Navigate to the macro that is to be edited:

```
MA User name[Fixture]>ChangeDestination Macro ["Macro_Name" or  
Macro_Number]
```

2. Edit the fields using the [Set](#) command. This simply overwrites the current content in the field.
3. When done editing, return to the root location:

```
MA User name@ShowData/DataPools/Default/Macros/Macro  
#>ChangeDestination Root
```

Read the [Create Macros](#) topic to learn about adding lines and labeling a macro.

Delete a Macro Row Using the Command Line

To delete a macro row by using the command line:

1. Navigate to the macro where the row is to be deleted:

```
MA User name[Fixture]>ChangeDestination Macro ["Macro_Name" or  
Macro_Number]
```

2. Use the **Delete** command followed by the row number:

```
MA User name@ShowData/DataPools/Default/Macros/Macro #>Delete  
[Macro_Row_Number_List]
```



Hint:

To delete multiple rows, type a list of numbers.

3. When done deleting rows, return to the root location:

```
MA User name@ShowData/DataPools/Default/Macros/Macro  
#>ChangeDestination Root
```

The macro rows are deleted.



34.4. Assign Macros to Keys and Buttons

The macros can be assigned to executors or view buttons for easy access.

The macro is still in the macro pool. The executor or view button simply runs the macro in the pool. It is, therefore, a requirement to have the macro in the pool before assigning it anywhere.

There are different ways to assign macros to keys or buttons:

Assign Macros by Using the Keys

This is maybe the fastest way to assign the macro when using a console:

1. Press the **Assign** keys.
2. Press the **MA** key while pressing the **X14 | Macro** key.
3. Press the number matching the macro number.
4. Press the key where the macro is to be assigned.

Alternatively, if the macro pool is visible, then it can be used:

1. Press the **Assign** key.
2. Tap the macro in the pool.
3. Press the key where the macro is to be assigned.

Assign Macros by Using the Swipey Commands

Requirement: Have the macro pool visible on the screen.

1. Tap and hold the macro pool object.
2. Swipe out of the pool object -> the swipey commands open.
3. Swipe to the **Assign** button and release the screen.
4. Press the key where the macro is to be assigned.

The macro is assigned to a key.

Assign Macros by Using the Assign Menu for Executors

1. Open the **Assign Menu** for the desired Executor.
2. Tap the **Object** button in the menu on the left.
3. Tap the **Marco** button at the top of the **Assign Menu**.
4. Tap the desired macro in the list in the menu.
5. Close the **Assign Menu**.

The macro is assigned to an executor.



Assign Using the Command Line to Assign to Executors

Make sure the input focus is on the command line and use this syntax:

```
MA User name[Fixture]>Assign Macro [macro_number] At Page [Page_Number].  
[Executor_Number]
```

The syntax above is universal and allows the macro to be assigned to a specific page.

If the macro is to be assigned to the current page, then the following syntax can be used:

```
MA User name[Fixture]>Assign Macro [Macro_Number] At Executor  
[Executor_Number]
```


34.5. Variables

It is possible to create variables that can be used in macros or command line entries.

Variables are containers that can contain different kinds of content. The variable has a user-defined name that is written instead of the variable content. When the command is executed, the variable name is replaced with the variable content.

A variable can contain three different types of data:

- **Integer:** A signed whole number.
- **Double:** A signed fixed-point number with six decimals.
- **Text:** Any text string.

Signed numbers can be negative numbers, for instance, -7.

Variables are not strong-typed but typeless. This means an existing variable can be changed to contain a different type.

The name of a variable can be composed of any character. Variable names are case-sensitive.



Hint:

It is not a requirement, but avoiding spaces in variables is a good idea. Instead, consider using camelCase or PascalCase for variable names.

There are two different variable scopes. A scope defines who and where a variable can be used.

Variables can be scoped for the user, or they can be global variables.

User variables can only be seen and used by the user profile that creates them. Global variables can be seen and used by all users in the session.

SetGlobalVariable and SetUserVariable

Creating a variable is the same whether it is a user or global variable, but we use two different keywords to set one or the other: **SetGlobalVariable** or **SetUserVariable**.

This is the syntax:

SetGlobalVariable ["Variable_Name"] ["Content text with or without spaces"]

SetGlobalVariable ["Variable_Name"] [Integer or Double]

SetUserVariable ["Variable_Name"] ["Content text with or without spaces"]

SetUserVariable ["Variable_Name"] [Integer or Double]



Important:

The quotation marks are very important if the variable name contains spaces. If the name does not contain spaces, then they can be omitted.

The same is true for text strings as the variable content. Variable content must be in quotation marks to ensure the software stores a text string.

Examples

```
MA User name[Fixture]> SetUserVariable MyFavoriteNumber 9
```

```
MA User name[Fixture]> SetUserVariable MyFavoriteText "9"
```

These two commands create two different variables that might seem to have the same content, but the first is a number, and the second is a text. So it is not because of the names but the quotation marks in the second example.

Single and Double Quotation Marks

It might be needed to store a variable with a text string that includes something in quotation marks. This presents an issue with how the software interprets the input and how it is interpreted when the variable is called during runtime.

The variable system supports single and double quote pairs.

This means that text strings can contain quoted text as long as the other quotation marks are used.

Examples

Working with a specifically named group ("Spots Grid") in different contexts is necessary. A variable with "Group" and the name can be stored as a variable:

```
MA User name[Fixture]> SetUserVariable MySpecialGroup "Group 'Spots Grid' "
```

Notice the single quotes around the group name and the double quotes around the variable text string. These two pairs can be reversed for the same result.

Now the variable can be used to address the group:

```
MA User name[Fixture]> $MySpecialGroup At 80
```

Notice the \$ sign before the variable name when using it. In this example we placed the \$ in front of the command so we can recall the variable.



Important:

To recall variables, place the \$ in front of the command.

This is the software's reply:

```
OK: Group "Spots Grid" At 80
```

The dollar sign and the variable name are replaced with the content of the variable.

GetGlobalVariable and GetUserVariable

The variables' content and type can be listed using the **GetGlobalVariable** or **GetUserVariable** keywords.

These keywords need to know what variable to show. The syntax is:

GetGlobalVariable ["Variable_Name"]

GetUserVariable ["Variable_Name"]

These commands show information about the variable name, content type, and content in the **Command Line Feedback window**.

They only show the variables in their scope. This means that the global version only shows global variables, and the user version only shows the user variables.

This assumes that the variable names are known.

The **asterisk (*)** wildcard can show all variables or a filtered list where a part of the name is known.

Examples

The current content of the variable called "MySpecialGroup" can be shown using this command:

```
MA User Name[Fixture]> GetUserVariable MySpecialGroup
```

This is the feedback in the command line window:

```
MySpecialGroup Text "Group 'Spots Grid' "  
OK: GetUserVariable "MySpecialGroup"
```

The following example shows a list of all the global variables where the name contains the word "Our":

```
MA User Name[Fixture]> GetUserVariable *ur*
```

The "O" is replaced with the asterisk to ensure the variables beginning with "Our" are also shown. The asterisk wildcard at the beginning defines that there must be additional letters in front of the letters we specify. In reality, the example will also show variables beginning with "Hour" or, in fact, anything with "ur" somewhere in the name. To specify variables starting with "Our," use the following example:

```
MA User Name[Fixture]> GetUserVariable Our*
```

This example shows all the user variables:

```
MA User Name[Fixture]> GetUserVariable *
```

Use Variables

Variables are used where the content of the variable is needed.



Important:

To recall a variable, use \$ in front of the command.
To define the variable, use \$ at the end of the command.

A \$ (dollar sign) is placed in front of the name of the variable to instruct the software that the following is the name of a variable and that it should replace the sign and the name of the variable with the content of the variable.

If this result is not valid in the command, an error message will be shown in the command line feedback window.

The following variable was used in a previous example:

```
MA User name[Fixture]> $MySpecialGroup At 80
```

The variable's text string ("Group 'Spots Grid' ") replaces the **\$MySpecialGroup**. This works because the variable contains a text string with commands.



Important:

Using quotation marks when using variables is essential and makes a difference.

If the variable is used with quotation marks

```
MA User name[Fixture]> $"MySpecialGroup" At 80
```

It will fail because the content of the variable is explicitly called as a string, and the resulting feedback is:

```
Illegal object: Fixture "Group 'Spots Grid' " At 80
```

The software tries to use the variable content as a text string and does not interpret it as a command.

This might be useful in other cases. The variable name must be in quotation marks when the variable contains a text string that should be used as text. See the example below.

Examples

This example uses a variable that contains a name of a group:

```
MA User name[Fixture]> SetUserVariable MyGroupName "Spots Grid"
```

Now this variable can be used where the text would be used otherwise as:

```
MA User name[Fixture]> Group $"MyGroupName" At 80
```

This command works. If the quotation marks are omitted, this will fail:

```
MA User name[Fixture]> Group $MyGroupName At 80
```

```
Illegal object: Group "Spots" Grid At 80
```

The text string is interpreted as a command.

DeleteGlobalVariable and DeleteUserVariable

Variables can be deleted using the **DeleteGlobalVariable** and **DeleteUserVariable** keywords.

This is the syntax:

DeleteGlobalVariable ["Variable_Name"]

DeleteUserVariable ["Variable_Name"]



Important:

Deleting a variable cannot be oopsed. There is not much else to say about deleting variables. When deleted, they do not exist anymore.

The name is needed to delete a specific variable, but the asterisk (*) wildcard can be used to delete a selection of variables.



Concatenate Variables

Variables can be concatenated by storing them into a new variable.

The desired result of the concatenation is to create a compound text string containing the concatenated variables. The output is always a text string. See below for the mathematical addition of number variables.

This is the syntax:

SetGlobalVariable [MyConcatResult] \$[VariableName]\$[VariableName]

SetUserVariable [MyConcatResult] \$[VariableName]\$[VariableName]

There must be no space between the variables that need to be concatenated or between any extra text that needs to be part of the concatenation.

Text can be added to the variable in quotation marks with no space between the end and start of quotation marks. However, there can be spaces inside the quotation marks. See the example below.

Examples

These examples use three variables:

Variable Name	Content Type	Content
MyInteger	Integer	9
MyKeyword	Text	Group
MyGroupName	Text	Perfect Macro

The goal is to concatenate these into a command string that can be used to store a group.

It can be done with one command but is split up to look at different functions here.

First, the keyword is combined with the integer into a new variable:

```
MA User name[Fixture]> SetUserVariable MyCommandString "$MyKeyword"  
"$MyInteger"
```

There are a lot of quotation marks in this command. Setting the variable names in quotation marks ensures the variable content is interpreted as text. This also means that the integer variable is converted to a text string.

Furthermore, there is a " " between the two variables to add a space between them.

Show results using this command:

```
MA User name[Fixture]> GetUserVariable MyCommandString
```

Next add the macro name to the string:



```
MA User name[Fixture]> SetUserVariable MyCommandString  
$"MyCommandString" "$"MyGroupName"" "
```

Again, it can be hard to see what is happening. The two names of the variable are in quotation marks. Between the two are a space and a single quotation mark in double quotation marks. After the last variable name, there is another single quotation mark inside the double quotation marks.

The command looks like this without the double quotations: **SetUserVariable MyCommandString \$MyCommandString '\$MyGroupName'**

The existing content of MyCommandString is added to MyCommandString, plus the content of MyGroupName in single quotations.

The result is that MyCommandString is **Group 9 'Perfect Macro'** as a text string.

Now some fixtures can be selected, and this command can be executed:

```
MA User name[Fixture]> Store $MyCommandString
```

And then, a group is stored at group 9 with the name. Quotation marks must not be used in the command above.

Variable Addition

Concatenating variables with integer and/or double types does a mathematical addition instead of an actual concatenation. Here there is no use of quotation marks.

The result needs to be stored in a variable.

Examples

This example uses three variables:

Variable Name	Content Type	Content
MyInteger	Integer	9
MySignedInteger	Integer	-2
MyDouble	Double	6.500000

The first two variables can be added to each other with this command:

```
MA User name[Fixture]> SetUserVar MyResult $MyInteger$MySignedInteger
```

The result can be seen by looking at the command line feedback. It shows that the variable is stored with a value of 7.

The double can be added with an integer. The result will be a double to avoid data loss.



```
MA User name[Fixture]> SetUserVar MyResult $MyInteger$MyDouble
```

This stores the value of 15.500000 in MyResult.

Type Conversion

Type conversion has been happening in the examples above.

An integer or double can be converted to text by putting the variable name in quotation marks. They will be converted when the command is executed.

Variables containing text that can be converted to an integer can be converted by setting the variable without quotation marks.

An integer can be converted to a double by adding it to a double with the value of **0.000000**.

A double cannot be converted to an integer.

Examples

These examples use the following variables:

Variable Name:	Content Type:	Content:
MyInteger	Integer	9
MyEmptyDouble	Double	0.000000
MyText	Text	Hello

This example converts the integer to text:

```
MA User name[Fixture]> SetUserVariable MyInteger $"MyInteger"
```

Converting this back to an integer can be done with the following command:

```
MA User name[Fixture]> SetUserVariable MyInteger $MyInteger
```

Converting an integer to a double can be done using this command:

```
MA User name[Fixture]> SetUserVariable MyResult $MyInteger$MyEmptyDouble
```


34.6. Example Macros

This topic has some example macros. They are meant as inspiration for other macros. In addition, they show some of the possibilities with macros.

Change User

This macro opens the login window where you can enter a new username and password.

Lock	No	Name	Command	Wait	Enabled	AddToCmdl	Execute
	1	MacroLine 1	LogIn	Follow	Yes	No	Yes

World Is Selection

This predefined macro will create a temporary world for the selected fixtures.

Lock	No	Name	Command	Wait	Enable	AddToCmdl	Execute
	1	MacroLine 1	Delete World 999 /NoConfirmation	Follow	Yes	No	Yes
	2	MacroLine 2	Store World 999	Follow	Yes	No	Yes
	3	MacroLine 3	World 999	Follow	Yes	No	Yes

Block Sequence

This macro will prompt you for a sequence ID and a cue number to block.

Lock	No	Name	Command	Wait	Enable	AddToCmdl	Execute
	1	MacroLine 1	Block Sequence (Sequence #) Cue (Cue #)	Follow	Yes	No	Yes



Hint:

If no cue number is entered, the entire sequence will be blocked.



35. Agenda

The agenda allows you to schedule objects, e.g., sequences, macros, or plugins, to be executed by the console based on the calendar. Commands can also be defined, for example, to shut down the system at a certain time.

You can also schedule events that will repeat every minute, every day, every week, every month, and/or every year, e.g., Saint-Patrick day cue every year.

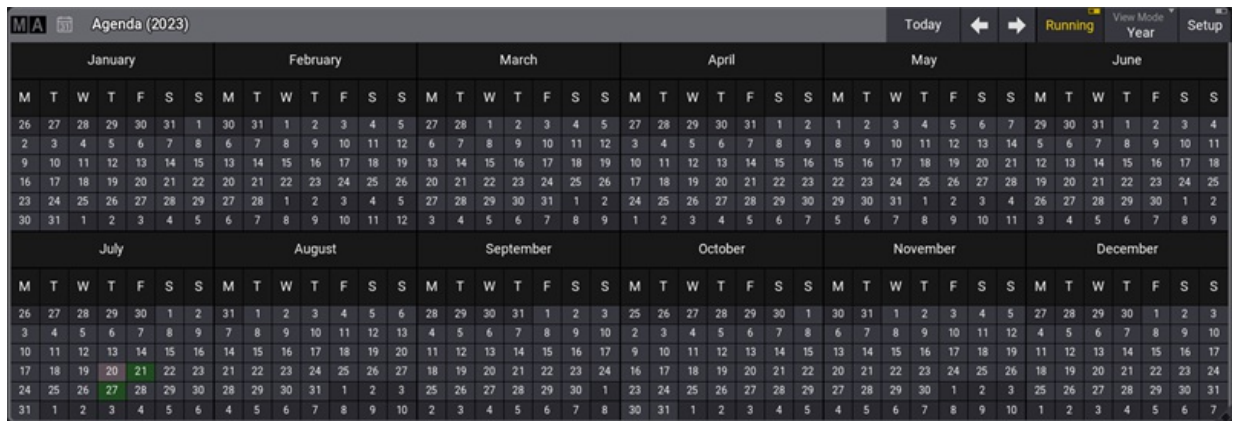
To learn about the Agenda keyword, see [Agenda](#)



35.1. View Modes

The agenda window can display the data using five different view modes.

Open an agenda window, see **Add Windows**. From the add windows dialog, tap **Tools**, and then tap **Agenda Viewer**.



Agenda window

To change the view, tap **View Mode** in the title bar.

- **Sheet:** Displays all agenda entries in a spreadsheet format.

Hint:
To sort the agenda entries in the sheet view by name, right-click or tap and hold a column header.

- **Year:** Displays a calendar view of the year. The current date has a light gray background. Days with at least one enabled event are indicated by a green background. Days that have all events disabled are indicated by a bright red background.
- **Month:** Displays all agenda entries of the selected month.
- **Week:** Displays all agenda entries of the selected week.
- **Day:** This mode is divided into two sections. It displays a smaller monthly overview on the left and entries for the selected day on the right.

Hint:


- Disabled events are displayed with red font color.
- Repeated events are displayed with on the right side. This applies to the month, week, and day view modes.

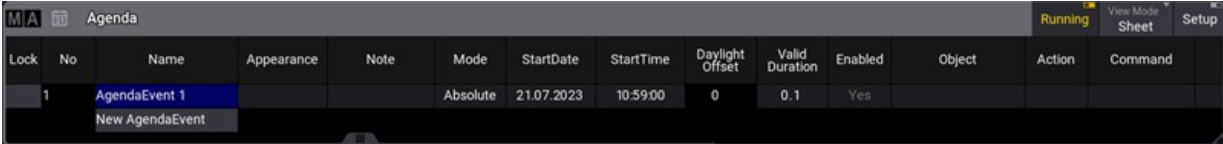
35.2. Create an Agenda Entry

Create an Agenda Entry Using the Sheet View Mode

1. Tap and hold **View Mode** in the title bar, then slide your finger in the list and select **Sheet**.
2. To create a new agenda event, right-click or tap and hold **New AgendaEvent**.

A new entry is created with the system date and time.

**Hint:**
It is also possible to create an entry from the **Sheet** layout mode by pressing **Edit**, then tapping **New AgendaEvent**.

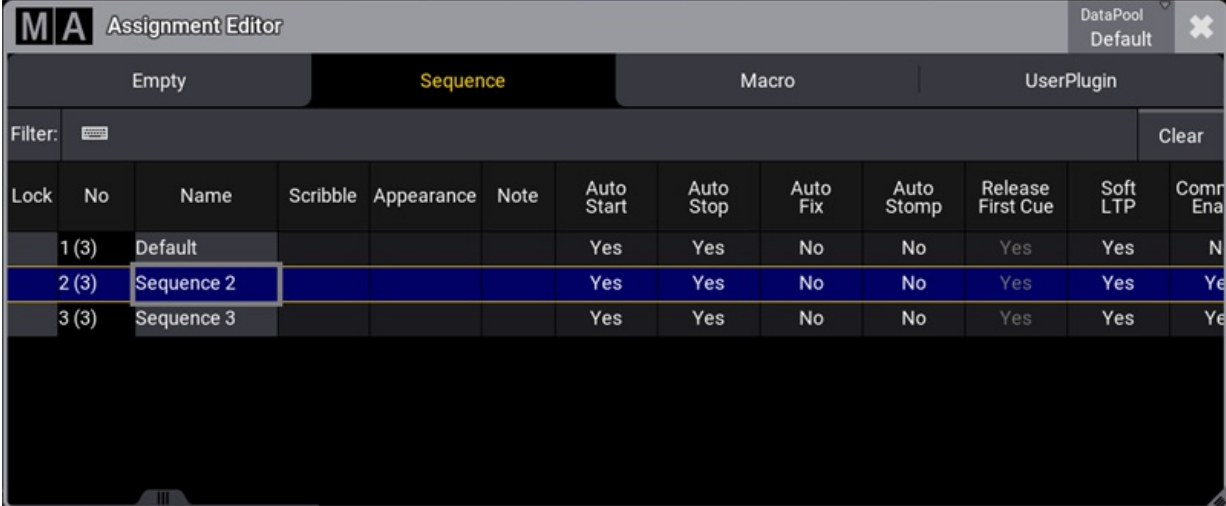


Lock	No	Name	Appearance	Note	Mode	StartDate	StartTime	Daylight Offset	Valid Duration	Enabled	Object	Action	Command
	1	AgendaEvent 1			Absolute	21.07.2023	10:59:00	0	0.1	Yes			
		New AgendaEvent											

Agenda Viewer

The following is a list of the properties to be defined by the user:

- **Name:** Enter the name of the event that will be displayed in all layout modes.
- **Appearance:** Assign an appearance to define how the entry looks in the calendar views.
- **Note:** For more information, see **Note**.
- **Mode:** When set to **Absolute**, the agenda uses the entered start date and time. Twilight times like **Dawn**, **Sunrise**, **Sunset**, and **Dusk** can also be used. See **Date and time** to learn how to configure the grandMA3 software to calculate the correct twilight time for your location.
- **StartDate:** Tap and hold a cell to open the **Edit StartDate** pop-up, then edit the day, month, and year. Use **Today** in the title bar to quickly enter today's date.
- **StartTime:** Tap and hold a cell to open the **Edit StartTime** pop-up, then edit the hours, minutes, and seconds. Use **Now** in the title bar to quickly enter the current time.
- **Daylight Offset:** When a twilight time is set in the mode column, it defines an offset for an event. For example, if you want an event to start 15 minutes before dawn. See **Date and time** to learn about twilight times.
- **Valid Duration:** Enter a duration value to allow the backcasting of events if the console is not switched on or the agenda is disabled during a scheduled entry.
- **Enabled:** To enable or disable an agenda entry. Disabled entries are displayed with a red color font.
- **Object:** Tap and hold a cell to open the Assignment Editor pop-up, select a **Plugin**, **Macro**, or **Sequence** to be executed.



MA Assignment Editor													DataPool Default	✕
Empty		Sequence				Macro				UserPlugin				
Lock	No	Name	Scribble	Appearance	Note	Auto Start	Auto Stop	Auto Fix	Auto Stomp	Release First Cue	Soft LTP	Comr Ena		
	1 (3)	Default				Yes	Yes	No	No	Yes	Yes	N		
	2 (3)	Sequence 2				Yes	Yes	No	No	Yes	Yes	Ye		
	3 (3)	Sequence 3				Yes	Yes	No	No	Yes	Yes	Ye		

Assignment Editor

- **Action:** Tap and hold a cell, then select an action to use when the defined object is executed.
- **Command:** Define a command here to be executed instead of defining an object to be triggered. E.g., Go+ Executor 101.
- **Repeat:** Defines if an entry is repeated and, if so, how often. Tap and hold a cell to open the **Edit Repeat of Agenda** pop-up.

They are two main settings to define repetition of an agenda entry, **Schedule** and **Iterations**.

Schedule Settings

Edit Repeat of Agenda 1 'AgendaEvent 1' Reset Pattern ✕

Schedule Iterations

Startdate: 14.06.2021 Starttime: 00:29:00

Enddate: 14.06.2021 Reset Endtime


<input checked="" type="checkbox"/> Mon	<input checked="" type="checkbox"/> Tue	<input checked="" type="checkbox"/> Wed	<input checked="" type="checkbox"/> Thu	<input checked="" type="checkbox"/> Fri	<input checked="" type="checkbox"/> Sat	<input checked="" type="checkbox"/> Sun
<input checked="" type="checkbox"/> 1st Week	<input checked="" type="checkbox"/> 2nd Week	<input checked="" type="checkbox"/> 3rd Week	<input checked="" type="checkbox"/> 4th Week	<input checked="" type="checkbox"/> 5th Week	<input checked="" type="checkbox"/> 6th Week	
<input checked="" type="checkbox"/> Jan	<input checked="" type="checkbox"/> Feb	<input checked="" type="checkbox"/> Mar	<input checked="" type="checkbox"/> Apr	<input checked="" type="checkbox"/> May	<input checked="" type="checkbox"/> Jun	
<input checked="" type="checkbox"/> Jul	<input checked="" type="checkbox"/> Aug	<input checked="" type="checkbox"/> Sep	<input checked="" type="checkbox"/> Oct	<input checked="" type="checkbox"/> Nov	<input checked="" type="checkbox"/> Dec	

Total Repeat: 0 Repeated days: 0

The start date and time are linked to the agenda event and therefore are identical. At least a different end date or end time must be set to get repetitions.

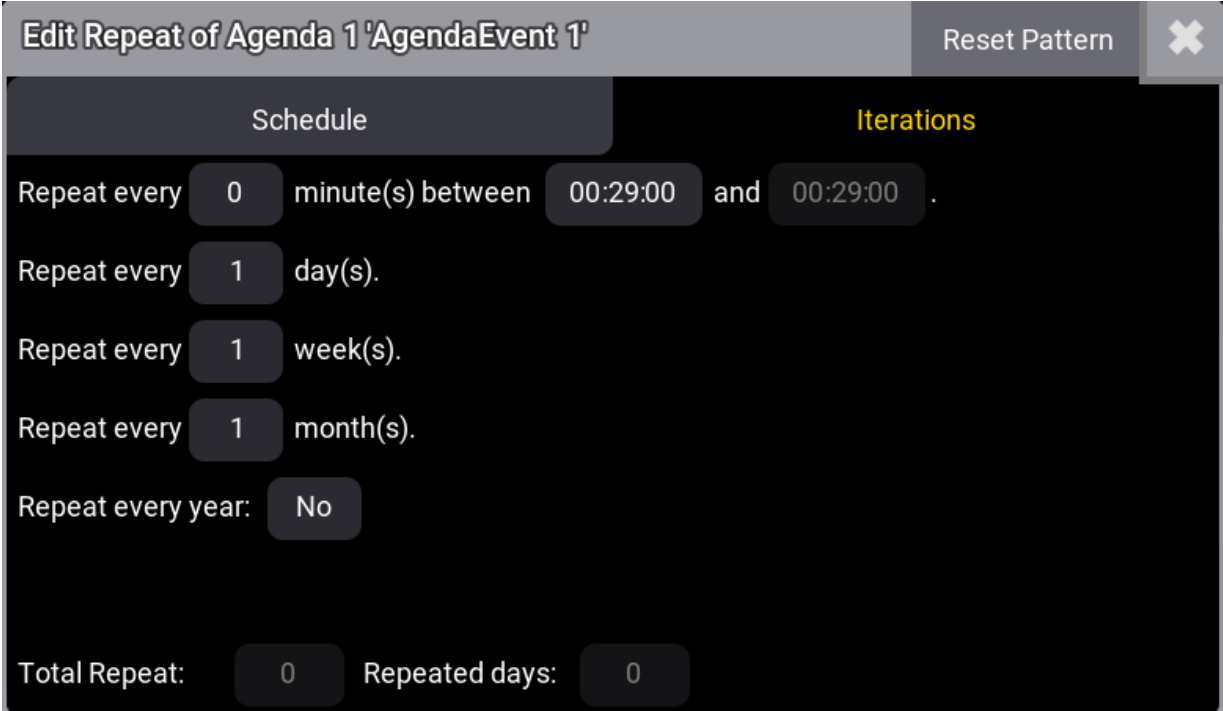
- Tap **Reset Pattern** in the title bar to reset the repeat settings you made previously.
- Tap **Reset Endtime** to reset the end date back to the agenda event date.

Tap a day, week, or month cell to enable or disable it. This will define at which days, weeks, or months the event will be repeated.



Hint:
The week settings are the weeks within a full month and not the weeks from the start date onwards.

Iterations Settings



Edit Repeat of Agenda 1 'AgendaEvent 1' Reset Pattern ✕

Schedule Iterations

Repeat every minute(s) between and .

Repeat every day(s).

Repeat every week(s).

Repeat every month(s).

Repeat every year:

Total Repeat: Repeated days:

Iterations settings allow defining repeat per minute, days, weeks, and months.

These settings are counting from the start date onwards.

To repeat an event every year again, set **Repeat every year** to yes.

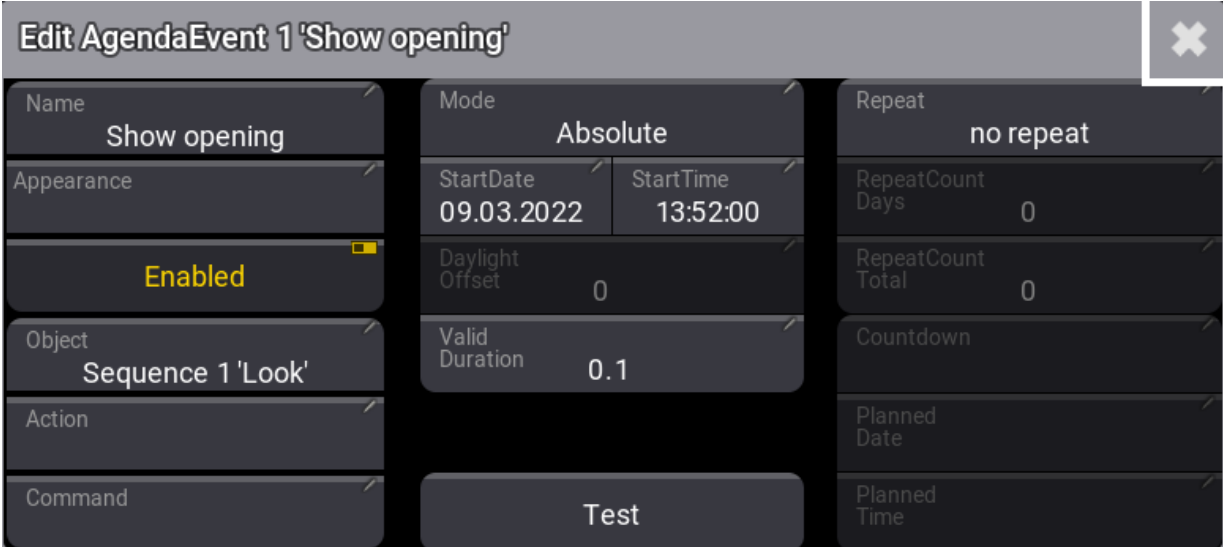
The following is a description of the last five columns of the agenda window, when **ViewMode** is set to **Sheet**, in the title bar. (These columns can not be edited).

- **Countdown:** Displays the remaining days or time until the next launch of an event.
- **Planned Date:** Displays the next date when the event will be launched.
- **Planned Time:** Displays the next time when the event will be launched,
- **Repeat Count Days** and **Repeat Count Total:** These two columns are similar to the edit repeat pop-up. This is a quick way to verify the repeat pattern.

Tap **Delete Old** in the title bar to erase all preceded events, including the valid duration.

35.3. Edit an Agenda Entry

To edit an agenda entry from the **Month**, **Week**, and **Day** layout mode, press **Edit**, then tap the agenda entry you wish to edit. You can also tap and hold or right-click an agenda entry. The **Edit Agenda Event** pop-up opens.



Edit AgendaEvent 1 'Show opening'		
Name	Mode	Repeat
Show opening	Absolute	no repeat
Appearance	StartDate	StartTime
Enabled	09.03.2022	13:52:00
Object	Daylight Offset	RepeatCount Days
Sequence 1 'Look'	0	0
Action	Valid Duration	RepeatCount Total
Command	0.1	0
		Countdown
		Planned Date
	Test	Planned Time

To learn about the properties that can be defined here, see [Create an agenda entry](#).



Hint:

Agenda events can also be edited using the toolbar. see, [Agenda toolbar](#).



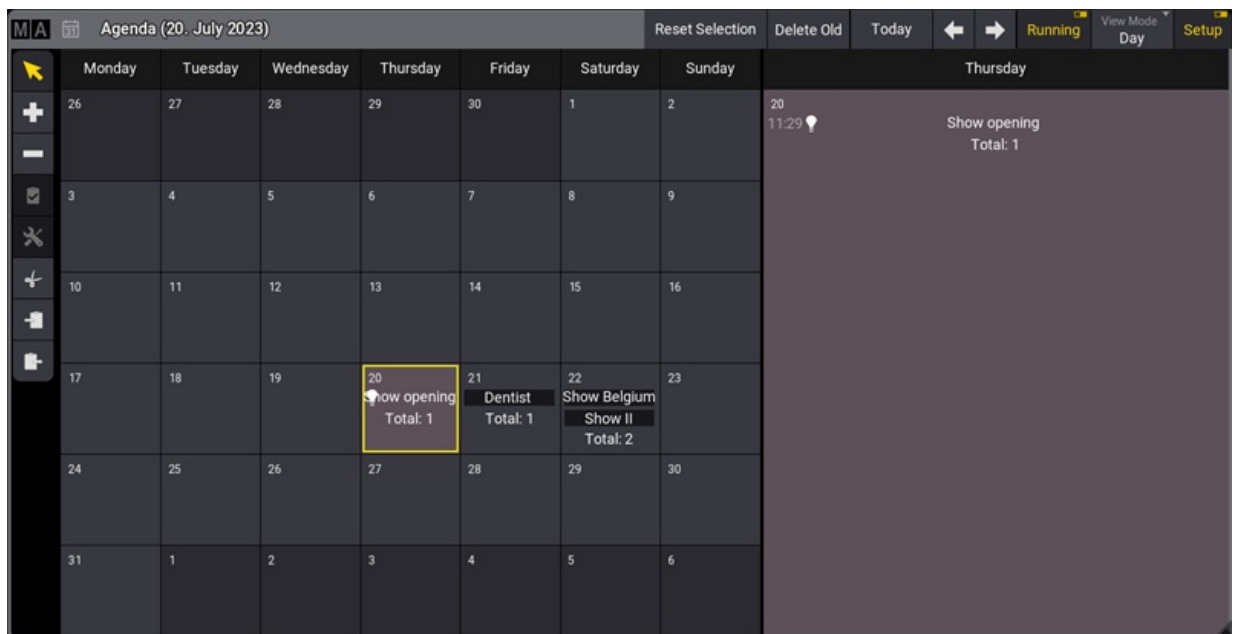
Hint:

Events can be deleted from any view mode by pressing **Delete**, then tapping the event you wish to delete.


35.4. Agenda Toolbar

Create or edit an agenda entry using the toolbar within the 5 available view modes; **Sheet**, **Year**, **Month**, **Week**, and **Day**. For more information, see [Agenda Modes](#).


To enable the toolbar, tap **Setup** in the title bar.





Agenda window with view mode day



To select an event, tap  in the toolbar, then tap the event you wish to select.



To create an event, tap  in the toolbar, then tap a day in the agenda.



To delete an event, tap  in the toolbar, then tap the event you wish to delete.

Select an event, then tap  in the toolbar to execute it immediately. This will verify if the set object or command will be executed correctly.

Select the event you wish to edit, then tap  in the toolbar. This opens the Edit Agenda pop-up. See [Edit an agenda entry](#) for more information.

To cut an agenda entry, tap  in the toolbar, then tap the entry you wish to cut. Notice that  (the paste tool) is selected in the toolbar and awaits for the user to tap a day where to paste the cut entry.

To copy an agenda entry, tap  in the toolbar, then tap the day you wish to copy. Notice that  (the paste tool) is selected in the toolbar and awaits for the user to tap a day where to paste the copied entry.

Use  and  in the title bar to change the day, week, month, or year.



Hint:

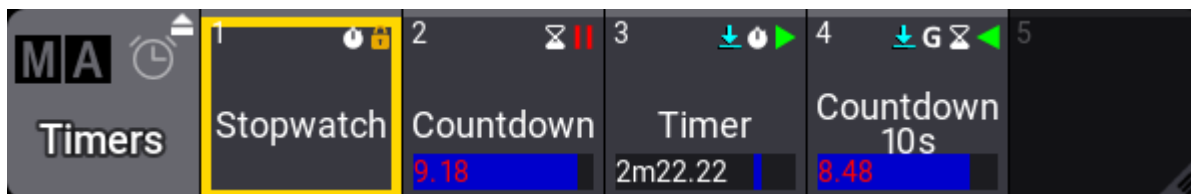
The day, week, month, or year is displayed in brackets in the title bar according to the selected **View Mode**.

Tapping **Today** in the title bar will set the agenda to today's date.

36. Timers

The Timer pool can be used to manage stopwatches and countdowns.

The pool can be created like any other pool window using the [Add Window pop-up](#).



Timers pool with two different timers

The Timers pool has a default action indicated by an icon in the title field's upper right corner. The default action is **Toggle** (⏻). The alternative action is **Select** (S). The default action is applied when a pool object is tapped without a valid keyword in the command line. Learn more about Pool Actions in the [Window Settings topic](#).

There are two different types of timers: Stopwatch and Countdown. Each of these two types has their own topics. Follow the links below or in the menu on the left.

The running state of the timer is not saved with the showfile. The time on the timer will be saved, and the timer will be on the saved time in a paused state when the showfile is reloaded.

Each object in the Timers pool can be one or the other.

The first pool object is a Stopwatch, locked from being edited, although It can be used.

Timers can be controlled using the pool objects or the [Timer keyword](#).

Active timers can be seen in the pool with the time and a progress bar at the bottom of the pool object. Active timers can also be seen in the [Running Playbacks window](#).

Running timers have a green play icon pointing right for stopwatches (▶) and left for countdowns (◀) in the upper right corner of the pool object. A paused timer has the pause icon (⏸) in the corner. Stopped timers do not have a playback icon.

Timers can be assigned to executors. Learn more in the [Assign Object to an Executor topic](#).

Playback Actions

A timer can be started with a **Go+** command or toggled On if it is Off or paused. It will start to count up or down in real time.

A timer can be paused in its running. This keeps the current time on the timer. Restarting the timer with a Go+ command can either restart the timer or let it continue. This can be set in the setting for the timer.

The **Top** command will reset the time. If the timer is running, it will continue to run. If the timer is paused, the time will be reset, and the timer will be stopped.

Stopping a timer resets the time to zero for stopwatches and to the set time value for countdowns.

Using the Clock to See the Time

The **Clock** can be used to show and control the timers. The **Clock Source** should be set to **Timer**. Then the **Timer** can be used to select the desired timer object.



Two clocks showing a stopwatch and a countdown

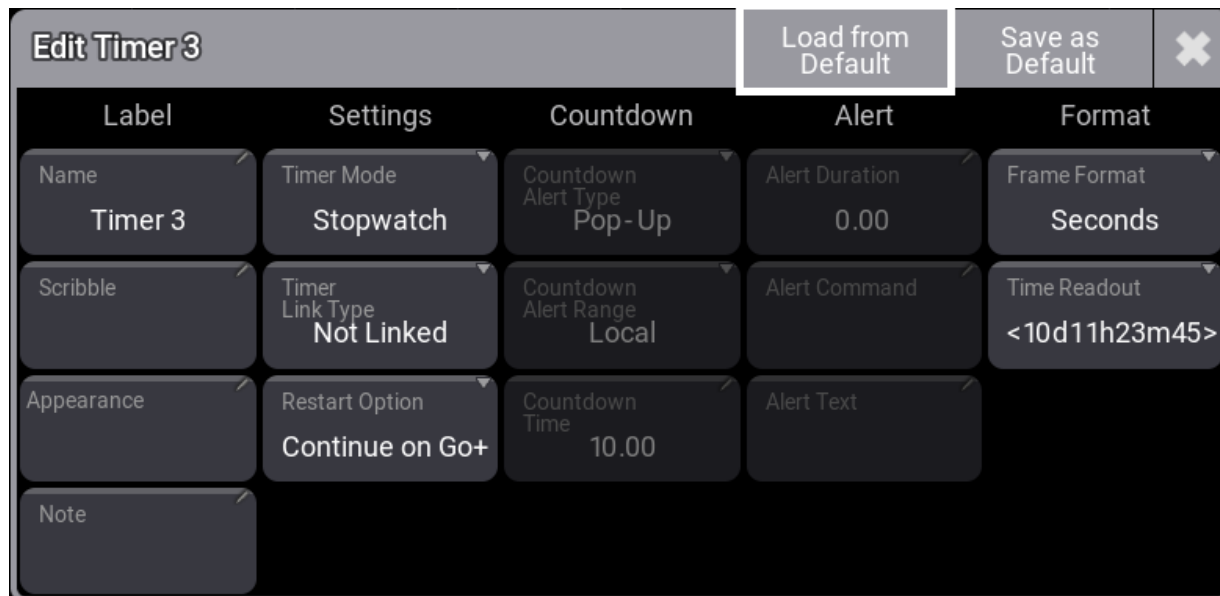
The controls in the title bar adjust to match the selected timer.

Stopwatches have numbers in white color. Countdowns have numbers in red color with a countdown bar at the bottom of the timer. These are the default colors.

Create and Edit a New Timer

A new timer can be created by editing an empty pool object.

Editing a timer pool object opens the editor:



Timer editor

There are some common settings for the two types.

Label

- **Name:**
This is the name of the timer object.
- **Scribble:**
This is used to assign a scribble to the pool object.
- **Appearance:**
This is used to assign an appearance to the pool object.
- **Note:**
This is used to add a note to the pool object.

Settings

- **Timer Mode:**
This is used to select one of the two modes: Stopwatch or Countdown.



- **Timer Link Type:**

These are the options:

- **Not Linked:**

Nothing happens when starting a sequence playback.

- **Selected Go+:**

Executing Go+ for the selected sequence also starts the Timer.

- **Selected Toggle:**

Executing Go+ for the selected sequence toggles the Timer.

- **LastGo Go+:**

Executing Go+ for any sequence also starts the Timer.

- **LastGo Toggle:**

Executing Go+ for any sequence toggles the Timer.

- **Restart Option:**

This has two possible options:

- **Continue on Go+:**

When executing Go+ on a stopped or paused timer, it will continue to run the time.

- **Reset on Go+:**

A stopped or paused timer will be reset and start from its default time when executing Go+ for the timer.

Format

There are two settings in the format column:

- **Frame Format**

- **Time Readout**

These are available for each timer object. They are described in the [User Settings topic](#).



36.1. Stopwatch

Stopwatches are the default timer. Learn more in the [Timers topic](#).

They count time up from 0 and do not have a practical upper limit.

The stopwatch timer has a stopwatch icon (🕒) in the upper right corner of the pool object.

A stopwatch will run for several days if it is not stopped or paused.

Stopwatches do not have any special settings.

36.2. Countdown

Stopwatches are the default mode for timers. The mode can be changed to the Countdown mode. Learn more in the [Timers topic](#).

Countdowns count time down from a set value; when zero is reached, a trigger can fire an alert pop-up and a command.

The countdown timer has an hourglass icon (🕒) in the upper right corner of the pool object.

Countdown Settings

The countdowns have more settings besides the common settings described in the Timer topic.

Countdown

- **Countdown Alert Type:**

The alert type defines what should happen when the countdown reaches zero. There are four options:

- **None:**
Nothing happens when the timer runs out, except it stops.
- **Pop-up:**
A pop-up appears with the text set in the **Alert Text** setting (see below). The pop-up appears for as long as the **Alert Duration** is set (see below).
- **Command:**
The timer executes the command set in the Alert Command setting (see below).
- **Command & Popup:**
The pop-up appears, and the command is triggered.

- **Countdown Alert Range:**

The countdown alert range defines where the pop-up appears.

- **Local:**
The alert pop-up appears on the stations with users logged in using the same user profile as the one who started the countdown.
- **All Stations:**
The alert pop-up appears on all stations in the session. Countdowns with this setting as a small G icon (G) in the upper right corner of the pool object to indicate that it is a global countdown.
The pop-up has a confirmation button to close the alert (before the duration closes it). **Confirm** only confirms for the station where it is tapped. **Confirm All Stations** confirms the alert on all stations.

- **Countdown Time:**

Sets the duration time of the countdown. The maximum possible countdown time is limited to 14 days. The calculator sets the duration time in seconds, minutes, hours, and days.

Alert



- **Alert Duration:**
Defines how long the pop-up should be displayed when the Countdown Timer reaches zero. An alert duration of zero seconds means the pop-up stays until confirmed. The countdown alert type needs to include "pop-up" for this setting to be active.
- **Alert Command:**
When the countdown reaches zero, the command is executed on the GlobalMaster/IdleMaster station or the Standalone station. The **countdown alert type** needs to include "command" for this setting to be active.
- **Alert Text:**
The defined text of this setting is displayed in the alert pop-up when the countdown reaches zero. The countdown alert type needs to include "pop-up" for this setting to be active.

37. Timecode

Timecode is a time signal used to let off the recordings of a playback or trigger cues.

This time signal can either be generated using an external source, (SMPTE), (MIDI), or (Art-Net) timecode or it can be generated internally using the console.

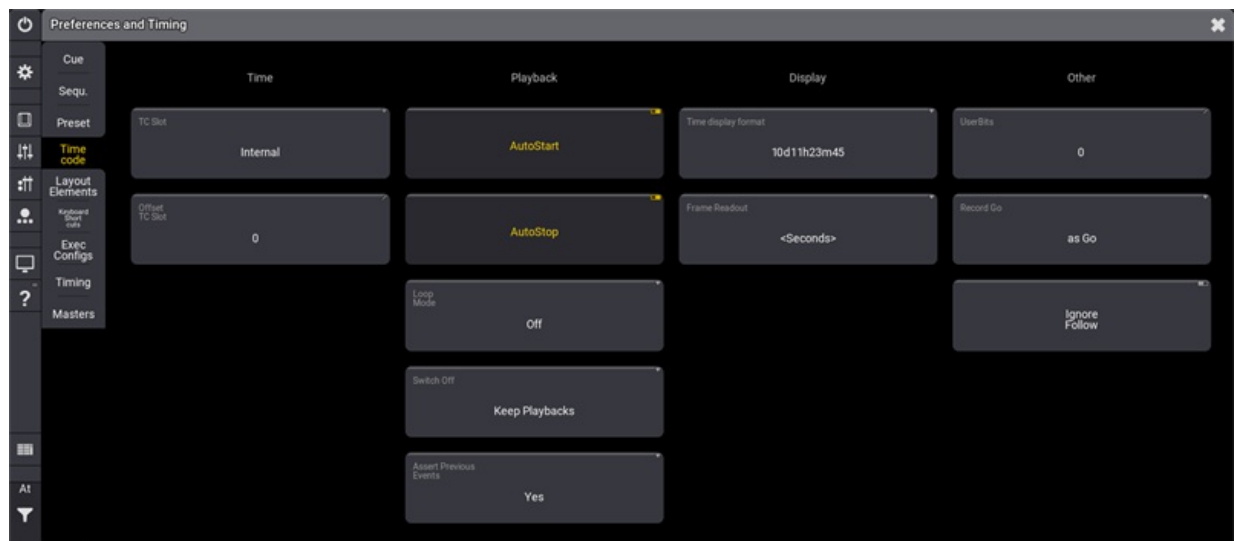
The grandMA3 can receive up to 8 external timecode signals at the same time.

These timecode signals are located in **Timecode Slots**.

To edit the default timecode settings:

1. Press **Menu** .
The menu pop-up opens.
2. Tap **Preferences and Timing** .
The Preferences and Timing window opens.
3. On the left side of the window, tap **Timecode** .

Here are the default timecode settings.



Preferences and Timing window

For more information about the different settings, see **timecode settings**.

37.1. What are Timecode Slots



Important:

The timecode slots can be edited. The timecode slots cannot be added, copied, deleted, or moved.

A timecode slot is an integrated interface that interprets a timecode signal in hours (h), minutes (m), seconds (s), and frames (f).

The grandMA3 can receive up to eight different external timecode signals at the same time.

Timecode slots are located in the Timecode Slots pool. For information on Pools, see [Pool windows](#).

For information on the timecode slots pool settings, see [Window settings](#).

Clock

To use the clock to display the timecode slot, open the clock window. For more information, see [Add windows](#).

For general information on the clock, see [System – Clock](#).

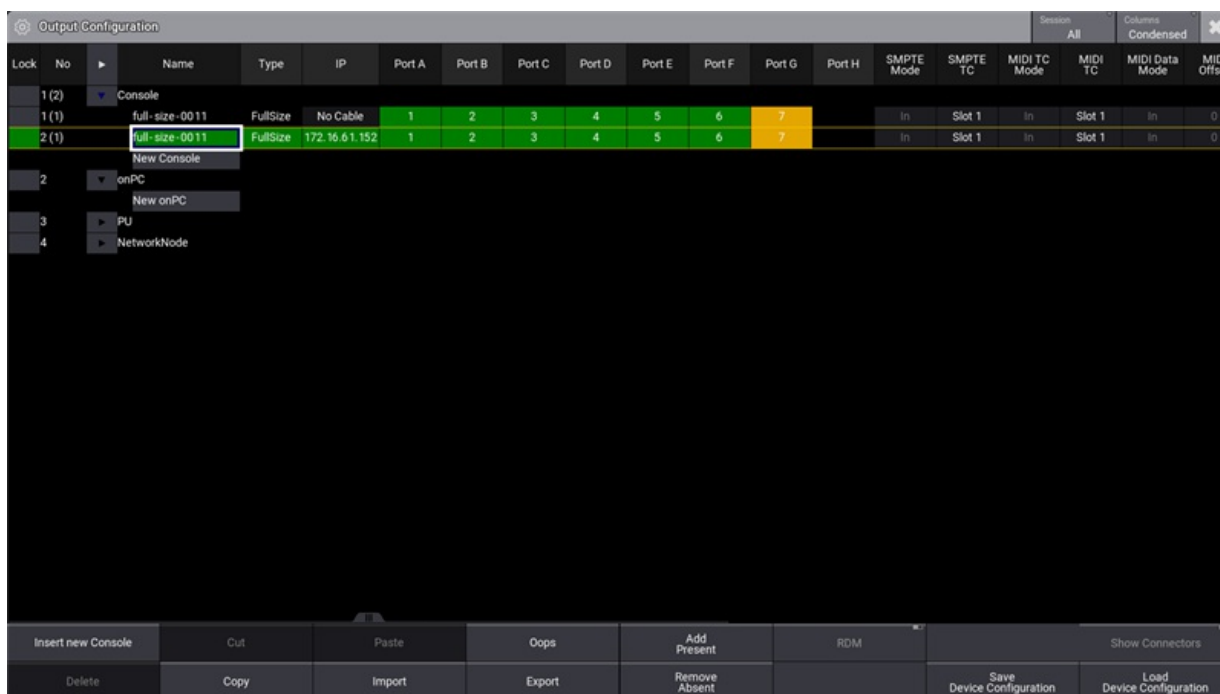
The clock window buttons can be used to control the timecode slot signal sent to external receptors—media servers, for example.

Output Configuration of Timecode Slots

To receive the external timecode signal or send out an internally generated timecode, connect the physical connector of the grandMA3 hardware to the timecode slots in the software.

1. To open the Output Configuration, press **Menu**, and tap **Connector Configuration**.
2. Tap and hold a cell in the SMPTE Mode or MIDI Mode column and set it to **In** or **Out**.
3. Tap and hold a cell in the SMPTE TC or MIDI TC column to open the drop-down menu.
4. Slide your finger to the desired timecode slot.

The physical output connectors are assigned to a timecode slot.



Output Configuration window



Important:

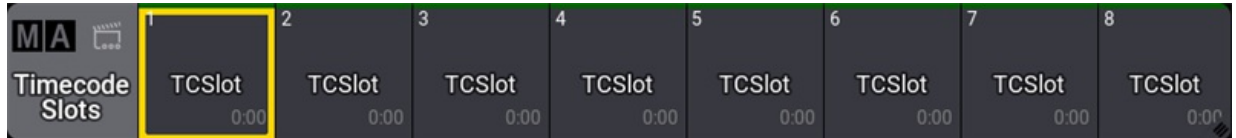
Timecode slot settings are not part of the show file. They will not be transmitted within the show file to another console.



Important:

ArtTimeCode can be sent to timecode slots via Art-Net. For more information on the configuration, see [Ethernet DMX – Art-Net Menu](#).

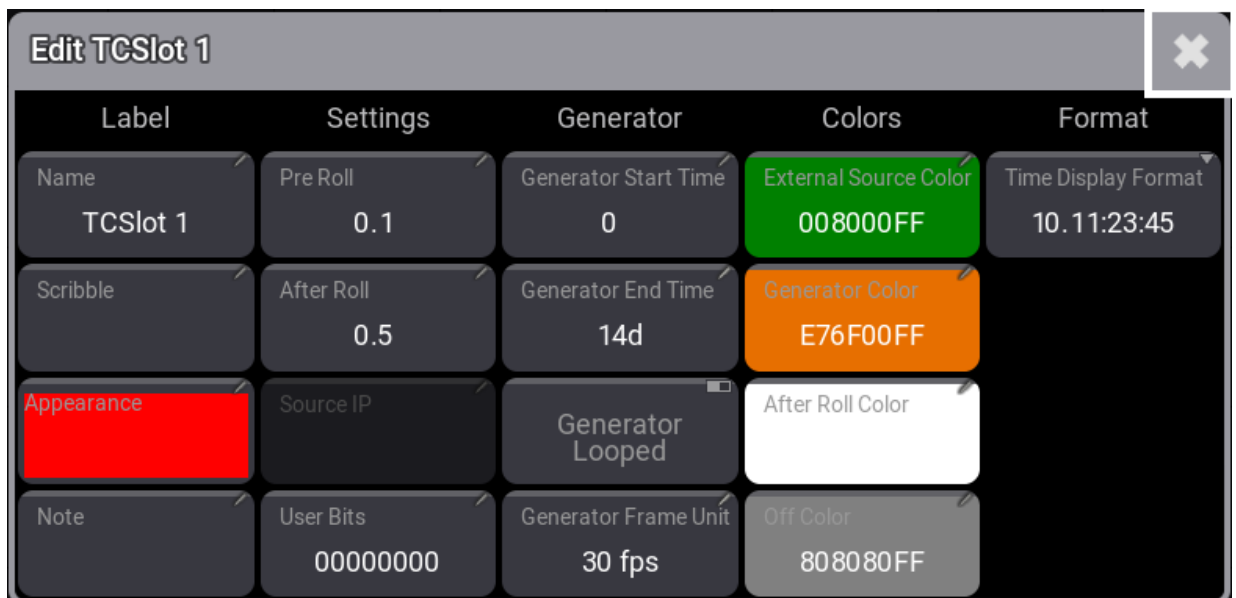
Edit Timecode Slot



Timecode Slots pool

- To edit a timecode slot, use the swipecy command or press **Edit** and tap a timecode slot.

The pop-up Edit Timecode Slot opens.



Edit Timecode Slots pop-up

Name:

Enter the name of a timecode slot.

Scribble:

Apply a scribble to the timecode slot pool element. For more information, see [scribbles](#).

Appearance:

Sets the appearance of the object in the timecode slot pool. This appearance will also apply to the background of the cursor button in the title bar of the timecode window. This is a quick visual way of knowing what timecode slot a timecode show is getting the signal from. For more information, see [Timecode settings](#).

Note: For more information, see [Notes](#).

Pre Roll:

States how long a signal is coming in until the console uses it.

After Roll:

If the external signal stops, the timecode runs internally in the After Roll. For example, if 10 seconds were set, the time runs for another 10 seconds, even though the external signal ceased.



Hint:

- Pre Roll and After Roll are used for external signals.
- Generator times are used for internal signals.

Source IP:

Displays the IP address of the grandMA3 device, which receives the running timecode signal.

User Bits:

Besides the 32 Bit for 8-digit timecode time, timecode executes 32 User Bits (8-digit) per frame.

Use User Bits to mark a timecode signal. For example, use User Bit 1 for light and User Bit 2 for pyro. This is not available when using an internal timecode source.



Important:

Do not change the user bits, unless you are told to do so by the timecode supplier. When the user bits are changed, and the timecode supplier does not know, the timecode signal will not be received anymore.

Generator Start Time:

Sets the start time for the internal timecode generator.

Generator End Time:

Sets the end time for the internal timecode generator.

Generator Looped:

Repeats the internal timecode indefinitely.

Generator Frame Unit:

Sets at which frame rate the signal is being emitted.

External Source Color:

Sets the color of the clock for the external timecode source.

Generator Color:

Sets the color of the clock for the internal timecode source.

After Roll Color:

Sets the timecode slot color to indicate that the external signal was interrupted and that the timecode slot is in After Roll.

Off Color:

Sets the color of the clock when no signal is coming in.



Time Display format:

Select the Time display format, including days or only in hours and two different formats.

37.2. Timecode Settings

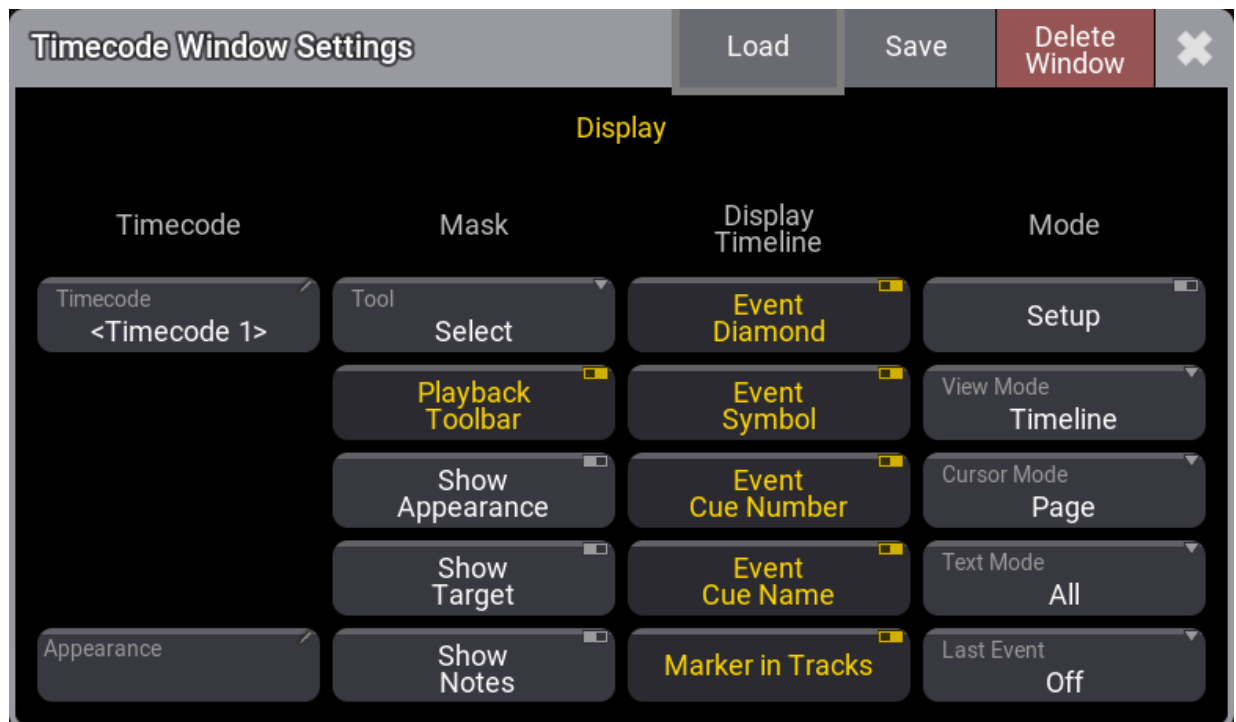
There are two different settings in a timecode:

- Window settings
- Timecode show settings

Open a timecode window. For more information, see [Add windows](#). You can find the **Timecode Viewer** under the **Tools** tab.

Open the timecode window settings by tapping the **MA** logo in the title bar.

Timecode Window Settings



Timecode Window Settings

Timecode:

Select the timecode show you would like to use here.

The timecode show is initially stored in the timecode pool. For more information, see [Record a Timecode Show](#).

Appearance:

Assign an appearance to the timecode window. For more information, see [Appearances](#).



Tool:

Toggles between the tool buttons in the toolbar of the timecode view. The toolbar is visible only in setup mode. For more information, see [Time ranges and events](#).

Playback Toolbar:

Enables the toolbar to navigate the timeline and playback or record the selected timecode show.

Show Appearance:

Toggle on to display the appearance column. For more information, see [Toggle View Mode](#).

Show Target:

Toggle to display the target column. For more information, see [Toggle View Mode](#).

Note:

Toggle on to display the note column. For more information, see [Notes](#).

Event Diamond, Event Symbol, Event Cue Number, and Event Cue Name:

Toggle on to display event diamonds, event symbols, event cue numbers, and event cue names on the timeline.

Marker in Tracks:

Toggle on to display markers on all tracks of a track group. When toggled off, the markers will only be shown in the track group section.

Setup:

Toggle to display the setup toolbar.

View Mode:

Tap and hold to open the Select ViewMode drop-down list, then change the timecode window view to **Text**, **Timeline**, or **Both**. For more information, see [Toggle View Mode](#).

Text Mode:

When View Mode is set to Both or Text, tap and hold to open the Select TextMode drop-down list, then change the text mode to **All**, **Tracks**, **Selected**, or **Markers** to filter the text display.

Cursor Mode:

When the stopwatch is enabled in the title bar, the cursor stays visible when the timecode runs.

● **Center:**

When an external signal is coming in or generated by the console, the cursor (green timeline) will move until it reaches the middle of the visible area of the timeline. The timeline will start moving to keep the cursor centered on the display.

● **Page:**

When an external signal is coming in or generated by the console, the cursor (green timeline) will move until it reaches the end of the visible part of the timeline, then a new section of the timeline will be displayed. The cursor will move from the beginning to the end of that new section, repeating until the timecode is stopped.



Hint:

The same applies when moving the cursor using the dual encoders. For more information, see [Edit a Timecode Show](#).

Last Event:

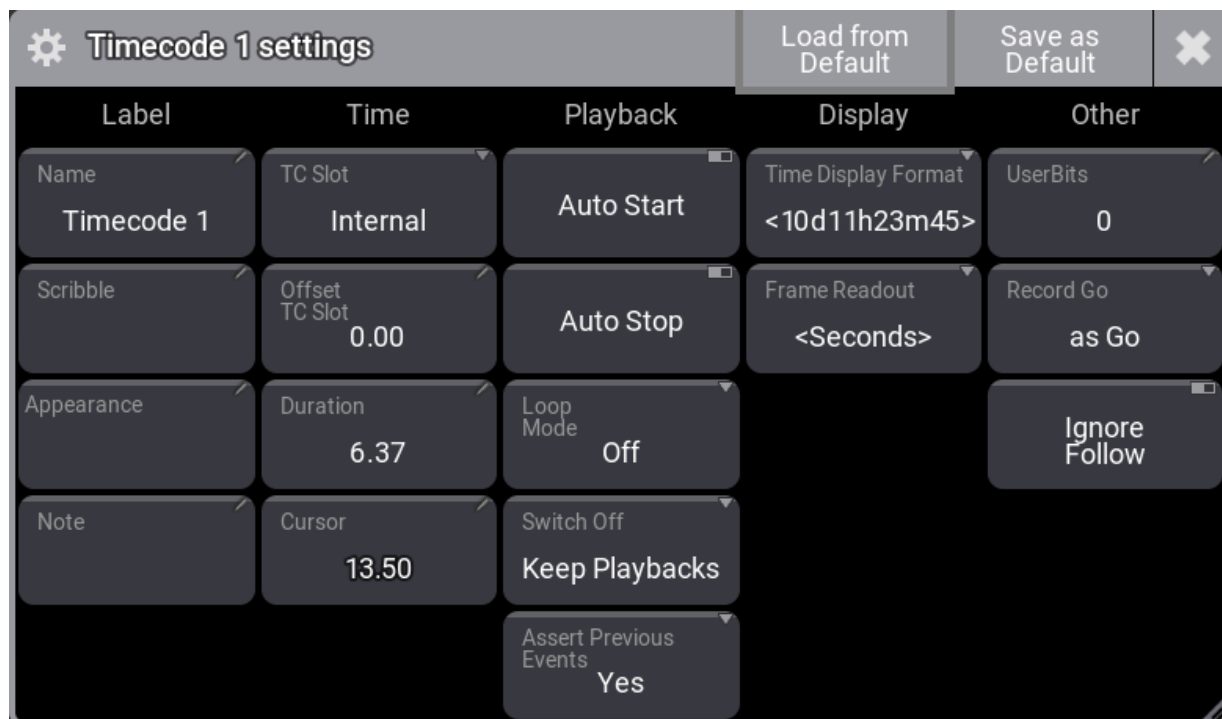
Defines if the last played-back event in the timecode show is selected.

- **Off:**
The last played-back event will not be selected.
- **Track:**
The last event played back in the selected track will be selected.
- **All:**
Selects the last played-back event, no matter which track.

Timecode Show Settings

To open the timecode show settings, tap **Settings** in the title bar.

Timecode show settings pop-up opens:



Timecode settings pop-up

Name:

Tap to open the Edit Name pop-up, then enter a name for this timecode.



Scribble:

Tap to assign a scribble. For more information, see [Scribbles](#).

Appearance:

Tap to assign an appearance. For more information, see [Appearances](#).

Note:

Tap to enter a note. For more information, see [Notes](#).

TC Slot:

Tap and hold to open the Select TC Slot pop-up list, then select **<Internal>**, **<Selected>**, or **TC Slot 1** thru **TC Slot 8**. To learn more about Timecode Slots, read [What Are Timecode Slots](#).

Offset TC Slot:

Tap to open the calculator, then set the offset for the timecode slot.

Duration:

Tap to open the calculator, then set the duration of the timecode show recording.

Cursor:

Displays the cursor at its current position on the timeline. Tap to open the calculator and enter a new position.



Hint:

The selected timecode slot appearance defines the cursor button background color. To learn more about timecode slots, read [What are timecode slots](#).

Auto Start:

Starts the timecode show automatically when an external signal is received.

Auto Stop:

Automatically stops the timecode show when an external signal is stopped.

Loop Mode:

When the timecode is internally generated:

- **Loop:**
The timecode show will repeat playing from the beginning to the end. Press the stop button in the playback bar to stop the timecode show.
- **Pause:**
The timecode show will pause at the end. Press the start button in the playback bar to restart the timecode show from the beginning.
- **Off:**
The timecode show will not loop or pause.

Switch Off:

Toggles between Playbacks Off and Keep Playback.



- **Playbacks Off:**

Switches off all playbacks that were started by the timecode show.

- **Keep Playback:**

Does not switch off the playbacks that were started by the timecode show.

Assert Previous Events:

When set to Yes, events preceding the play head will be asserted.

Time Display Format:

Select the time display format you want to use **<Default>**, **10d11h23m45**, **25 1h23m45**, **10.11:23:45**, or **251:23:45**

Frame Readout:

Select the frame readout between **<Default>**, **Seconds**, **24 fps**, **25 fps**, **30 fps**, or **60 fps**.

UserBits:

Besides the 32 Bit for 8-digit timecode time, timecode executes 32 **User Bits** (8-digit) per frame. Use User Bits to mark a timecode signal. For example, use User Bit 1 for light and User Bit 2 for pyro. This is not available when using an internal timecode source. Do not change this unless you are expressly told to do so by the timecode supplier.

Record Go:

When set to **as Go**, events will be recorded with a Go+ action and a Goto+ action when set to **as Goto (Status)**.

Ignore Follow:

When toggled on, cues with a **Follow** trigger type, see [Cues and sequences - Sequences sheet](#), will not be recorded on the timecode show. For more information, see [Record a timecode show](#):

Timecode events for follow cues already recorded before enabling this setting will be ignored, and the follow cues will be played back with the sequence timing.

37.3. Track Groups

Timecode shows are arranged in a hierarchical tree structure where **Track Groups** can contain one or more **Tracks**.

Markers can be assigned to the selected track group. Time ranges are assigned to tracks. For more information, see [Time ranges and events](#).

To select a track group, press **Select** and tap the track group you wish to select. The selected track group has a yellow text label.

Track groups can be expanded and collapsed by tapping the arrow on the left-hand side of the track group name.

Track groups can also be selected via the command line. For example,

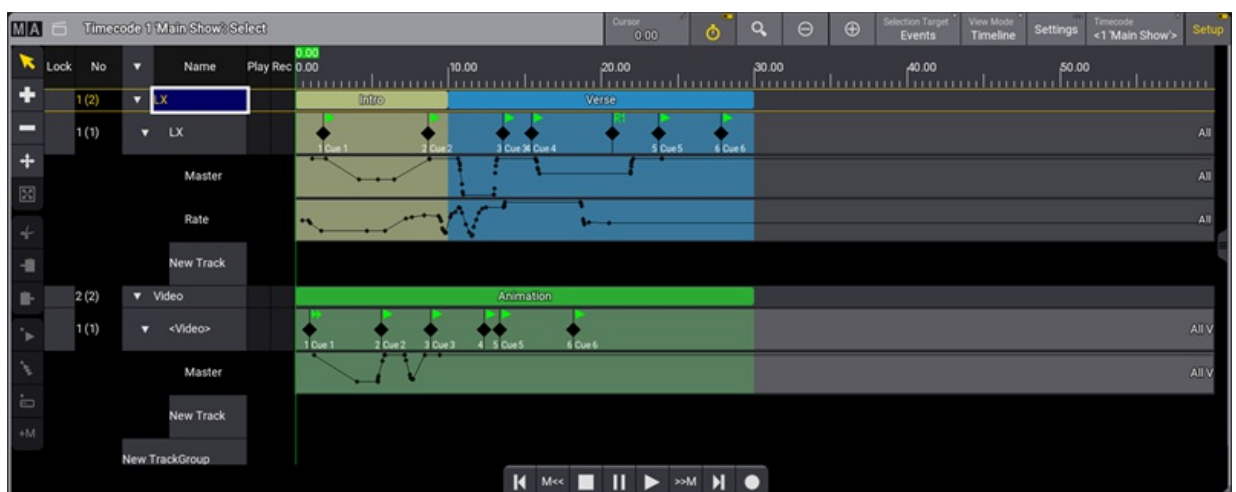
Select track group 2 of timecode 1:



Tracks can be added to track groups by tapping and holding a cell in the **Target** column. For more information, see [Toggle view mode](#).

When pressing the corresponding target executor key, a track is added to the selected track group. For more information, see [Record a timecode show](#).

Fader movements are recorded on separated sub-tracks within a track. For example, **Master**, **Rate**, and **Speed**. See the image below.



Recorded fader movements.



37.4. Time Ranges and Events

Time Ranges

Time ranges and markers are defined regions drawn on the timeline. They can be labeled, and they can get appearances assigned. A marker is created on the timeline section of a track group and will be applied downward to all the tracks in the track group. A time range is created on the timeline section of a track, and it is track specific.

Time ranges and markers are tools to better navigate and visualize the timecode show. They do not affect timecode playback or recording.

When a track is added, a new time range is created for that track, and the duration is set to the total length of the timecode show. To set the duration of the timecode show, see [Timecode settings](#).

Events

Events are tags positioned on the timeline that can get **Tokens** assigned.

Tokens can be assigned from the text display of the timecode view or with the event encoder in the encoder bar. For more information, see [Toggle view mode](#).

Select Token		×
Empty	LearnSpeed	Top
>>>	Load	
<<<	On	
Black	Off	
Call	Pause	
DoubleSpeed	Rate 1	
Flash	Select	
Go+	SelectFixtures	
Go-	Speed1	
Goto	Swap	
HalfSpeed	Temp	
Kill	Toggle	

The assigned tag is executed when the cursor (green timeline) reaches an event.

Edit Time Ranges and Events

Events and time ranges can be selected, added, deleted, moved, resized, and more using the toolbar.

To enable the timecode toolbar, tap **Setup** in the title bar.

The toolbar is displayed on the left side of the timecode view, and the encoder bar is switched to the timecode encoder bar.



Hint:


Press and hold any tool icon in the toolbar to display a short description in the command line.



Hint:

Some toolbar buttons will be enabled depending on the commands you will execute.


You must first define the target to edit Time Ranges or events using the toolbar. To do so, tap and hold **Selection Target** in the title bar. This will open the **Selection Target Type** pop-up, then select **Time Ranges** or **Events**.

Tap  then tap a time range or an event in the timeline. The selected event is displayed in yellow, and the selected time range as a yellow frame.



Hint:

- It is possible to select multiple events or time ranges using the lasso selection method.
- A random selection can be performed by holding down the keyboard's **Ctrl** key and tapping the desired events or time ranges.

To add an event to a track, tap  then tap in the timeline where you want to add the event.

To add a time range, tap  then do one of the following:

1. Tap in the timeline where you want to add the time range. A one-frame Marker is created when tapping in a TrackGroup section of the timeline. When tapping in a **Track** section of the timeline, a **Time Range** is created from this point to the end of the timecode.
2. Tap in the timeline where you want the time range to start, hold and drag your finger until you have reached the desired length, then release your finger.







Hint:


Events and time ranges can be adjusted in detail and accuracy using the dual encoders. For more information, see [Edit a timecode show](#).




Delete an event or a time range by tapping , then tap the event or time range you want to delete.

To move an event or a time range, tap , then tap-hold and drag your selection along the timeline.

Tap , then tap-hold, and drag your finger anywhere on a time range to resize it. Drag to the left to reduce the time range duration or to the right to increase it. This tool is enabled when **Time Ranges** is the selected target.

Events and time ranges can be  (cut),  (copied), and  (pasted).


1. Tap , then tap the event or time range you want to cut or copy.
2. Position the cursor in the timeline where you wish to paste your selection.


3. Select the track where you want to paste your selection.
4. Tap , or  then, tap . The selection is cut or copied to the desired track at the cursor position.





Hint:


Events can only be pasted to tracks. Time ranges can be cut or copied from tracks to track groups.



Tap  to add an event at the cursor position. This tool is handy to add events while recording.

1. Select **Events** as the target.
2. Select the track you want to add events to.
3. An event is added each time you tap .

Tap  to add multiple events at the cursor position.


1. From the title bar, select **Events** as the target.
2. Select the track you want to add events to.
3. Position the cursor in the timeline where you want to add multiple events.
4. Tap , the **Add multiple events** pop-up opens.




Add multiple events	
Number	 6
Time interval	 4s
Cancel	Add

5. Enter the number of events you want to add and specify the time interval between them.

The events are added.


To add a marker at the cursor position, tap .

1. From the title bar, select **TimeRanges** as the target.
2. Select the track group where you want to add the marker.
3. Position the cursor in the timeline.
4. Tap , the virtual keyboard opens.
5. Enter a name for the new marker and press **Please** or tap **Enter**.
6. The new Marker is added.

To quickly add a marker at the cursor position, tap . This tool is handy to add markers while recording.

1. From the title bar, select **TimeRanges** as the target.
2. Select the track group where you want to add the marker.



3. A marker is added every time you tap  .

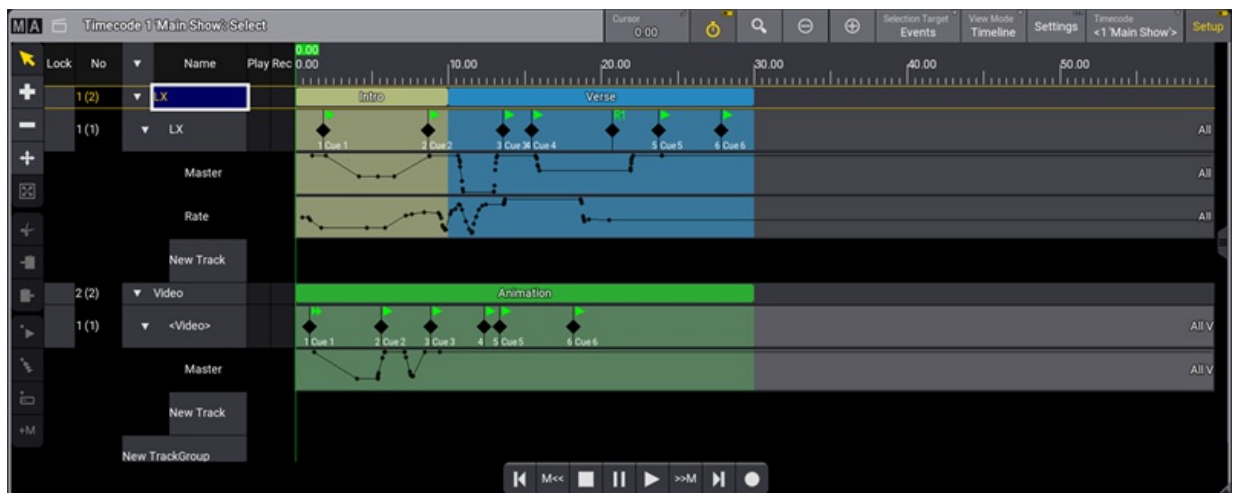
37.5. Toggle View Mode

There are three different view modes of a timecode show.

Tap and hold **View Mode** in the title bar, then select a mode.

Timeline Mode

Displays the timecode show as a timeline.



Timecode view with View Mode set to Timeline

The timecode window's left side is always visible, whether the view mode is set to **Text**, **Timeline**, or **Both**.

Here is a description of that section.

Lock:

Tap and hold a cell in this column and set it to yes to disable editing.

No:

Displays the number of the track. The number of children for the selected track is displayed in brackets.

Name:

Displays the name of the track groups and tracks. Tap and hold a cell in this column to open the keyboard, then set a new name.

Play:

Tap and hold a cell in this column to disable the playback of a track. When disabled, events on that track will not be executed when the timecode runs.

Rec:

Tap and hold a cell in this column to disable the recording of a track. When disabled, events can not be recorded on that track.



Hint:

The next two columns need to be toggled on to be displayed. For more info, see [Timecode settings](#).

Appearance:

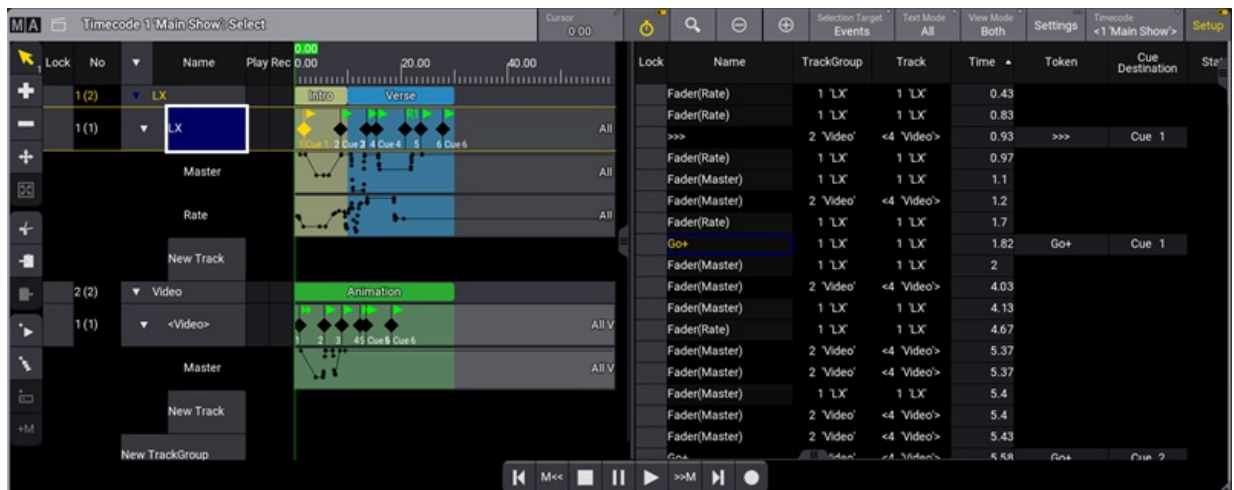
Tap and hold a cell in this column. This will open the select appearance pop-up list, then select an appearance. For more information, see [Appearances](#).

Target:

Tap and hold a cell in this column to open the assignment editor pop-up menu. Tap **Empty**, **Sequence**, **Timecode**, or **TCSlot**, then select a target.

Both Mode

Divides the display in two, displaying the timecode show as a timeline on the left side and text on the right.



Timecode view with View Mode set to Both

Tapping **Selection Target** in the title bar will set the focus to **Events** or **TimeRanges**.

Tap **Text Mode** in the title bar to filter the information.:

- **All:**

- When the Selection Target is set to **TimeRanges**, this mode will display information for all markers and time ranges in the timecode show.
- When the Selection Target is set to **Events**, this mode will display information for all events in the timecode show.



- **Tracks:**
 - When the Selection Target is set to **TimeRanges**, this mode will display information for markers and time ranges of the selected track.
 - When the Selection Target is set to **Events**, this mode will display information for events of the selected track.
 - **Selected:**
 - When the Selection Target is set to **TimeRanges**, this mode will display information for the selected markers and time ranges.
 - When the Selection Target is set to **Events**, this mode will display information for the selected events.
 - **Markers:**

This mode will display markers information for the selected track group. The Selection Target will automatically switch to **TimeRanges**.
-

This is a description of the column headers in the text display.

Lock:

Set to yes, to disable editing.

Name:

- When the Selection Target is set to **Events**, it will display the Selection Target's handle used to record the event. For more information, see [Executor Configurations](#).
- When the Selection Target is set to **TimeRanges**, it will display markers, tracks, and track group information.

TrackGroup:

Displays the name of the track groups.

Track:

Displays the name of the tracks.

Time:

Displays in seconds the location of an event on the timeline. Tap and hold a cell in this column to open the calculator, then set a new value.

Token:

Displays the action executed when the event is played back. Tap and hold a cell in this column to open the select token pop-up, then select a token.

Cue Destination:

Displays the cue that will be executed when the event is played back. Tap and hold a cell in this column, then select a new cue.

Status:

Displays the status of an event recorded when an executor was set to flash, black, or temp. For more information, see [Executor Configurations](#).



Execute Command:

Defines whether cue commands are executed when an event is played back. Tap and hold a cell in this column and set it to Yes or No. The default is Yes. For more information, see [Cues and sequences – Look at cues in sequences](#).

FaderValue:

When recording a track using a fader, the fader movement is recorded in steps between 0% and 100%. This column displays the value of the fader at each step. Tap and hold a cell in this column to open the calculator, then set a new value.

Appearance:

Displays the appearances assigned to markers and time ranges. Tap and hold to open a cell in this column. The select pop-up list will open, then assign a new appearance. For more information, see [Appearances](#).

Start:

Displays the start position of markers and time ranges on the timeline. Tap and hold a cell in this column to open the calculator, then set a new value. For more information on **Time display format**, see [Timecode Settings](#).

Duration:

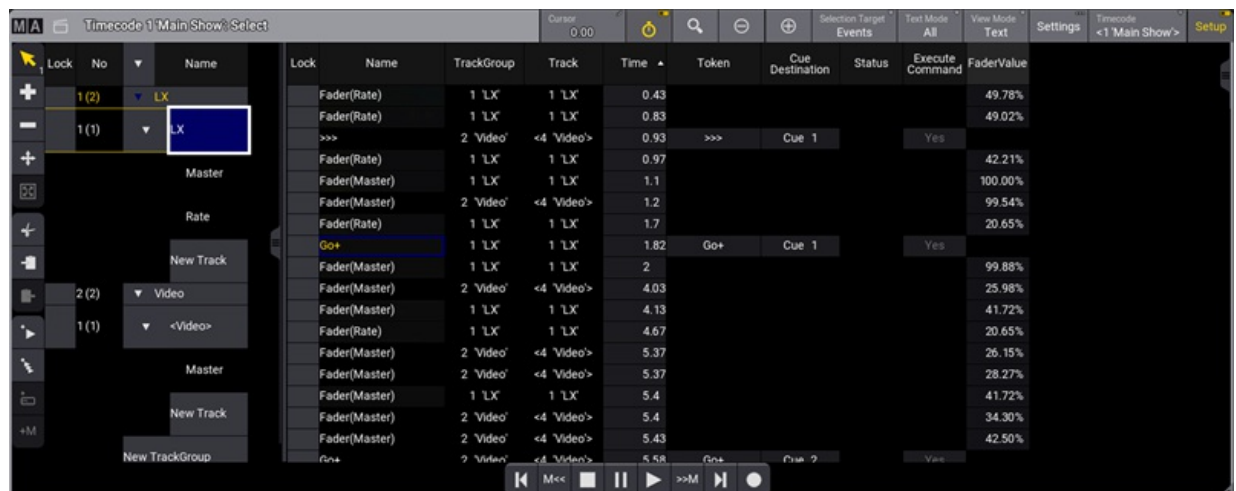
Displays in seconds the time duration of markers and time ranges. Tap and hold a cell in this column to open the calculator, then set a new value. The value is displayed in angle brackets < > when the duration is set to the end of the timecode show.

Speed Factor:

Tap and hold to open, then set a value to multiply or divide the speed.

Text Mode

When View Mode is set to Text, the timeline is hidden.



Timecode view with View Mode set to Text



Hint:

Only cells with a gray background can be edited.



Hint:

Tap and hold a cell of a column header to sort the display.

37.6. Record a Timecode Show

Requirements:

- Patched fixtures. For more information, see [Patch and Fixture Setup](#).
- Stored values in cues and created sequences. For more information, see [Cues and Sequences](#).
- Open the timecode pool and the timecode view. For more information, see [Add windows](#).

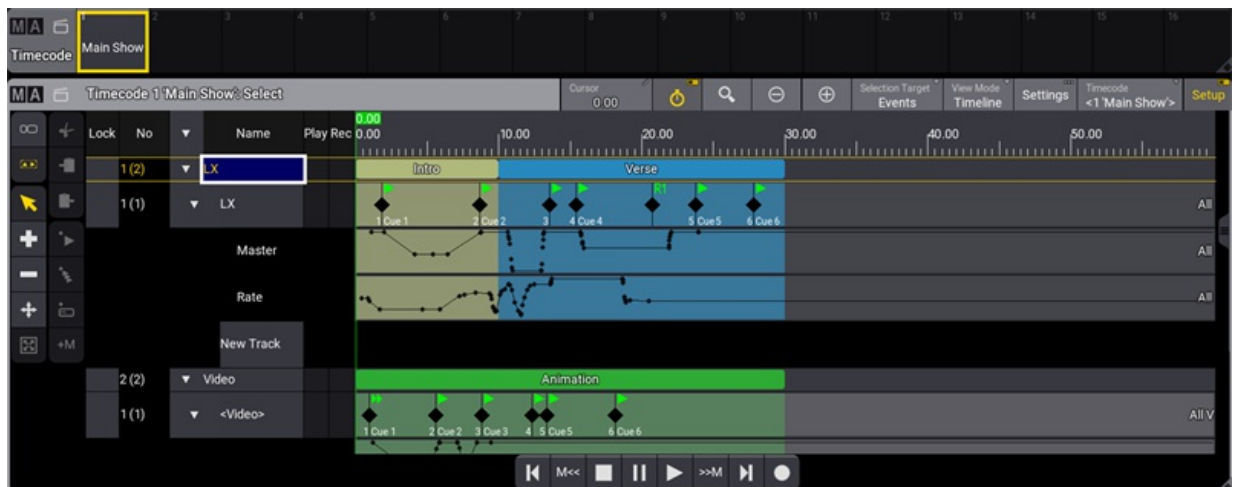
Example Internal Timecode



Important:

This sequence of operations is based on an internally generated timecode.


1. Store a new object in the Timecode Pool and label it.
For more information, see [Windows, Views, and Menus](#).
2. Select the Timecode in the pool.



Timecode window in setup mode


3. In the title bar, tap **Setup**.
For more information on different views, see [Toggle view mode](#).
4. To enable the follow time cursor, tap the stopwatch in the title bar. When enabled, the view follows the time cursor.
If it is disabled, the cursor moves beyond the displayed view.
5. In the title bar, tap **Settings** in the pop-up menu, tap and hold **TC Slot** then select **<Internal>**.
For more information, see [What are timecode slots](#).



6. To start recording, tap .
The time starts to run.

7. To record an event, press the corresponding target executor key. This will create a new track with a new target.

8. To record fader movements, move the corresponding target fader.

9. To finish recording, tap .

The timecode show is recorded.

37.7. Record an External Timecode Show


The grandMA3 can receive 3 types of external timecode signals: SMPTE, MIDI (MTC), or Art-Net Timecode (ArtTimeCode).

Requirements:

- Patch fixtures
- Store values in cues and create a sequence. For more information, see [Patch and Fixture Setup](#) and [Cues and Sequences](#).
- A visible timecode window and timecode pool. For more information, see [Add Windows](#).

Here are the steps to record a timecode show from an externally generated source.


SMPTE:

1. Connect the external SMPTE source to the LTC XLR connector at the back of the console. For more information, see [Device overview](#).
2. Press **Menu**, then tap **Connector Configuration**. This will open the **Output Configuration** menu.
3. Tap and hold a cell in the SMPTE Mode column and set the mode to In. (The default mode is In).
4. Tap and hold a cell in the SMPTE TC column to open the TC slot pop-up list. (The default is Slot 1).
5. Select the slot that will receive the external SMPTE signal.
6. Store a new object in the Timecode Pool. Label it and select it. For more information, see [Windows, Views, and Menus](#).
7. Tap **Settings** in the title bar of the timecode window. For more information, see [Timecode settings](#).
8. Tap **TC Slot** and select a slot from 1 thru 8. This has to match the slot set for **SMPTE TC** in the Output Configuration menu.
9. Tap **Mode** in the title bar of the timecode view and set it to **Setup**.
10. Assign a new **Target** to a track. For more information, see [Toggle view mode](#).
11. Turn on the **Playback Toolbar**. For more information, see [Timecode Settings](#).
12. Tap  in the Playback Toolbar to start recording.




Hint:

At this point, the timecode show is pending. The cursor will start moving when the external signal is detected.

13. To record a cue, press the corresponding target executor key. For more information, see [Toggle view mode](#).
14. To record fader movements, move the corresponding target fader.
15. To finish recording, tap  in the toolbar.

MIDI:


1. Connect the external MIDI source to the MIDI In connector at the back of the console. For more information, see [Device overview](#).
2. Press **Menu**, then tap **Connector Configuration**. This will open the **Output Configuration** menu.

3. Tap and hold a cell in the MIDI Mode column and set the mode to In. (The default mode is In).
4. Tap and hold a cell in the MIDI TC column to open the TC slot pop-up list. (The default is Slot 1).
5. Select the slot that will receive the external MIDI signal.
6. Store a new object in the Timecode Pool. Label it and select it. For more information, see [Windows, Views, and Menus](#).
7. Tap **Settings** in the title bar of the timecode window. For more information, see [Timecode settings](#).
8. Tap **TC Slot** and select a slot from 1 thru 8. This must match the slot set for **MIDI** in the Output Configuration menu.
9. Tap **Mode** in the title bar of the timecode view and set it to **Setup**.
10. Assign a new **Target** to a track. For more information, see [Toggle view mode](#).
11. Turn on the **Playback Toolbar**. For more information, see [Timecode Settings](#).
12. Tap  in the Playback Toolbar to start recording.




Hint:

At this point, the timecode show is pending. The cursor will start moving when the external signal is detected.

13. To record a cue, press the corresponding target executor key. For more information, see [Toggle view mode](#).
14. To record fader movements, move the corresponding target fader.
15. To finish recording, tap  in the toolbar.

ArtTimeCode:

1. Connect a cable to one of the 3 Ethernet interfaces at the back of the console.
2. Press **Menu**, then tap **DMX Protocols**.
3. Tap **Art-Net** in the top left corner.
4. In the **Mode column**, tap and hold a cell; from the pop-up list, tap **Input**.
5. In the **Timecode Slot Column**, tap and hold a cell. This will open the calculator. Enter the slot number that will receive the external signal.
6. Set the **Interface** IP address and **Enable Input**. For more information, see [Art-Net menu](#).
7. Store a new object in the Timecode Pool. Label it and select it. For more information, see [Windows, Views, and Menus](#).
8. Tap **Settings** in the title bar of the timecode window. For more information, see [Timecode settings](#).
9. Tap **TC Slot** and select a slot from 1 thru 8. This must match the slot number set in the Art-Net configuration menu.
10. Tap **Mode** in the title bar of the timecode view and set it to **Setup**.
11. Assign a new **Target** to a track. For more information, see [Toggle view mode](#).
12. Turn on the **Playback Toolbar**. For more information, see [Timecode Settings](#).
13. Tap  in the Playback Toolbar to start recording.


- 14.



Hint:

At this point, the timecode show is pending. The cursor will start moving when the external signal is detected.



15. To record a cue, press the corresponding target executor key. For more information, see **Toggle view mode**.
16. To record fader movements, move the corresponding target fader.
17. To finish recording, tap  in the toolbar.

37.8. Edit a Timecode Show

Requirement:

- Record a timecode show.

For more information, see [Record a Timecode Show](#).

It is possible to edit an existing timecode show to adjust details and accuracy in time.




Important:

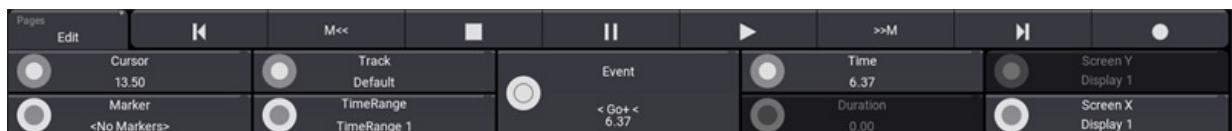
A track is created for each sequence, and an event is created for each cue.

1. Tap **Setup** in the title bar to enable **Setup** mode. The timecode encoder bar is displayed.
2. Use the dual encoders to navigate the tracks and select events or time ranges.



Hint:

It is possible to select multiple events, markers, or time ranges using the lasso selection method. The number on the right of the tool button  indicates the number of selected events.



Timecode encoder bar

Edit Using Dual Encoders

Cursor: (First dual encoder)

- Rotate the dual encoder inner ring clockwise or counterclockwise to move the cursor (green timeline) on the time axis.
- Press and rotate the dual encoder inner ring to move the cursor twice as fast.
- Press the dual encoder inner ring to open the calculator and enter a new value.

Marker: (First dual encoder)

- Rotate the dual encoder outer ring clockwise or counterclockwise to select markers on the selected track group.
- Press the dual encoder key, and select a marker from the drop-down list.

Track: (Second dual encoder)

- Rotate the dual encoder inner ring to select a track group or a track.
- Press the dual encoder inner ring, and select a track from the drop-down list.



TimeRange: (Second dual encoder)

- Rotate the dual encoder outer ring to select a time range on the selected track or track group.
- Press the dual encoder key, and select a time range from the drop-down list.

Event: (Third dual encoder)

- Rotate the inner or outer ring of the dual encoder to select an event on the selected track.
- Press the inner ring of the dual encoder and select a token from the drop-down list.
- Press the dual encoder key, and select an event from the drop-down list.

Time: (Fourth dual encoder)

- When an event is selected, rotate the dual encoder inner ring to move the event on the timeline.
- After, when selecting a marker or time range, rotate the dual encoder inner ring to adjust the starting point.

Duration: (Fourth dual encoder)

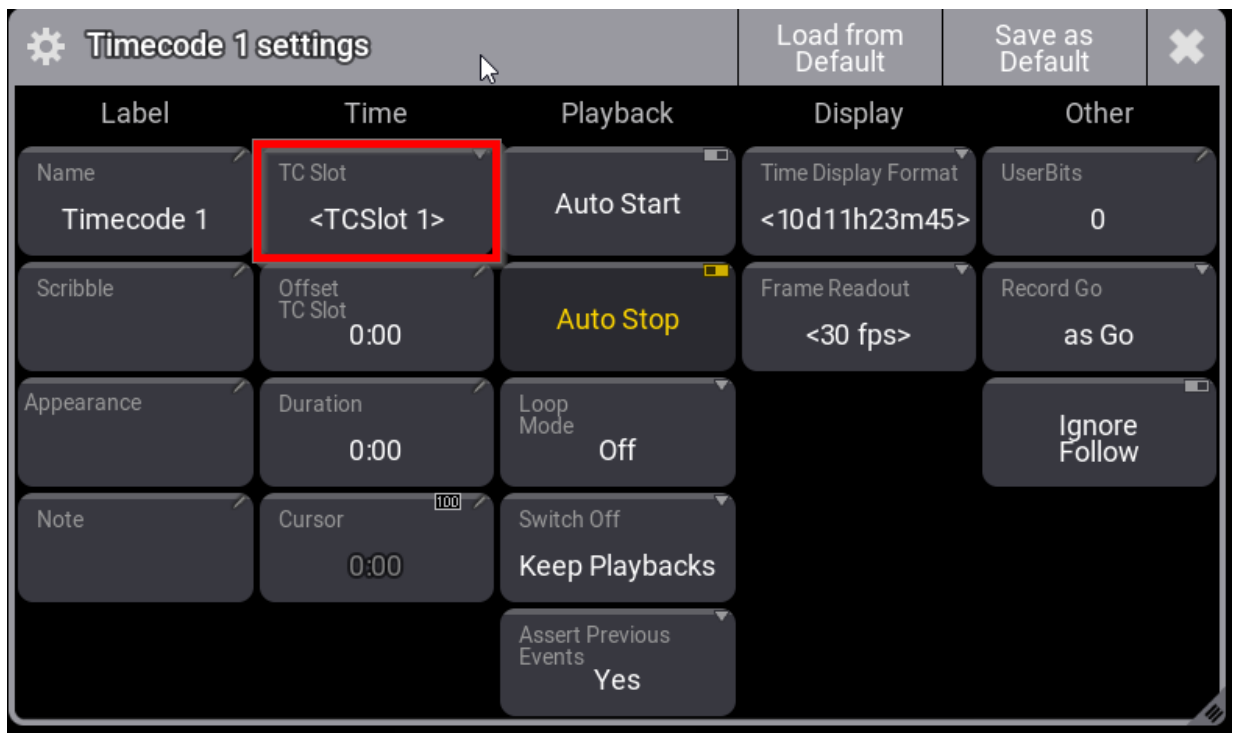
- Rotate the dual encoder outer ring to adjust the duration of the selected marker or time range.

37.9. External Sources

Connect the external timecode source to a grandMA3 console using an XLR cable for LTC/SMPTE or MIDI cable. For more information, see [Connect LTC](#) / [Connect MIDI](#).

For the following example, the LTC connector was used for timecode input.

- For more information about Timecode, see [Timecode](#) topic.
 1. Tap **Settings** in the title bar of the timecode window.
 2. Tap and hold **TC Slot**, then select the slot you wish to use.

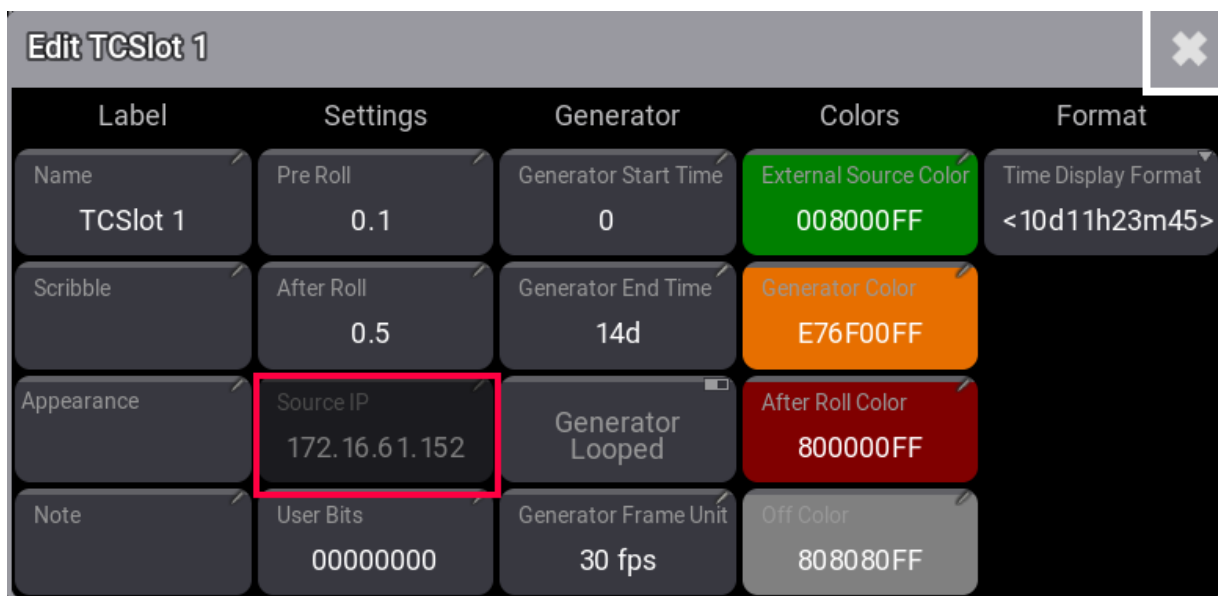


Timecode setting pop-up.

3. Tap **Menu**, then tap **Connector Configuration**; this will open the **Output Configuration** menu. For more information, see [What are timecode slots](#).

To see the IP address of the timecode source:

1. Start to run the timecode on the external source.
2. Open the Timecode Slots pool window.
3. Select and edit the first **TCSlot**. A pop-up opens.
4. The IP source of the timecode is shown in the **Settings** row. The Source IP button is an info **window** and can not be edited.



The connected Source IP is shown.

The text "No Cable" in the Source IP button can be an indicator that:

- Indicates that no network cable is connected to the selected MA-Net-Interface.
- No timecode show is currently running on the selected TCSlot.
- The time is running on another timecode slot.



38. Layouts

Layouts are basically two-dimensional drafts where it is possible to arrange fixtures, macros, groups, and other pool objects.

Layouts are created in the layout pool and are displayed and edited in the layout window.



Restriction:

The maximum number of elements per layout is 10 000.

Layouts are stored in the layout pool.

To set the defaults for new layout elements, press **Menu**, tap **Preferences and Timings**, then tap **Layout Elements** on the left side of the window.

38.1. Create a Layout

It is possible to create layouts for every need and taste.



Important:

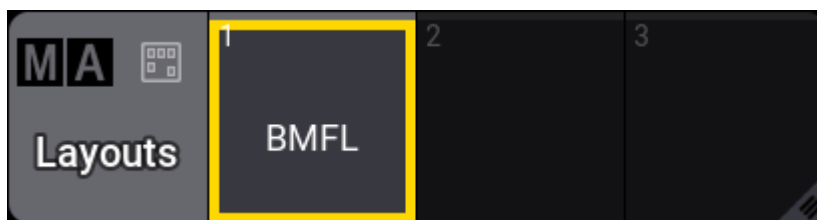
Create a layout in the layout pool and edit the layout in the layout window. For more information, see [Edit layout](#).

To add a layout pool window, follow the instructions under [Add windows](#).

In the Add Window pop-up:

1. Tap **Data Pools**.
2. Tap **Layouts**.

The layout pool opens.



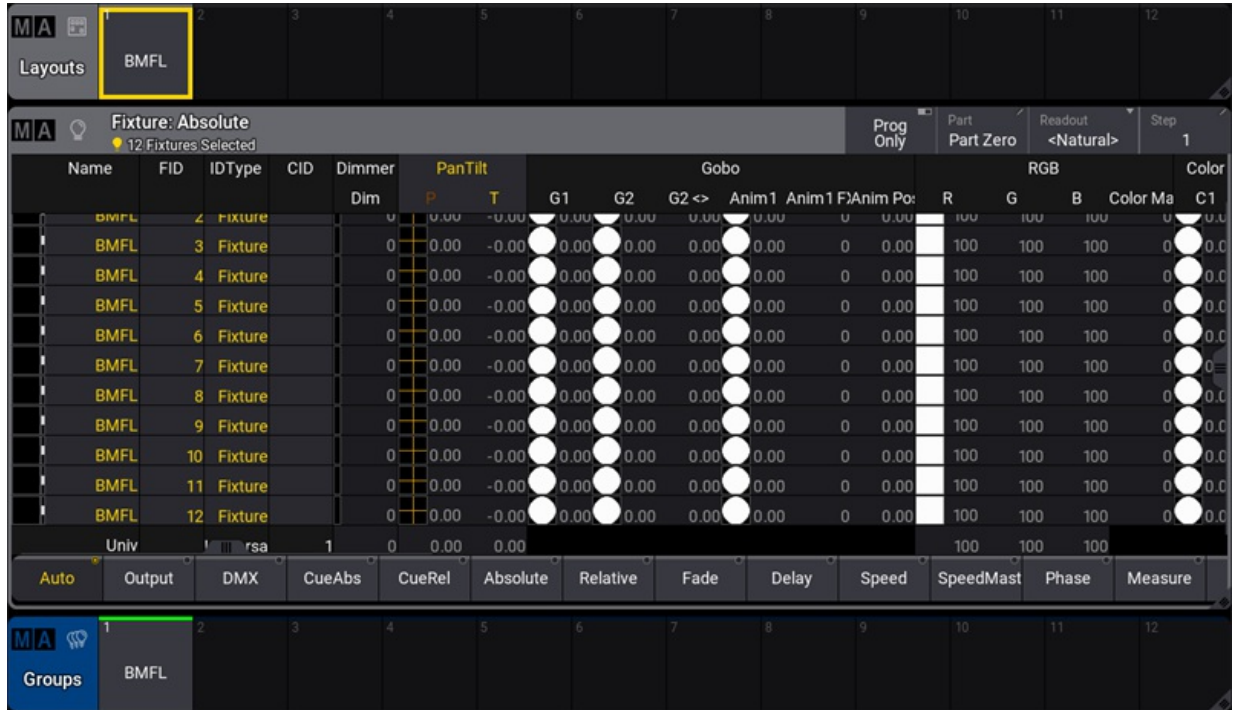
Layout pool window

3. Select fixtures using the fixture sheet or select a group.



Hint:

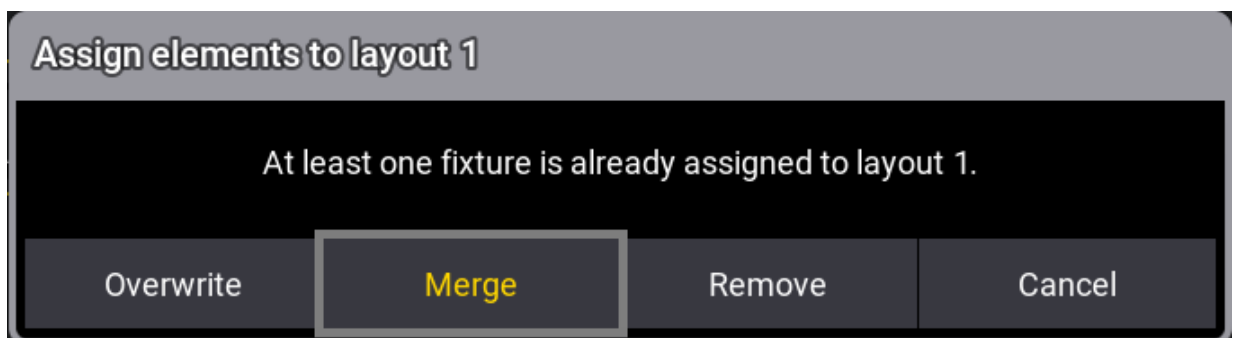
The selection grid position of the selected fixtures is assigned to layouts. For more information, see [Selection Grid](#).



Fixture Sheet, Layouts Pool and Groups Pool


- To add the selected fixtures to a layout, press **Assign** and tap a cell in the layout pool or tap the area in the layout window to which you would like to assign the fixtures. The fixtures are added to the layout.

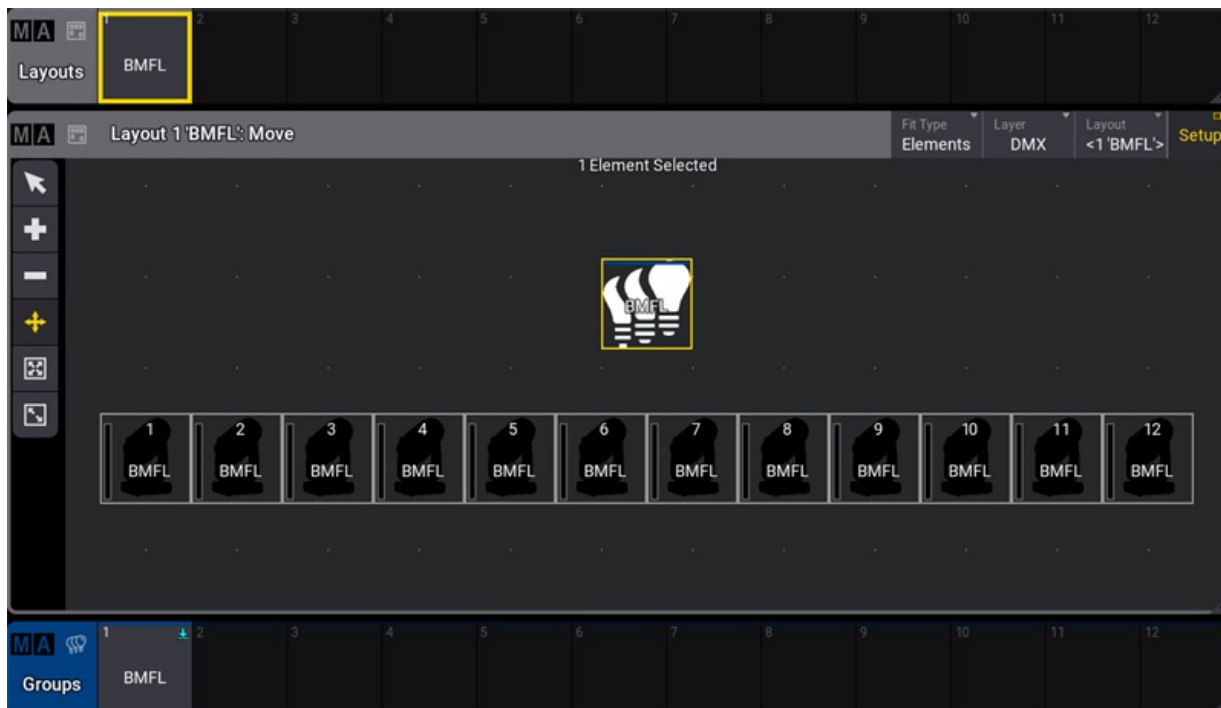
When assigning a fixture to a layout where that fixture is already assigned, a pop-up will open and offer the following options:



Assign pop-up


To add pool elements in the layout window, press, for instance, **Assign Group** **3** and then tap the area of the layout window to which you would like to assign this group. The pool element is added to the layout window.

 **Hint:**
To operate the layouts faster and to have all at a glance, arrange the layout pool and the layout window on one screen.




Layout view

Tap **Setup** in the title bar to enable the toolbar on the left side of the layout view.


 **Hint:**
When **Setup** is enabled, the layout encoder bar is also enabled. See, [Layout encoder bar](#).

Select an Element

1. To select an element, tap .
2. Tap the element you wish to select or use the lasso command to select multiple elements.
3. The element is selected.

Add an Element


The upper left corner of a layout element is set to the insertion point.

1. To add an element in the layout window, tap .
2. Tap an empty space in the layout window.
3. The element is added.




4. To add an object to the element, see [Edit layout view](#).


Delete an Element

1. To delete an element in the layout window, tap .
2. Tap the element you want to delete.
3. The element is deleted.


Move an Element

1. To move an element within the layout window, tap .
2. Tap, hold and drag the element to move it.
The element is moved.

Resize an Element


1. To resize an element, tap .
2. Select the element you would like to resize.
3. Tap, hold and drag to resize.
The element is resized.

Resize an Element with Fixed Ratio

1. To resize an element and preserve the actual size ratio, tap .
2. Select the element you would like to resize.
3. Tap, hold and drag to resize.
The element is resized with a fixed ratio.

Layout View Fit Options

In the title bar, tap **Fit Type** and select the fit option you wish to perform.


- Select **Elements** to fit all layout elements in the window.
- Select **Canvas** to fit the entire canvas in the window. If some elements are assigned far outside the canvas, they will not be displayed.
- Selecting **Both** will fit the canvas and all elements.
- Tap  to execute "Zoom to Fit" depending on the selected **Fit Type**.



Hint:

The canvas fit mode can be modified. For more information, see [Layout View Settings](#).

- Create a layout with the command line using the layout keyword. For more information, see [Layout keyword](#) and [Assign keyword](#).

 User name[Fixture]> Assign Layout 1

Cloning in Layouts

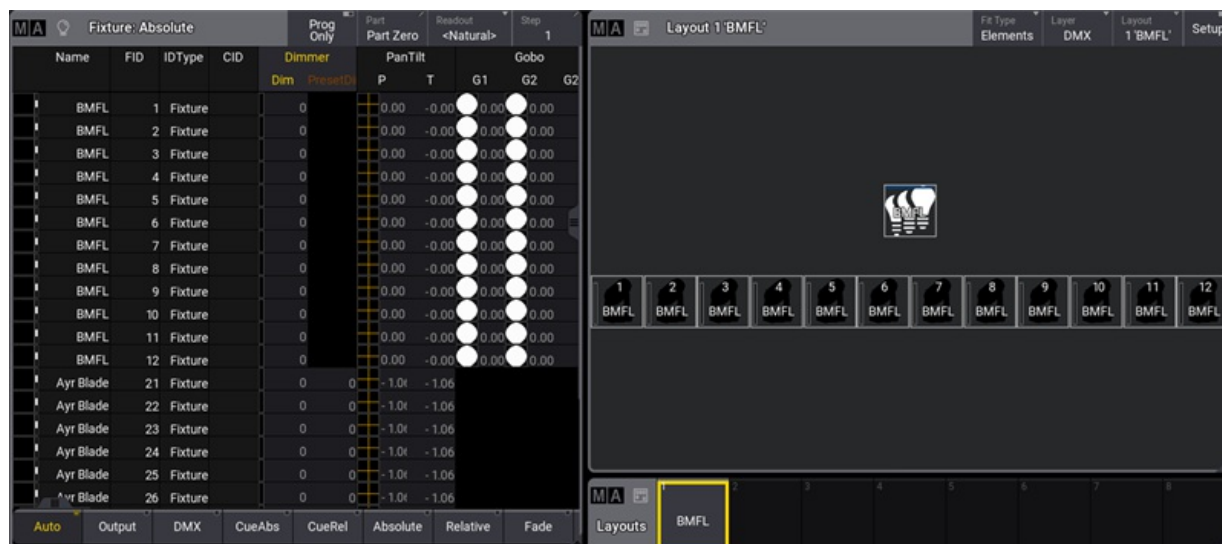
It is possible to clone fixtures into a layout using a clone syntax. For more information, see [Clone keyword](#).

Requirements:

- Some patched fixtures. For more information, see [Patch and Fixture Setup](#).
- Layout Viewer window, Fixture Sheet and Layout Pool is open.

To clone fixtures into a layout, follow the steps in the example below:

1. Tap the **BMFL** pool in the Layout Pool window. The BMFL layout is shown in the Layout Viewer window as shown in the image below:

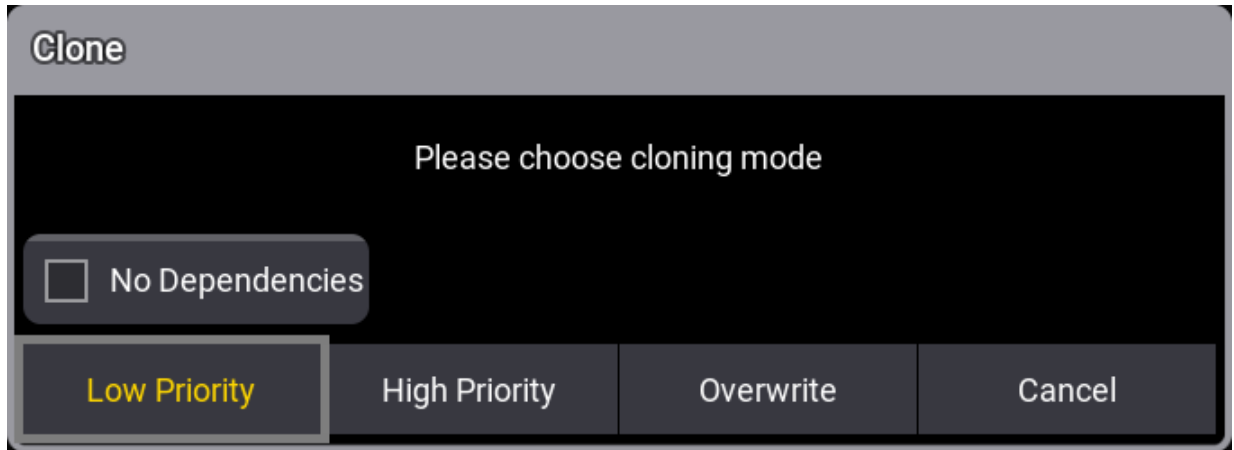


BMFL Pool is selected and is shown in the Layout Viewer window above.

2. To clone new fixtures into the first layout pool, type the following command into the command line:

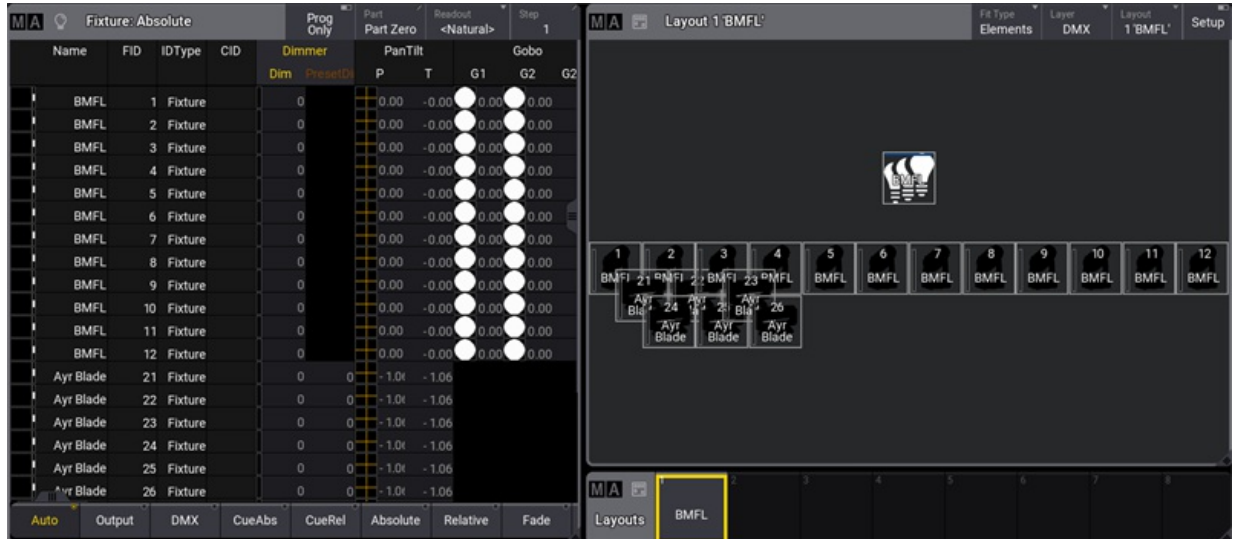
```
MA User name[Fixture]> Clone Fixture 1 Thru 3 At Fixture 21 Thru 26 If Layout 1
```

- The Clone pop-up appears, offering the choices of Low Priority, High Priority, Overwrite, and Cancel. Tap **Overwrite**. To learn more about the clone priority options, see [Clone Keyword](#).



Clone pop-up.

- The new layout elements will be placed with a small offset to the existing layout elements. All settings of the layout element will be cloned, too:



Fixtures are placed with an offset in the Layout Viewer window.



38.2. Assign Multipatch Fixtures

Requirement:

Patched multipatch fixtures. See [Add Multipatch Fixtures](#).

Multipatch fixtures can be selected via the command line by typing their CID number or global position order in the patch. See [Add Multipatch Fixtures](#).


The following is an example of how to assign primary fixture 1 and its multipatch fixtures to a layout:

Assign the primary fixture:

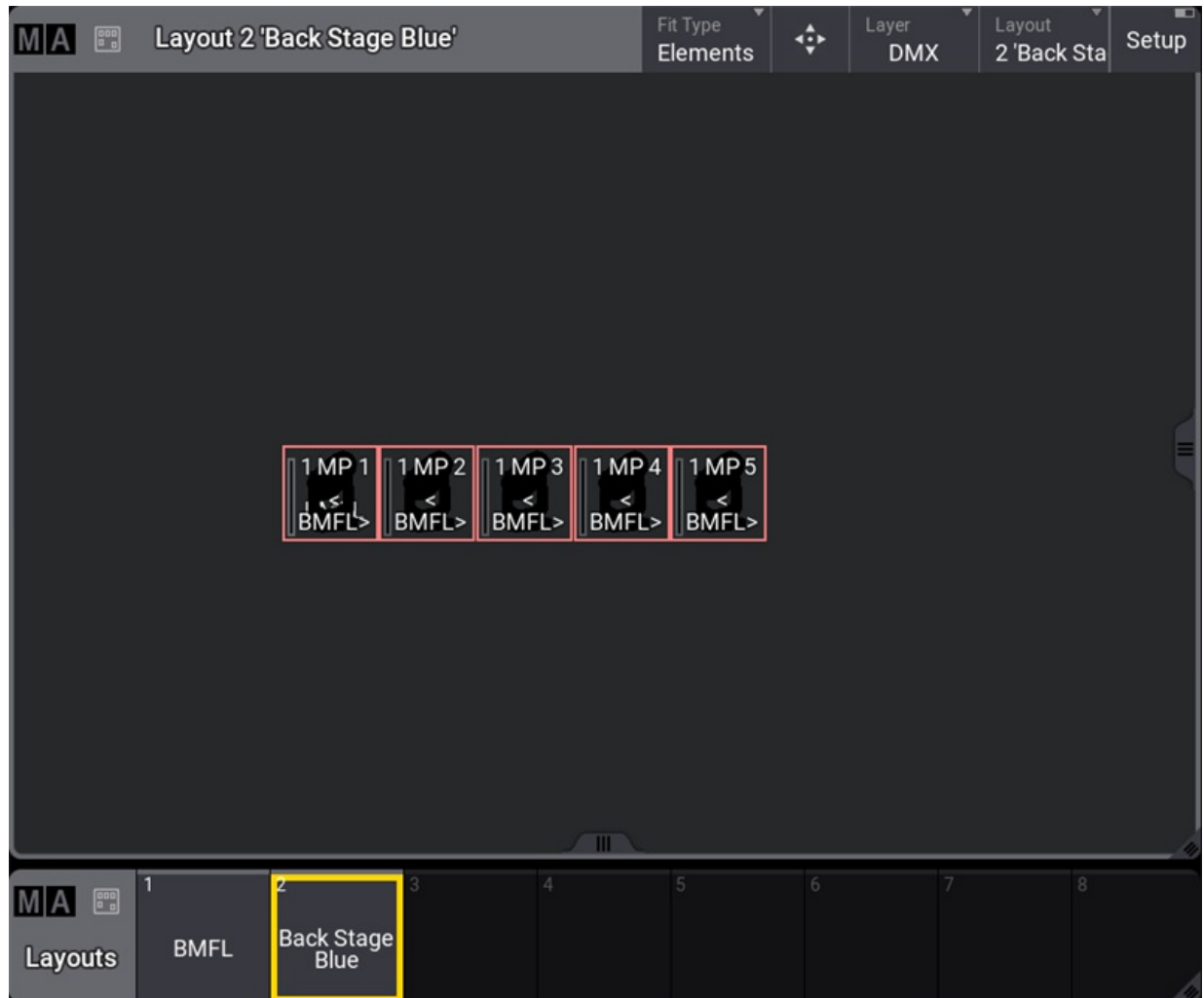
1. Press **Assign** **Fixture** **1**.
2. Tap an empty layout pool object, then label it.

To assign fixture 1 multipatch fixtures 1 thru 5:

1. Select the new layout pool object you have created.
2. Write the following syntax in the command line.

```
 User name[Fixture]> Assign Fixture 1 Multipatch 1 Thru 5
```

3. Tap in the layout view where you want the multipatch fixtures to be assigned.



Layout 2

The selection color is light red instead of yellow (default colors). In the fixture sheet, the primary ID will flash between yellow and light red when a multipatch fixture is selected.

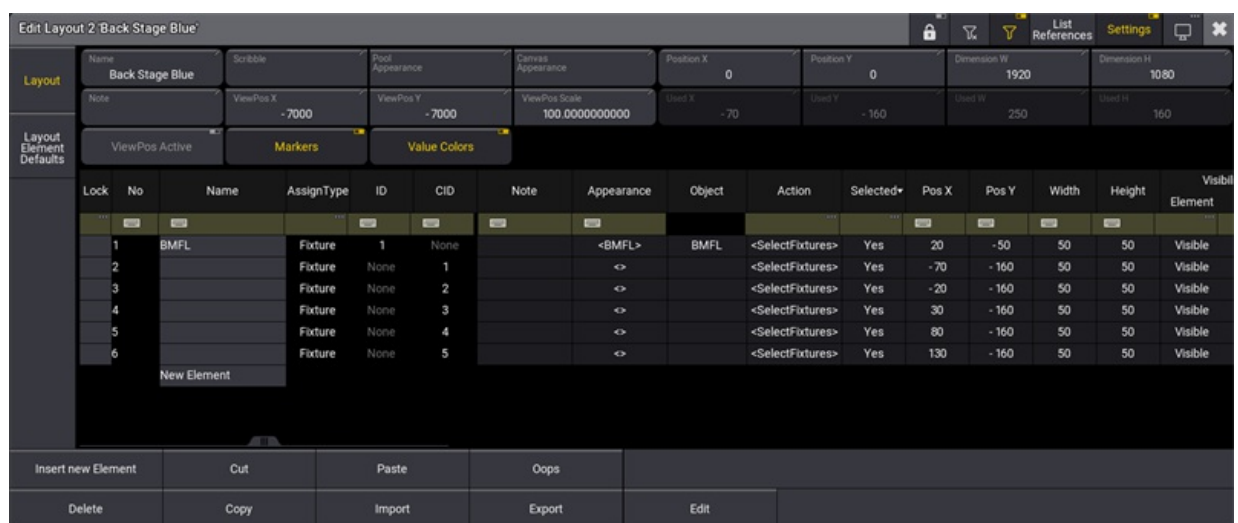
38.3. Edit Layout

Requirements:

A created layout. For more information, see [Create a Layout](#).

- To edit a layout, use the swipecy command or press **Edit**, then tap the layout pool object.


The Edit Layout overlay opens:



Edit layout window

The following is a description of the buttons in the Edit Layout window:

- To rename the layout, tap **Name**, enter the new name, tap **Enter** on the virtual keyboard, or press **Please**. For more information, see [Label pool objects](#).
- To add a scribble to the layout pool object, tap **Scribble**. For more information, see [Scribbles](#).
- To add an appearance to the layout pool object, tap **Pool Appearance**. For more information, see [Appearances](#).
- To add a canvas appearance to the layout, tap **Canvas Appearance**. This will apply to the layout window background. For more information, see [Appearances](#).



Hint:
If no dedicated canvas appearance is set, the pool appearance will be taken as canvas appearance.

Tap the following buttons to set the position and size of the canvas:

- To set the start X position, tap **PositionX**.
- To set the start Y position, tap **PositionY**.
- To set the width dimension, tap **DimensionW**.
- To set the horizontal dimension, tap **DimensionH**.



Hint:

This feature allows you to store two layouts with the same fixtures but with different positions and zoom values.

The following buttons can be toggled:

- When **ViewPosActive** is switched on, loading a layout again will recall the previously stored view zoom and position values. This button is switched off by default.
- The layout property **Markers** allows to display the layer markers of fixtures at their corresponding layout elements when it is enabled.
- Enabling **Value Colors** in the layout settings displays the dimmer values of fixtures in their corresponding layout elements in the same colors as in the fixture sheet. When the background of the dimmer attribute cell in the fixture sheet is colored, the shadow of the value in the layout element gets colored equally.



Hint:

To see the same colors for values in a layout element as in the fixture sheet, the layout element needs to have a fixture with a dimmer assigned. The visibility for the value needs to be enabled for the layout element, too.

ViewPosX, **ViewPosY**, and **ViewPosScale** are the stored values for the layout view position and zoom.

Layout Element Defaults

Layout element defaults are part of the user profile and will apply every time you assign an object into a layout.

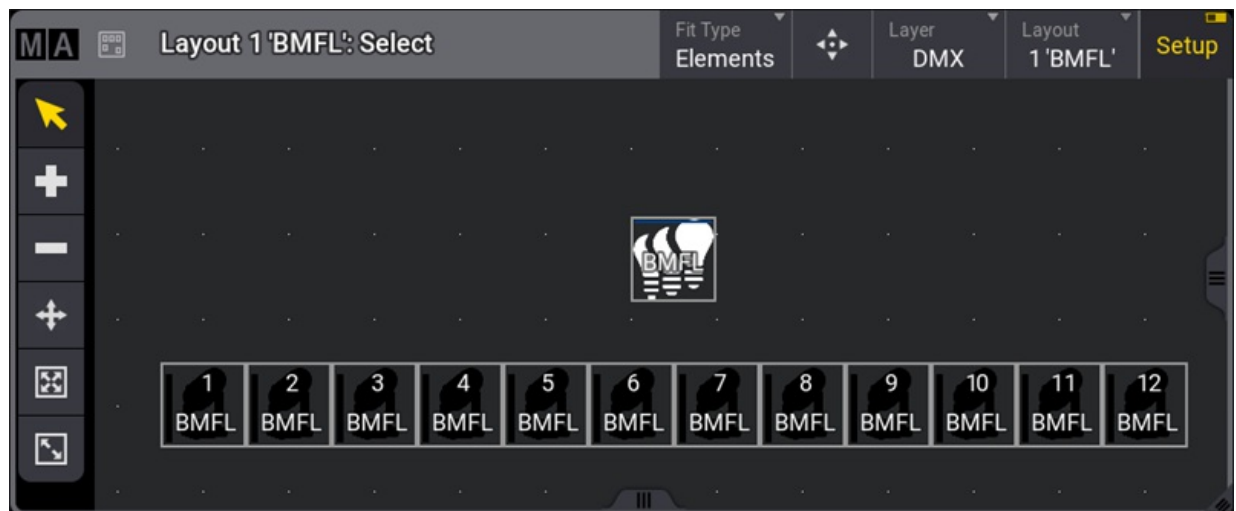
To set the default values for layout elements, press **Menu**, then tap **Preferences and Timings**. On the left-hand side, tap **Layout Elements**.

38.4. Layout View Settings

To open the Layout View window, follow the instructions under **Add windows**.

In the Add Window pop-up:

1. Tap **Tools**.
2. Tap **Layout Viewer**.



Layout View window

Open Layout View Settings

- To open the Layout View Settings, tap **MA** in the upper left corner of the layout window.

The Layout View Settings pop-up opens.



Layout View Settings pop-up

This is a description of each button in the layout view settings:

- **Layout:**
Link selected, select an existing layout, or create a new one.
- **Setup:**
Opens the setup tool bar.
- **Layer:**
Select the layer readout between **DMX**, **Value**, or **Output**.
- **Show Selection:**
Displays the selected fixtures by a yellow frame in the layout view, or not. Show Selection is not available in the Setup mode.
- **Right Click To Edit:**
When this button is toggled on, it will allow editing of objects assigned to layout elements by right-clicking or using the two-finger edit gesture.
Toggle this button off to disable the right-click and two-finger edit gesture to prevent accidental editing of objects.



Hint:

To edit an object when this button is toggled off, press **Edit** and tap the assigned object you wish to edit.

- **Lock Position:**
Toggle on to protect the layout arrangement.
- **Auto Fit:**
Toggle on to fit elements according to the window size.
- **Canvas Fit Mode:**
There are 3 fit modes, **Bar** will fit the entire canvas, **Crop** fits the canvas to its total width, and **Stretch** fits the canvas to its total height.
- **Fit Type:**
Toggle to fit the **Elements**, the **Canvas**, or **Both**.
- **Snap To Grid:**
Toggle on for objects to snap to the grid when in setup mode.
- **Snap Grid:**
Tap to open the calculator, then enter a snap value between 1 and 14 000 pixels.
Make this number smaller to be as precise as possible when moving elements in the layout view.
- **Visible Grid:**
Tap to open the calculator, then enter a value for the visible grid size.
- **Grid Color:**
Use the edit grid color popup to create the desired grid color.
- **Grid Style:**
Select the grid style between **Off**, **Lines**, or **Dots**.
- **Scale:**
When Auto Fit is off, the fader can be used to scale the layout.



- **Show Title Bar:**
Displays or hides the title bar in the layout view.

38.5. Edit Layout View

To open the layout view, follow the instructions under **Add windows**.

In the Add Window pop-up:

1. Tap **Tools**.
 2. Tap **Layout Viewer**.
- The layout view opens.



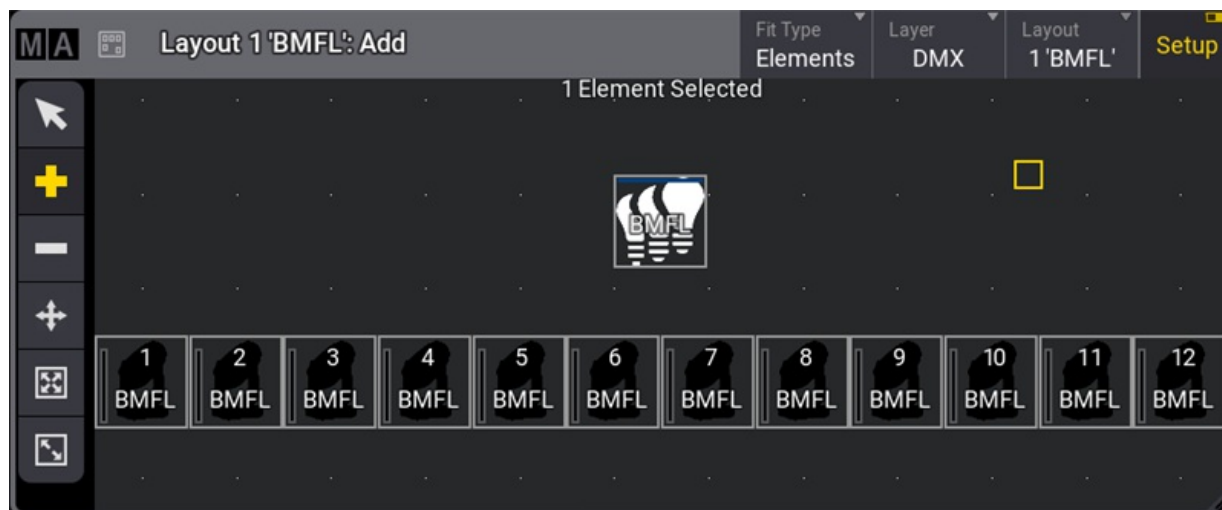
Important:

When **Setup** is enabled, it allows the editing of single layout elements. When disabled, it allows editing as in the programmer.


Add a Layout Element

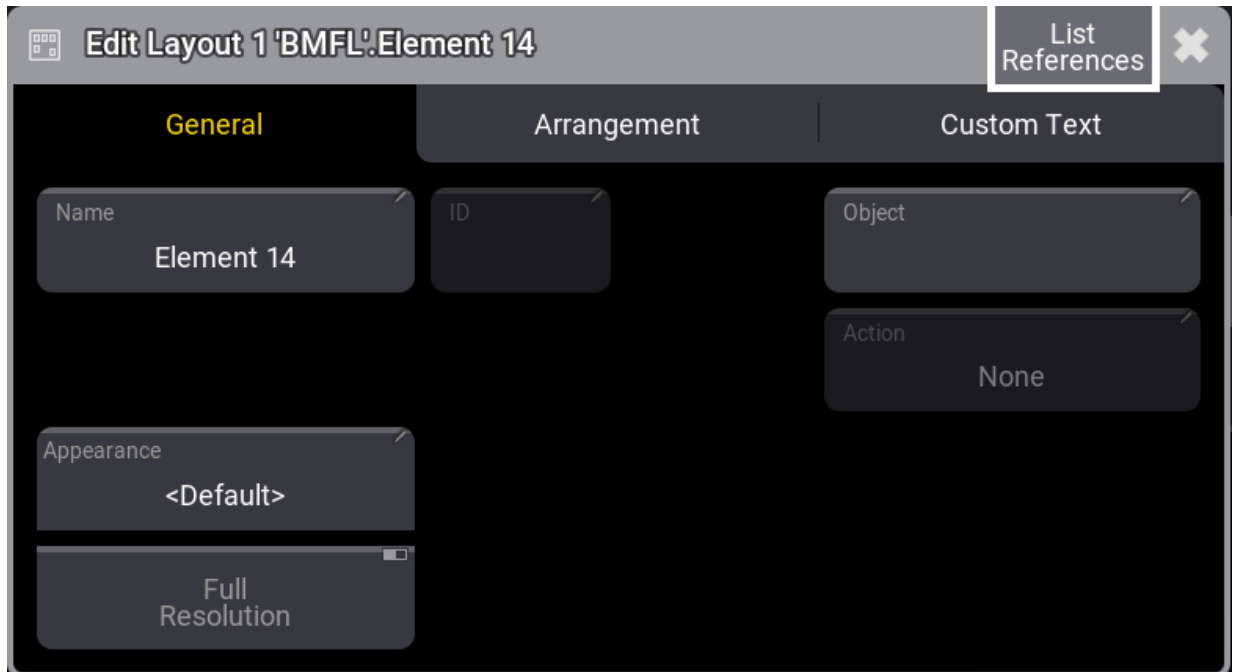
In the layout viewer window title bar, tap **Setup** to enable the setup mode.

- To add a layout element in the layout window, tap **+**.
- Tap an empty area in the window, and a new empty layout element is added. (Empty yellow square)



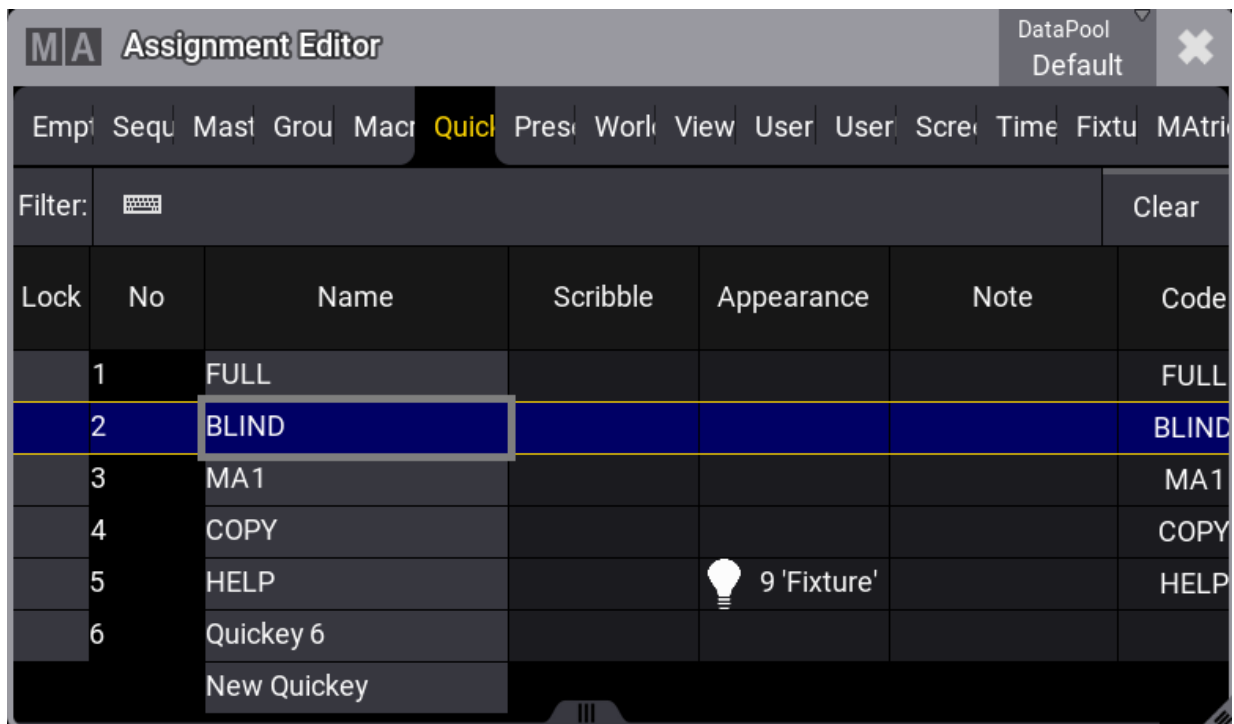
Layout View

- To assign a function/object to the element you have created in the layout, tap  in the toolbar and open the new element with the **2-finger edit** or press **Edit** and tap the new element. The edit layout element pop-up opens.



Edit Layout pop-up

- Under the **General** tab, tap **Object**, and the assignment editor opens.
- Select the function or object you wish to assign to the new element you have created.



Assignment Editor pop-up



Add an Element Using the GUI

The group pool and the Layout window should be visible on the screen for this.

For example, add group 1 to layout 1:

1. Press **Assign**.
2. Tap group 1 in the group pool.
3. Tap an empty area in the layout window.



Hint:

This can be done in setup mode and normal mode.

Add an Element Using the Command Line


```
MA User name[Fixture]> Assign Macro 1 At Layout 1
```

Layout Toolbar

Use the toolbar to edit elements. For more information, see [Create a layout](#).

38.6. Edit Layout Elements

To edit a layout element:

1. Tap **Setup** to activate the setup mode in the Layout View window.
2. Tap  and open the element with the **2-finger edit**.

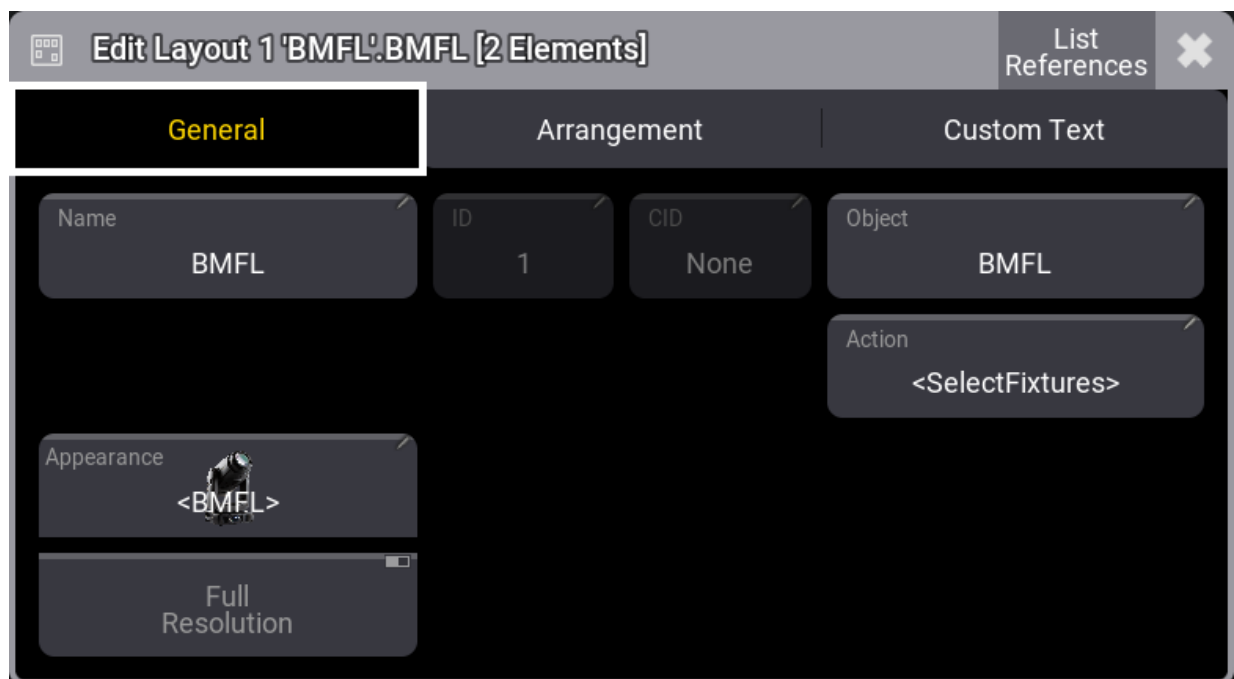
-Or-

1. Tap **Setup** to activate the Layout Encoder Bar.
2. Select the layout elements to edit in the Layout View window.
3. Tap **Edit Selected** in the Layout Encoder Bar to open the layout element editor.

The Edit Layout window is separated into three tabs. General, Arrangement, and Custom Text.

General Tab

Manages all properties regarding the object assigned to the layout element.



General tab

- Tap **Name** to open the virtual keyboard, then enter a name for the selected element.
- **ID**: When a fixture is assigned to a layout, the fixture ID is displayed here. For more information, see [What are fixtures](#). When a pool object is assigned, ID represents the pool object number.
- **CID**: if the **IDType** is something different than "Fixture", the CID is displayed here. For more information, see [Add fixtures to the show](#).
- Tap **Appearance** to open the **Select Appearance** pop-up. **<Default>** will use the appearance of the assigned object. For more information, see [Appearances](#).

- Tap **Object** to open the **Assignment Editor** pop-up, then select the object you wish to assign.
- Tap **Action** to open the **Select Action** pop-up. Select an action to be performed when tapping the element. Selecting **<Default>**, will set the default action for the assigned object. For example, if the assigned object is a fixture, the default action will be **<SelectFixture>**.
- Tap **Full Resolution** to define if an image is displayed with its full resolution (enabled) or reduced quality (disabled).

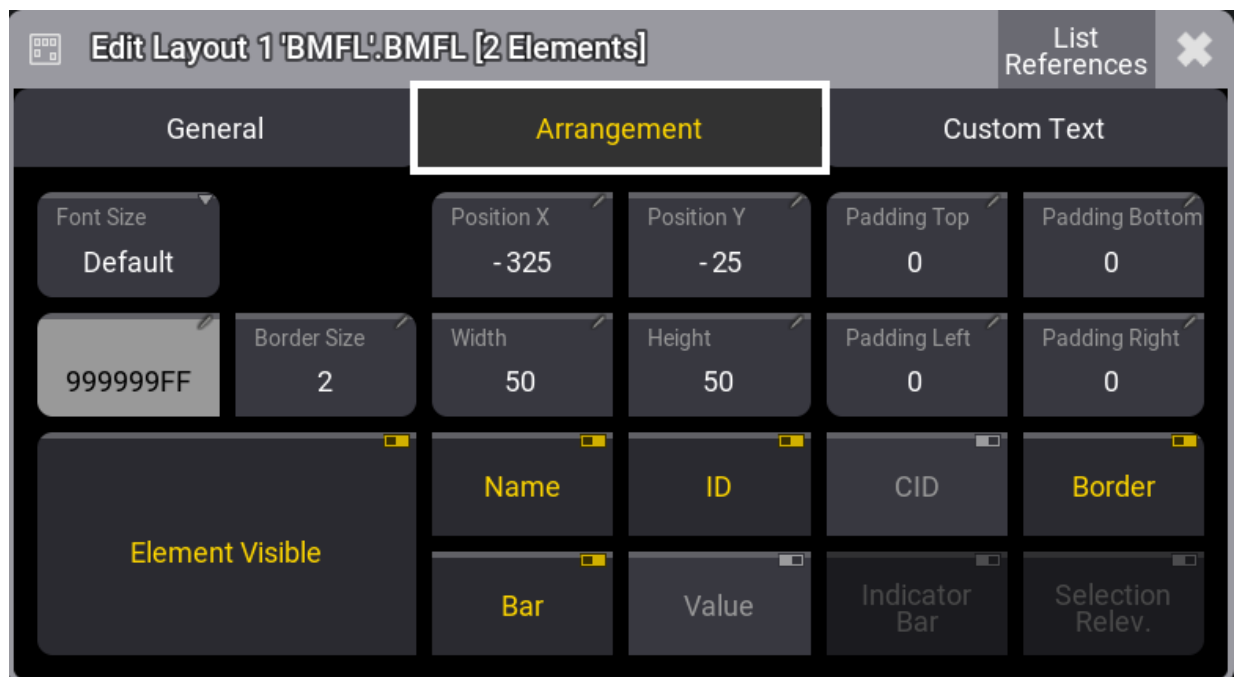


Important:

Having a lot of high-resolution images displayed may slow down the graphical user interface. In this case, the software falls back, rendering images with a lower quality. Images with 150 pixels or below will always be displayed with a lower quality.

Arrangement Tab

Manages the position and look of a layout element.

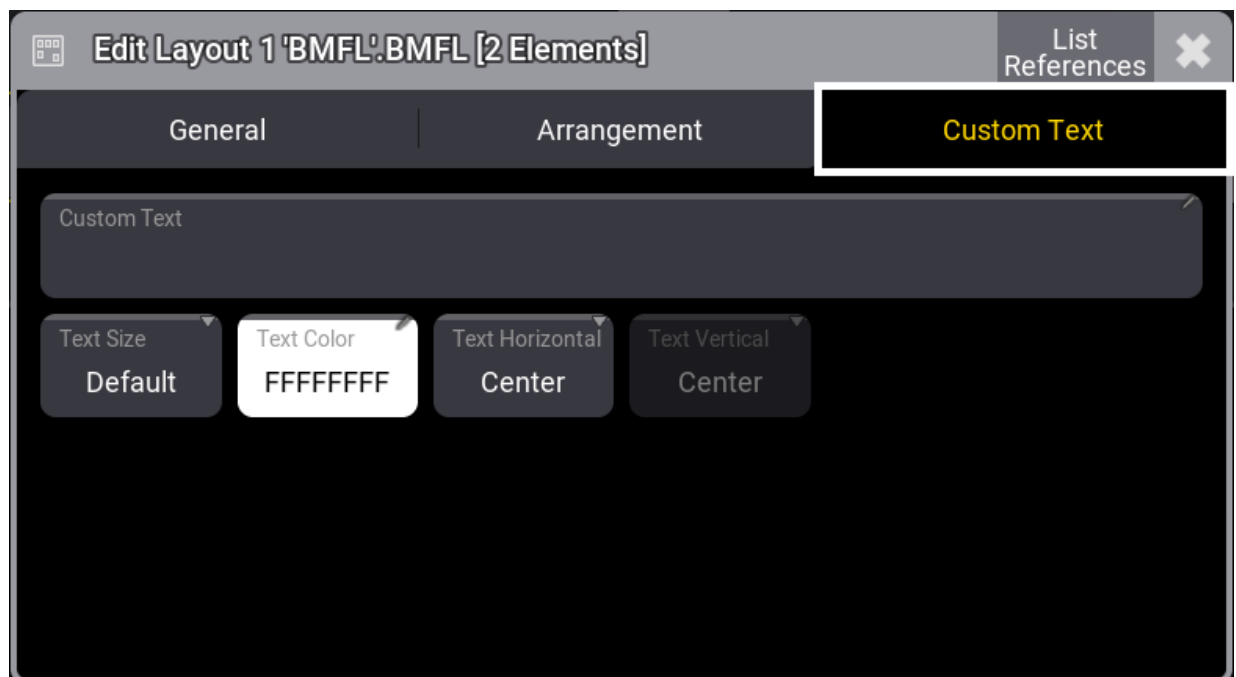


Arrangement tab

- **Font Size:** Use these buttons to personalize the custom text alignment. Setting layout elements to a dedicated font size displays the text in the set size.
- **Position X** and **Position Y:** Specify the position along X and Y in the layout.
- **Padding Top, Padding Bottom, Padding Left, Padding Right:** Padding is the distance between the edge of the element and the content inside the element.
- **Border Color** and **Border Size:** Adjust the outline color and size.
- **Width** and **Height:** This is to adjust the dimensions of the element.

- **Element Visible:** Toggle to display or hide the element.
- **Name, ID, CID, Boarder, Bar, and Value:** Toggle to display or hide the name, the fixture ID, the custom ID, the dimmer bar, the output value, and the border of the layout element.
- **Indicator Bar:** When a pool object is assigned to a layout (e.g., a group), it defines if the indicator bar will be displayed on top of the element.
- **Selection Relevance:** When enabled, the image of the layout element will be colored with the For All or For Some color. For more information, see [Window settings, Pool Settings](#).

Custom Text Tab



Custom Text tab

- **Custom Text:** Add a custom label to the element. Text on layout elements that is too long to display is cut off.
- **Text Size** and **Text Color:** Use these buttons to personalize the custom text size and color.
- **Text Horizontal:** The horizontal custom text alignment.
 - **Center:** Will align the text to the horizontal center of the layout element.
 - **Left:** The left end of the text will be aligned with the left edge of the layout element.
 - **Right:** The right end of the text will be aligned with the right edge of the layout element.
- **Text Vertical:** The vertical custom text alignment.
 - **Center:** Will align the text to the vertical center of the layout element.
 - **Top:** Places the text within the layout element at the top edge.
 - **Bottom:** Places the text within the layout element at the bottom edge.
 - **Above:** Places the text outside the layout element at the top edge.



Hint:

Multiple selected layout elements can be edited simultaneously.



The layout element selection order will determine the alignment order.

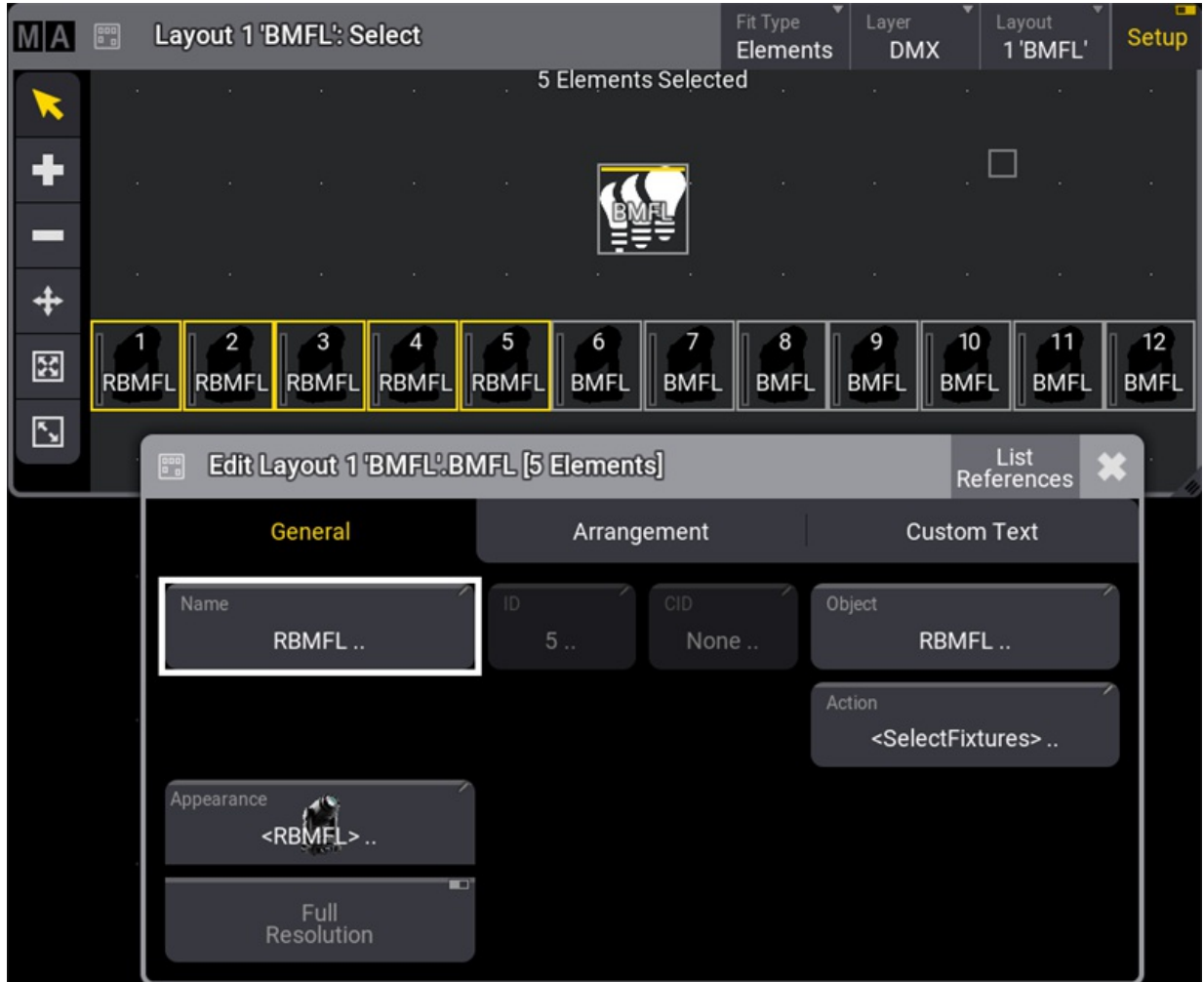


When toggling layout element properties while having multiple layout elements selected with different property values, they will switch to the same value and change their value together.

Example

To edit multiple selected layout elements simultaneously:

1. Tap **Setup** in the title bar to enable **Setup**.
2. Select multiple fixtures in the Layout Viewer window.
3. Tap **Edit Selected** in the Layout Encoder Bar to open the layout element editor. For more information, see **Layout encoder bar**.
4. Tap the **Name** button; this opens the virtual keyboard. Enter a new name and tap **Enter**.
5. Close the Edit Layout dialogue.



All selected fixtures are renamed simultaneously

38.7. Layout Encoder Bar

Use the layout encoder bar to position and arrange layout elements.

Requirements:

- A visible layout pool and a visible layout window.

For more information on how to add a window, see [Add windows](#).

There are two function settings for the layout encoder bar: **Position** and **Arrangement**.

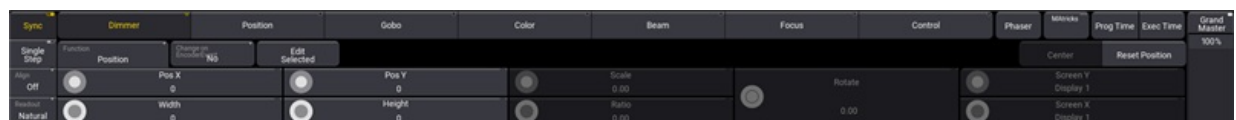
Layout Encoder Bar Function Set to Position

1. Create a layout and select it. See [Create a layout](#).
2. Tap **Setup** in the title bar to enable **Setup**; the layout encoder bar is displayed.
3. Tap **Function** in the layout encoder bar until it is set to **Position**.
4. Select the layout element (e.g., fixture) you wish to position.
5. Use the encoders to position and/or resize the element.



Hint:

When multiple elements are selected, use the **Align** function combined with the dual encoders to spread out your selection evenly.



Layout Encoder Bar

- **Pos X:** Rotate the inner ring of the dual encoder to move the selected element left and right.
- **Pos Y:** Rotate the inner ring of the dual encoder to move the selected element up and down.
- **Width:** Rotate the outer ring of the dual encoder to set the width of the selected element.
- **Height:** Rotate the outer ring of the dual encoder to set the height of the selected element.
- **Scale:** Scales the arrangement of the selected elements in the X and Y direction.
- **Ratio:** Scales the arrangement of the selected elements in the X direction.

- **Rotate:** Rotates the selected fixtures on the canvas and changes their position as a group. When only one fixture is selected, the encoder is grayed out.
- **Change on EncoderEvent:** When set to **Yes**, encoder changes are instantly transmitted to the network. If set to **No**, after stopping to turn an encoder the actual state will be transmitted after 2s.
- **Reset Position:** Tap to position the bottom left corner of the selected elements to Pos X and Pos Y zero.
- **Edit Selected:** Tap to enter the layout element editor.

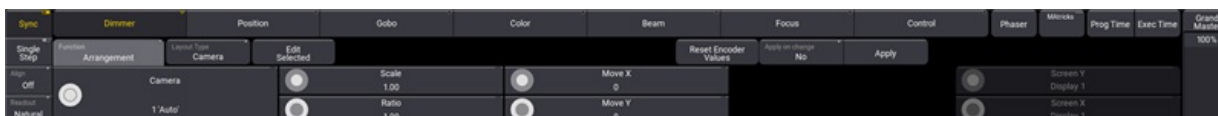
Layout Encoder Bar Function Set to Arrangement

1. Create a layout and select it.
2. Tap **Mode** in the title bar to enable **Setup**; the layout encoder bar is displayed.
3. Tap **Function** in the layout encoder bar until it is set to **Arrangement**.
4. Select the layout elements (e.g., fixtures) you wish to position.
5. Tap and hold **LayoutType**, the Select Layout Type drop-down menu opens.
6. Select one of the four arrangement types: **Line**, **Grid**, **Circle**, or **Camera**.

For more information on **Line**, **Grid**, and **Circle** arrangements, see [Position fixtures in the 3D space](#).

Camera Arrangement

Arrange your current selection similar to your camera view in the 3D window. Use the encoders to select a camera, set the scale and ratio factor, and move the selected elements.



Layout encoder bar function set to Arrangement

To do so:

1. Tap **Function** until it is set to **Arrangement**.
2. Tap **Layout Type** until it is set to **Camera**.

Set the following parameters using the dual encoders:

- **Camera:** Select the desired camera.
- **Scale:** Scales the whole selection in X and Y direction.
- **Ratio:** Defines the ratio of the selection.
- **Move X:** Move the whole selection in the X direction for a more convenient position.

- **Move Y:** Move the whole selection in the Y direction for a more convenient position.



Hint:

Press the first dual encoder inner ring or the first dual encoder key to open the camera calculator. Select a camera from the list or enter a camera number.

3. When you're satisfied with your layout setup, tap **Apply** to confirm.



Hint:

When **Apply on change** is active, **Apply** is disabled, and changes are made in real-time.

39. Plugins

Plugins are pieces of Lua code that can add features or functionality to the grandMA3.

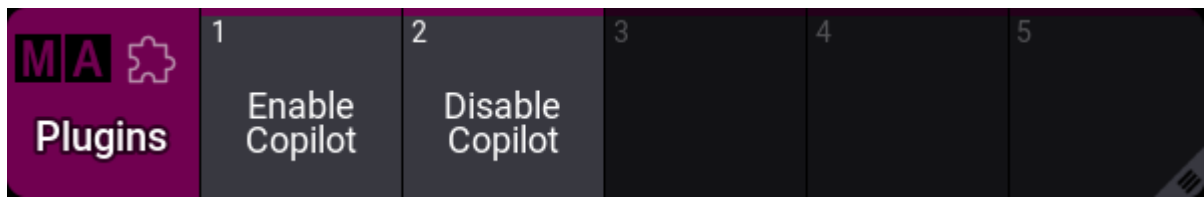
With plugins, it is possible to do more than what can be achieved with macros.



Important:

The creation and use of plugins can go deeper into the system than the "normal" usage of a console. Therefore the technical support team of MA Lighting may not be able to help you in all circumstances when using complex plugins, and plugins might have to be rewritten when migrating show files to future grandMA3 software versions.

The plugins are stored in the Plugins Pool.



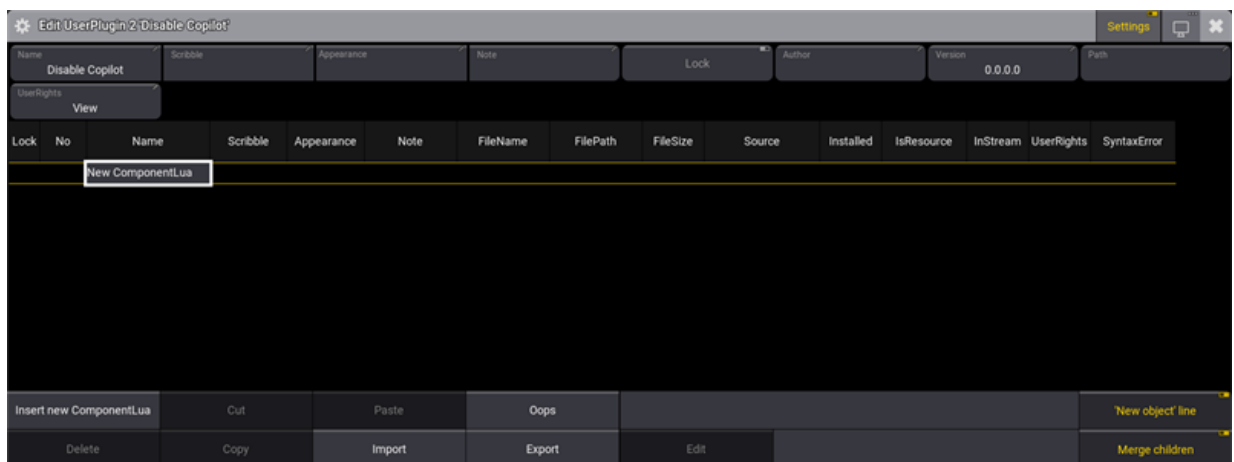
Plugin pool window

The pool can be created like any other using the **Add Window pop-up**.

Create a New Plugin

To create or add a plugin to the plugin pool, edit an empty pool object using any edit method.

The **Edit UserPlugin** pop-up opens:



Edit Plugin window

This editor can show the settings for the plugin if **Setting** is On in the title bar.



These are the settings:

- **Name:**
The name of the plugin. Tap this to edit the name.
- **Scribble:**
The assigned scribble. Tap this to select a scribble or open the editor to create a new one.
- **Appearance:**
The assigned appearance. Tap this to select an appearance or open the editor to create a new one.
- **Note:**
A note can be added to the plugin.
- **Lock:**
Toggle this On to lock the plugin from changes.
- **Author:**
This can be used to add the name of the plugin author.
- **Version:**
This custom field can be used to indicate a version of the plugin.
- **Path:**
This is a path (sub-folders) for the files inside the plugin folder (gma3_library/datapools/plugins).
- **UserRights:**
This can be used to set the needed user rights for running this plugin.

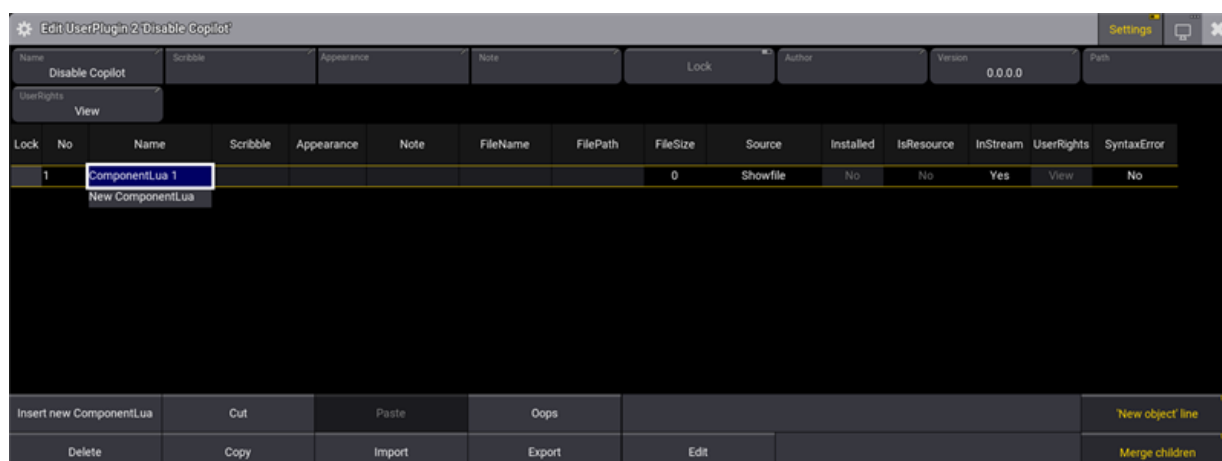
A plugin can contain several Lua components but should at least have 1.

Create and Edit a Component

A Lua component contains the Lua code. Each component is usually one file.

Add the component by pressing and holding the **New ComponentLua**.

A new component can also be added by selecting a line in the component list and tapping **Insert new ComponentLua**. This creates a new line above the selected line.





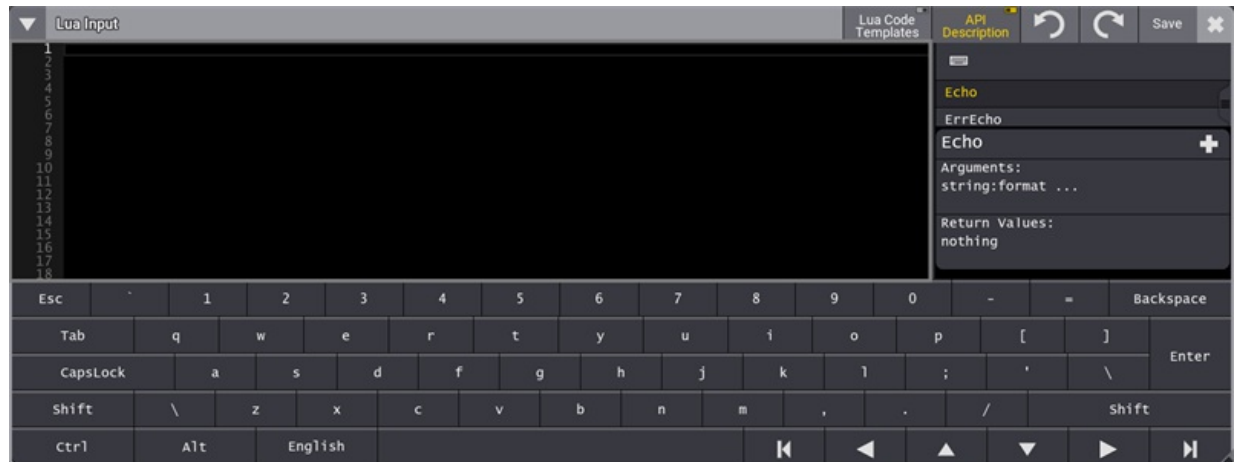
Plugin with 1 added Lua component

Each component has some settings and some information. It is the different columns in the list.

- **Name:**
This is the name of the component.
- **Scribble:**
This is the assigned scribble - this is not displayed anywhere.
- **Appearance:**
This is the assigned appearance - this is not displayed anywhere.
- **Note:**
A note can be added to the component.
- **FileName:**
This is the file name for the Lua component file. A file name is needed if the plugin is exported. Giving the component a name also creates a file name.
- **FilePath:**
This is a path (sub-folders) for the Lua file inside the plugin folder (gma3_library/datapools/plugins).
- **FileSize** (Information only):
This displays the file size of the Lua component file in bytes.
- **Installed:**
If this is set to Yes, the Lua components will be updated from the file archive they were imported from. This is useful when Lua files are edited and copied into the folder using an external editor.
To update any changes in these Lua components, the **ReloadAllPlugins keyword** must be executed.
- **IsResource** (Information only):
This indicates if the Lua file is an internal resource stored in an internal library. These files are not stored in the showfile and are not streamed.
- **InStream** (Information only):
Yes means that the Lua code is saved in the show file and streamed in the session but stays as saved in the show file.
No means that the Lua code is locally saved on the hard drive. The content of this Lua code can be updated using the **ReloadAllPlugins keyword**.
Setting **Installed** to Yes will set InStream to No.
- **UserRights:**
This can be used to set user rights on specific components.
- **SyntaxError** (Information only):
If syntax errors prevent the Lua component from being loaded into the Lua engine, the SyntaxError property turns to Yes, and the component line turns red.

Select the component that needs to be edited and tap **Edit**.

This opens the **Lua Input** pop-up:



Empty Lua Input pop-up

Text can be copied into the pop-up from an external editor (when using the grandMA3 onPC), or it can be written directly in the pop-up.

Editing existing code can also be done in the Lua Input pop-up.


Line numbers help when troubleshooting code or as general help (the numbers are not part of the code). The number of the active line is in white color, and the others are in gray color.

When **Tab** is used to indent text, a gray ">" appears where the tab was pressed.

When the desired code is input, it needs to be saved by tapping **Save** in the title bar.

The title bar has other buttons:

- **▼ (Virtual Keyboard):**
This toggles the on-screen virtual keyboard.
- **Lua Code Templates:**
This toggles the Lua Code Templates that can be used as templates for writing new code. Learn more about this [below](#).
- **API Description:**
This toggles the grandMA3-specific Lua API description on the right-hand side of the Lua Input. Learn more about this [below](#).
- **↶ (Undo):**
This undoes the last edit.
- **↷ (Redo):**
This redoes undone actions.

After the code is saved, the Lua Input pop-up can be closed by tapping the  in the upper right corner.

Run a Plugin

The plugins can be run by tapping them in the pool or using the **Plugin keyword**.


Running the plugin will execute the first component. This component needs to call other components, or specific components can be called directly using this syntax:

Plugin [Plugin_Number or "Plugin_Name"].[Component_Number or "Component_Name"]

Plugins can also be assigned to executors and view buttons. Learn how in the **Assign Object to an Executor topic**.

Lua Code Templates

The templates are example codes that can be copied into the editor.

**Restriction:**
Using a template will delete any existing code.

Tap **Lua Code Templates** in the title bar to show the templates.



The Lua code templates

The different templates are shown next to each other, and there is a horizontal scroll bar making it possible to scroll through the different templates.

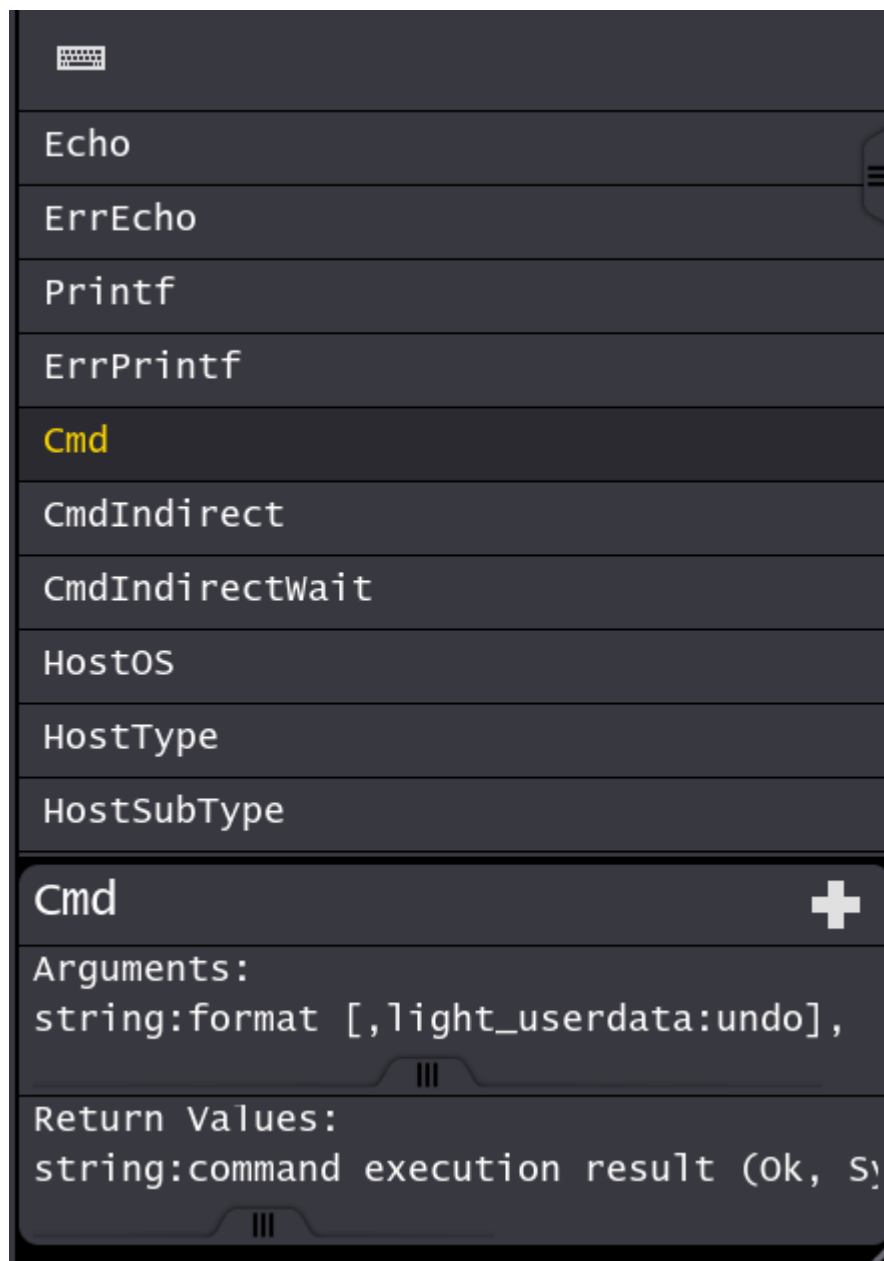
To import one of the templates, select it (yellow frame), and tap **Use Template**.

The templates can be closed without importing a template by tapping **Lua Code Templates** in the title bar or the **Back** in the lower left corner.

API Description

The API description can be shown on the right side of the Lue Input pop-up. It can be toggled On or Off by tapping **API Description** in the title bar.


It lists all the grandMA3-specific Lua functions and shows a short explanation of the required arguments and the return values.



The API description

There are three elements to the description. At the top is a search input field where text can be input to search the list of functions.

Below is the actual list of functions in alphabetic order. Selecting one shows the description of the selected function below the list.

Tapping the  in the description area adds the function to the Lua input code, with the arguments, at the cursor's current location.

Export a Plugin

The plugin can be exported from the **Edit UserPlugin** by tapping .

This creates an XML file for the plugin.

If the Lua components have a file name ending in ".lua", an extra LUA file is exported for the component. If there is no valid file name for the component, it is then stored in the XML file, coded in the "base64" format.





Restriction:

The Lua files are only exported if the Lua component has "Yes" in the "InStream".

Import a Plugin

All the required files are needed to import a plugin successfully. The XML file is always needed; if the Lua components are stored as extra Lua files, they are also needed. These files must be in the `../gma3_library/datapools/plugins` folder on the desired drive.

1. Create a new plugin.
2. Tap .
3. Select the desired drive and XML file.
4. Tap .
5. The plugin is imported, and the editor can be closed.



39.1. What is Lua

Lua is a scripting language designed to support general procedural programming. It offers support for object-oriented programming, functional programming, and data-driven programming. Lua is implemented as a library, written in *clean C* (a common subset of ANSI C and C++).

For more information on scripting with Lua, see www.lua.org.

The grandMA3 software supports Lua version 5.4.4 and all the built-in standard libraries.

To check which Lua version your device is currently running, use the **Version keyword** in the command line.

Besides the standard Lua libraries, there are some custom grandMA3 Functions.

A list of all the current grandMA3 functions can be created by using the **HelpLua keyword**. This keyword creates a file called "grandMA3_lua_functions.txt" in the **gma3_library** folder. Learn more about how to find this in the **Folder Structure topic**.

This list separates the functions into two categories: **Object-Free API** and **Object API**. Individual functions are described in sub-topics in these two topics.

Lua code can be executed from plugins or directly in the command line using the **Lua keyword**.



39.2. Handle - light_userdata

Many functions require a handle as an argument or return a handle.

The handle is a custom data type called "light_userdata".

The handle is a unique identifier that refers to a grandMA3 object, for instance, a specific sequence, cue, preset, or fixture.

Related Functions

- **Addr()** - Converts a handle into a numbered address path.
- **AddrNative()** - Converts a handle into a named address path.
- **FromAddr()** - Converts a numbered or named address into a handle.
- **HandleToStr()** - Converts a handle into a string with a hexadecimal number.
- **HandleToInt()** - Converts a handle into an integer.
- **StrToHandle()** - Converts a hexadecimal number string into a handle.
- **IntToHandle()** - Converts an integer into a handle.
- **ToAddr()** - Converts a handle into an address string.

39.3. Interface Functions

There are several Lua functions related to creating and displaying interface elements.

They are often some of the most complex functions that can need a lot of different input elements that allow adjustment to specific needs.

Some of them can also return user input to Lua functions.

Individual functions are described like all others, but there are sub-topics here to explain further and provide a collection of functions.



Info:

Not all Lua functions are described yet. This part of the manual will expand in the future.



39.3.1. Progress Bar

The Progress Bar is a Lua function that can create a moving progress bar on the screens.

There are several Lua functions connected to creating and running a progress bar. See links to the topics below the example.

Example

This example uses all the progress bar functions:

```
return function()
  -- create the progress bar
  local progressBarHandle = StartProgress("myProgressTitle")

  -- set start index and end index of the progress bar
  local progressRangeStart, progressRangeEnd = 1, 10

  -- Define the range of the progress bar
  SetProgressRange(progressBarHandle, progressRangeStart, progressRangeEnd)
  -- Define the text of the progress bar
  SetProgressText(progressBarHandle, "This is my ProgressBar Text")
  -- Set the progress bar value to the start of range
  SetProgress(progressBarHandle, progressRangeStart)

  -- Loop that goes through the progress bar
  for i = progressRangeStart, progressRangeEnd do
    -- Add a yield to allow other functions and delay the progress
    coroutine.yield(1)
    -- Increment the progress state of the progress bar
    IncProgress(progressBarHandle, 1)
  end

  -- remove the progress bar:
  StopProgress(progressBarHandle)

end
```

Related Functions

- [**StartProgress**](#)
- [**SetProgressRange**](#)
- [**SetProgress**](#)
- [**SetProgressText**](#)
- [**IncProgress**](#)
- [**StopProgress**](#)



39.4. Variable Functions

This topic is not about local variables in Lua plugins. This is about interactions with variables outside the plugins. For instance, the same user and global variables can be used in macros and the regular command line. Read more about these in the [Variables](#) topic in the Macro section.

There are two different sets of variables.

The sets are:

- **UserVars()** - These are the variables stored as User variables.
- **GlobalVars()** - These are the variables stored as Global variables.

There are three functions regarding interaction with variables in a set:

- **GetVar()** - Gets the value from a specific variable in one of the sets.
- **SetVar()** - Sets a value in a specific variable in one of the sets. This also creates the variable if it does not exist.
- **DelVar()** - Deletes a specific variable in a set of variables.

Common for the three functions is a need to know the variable's name. The user and global variables can be listed using the **GetUserVariable** and **GetGlobalVariable** keywords combined with an asterisk wildcard.



39.5. Lua Functions - Object-Free API

Object-Free API means Lua functions that are not related to an object.

39.5.2. AddonVars(string)

Description

The **AddonVars** function returns a handle to the set of variables connected to a specific addon.



Restriction:

The addon variable set is not helpful at the moment.

Arguments

- **String:**
The string needs to be the name of the addon.

Return

- **Handle:**
The function returns a handle of the set of variables.

Example

This example prints information connected to the "Demo" addon variable set:

```
return function()  
  - Stores the handle to a variable set connected to the addon named 'Demo'.  
  local variableSet = AddonVars("Demo")  
  - Check if the return is nil and print an error message  
  if variableSet == nil then  
    ErrPrintf("The variable set does not exists")  
    return  
  end  
  Printf("===== START OF DUMP =====")  
  variableSet:Dump()  
  Printf("===== END OF DUMP =====")  
end
```



39.5.3. BuildDetails()

Description

The BuildDetails function returns a table with key-value pairs about the software build.

Arguments

This function does not accept any arguments.

Return

- **Build details:**

This is the table with key-value pairs. These are the possible keys in the table:

- **GitDate:** String with the date for the repository branch of the software.
- **GitHead:** String with the branch of the repository.
- **GitHash:** String with the hash for the repository.
- **CompileDate:** String with the date for the compile.
- **CompileTime:** String with the time for the compile.
- **BigVersion:** String indicating the software version.
- **SmallVersion:** String with the small version number of the software. Devices that only listen to a DMX data stream need to have this version to "understand" the streaming data.
- **HostType:** String with the host type, for instance, "Console" or "onPC".
- **HostSubType:** String with the host sub-type, for instance, "FullSize" or "Light".
- **CodeType:** String showing the type of code, for instance, "Release".
- **IsRelease:** Boolean indicating if the software is a release version.

Example

This example prints the content of the BuildDetails table:

```
local function main()
    local myBuild = BuildDetails()

    Printf("GitDate: " .. myBuild.GitDate)
    Printf("GitHead: " .. myBuild.GitHead)
    Printf("GitHash: " .. myBuild.GitHash)
    Printf("CompileDate: " .. myBuild.CompileDate)
    Printf("CompileTime: " .. myBuild.CompileTime)
    Printf("BigVersion: " .. myBuild.BigVersion)
    Printf("SmallVersion: " .. myBuild.SmallVersion)
    Printf("HostType: " .. myBuild.HostType)
    Printf("HostSubType: " .. myBuild.HostSubType)
    Printf("CodeType: " .. myBuild.CodeType)
    Printf("IsRelease: " .. tostring(myBuild.IsRelease))
```



end

return main

Related Lua Functions

- [Version\(\)](#)
- [HostType\(\)](#)
- [HostSubType\(\)](#)



39.5.6. ClassExists(string)

Description

The **ClassExists** Lua function returns a boolean indicating whether the provided string is a class.

Arguments

- **String:**
A string containing a single word that could be a class.

Return

- **Boolean:**
The function returns a boolean.
 - **True:**
The provided word is a class.
 - **False:**
The provided input is not a class.

Example

This example asks if the word "Display" is a class and returns proper feedback.

```
return function()  
  – Store a string with the class name  
  local className = "Display"  
  – Check if the class exists and then provide proper feedback  
  if ClassExists(className) then  
    Printf("The class '%s' exists", className)  
  else  
    Printf("The class '%s' does not exists", className)  
  end  
end  
end
```



39.5.7. CloseAllOverlays()

Description

The **CloseAllOverlays** function closes any pop-ups or menus (overlays) open on any screen.

Arguments

This function does not accept any arguments.

Return

This function does not return anything.

Example

This example simply closes any overlay.

```
return function()  
    CloseAllOverlays()  
end
```



39.5.8. CloseUndo(handle)

Description

The CloseUndo Lua function closes an open undo list. The function returns a boolean indicating if the function succeeds.

Undo lists needs to be created to be closed. See more about this in the [CreateUndo](#) function.

Arguments

- **Handle:**
The handle of a created undo list.

Return

- **Boolean:**
 - True / 1: The undo list was closed.
 - False / 0: The undo list is still in use and cannot be closed.

Example

This example creates an undo list, performs a series of commands that are added to the undo list, and closes the undo list. Now the series of commands can be oopsed with one oops command.

```
local function main()
  local MyNewUndo = CreateUndo("MySelection") -- Creation of the undo group.
  Cmd("ClearAll", MyNewUndo)
  Cmd("Fixture 1", MyNewUndo)
  Cmd("Fixture 2", MyNewUndo)
  Cmd("Fixture 5", MyNewUndo)
  Cmd("Fixture 7", MyNewUndo)
  local closeSuccess = CloseUndo(MyNewUndo) -- Closing the undo group and store it's return in a variable.

  Printf(closeSuccess) -- Print the feedback from the closing action - 1 = Success / 0 = Failure.
end

return main
```



39.5.9. Cmd(string[, handle])

Description

The CMD Lua function executes a command in the grandMA3 command line. It is executed in a Lua task - not the Main task (standard typed commands are run in the Main task). It is executed synchronously, and it blocks the Lua task while executing. This means that a bad command has the potential to block the system.

Alternative functions are [CmdIndirect\(\)](#) and [CmdIndirectWait\(\)](#).

Argument

- **String:**
A string with the command to be executed in the command line. Do not add a please or enter to execute the command.
- **Handle** (optional):
A handle to an undo (oops) list. Learn more in the [CreateUndo topic](#).

Return

- **String:**
A string is returned with the execution feedback known from the command line feedback
 - **OK:** Command executed.
 - **Syntax Error:** Command not executed because of syntax error.
 - **Illegal Command:** Command not executed because of some illegal command or action.

The returned string does not need to be used.

Example

This example executes the two commands "ClearAll" and "Fixture Thru At 0" in the command line.

```
local function main()
  Cmd("ClearAll")
  local CmdFeedback = Cmd("Fixture Thru At 0")
  Printf(CmdFeedback)
end

return main
```

39.5.10. CmdIndirect(string[, handle[, handle]])

Description

The CmdIndirect Lua function executes a command within the grandMA3 command line. It is executed asynchronously in the Main task. It does not block the Lua execution since it is not executed in the Lua Task.

Arguments

- **String:**
A string with the command to be executed in the command line. Do not add a please or enter to execute the command.
- **Handle** (optional):
A handle to an undo (oops) list. Learn more in the [CreateUndo topic](#).
- **Handle** (optional):
This is a handle for the target for the command. The target can be a specific screen. See the example below.

Return

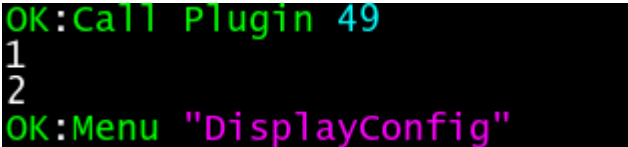
This function does not return anything.

Example

This example prints "1" and "2" in the Command Line History and let the main task open the [Configure Display pop-up](#) on screen 2.

```
local function main()  
  Printf("1")  
  CmdIndirect("Menu DisplayConfig", nil, GetDisplayByIndex(2))  
  Printf("2")  
end  
  
return main
```

The Command Line History shows:

A screenshot of a terminal window with a black background. The text is displayed in green and pink. The first line is "OK:Call Plugin 49" in green. The second line is "1" in green. The third line is "2" in green. The fourth line is "OK:Menu \"DisplayConfig\"" in pink.

```
OK:Call Plugin 49  
1  
2  
OK:Menu "DisplayConfig"
```




39.5.11. CmdIndirectWait(string[, handle[, handle]])

Description

The CmdIndirectWait Lua function executes a command within the grandMA3 command line. It does not block the Lua execution and is executed synchronously in the main task. Synchronous commands wait for the command to be executed before executing any following command.

Arguments

- **String:**
A string with the command to be executed in the command line. Do not add a please or enter to execute the command.
- **Handle (optional):**
A handle to an undo (oops) list. Learn more in the [CreateUndo](#) topic.
- **Handle (optional):**
This is a handle for the target for the command. The target can be a specific screen. See the example below.

Return

This function does not return anything.

Example

This example prints "1" and "2" in the Command Line History and let the main task open the [Configure Display pop-up](#) on screen 2.

```
local function main()  
  Printf("1")  
  CmdIndirectWait("Menu DisplayConfig", nil, GetDisplayByIndex(2))  
  Printf("2")  
end
```

return main

The Command Line History shows:

A screenshot of a terminal window with a black background. The text is displayed in green and pink. The first line is 'OK:Call Plugin 50' in green. Below it is a green '1'. The second line is 'OK:Menu "DisplayConfig"' in pink. Below it is a green '2'.



39.5.12. CmdObj()

Description

The CmdObj Lua function returns information about the command line object.

Argument

This function does not have any arguments.

Return

- **Handle:**
The function returns a handle to the command line object.

Example

This example uses the **Dump** function on the command object. It lists all the properties and lists the children:

```
local function main()

    CmdObj():Dump()

end

return main
```



39.5.15. ConfigTable()

Description

The **ConfigTable** Lua function returns a table with some configuration information. These are information only. The function does not have any actual functions. The table is not sorted.

Arguments

This function does not accept any arguments.

Return

- **Table:**
The returned table contains key value pairs with configuration information. See the example below.

Example

This example prints the content of the returned table.

```
return function ()  
    - Prints the content of the ConfigTable  
    for key,value in pairs(ConfigTable()) do  
        Printf(key .. " : " .. value)  
    end  
end
```



39.5.16. Confirm(string[, string[, integer[, boolean]]])

Description

The Confirm Lua function provides a simple confirmation pop-up for a true/false query. It is part of the user interface functions.

Arguments

- **String:**
This string is the title for the pop-up.
- **String** (optional):
This string is the text in the pop-up.
- **Integer** (optional):
This integer is not used since the pop-up appears on all screens. The value can be *nil*.
- **Boolean** (optional):
This boolean defines if there is a **Cancel** button in the pop-up or not.
 - true: There is a **Cancel** button in the pop-up. This is the default option used if it is not defined.
 - false: There is only an OK button in the pop-up.

Return

- **Boolean:**
 - True / 1: The pop-up was confirmed with the **OK**.
 - False / 0: The pop-up was not confirmed with **Cancel**. This is only a possible option if the **Cancel** button is visible.

Example

This example creates a confirmation pop-up with printed feedback in the Command Line History:

```
local function main()
  if Confirm("Confirm me", "Tap OK or Cancel", nil, true) then
    Printf("Pop-up result: OK")
  else
    Printf("Pop-up result: Cancel")
  end
end

return main
```



39.5.19. CreateUndo(string)

Description

The CreateUndo Lua function returns a handle to a list of commands and function calls grouped in the same oops action.

Functions can be executed with a reference to the undo handle. This adds the function to the undo list.

Undo lists needs to be closed using the **CloseUndo** function.

Arguments

- **String:**
A text string must be added. It can be used to identify the undo list.

Return

- **Handle:**
The function returns the handle to the undo list.

Example

This example creates an undo list, performs a series of commands being added to the undo list, and closes the undo list. Now the series of commands can be oopsed with one oops command.

```
local function main()
  local MyNewUndo = CreateUndo("MySelection") -- Creation of the undo group.
  Cmd("ClearAll", MyNewUndo)
  Cmd("Fixture 1", MyNewUndo)
  Cmd("Fixture 2", MyNewUndo)
  Cmd("Fixture 5", MyNewUndo)
  Cmd("Fixture 7", MyNewUndo)
  local closeSuccess = CloseUndo(MyNewUndo) -- Closing the undo group and store it's return in a variable.

  Printf(closeSuccess) -- Print the feedback from the closing action - 1 = Success / 0 = Failure.
end

return main
```



39.5.20. CurrentEnvironment()

Description

The **CurrentEnvironment** Lua function returns a handle to the current users' selected environment.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The returned handle to the environment.

Example

This example prints the data connected to the handle. It uses the **Dump()** function.

```
return function()  
  – Dumps information about the current environment  
  CurrentEnvironment():Dump()  
end
```



39.5.21. CurrentExecPage()

Description

The **CurrentEnvironment** Lua function returns a handle to the current users' selected executor page.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The returned handle to the executor page.

Example

This example prints the data connected to the handle. It uses the **Dump()** function.

```
return function()  
  – Dumps information about the current executor page  
  CurrentExecPage():Dump()  
end
```



39.5.22. CurrentProfile()

Description

The **CurrentProfile** Lua function returns a handle to the current users' profile.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The returned handle to the user profile.

Example

This example prints the data connected to the handle. It uses the **Dump()** function.

```
return function()  
  – Dumps information about the current executor page  
  CurrentProfile():Dump()  
end
```




39.5.23. CurrentUser()

Description

The **CurrentUser** Lua function returns a handle to the current user.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The returned handle to the user.

Example

This example prints the data connected to the handle. It uses the **Dump()** function.

```
return function()  
  – Dumps information about the current executor page  
  CurrentUser():Dump()  
end
```



39.5.24. DataPool()

Description

The DataPool Lua function references the currently selected DataPool and is used to read or edit properties within the data pool.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The function returns a handle to the DataPool object.

Example

This example uses the **Dump** function on the data pool object. Dump lists all the properties and lists the children. Finally the example also prints the name of the first sequence in the data pool.

```
local function main()

    DataPool():Dump()

    Printf("Name of sequence 1: " .. DataPool().Sequences[1].Name)

end

return main
```



39.5.25. DefaultDisplayPositions()

Description

DefaultDisplayPositions is an object-free function.

It returns the handle of the conventional default display positions, which contains the first seven screens as children.

For example, if the command line, view bar, and encoder/playback bar are displayed or not.

Argument

This function does not have any arguments.

Return

- **Handle:**
The function returns a handle to the command line object.

Examples

This example prints all the information about display 1 (child 1 of the default displays) using the **Dump** function:

```
local function main()
  DefaultDisplayPositions():Children()[1]:Dump()
end
```

```
return main
```

This example toggles the **Control Bar** for display 1 with the help of the DefaultDisplayPositions object:

```
local function main()
  local display1 = DefaultDisplayPositions():Children()[1]
  display1.ShowMainMenu = not display1.ShowMainMenu;
end
```

```
return main
```



39.5.26. DelVar(handle, string)

The DelVar Lua function deletes a specific variable in a set of variables. To learn more about the variables in plugins have a look at the [Variable Functions](#) topic.

Arguments

- **Handle:**
The handle of variable set.
- **String:**
The name of the variable. It needs to be in quotation marks.

Return

- **Boolean:**
 - True / 1: The variable was deleted.
 - False / 0: The variable was not deleted.

If the variable does not exist then false is also returned.

Example

This example deletes a variable called "myUserVar" in the set of user variables.

```
local function main()

    local success = DelVar(UserVars(), "myUserVar")

    if success then
        Printf("Variable is deleted.")
    else
        Printf("Variable is NOT deleted!")
    end

end

return main
```



39.5.27. DeskLocked()

Description

The function returns a boolean indicating if the station is locked.

Arguments

This function does not accept any arguments.

Return

- **Boolean:**
 - The boolean indicates if the station is desk locked or not.
 - **True** (or 1): The station is locked.
 - **False** (or 0): The station is not locked.

Example

This example prints the boolean number indicating the "DeskLocked" status to the **Command Line History**.

```
local function main()
  Printf(DeskLocked())      -The DeskLocked() return is printed.
end

return main
```



39.5.28. DeviceConfiguration()

Description

The **DeviceConfiguration** Lua function returns a handle to the DeviceConfiguration object.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The returned handle to the DeviceConfiguration.

Example

This example prints the data connected to the handle. It uses the **Dump()** function.

```
return function()  
  – This example dumps all information about the DeviceConfiguration object  
  Printf("===== START OF DUMP =====")  
  DeviceConfiguration():Dump()  
  Printf("===== END OF DUMP =====")  
end
```



39.5.31. DrawPointer(integer,table[,integer])

Description

The **DrawPointer** function draws a red pointer on the display. There can only be one pointer at a time on each station.

Arguments

- **Integer:**
This integer is the display index where the pointer should be drawn.
- **Table:**
This key-value table must have 'x' and 'y' keys with values indicating a position on the display. See the example below.
- **Integer (optional):**
This optional integer defines a duration for the pointer in milliseconds. It fades out. If a duration is not set, then it stays visible.

Return

This function does not return anything.

Example

This example draws a pointer on display 1 for 5 seconds:

```
return function()  
  -Set a display index  
  local displayIndex = 1  
  -Create and set the position in a table  
  local position = {}  
  position.x = 150  
  position.y = 25  
  -Set a 5 seconds duration - in milliseconds  
  local duration = 5000  
  -Draw the actual pointer  
  DrawPointer(displayIndex,position,duration)  
end
```



39.5.33. Echo(string)

Description

The Echo Lua function prints a string in the **System Monitor**.

Argument

- **String:**
The string text to be printed to the System Monitor.

Return

This function does not return anything.

Example

This example prints "Hello World!" on the System Monitor:

```
local function main()  
    Echo("Hello World!")  
end  
  
return main
```




39.5.34. ErrEcho(string)

Description

The ErrEcho Lua function prints a red error message on the **System Monitor**.

Argument

- **String:**
The string text to be printed to the System Monitor.

Return

This function does not return anything.

Example

This prints "This is a red error message!" on the system monitor:

```
local function main()  
    ErrEcho("This is a red error message!")  
end  
  
return main
```



39.5.35. ErrPrintf(string)

Description

The ErrPrintf Lua function prints a red error message in the **Command Line History** and **System Monitor**.

Argument

- **String:**
The string text to be printed to the Command Line History.

Return

This function does not return anything.

Example

This example prints "This is a red error message!" in the Command Line History and System Monitor:

```
local function main()  
    ErrPrintf("This is a red error message!")  
end  
  
return main
```



39.5.36. Export(filename, export_data)

Description

The object-free Export Lua function exports a Lua table in XML format.

This Lua function correlates with the **Import Lua function**.

There is a related object version of **Export**.

Arguments

- **Filename:**
This is a string containing the file name of the exported file. It should contain the file name, including the entire path. See the example below.
- **Export_data:**
This is the data that is going to be exported. It should be a table object.

Return

This function does not return anything.

Example

To export the build details table, create a plugin with this code:

```
local function main()
    local build = BuildDetails()                --BuildDetails() creates a table with information about the softw

    local exportPath = GetPath(Enums.PathType.Library) .. "/BuildDetails.xml" --The path and filename is stored in a va

    Export(exportPath, build)                  --The actual export function.

    Printf("The build details is exported to: " .. exportPath)        --Print some feedback.
end

return main
```



39.5.37. ExportCSV(filename, export_data)

Description

The object-free ExportCSV Lua function exports a Lua table in CSV format.

Arguments

- **Filename:**
This is a string containing the file name of the exported file. It should contain the file name, including the entire path. See the example below.
- **Export_data:**
This is the data that is going to be exported. It should be a table object.

Return

This function does not return anything.

Example

To export the build details table, create a plugin with this code:

```
local function main()
    local build = BuildDetails()                --BuildDetails() creates a table with information about the softw

    local exportPath = GetPath(Enums.PathType.Library) .. "/BuildDetails.csv" --The path and filename is stored in a var

    ExportCSV(exportPath, build)                --The actual export function.

    Printf("The build details is exported to: " .. exportPath)        --Print some feedback.
end

return main
```



39.5.38. ExportJson(filename, export_data)

Description

The object-free ExportJson Lua function exports a Lua table in JSON format.

Arguments

- **Filename:**
This is a string containing the file name of the exported file. It should contain the file name, including the entire path. See the example below.
- **Export_data:**
This is the data that is going to be exported. It should be a table object.

Return

This function does not return anything.

Example

To export the build details table, create a plugin with this code:

```
local function main()
    local build = BuildDetails()                --BuildDetails() creates a table with information about the softw

    local exportPath = GetPath(Enums.PathType.Library) .. "/BuildDetails.json" --The path and filename is stored in a va

    ExportJson(exportPath, build)              --The actual export function.

    Printf("The build details is exported to: " .. exportPath)        --Print some feedback.
end

return main
```



39.5.45. FromAddr(string[, handle])

The FromAddr Lua function converts a numbered string address into a handle that can be used in commands.

Arguments

- **String:**
A text string identifying an object. It can be a numbered or named address.
- **Handle (optional):**
The default is to write the address from the root location. This optional handle can specify a different base location. It still needs to be a base location in the address path from the root to the object.

Return

- **Handle:**
The handle for the addressed object.

Example

This example prints the address of the first sequence:

```
local function main()
    local mySequenceHandle = FromAddr("13.13.1.5.1")      – Converts the string to a handle and store in a variabel
    Printf("The address is: " ..mySequenceHandle:Addr())  – Converts the handle back to a numbered string and prints
    Printf("The address is: " ..mySequenceHandle:AddrNative()) – Converts the handle to a named string and prints it

    local myDataPool = DataPool() – Store the handle of the selected datapool
    Printf(myDataPool:Addr()) – Prints the address of the selected datapool

    – The follwoing example uses the name of a sequence in the sequence pool. Please adjust the "Default" name in the i
    local alsoMySequenceHandle = FromAddr("Sequences.Default", myDataPool) – Finds the address based on the base
    Printf("The address is: " ..alsoMySequenceHandle:Addr()) – Converts the handle back to a numbered string ar
    Printf("The address is: " ..alsoMySequenceHandle:AddrNative()) – Converts the handle to a named string and pri
end

return main
```



39.5.46. GetApiDescriptor()

Description

The **GetApiDescriptor** Lua function returns a table with a description of all the object-free Lua functions. These are descriptions only. The function does not have any actual functions. The table is not sorted.

Arguments

This function does not accept any arguments.

Return

- **Table:**
The returned table contains elements with other tables.
 - **String:**
This is the Api function name.
 - **String:**
This is the description of the Api arguments.
 - **String:**
This is the description of the Api returns.

Example

This example prints the content of the returned table.

```
return function ()  
  - This returns information about all the Lua "object-free" functions.  
  - GetApiDescriptor() returns a table with all the functions.  
  - Each table element is another table with the name, argument description, and return description.  
  for key,value in ipairs(GetApiDescriptor()) do  
    if value[1] ~= nil then  
      Printf("Api " .. key .. " is: " .. value[1])  
    end  
    if value[2] ~= nil then  
      Printf("Arguments: " .. value[2])  
    end  
    if value[3] ~= nil then  
      Printf("Returns: " .. value[3])  
    end  
    Printf("-----")  
  end  
end
```



39.5.47. GetAttributeByUIChannel(integer)

Description

The **GetAttributeByUIChannel** Lua function returns the handle to an attribute based on a "UI Channel Index". The index number can be found in the [Parameter List](#).

Arguments

- **Integer:**
The integer is the UI Channel index number.

Return

- **Handle:**
The handle to the attribute connected to the UI Channel.

Example

This example prints the "native" address to the first attribute of the first fixture in the current selection:

```
return function()  
  – Get a handle to the first fixture in the current selection  
  local fixtureIndex = SelectionFirst()  
  – Get the UI Channel Index number for the first attribute for the fixture  
  local channelIndex = GetUIChannelIndex(fixtureIndex,0)  
  – Print the native address for the attribute with the handle  
  Printf("The native addr for the attribute is: %s",GetAttributeByUIChannel(channelIndex):AddrNative())  
end
```




39.5.48. GetAttributeCount()

Description

The **GetAttributeCount** Lua function returns the total number of attribute definitions in the show.

Arguments

This function does not accept any arguments.

Return

- **Integer:**
The returned integer number represents the total amount of attribute definitions in the show file.

Example

This example prints the returned number in the Command Line History.

```
return function()  
  Printf("The attribute count is %i", GetAttributeCount())  
end
```



39.5.49. GetAttributeIndex(string)

Description

The **GetAttributeIndex** Lua function returns the (0 based) index number of the attribute definition based on the system name of the attribute.

Arguments

- **String:**
The string text of the attribute system name.

Return

- **Integer:**
The returned integer number represents the total amount of patched fixtures on all the stages in the show file.

Example

This example prints the index number of the attribute in Command Line History if it exists:

```
return function()  
  – store the returned index or nil of "Gobo1"  
  local attributeIndex = GetAttributeIndex("Gobo1")  
  – Check if the returned value is not nil and print a useful feedback  
  if attributeIndex~=nil then  
    Printf("Attribute is index number %i", attributeIndex)  
  else  
    Printf("The attribute is not found")  
  end  
end  
end
```



39.5.51. GetChannelFunction(integer, integer)

Description

The **GetChannelFunction** Lua function returns a handle to a channel function based on two index inputs.

Arguments

- **Integer:**
The first integer is a UI Channel Index. This can be found in the **Parameter List** or by the **GetUIChannelIndex()** Lua function.
- **Integer:**
This integer is an Attribute Index (0-based). This can be found in the **Attribute Definitions** or by the **GetAttributeIndex()** Lua function.

Return

- **Handle:**
The returned handle to the channel function.

Example

This example prints the data connected to the handle. It uses the **Dump()** function.

```
return function()
    – Select the first fixture in the current selection
    local subfixtureIndex = SelectionFirst()
    – End the function if there is no selection
    if subfixtureIndex == nil then
        ErrPrintf("Please select a fixture with a Dimmer")
        return
    end
    – Get the Attribute index and UIChannel index
    local attributeIndex = GetAttributeIndex("Dimmer")
    local uiChannelIndex = GetUIChannelIndex(subfixtureIndex,attributeIndex)
    Printf("The SubFixture Index is: %i. The Attribute Index is: %i. The UIChannel Index is: %i.",subfixtureIndex, attributeIndex)
    – End the function if any of the index return nil
    if (attributeIndex == nil or uiChannelIndex == nil) then
        ErrPrintf("Something wrong happened, maybe your first selected fixture don't have a Dimmer - Please try again")
        return
    end
    – The following prints the dump for the dimmer channel function.
    Printf("===== START OF DUMP =====")
    GetChannelFunction(uiChannelIndex,attributeIndex):Dump()
    Printf("===== END OF DUMP =====")
end
```



39.5.52. GetChannelFunctionIndex()

Description

The **GetChannelFunctionIndex** Lua function returns the integer matching a channel function based on two index inputs.

Arguments

- **Integer:**
The first integer is a UI Channel Index. This can be found in the [Parameter List](#) or by the [GetUIChannelIndex\(\)](#) Lua function.
- **Integer:**
This integer is an Attribute Index (0-based). This can be found in the [Attribute Definitions](#) or by the [GetAttributeIndex\(\)](#) Lua function.

Return

- **Integer:**
The returned integer to a channel function.

Example

This example prints the indexes based on the fixture selection and the "Dimmer" attribute.

```
return function()
  – Get the Attribute index and UIChannel index
  local attributeIndex = GetAttributeIndex("Dimmer")
  local uiChannelIndex = GetUIChannelIndex(SelectionFirst(),attributeIndex)
  – End the function if any of the index return nil
  if (attributeIndex == nil or uiChannelIndex == nil) then
    ErrPrintf("Something wrong happened, maybe your first selected fixture don't have a Dimmer - Please try again")
    return
  end
  – Get the Channel Function Index and store it in a variable
  local channelFunctionIndex = GetChannelFunctionIndex(uiChannelIndex,attributeIndex)
  Printf("The UIChannel Index is: %i. The Attribute Index is: %i. The Channel Function Index is: %i", uiChannelIndex, attri
end
```



39.5.53. GetClassDerivationLevel(string)

Description

The **GetClassDerivationLevel** Lua function returns an integer indicating the derivation level index for a class based on a class name.

Arguments

- **String:**
This string needs to be the name of a class.

Return

- **Integer:**
The returned integer indicates the class derivation level.

Example

This example prints the index integer for the Pool class in the Command Line History:

```
return function()  
  – Get the index integer for the "Pool" class  
  local classDerivationLevel = GetClassDerivationLevel("Pool")  
  – Create a valid Printf return  
  if classDerivationLevel == nil then  
    Printf("The return is nil")  
  else  
    Printf("The ClassDerivationLevel index for 'Pool' is: %i", classDerivationLevel)  
  end  
end
```

39.5.54. GetDisplayByIndex(integer)

Description

The **GetDisplayByIndex** Lua function returns a handle to the display object matching the provided index number.

Arguments

- **Integer:**
This function needs an index number for one of the displays.

Return

- **Handle:**
The returned handle to the display object.

Example

This example prints the data connected to the handle. It uses the **Dump()** function.

```
return function()
  - Get the index number for "Display 1"
  local displayIndex = GetDisplayCollect()["Display 1"].INDEX
  - Return an error text if the return should be nil
  if displayIndex == nil then
    ErrPrintf('Something went wrong. It appears that there is no "display 1"')
    return
  end
  - Dump all information about the display with the index number
  Printf("===== START OF DUMP =====")
  GetDisplayByIndex(displayIndex):Dump()
  Printf("===== END OF DUMP =====")
end
```



Hint:

The example uses the **GetDisplayCollect()** function to get the index number. The displays are children of the Display Collect, and this function can be used to access the same information using:

```
GetDisplayCollect()["Display 1"]
```



39.5.55. GetDisplayCollect()

Description

The **GetDisplayCollect** Lua function returns a handle to the DisplayCollect object.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The returned handle to the DisplayCollect .

Example

This example prints the data connected to the handle. It uses the **Dump()** function.

```
return function()  
  – This example dumps all information about the DisplayCollect object  
  Printf("===== START OF DUMP =====")  
  GetDisplayCollect():Dump()  
  Printf("===== END OF DUMP =====")  
end
```



39.5.56. GetDMXUniverse(integer[, boolean])

Description

The **GetDMXUniverse** Lua function returns a table with the DMX channels and their current value.

Arguments

- **Integer:**
The integer is the universe number. Valid range is 1 to 1024.
- **Boolean (optional):**
The boolean indicates if the returned value is in percent or DMX value.
 - True:
The returned value is in percent. The range is 0 to 100.
 - False:
The returned value is in DMX value. The range is 0 to 255.

Return

- **Table:**
The returned table lists all the DMX addresses and the corresponding value.

– OR –

- **Nil:**
Nil is returned if the universe is not granted or the input value is out of range.

Example

This example prints the table in a list for DMX universe 1 (if it is granted):

```
return function()  
  – This gets a table for universe 1 with the returned value in percent  
  local tableDMXUniverse = GetDMXUniverse(1,true)  
  
  – Check the returned table and print information if nil  
  if tableDMXUniverse == nil then  
    Printf("No value is returned. The univers is not granted or input is out of range")  
    return  
  end  
  
  – Prints the table if not nil  
  for addr, value in ipairs(tableDMXUniverse) do  
    Printf("DMX Addr: %i - DMX value : %i", addr, value)  
  end  
end
```




39.5.57. GetDMXValue(integer[, integer, boolean])

Description

The **GetDMXValue** Lua function returns a number indicating the DMX value of a specified DMX address.

Arguments

- **Integer:**
The integer is the DMX address.
- **Integer (optional):**
The integer is the universe number.
- **Boolean (optional):**
The boolean indicates if the returned value is in percent or DMX value.
 - True:
The returned value is in percent. The range is 0 to 100.
 - False:
The returned value is in DMX value. The range is 0 to 255.

Return

- **Integer:**
The returned integer value corresponds with the value of the selected DMX address.

Example

This example prints the value for DMX address 1 in Universe 4:

```
return function()  
  – This prints the value of DMX address 1 in universe 4 in a range of 0 to 255  
  local universe = 4   – The DMX universe  
  local address = 1   – The DMX address  
  local percent = false – Readout in percent or DMX value  
  local value = GetDMXValue(address, universe, percent)  
  Printf("DMX address %i.%03d is %03d", universe, address, value)  
end
```



39.5.59. GetExecutor(integer)

Description

The **GetExecutor** Lua function returns the handles of the executor and the page based on the executor number.

Arguments

- **Integer:**
The integer number for the executor.

Return

- **Handle - Executor:**
The returned handle to the executor.
- **Handle - Page:**
The returned handle to the page.

Example

This example stores the handles for executor number 201. It then dumps the data for the two handles.

```
return function ()  
    – This saves the handles for executor 201 on the selected page  
    local executorHandle, pageHandle = GetExecutor(201)  
    – The following prints the dumps of the two handles  
    Printf("This is the dump of the executor:")  
    Printf("————DUMP START————")  
    executorHandle:Dump()  
    Printf("————END————")  
    Printf("This is the dump of the page:")  
    Printf("————DUMP START————")  
    pageHandle:Dump()  
    Printf("————END————")  
end
```



39.5.60. GetFocus()

Description

The **GetFocus** Lua function returns a handle to the object who current got focus in the UI.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The returned handle to the object.

Example

This example prints the data connected to the handle. It uses the **Dump()** function.

```
return function()  
  – This example dumps all information about the object who currently got focus  
  Printf("===== START OF DUMP =====")  
  GetFocus():Dump()  
  Printf("===== END OF DUMP =====")  
end
```



39.5.61. GetFocusDisplay()

Description

The **GetFocusDisplay** Lua function returns a handle to the display object who current got focus in the UI.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The returned handle to the display object.

Example

This example prints the data connected to the handle. It uses the **Dump()** function.

```
return function()  
  – This example dumps all information about the display object who currently got focus  
  Printf("===== START OF DUMP =====")  
  GetFocusDisplay():Dump()  
  Printf("===== END OF DUMP =====")  
end
```



39.5.62. GetObjApiDescriptor()

Description

The **GetObjApiDescriptor** Lua function returns a table with a description of all the object Lua functions. These are descriptions only. The function does not have any actual functions. The table is not sorted.

Arguments

This function does not accept any arguments.

Return

- **Table:**
The returned table contains elements with other tables.
 - **String:**
This is the Api function name.
 - **String:**
This is the description of the Api arguments.
 - **String:**
This is the description of the Api returns.

Example

This example prints the content of the returned table.

```
return function ()  
  - This returns information about all the Lua "object" functions.  
  - GetObjApiDescriptor() returns a table with all the functions.  
  - Each table element is another table with the name, argument description, and return description.  
  for key,value in ipairs(GetObjApiDescriptor()) do  
    if value[1] ~= nil then  
      Printf("Api " .. key .. " is: " .. value[1])  
    end  
    if value[2] ~= nil then  
      Printf("Arguments: " .. value[2])  
    end  
    if value[3] ~= nil then  
      Printf("Returns: " .. value[3])  
    end  
    Printf("-----")  
  end  
end
```



39.5.63. GetPath(string[, boolean] | integer)

Description

The GetPath Lua function delivers a string with the path of a grandMA3 folder.

The function has two possible argument types - use one of them with each function call.

Argument

- **String:**
A text string with the folder name.
- **Boolean** (optional with string):
If this boolean is true, then the folder at the path is created if it does not exist.

- OR -

- **Integer:**
An integer identifying an index in the "Enum.PathType" table.



Restriction:

Folder creation only works with string arguments.

Return

- **String:**
The returned string is the *first found* full path related to the provided argument.

Example

This example prints the paths of the show folder on the system monitor twice. It demonstrates the two different input types:

```
local function main()
```

```
    Printf("Path of show files (string) is: " .. GetPath("shows", true))  
    Printf("Path of show files (integer) is: " .. GetPath(Enums.PathType.Showfiles))
```

```
end
```

```
return main
```



39.5.65. GetPathSeparator()

Description

The **GetPathSeparator** function returns a string with the path separator for the operating system.

Arguments

This function does not accept any arguments.

Return

- **String:**
The string is a single character indicating the path separator based on the operating system.

Example

This example prints the path separator.

```
return function()  
  -- This prints the path separator. It is different between a Linux and macOS (/) and a Windows (\) operating system.  
  Printf("The path separator is " .. GetPathSeparator())  
end
```

39.5.71. GetRTChannel(integer)

Description

The **GetRTChannel** Lua function returns a table with information about the related RT Channel.

Arguments

- **Integer:**
The integer should be the index number for an RT Channel.

Return

- **Table:**
The returned table contains related numbers, tables, and handles with a named identifying key:
 - handle "subfixture"
 - handle "dmx_channel"
 - integer "dmx_default"
 - integer "dmx_highlight"
 - integer "ui_index_first"
 - handle "fixture"
 - table "info"
 - integer "dmx_lowlight"
 - integer "freq"
 - integer "rt_index"
 - table "patch"

Example

This example prints all information related to the first RT Channel for the first fixture in the selection.

```
return function()  
  - Get the index number for the first RT Channel for the first fixture in the current selection  
  local channelRTIndex = GetRTChannels(SelectionFirst())[1]  
  - Print an error message if returned index is nil  
  if channelRTIndex == nil then  
    ErrPrintf("Please select a fixture and try again")  
    return  
  end  
  - Print all information about the RT Channel  
  local rtChannel = GetRTChannel(channelRTIndex)  
  Printf("===== RT CHANNEL =====")  
  Printf("ui_index_first = " .. rtChannel["ui_index_first"])  
  Printf("dmx_lowlight = " .. rtChannel["dmx_lowlight"])  
  Printf("dmx_highlight = " .. rtChannel["dmx_highlight"])  
  Printf("dmx_default = " .. rtChannel["dmx_default"])
```




```
Printf("freq = " .. rtChannel["freq"])
Printf("rt_index = " .. rtChannel["rt_index"])
Printf("=====  
rtChannel["dmx_channel"]:Dump() – Handle for relevant DMX channel
Printf("=====  
rtChannel["fixture"]:Dump() – Handle for relevant fixture
Printf("=====  
rtChannel["subfixture"]:Dump() – Handle for relevant subfixture
Printf("=====  
Printf("normed Phaser_time = " .. rtChannel["info"]["normed Phaser_time"])
Printf("=====  
Printf("group_master = " .. rtChannel["info"]["flags"]["group_master"])
Printf("additive_master = " .. rtChannel["info"]["flags"]["additive_master"])
Printf("solo = " .. rtChannel["info"]["flags"]["solo"])
Printf("highlight = " .. rtChannel["info"]["flags"]["highlight"])
Printf("lowlight = " .. rtChannel["info"]["flags"]["lowlight"])
Printf("=====  
Printf("break = " .. rtChannel["patch"]["break"])
Printf("coarse = " .. rtChannel["patch"]["coarse"])
Printf("fine = " .. rtChannel["patch"]["fine"])
Printf("ultra = " .. rtChannel["patch"]["ultra"])
end
```



39.5.72. GetRTChannelCount()

Description

The **GetRTChannelCount** Lua function returns a number indicating the total amount of RT channels.

Arguments

This function does not accept any arguments.

Return

- **Integer:**
The function returns an integer number depicting the total amount of RT channels.

Example

This example prints the number of RT channels to the Command Line History:

```
return function()  
  Printf("The number of RT channels is " .. GetRTChannelCount())  
end
```



39.5.73. GetRTChannels(integer[,boolean] OR handle[,boolean])

Description

The **GetRTChannels** Lua function returns a table with RT Channel indexes or a table with handles to the RT Channel objects. There are two different types of arguments for this function.

Arguments

- **Integer:**
The integer should be the index number for a (sub)fixture.
- **Boolean** (Optional):
 - **True:**
The returned table contains handles for RT Channel objects.
 - **False** (default):
The returned table contains integers index values to the RT Channel objects.

- OR -

- **Handle:**
The handle should relate to a (sub)fixture object.
- **Boolean** (Optional):
 - **True:**
The returned table contains handles for RT Channel objects.
 - **False** (default):
The returned table contains integers index values to the RT Channel objects.

Return

- **Table:**
The returned table can be a list of RT Channel indexes or handles to the same RT Channels.

Examples

Example 1

This example prints a list of RT Channel indexes for the first fixture in the selection. It uses an index number as input.

```
return function()  
  - Get the index number for the first fixture in the current selection  
  local fixtureIndex = SelectionFirst()  
  - Get the indexes of the RT channels  
  local rtChannels = GetRTChannels(fixtureIndex, false)  
  - Print an error message if returned table is nil  
  if rtChannels == nil then  
    ErrPrintf("Please select a fixture and try again")  
  return
```



```
end
– Print the table content
for key,value in ipairs(rtChannels) do
    Printf("List index number ".. key .. " : RTChannel index number = ".. value)
end
end
```

Example 2

This example prints a list of RT Channel indexes and attributes for the first fixture in the selection. It uses a handle as the input.

```
return function()
    – Get a handle to the first fixture in the current selection
    local fixtureHandle = GetSubfixture(SelectionFirst())
    if fixtureHandle == nil then
        ErrPrintf("Please select a fixture and try again")
        return
    end
    – Creates a table of handles to the RT channels of the first selected fixture.
    local rtChannels = GetRTChannels(fixtureHandle, true)
    if rtChannels == nil then
        ErrPrintf("Please select a fixture and try again")
        return
    end
    – Print DMX addresses of the RT Channels for the fixture
    for key,value in ipairs(rtChannels) do
        Printf("List index number ".. key .. " : RTChannel Index = %i, Coarse DMX addr. = %s, Fine DMX addr. = %s", value.INC
    end
end
```

39.5.75. GetSample(string)

Description

The **GetSample** Lua function returns a number representing a percentage usage based on a string input.

Arguments

- **String:**
Only a specific list of strings can be input:
 - MEMORY
 - CPU
 - CPUTEMP
 - GPUTEMP
 - SYSTEMP
 - FANRPM

Return

- **Number:**
A number (float) is returned.

Example

This example stores the different samples in a table and then prints the content of the table.

```
return function()  
  – Gather the sample information in a table  
  local sample = {}  
  sample["MEMORY"] = GetSample("MEMORY")  
  sample["CPU"] = GetSample("CPU")  
  sample["CPUTEMP"] = GetSample("CPUTEMP")  
  sample["GPUTEMP"] = GetSample("GPUTEMP")  
  sample["SYSTEMP"] = GetSample("SYSTEMP")  
  sample["FANRPM"] = GetSample("FANRPM")  
  – Print the collected data  
  Printf("Memory ;".. sample["MEMORY"])  
  Printf("CPU ;".. sample["CPU"])  
  Printf("CPU temperature ;".. sample["CPUTEMP"])  
  Printf("GPU temperature ;".. sample["GPUTEMP"])  
  Printf("System temperature ;".. sample["SYSTEMP"])  
  Printf("Fan RPM ;".. sample["FANRPM"])  
end
```



39.5.76. GetSelectedAttribute()

Description

The **GetSelectedAttribute** Lua function returns a handle to the currently selected attribute.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The returned handle to the selected attribute.

Example

This example prints the data connected to the handle. It uses the **Dump()** function.

```
return function()  
  – This example dumps all information about the currently selected attribute  
  Printf("===== START OF DUMP =====")  
  GetSelectedAttribute():Dump()  
  Printf("===== END OF DUMP =====")  
end
```



39.5.77. GetShowFileStatus()

Description

The GetShowFileStatus Lua function returns a string with the current device's show file status, for example, "NoShow", "ShowLoaded", "ShowDownloaded", "ShowSaving", and "DataNegotiationActive".

Arguments

This function does not accept any arguments.

Return

- **String:**
The returned string is the enum string from "Enums.ShowFileStatus" that matches the current status.

Example

This example prints the current device's show file status in the Command Line History:

```
local function main()
  Printf("ShowfileStatus: "..GetShowFileStatus())
end

return main
```



39.5.78. GetSubfixture(integer)

Description

The **GetSubfixture** Lua function returns the handle of the fixture specified by its patch index number.

Arguments

- **Integer:**
The patch index number for a fixture. This is also known as the "subfixtureindex".

Return

- **Handle:**
The function returns a handle to the fixture object matching the provided index number.

Example

This example uses a fixture selection to print all the information (in the Command Line History) about the first fixture in the selection using the **Dump()** function:

```
return function ()  
  – Check for a fixture selection, by returning an index for the first fixture  
  if (SelectionFirst()) then  
    – There is a fixture selection, store the index for the first fixture  
    local fixtureIndex = SelectionFirst()  
    – Dump all information about the fixture  
    GetSubfixture(fixtureIndex):Dump()  
  else  
    – There needs to be a selection of at least one fixture  
    Printf('Please select a fixture')  
  end  
end  
end
```

Related Functions

- **SelectionFirst**
- **SelectionNext**
- **GetSubfixtureCount**



39.5.79. GetSubfixtureCount()

Description

The **GetSubfixtureCount** Lua function returns the total number of fixtures that are patched within the show file.

Arguments

This function does not accept any arguments.

Return

- **Integer:**
The returned integer number represents the total amount of patched fixtures on all the stages in the show file.

Example

This example prints the total number of patched fixtures in the Command Line History:

```
return function ()  
    Printf("Total number of patched fixtures: %i", GetSubfixtureCount())  
end
```



39.5.80. GetTokenName(string)

Description

The **GetTokenName** Lua function returns a string with the full keyword based on the short version string input or nil if there is no corresponding keyword.

Arguments

- **String:**
The string input should correspond to a short version of a keyword.

Return

- **String:**
A string with the full keyword is returned.
- OR -
- **Nil:**
If there is no corresponding keyword, then nil is returned.

Example

This example returns the full keyword matching the short "seq" string.

```
return function()  
  – Store a short string to be used as input  
  local shortToken = 'seq'  
  – Get the full token name  
  local tokenName = GetTokenName(shortToken)  
  – Print useful output if nil is not returned  
  if tokenName ~= nil then  
    Printf("The full version of ".. shortToken .. " is " .. tokenName .. """)  
  end  
end
```



39.5.81. GetTokenNameByIndex(int)

Description

The **GetTokenNameByIndex** Lua function returns a string with the keyword based on the index number provided.

Each keyword is described in the [Command Syntax and Keywords section](#).

Arguments

- **Integer:**
The integer input is the index number for a corresponding keyword. There is no apparent logic to the index number and the keyword.

Return

- **String:**
A string with the full keyword is returned.
- OR -
- **Nil:**
If there is no corresponding keyword, then nil is returned.

Example

If the keyword exists, this example returns the keywords matching the first 443 index numbers.

```
return function()  
  – Create a variable to hold the keyword string  
  local tokenName = ""  
  – Print the keywords to the first 443 indexes if possible  
  for index = 1, 443, 1 do  
    tokenName = GetTokenNameByIndex(index)  
    if tokenName ~= nil then  
      Printf("Token index " .. index .. " = " .. tokenName)  
    end  
  end  
end
```



39.5.85. GetUIChannelCount()

Description

The **GetUIChannelCount** Lua function returns a number indicating the total amount of UI channels.

Arguments

This function does not accept any arguments.

Return

- **Integer:**
The function returns an integer number depicting the total amount of UI channels.

Example

This example prints the amount of UI channels to the Command Line History:

```
return function()  
  Printf("The number of UI channels is " .. GetUIChannelCount())  
end
```



39.5.86. GetUIChannelIndex(integer, integer)

Description

The **GetUIChannelIndex** Lua function returns the index integer matching a UI channel based on two index inputs.

Arguments

- **Integer:**
The first integer is the patch index of a fixture.
- **Integer:**
This integer is an attribute index (0-based). This can be found in the **Attribute Definitions** or by the **GetAttributeIndex()** Lua function.

Return

- **Integer:**
The returned integer to a channel function.

Example

This example prints the UI channel index of the "Dimmer" attribute of the first fixture in the current selection.

```
return function()
  -- Get the Attribute index and UIChannel indexes
  local attributeIndex = GetAttributeIndex("Dimmer")
  local uiChannelIndex = GetUIChannelIndex(SelectionFirst(),attributeIndex)
  -- End the function if any of the index return nil
  if (attributeIndex == nil or uiChannelIndex == nil) then
    ErrPrintf("Something went wrong, maybe your first selected fixture don't have a Dimmer - Please try again")
    return
  end
  Printf("The UI Channel Index is " .. uiChannelIndex)
end
```



39.5.87. GetUIChannels(integer[,boolean] OR handle[,boolean])

Description

The **GetUIChannels** Lua function returns a table with UI Channel indexes or a table with handles to the UI Channel objects. There are two different types of arguments for this function.

Arguments

- **Integer:**
The integer should be the index number for a (sub)fixture.
- **Boolean** (Optional):
 - **True:**
The returned table contains handles for UI Channel objects.
 - **False** (default):
The returned table contains integer index values to the UI Channel objects.

- OR -

- **Handle:**
The handle should relate to a (sub)fixture object.
- **Boolean** (Optional):
 - **True:**
The returned table contains handles for UI Channel objects.
 - **False** (default):
The returned table contains integer index values to the UI Channel objects.

Return

- **Table:**
The returned table can be a list of UI Channel indexes or handles to the same UI Channel indexes.

Examples

Example 1

This example prints a list of UI Channel indexes for the first fixture in the selection. It uses an index number as input.

```
return function()  
    – Creates a table of indexes of the UI channels of the first selected fixture.  
    local uiChannels = GetUIChannels(SelectionFirst())  
    if uiChannels == nil then  
        ErrPrintf("Please select a fixture and try again")  
        return  
    end  
    for key,value in pairs(uiChannels) do  
        Printf(key .. " : UIChannel Index = " .. value)
```



```
    end  
end
```

Example 2

This example prints a list of UI Channel indexes and attributes for the first fixture in the selection. It uses a handle as the input.

```
return function()  
    local fixtureHandle = GetSubfixture(SelectionFirst())  
    – Creates a table of handles to the UI channels of the first selected fixture.  
    local uiChannels = GetUIChannels(fixtureHandle, true)  
    if uiChannels == nil then  
        ErrPrintf("Please select a fixture and try again")  
        return  
    end  
    for key,value in pairs(uiChannels) do  
        Printf(key .. ": UIChannel Index = %i, (Sub)Attribute = %s", value.INDEX-1, value.SUBATTRIBUTE)  
    end  
end
```



39.5.89. GetVar(handle, string)

The GetVar Lua function returns the value of a specific variable in a set of variables. To learn more about the variables in plugins, look at the [Variable Functions](#) topic.

Arguments

- **Handle:**
The handle of variable set.
- **String:**
The name of the variable. It needs to be in quotation marks.

Return

- **Value:**
This is the value of the variable.

If the variable does not exist, then nil is returned.

Example

This example returns the value of a variable called "myUserVar" in the set of user variables if it exists.

```
local function main()

    local varValue = GetVar(UserVars(), "myUserVar")

    if varValue == nil then
        Printf("Variable returns nothing!")
    else
        Printf("Variable value is: " .. varValue)
    end

end

return main
```




39.5.90. GlobalVars()

The GlobalVars function returns a handle to the set of global variables. Read more about these in the [Variables](#) topic in the Macro section.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The function returns a handle of the set of global variables.

Example

This example sets, gets, and deletes a global variable:

```
local function main()

    -- Stores a local Lua variable with the handle for the global variables.
    local variableSection = GlobalVars()

    -- Sets a global variable with an integer value using the SetVar function.
    SetVar(variableSection, "myGlobalVar", 42)

    -- Prints the global variable using the GetVar function.
    Printf("The value of myGlobalVar is: " .. GetVar(variableSection, "myGlobalVar"))

    -- Deletes the global variable using the DelVar function.
    DelVar(variableSection, "myGlobalVar")

end

return main
```



39.5.91. HandleToInt(handle)

The object-free HandleToInt Lua function converts a handle into an integer format.

See the [Handle topic](#) for more info regarding handles and links to other related functions.

Arguments

- **Handle:**
The handle of the object.

Return

- **Integer:**
The returned integer is the handle converted to an integer.

Example

This example prints the handle integer number for the selected sequence. It also converts the integer back to a handle and uses this to print the name of the sequence:

```
local function main()
    local mySequence = SelectedSequence()    – Store the handle of the selected sequence
    local mySeqInt = HandleToInt(mySequence) – Store an integer with the handle converted to integer

    Printf("The handle for the selected sequence (integer): " .. mySeqInt)

    Printf("The name of the selected sequence is: " .. IntToHandle(mySeqInt).name)

end

return main
```



39.5.92. HandleToStr(handle)

The object-free HandleToStr Lua function converts a handle into a string in a hexadecimal number format.

See the [Handle topic](#) for more info regarding handles and links to other related functions.

Arguments

- **Handle:**
The handle of the object.

Return

- **String:**
The returned string is the handle number converted to a hexadecimal format.

Example

This example prints the handle hex number for the selected sequence. It also converts the string back to a handle and uses this to print the name of the sequence:

```
local function main()
    local mySequence = SelectedSequence()    – Store the handle of the selected sequence
    local mySeqStr = HandleToStr(mySequence) – Store a string with the handle converted to hex

    Printf("The handle for the selected sequence (string): " .. mySeqStr)

    Printf("The name of the selected sequence is: " .. StrToHandle(mySeqStr).name)
end

return main
```



39.5.93. HookObjectChange

Description

The HookObjectChange Lua function is used to automatically call a function when an grandMA3 object changes.

Syntax

**HookObjectChange(function:callback, light_userdata:handle, light_userdata:plugin_handle[, light_userdata:target]):
nothing**

Returns: nothing

Example

To call a function every time the content of the sequence pool changes, create a plugin with this code:

```
local pluginName = select(1,...);
local componentName = select(2,...);
local signalTable = select(3,...);
local my_handle = select(4,...);

local function main()
    local hookObject = DataPool().Sequences;
    local pluginHandle = my_handle.Parent();
    local callbackFunction = function(obj)
        Printf(tostring(obj.name).. " changed!")
    end
    HookObjectChange(callbackFunction, hookObject, pluginHandle)
end

return main
```



39.5.94. HostOS()

Description

The HostOS Lua function returns a string with the type of operating system of the device where the plugin is executed (for instance, "Windows", "Linux", or "Mac").

Arguments

This function does not accept any arguments.

Return

- **String:**
The returned string is the operating system of the grandMA3 hardware or grandMA3 onPC computer.

Example

This example prints the operating system of the device in the Command Line History:

```
local function main()
  Printf("HostOS: " .. HostOS())
end

return main
```



39.5.95. HostSubType()

Description

The HostSubType Lua function returns a string with the host sub type of the station where the plugin is executed (for example, "FullSize", "Light", "RPU", "onPCRRackUnit", "Undefined").

Arguments

This function does not accept any arguments.

Return

- **String:**
The returned string is the host sub-type of the device.

Example

This example prints the host sub-type of the station in the Command Line History:

```
local function main()
  Printf("HostSubType: " .. HostSubType())
end

return main
```



39.5.96. HostType()

Description

The HostType Lua function returns a string with the host type of the device where the plugin is executed (for example, "Console" or "onPC").

Arguments

This function does not accept any arguments.

Return

- **String:**
The returned string is the host type of the device.

Example

This example prints the host type of the device in the Command Line History:

```
local function main()
  Printf("HostType: "..HostType())
end

return main
```

39.5.97. Import

Description

The Import Lua function is used to import a Lua table in XML format.

This Lua function correlates with the [Export Lua function](#).

Syntax

Import(string:filename): table:content

Returns: nothing

Example

To import the table "mySettings.xml", create a plugin with this code:

```
local function main()
    local mySettings = {
        count = 10,
        indices = {1, 2, 3, 4},
        names = "SomeString"
    }

    local importPath = GetPath(Enums.PathType.Library) .. "/mySettings.xml"
    local importedTable = Import(importPath)
    Printf("Imported table from " .. importPath)
    Printf("Imported count = " .. importedTable.count)
    Printf("Imported names = " .. importedTable.names)
    for index in pairs(importedTable.indices) do
        Printf("Index = " .. index)
    end
end

return main
```

The command line history shows:



```
OK:Call Plugin 4
Imported table from C:/ProgramData/MALightingTechnology/gma3_library/mySettings.xml
Imported count = 10
Imported names = somestring
Index = 1
Index = 2
Index = 3
Index = 4
OK:Menu "commandLineHistory"
MA Admin[Fixture]>
```






39.5.98. IncProgress(handle, integer)

Description

The IncProgress Lua function changes the value on the range for a progress bar using an integer input. A handle input argument defines the progress bar. The progress bar needs have been created using the [StartProgress](#) function.

See the [ProgressBar topic](#) for more info regarding progress bars and links to other related functions.

Arguments

- **Handle:**
The handle for the progress bar.
- **Integer:**
The desired value for the range. This can be a negative value to decrease the value.

Return

This function does not return anything.

Examples

These two examples increase and decrease the range value for the progress bar created using the example in the StartProgress topic (link above):

```
return function()  
  – Increase the current value for a progress bar with the matching handle  
  IncProgress(progressHandle, 1)  
end
```

```
return function()  
  – Decrease the current value for a progress bar with the matching handle  
  IncProgress(progressHandle, -1)  
end
```



39.5.99. IntToHandle(integer)

The object-free IntToHandle Lua function converts an integer number into a handle. The integer needs to correlate with an actual handle.

See the [Handle topic](#) for more info regarding handles and links to other related functions.

Arguments

- **Integer:**
The integer that correlates to an object's handle.

Return

- **Handle:**
The returned handle of the object correlates with the integer.

Example

This example prints the handle integer number for the selected sequence. It also converts the integer back to a handle and uses this to print the name of the sequence:

```
local function main()
  local mySequence = SelectedSequence()    -- Store the handle to the selected sequence
  local mySeqInt = HandleToInt(mySequence) -- Store an integer with the handle converted to integer

  Printf("The handle for the selected sequence (integer): " .. mySeqInt)

  Printf("The name of the selected sequence is: " .. IntToHandle(mySeqInt).name)

end

return main
```



39.5.101. IsObjectValid(handle)

Description

The **IsObjectValid** function returns a boolean true or nil depending on whether the supplied argument is a valid object.

Arguments

- **Handle:**
The argument should be the handle to a possible object.

Return

- **Boolean or nil:**
The returned value is a boolean True if the handle is a valid object or it returns nil if it is not a valid object.

Example

This example below examines if "Root()" is a valid object and prints meaningful feedback:

```
return function()  
  -Create a variable with the possible object  
  local myObject = Root()  
  -Check if it is an object  
  local myReturn = IsObjectValid(myObject)  
  -Print the result  
  if myReturn == nil then  
    ErrPrintf("It is not a valid object")  
  else  
    Printf("It is an object")  
  end  
end  
end
```



39.5.103. KeyboardObj()

Description

The **KeyboardObj** function returns the handle to the first found keyboard object.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The function returns the handle to the keyboard object.

Example

This example prints the information of the keyboard object:

```
return function()  
  - Print all informatin about the KeyboardObj object  
  Printf("===== START OF DUMP =====")  
  KeyboardObj():Dump()  
  Printf("===== END OF DUMP =====")  
end
```



39.5.105. MasterPool()

Description

The MasterPool function returns the handle to the masters.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The function returns the handle to the pool of masters.

Example

This example prints the information of the MasterPool object:

```
return function()  
  - Print all informatin about the MasterPool object  
  Printf("===== START OF DUMP =====")  
  MasterPool():Dump()  
  Printf("===== END OF DUMP =====")  
end
```



39.5.106. MessageBox(table)

Description

The MessageBox Lua function is used to create pop-up message boxes. These can be simple or complex information pop-ups with many different options and user inputs.

The message box contains multiple elements that must be defined in a table. This table is the single input argument to the function.

The elements in the message box are displayed in alphabetical order.

This function is part of the User Interface functions.

Arguments

- **Table:**

The input to the function must be formatted as a table using key-value pairs. The needed elements have default values that will be used if not overwritten. The values can be defined in any order if the key is defined. The table can have the following elements:

- **title:**

This is the title of the pop-up message box.

- **titleTextColor:**

This is the text color for the title text. The value is a number or string that refers to the **UI Colors** in the color theme. See the link below.

- **backColor:**

This is the color of the frame or border of the pop-up. The value is a number or string that refers to the **UI Colors** in the color theme. See the link below.

- **icon:**

This is an icon that can be shown in the upper left corner of the pop-up. The value can be an integer or a string that refers to the number or name of a texture image (without the file format).

The icons can be listed by navigating to the texture folder [ChangeDestination GraphicsRoot/TextureCollect/Textures](#) and then doing a [List](#) command.

This shows a long list with numbers and names of all the textures in the Command Line History window.

- **message:**

This message text string is displayed in the main part of the pop-up. A new line can be created by adding a "\n" in the text.

- **messageTextColor:**

This is the text color for the message text. The value is a number or string that refers to the **UI Colors** in the color theme. See the link below.

- **timeout:**

The timeout value is an integer that indicates how long the message box is displayed in milliseconds. It will show a countdown timer at the top of the message area. When the countdown runs out, it will close the pop-up.

When the timeout closes the pop-up, it returns a **success** element with a **true** value (see more about the return below). It was closed "normally".



- **timeoutResultCancel:**

This element can change the return to mimic a cancel of the pop-up, which returns a **false** instead of a **true** for the **success** element.
- **timeoutResultID:**

When the timeout closes the pop-up with the **success** value = **true**, a special return result can be defined using this, for instance, setting the value to 99 - then 99 is returned when the message box is closed by the timeout.
- **commands:**

The commands are buttons at the bottom of the message box pop-up. The input here is a table of objects using the following structure:

 - **value:**

This integer value will be returned as the result value - see more about the return below.
 - **name:**

This is a string which will be shown on the button.
- **inputs:**

The inputs are user input fields where text or numbers can be input. The input fields will be displayed in alphabetical order.

The input fields are defined using an table with the following structure (see example 5 below):


 - **name:**

This is a string value - the text will be shown as a label for the input field.
 - **value:**

This is a string value - it is a default input value for the input field.
 - **blackfilter:**

This is a string value - it defines input characters that are not allowed.
 - **whitefilter:**

This is a string value - it defines which input characters are allowed.
 - **vkPlugin:**

This is a string value - it is the name of the input pop-up, which is opened if the on-screen keyboard icon () is tapped in the pop-up. Example 5 below has a list of possible vkPlugin values.
 - **maxTextLength:**

This is an integer value - it defines the maximum number of characters for the input.
- **states:**

The states are buttons in the pop-up. State buttons have a small checkbox and can have a true or false state. The buttons will be displayed in alphabetical order.

The buttons are defined using a table with the following structure (see example 4 below):

 - **name:**

This is a string value - the text will be shown on the button.
 - **state:**

This is the initial state of the button checkbox.



- o **selectors:**

Selector buttons are two different types of buttons. Each type can have a selected value based on a list of available values. The two types are **Swipe** button (type 0) and **Radio** button (type 1).

The buttons are defined in a table with the following structure:

- **name:**

This is a string value - the text will be shown on the button (swipe button) or as a label above the buttons (radio buttons).

- **selectedValue:**

This is an integer value - it defines the default selected value

- **type:**

This is an integer value - it defines the type of selector button. The options are:

- **0:**

This defines the button as a swipe button.

- **1:**

This defines the button as a radio button.

- **values:**

This is another table containing the different values available for the selector button. Each value element in the table has the following structure: **"string"]=integer**

The string is the name displayed for the value. The integer is the value returned and the one used for **selectedValue** (see above). See example 6 below for an example of use.

The table can contain some or all of the elements described above.

The colors mentioned above can be a string or number value. It refers to a defined UI Color in the color theme, for instance, "**Global.Text**" or **1.27**. See more in the **Color Theme topic**.

A message box pop-up should have at least a title, message, and *either* a timeout *or* some basic command buttons. See the first three examples below.

Return



- **Table:**

The return from a message box pop-up is formatted in a table. The returned table adjusts to match the elements of the message box. For instance, if there are selection buttons in the message box, then there is another table inside the result table containing the return from the selection buttons. See the examples for details on how to retrieve the results. The table can contain the following elements:

- **success:**
This is a boolean - it returns true if the message box was closed by tapping a command button or by a timeout (see **timeoutResultCancel** above for exceptions).
- **result:**
This is an integer - it returns the value of the tapped command button or the timeout result ID.
- **inputs** (only if the message box has inputs fields):
This is a table with a list of the input fields' name and string value in a key-value pair table.
- **states** (only if the message box has state buttons):
This is a table with a list of the state buttons' name and boolean value in a key-value pair table.
- **selectors** (only if the message box has selector buttons):
This is a table with a list of the selector buttons' name and integer value in a key-value pair table.

An error is thrown if the message box does not have inputs, states, and selectors, but the script tries to use the table elements.

See the examples to see how to extract the results.

Examples

There are six different examples demonstrating different elements of the message box. The elements can be combined, but the examples highlight different functions.

Example 1

A simple message box pop-up that shows a single confirm button:

```
return function ()  
  - This creates a small pop-up with a single button.  
  local returnTable = MessageBox(  
    {  
      title = "Please confirm This",  
      commands = {{value = 1, name = "Confirm"}}  
    }  
  )  
  
  - Print the content of the returned table.  
  Printf("Success = "..tostring(returnTable.success))  
  Printf("Result = "..returnTable.result)  
end
```



Example 2

This example opens a pop-up with some text and two command buttons:

```
return function ()
  -- A table with two default buttons for the pop-up
  local defaultCommandButtons = {
    {value = 2, name = "OK"},
    {value = 1, name = "Cancel"}
  }

  -- A table with the elements needed for the pop-up
  local messageTable = {
    icon = "object_smart",
    backColor = "Window.Plugins",
    title = "This is the title",
    message = "This is a message\nThat can have multiple lines",
    commands = defaultCommandButtons,
  }

  -- The creation on the actual pop-up with the result stored in a variable
  local returnTable = MessageBox(messageTable)

  -- Print the content of the returned table
  Printf("Success = "..tostring(returnTable.success))
  Printf("Result = "..returnTable.result)
end
```

Example 3

This example displays a message box for 3 seconds and then closes itself:

```
return function ()
  -- This variable contains the table used as argument for the messagebox
  local messageTable = {
    title = "Do not worry",
    message = "This message will self destruct - Goodbye!",
    timeout = 3000,
    timeoutResultCancel = false,
    timeoutResultID = 99,
  }

  -- This creates the messagebox pop-up and store the return table in a variable
  local returnTable = MessageBox(messageTable)

  -- Print the content of the returned table
  Printf("Success = "..tostring(returnTable.success))
end
```



```
Printf("Result = "..returnTable.result)
end
```

Example 4

This example adds state buttons to the message box. The buttons are added to a table for a better overview.

```
return function ()
  -- A table with two default buttons for the pop-up
  local defaultCommandButtons = {
    {value = 2, name = "OK"},
    {value = 1, name = "Cancel"}
  }
  -- A table with three state buttons
  -- The buttons will be displayed alphabetically in the pop-up
  local stateButtons = {
    {name = "State B", state = false},
    {name = "State A", state = false},
    {name = "New State", state = false}
  }

  -- A table with the elements needed for the pop-up
  local messageTable = {
    icon = "object_smart",
    backColor = "Window.Plugins",
    title = "This is state buttons",
    message = "Toggle the states and click "Ok"",
    commands = defaultCommandButtons,
    states = stateButtons,
  }

  -- The creation on the actual pop-up with the result stored in a variable
  local returnTable = MessageBox(messageTable)

  -- Print the content of the returned table
  Printf("Success = "..tostring(returnTable.success))
  Printf("Result = "..returnTable.result)
  -- Print a list with the state of the stateButtons
  for name,state in pairs(returnTable.states) do
    Printf("State '%s' = '%s'",name,tostring(state))
  end
end
```

Example 5

This example shows the input fields.



```
return function ()
  -- A table with two default buttons for the pop-up
  local defaultCommandButtons = {
    {value = 2, name = "OK"},
    {value = 1, name = "Cancel"}
  }
  -- A table with three input fields
  -- The fields will be displayed alphabetically in the pop-up based on name
  local inputFields = {
    {name = "Numbers Only", value = "1234", whiteFilter = "0123456789", vkPlugin = "NumericInput"},
    {name = "Text Only", value = "abcdef", blackFilter = "0123456789"},
    {name = "Maximum 10 characters", value = "", maxLength = 10}
  }
  -- Possible vkPlugin values:
  -- "TextInput" : same as default - standard on-screen keyboard
  -- "TextInputNumOnly" : text input but only with number buttons
  -- "TextInputNumOnlyRange" : text input but only with number and related range buttons
  -- "TextInputTimeOnly" : text input styled for time input - includes buttons for time values
  -- "NumericInput" : general number input
  -- "CueNumberInput" : number input styled for cue number
  -- "RelCueNumberInput" : number input with the relative "delta" button
  -- "IP4Prefix" : designed for inputting an IPv4 address allowing CIDR notation

  -- A table with the elements needed for the pop-up
  local messageTable = {
    icon = "object_smart",
    backColor = "Window.Plugins",
    title = "This is input fields",
    message = 'Change the values in the input fields and click "Ok"',
    commands = defaultCommandButtons,
    inputs = inputFields,
  }

  -- The creation on the actual pop-up with the result stored in a variable
  local returnTable = MessageBox(messageTable)

  -- Print the content of the returned table
  Printf("Success = "..tostring(returnTable.success))
  Printf("Result = "..returnTable.result)
  -- Print a list with the values of the input fields
  for name,value in pairs(returnTable.inputs) do
    Printf("Input '%s' = '%s'",name,tostring(value))
  end
end
```

Example 6

This example shows the different selector buttons.

```
return function ()
  - A table with two default buttons for the pop-up
  local defaultCommandButtons = {
    {value = 2, name = "OK"},
    {value = 1, name = "Cancel"}
  }
  - A table with selector buttons
  - The buttons will be displayed alphabetically in the pop-up based on name
  local selectorButtons = {
    { name="l_mSwipeSelector", selectedValue=1, type=0, values={"Swipe1"]=1,["Swipe2"]=2}},
    { name="l_mRadioSelector", selectedValue=2, type=1, values={"Radio1"]=1,["Radio2"]=2}},
    { name="l_mAnotherRadio", selectedValue=3, type=1, values={"Radio3"]=3,["Radio4"]=4}}
  }

  - State button to show grouping with swipe Selector button
  local stateButton = {
    {name = "State Button", state = false},
  }
  - A table with the elements needed for the pop-up
  local messageTable = {
    icon = "object_smart",
    backColor = "Window.Plugins",
    title = "This is input fields",
    message = 'Change the values in the input fields and click "Ok"',
    commands = defaultCommandButtons,
    states = stateButton,
    selectors = selectorButtons,
  }

  - The creation on the actual pop-up with the result stored in a variable
  local returnTable = MessageBox(messageTable)

  - Print the content of the returned table
  Printf("Success = ..tostring(returnTable.success))
  Printf("Result = ..returnTable.result)

  - Print a list with the values of the selection buttons
  for name,value in pairs(returnTable.selectors) do
    Printf("Input '%s' = '%s'",name,tostring(value))
  end

  - Print a list with the state of the stateButton
```



```
for name,state in pairs(returnTable.states) do
  Printf("State '%s' = '%s'",name,tostring(state))
end
end
```



39.5.108. MouseObj()

Description

The **MouseObj** function returns the handle to the first found mouse object.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The function returns the handle to the mouse object.

Example

This example prints the information of the mouse object:

```
return function()  
  - Print all informatin about the MouseObj object  
  Printf("===== START OF DUMP =====")  
  MouseObj():Dump()  
  Printf("===== END OF DUMP =====")  
end
```




39.5.109. ObjectList

Description

The ObjectList Lua function generates a list of objects based on a text. The text has the same syntax as the interaction with the command line (e.g. "Fixture 1 Thru 10").

Syntax

ObjectList(string:address)

Returns: table of light_userdata:handle

Example

To create a list of fixtures 1 to 10 and output the name of the first fixture of this list, create a plugin with this code:

```
local function main()
    local myObjects = ObjectList("Fixture 1 Thru 10")

    Printf("Name of Fixture 1 = " .. myObjects[1].name)
end

return main
```



39.5.111. Patch()

Description

The **Patch** Lua function returns a handle to the patch object.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The returned handle to the patch.

Example

This example prints the data connected to the handle. It uses the **Dump()** function.

```
return function()  
  – This example dumps all information about the patch object  
  Printf("===== START OF DUMP =====")  
  Patch():Dump()  
  Printf("===== END OF DUMP =====")  
end
```



39.5.113. PopupInput

Description

The PopupInput Lua function creates a popup input field in the UI, where the user can select an item out of a list of different variables.

Syntax

PopupInput({title:str,caller:handle,items:table:{{'str'|'int'|'lua'|'handle', name, type-dependent}...},selectedValue:str,x:int,y:int,target:handle,render_options:{left_icon,number,right_icon},useTopLeft:bool,properties:{prop:value}}): string: value

Example

To open a popup input and print the selected result in the command line history, create a plugin with this code:

```
local function main(displayHandle)
  local descTable = {
    title = "Demo",
    caller = displayHandle,
    items = {"Select","Some","Value","Please"},
    selectedValue = "Some",
    add_args = {FilterSupport="Yes"},
  }
  local a,b = PopupInput(descTable)
  Printf("a = %s",tostring(a))
  Printf("b = %s",tostring(b))
end

return main
```



39.5.115. Printf(string)

Description

The Printf Lua function prints a string in the **Command Line History** and **System Monitor**.

Argument

- **String:**
The string text to be printed to the Command Line History.

Return

This function does not return anything.

Example

This example prints "Hello World!" in the Command Line History:

```
local function main()  
    Printf("Hello World!")  
end  
  
return main
```



39.5.117. ProgrammerPart

Description

ProgrammerPart is an object-free function that does not take an input argument.

ProgrammerPart() returns the current programmer part object.

Syntax

ProgrammerPart(nothing): light_userdata:handle

Example

This simple example prints the information of the ProgrammerPart object:

```
local function main()  
  ProgrammerPart():Dump()  
end
```

```
return main
```



39.5.118. Pult()

Description

The **Pult** Lua function returns a handle to the current "Pult" object at position Root/GraphicsRoot/PultCollect. The "Pult" object contains display and device information.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The returned handle to the pult object.

Example

This example prints the data connected to the handle. It uses the **Dump()** function.

```
return function()  
  - The following prints the dump for the pult object  
  Printf("===== START OF DUMP =====")  
  Pult():Dump()  
  Printf("===== END OF DUMP =====")  
end
```



39.5.120. ReleaseType()

Description

The ReleaseType Lua function returns a string with the type of release for the MA software. All the software versions available from MA Lighting will return "Release". Internally and during development, there can be other release types.

Arguments

This function does not accept any arguments.

Return

- **String:**
The returned string is the release type of the grandMA3 software.

Example

This example prints the release type in the Command Line History:

```
return function()  
    Printf("The ReleaseType is "..ReleaseType())  
end
```



39.5.122. Root()

Description

Root() is an object-free function that returns a handle to the object at root position.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The function returns a handle to the Root object.

Example

This simple example prints the information of the Root object in the Command Line History using the **Dump()** function:

```
local function main()
  Root():Dump()
end

return main
```




39.5.124. SelectedSequence()

Description

The **SelectedSequence** Lua function returns the handle of the selected sequence.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The returned handle for the selected sequence.

Example

This example prints all information about the selected sequence in the Command Line History using the **Dump()** function:

```
return function ()  
    SelectedSequence():Dump()  
end
```



39.5.125. Selection()

Description

The Selection Lua function returns a handle to the object holding the current selection of fixtures.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The function returns a handle to the Selection object.

Example

This example prints the information of the Selection object in the Command Line History using the **Dump()** function:

```
local function main()  
  Selection():Dump()  
end
```

```
return main
```



39.5.129. SelectionCount()

Description

The **SelectionCount** Lua function returns a number indicating the total amount of currently selected fixtures.

Arguments

This function does not accept any arguments.

Return

- **Integer:**
The function returns an integer number depicting the total amount of fixtures in the current selection.
If there is no selection, then it returns 0.

Example

This example prints the number of fixtures in the current selection to the Command Line History:

```
return function()  
  Printf('Number of fixtures in the current selection: %i', SelectionCount())  
end
```



39.5.130. SelectionFirst()

Description

The **SelectionFirst** Lua function returns a set of integers for the selection's first fixture. It is the patch index number and the XYZ grid values in the selection grid.

It is not required to use all four returned integers, but they are returned in order.

Arguments

This function does not accept any arguments.

Return

- **Integer:**
The returned number is the patch index of the first fixture in the current selection. It is not the FID or CID. The index is 0-based.
- **Integer:**
The returned number is the current position on the X-axis in the selection grid. The selection grid is 0-based.
- **Integer:**
The returned number is the current position on the Y-axis in the selection grid. The selection grid is 0-based.
- **Integer:**
The returned number is the current position on the Z-axis in the selection grid. The selection grid is 0-based.

Example

This example prints the returned numbers, of the first fixture in the selection, to the Command Line History:

```
local function main()
  -- Store the return in a local variable
  local fixtureIndex, gridX, gridY, gridZ = SelectionFirst();

  -- Uncomment the following lines to print the types of the return
  -- Printf("Index type is: "..type(fixtureIndex));
  -- Printf("gridX type is: "..type(gridX));
  -- Printf("gridY type is: "..type(gridY));
  -- Printf("gridZ type is: "..type(gridZ));

  -- Cancel the plugin if no fixture is selected
  assert(fixtureIndex,"Please select a fixture and try again.");

  -- Print the index number of the first fixture in the selection
  Printf("First selected fixture has index number: "..fixtureIndex
  .." and gridX value: "..gridX
  .." and gridY value: "..gridY
  .." and gridZ value: "..gridZ);
```



end

return main

Related Functions

- **SelectionNext**
- **GetSubfixture**
- **GetSubFixtureCount**

39.5.131. SelectionNext()

Description

The SelectionNext function returns a set of integers for the next fixture in a selection based on the index number input as an argument. It is the index number in the patch and the XYZ grid values in the selection grid.

It is not required to use all four returned integers, but they are returned in order.

Arguments

- **Integer:**
The index number used to find the next fixture. The index number needs to be part of the current selection.

Return

- **Integer:**
The returned number is the patch index of the first fixture in the current selection. It is not the FID or CID. The index is 0-based.
- **Integer:**
The returned number is the current position on the X-axis in the selection grid. The selection grid is 0-based.
- **Integer:**
The returned number is the current position on the Y-axis in the selection grid. The selection grid is 0-based.
- **Integer:**
The returned number is the current position on the Z-axis in the selection grid. The selection grid is 0-based

Example

This example prints the patch index number and grid positions of all the fixtures in the current selection:

```
local function main()
  -- Store the return in a local variable
  local fixtureIndex, gridX, gridY, gridZ = SelectionFirst()

  -- Cancel the plugin if no fixture is selected
  assert(fixtureIndex, "Please select a (range of) fixture(s) and try again.")

  -- Loop that prints the index and gridpositions of all the fixtures in the selection
  while fixtureIndex do
    Printf("The fixture has index number: %i and gridposition %i / %i / %i",
      fixtureIndex, gridX, gridY, gridZ);

    -- Here is SelectionNext actually used to find the next fixture in the selection
    fixtureIndex, gridX, gridY, gridZ = SelectionNext(fixtureIndex)
  end
end

return main
```



Related Functions

- [SelectionFirst](#)
- [GetSubfixture](#)
- [GetSubfixtureCount](#)



39.5.135. SerialNumber()

Description

The **SerialNumber** Lua function returns the serial number of the grandMA3 hardware or grandMA3 onPC.

Arguments

This function does not accept any arguments.

Return

- **String:**
The returned string is the serial number of the grandMA3 hardware or grandMA3 onPC.

Example

This example prints the serial number in the Command Line History:

```
return function()  
  Printf("Serial number: " .. SerialNumber())  
end
```


39.5.136. SetBlockInput(boolean)

Description

The **SetBlockInput** function is an internal function used during the system tests. It stops input from USB-connected keyboards and mouse. The built-in keyboard on some models is internally connected using a USB connection, which is also blocked by this function. The block affects the station where the function is executed.



Important:

Should the station be blocked and cannot be unblocked using a new Lua command, then pressing the keyboard keys **A**, **S**, **D**, and **F** simultaneously unblock the input again.

Arguments

- **Boolean:**
 - The boolean indicates if the stations' input should be blocked or unblocked.
 - **true** (or 1): The station input is blocked.
 - **false** (or 0): The station input is unblocked.

Return

This function does not return anything.

Example

This example blocks mouse and keyboard input for 10 seconds:

```
return function()  
  – Set a variable for yield time in seconds  
  yieldTime = 10  
  – Set the block to true  
  SetBlockInput(true)  
  – Wait the [yieldtime]  
  coroutine.yield(yieldTime)  
  – Unblock the station  
  SetBlockInput(false)  
end
```



39.5.141. SetProgress(handle, integer)

Description

The SetProgress Lua function defines a value on the range for a progress bar. A handle input argument defines the progress bar. The progress bar needs have been created using the [StartProgress](#) function.

See the [ProgressBar topic](#) for more info regarding progress bars and links to other related functions.

Arguments

- **Handle:**
The handle for the progress bar.
- **Integer:**
The desired value indicating the current status or position of the progress bar.

Return

This function does not return anything.

Example

This example sets a range value for the progress bar created using the example in the StartProgress topic (link above):

```
return function()  
  – Sets the current value to 5 for a progress bar with the matching handle  
  SetProgress(progressHandle, 5)  
end
```



39.5.142. SetProgressRange(handle, integer, integer)

Description

The SetProgressRange Lua function defines a range for a progress bar.

A handle input argument defines which progress bar it defines a range for. The progress bar must exist to have a handle. Progress bars can be created using the [StartProgress](#) function.

See the [ProgressBar topic](#) for more info regarding progress bars and links to other related functions.

Arguments

- **Handle:**
The handle for the progress bar.
- **Integer:**
The start value for the range.
- **Integer:**
The end value for the range.

Return

This function does not return anything.

Example

This example sets a range for the progress bar created using the example in the StartProgress topic (link above):

```
return function()  
  – Sets the range of a progress bar with the matching handle  
  SetProgressRange(progressHandle, 1, 10)  
end
```



39.5.143. SetProgressText(handle, string)

Description

The SetProgressText Lua function defines a text string to be displayed in a progress bar next to the progress bar title text. The title cannot be changed after creation, but this text can be changed. It could be used to describe the current step in the progress.

A handle input argument defines which progress bar it defines a text for. The progress bar must exist for it to have a handle. Progress bars can be created using the [StartProgress](#) function.

See the [ProgressBar topic](#) for more info regarding progress bars and links to other related functions.

Arguments

- **Handle:**
The handle for the progress bar.
- **String:**
The text string to be displayed.

Return

This function does not return anything.

Example

This example sets a text string for the progress bar created using the example in the StartProgress topic (link above):

```
return function()  
    – Sets the text next to progress title text  
    SetProgressText(progressHandle, "- This is text next to the progress title")  
end
```



39.5.144. SetVar(handle, string, value)

The SetVar Lua function sets a value to a specific variable in a set of variables. To learn more about the variables in plugins, look at the [Variable Functions](#) topic.

If the variable exists, then the value is overwritten. If it does not exist, then it is created with value.

Arguments

- **Handle:**
The handle of variable set.
- **String:**
The name of the variable. It needs to be in quotation marks.
- **Value:**
The value can be a string, integer, or double.

Return

- **Boolean:**
 - True / 1: The variable was set.
 - False / 0: The variable was not set.

Example

This example sets a value to the variable called "myUserVar" in the set of user variables if it exists.

```
local function main()

    local success = SetVar(UserVars(), "myUserVar", "Hello World")

    if success then
        Printf("Variable is stored.")
    else
        Printf("Variable is NOT stored!")
    end

end

return main
```



39.5.145. ShowData()

Description

ShowData is an object-free function that returns a handle to the object at position Root/ShowData.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The function returns a handle to the ShowData object.

Example

This simple example prints the information of the ShowData object in the Command Line History using the **Dump()** function:

```
local function main()  
    ShowData():Dump()  
end
```

```
return main
```



39.5.146. ShowSettings()

Description

ShowSettings is an object-free function that returns a handle to the object at Root/ShowData/ShowSettings.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The function returns a handle to the ShowSettings object.

Example

This simple example prints the information of the ShowSettings object using the **Dump** function:

```
local function main()  
    ShowSettings():Dump()  
end  
  
return main
```

39.5.147. StartProgress(string)

Description

The StartProgress Lua function creates and displays a progress bar on all screens. A string input argument creates a title for the progress bar. The function returns a handle that is used to further interact with the progress bar.

Executing the function displays the progress bar on the screens. It only disappears using the **StopProgress** function - which needs the handle. So it is highly recommended to store the returned handle from the start function.

See the **ProgressBar topic** for more info regarding progress bars and links to other related functions.

Arguments

- **String:**
The string is used as the title for the progress bar.

Return

- **Handle:**
The returned handle is the identifier for the progress bar.

Example

This creates and displays a progress bar on all screens. The progress bar does not disappear using this example - see the example in the StopProgress (link above) function to remove:

```
return function ()  
    - Create and display a progress bar with a title  
    progressHandle = StartProgress("ProgressBar Title")  
end
```



Important:

Running this example multiple times creates new progress bars that cannot be removed. So only run this once, and then remove it again using the StopProgress function (link above).



39.5.148. StopProgress(handle)

Description

The StopProgress Lua function removes a progress bar. A handle input argument defines which progress bar it removes. The progress bar must exist before it can be removed. Progress bars are created using the [StartProgress](#) function.

See the [ProgressBar topic](#) for more info regarding progress bars and links to other related functions.

Arguments

- **Handle:**
The handle for the progress bar to be stopped.

Return

This function does not return anything.

Example

This example stops the progress bar created using the example in the StartProgress topic (link above):

```
return function()  
  - Stops and closes the progress bar with the matching handle  
  StopProgress(progressHandle)  
end
```



39.5.149. StrToHandle(string)

The object-free StrToHandle Lua function converts a string with a hexadecimal number format into a handle. The string needs to correlate with an actual handle.

See the [Handle topic](#) for more info regarding handles and links to other related functions.

Arguments

- **String:**
The string with a handle number in a hexadecimal format.

Return

- **Handle:**
The returned handle based on the string with a hexadecimal number.

Example

This example prints the handle hex number for the selected sequence. It also converts the string back to a handle and uses this to print the name of the sequence:

```
local function main()
  local mySequence = SelectedSequence()    -- Store the handle of the selected sequence
  local mySeqStr = HandleToStr(mySequence) -- Store a string with the handle converted to hex

  Printf("The handle for the selected sequence (string): " .. mySeqStr)

  Printf("The name of the selected sequence is: " .. StrToHandle(mySeqStr).name)
end

return main
```



39.5.151. TextInput([string[, string[, integer[, integer]]])

Description

The TextInput Lua function opens a text input pop-up and returns the typed input as a string. It is part of the user interface functions.

Arguments

- **String** (optional):
This string is the title for the pop-up. The title bar has a default "Edit" text at the beginning of the title that cannot be removed.
- **String** (optional):
This string is the text already in the input field - can be used to provide user guidance.
- **Integer** (optional):
This integer defines a position on the x-axis where the pop-up should appear (on all screens). "0" is on the left side of the screen. Nil or undefined is centered.
- **Integer** (optional):
This integer defines a position on the y-axis where the pop-up should appear (on all screens). "0" is at the top of the screen. Nil or undefined is centered.

Return

- **String**:
The returned user input.

Example

To open a text input and print the entered value in the Command Line History, create a plugin with this code:

```
local function main()  
    local input = TextInput(" This is the title","Please provide your input here")  
    Printf("You entered this message: %s",tostring(input))  
end  
  
return main
```



39.5.152. Time()

Description

The **Time** function returns the time (in seconds) the station has been on, as a number (float). It is basically a stopwatch that starts when the grandMA3 application starts. It is not the current time of day or the session online time.

Arguments

This function does not accept any arguments.

Return

- **Number:**
The returned number (float) is the on-time for the station.

Example

This example finds, formats, and prints the time.

```
return function()  
  - Get the current time  
  local time = Time()  
  
  -Calculate the different elements  
  local days = math.floor(time/86400)  
  local hours = math.floor((time % 86400)/3600)  
  local minutes = math.floor((time % 3600)/60)  
  local seconds = math.floor(time % 60)  
  
  -Print the result  
  Printf("The time is %dd%02dh%02dm%02ds", days, hours, minutes, seconds)  
end
```



39.5.153. Timer

Description

The Timer Lua function is used to call a Lua function with a time delay.

Syntax

**Timer(function:name, number:delaytime, number:max_count, [function:cleanup], [light_userdata:context object]):
nothing**

Returns: nothing

Example

To call a Lua function three times with a delay of one second between each function call, create a plugin with this code:

```
local function main()
    local function timedFunction()
        Printf("Do something")
    end
    local waitSeconds = 1
    local iterations = 3
    Timer(timedFunction, waitSeconds, iterations);
end

return main
```



39.5.156. TouchObj()

Description

The **TouchObj** function returns the handle to the first found touch object.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The function returns the handle to the touch object.

Example

This example prints information about the touch object using the **Dump()** function:

```
return function()  
  - Print all informatin about the TouchObj object  
  Printf("===== START OF DUMP =====")  
  TouchObj():Dump()  
  Printf("===== END OF DUMP =====")  
end
```



39.5.159. UserVars()

The UserVars function returns a handle to the set of user variables. Read more about these in the [Variables](#) topic in the Macro section.

Arguments

This function does not accept any arguments.

Return

- **Handle:**
The function returns a handle of the set of user variables.

Example

This example sets, gets, and deletes a user variable:

```
local function main()

    -- Stores a local Lua variable with the handle for the user variables.
    local variableSection = UserVars()

    -- Sets a user variable with an integer value using the SetVar function.
    SetVar(variableSection, "myUserVar", 42)

    -- Prints the user variable using the GetVar function.
    Printf("The value of myUserVar is: " .. GetVar(variableSection, "myUserVar"))

    -- Deletes the user variable using the DelVar function.
    DelVar(variableSection, "myUserVar")

end

return main
```



39.5.160. Version()

Description

The Version Lua function returns the software version.

Arguments

This function does not accept any arguments.

Return

- **String:**
The returned string is the version of the grandMA3 software.

Example

This example prints the software version in the Command Line History:

```
local function main()  
  Printf("Software version: " .. Version())  
end  
  
return main
```




39.6. Lua Functions - Object API

Object API means Lua functions that are related to an object.

Many of the object functions take a handle as an argument. In most cases, this can be omitted if the function is used with the colon notation.

Syntax with the handle:

object.function(object)

Syntax with colon operator:

object:function()

Most examples here use the colon operator.



39.6.1. Addr(handle[, handle[, boolean]])

The **Addr** Lua object function converts a handle to an address string that can be used in commands.

See the [Handle topic](#) for more info regarding handles and links to other related functions.

Arguments

- **Handle:**
The function takes a handle of the type "light_userdata" as an argument. It can be omitted when using the colon notation on an object. See the examples below.
This is the handle to the object where the address is requested.
- **Handle** (optional):
The returned address is from the root as a default. This optional handle can specify a different base location. It still needs to be a base location in the address path from the root to the object.
- **Boolean** (optional):
This can be useful if there is a difference between the ToAddr() and Addr(). Setting this to "true" uses the index number from the ToAddr() instead of the Addr() index number. See the example below.

Return

- **String:**
Text string with the address in a parent-child number format separated by dots.

Example

This example prints different versions of the address to a cue in a sequence:

```
return function()  
  - Creates a cue in sequence 1  
  Cmd("Store Sequence 1 Cue 100 /Merge /NoConfirmation")  
  -Store a handle to the created cue  
  local cueObject = ObjectList("Sequence 1 Cue 100")[1]  
  -Print different version of the handle address  
  Printf("ToAddr:      " .. cueObject:ToAddr())  
  Printf("Addr:       " .. cueObject:Addr())  
  Printf("Addr(Parent, false): " .. cueObject:Addr(cueObject:Parent(), false))  
  Printf("Addr(Parent, true): " .. cueObject:Addr(cueObject:Parent(), true))  
end
```



39.6.2. AddrNative(handle[, handle[, boolean]])

The AddrNative Lua object function converts a handle to an address string that can be used in commands.

See the [Handle topic](#) for more info regarding handles and links to other related functions.

Arguments

- **Handle:**
The function takes a handle as an argument. It can be omitted when using the colon notation on an object.
See the example below.
This is the handle to the object where the address is requested.
- **Handle** (optional):
The returned address is from the root as a default. This optional handle can specify a different base location.
It still needs to be a base location in the address path from the root to the object.
- **Boolean** (optional):
Set this to "true" to get the returned names in quotation marks.

Return

- **String:**
Text string with the address in a parent-child name format separated by dots.

Example

This example prints the address of the first sequence:

```
local function main()
  local mySequence = DataPool().Sequences[1]      -- Stores the handle to the first sequence

  Printf("The full address is: " .. mySequence:AddrNative())

  local myDataPool = DataPool()                  -- Stores a handle to the default DataPool.

  Printf("The address in the datapool is: " .. mySequence:AddrNative(myDataPool))

  Printf("The address in the datapool with quotes around the names is: " .. mySequence:AddrNative(myDataPool, true))
end

return main
```



39.6.3. Children(handle)

Description

The Children Lua function creates a table of handles of the children of an object.

Arguments

- **Handle:**
The function takes a handle of the type "light_userdata" as an argument. It can be omitted when using the colon notation on an object. See the example below.

Return

- **Table:**
The function returns a table with the child objects. If there are no children, then it returns an empty table.

Example

This example returns the name of the cues in the first sequence of the selected data pool:

```
local function main()
  local mySequence = DataPool().Sequences[1] -- Stores the handle for sequence 1 in a variable.
  local cues = mySequence:Children()       -- Use the "Children()" function to store a table with all the children in a new

  for i = 1, #cues do                       -- For loop that uses the length operator on the cue variable.
    Printf("Sequence 1 Child " .. i .. " = " .. cues[i].name) -- Text is printed for each child.
  end
end

return main
```



39.6.4. Dump (handle)

Description

The function returns a string with information about the object, for instance, the name, class, path of the object, its properties, and children.

Arguments

- **Handle:**

The function takes a handle of the type "light_userdata" as an argument.

Often, Dump is used without an argument but is called with the colon notation for object-oriented calls. See the examples below.

Return

The function does not return anything, but it outputs information about the object in the **Command Line History window**.

Example

These examples all print information about the selected sequence in the Command Line History.

First example using the colon operator:

```
local function main()
    SelectedSequence():Dump()      -Dump() is called on a function
end

return main
```

The second example uses an argument with the same result:

```
local function main()
    SelectedSequence().Dump(SelectedSequence()) -Dump() is called with function as an argument
end

return main
```

The third example uses a variable with the same result:

```
local function main()
    local mySeqHandle = SelectedSequence() -Stores the handle for the selected sequence in a local variable
    mySeqHandle:Dump()                    -Dump() is called on the variable
end
```



end

return main



39.6.5. Export(handle, file_path, file_name)

Description

The object Export Lua function exports an object into an XML file.

Arguments

- **Handle:**
The function takes a handle of the type "light_userdata" as an argument. It can be omitted when using the colon notation on an object. See the examples below.
- **File_path:**
This is a string with the file path.
- **File_name:**
This is a string containing the file name of the exported file.

Return

The function does not return anything. It does export the object.

Example

These two examples both export the selected sequence into an XML file. It is two examples of how to achieve the same outcome.

First example using the colon operator:

```
local function main()
    local selectedSequence = SelectedSequence()           --SelectedSequence() creates a handle to the selected sequence
    local exportPath = GetPath(Enums.PathType.UserSequences)  --The path is stored in a variable.
    selectedSequence:Export(exportPath, "mySelectedSequence.xml") --The actual export function.
    Printf("The sequence is exported to: " .. exportPath)    --Print some feedback.
end

return main
```

The second example uses an argument with the same result:

```
local function main()
    local selectedSequence = SelectedSequence()           --SelectedSequence() creates a handle to the selected sequence
    local exportPath = GetPath(Enums.PathType.UserSequences)  --The path is stored in a variable.
```



```
selectedSequence.Export(selectedSequence, exportPath, "mySelectedSequence2.xml") --The actual export function.  
  
Printf("The sequence is exported to: " .. exportPath) --Print some feedback.  
end  
  
return main
```




39.6.6. HasActivePlayback(handle)

Description

The HasActivePlayback Lua function returns a boolean indicating if an object has a currently active playback, for instance, if a sequence has an active cue.

Arguments

- **Handle:**
The function takes a handle of the type "light_userdata" as an argument. It can be omitted when using the colon notation on an object. See the example below.

Return

- **Result:**
The function returns a boolean indicating the playback status:
 - **True:** There is active playback.
 - **False:** There is no active playback.

Example

To return the information if the selected sequence has an active playback, create a plugin with this code:

```
local function main()
    local selectedSequence = SelectedSequence() -- Stores the handle of the selected sequence.

    -- The following is an 'if' that gives different feedback based on the playback status.
    if selectedSequence:HasActivePlayback() then
        Printf("Sequence " ..selectedSequence.name.. " has active playback.")
    else
        Printf("Sequence " ..selectedSequence.name.. " has NO active playback.")
    end
end

return main
```

39.6.7. Import(handle, file_path, file_name)

Description

The object version of the Import Lua function is used to import an object written in XML format.



Restriction:

The files need to already exist to be imported.



Important:

The Lua import will merge the content of the XML file into the object without any confirmation pop-up.

Arguments

- **Handle:**
The function takes a handle of the type "light_userdata" as an argument. It can be omitted when using the colon notation on an object. See the examples below.
- **File_path:**
This is a string with the path to the file location.
- **File_name:**
This is a string containing the file name of the desired file.

Return

The function does not return anything. It does import the object.

Example

These two examples import the content of the XML file into the selected sequence. The file is called "MySelectedSequence", and it is located at "../gma3_library/datapools/sequences". It is two examples of how to achieve the same outcome.

First example using the colon operator:

```
local function main()  
    local selectedSequence = SelectedSequence()      --SelectedSequence() creates a handle to the selected sequence  
  
    local path = GetPath(Enums.PathType.UserSequences)  --The path is stored in a variable.  
  
    selectedSequence:Import(path, "mySelectedSequence.xml") --The actual import function.  
  
    Printf("The sequence is imported from: " .. path)  --Print some feedback.  
end  
  
return main
```



The second example uses a handle as an argument with the same result:

```
local function main()
    local selectedSequence = SelectedSequence()           --SelectedSequence() creates a handle to the selector
    local path = GetPath(Enums.PathType.UserSequences)   --The path is stored in a variable.
    selectedSequence.Import(selectedSequence, path, "mySelectedSequence.xml") --The actual import function.
    Printf("The sequence is imported from: " .. path)    --Print some feedback.
end
return main
```



39.6.8. ToAddr(handle)

Description

The ToAddr Lua object function converts a handle to an address string that can be used in commands.

See the [Handle topic](#) for more info regarding handles and links to other related functions.

Arguments

- **Handle:**
The function takes a handle of the type "light_userdata" as an argument. It can be omitted when using the colon notation on an object. See the examples below.

Return

- **Address:**
String with the address value.

Example

These two examples return the address of the first sequence of the selected data pool and create a grandMA3 command with a "Go" keyword in front of the address. This command is sent to the grandMA3 command line.

First example using the colon operator:

```
local function main()  
  local mySequence = DataPool().Sequences[1]    – Stores the handle in a variable.  
  local mySequenceAddress = mySequence.ToAddr() – Converts the handle to the address and store in variable.  
  Cmd("Go %s", mySequenceAddress)             – Send a go command with the address appended.  
end
```

return main

The second example uses a handle as an argument with the same result:

```
local function main()  
  local mySequence = DataPool().Sequences[1]    – Stores the handle in a variable.  
  local mySequenceAddress = ToAddr(mySequence) – Converts the handle to the address and store in variable  
  Cmd("Go %s", mySequenceAddress)             – Send a go command with the address appended.  
end
```

return main

40. What are Quickeys?

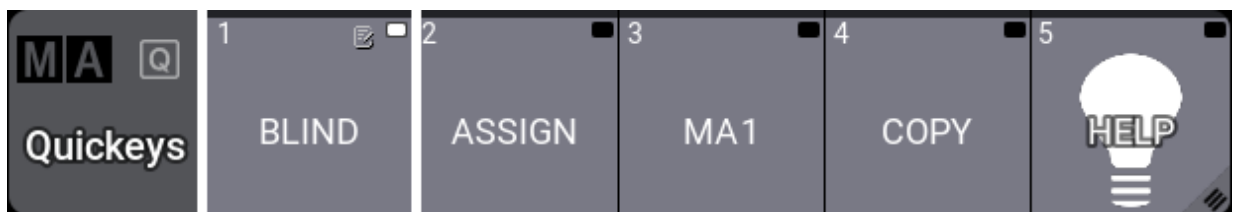
Quickeys can be used to set soft versions of various hardkeys and functions.

Quickeys can be addressed by tapping on the particular pool object or by using the **Quickey keyword**.

Quickeys can be assigned to XKeys, layouts and executors or can be used directly in the Quickeys pool.

Quickeys are stored in the Quickeys pool.

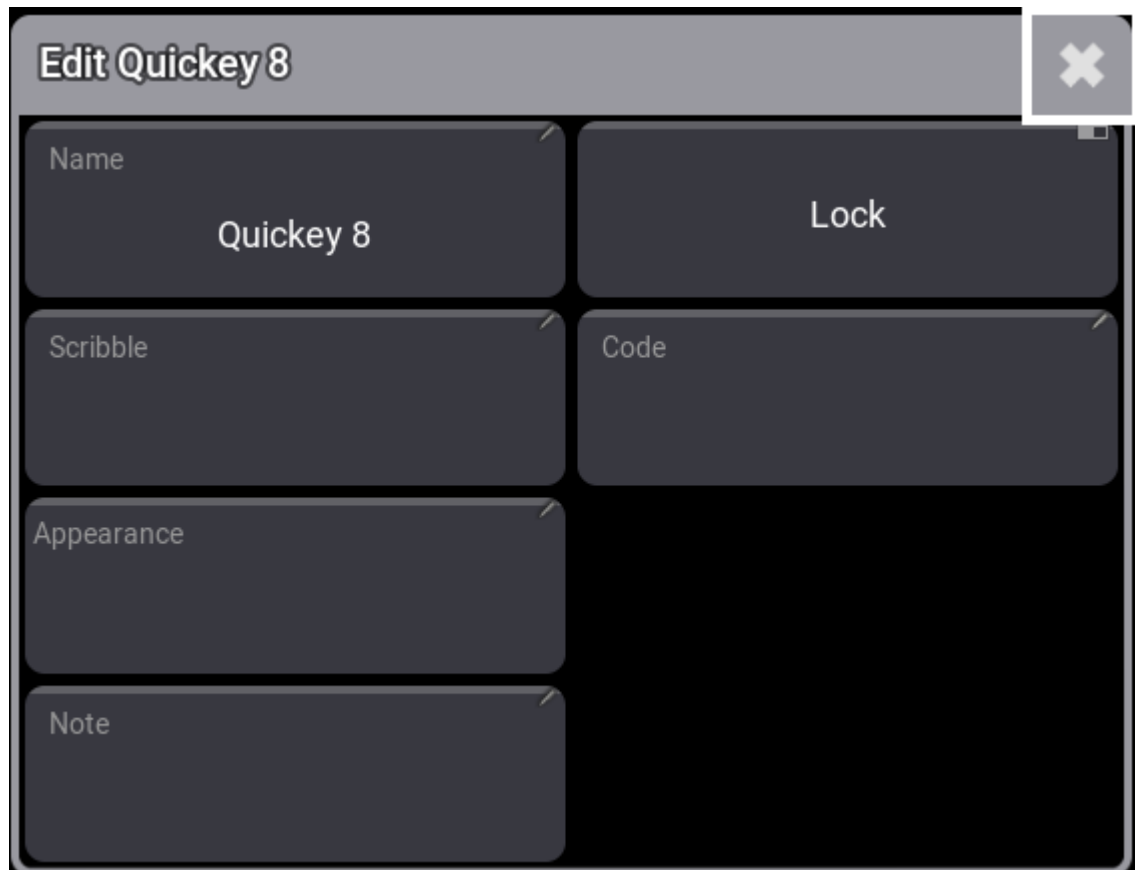
- To open the Quickeys pool, open the Add Window dialog, tap **Data Pools** and then tap **Quickeys**.



Quickeys pool

40.1. Quickey Editor

1. To open the Quickey editor, use Swipecy on an empty pool object and then tap **Edit**.
2. The Quickey editor opens:



Quickey editor




Hint:

To edit an existing Quickey, enable **CLI** in the Quickey Pool Settings or use the command line, for example **Edit Quickey 1**.

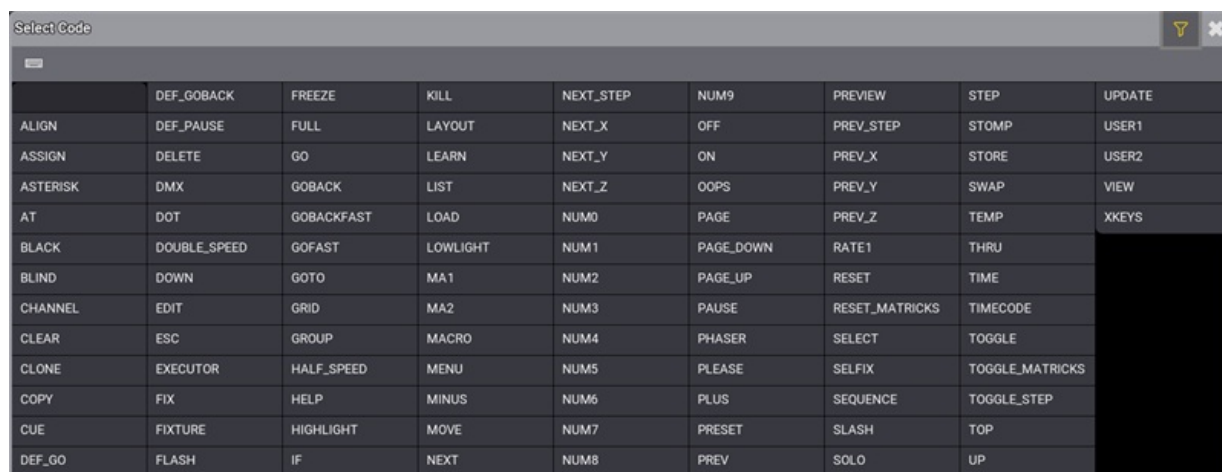


The following six settings are available in the Quickey editor:

- Name:**
 This is the name of the pool object.
 Selecting `Code` first takes over the name of the function.
- Scribble:**
 Assigns a scribble to the pool object. See [Scribble](#).
- Appearance:**
 Assigns an appearance to the pool object. See [Appearances](#).
- Note:**
 Edits a note to the pool object. See [Notes](#).
- Lock:**
 When enabled the pool object is looked for further changes.
- Code:**
 Opens the code list.

Tapping `Code` in the pool editor opens the Quickey code list. To filter codes, tap  and type the function into the text field.

For more information about the individual functions, see [Keys](#).



	DEF_GOBACK	FREEZE	KILL	NEXT_STEP	NUM9	PREVIEW	STEP	UPDATE
ALIGN	DEF_PAUSE	FULL	LAYOUT	NEXT_X	OFF	PREV_STEP	STOMP	USER1
ASSIGN	DELETE	GO	LEARN	NEXT_Y	ON	PREV_X	STORE	USER2
ASTERISK	DMX	GOBACK	LIST	NEXT_Z	OOPS	PREV_Y	SWAP	VIEW
AT	DOT	GOBACKFAST	LOAD	NUM0	PAGE	PREV_Z	TEMP	XKEYS
BLACK	DOUBLE_SPEED	GOFAST	LOWLIGHT	NUM1	PAGE_DOWN	RATE1	THRU	
BLIND	DOWN	GOTO	MA1	NUM2	PAGE_UP	RESET	TIME	
CHANNEL	EDIT	GRID	MA2	NUM3	PAUSE	RESET_MATRICKS	TIMECODE	
CLEAR	ESC	GROUP	MACRO	NUM4	PHASER	SELECT	TOGGLE	
CLONE	EXECUTOR	HALF_SPEED	MENU	NUM5	PLEASE	SELFIX	TOGGLE_MATRICKS	
COPY	FIX	HELP	MINUS	NUM6	PLUS	SEQUENCE	TOGGLE_STEP	
CUE	FIXTURE	HIGHLIGHT	MOVE	NUM7	PRESET	SLASH	TOP	
DEF_GO	FLASH	IF	NEXT	NUM8	PREV	SOLO	UP	

Select Code pop-up.

40.2. Use Quickey Pool Objects

For general information about pool objects, see [Pool Windows](#).

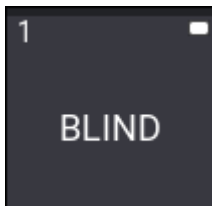
To activate or deactivate a Quickey, tap on the particular pool object.

A virtual LED in the top right corner in each pool object indicates the current status of the Quickey, depending on the LED color:

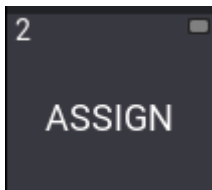
- **Black:** Quickey is off. Some Quickeys are triggered without a LED color, for example, MENU or NUM1.



- **White:** Quickey is active.



- **Grey:** Certain Quickeys trigger keywords. The appropriate keyword is entered into the command line, when the virtual LED is grey.



Hint:

The Quickeys MA1 and MA2 can be latched. For more information about latch, see [Command area](#).


40.3. Example

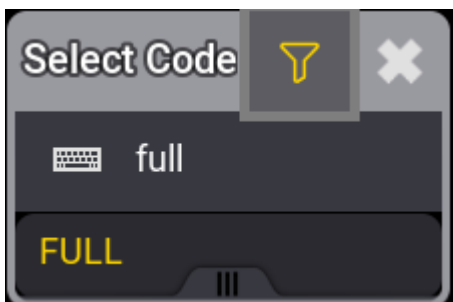
Requirements:

- Demo show file is loaded.
- Fixture Sheet is open.
- Quickeys pool is open.
- Clear the Programmer and turn the running playback off first. For more information, see [Clear Key](#) and [Running Playbacks](#).
- Type **Fixture 9 Thru 13** into the command line and then press **Please**, to select fixtures.
 1. To open the Quickey editor, use **Swipey** on an empty pool object and then tap **Edit**.




Swipey

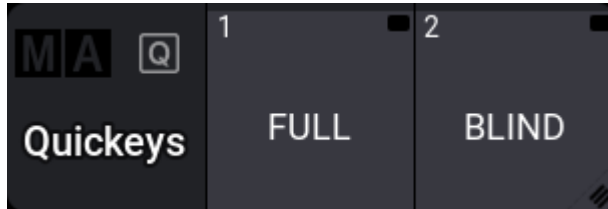
2. To open the code list, tap **Code**.
3. To open the filter bar, tap  and type "full":



Filter

4. Tap **FULL**, displayed in yellow font. The code list closes.
5. Tap  to close the editor. A new Quickey pool object is created.

6. Create a second Quickey with the **BLIND** function, using steps 1 to 5. The pool should look like this:



Quickey pool

7. To set the dimmer of the selected fixtures to 100%, tap **FULL** in the Quickey pool.
8. To toggle blind, tap **BLIND** repeatedly.

For more information about the used keywords, see [Full](#) and [Blind](#).

41. Data Pools

Much of the different data belonging to the show file are stored in pools. Many of these pools exist as children inside a **Data Pool** parent object.

A new show file creates a default **Data Pool**. The Data Pool object contains the other pools.

New **Data Pool** objects can be created, giving an entirely new set of pools.

This can be very useful if several shows or acts use the same patch; for instance, each song in an extensive band catalog can be in its own data pool.

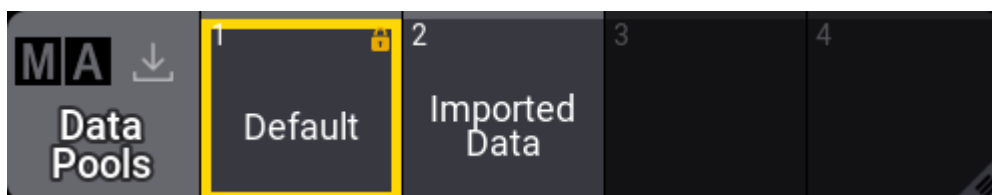
These are the pools inside the Data Pool objects:

- **Configurations**
- **Filters**
- Functionals
- **Groups**
- **Layouts**
- **Macros**
- **MAtricks**
- **Pages**
- **Plugins**
- **PresetPools**
- **Quickeys**
- **Sequences**
- **Timecodes**
- **Timers**
- **Worlds**

Each window of pools can be linked to the selected data pool or a specific data pool. Learn more in the **Window Settings topic**.

Data Pool Window

The best way to see the different data pools is in the **Data Pool window**. This can be created like any other window - learn how in the **Add Windows topic**.



Data pool with some data pool objects

The pool can be used to select the desired data pool by tapping it. Any of the different data pool operations mentioned below can be done using the appropriate keyword in combination with the pool window. But it can also be done using the command line and the **DataPool keyword**.

The data pools can also be found in **Menu** / **Settings** / **User Configuration** / **Pools**. This is a list form of the data pools.

Create a New Data Pool Object

Data pools need to be stored to be created. These are different ways to create a new Data Pool.

- Press **Store** and then tap an empty pool object.
- Open the swipecommands on an empty pool object and choose **Store**.
- Using the command line: **Store DataPool [DataPool_number]**

Performing an "edit" command on an empty pool object also creates the data pool object.

Copy a Data Pool Object

A data pool can be copied. This makes a copy of all the elements in the data pool. The copies are not linked, but objects inside the copy might be referencing objects in the source data pool. For instance, a copy of a sequence might contain links to presets in the source data pool - the presets are not copied automatically, and the references are not changed in the copy.

Changing one of the copies after the copy action does not change the other.

- Press **Copy**, tap the source pool object, and then an empty pool object.
- Open the swipecommands on the source pool object, choose **Copy**, then tap an empty pool object.
- Using the command line: **Copy DataPool ["DataPool_Name" or DataPool_Number] At ["DataPool_Name" or DataPool_Number]**

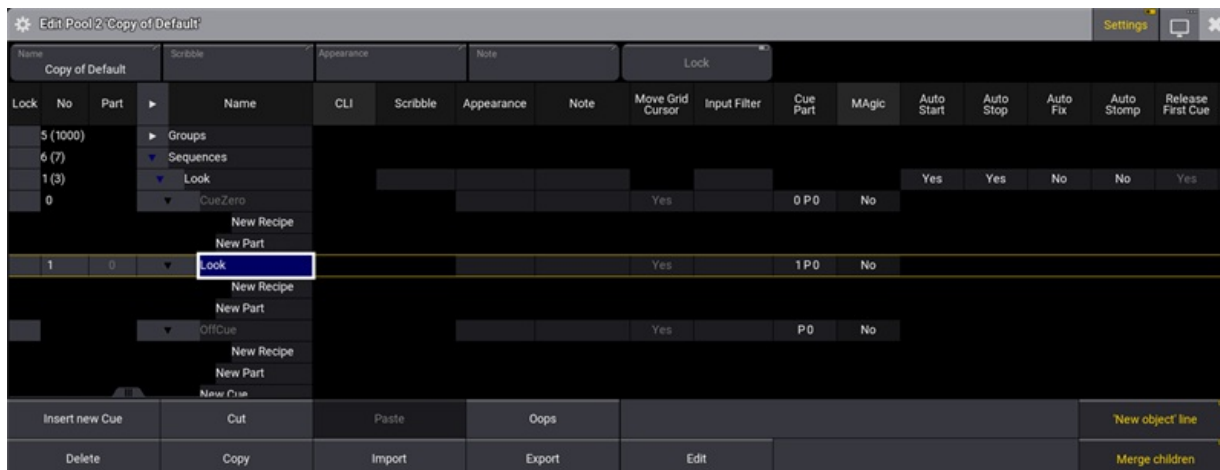


Hint:

Objects inside a data pool can be copied from one pool to another using the normal **Copy** and **Paste** commands.

Edit a Data Pool Object

Editing a data pool object opens an editor with a structured list of the elements in the data pool. Unfolding the elements in the list shows the settings for the elements.



Data pool in edit mode - Sequence unfolded

This is a deep look into the structure of the data.

Changing a setting here changes the setting for the object.

Edit a pool object using one of these methods:

- Press **Edit** and then tap the desired pool object.
- Open the swipecy commands on the pool object and choose **Edit**.
- Using the command line: **Edit DataPool ["DataPool_Name" or DataPool_Number]**.

Reference Data Pool Objects from a Different Data Pool

Other data pools can use a pool element inside a different data pool.

For instance, imagine a repertoire theater. All default elements are stored in the data pool one, including a sequence that controls the house lights. Tonight's show is programmed in data pool two. The sequence can be copied from data pool one, but if changes are made to it later, it would need to be copied again into the second data pool. Instead, the original sequence can be assigned to an executor in data pool two.

The image of the edit menu above shows that the structure in the data pool is numbered. So sequence 2 in this data pool is data pool object 1.6.2. The first number is the data pool number. The second number specifies that it is a sequence. The third number specifies that it is sequence number 2. If this element needs to be assigned somewhere, the data pool object must be included in the assign command. This can be done using the data pool object number or simply by adding the data pool information before the object - see the example below.

Example

Data pool 1 has sequence 2. This needs to be used in data pool 2 on (page 1) executor 201.

Make sure data pool 2 is the selected pool.

The example can be achieved using the command line: **Assign DataPool 1 Sequence 2 At Page 1.201.**

Now the sequence in data pool 1 can be controlled by the executor in data pool 2.

Some menus give access to select the data pool. In the title bar of the menus, there is a button called **DataPool** that can toggle between the different data pools.

Delete a Data Pool Object



Important:

Deleting a data pool object also deletes everything inside the data pool!

Unlocked data pools can be deleted using any of the following methods:

- Press **Delete** and then tap the pool object.
- Open the swipecy commands on the relevant pool object and choose **Delete**.
- Using the command line: **Delete DataPool ["DataPool_Name" or DataPool_Number].**



42. System

This section holds system windows that provide information on the system, operational views, and overall system settings.

42.1. Date and Time

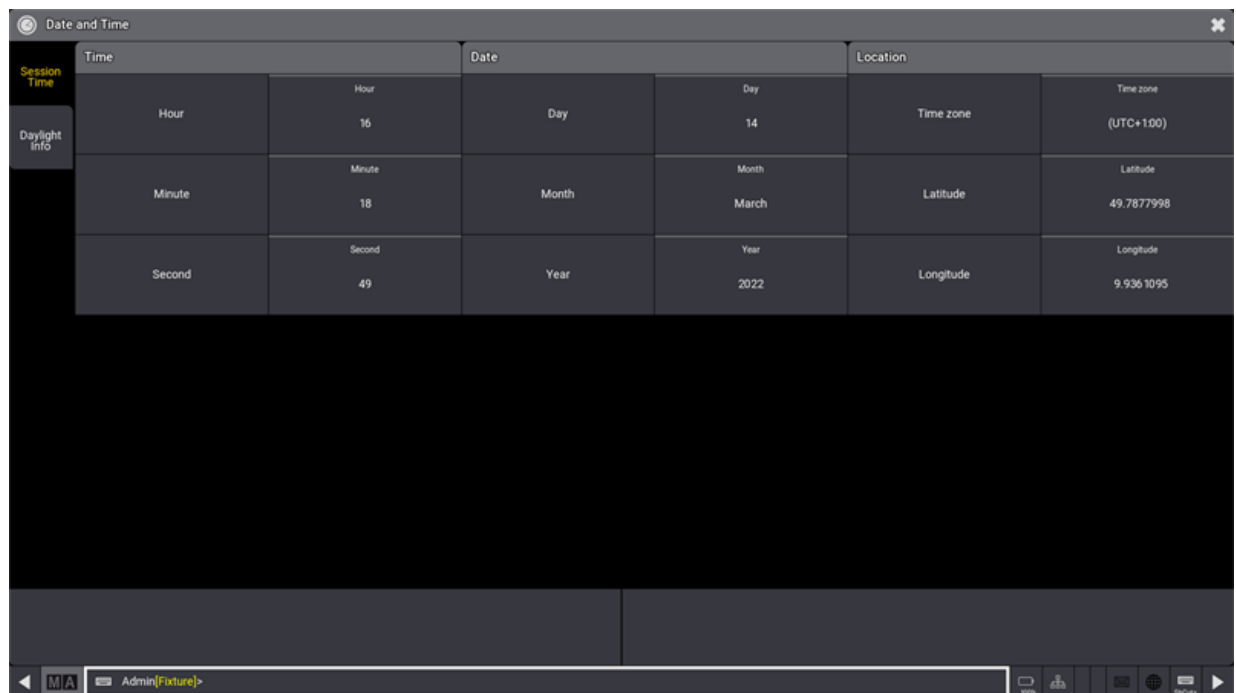
Sets date and time.

To access **Date and Time**:

1. Press **Menu**
2. Tap **Settings**.

A drop-down menu opens.

3. Tap **Date and Time**.
4. The Date and Time window opens.



Open Date and Time



Date and Time

1. To set the date and time, tap the buttons with a title bar on the right of the columns Time and Date. The calculator opens.
2. Enter values and tap **Please**.

Date and time are set.

Location

The time zone refers to the international time standard UTC.

1. To enter the current location, tap each of the buttons –Timezone, Longitude, Latitude – on the right of the column Location. The calculator opens.
2. Enter values and tap **Please**.

Location is set.

Daylight Info

The **Daylight Info** tab, displays yesterday's, today's, and tomorrow's dawn, sunrise, sunset, and dusk.

Session Time	Yesterday	Today	Tomorrow
Dawn	05:29:24	05:27:13	05:25:00
Sunrise	06:38:17	06:36:08	06:33:58
Sunset	18:21:40	18:23:15	18:24:51
Dusk	19:31:44	19:33:23	19:35:02

All times are based on the Location settings in the tab Session Time.
The times are calculated at sea level without any elevation on the horizon.
Maximum precision for all times is +/- 1 minute.

All times are calculated from the Session Time settings. Predictions require location settings. Times are calculated for a flat horizon at sea level. Maximum precision for all times is +/- 1 minute.



42.2. Clock

Open the clock window. For more information, see [Add Window](#).

1. Tap **Tools**.
2. Tap **Clock Viewer**.

The clock window opens.

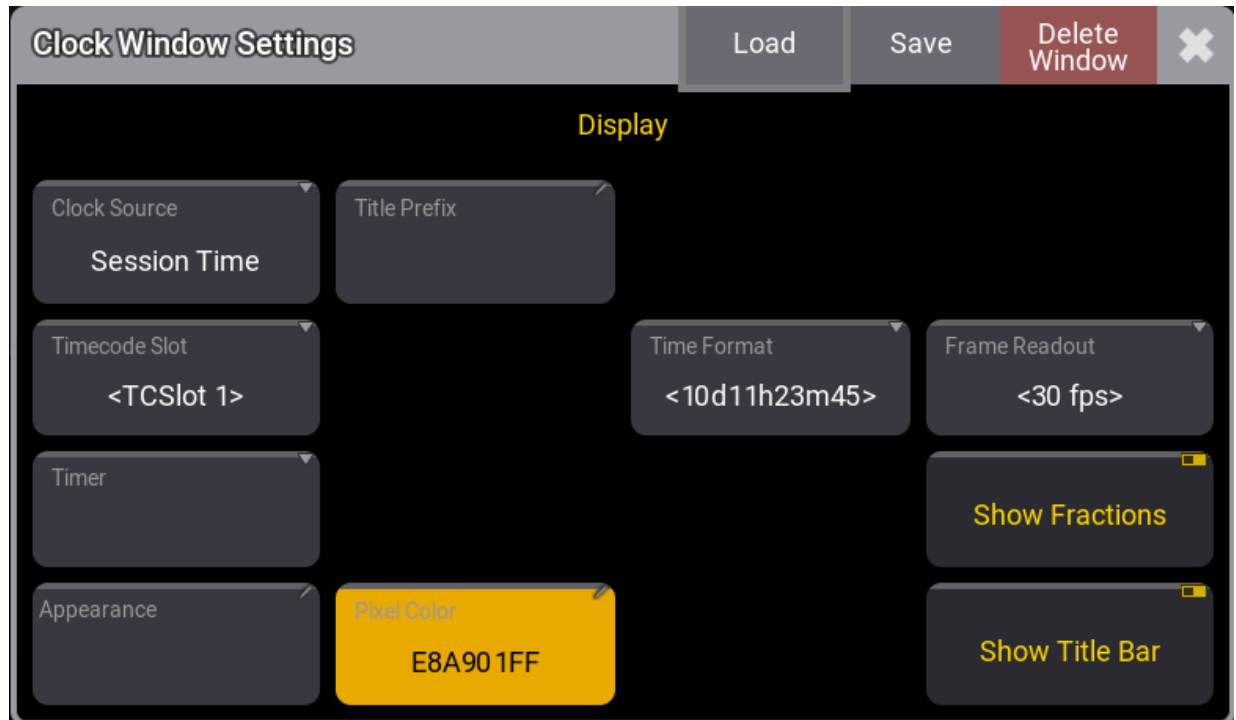


Clock window

- The clock is displayed in yellow digits by default.
- Tap and hold **Clock Source** in the title bar to change the clock source. This will be explained in detail later in this topic.

Clock Window Settings

Tap **MA** in the left corner of the clock viewer title bar to open the clock window settings.



Clock window settings pop-up

1. **Clock Source** : Tap and hold to open the dropdown menu, then select the source.
2. **Timecode Slot** : for more information, see [What are timecode slots.](#)
3. **Timer** : for more information, see [Timers.](#)
4. **Appearances** : for more information, see [Appearances.](#)
5. **Title Prefix** : Tap to open the virtual keyboard, then enter a prefix name for the time zone clock.
6. To learn more about **Time Format** and **Frame Readout**, see [Single User and Multi User Systems - User Settings.](#)
7. **Show Fractions** : Tap to toggle on or off the fractions of a second when the **Clock Source** is set to **Timer** or frames when the **Clock Source** is set to **Timecode**.
8. **Show Title Bar** : Tap to show or hide the title bar of the clock window.

Time Zone Clock

- Set the **Clock Source** to Time Zone Clock.

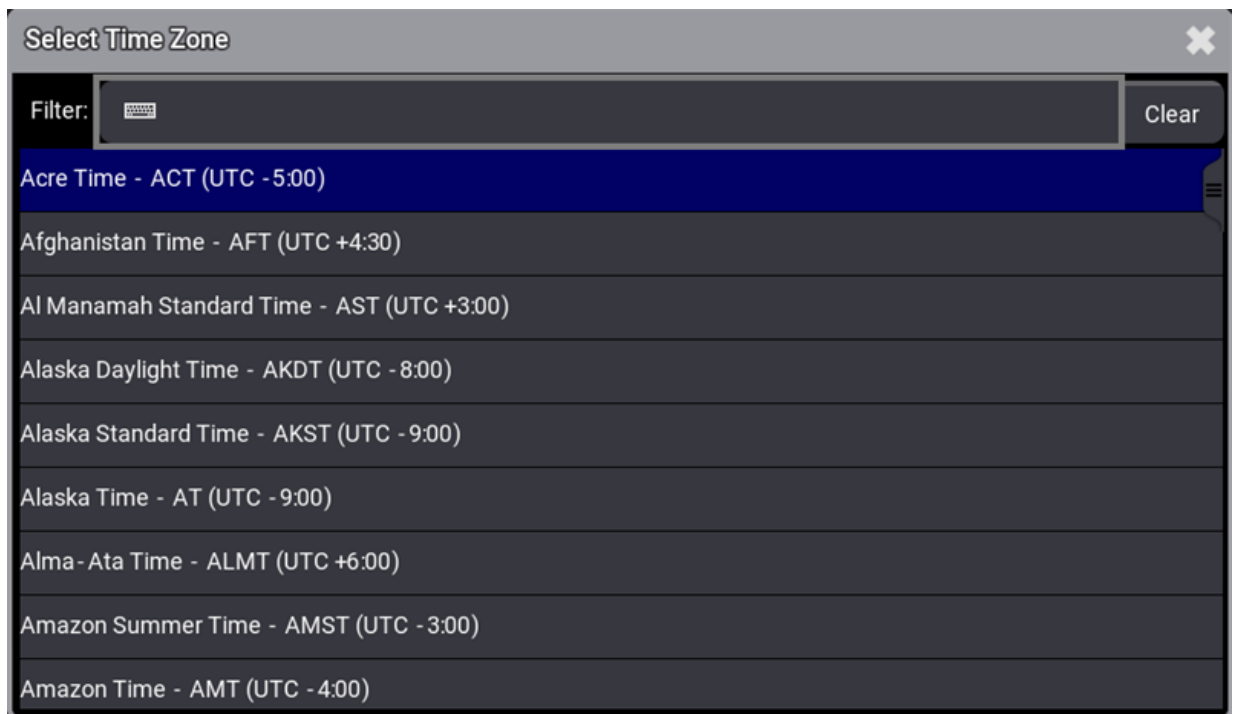


Time Zone Clock window

- Tap **Time Zone**.

The Select Time Zone pop-up opens.

Select the time zone.



Select Time Zone pop-up



Hint:

For more information on how to set the **Session Time**, see **Date and Time**.

Timecode Slot

To set the clock to a timecode slot:

1. Tap and hold **Clock Source** in the title bar, then tap **Timecode**.
2. Tap and hold **Timecode Slot** in the title bar, then tap the desired timecode slot.
To use the timecode slot selected in the **Timecode Slots** pool, tap **<Selected>**.

The clock displays the selected timecode slot.



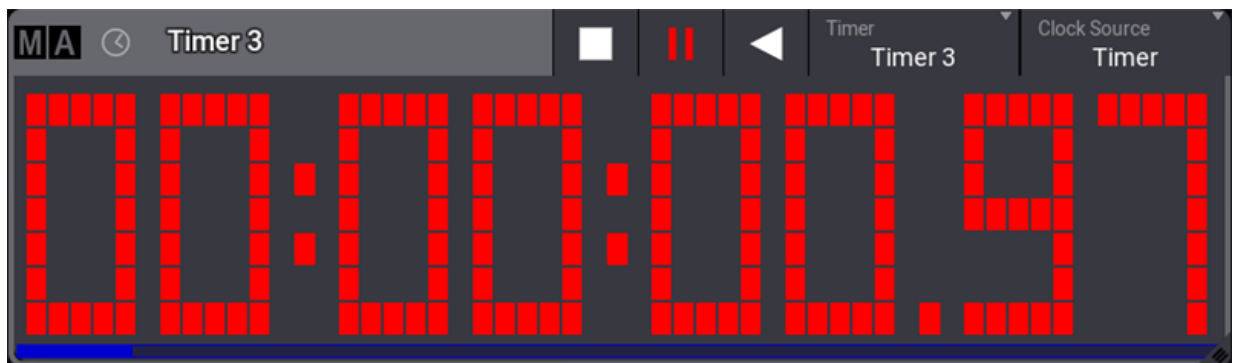
Timecode Slot clock window

Timer


To set the clock to display a timer:

1. Tap and hold **Clock Source** in the title bar, then tap **Timer**.
2. Tap and hold **Timer** in the title bar, then tap the desired timer.
To use the timer selected in the **Timer** pool, tap **<Link Selected>**. For more information, see [Timers](#).


The clock displays the selected timer.



Timer clock window

Tap  to start the countdown for the displayed timer. A descending progress bar is displayed at the bottom of the window.

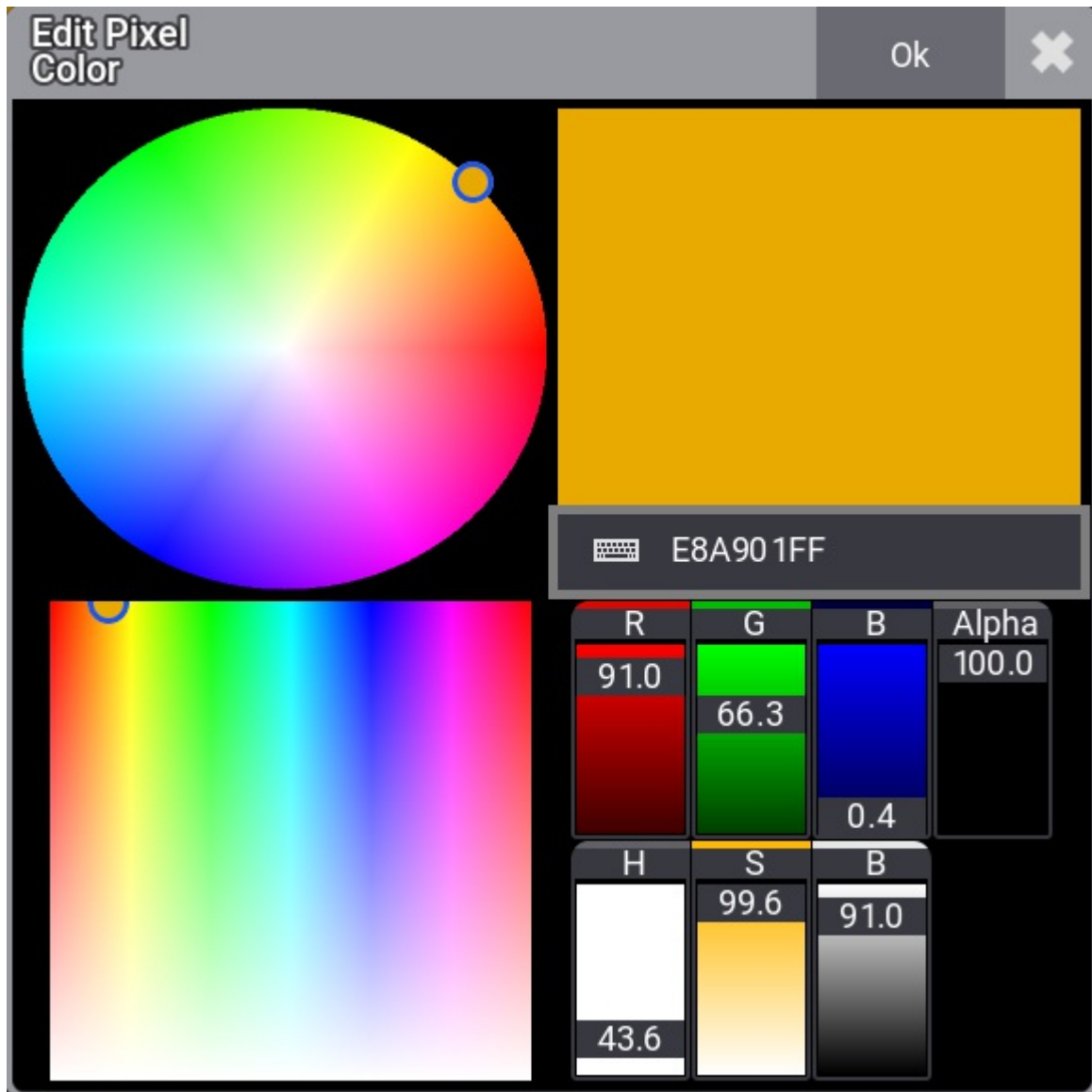


Tap  to pause the countdown.

Tap  to stop the clock, this will also reset the **Countdown Time** set for this timer. For more information, see [Timers](#).

Edit Color of Digits

1. To change the color of the digits, tap **MA** in the top left corner, then tap **Pixel Color**.
2. The pop-up **Edit Pixel Color** opens:



Edit Pixel Color pop-up

3. Tap to select a color in the color picker.
4. Tap **Ok**.

The color of the digits is modified.



Appearance as Background Color

1. Create an appearance in the appearance pool.
For more information, see **Create Appearances**.
2. Press **Assign**, tap an appearance in the pool, and tap the clock title bar.
For more information, see **Use Appearances**.

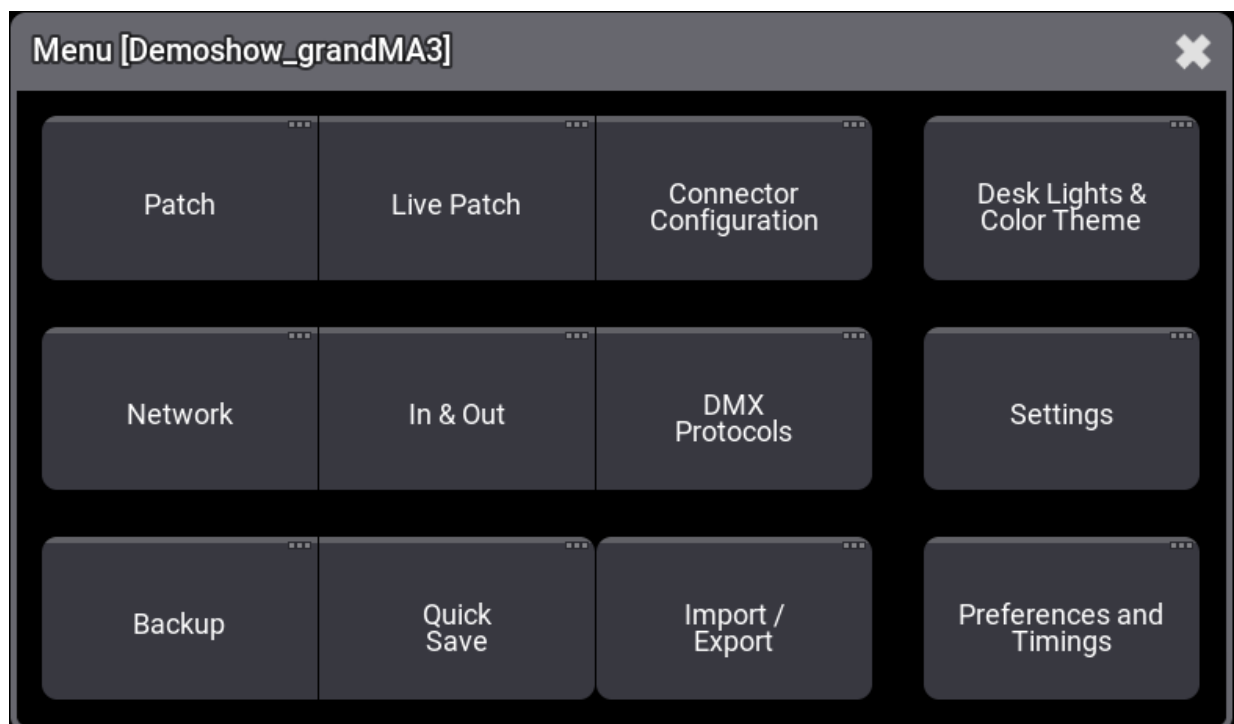
42.3. Desk Lights

This topic describes the setting of desk lights.

To learn more about color themes, see [Color Themes](#).

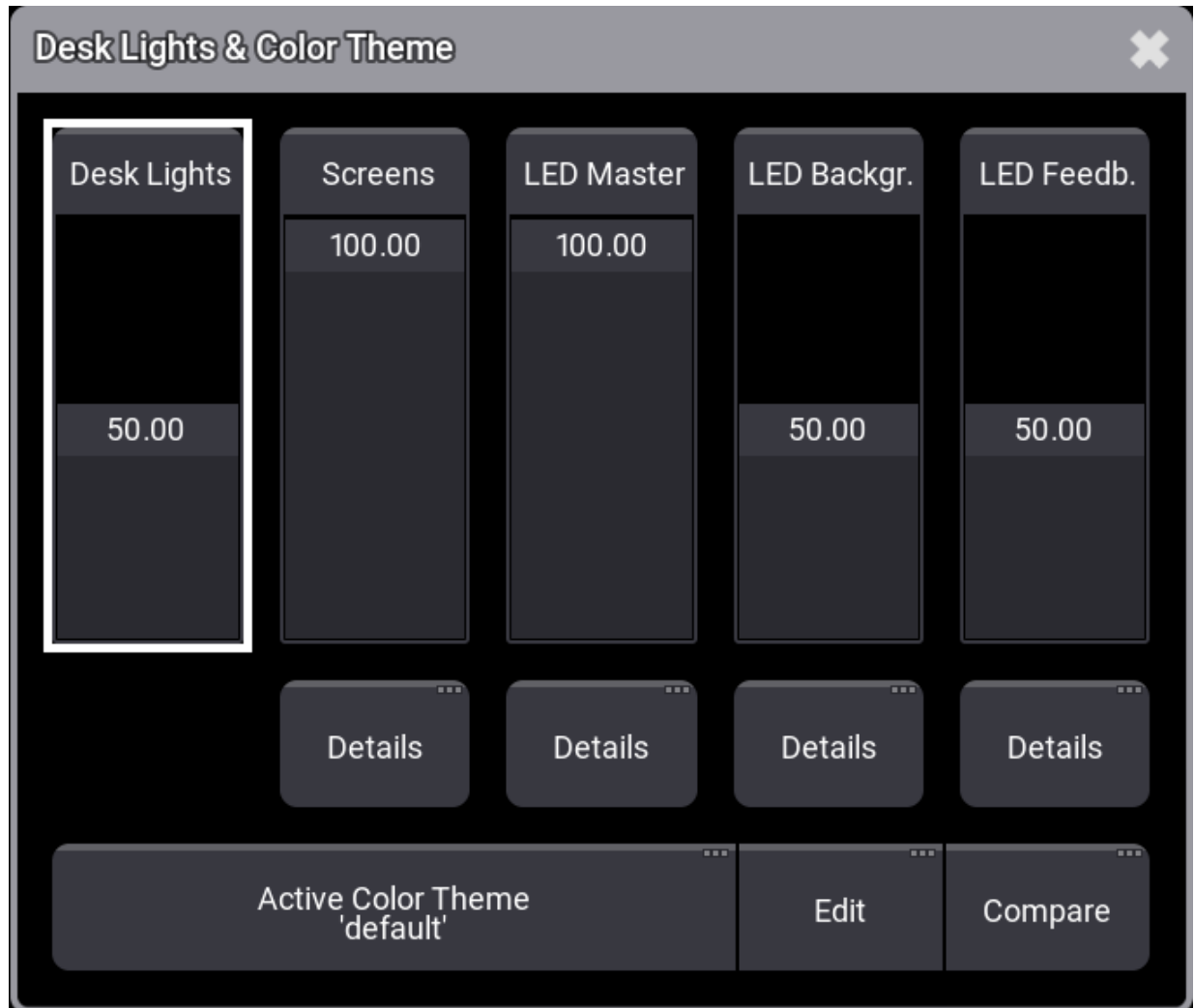
To access the Desk Lights Color Theme menu:

1. Press **Menu**
2. Tap **Desk Lights & Color Theme**



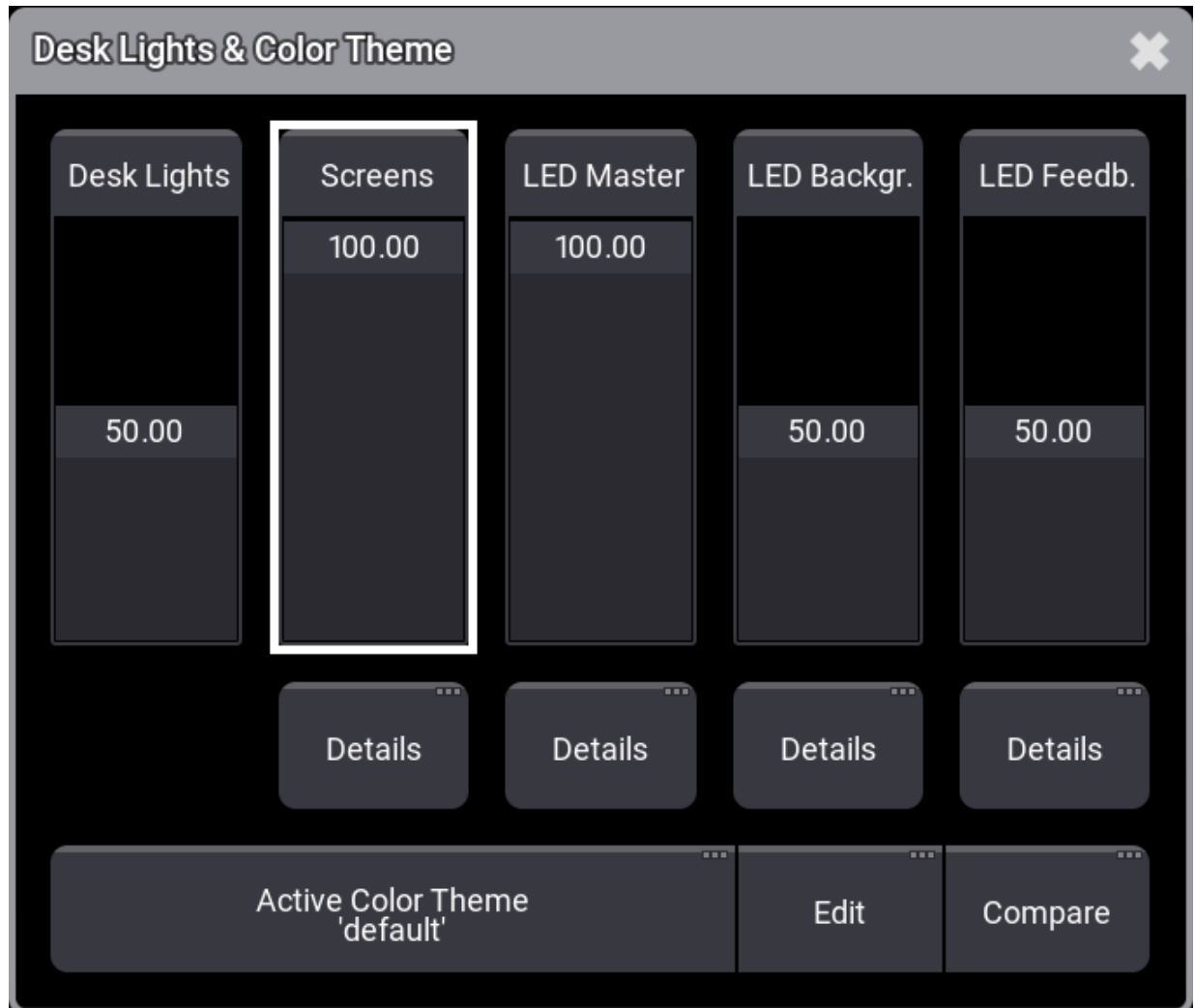
Add window dialogue

The Desk Lights Color Theme menu opens.



Desk Lights & Color Theme menu

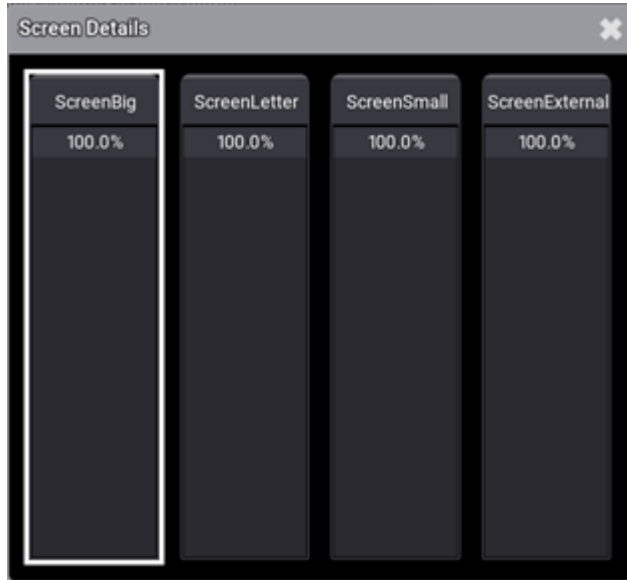
- To adjust the desk light intensity, move the Desk Lights fader up or down.
- To adjust the general brightness of all screens, move the Screens fader up or down.



Screen brightness fader

- To adjust the brightness of individual screens, tap **Details** below the Screens fader.

The Screen Details pop-up opens.



Screen Details pop-up

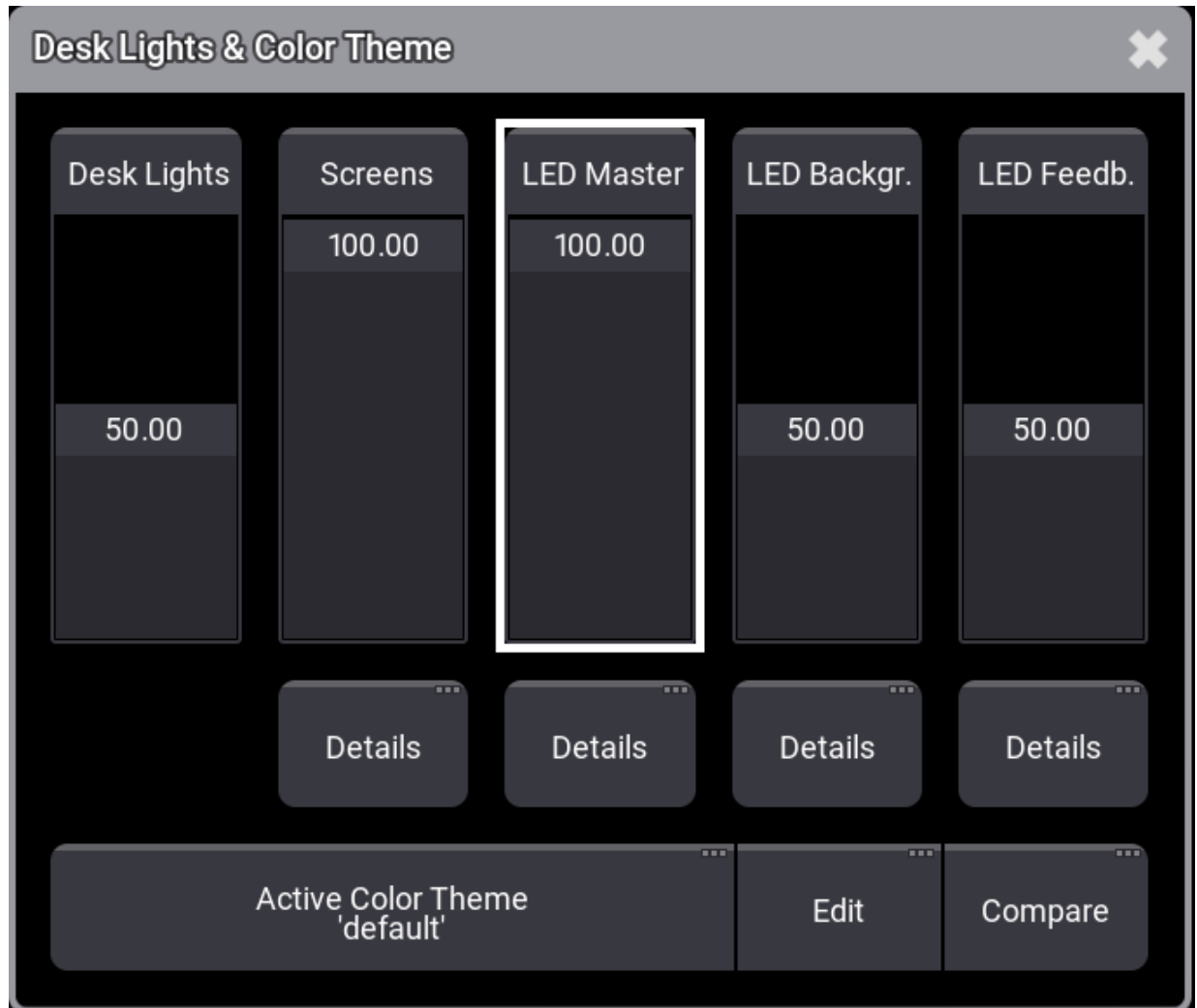
- To adjust the brightness of the desired screen, move the respective Screen fader up or down.



Restriction:

The Screen External fader currently has no function.

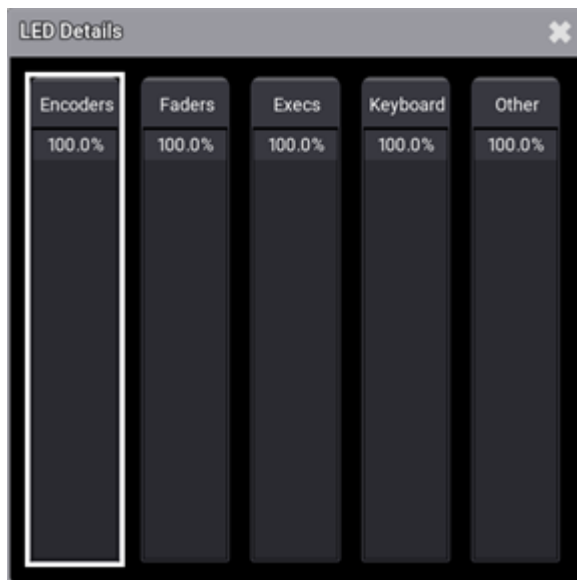
- To adjust the general intensity of all LEDs, move the LED Master fader up or down.



LED Master fader

- To adjust the LED intensity of the individual elements, tap **Details** below the LED Master fader.

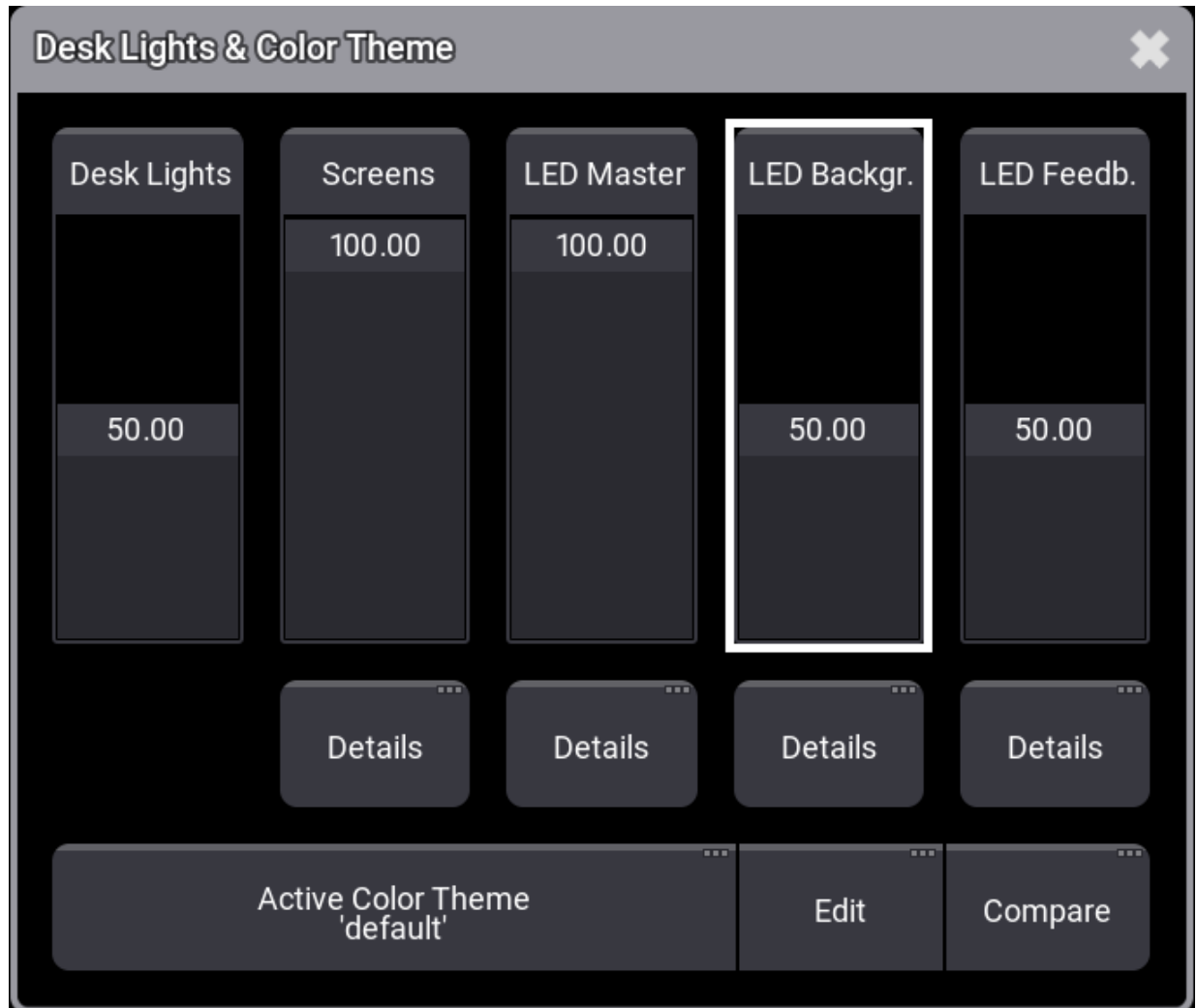
The LED Details pop-up opens.



LED Details pop-up

To adjust the intensity of the desired element, move the respective LED fader up or down.

- To adjust the general LED Background intensity, move the LED Background fader up or down.



LED Background fader

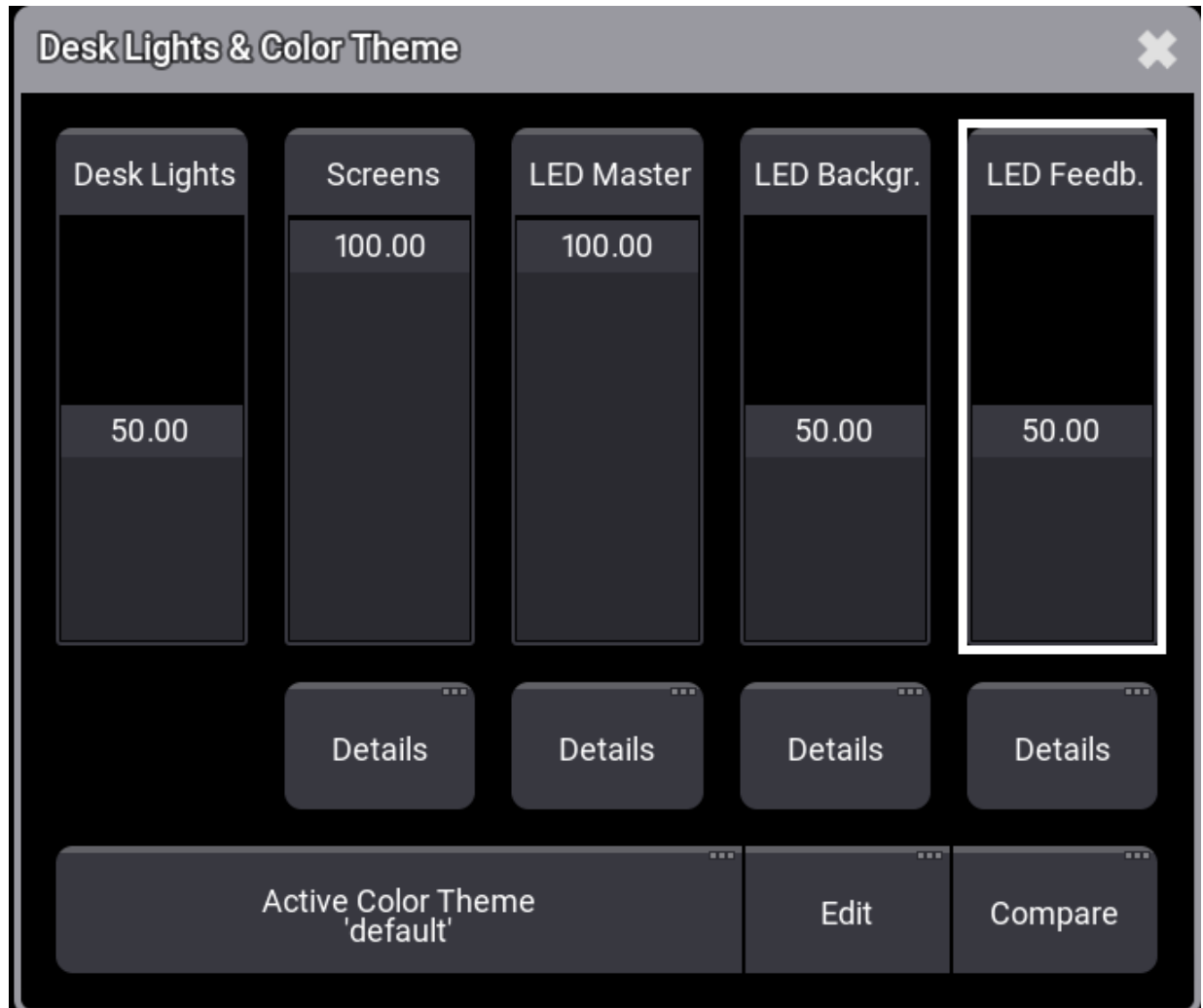
- To adjust the LED background intensity of the individual elements, tap **Details** below the LED Master fader.

The LED Background Details pop-up opens.



LED Background Details pop-up

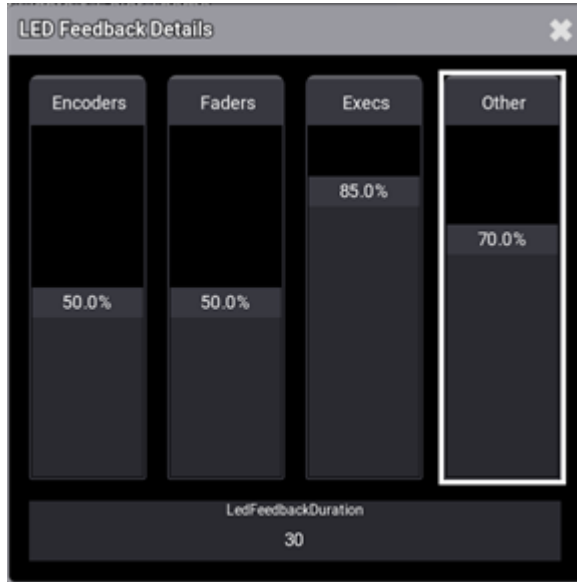
- To adjust the general LED Feedback intensity, move the LED Feedback fader up or down.



LED Feedback fader

- To adjust the LED Feedback intensity of the individual elements, tap **Details** below the LED Feedback fader.

The LED Feedback Details pop-up opens.



LED Feedback Details pop-up



42.4. System Information



System Info is a mixture of system and performance monitoring of the console.

To add a System Info window, follow the instructions under **Add windows**.

In the Add Window pop-up:

1. Tap **More**.
2. Tap **System Info**.

The System Info window opens.

M A  System Info		Details 
Serial Number	155 1000000 - 9000 10	
Host Type	onPC	Wing - onPC
Host OS	Windows	
Production Date	19.09.2019	
Software Version	1.9.2.0	
Release Type	Software: Release	Code: Release
Build Date	Apr 4 2023	06:39:38
Git Head	UnknownBranch	
Git Hash	8e230cf0a38091fc05839f0329a	2023-04-03 16:27:35 +0200
Parameter Count	Used: 931	Available: 4096
ShowData	Used: 39.7 MB	Maximum: 10.0 GB
Media Pools	Used: 648.4 KB	Maximum: 200.0 MB

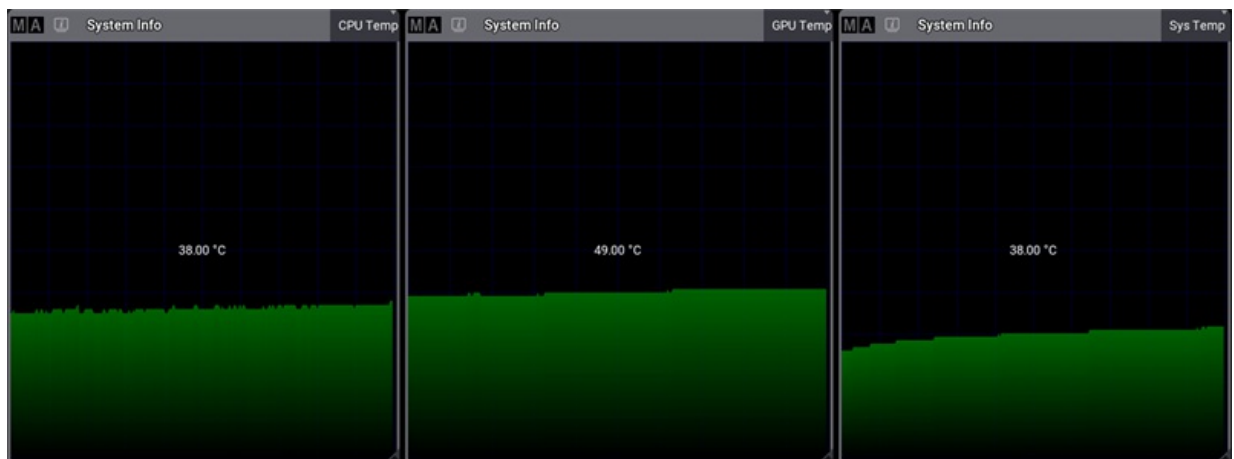
System Info – details on onPC

To adjust the System Info window settings, follow the instructions under **Window Settings**.

Temperature Check

CPU, GPU, and system temperatures can be displayed on:

- grandMA3 full-size
- grandMA3 light
- grandMA3 replay unit
- grandMA3 compact XT



Example of temperature windows

The grandMA3 compact can only display CPU and system temperatures.

These products are not able to display temperature information:

- grandMA3 extension
- grandMA3 processing unit
- grandMA3 nodes
- grandMA3 viz-key
- All grandMA3 onPC solution devices

Fan Check

The fan view in the System Info window displays the system/CPU cooling fan speed in correlation to its maximum speed.

Fan speeds can be displayed on:



- grandMA3 full-size
- grandMA3 light
- grandMA3 replay unit
- grandMA3 compact (XT)

These products are not able to display fan speed information:

- grandMA3 extension
- grandMA3 processing unit
- grandMA3 nodes
- grandMA3 viz-key
- All grandMA3 onPC solution devices

System Info is divided into different views, which can be selected using the button in the title bar on the right-hand side.

Tap and hold the button to toggle between:

- **Details** : Details displays other relevant system info such as the serial number or the build date.
- **Realtime** : Realtime displays the workload of the system in milliseconds per DMX frame.
- **Timing** : Timing displays the time that is required to render screens on the console. MainLoop, Swap, and End show the internal processing and in which displays it takes place.
- **CPU** : CPU displays the workload of the main processor.
- **Memory** : Memory displays the RAM of the system in GB.
- **CPU Temp** : CPU Temp displays the current temperature of the console.
- **GPU Temp** : GPU Temp displays the current temperature of the graphics processing unit of the console.
- **Sys Temp** : Sys Temp displays the current temperature of the central computer board.
- **Fan** : Fan displays the rotational speed of the fan of the cooling system in RPM.
- **HDD** : HDD lists relevant information about the available hard drives.



42.5. System Monitor

The system monitor is a window that can be created like any other window using the [Add Window pop-up](#).

```
M/A System Monitor
MainTask 6.907s : SystemMessage::Send ShowStatusChanged
MainTask 6.907s : Addon got new show status ShowUploading
MainTask 6.907s : Addon got new host status GlobalMaster
MainTask 6.907s : Addon got data valid status 1
MainTask 6.910s : Sending fast dummy
MainTask 6.910s : *****
MainTask 6.910s : * New data status 'Show loaded'
MainTask 6.910s : *****
MainTask 6.910s : SystemMessage::Send ShowStatusChanged
MainTask 6.910s : Addon got new show status ShowLoaded
MainTask 6.910s : Addon got new host status GlobalMaster
MainTask 6.910s : Addon got data valid status 1
MainTask 6.912s : ***** Output table *****
MainTask 6.912s : Slot 0 Subfixtures= 171 Parameters= 1039
MainTask 6.912s : SlotMask 00000001 MyMask 00000000
MainTask 6.914s : RT Channel distribution has changed. Invalidating DMX
MainTask 15.909s : OK:Set ScreenContent 1.1.1 Property "ActiveDisplay" "Wav
MainTask 18.030s : OK:Set ScreenContent 1.1.1 Property "ActiveDisplay" "Sou
WorldServer 18.179s : SecureProtocolWorld switching to FastTimeout
MainTask 21.535s : OK:Set ScreenContent 1.1.1 Property "ActiveDisplay" "Bea
MainTask 25.404s : Sound: opened record for device: Mikrofon (Logitech Stre
MainTask 25.404s : OK:Edit Root 2.2 Property "OnPCAudioInDeviceName"
MainTask 39.617s : Sound: opened record for device: VoiceMeeter Aux Output
MainTask 39.617s : OK:Edit Root 2.2 Property "OnPCAudioInDeviceName"
```

System Monitor

It shows what is happening at the station. This includes feedback on user commands. It is a log of the different things happening in the background. It also shows warnings, errors, and changes to the system.

It is a debugging window that can provide useful information if there are any problems with the system.

There are only two settings for the window. Appearance and Front Size. Both are described in the [Window Settings topic](#).

The default background is black. The bottom line shows the latest thing that happened.

If the system monitor is scrolled up, then the background changes to a red color to indicate that is not the latest data, and the auto-scrolling function that automatically shows the latest info is not active. Scroll back down to the bottom to reactivate the auto-scroll function.

42.6. Info Window

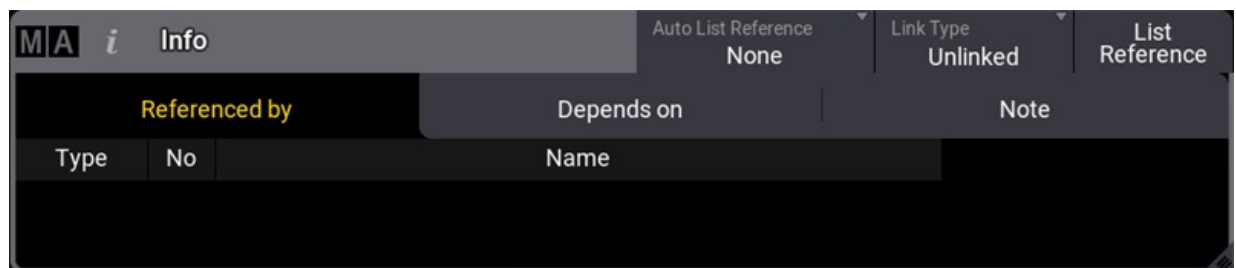
The info window is a helpful tool to show operators **references**, **dependencies** and **notes** for objects in the show file.

For more information, see [ListReference](#) keyword.

To open the Info window:

- Open the Add Window dialogue, tap **More**, then tap **Info**.

The Info window opens.



Info window

Referenced by (x) -tab displays the following information:

- **(x)**, displays the number of references for this object.
- **Type**, displays the targeted object reference type.
- **No**, displays the targeted object reference number.
- **Name**, displays the targeted object reference name.

Depends on (x) -tab displays the following information:

- **Type**, displays the type of the targeted object dependencies.
- **No**, displays the number of the targeted object dependencies.
- **Name**, displays the name of the targeted object dependencies.

The **Note (x)** -tab displays the notes of a object.

- Text can be changed or added in the Note-tab directly. Changes are save within one second after the last character was entered.

For more information about Notes, see [Notes](#) topic.

List Reference

To list the references of an object, for example a sequence pool object:

1. Open the grandMA3 demo show file, the Sequence pool and the Info window.
2. Tap **List Reference** in the title bar of the Info window or type **ListReference** in the command line.
3. Tap **Look** in the Sequence pool. The related information about the selected sequence pool object is shown in the Info window.



Info window displaying information of Sequence 1.



Important:

The results of **List Reference** for a targeted object will stay visible until you perform a new search or change the view. It can not be cleared or deleted.

Auto List Reference

The **Auto List Reference** button defines if List Reference is set automatically to a specific object:

- **None:** References are not listed automatically.
- **Selected Sequence:** List References automatically lists the selected sequence in the Info window.



Hint:

When Auto List Reference is set to Selected Sequence, the List Reference button will be grayed out.

Link Type

To specify the displayed information in the Info window, **Link Type** can be set to the following values:

- **Unlinked:** Displays the note of the selected object (in this case sequence 1) via the List Reference button.
- **Current Cue:** Displays the note of the currently active cue.
- **Next Cue:** Displays the note of the cue that will be played back next when executing **Go+** again on the sequence. When using **Load**, the note of the loaded cue will be displayed.

Values are named by the children after selecting an object via the List Reference button.

For example, to set different Link Types in the Note tab:

1. Store two cues into Sequence 1. For more information to store cues, see **Store cues**.
2. Add notes to cues. For more information to store cues, see **Notes** topic.
3. Tap **List Reference** and then tap **Sequence 1** in the sequence pool.
4. Use **Go+** to go through the cues.



43. Sound

The grandMA3 system can receive a sound signal. This signal can be used in Phasers, and different **Sound Channels** can be used to give value to attributes.

The **Sound** window displays an incoming audio signal as a raw waveform. This window also breaks down the strength of that signal into several bands of a few different widths. The **Sound** window also displays calculations performed by the console to find steady beats within the signal.

Requirement:

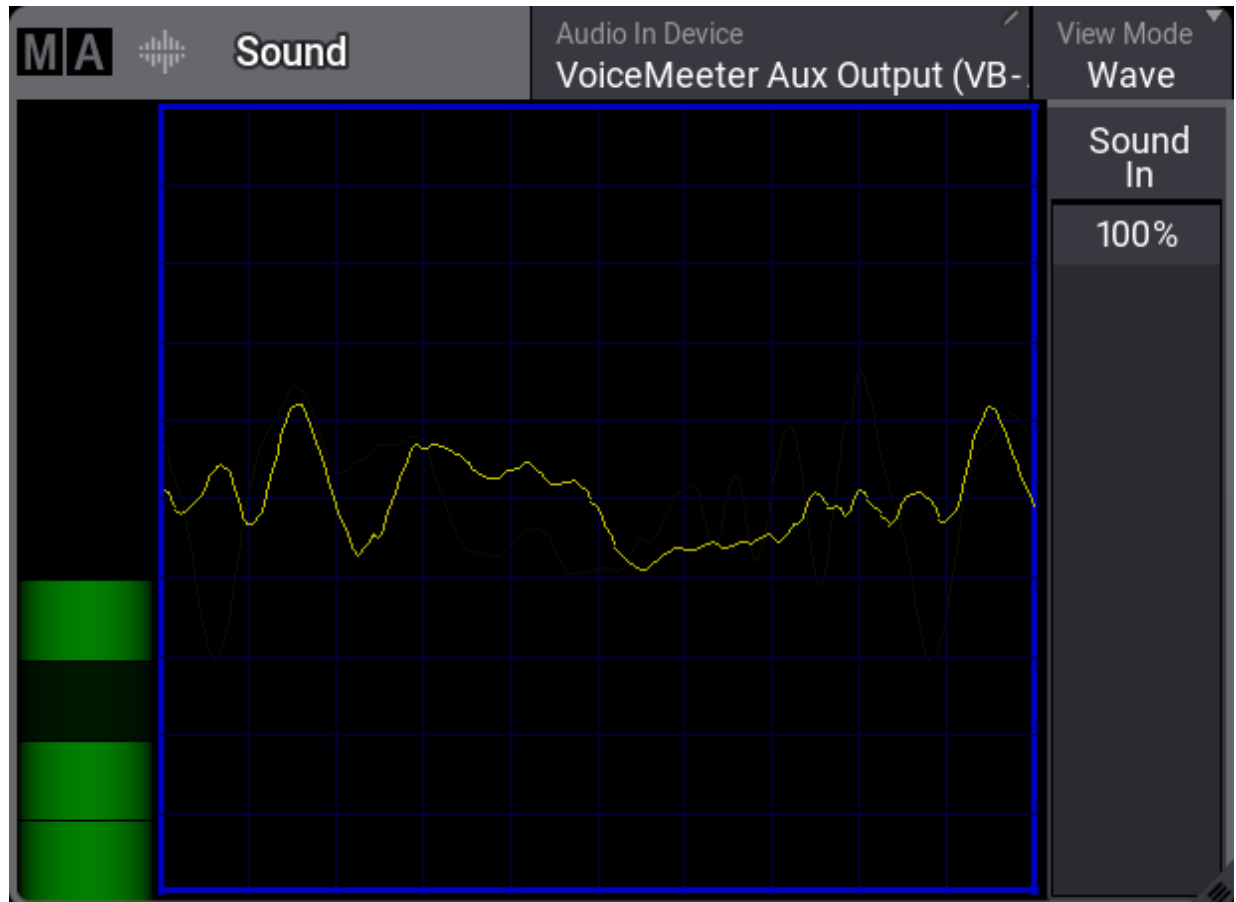
Active sound input is required in order for the console to display useful information in the **Sound** window. For information about connecting an audio input to a console, see the **Connect Audio In** topic. The grandMA3 onPC software uses any sound input hardware designated by the operating system.

The **Add Window** pop-up includes the **Sound** window under the **More** tab. For more information about the **Add Window** pop-up, see the **Add Windows** topic.

The right corner of the **Sound** window title bar includes a **View Mode** button. Tap this button to cycle through available display mode options, or tap and swipe to open a pop-up menu containing all of the available display mode options for the **Sound** window.

On the grandMA3 onPC, there is another button in the title bar called **Audio In Device**. This is a list of available sound devices on the computer that can be used as sound input devices.

Wave View Mode



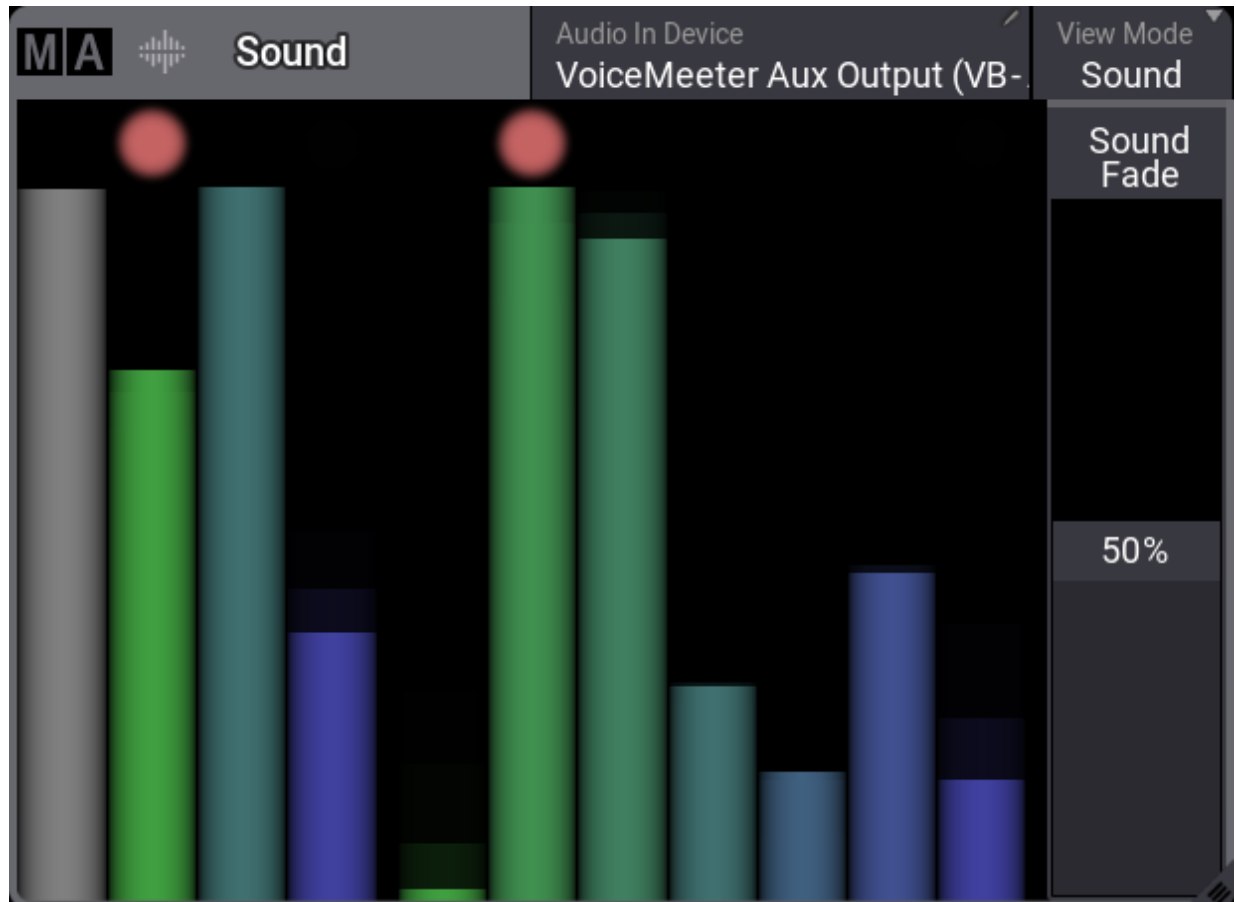
Sound window in Wave view mode

The left side of the window includes a simple VU meter shown as a vertical bar.

The main area of the window shows the raw waveform of the incoming audio signal.

The right side of the window includes quick access to the **Sound In** master. Adjusting the **Sound In** master changes the volume of the input signal. This change is immediately visible in both the VU meter and the waveform. For more information about the **Sound In** master, see the [Grand Masters](#) topic.

Sound View Mode



Sound window in Sound view mode

The **Sound** view mode of the **Sound** window displays the incoming audio signal as a series of bars representing the volume of different bands of frequencies. When a given bar reaches a peak, a dot appears above the bar. This dot represents a possible sound trigger based on that frequency band.

The eleven bars shown in the window break down the frequencies of the audio signal in three different ways, displaying all three breakdowns simultaneously. The first bar represents the volume of the whole signal. The next three bars divide the signal into **Bass**, **Mid**, and **High** frequencies. And, the remaining seven bars divide the frequencies equally into narrower bands.

These eleven bars relate directly to the first eleven **Sound Channels** and inversely to the remaining **Sound Channels**. Available **Sound Channels** include:

1. **All**
2. **Bass**
3. **Mid**
4. **High**
5. **Band1**
6. **Band2**

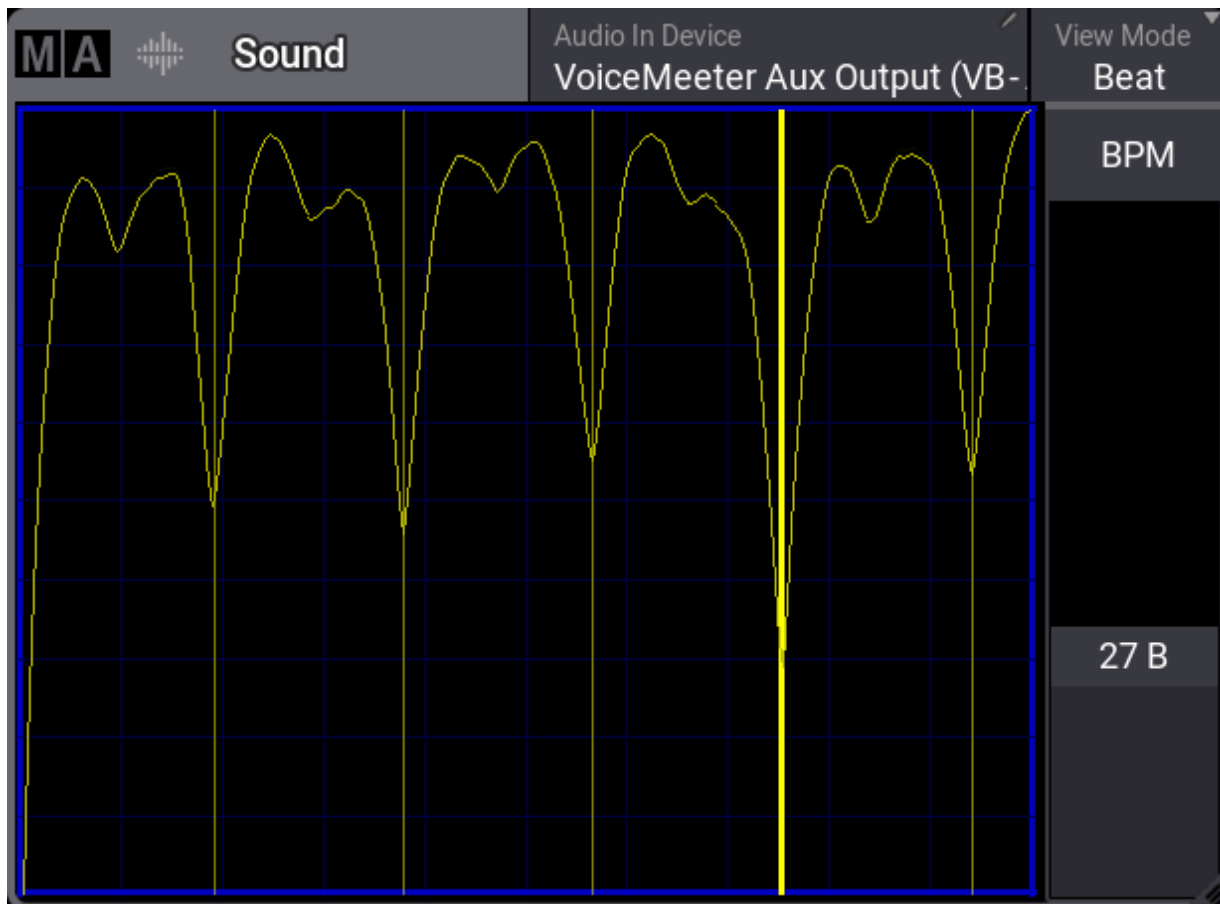


7. **Band3**
8. **Band4**
9. **Band5**
10. **Band6**
11. **Band7**
12. **InvAll**
13. **InvBass**
14. **InvMid**
15. **InvHigh**
16. **InvBand1**
17. **InvBand2**
18. **InvBand3**
19. **InvBand4**
20. **InvBand5**
21. **InvBand6**
22. **InvBand7**

Set the value of any parameter to follow one of the **Sound Channels** in the command line, using the **SoundChannel** keyword, or the **Sound Codes** tab of the **Calculator**.

The right side of the window includes quick access to the **Sound Fade** master. Adjusting the **Sound Fade** master changes the fall-off speed of all of the channels. While a lower number results in more accurate tracking of quickly changing dynamics, the response may appear undesirably erratic. A higher value results in a smoother response. For more information about the **Sound Fade** master, see the **Grand Masters** topic.

Beat View Mode



Sound window in Beat view mode

The **Beat** view mode of the **Sound** window displays the incoming audio signal after some additional processing. This processing looks for repeating beats. The processed signal appears as a series of hills and valleys. Deeper valleys form where louder pulses in the incoming signal line up more consistently over time.

Dim, yellow, vertical lines appear in the valleys with the most consistent repeated pulses. These lines represent beat candidates, which the **BPM** speed master can follow.

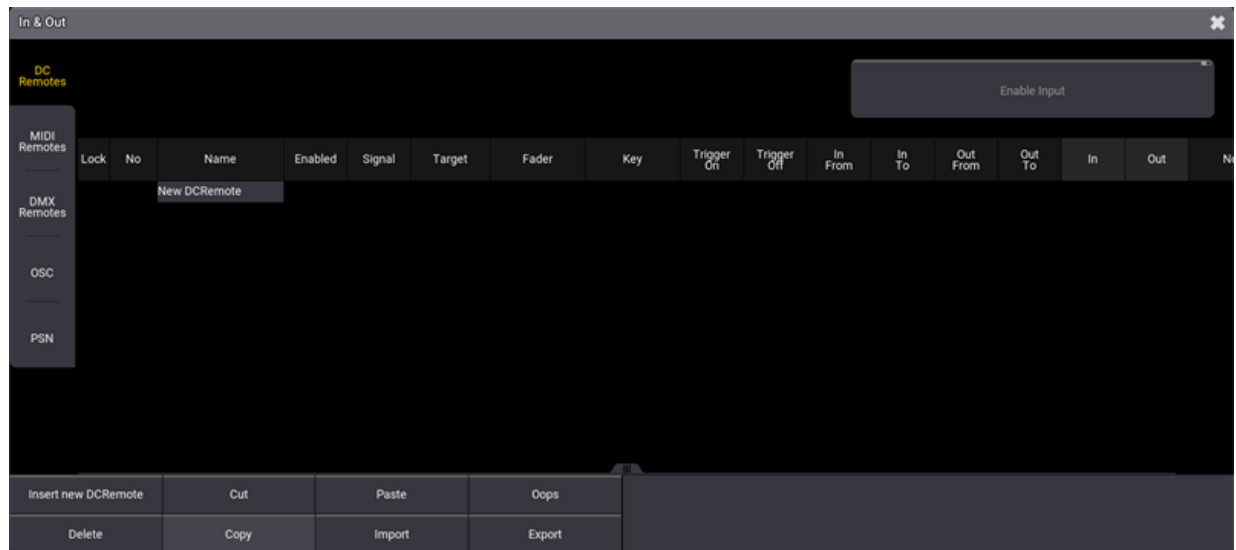
A bold, yellow, vertical line appears when the **BPM** master locks onto a beat. The **BPM** master adjusts automatically to follow any changes detected in the incoming signal.

The right side of the window includes quick access to the **BPM** speed master fader. For more information about the **BPM** speed master, see the [Speed Masters](#) topic.

44. Remote In and Out

Remote Inputs are handled through the In & Out menu: DC Remotes, MIDI Remotes, DMX Remotes, OSC, and PSN.

- To open the **In & Out** window, press **Menu** and then tap **In & Out**.



In & Out window

There are five tabs: **DC Remotes**, **MIDI Remotes**, **DMX Remotes**, **OSC**, and **PSN**.



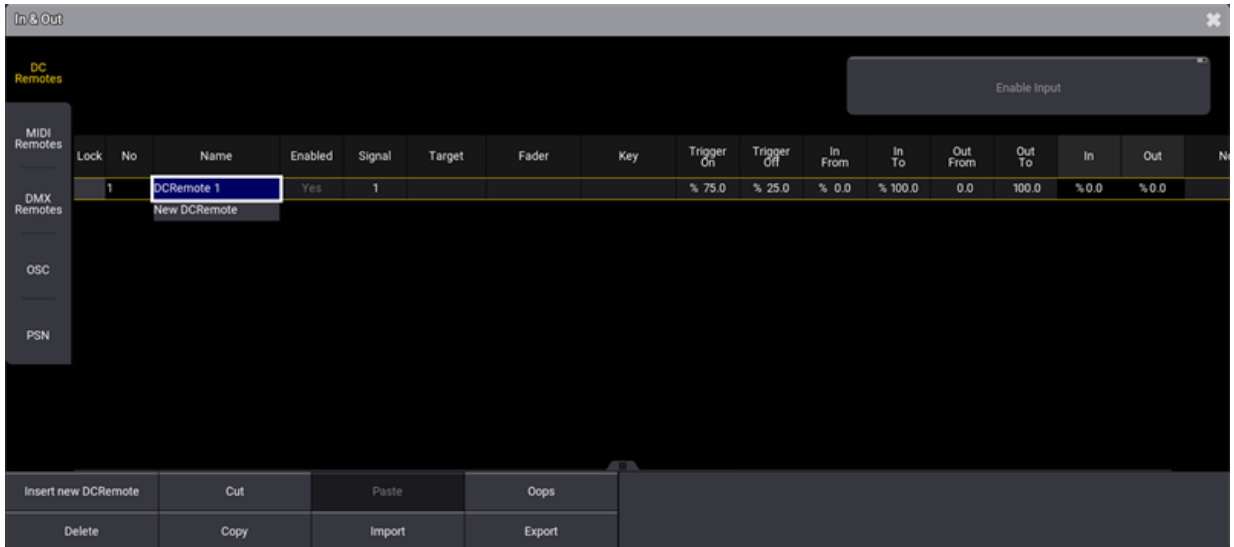
Hint:

The different readout options take effect on the Trigger On/Off, In From/To, Out From/To values.



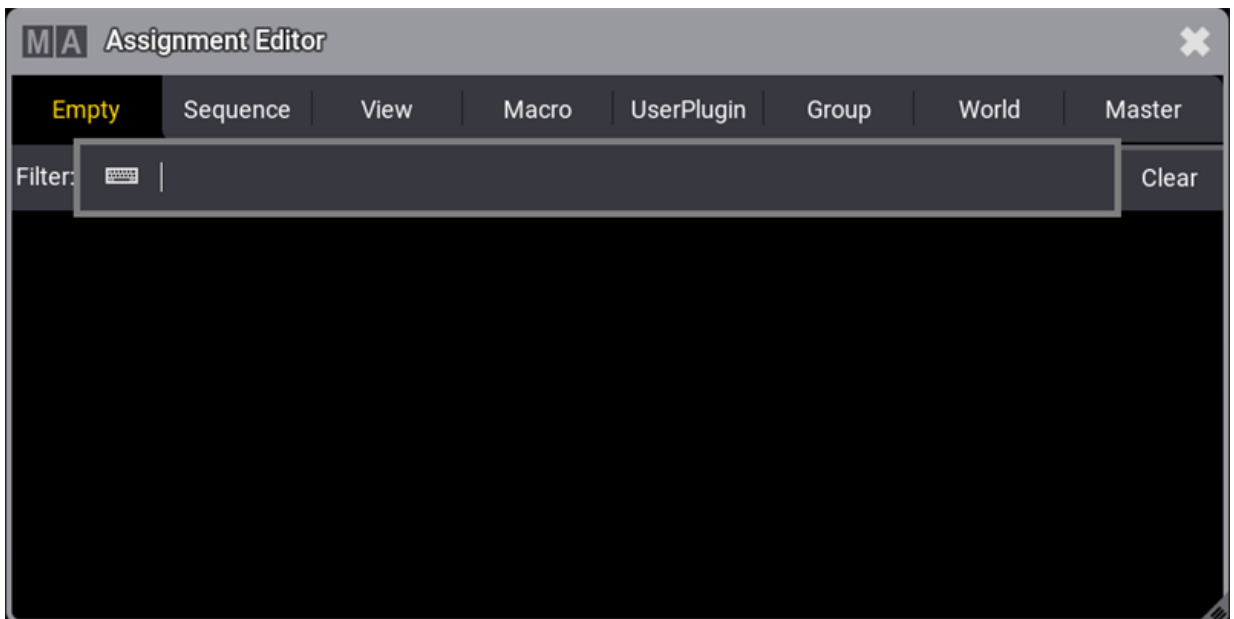
Common procedures and pop-ups for the DC Remotes, MIDI Remotes, and DMX Remotes tabs are described below:

1. To add a new element, for example DC Remote, tap **Insert new DCRemote**.



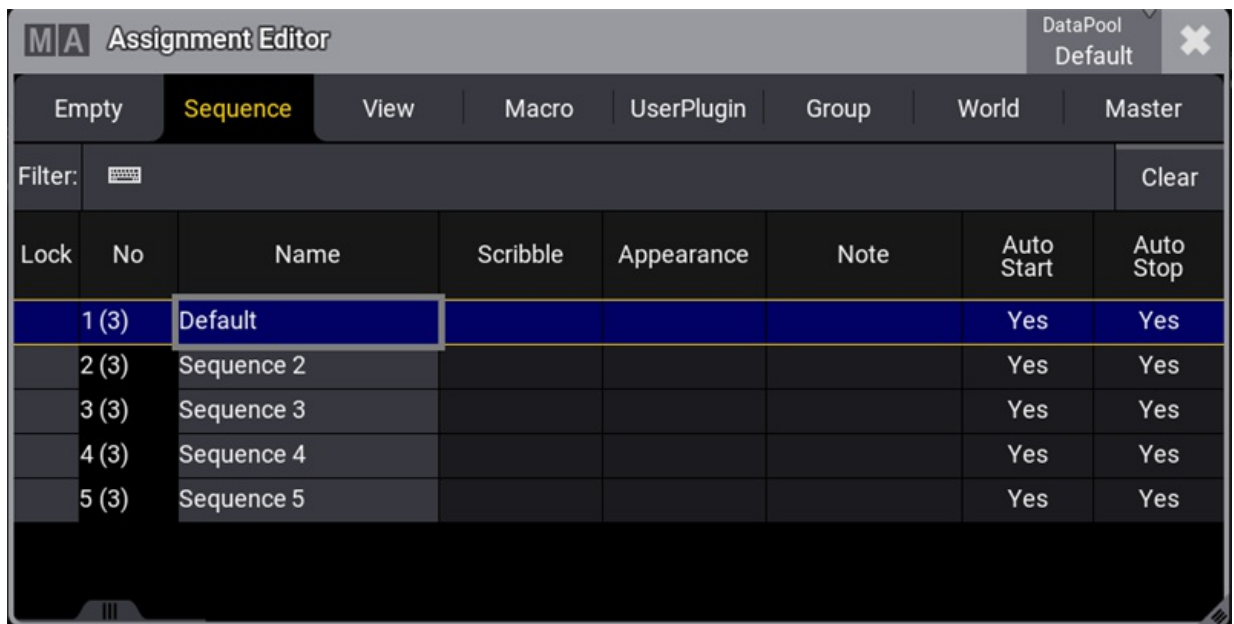
New DCRemote

2. To enable or disable the Remote Input, right-click or tap and hold **Enabled** to toggle between Yes and No.
3. To assign the target, right-click or tap and hold **Target**. The Assignment Editor window opens:

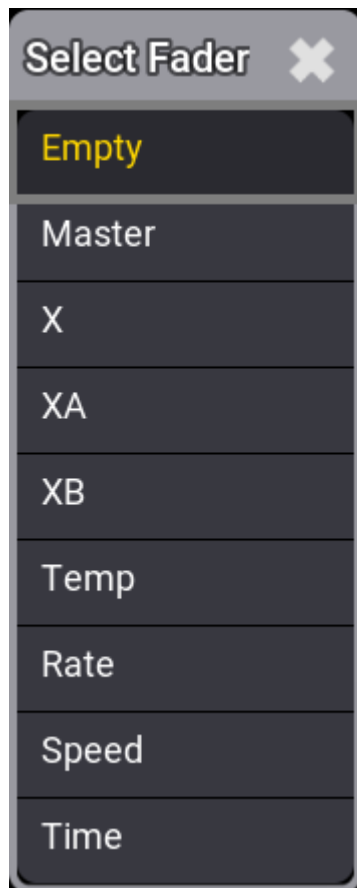


Assignment Editor

4. Select the target, for example a sequence. The target is selected and the editor closes.

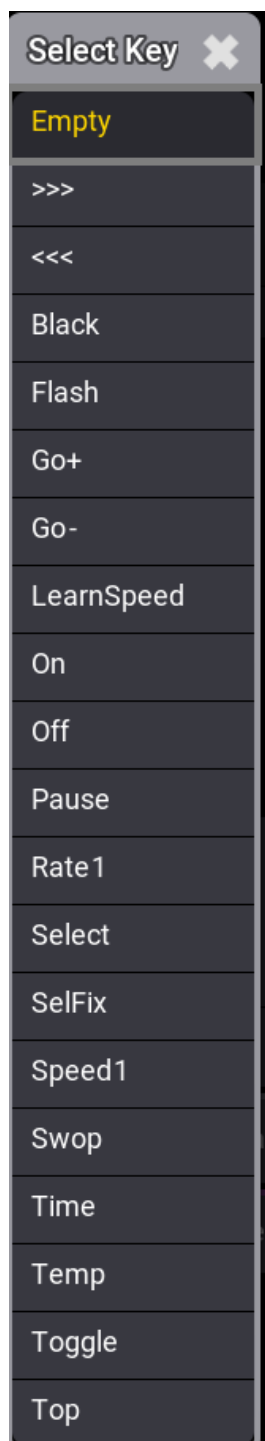


5. To select a fader function, right-click or tap and hold **Fader**. The Select Fader pop-up opens:



Select Fader pop-up

6. To select a key function, right-click or tap and hold **Key**. The Select Key pop-up opens:



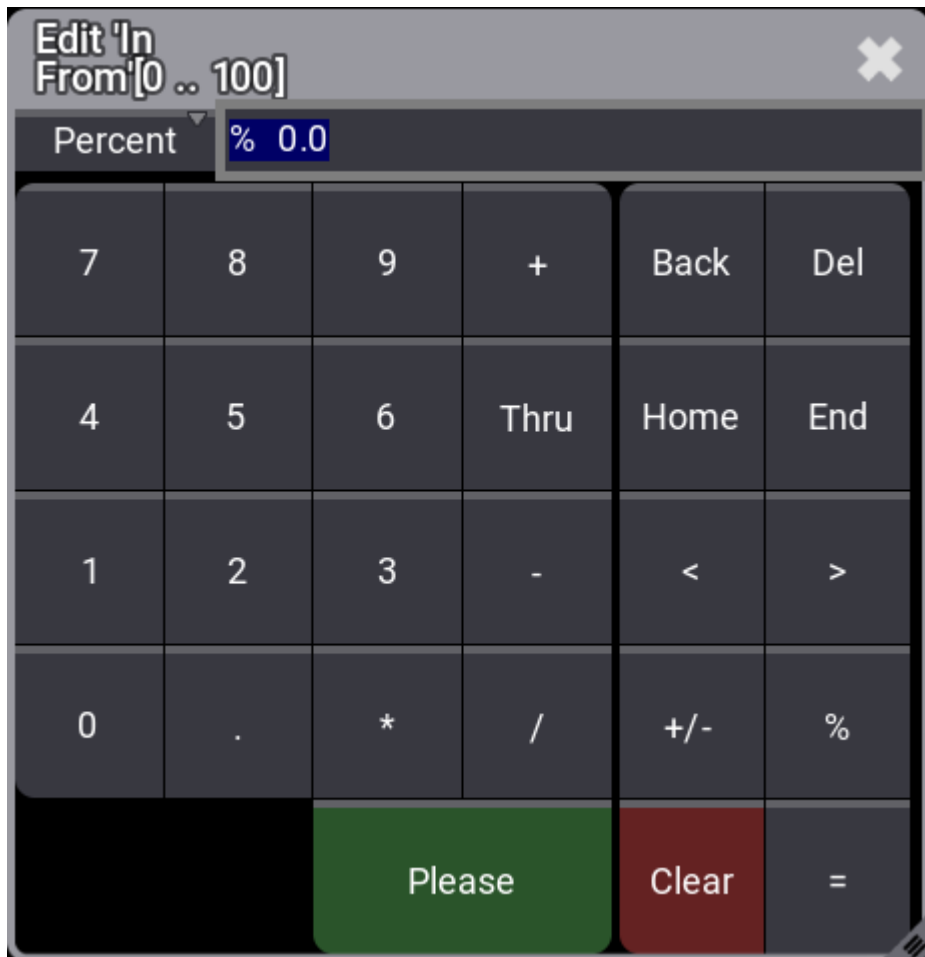
- **Trigger On / Trigger Off:** These values define the range of the key reaction. If the Trigger On, for example, is set to 75 %, the Remote reacts as soon as the incoming signal exceeds this value. If the Trigger Off, for example, is set to 25 %, the Remote stops reacting as soon as the incoming signal has fallen below this value.
- **In From / In To:** These values define the range of the input signal reaction for the defined fader.



Hint:

DC Remotes and DMX Remotes thresholds are defined in percent, while MIDI Remotes thresholds are defined between 1 and 127 MIDI velocity.

7. To set the In From value to a new value, right-click or tap and hold **In From** . The calculator opens:



Calculator In From

8. To set the In To value to a new value, right-click or tab and hold **In To** . The In To calculator opens.



- **Out From / Out To:** These values recalculate the input signal range to match up the output signal range. The range of a fader is defined by the output signal range. If a fader, for example, should not be moved to 100 %, restrict the Out To value to 90 %.
- **In / Out:** These columns at the end of the table display the signal value of the incoming signal (In) and the resulting value for the selected fader function (Out).



Hint:

For detailed information about the values, read the **Remote keyword** topic.

Example 1

- In From/To: 20-80%
- Out From/To: 0-100%
 - When the incoming signal is 10%, the fader is at 0 %.
 - When the incoming signal is 20%, the fader is at 0 %.
 - When the incoming signal is 30%, the fader is at 17 %.
 - When the incoming signal is 50%, the fader is at 50 %.
 - When the incoming signal is 80%, the fader is at 100 %.
 - When the incoming signal is 90%, the fader is at 100 %.

Example 2

- In From/To: 20-80 %.
- Out From/To: 10-90 %.
 - When the incoming signal is 0%, the fader is at 10 %.
 - When the incoming signal is 20%, the fader is at 10 %.
 - When the incoming signal is 30%, the fader is at 23 %.
 - When the incoming signal is 50%, the fader is at 50 %.
 - When the incoming signal is 80%, the fader is at 90 %.
 - When the incoming signal is 100%, the fader is at 90 %.



44.1. DC Remotes

The DC Remotes tab is used to configure the DC Remote Control input on the rear panel.

To learn more about the hardware part of the input, read the [Connect DC Remote In topic](#).

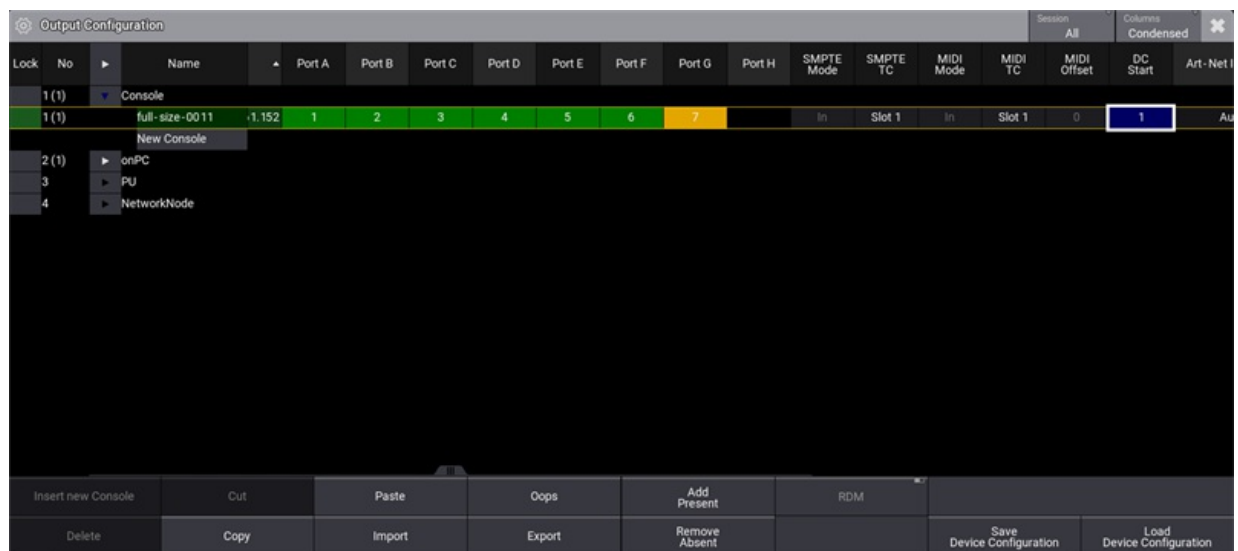
To set the DC start signal:

1. Press **Menu** and tap **Connector Configuration**. The **Connector Configuration** menu opens:

-Or-

1. Use the command line to open the menu:

```
MA User name[Fixture]> Menu "ConnectorConfig"
```



Output Configuration menu

2. Right-click or tap and hold **DC Start**, the DC Start editor opens. Set the DC Start value.



DC Start pop-up

To adjust the settings of the DC Remotes:

1. Open the In & Out window, press **Menu**, tap **In & Out** and then tap **DC Remotes**. The DC settings window opens.
- The Signal value in the In & Out window corresponds with the DC Start value in Connector Configuration.

To learn more about the general DC Remotes settings, read the [Remote In and Out topic](#).

44.2. MIDI Remotes


The **MIDI Remotes** tab is used to define actions for incoming MIDI notes or MIDI Control Changes (CC).

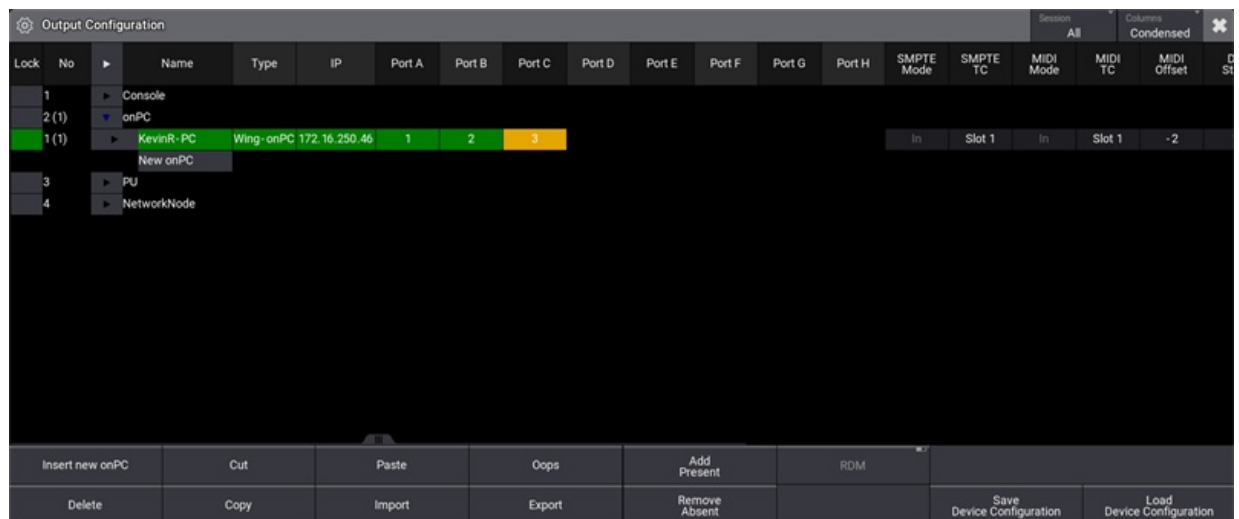
To learn more about the hardware part of the input, read the **Connect MIDI topic**.

Output Configuration Window

Adjusting the MIDI Offset will shift the MIDI Index in general.

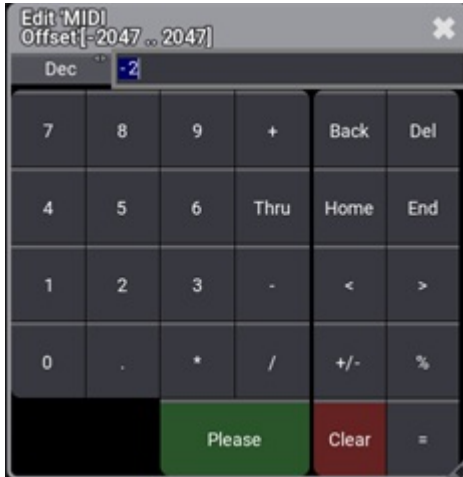
The MIDI Offset can be adjusted in the Output Configuration window as followed:

1. To set the MIDI Offset, open the **Output configuration** menu.
2. Open the product in the Output Configuration category tree using the arrows ().



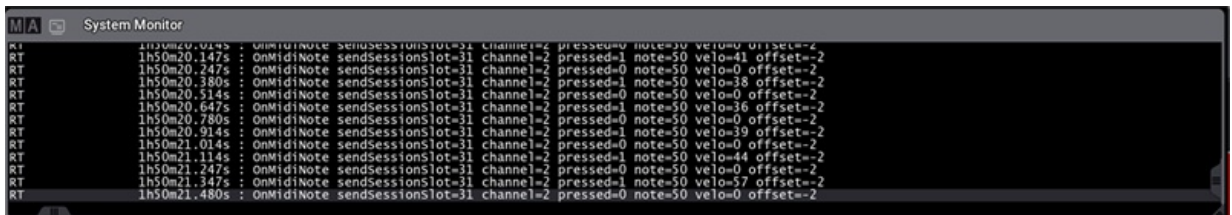
Output Configuration menu

3. To set the MIDI Offset to a new value, for example -2, right-click or tap and hold **MIDI Offset** in the product row. A pop-up opens:




MIDI Offset pop-up

4. The MIDI Offset is adjusted. For example, MIDI Index 50 is changed to 48, after MIDI Offset is set to -2 as shown below:



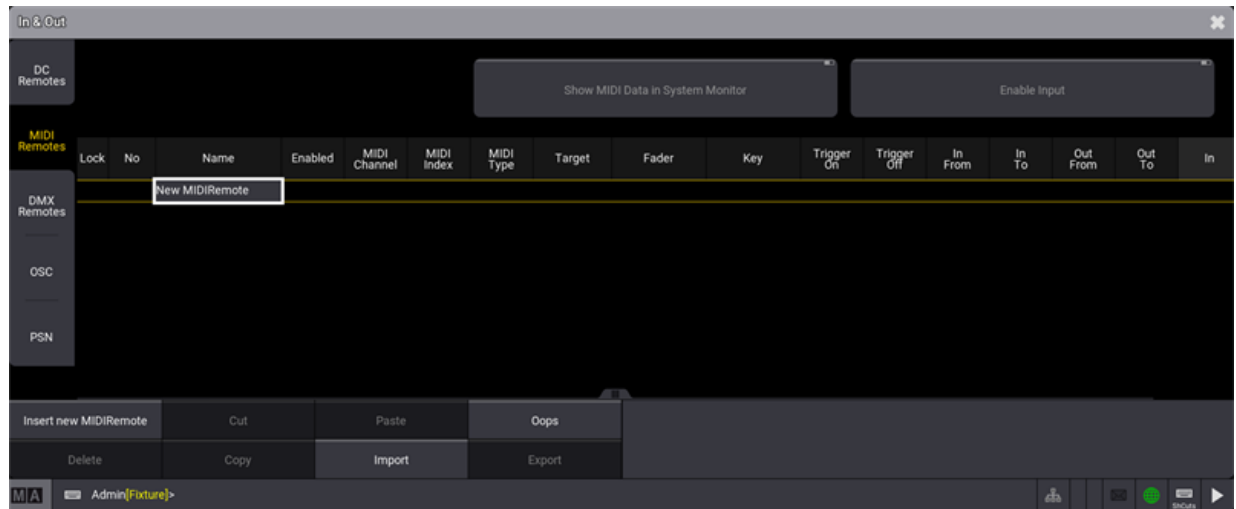
MIDI Offset changes the MIDI Index (note) shown in the System Monitor.

**Hint:**
For more information about MIDI in general, see [MIDI](#) topic.

In & Out Window

To learn more about the general Remotes settings, see [Remote In and Out](#).

1. To adjust the settings of the MIDI Remotes, switch to the In & Out window.
2. To open the In & Out window, press **Menu** and then tap **In & Out**.
3. To open MIDI Remotes, tap **MIDI Remotes**.



MIDI Remotes menu

4. To set the MIDI Channel value to a new value, right-click or tap and hold **MIDI Channel**. The Edit MIDI Channel pop-up opens:



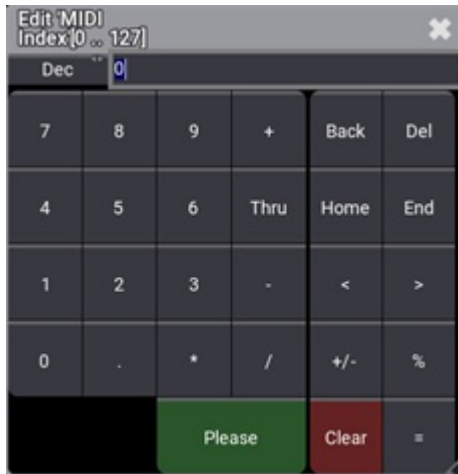
MIDI Channel Editor



Hint:

The MIDI channel value features 16 channels that can be controlled individually. The MIDI receiver must use the same MIDI channel as the MIDI sender to understand each other.

5. To set the MIDI Index value to a new value, right-click or tap and hold **MIDI Index**. The Edit MIDI Index pop-up opens:



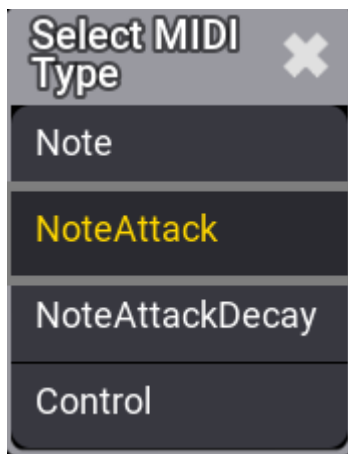
MIDI Index Editor



Hint:

The MIDI index value is the MIDI note or the MIDI control change number, depending on the set MIDI type. The MIDI receiver must use the same MIDI index value as the MIDI sender to understand each other.

6. To set the MIDI Type value to a new value, right-click or tap and hold **MIDI Type**. A pop-up opens:



Select MIDI Type

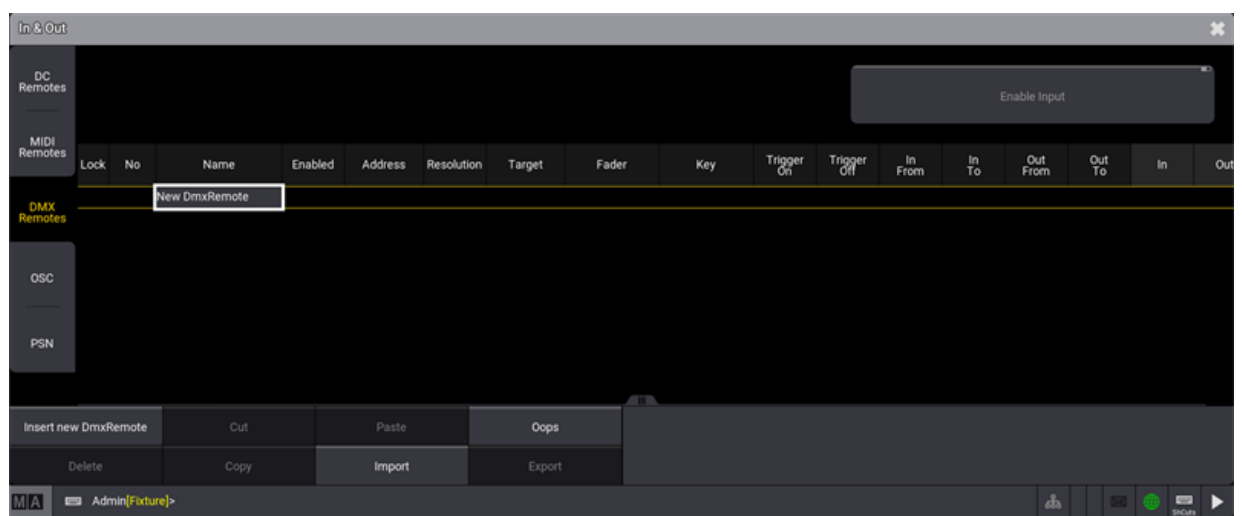
- **Note:** Only MIDI note is analyzed.
- **NoteAttack:** In addition to the MIDI note, the attack of the note will be analyzed. The higher the attack value is, the more a fader moves.
- **NoteAttackDecay:** Together with the MIDI note, the Attack and Decay of the MIDI Note will be used.
- **Control:** MIDI Control Change will be used.

44.3. DMX Remotes

The **DMX Remotes** tab uses DMX channels as remote triggers. The DMX source can also be the console itself.

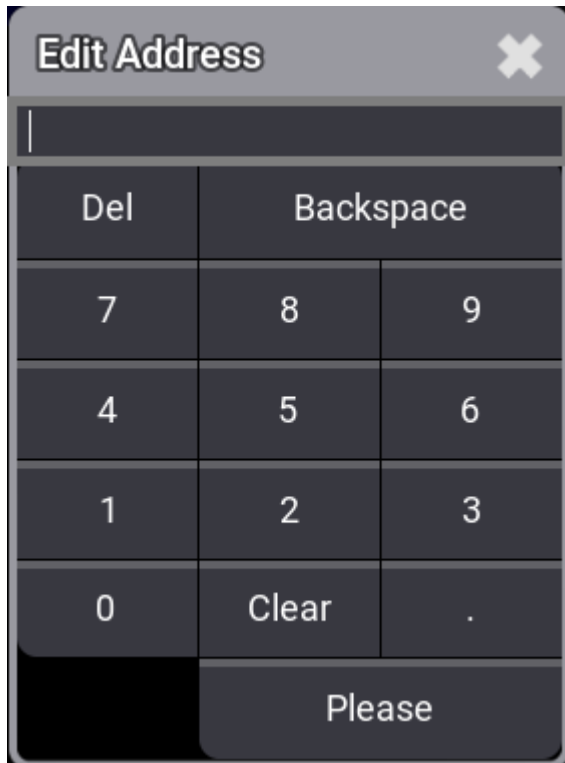
- To learn more about the hardware part of the input, read the [Connect DMX topic](#).
- To learn more about the general Remotes settings, see [Remote In and Out](#).

1. To adjust the settings of the DMX Remotes, tap **DMX Remotes**. The DMX Remotes menu opens:



DMX Remotes menu

2. To set the DMX address the DMX Remote input should listen to, right-click or tap and hold **Address**.



DMX address pop-up

3. To adjust the DMX resolution, open the Resolution cell of the DMX Remote entry you want to edit.



DMX resolution pop-up

- The DMX Remote can be controlled by 16 bit or 24 bit DMX channels. This allows for more precise control.
- On the controlling DMX device, the 16 bit and 24 bit channels must be patched directly after the 8 bit channel.



44.4. OSC (Open Sound Control)

OSC is a client and server system that defines a message address pattern used to address elements in the receiving server.

The grandMA3 software supports OSC 1.1. For more information about how to format OSC packets, see https://ccrma.stanford.edu/groups/osc/spec-1_0.html.

Open Sound Control, or OSC, is a networking protocol used to allow devices of various types to control other devices of other types. OSC messages are human-readable (unlike, for example, MIDI Show Control, or MSC), and follow this general pattern:

`"(/prefix)/[OSC Address],[OSC Type],[Value]"`

prefix - this is optional, depending on your system setup. It can be used in a more complex OSC network to differentiate messages meant for one set of devices (e.g. lighting consoles) and not others (e.g. sound consoles).

OSC Address - this is the target you are controlling on the receiving device(s), for example `/Fader201` would be the address to move the fader for executor 201 in grandMA3. Sometimes the address will be more complex, for example `/Page1/Fader201` would be the address to move the fader for executor 201 on page 1 in grandMA3.

OSC Type - this is the type of value you're sending, for example:

i = integer

f = float

s = string

t = true

f = false

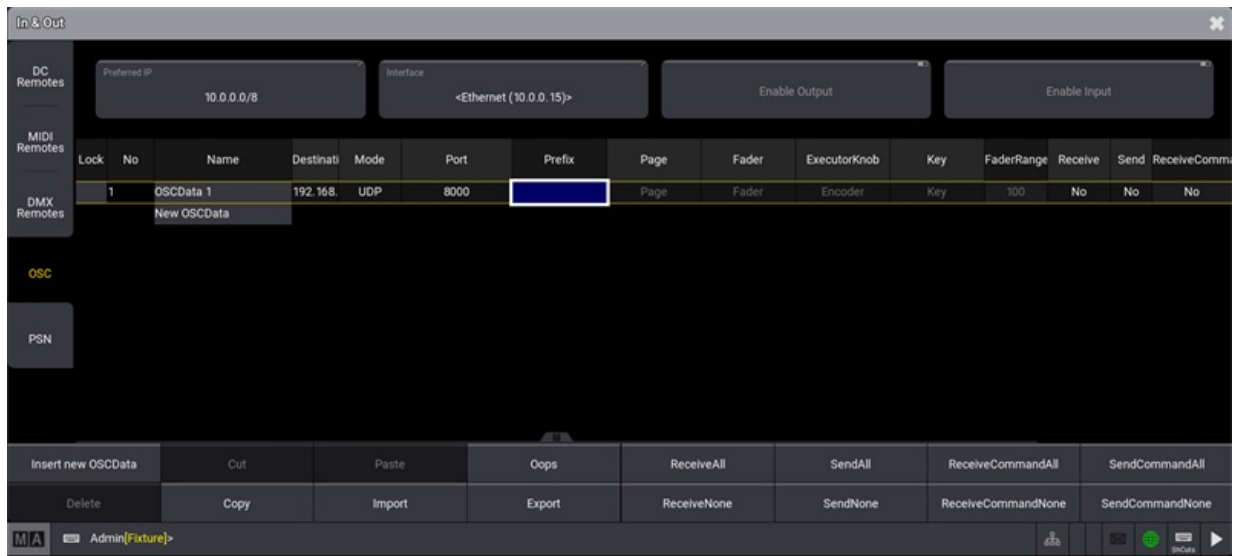
Value - this is the value you are actually sending for the target.

An example OSC command to set the fader for executor 201 to 100 might be:

- `"/Page1/Fader201,i,100"`
- or with a prefix to specify only, e.g. grandMA3 devices: `"/gma3/Page1/Fader201,i,100"`

For more information about the OSC address, OSC type, and OSC structure, see [SendOSC keyword](#).

- To adjust the settings of OSC, tap **OSC** in the In & Out window.
The OSC settings window opens.



OSC settings window

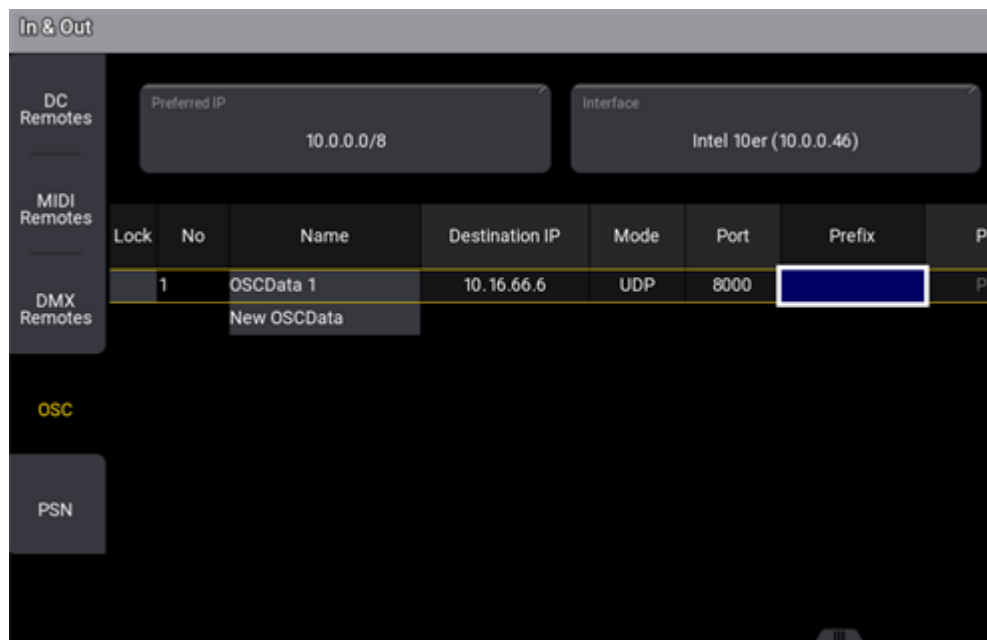


Hint:

Please note that in the example above, no prefix is defined.

The most important OSC setting is the correct network configuration.

Make sure that the IP address, the network protocol (UDP, TCP), and the port are set correctly.



OSC network configuration



Hint:

Please note that the port configuration is used for sending and receiving OSC data.

When receiving OSC messages, **Input** will highlight its title bar. When sending OSC messages, the title bar of **Output** will be highlighted.

Each configuration line for OSC can be used for input and/or output.

These properties can be configured:

- Name: Sets the name for this configuration.
- Destination IP: Sets the IP address for sending OSC data. A specific IP address or a broadcast IP can be set.
- Mode: OSC packets can be sent via UDP or TCP.
- Port: Specifies the network port of the incoming and/or outgoing OSC packets.
- Prefix: A prefix can be set by the user if he needs to. A prefix can be used for example as a criterion for limiting the range of possible receivers, e.g. /lighting would only take packets with /lighting into account, and discard OSC packets with the /sound-prefix.
- Page: Specifies which OSC Address of incoming OSC messages is routed to pages.
- Fader: Specifies which OSC Address of incoming OSC messages is routed to faders.
- ExecutorKnob: Specifies which OSC Address of incoming OSC messages is routed to the mini encoders.
- Key: Specifies which OSC Address of incoming OSC messages is routed to keys.

- **FaderRange:** Specifies which OSC value range is used for the fader, e.g. FaderRange 255 sets OSC 0-255 to 100%.
- **Receive:** Specifies if OSC data (but no commands) shall be received.
- **Send:** Specifies if this OSC configuration sends OSC data (but no commands).
- **Receive Command:** Specifies if commands for the command line will be received via OSC. This setting is independent of the general receive setting.
- **Send Command:** Specifies if commands of the command line will be sent via OSC. This setting is independent of the general send setting.
- **EchoInput:** Specifies if the input data shall be displayed in the system monitor.
- **EchoOutput:** Specifies if the output data shall be displayed in the system monitor.

With the buttons

`ReceiveAll`, `ReceiveNone`, `SendAll`, `SendNone`, `Receive CommandAll`, `Receive CommandNone`, `Send CommandAll` and `Send CommandNone` all OSC configuration lines can be modified together for the properties Receive, Send, Receive Command and Send Command.



Hint:

The addresses defined for Page, Prefix, Fader, ExecutorKnob, and Key are case-sensitive.

In addition to the example OSC strings mentioned above, the entirety of the grandMA3 command line can be accessed via OSC, using the `/cmd` OSC Address and the string `'s'` OSC Type. *Note - this requires "Receive Command" to be set to Yes.* For example:

- `/cmd,s,FaderMaster Page 1.201 At 100"`
 - Bring fader 201 on page 1 to 100% (same as the examples above but using gMA3 command line syntax instead)
- `/cmd,s,Fixture 1 At 75"`
 - Use gMA3 command line syntax to execute the command "Fixture 1 At 75" in the command line
- `/cmd,s,Go+ Exec 402"`
 - Trigger whatever is assigned to exec 402
- `/cmd,s,Patch Fixture 1 3.42"`
 - Patch fixture 1 to address 42 in universe 3



Hint:

For more information about regularly used programs to trigger grandMA3, see [Examples for Using OSC](#).



44.4.1. Examples

The use of OSC in different situations is described in the following topics.

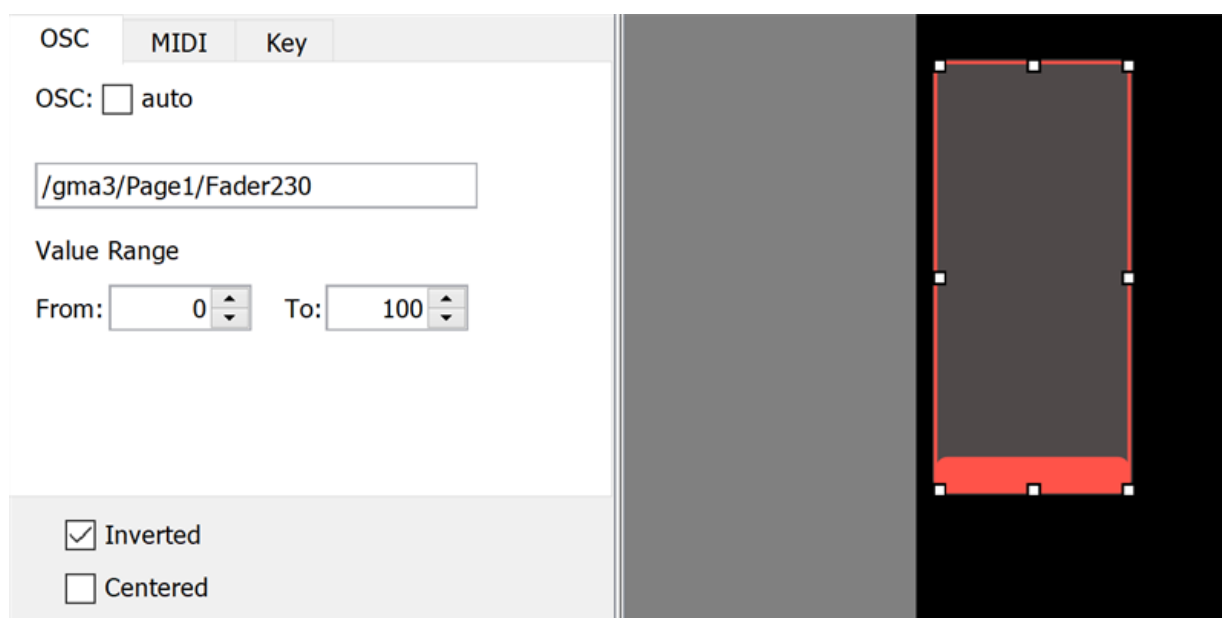
44.4.2. TouchOSC

TouchOSC is a modular OSC and MIDI control surface for Windows, macOS, and Android by hexler.net.

It supports sending and receiving Open Sound Control and MIDI messages over Wi-Fi and CoreMIDI inter-app communication and compatible hardware.

Fader

This example will control the fader of Executor 230 of Page 1:

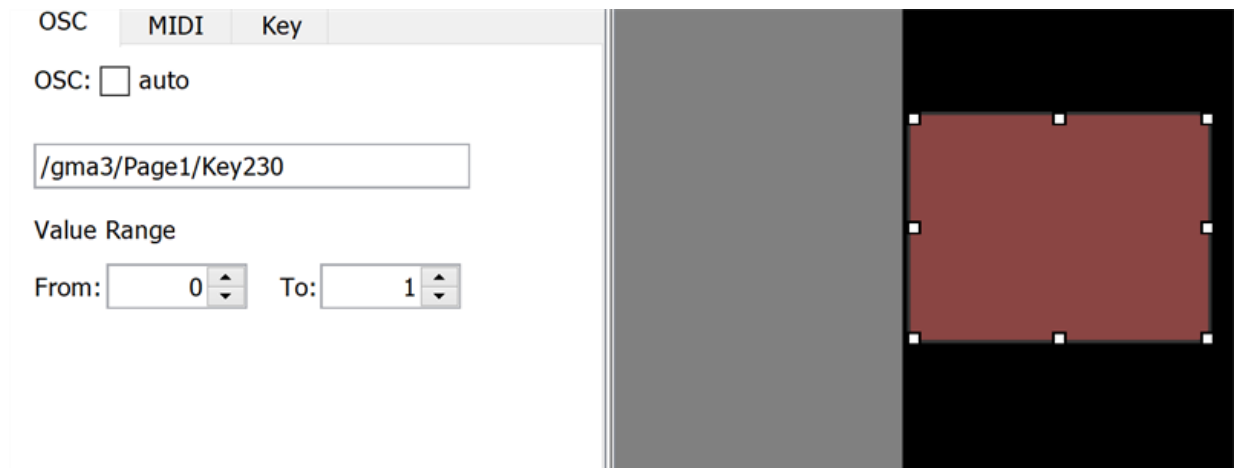


Notes:

- Assumes the OSCData line on the console has a prefix of "gma3" configured. If the prefix is empty, this would just be /Page1/Fader230.
- Assumes the "Page" and "Fader" cells in the OSCData line on the console are set to "Page" and "Fader" respectively (this is the default).

Executor Button

This example will press the button for Executor 230 of Page 1:



Notes:

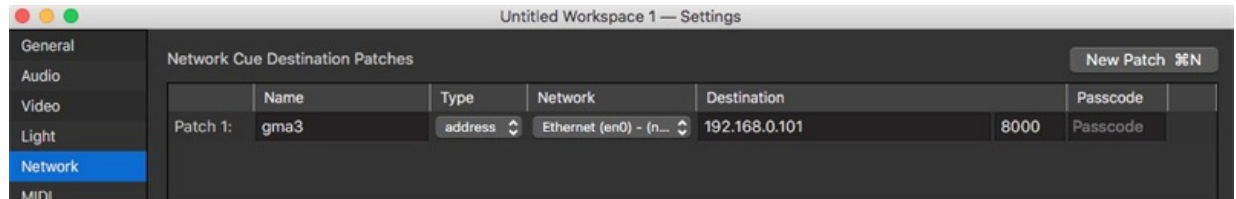
- Assumes the OSCData line on the console has a prefix of "gma3" configured. If the prefix is empty, this would just be /Page1/Key230.
- Assumes the "Page" and "Key" cells in the OSCData line on the console are set to "Page" and "Key" respectively (this is the default).
- The {Send on Press} and {Send on Release} settings (not pictured above) should both be enabled/checked.



44.4.3. QLab

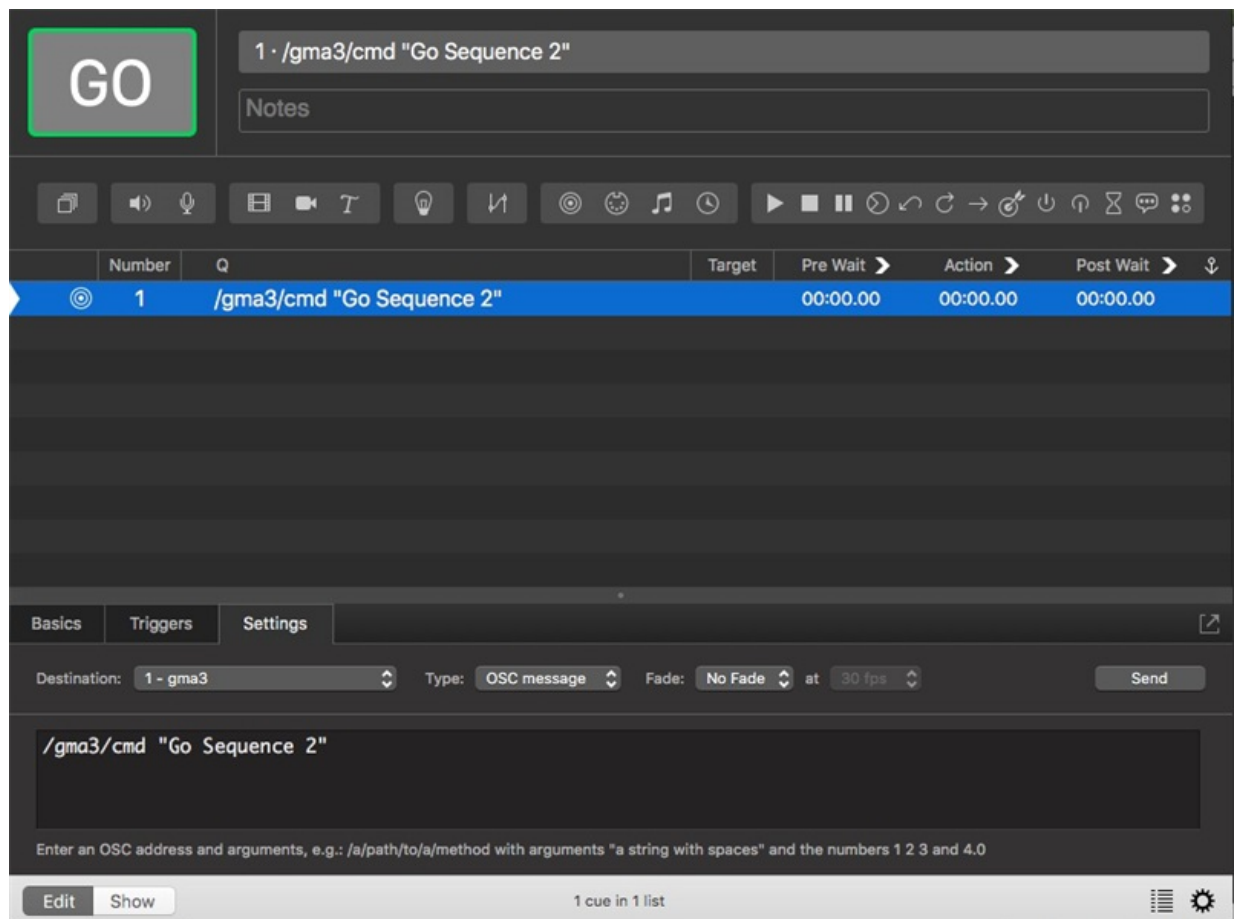
QLab is sound, video, and lighting control for macOS by qlab.app.

QLab is fairly simple to use with OSC. The QLab network setting are shown in the example below:



- Name = something to identify this particular configuration
- Network = the network interface on your computer connected to the grandMA3 system
- Destination
 - IP address of the grandMA3 console
 - Port set in the OSCData line configuration in the console (8000 is default)

Cue setup in QLab is shown below:

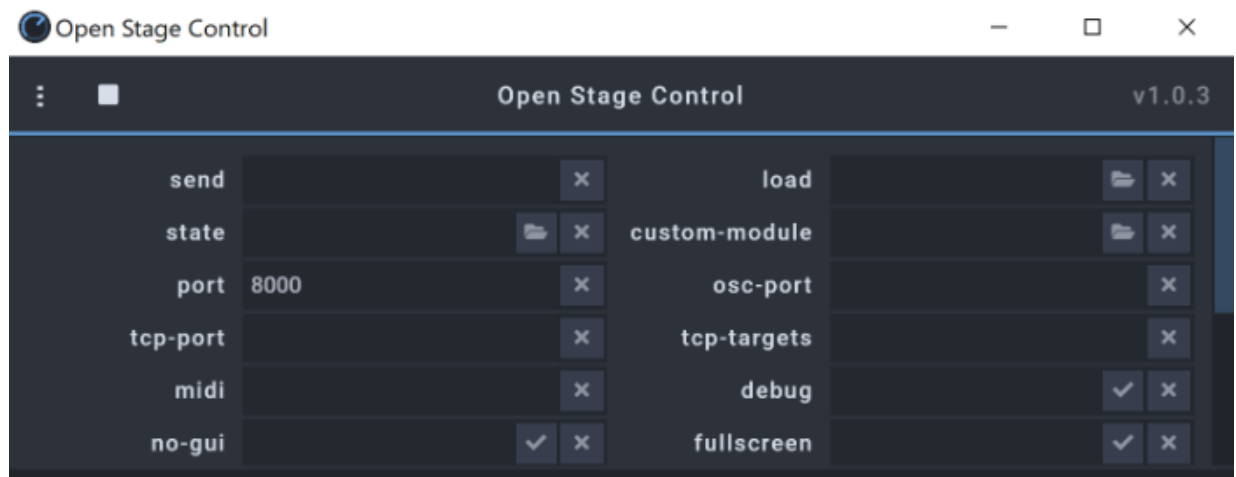




- Destination = configuration as set in the "Network Cue Destination Patches" above
- Type = OSC message
- Enter the desired OSC message
 - OSCAddress - in this example we are sending a command syntax string directly, so the address is:
/gma3/cmd
 - This assumes the OSCData line on the console has a prefix of "gma3" configured! If the prefix is empty, this would just be /cmd
 - *Requires "Receive Command" to be enabled for that OSCData line on the console!*
 - Argument - enter the command syntax in quotes, for example, "Go Sequence 2" (advancing to the next cue in Sequence 2)

44.4.4. Open Stage Control

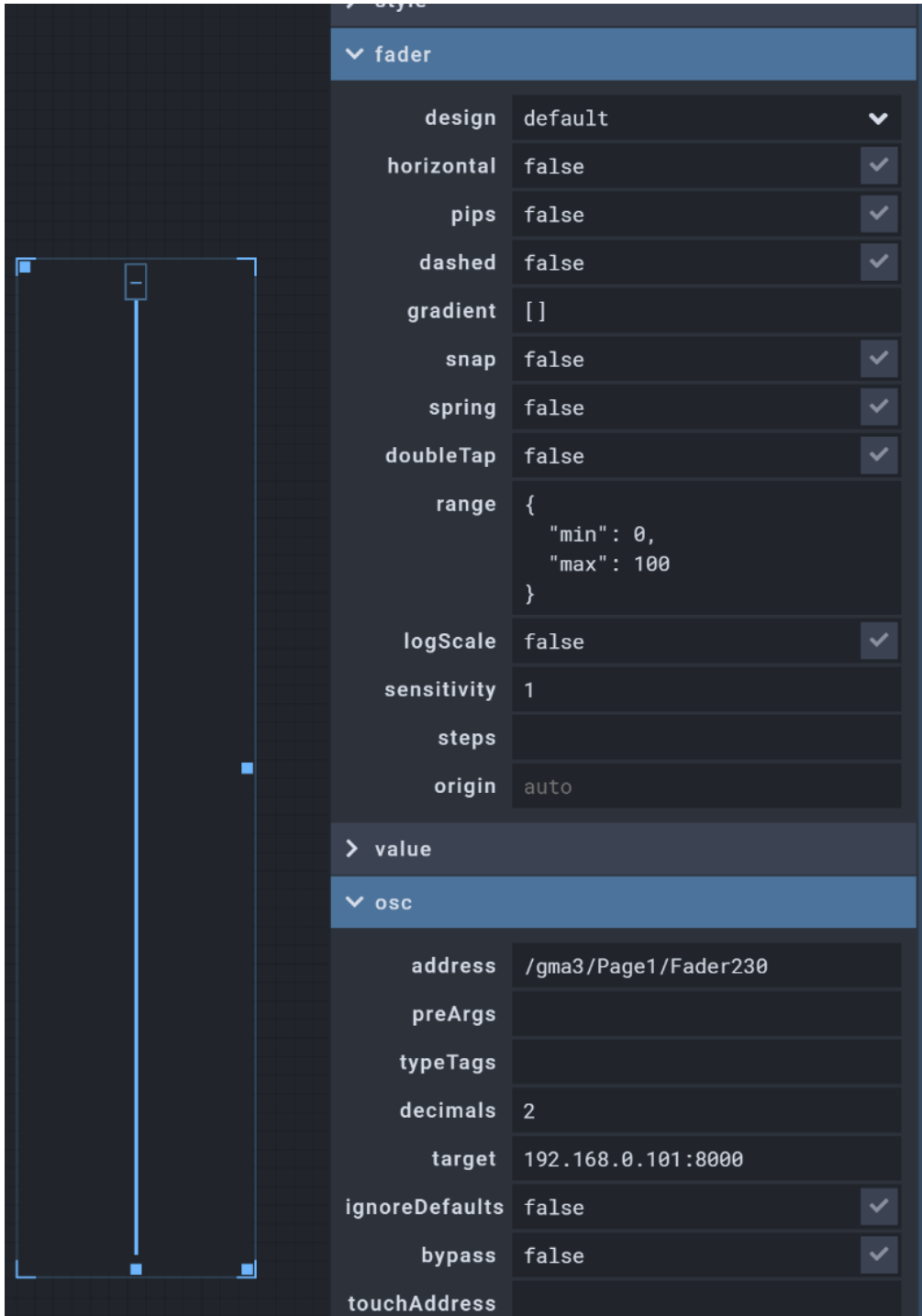
Open Stage Control is a free program used to build a simple OSC interface. When first opening the program, the network configuration settings pop up. Besides telling it which network interface to use on the computer, the only thing to fill in here is the port:



This needs to match the port that has been set in the corresponding OSCData line in the console. Port 8000 is the default. Afterwards the Open Stage Control session can be started.

Fader

This example will control the fader for Executor 230 on Page 1:



The screenshot displays the configuration panel for a fader control. On the left, a vertical fader control is visible with a blue slider bar and a minus sign at the top. The configuration panel on the right is organized into sections: 'fader', 'value', and 'osc'.

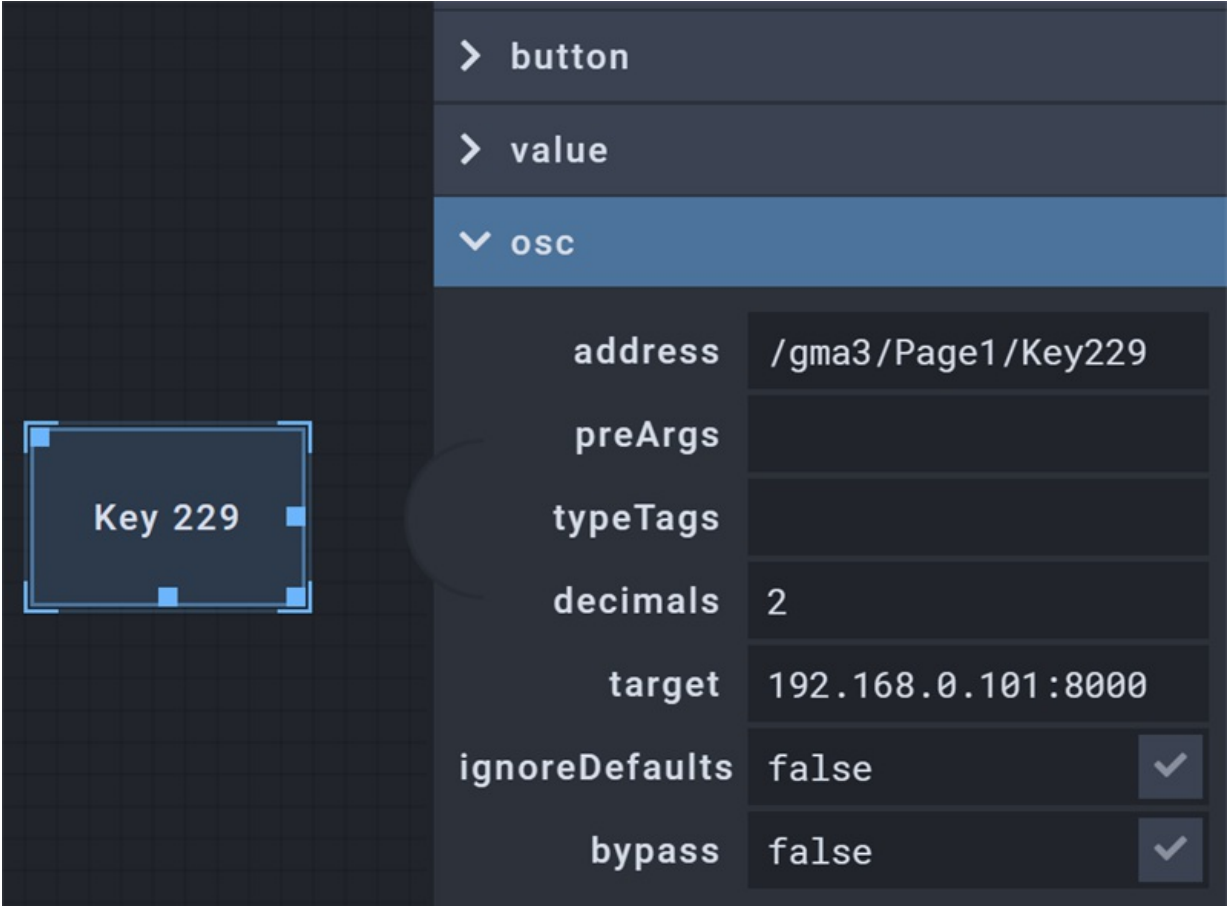
Property	Value
design	default
horizontal	false
pips	false
dashed	false
gradient	[]
snap	false
spring	false
doubleTap	false
range	{ "min": 0, "max": 100 }
logScale	false
sensitivity	1
steps	
origin	auto
> value	
▼ osc	
address	/gma3/Page1/Fader230
preArgs	
typeTags	
decimals	2
target	192.168.0.101:8000
ignoreDefaults	false
bypass	false
touchAddress	

Notes:

- Assumes the OSCData line on the console has a prefix of "gma3" configured. If the prefix is empty, this would just be /Page1/Fader230.
- Assumes the "Page" and "Fader" cells on the OSCData line in the console are set to "Page" and "Fader" respectively (this is the default).
- All of the settings in the picture above are at their defaults except for:
 - fader settings: range: change the 'max' to 100 instead of 1
 - osc settings: address

Executor button

This example will press the button for Executor 229 on Page 1:



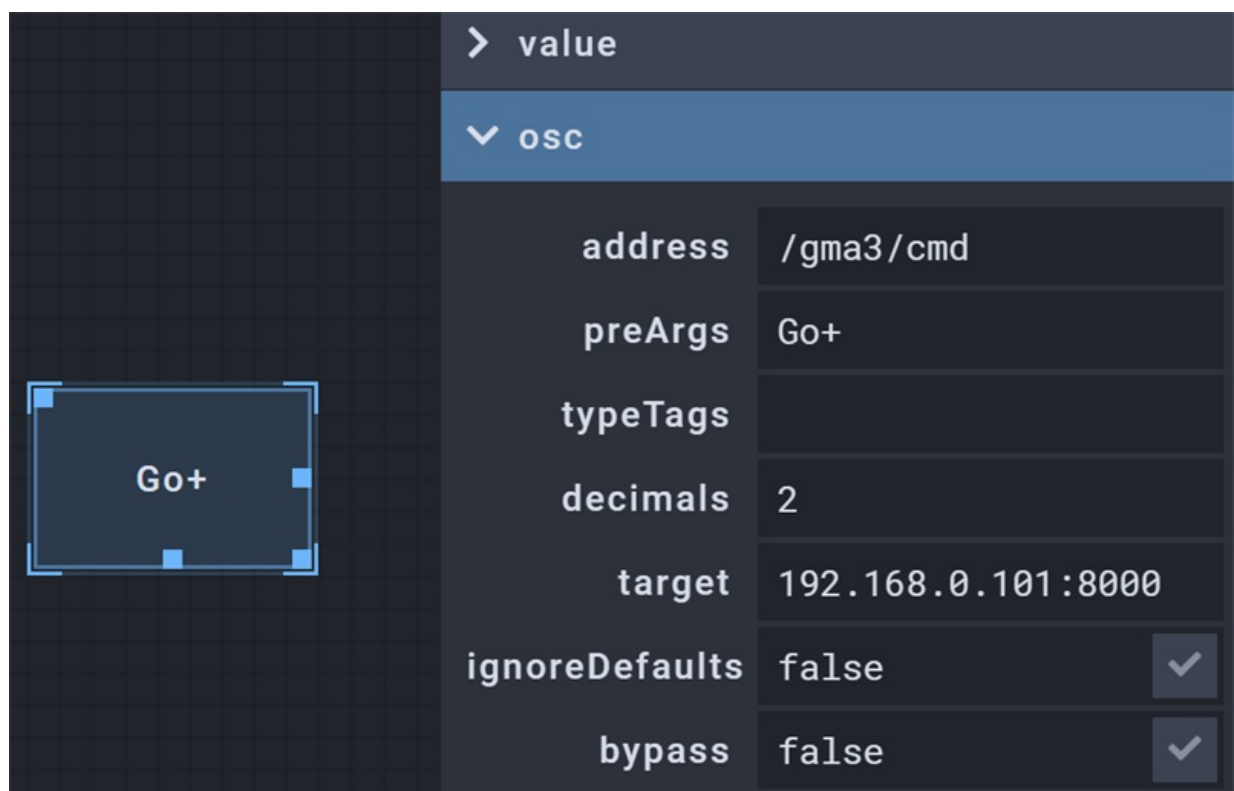
>	button
>	value
∨	osc
address	/gma3/Page1/Key229
preArgs	
typeTags	
decimals	2
target	192.168.0.101:8000
ignoreDefaults	false <input checked="" type="checkbox"/>
bypass	false <input checked="" type="checkbox"/>

Notes:

- Assumes the OSCData line on the console has a prefix of "gma3" configured. If the prefix is empty, this would just be /Page1/Key229.
- Assumes the "Page" and "Key" cells in the OSCData line on the console are set to "Page" and "Key" respectively (this is the default).
- All of the settings in the picture above are at their defaults except for the address in the osc settings.
- Open Stage Control buttons default to functioning as 'toggle' - you may wish to change this to 'tap'.

Command Line Syntax

Here we have a button that will execute command line syntax on the console, in this case triggering the Selected Sequence:



Notes:

- Assumes the OSCData line on the console has a prefix of "gma3" configured. If the prefix is empty, this would just be /cmd.
- Requires **Receive Command** to be enabled for that OSCData line on the console.



- All of the settings in the picture above are at their defaults except for:
 - button mode set to "momentary"
 - address
 - preArgs - this is where you enter the syntax string you wish to execute



44.4.5. zactrack

zactrack is an automated full-tracking follow system designed for open-air events, theater stages, and studios. Any number of performers can be placed in a 2D or 3D space. zactrack precisely aligns output devices and special effects equipment and sends XYZ coordinates to grandMA3.

Example 1:

Trigger cues on a grandMA3 console with a performance area defined by a 3x3 2D grid. The position will trigger cues in that specific zone number, when walking with a tracker.

1. Enter the console IP and Port.
2. Enter the exact string address and argument the console expects. In the example below on the console, it is executor 202 on page1.

Fixture Type Settings

Name: OSC Cue

Short Name: OSC Cue

Destination: Target IPs (1) +

192.168.0.69:8000 -

Address ? Use placeholder address string ?

/cmd

Preview: /cmd

Argument(s) Use placeholder argument string ?

Goto Page 1.202 Cue %GRID_INDEX%

Preview: Goto Page 1.202 Cue GRID_INDEX

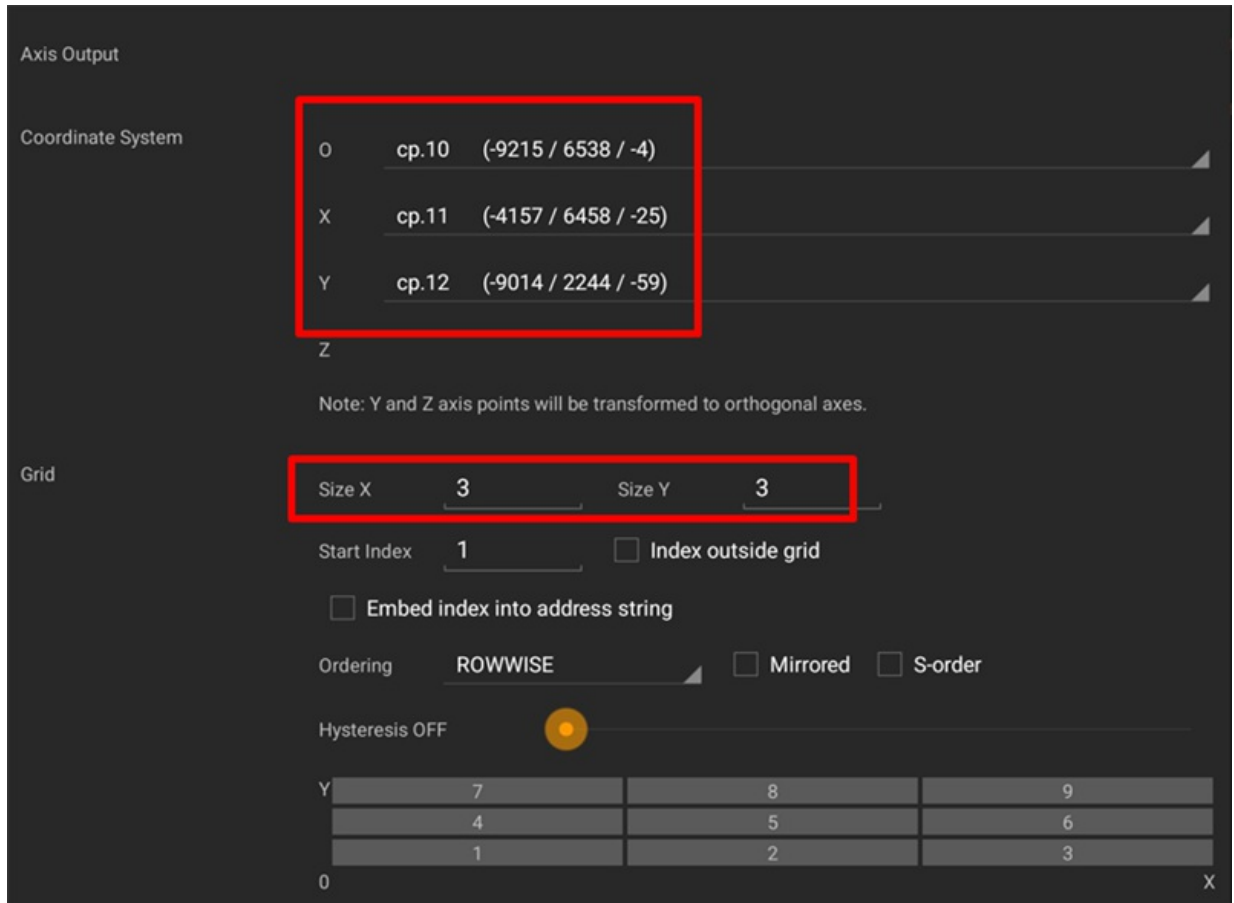
Mode: Grid 2D (int)

Send OSC only on change

Limit OSC packets per second



3. Define the Coordinate System using calibration points (via Pucks / Trackers / Disto).
4. Set the grid size and order.



The 2D grid is created.



Example 2:

Trigger a fader on a grandMA3 console. When walking in defined axis with a Tracker the position will trigger for example, linear intensity, for that specific fader.

1. Enter the console IP and Port.
2. Enter the exact string address the console expects. In the example, it is the fader of executor 201 on page 1 of the grandMA3 software.
3. Set `Range (int)` as Mode.



4. Define the Axis Output, X only.
5. Define the Coordinate System.
6. Set the X Range for the fader. These values must match the corresponding setting in the grandMA3 software. The default value is 0 to 100.

The screenshot shows the configuration menu for the grandMA3 software. The 'Axis Output' is set to 'X'. The 'Coordinate System' is set to 'X' with 'cp.11' and coordinates '(-4157 / 6458 / -25)'. The 'Ranges' section shows 'X From 0' and 'To 100'. Below this is a table for 'XY-Axes' and 'Z-Axis' settings.

Reset	XY-Axes		Z-Axis	
	prediction (ms)	smoothing (%)	prediction (ms)	smoothing (%)
slow	0	30	0	80
medium	0	20	0	80
fast	0	10	0	80

Buttons: Cancel, OK

44.4.6. Protocol Viewer

The Protokol app by **Hexler** is a test / monitoring tool to check OSC. It works on any device including smartphones.

```
0.20.1.1]:43904)
ADDRESS(/zactrack5) INT32(5302)
INT32(1956) INT32(0)
RECEIVE | ENDPOINT([::ffff:1
0.20.1.1]:43904)
ADDRESS(/zactrack5) INT32(5307)
INT32(1932) INT32(0)
RECEIVE | ENDPOINT([::ffff:1
0.20.1.1]:43904)
ADDRESS(/zactrack5) INT32(5313)
INT32(1909) INT32(0)
```

Absolute Millimeters (Integer) -
This example uses the X and Y Axes
only

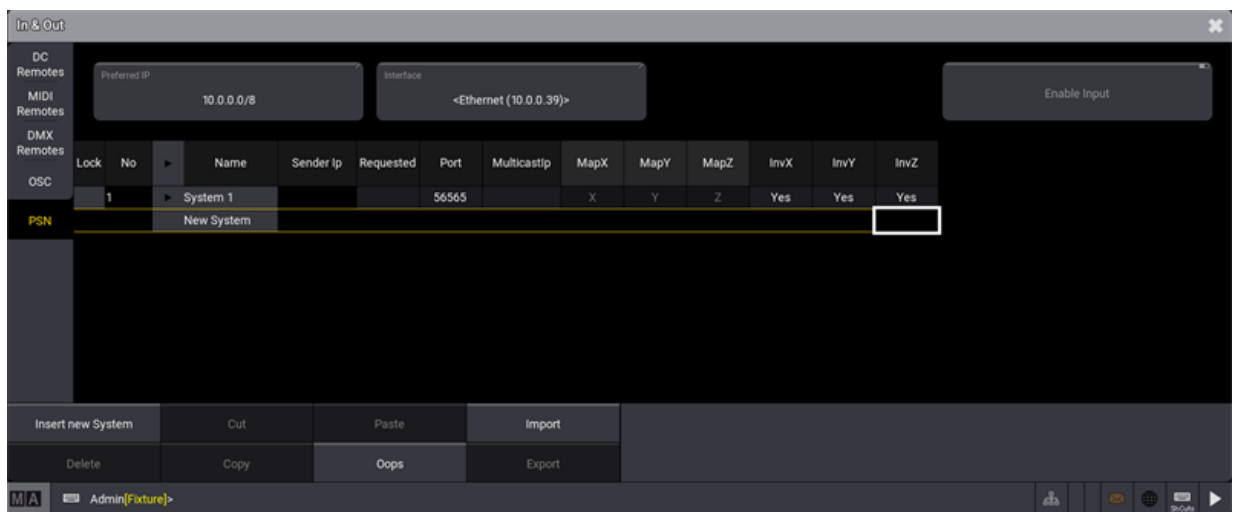
```
0.20.1.1]:56928)
ADDRESS(/zactrack5)
FLOAT(0.9940467)
FLOAT(1.7281017) FLOAT(0)
RECEIVE | ENDPOINT([::ffff:1
0.20.1.1]:56928)
ADDRESS(/zactrack5)
FLOAT(1.0040512)
FLOAT(1.7269224) FLOAT(0)
RECEIVE | ENDPOINT([::ffff:1
0.20.1.1]:56928)
ADDRESS(/zactrack5)
FLOAT(0.9990025)
FLOAT(1.7262422) FLOAT(0)
```

Absolute Metres (Float) -
This is example uses
the XYZ Axes with a high resolution
float value

44.5. PSN (PosiStageNet)

grandMA3 stations can receive PosiStageNet (PSN) data.

- To adjust the settings of PSN, tap **PSN** in the In & Out window.
The PSN settings window opens.



PosiStageNet (PSN) window

When a PSN source is available in the network, adding a new PSN System by tapping **Insert new System** is required.

Depending on the type of transmitting the data, the user may have to make some settings first:

- In case of sending the PSN data to a multicast address, the user has to manually add the multicast IP of the PSN system.
- In most cases, this may be IP 236.10.10.10.
- If the sender is transmitting the data via unicast to the selected interface in the PSN menu, the PSN system automatically detects the data.

For each PSN system that is received, the user can map each axis to a different axis and/or invert the incoming data per axis:

- Tap and hold the cell of an axis or of an invert option.
- In the case of mapping, a drop-down opens to choose the target axis. Invert toggles the cell between No (=empty cell) and Yes (= data will be inverted).

As soon as an axis is mapped or inverted, the individual trackers display their values according to the settings made by the user.



44.6. MIDI

MIDI stands for **M**usical **I**nstrument **D**igital **I**nterface. MIDI is used as a world-wide standard protocol that allows a communication between all kinds of digital devices. For example:

- a MIDI keyboard and a grandMA3 command wing.
- a MIDI pad controller and a grandMA3 console.
- a grandMA3 console and an audio mixing console.

An external MIDI device can be used as a wing extension for an existing grandMA3 console or for controlling an external sound generator like a buzzer, for example.

To communicate between devices, MIDI messages are used. For example, when connecting a MIDI keyboard to a grandMA3 console, a numeric description about channel, note and velocity are sent to the connected console via a MIDI message.



Hint:

To get more information about MIDI, see [Connect MIDI](#) topics.



Restriction:

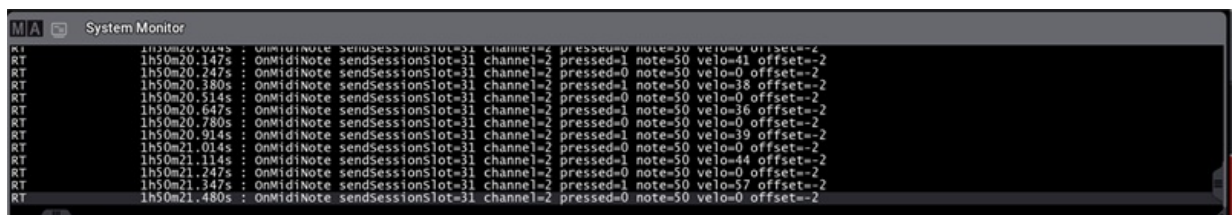
MIDI does not transmit an audio signal.

Configure an External MIDI Device

It is possible to connect an external MIDI device to receive messages to certain grandMA3 products. Therefore, a standard MIDI cable can be used.

Requirements:

- 1x **grandMA3 console** or **grandMA3 onPC** station either with a **command wing**, a **I/O node** or a third-party USB device connected
 - 1x MIDI capable device
 - 1x standard MIDI cable
1. Connect the MIDI cable to the MIDI data output connector (MIDI Out) from the external MIDI device to the MIDI data input connector (MIDI In) on the console. Make sure the external MIDI device is powered, too!
 2. Open **Menu** - **In & Out** - **MIDI Remotes**.
 3. To make the messages of the external MIDI device visible in grandMA3, enable **Show MIDI Data in System Monitor**.
 4. To allow MIDI inputs on a grandMA3 console from an external MIDI device, make sure to enable **Enable Input**.
 5. To see the incoming MIDI messages of the external MIDI device on the console, open the System Monitor window and press different hardkeys (for example the keys) on the external MIDI device. The MIDI messages are listed in the System Monitor as shown in the image below:



```
MA System Monitor
RT 1h50m20.0145 : OnMidiNote sendSessionsSlot=31 channel=2 pressed=0 note=50 velo=0 offset=-2
RT 1h50m20.1475 : OnMidiNote sendSessionsSlot=31 channel=2 pressed=1 note=50 velo=41 offset=-2
RT 1h50m20.2475 : OnMidiNote sendSessionsSlot=31 channel=2 pressed=0 note=50 velo=0 offset=-2
RT 1h50m20.3805 : OnMidiNote sendSessionsSlot=31 channel=2 pressed=1 note=50 velo=38 offset=-2
RT 1h50m20.5145 : OnMidiNote sendSessionsSlot=31 channel=2 pressed=0 note=50 velo=0 offset=-2
RT 1h50m20.6475 : OnMidiNote sendSessionsSlot=31 channel=2 pressed=1 note=50 velo=36 offset=-2
RT 1h50m20.7805 : OnMidiNote sendSessionsSlot=31 channel=2 pressed=0 note=50 velo=0 offset=-2
RT 1h50m20.9145 : OnMidiNote sendSessionsSlot=31 channel=2 pressed=1 note=50 velo=39 offset=-2
RT 1h50m21.0145 : OnMidiNote sendSessionsSlot=31 channel=2 pressed=0 note=50 velo=0 offset=-2
RT 1h50m21.1145 : OnMidiNote sendSessionsSlot=31 channel=2 pressed=1 note=50 velo=44 offset=-2
RT 1h50m21.2475 : OnMidiNote sendSessionsSlot=31 channel=2 pressed=0 note=50 velo=0 offset=-2
RT 1h50m21.3475 : OnMidiNote sendSessionsSlot=31 channel=2 pressed=1 note=50 velo=37 offset=-2
RT 1h50m21.4805 : OnMidiNote sendSessionsSlot=31 channel=2 pressed=0 note=50 velo=0 offset=-2
```

After pressing several keys and turning knobs on the external MIDI device, the MIDI messages are shown in the System Monitor.

A MIDI message can contain several parameters:

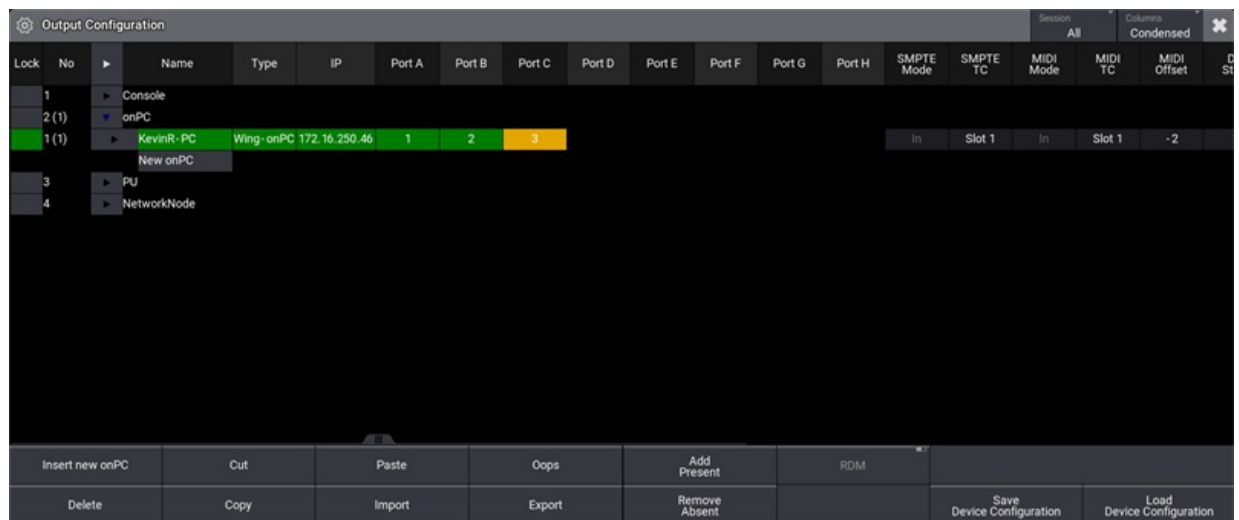
- **channel**: Displays the external devices MIDI channel.
- **pressed**: Shows if a MIDI key is pressed (1) or released (0).
- **note**: Shows the MIDI note which is oblique to a key on the external MIDI device.
- **velo**: Shows the velocity which displays the intensity of a pressed key from 0 to 127.
- **control**: Shows the control change address and its absolute values (=value).
- **offset**: See **MIDI Remotes** topic.



Receive MIDI

Requirement:

- 1x grandMA3 demo show file
- To send MIDI messages, make sure the MIDI Mode is set to **In** in the Output Configuration menu:

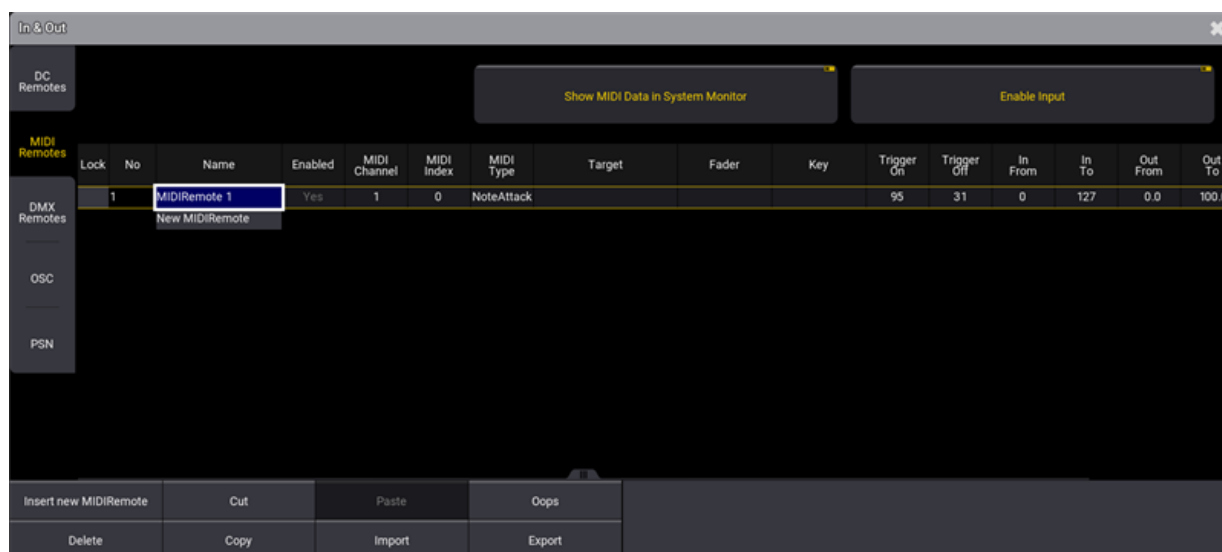


MIDI Mode is set to In in the Output Configuration menu.

After connecting an external MIDI device to the grandMA3 console, it is time to insert and customize a MIDI remote.

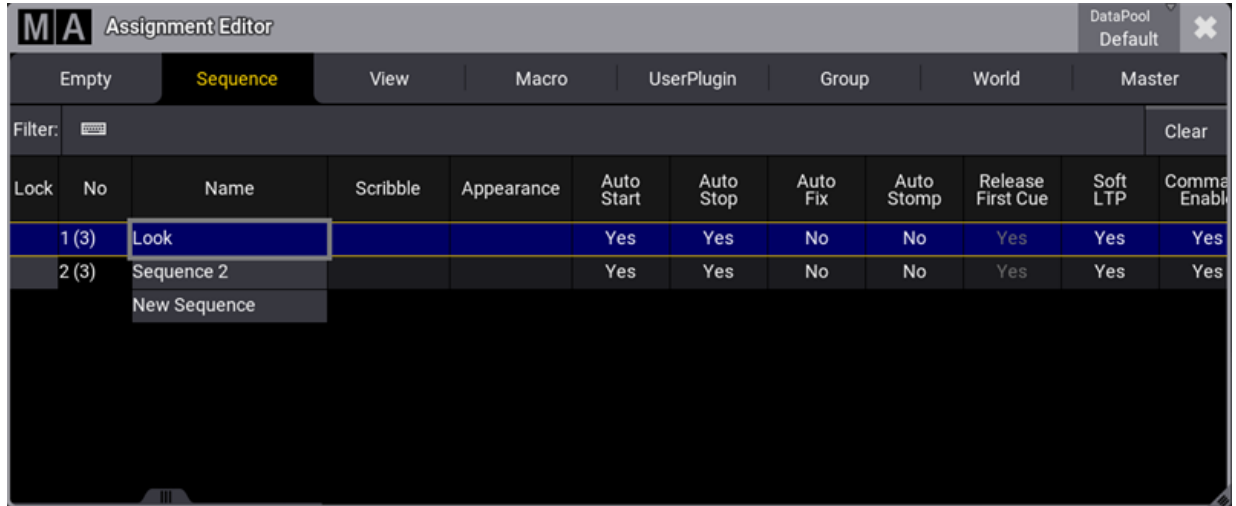
In this example, the sequence on an executor is controlled by pressing a key on a external MIDI keyboard:

1. To insert a new MIDI remote, tap **Insert new MIDIRemote**. The row MIDIRemote 1 is added automatically as shown below:



A new MIDIRemote is set in the MIDI Remotes-tab.

2. To rename, tap **MIDIRemote 1** in the Name column. Start to type "Key 1", tap **Enter** on the virtual keyboard, or press **Please**.
3. To allocate a specific hardkey on the MIDI keyboard, press a key on the keyboard and use the information **channel** and **note** from the System Monitor window to edit the MIDI Channel and MIDI Index columns. For this example MIDI Channel is set to 2 and MIDI Index is set to 48.
4. Set the MIDI Type to **NoteAttack**, to implement velocity.
5. To open the Assignment Editor, tap and edit the empty cell in the **Target** column. Tap the **Sequence** - tab and select **Look** as shown in the image:



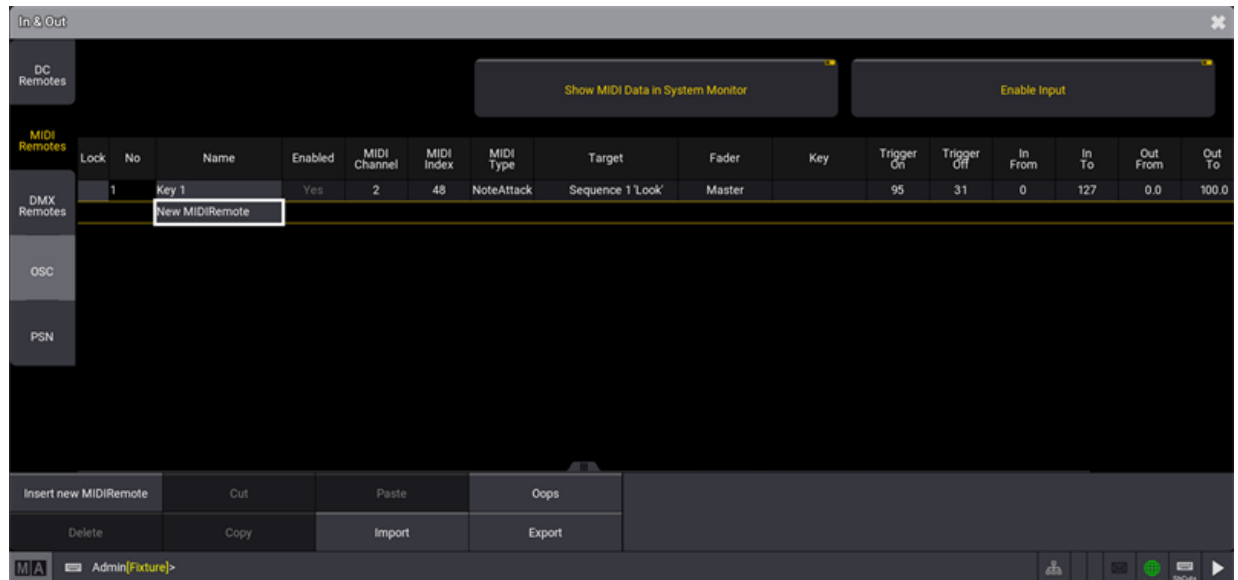
Choose a target in the Assignment Editor

- To open the Select Fader pop-up, edit the empty Fader column. Set the Fader to **Master** as shown in the image below:



Select Fader pop-up.

- The new MIDI remote is set and the MIDI Remotes window should look like this:



A new MIDI remote is set.



Hint:

The MIDI output channel can be changed on the external MIDI device.



Hint:

To reproduce this example using syntax, see **Remote** keyword.

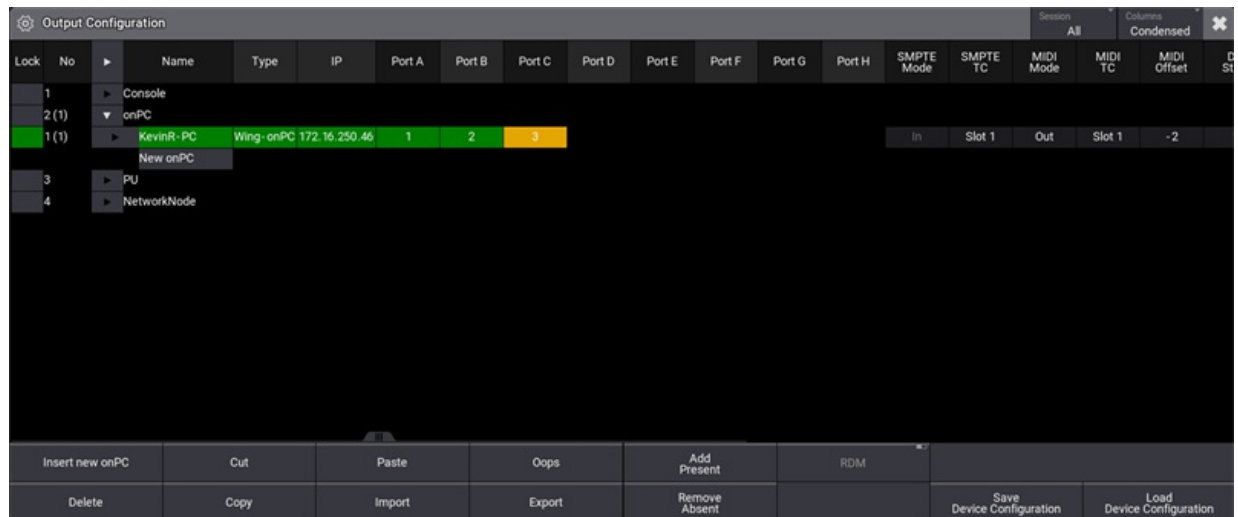


Send MIDI

It is also possible to send MIDI messages from a grandMA3 console to an external MIDI device.

To send MIDI messages from a grandMA3 console:

- Connect the MIDI cable to the MIDI data input connector (MIDI In) from the external MIDI device to the MIDI data output connector (MIDI Out) on the console.
- MIDI messages can be sent using the **SendMIDI** keyword.
- To send MIDI messages, make sure the MIDI Mode is set to **Out** in the Output Configuration menu:



MIDI Mode is set to Out in the Output Configuration menu.

The given MIDI channel (1 to 16) in the command line will be used to send out a MIDI message. For the example below, MIDI channel 10, Index (note) 46 and velocity 99/0 is used:



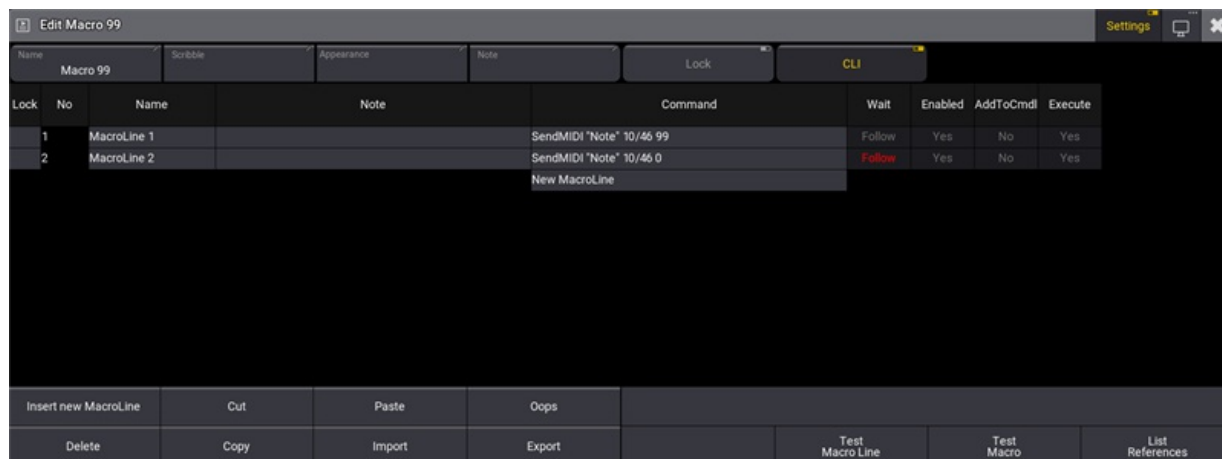
For the example, MIDI channel 10 is used.



The MIDISend command applies the following rules:

- If a MIDI channel is entered in the command, the entered MIDI channel will be used. If no MIDI channel is entered, MIDI channel 1 will be used.
- If a velocity is entered in the command, the entered velocity will be used. If no velocity is entered, a velocity full (127) will be used.
- If no status (On or Off) is entered in the command, On will be used.

Depending on the external MIDI device, it is necessary to send a MIDI message to the external MIDI device that mimics the release of a hardkey, by sending an **Note Off** (Value 0) message. This is shown in the image below:



A button is assigned to a send a MIDI message with velocity on and off.

MIDI Connection via USB

A external device can also be connected to the grandMA3 onPC Software using a third-party USB MIDI device.



Restriction:

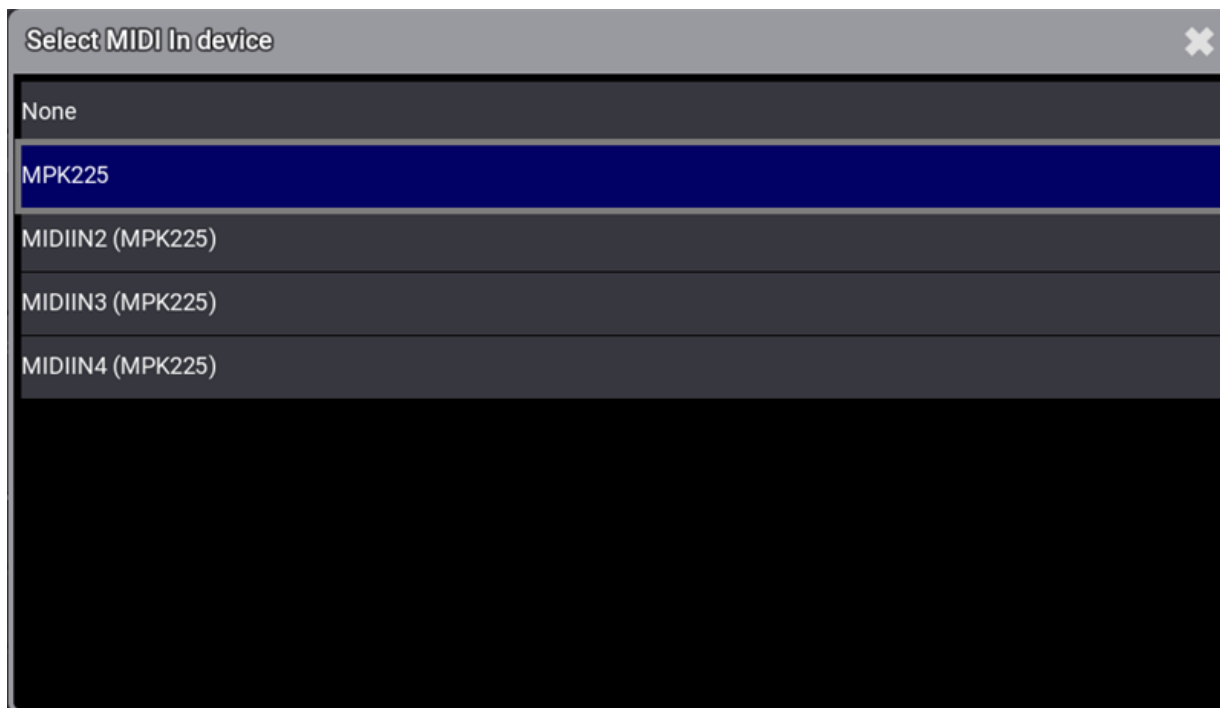
Connecting an external MIDI device via USB is only possible using the grandMA3 onPC Software.

Requirements:

- 1x grandMA3 onPC Software
- 1x MIDI capable device
- 1x USB cable

To connect an external MIDI device via USB:

1. Connect both, the external MIDI device and computer, using an USB cable.
2. Press **Menu**, tap **Settings** and then tap **onPC Settings**. The onPC Settings window opens.
3. To select the external MIDI device, tap **MIDI In Device**. The Select MIDI In device pop-up opens:
4. Select the external MIDI device. In the example it is **MPK225** as shown in the image below:



The Select MIDI IN device pop-up shows the connected keyboard.

5. The external MIDI device is connected to the grandMA3 onPC Software. To set up a MIDI remote, go back to **Receive MIDI**.



Hint:

For more information about the grandMA3 onPC settings, see [**onPC settings**](#).



45. Remote Device Management (RDM)

Remote Device Management is a protocol that allows bi-directional communication between a grandMA3 device and RDM-ready devices attached to it (= RDM-ready fixtures) over a standard DMX line. ANSI E1.20 - 2010 by PLASA specifies the RDM standard as an extension of the DMX 512-A protocol (ANSI E1.11).

Manual settings, such as adjusting the DMX starting address, are no longer needed. This is especially useful for devices installed in a remote area.

RDM is integrated in DMX without influencing the connections. The RDM data is transmitted via the standard XLR-poles – new DMX cables are not required. RDM-ready and conventional DMX devices can be operated in one DMX line. The RDM protocol sends its own data packages in the DMX512 data feed and does not influence conventional devices.

To be able to use RDM, RDM has to be enabled at two different spots:

1. Globally within the show file:

- Tap **RDM** in **Menu** - **Network** ,
- Or tap **RDM** in **Menu** - **Live Patch** - **RDM** .
- Or tap **RDM** in **Menu** - **Connector Configuration** . All three RDM buttons are linked to each other.

2. Per XLR port that shall use RDM. Therefore, the mode of an XLR Port in the Connector Configuration needs to be set from Out to RDM.

In RDM mode, DMX data is only sent when there are changes of DMX values. In addition, every 500ms a refreshing packet will be sent so that DMX fixtures will not switch into DMX fail mode.

This RDM output mode allows more time on the DMX line for RDM configuration.

The RDM Devices window lists all devices that are discovered via RDM. It can be opened via the Add Window dialog and is located in the **Tools** -tab.

The same list is also displayed in **Menu** - **Live Patch** - **RDM** .

When an RDM fixture is detected on an XLR port, a new node called RDMPort will be added to the RDM devices list. Within each RDMPort node, all fixtures that are detected via RDM on this physical XLR port are listed. An RDM port is labeled with the IP of the device and the XLR port of the device, e.g., 192.168.0.4 - XLR D. If the device is not available anymore with the port, the font color turns red.

For each different fixture type, detected per RDM, a new node in the RDMFixtureTypes node in the RDM Devices window will be created. Each RDMFixtureType contains general information of the RDM fixtures, that is similar to all fixtures of the same product, e.g., the parameter description, or the available DMX personalities.

The grandMA3 creates RDMFixtureTypes by itself depending on these three parameters: ManufacturerID, DeviceModelID, and SoftwareID.

As soon as the same physical type of lighting fixtures have different software IDs due to different firmware versions, different RDMFixtureTypes are created.



This first implementation of RDM supports these RDM parameters, which can be set by the user:

- IDENTIFY_DEVICE, 0x1000
- DEVICE_LABEL, 0x0082
- DMX_PERSONALITY, 0x00E0
- DMX_START_ADDRESS, 0x00F0
- PAN_INVERT, 0x0600
- TILT_INVERT, 0x0601
- DISPLAY_INVERT 0x0500
- DISPLAY_LEVEL 0x0501
- RESET_DEVICE 0x1001
- FACTORY_DEFAULTS 0x0090
- LAMP_STATE 0x0403

Only the cells of properties that an RDM fixture provides as set-able can be edited in the RDM devices window.

In grandMA3 the RDM communication follows this process:

1. Discovery for new RDM fixtures.
 1. Check if detected fixtures are still available.
 2. Check for new RDM fixtures.
2. Get parameter and sensor data.
3. 1s Pause
4. Start again at 1.

Parameters that are not changing during the runtime of a fixture, e.g., Device Info, are only pulled via RDM once when creating the corresponding RDMFixtureType. All other parameters and sensors are pulled every time in step 2.

As soon as an RDM fixture is not available for 3 discoveries in a row, it will be displayed in red in the list of RDM fixtures.

It is possible to match an RDM fixture with a fixture of the grandMA3 patch. To do so, edit the fixture cell of the desired RDM fixture in the RDM devices window. A pop-up opens and offers all fixtures of the current show file. In addition, it is also possible to match fixtures within the RDM window in the live patch. In the live patch, it is possible to open the fixture list in the same way as described above, and by selecting any cell of a fixture then tapping **Match** at the bottom of the window.

To remove a match between an RDM fixture and a grandMA3 fixture, it is possible to tap **Unmatch** in the RDM window in the live patch or to tap **Clear** in the match pop-up.



46. Control other MA Devices


This section describes how to control other MA devices via the console.



46.1. Control grandMA3 Nodes

To adjust the settings in the grandMA3 Nodes, it may be convenient to control them from a connected console or onPC.

For more information, read the [Session topic](#) in the [Networking section](#).

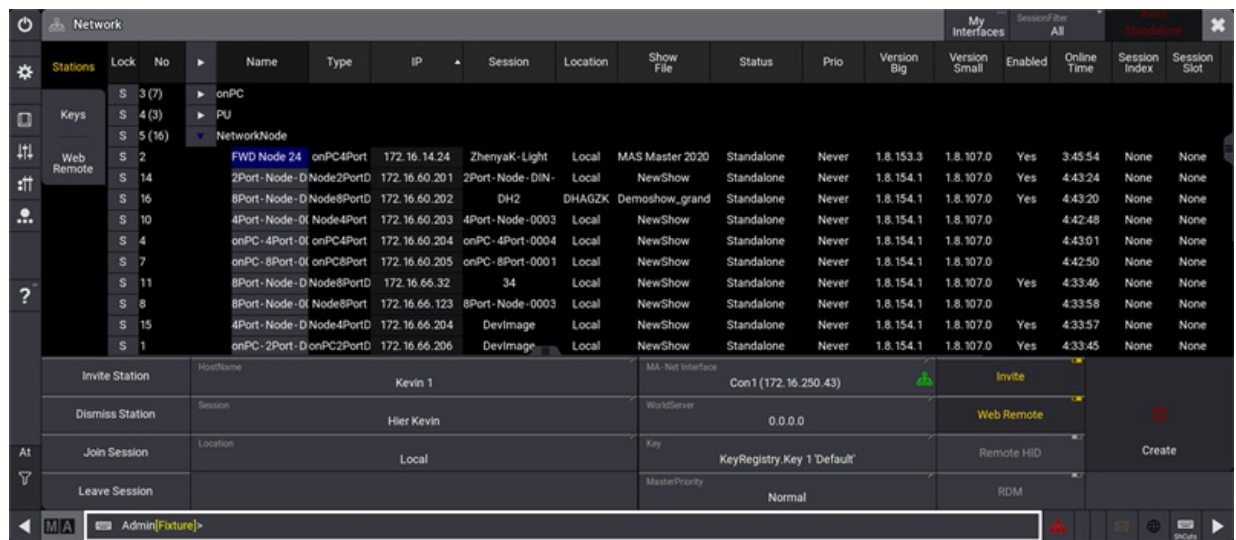


Hint:
 If you want to use Art-Net or sACN using grandMA3 nodes, make sure to switch the console to **Mode2**.
 For information on how to configure the xPort Nodes in Mode2 see [Configure xPort Nodes in the Console](#).


Change Name and Set IP Address

To change the name and to set the IP address of a grandMA3 Node, open the Network Interface Menu:

- Press **Menu** .
 - Opens the [menu select pop-up](#).
- Tap **Network** .
 - Opens the Network menu.



Network menu

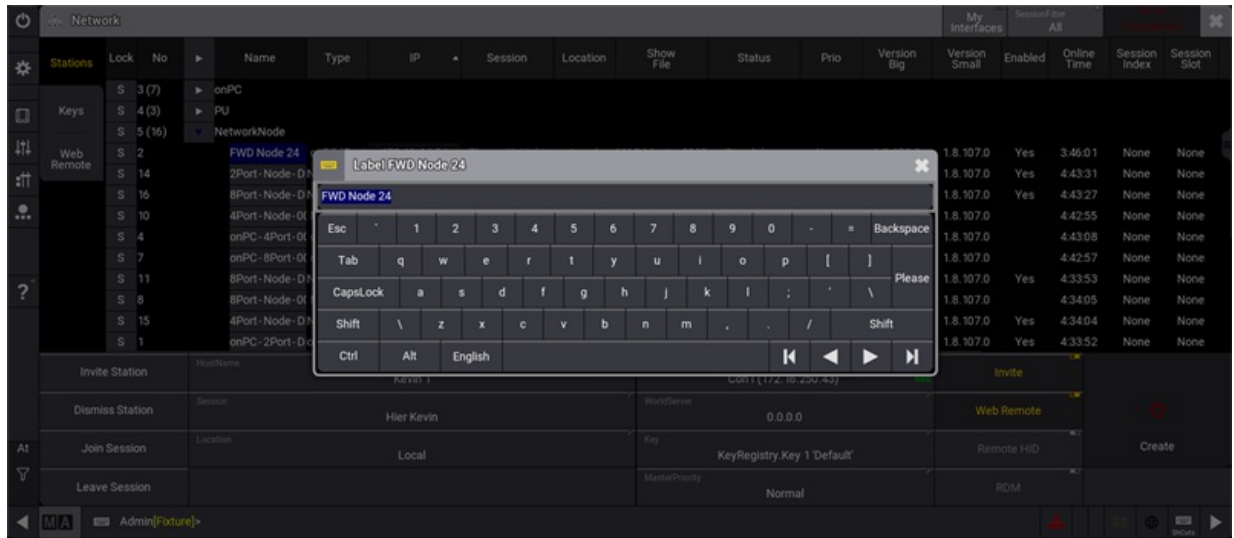


Important:
 You can only change the name or IP address of a node if it is not in another session.



Change the Name

1. Edit the name of the respective node by right-clicking with a mouse or by using **gestures**.



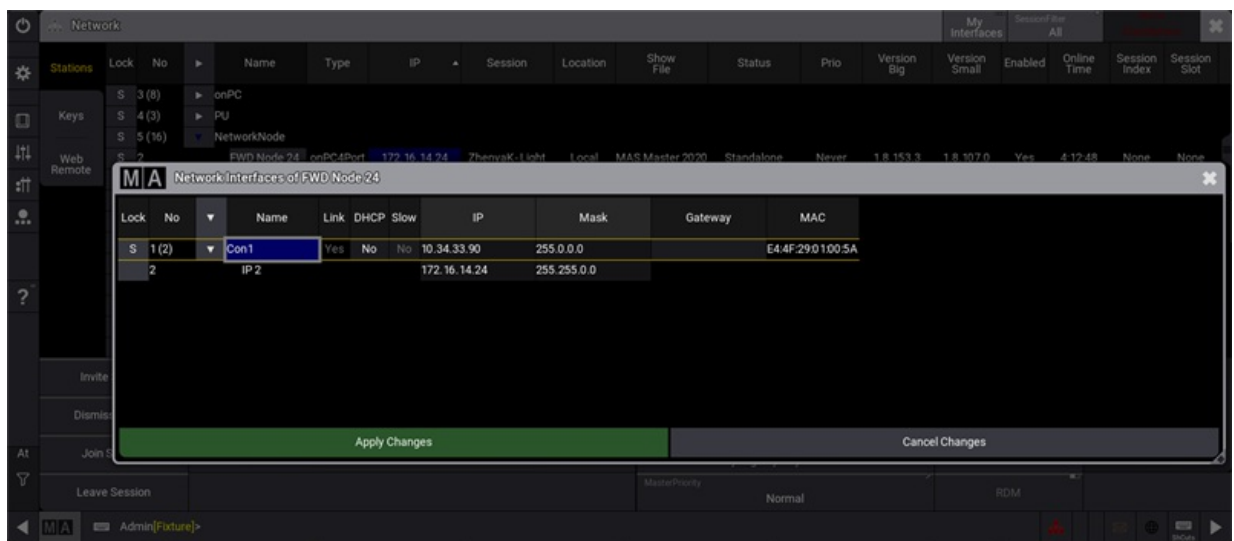
Label Node pop-up

2. Enter the desired name.
3. To confirm the changes, press **Please**.

The Node has the new name.

Change the IP Address

1. Change the IP address of the respective node by right-clicking with a mouse or by using **gestures**.
2. Edit the DHCP settings by right-clicking the DHCP field with a mouse or by using **gestures**.



Network Interfaces pop-up of a Node



3. Set DHCP to **Yes** or **No**.
4. Edit the IP address of the respective node by right-clicking with a mouse or by using **gestures**.
The Edit IP pop-up opens.
5. Enter the new IP address.
6. Tap **Apply Changes** to confirm the changes.

The new IP address is set.

For more information about the network menu, read the **Network Update topic** in the **Update the Software section**.



Hint:

It is also possible to update the Nodes via **manual update**.

Change the Output Configuration

To adjust the output configuration settings in the grandMA3 Nodes, control them from a connected console or onPC.

For more information, read the **DMX port configuration topic** in the **DMX In and Out section**.

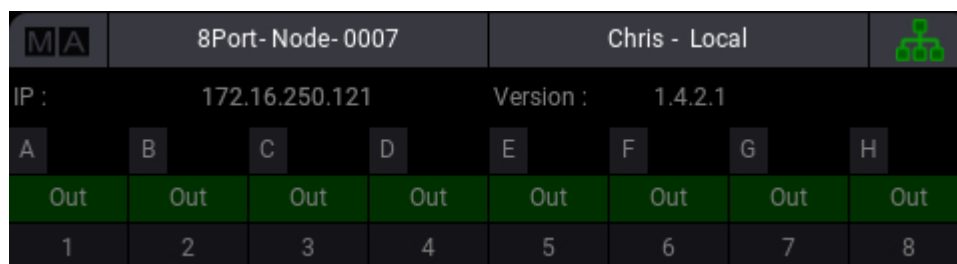
To connect a grandMA3 Node with the console, set the Node to grandMA3 mode.

Change Modes

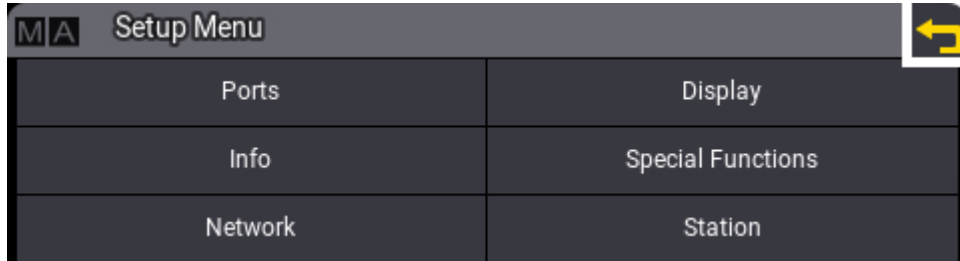
From grandMA3 to Mode2

- To change the Node from grandMA3 to Mode2, follow these instructions:

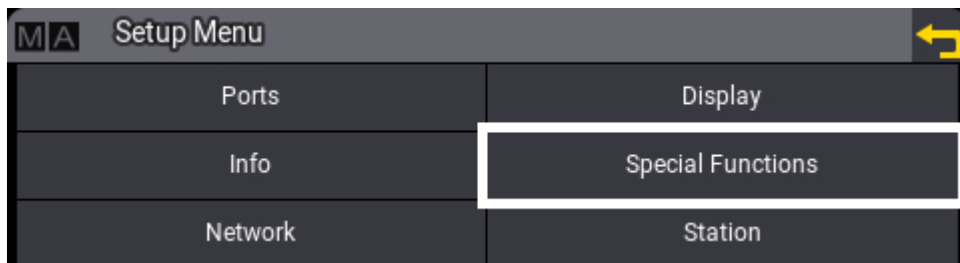
grandMA3 screen:



- To switch to Mode2, press the rotary knob:



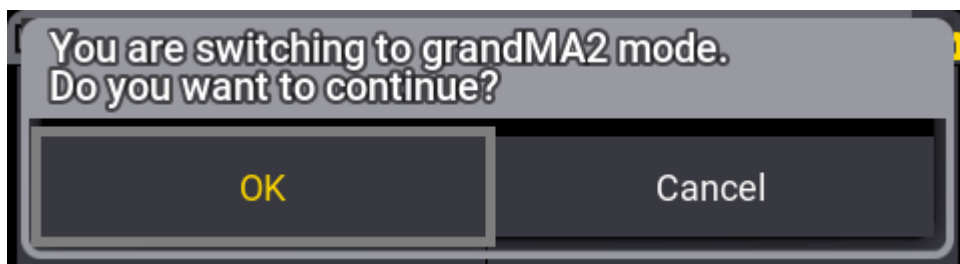
- Turn the rotary knob to select **Special Functions** and press the rotary knob to confirm:



- Select **Switch to grandMA2** and press the rotary knob to confirm:



- Select **OK** and press the rotary knob to confirm:



The Node reboots in Mode2.

For more information about how to switch a xPort Node from Mode2 to grandMA3, see [change modes](#).

For more information about changing to grandMA3 Mode2, read the [Mode2 topic](#) in the section grandMA3 Mode2 of the [grandMA2 User Manual](#).



46.2. Control an MA Network Switch

The MA Network Switch is the perfect device for a full lighting control solution. The combination of console, switch, and networking devices provides the easiest solution even for complex lighting systems.

You can control the MA Network Switch by using the web interface or directly with a grandMA3 console in Mode2 or a grandMA2 onPC. For more information about how to control the MA Network Switch by using the web interface see the [**MA Network Switch manual**](#).

To learn how to control the MA Network Switch from a grandMA3 console in [**Mode2**](#) or a grandMA2 onPC, read the topic [**Control the MA Network Switch**](#) in the [**grandMA2 User Manual**](#). There, you will find out how to add the MA Network Switch to the network configuration, configure ports, or edit groups and presets.

For more information about changing to grandMA3 Mode2, read the [**Mode2 topic**](#) in the section grandMA3 Mode2 of the [**grandMA2 User Manual**](#).

46.3. RemoteHID

The Remote HID feature allows using the local HID devices (typically mouse and keyboard) to control other grandMA3 stations that are available in the network as if the local HID devices are connected to the remote device.

A use case for this feature is controlling onPC stations that are used for visualization right at a console. Instead of placing several mice and keyboards on the table, only one set of mouse and/or keyboard is needed.



Restriction:

It is not a remote desktop feature. The display of the remote device needs to be visible.

To control another station from the local console, follow these steps:

1. Enable **RemoteHID** in **Menu** - **Network** menu on the remote station.
2. Execute the RemoteHID command on the local station:

RemoteHID IP [remote_ip_address]

- Or -

RemoteHID [remote_station_type] ["remote_hostname"]

3. When the connection has been established, the screen of the local station changes to olive green. During the connection, only the executors including the 100mm faders and **Go+[Large]**, **Go-[Large]**, and **Pause[Large]** remain usable on the local station. All other elements are blocked.

The remote function can be ended using one of the following options:

- Press **MA** + **MA** + **Off**
- Use the keyboard shortcut **Shift** + **Ctrl** + **Alt** + **E**


Example

A preprogramming setup where there are a console and a powerful graphics computer running grandMA3 onPC optimized for 3D visualization. They are connected in a session on a local LAN network.

The grandMA3 onPCs hostname is "3D" and RemoteHID is enabled.

From the console, there is a need to change a setting in the 3D window on the grandMA3 onPC.

In the command line type:

```
 User name[Fixture]> RemoteHID onPC "3D"
```

Now the mouse and keyboard connected to the console can be used to operate the computer.



Finish the connection by pressing **MA** + **MA** + **Off**



Important:

grandMA3 onPC on Windows controlled via RemoteHID, is not only limited to control the grandMA3 onPC application. Furthermore, the mouse and keyboard can also access remotely the entire system.



Restriction:

RemoteHID can only be used on grandMA3 consoles and grandMA3 onPC on Windows.



Restriction:

At the moment, it is not possible to use the touch displays of grandMA3 consoles to control the mouse remotely.



47. Update the Software

Every grandMA3 device is delivered with the latest version of the grandMA3 software.

All devices in a network with software versions higher than 1.0 can see and update each other.

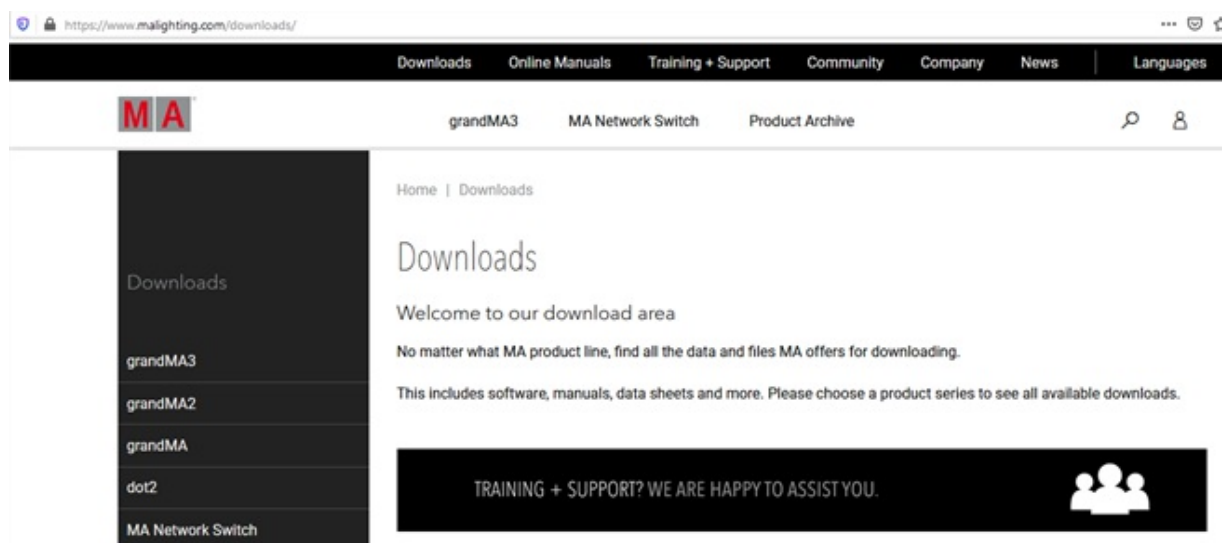
This is to give you information on how to update or downdate the grandMA3 software if needed.

To check which version your device is currently running, use the **Version keyword** in the command line (does not work with PUs and Nodes).

Check for Updates and Download the Latest Version

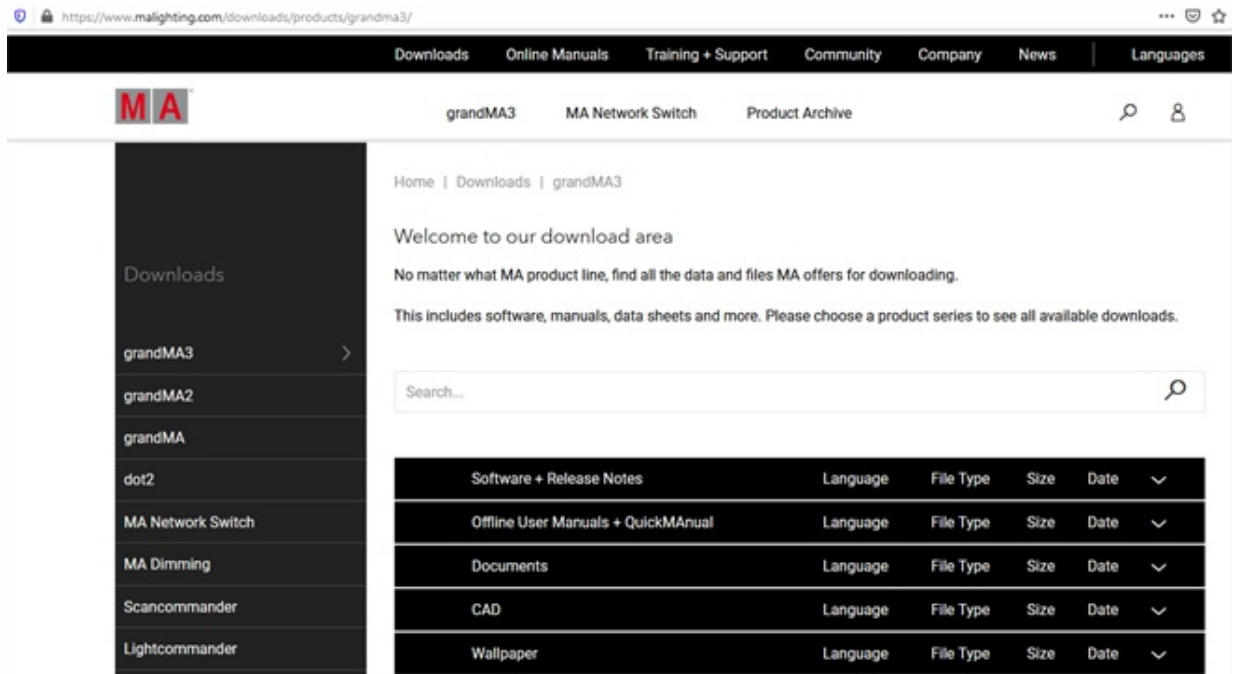
To check if there is a new grandMA3 software update:

1. Go to www.malighting.com, click **Downloads**.



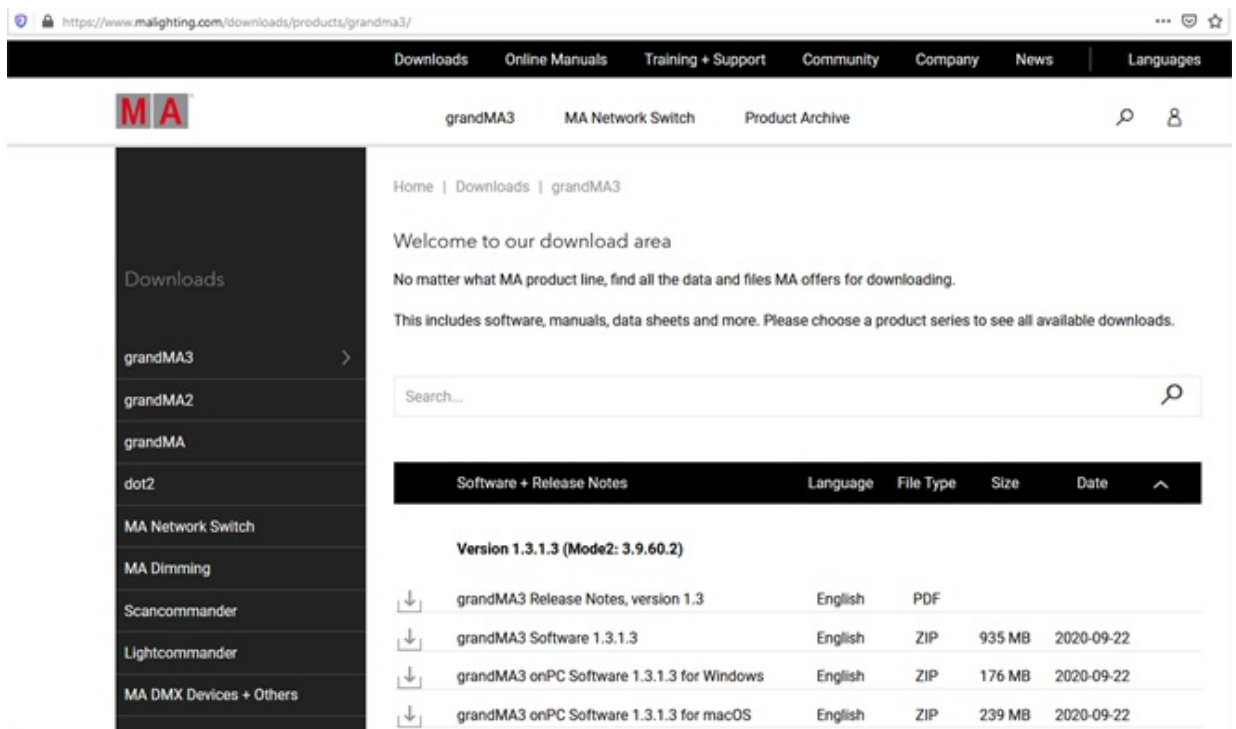
Website malighting.com Downloads

2. Click **grandMA3** in the bar on the left.



grandMA3 section

3. Click **Software + Release Notes** to find the latest software version.



grandMA3 Software section

4. Click the current version to download the desired installation package.



Important:

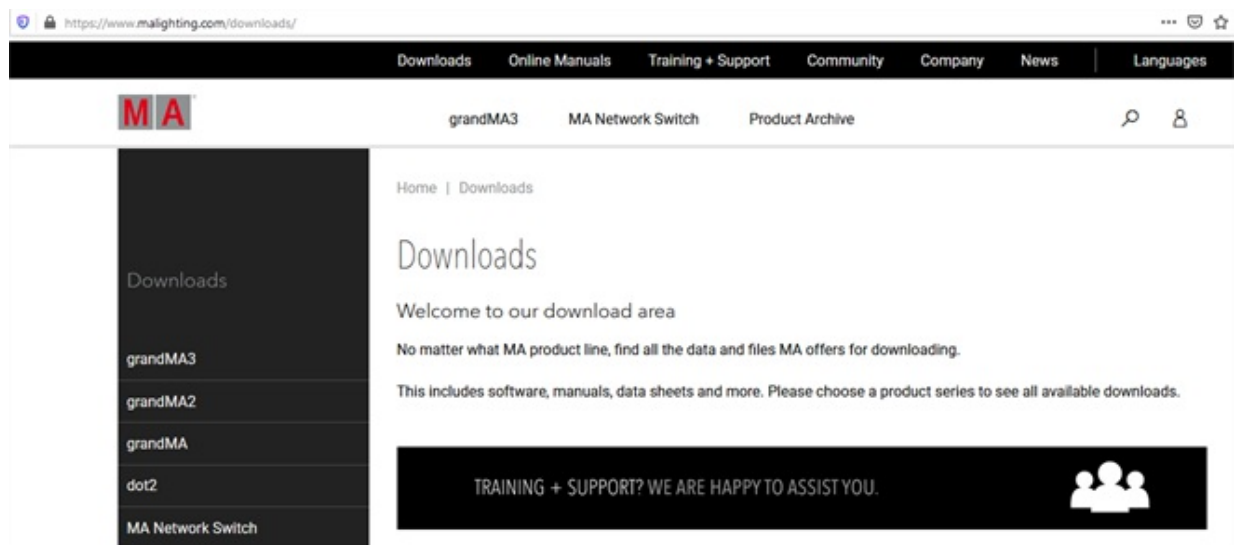
Every grandMA3 device except with Windows or macOS operating system uses the same software and needs the main installation package grandMA3 Software.
To update the grandMA3 software on a PC or laptop, choose the installation package grandMA3 onPC Software for Windows or grandMA3 Software for macOS.
For network updates via PC or laptop, you also need to download the installation package grandMA3 onPC Software for the connected grandMA3 devices.

The download process starts.

Download an Older Version

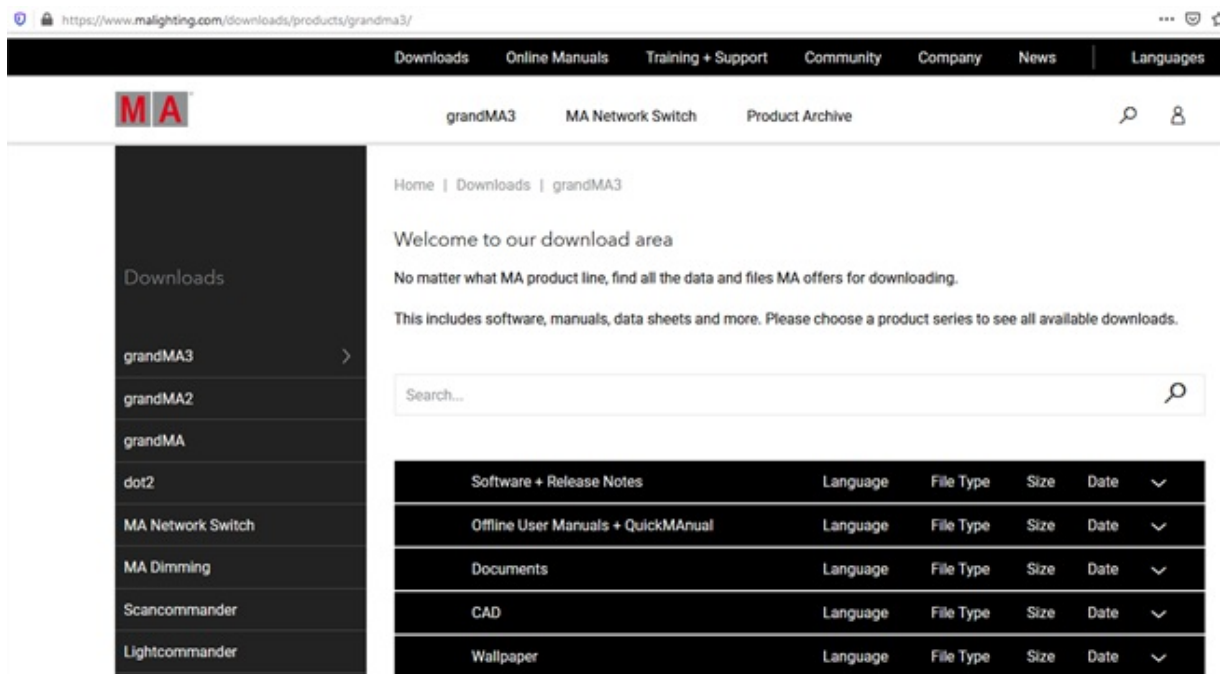
To downdate a grandMA3 device to an older software version, e.g. in a permanent installation, follow these steps:

Go to www.malighting.com, click **Downloads**.



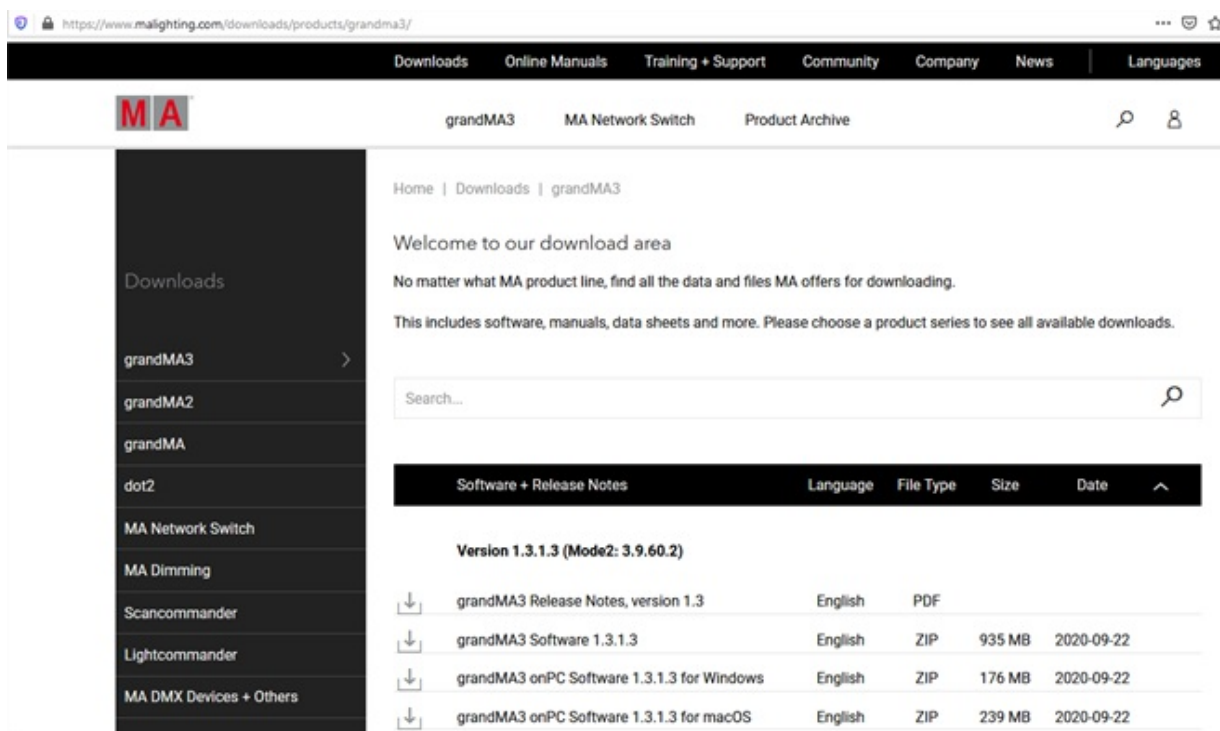
Website [malighting.com](https://www.malighting.com) Downloads

2. Click **grandMA3** in the bar on the left.



grandMA3 section

3. Click **Software + Release Notes**.



grandMA3 Software section

4. Scroll down and click **Archive** to find older software versions.



The screenshot shows the MA Lighting website's 'grandMA3' software archive page. The page features a dark navigation bar with links for Downloads, Online Manuals, Training + Support, Community, Company, News, and Languages. Below the navigation bar, there is a search bar and a list of product categories including grandMA3, grandMA2, grandMA, dot2, MA Network Switch, MA Dimming, Scancode, Lightcommander, and MA DMX Devices + Others. The main content area displays a table of software releases under the heading 'Software + Release Notes'. The table has columns for Language, File Type, Size, and Date. The current version shown is 'Version 1.3.1.3 (Mode2: 3.9.60.2)'. Below this, there is an 'Archive' section with a list of previous versions and their corresponding Mode2 numbers.

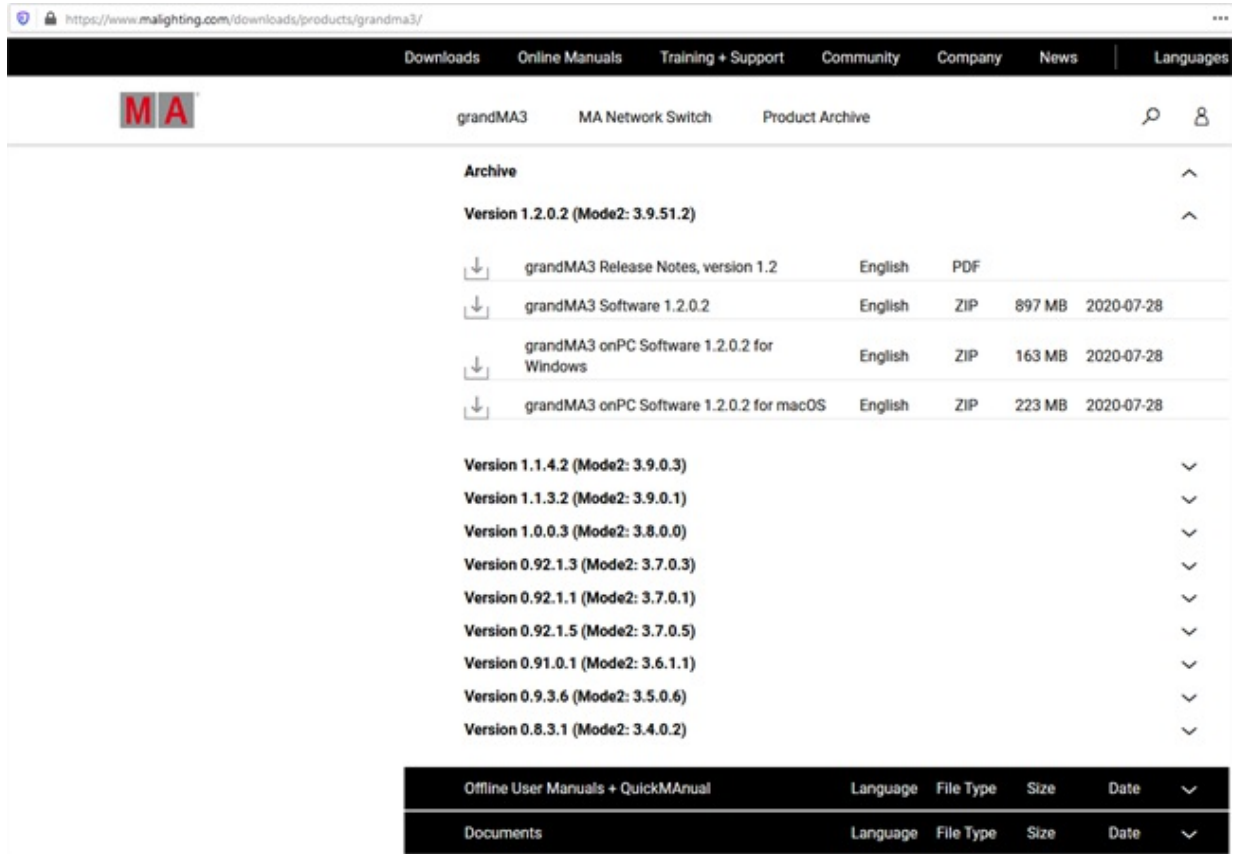
Language	File Type	Size	Date
English	PDF		
English	ZIP	935 MB	2020-09-22
English	ZIP	176 MB	2020-09-22
English	ZIP	239 MB	2020-09-22

Archive

- Version 1.2.0.2 (Mode2: 3.9.51.2)
- Version 1.1.4.2 (Mode2: 3.9.0.3)
- Version 1.1.3.2 (Mode2: 3.9.0.1)
- Version 1.0.0.3 (Mode2: 3.8.0.0)
- Version 0.92.1.3 (Mode2: 3.7.0.3)
- Version 0.92.1.1 (Mode2: 3.7.0.1)
- Version 0.92.1.5 (Mode2: 3.7.0.5)
- Version 0.91.0.1 (Mode2: 3.6.1.1)
- Version 0.9.3.6 (Mode2: 3.5.0.6)

grandMA3 Software archive

5. To open the folder, click the desired version, e.g. Version 1.2.0.2.



grandMA3 Software archive versions

6. Click the desired installer version to download the required installation package.

The download process starts.

Available Installation Packages

There are two installation packages available:

- grandMA3_vx.x.x.x_full_package.zip

Use the full package to update the grandMA3 installations:

Console, RPU, processing unit, onPC (Windows and macOS), viz-key (Windows and macOS), xPort Node, and I/O node.

- grandMA3_vx.x.x.x_onPC_package.zip

Use the onPC package to update the grandMA3 onPC software installations:

onPC (Windows and macOS), and viz-key (Windows and macOS), xPort Node, and I/O node.



Store the Installation Packages

All installation packages for grandMA3 software are included in one zip-file.

Extract the entire data from the installation package zip-file.

The folders EFI, ma, and other data included are extracted.

For Windows systems, copy the files from the ma folder into the directory

C:\ProgramData\MALightingTechnology\installation_packages.

For macOS systems, copy the files from the ma folder into the directory

~/MALightingTechnology/installation_packages.

Follow all the onscreen instructions that appear during the update.

47.1. Update grandMA3 Consoles

This topic describes the software update with additional options such as factory reset, etc.

For a simple software update via the user interface, follow the instructions in the **Network Update topic**.

For a processing unit, the following update process is also valid.



Important:

- The folders EFI, ma, and the update.scr file have to be directly accessible on the flash drive and must not be located in an extra folder.
- The USB flash drive's data system has to be FAT32.

Download the Software Package

1. To update a grandMA3 device, download the latest software version from **www.malighting.com**.
The required installer is called grandMA3 Software x.x.x.x.
2. Extract the zip file "grandMA3_stick_v.x.x.x.x.zip" and copy the folders EFI, ma, and the update.scr file into the root directory of your USB flash drive.

Install the Software Package on the Console

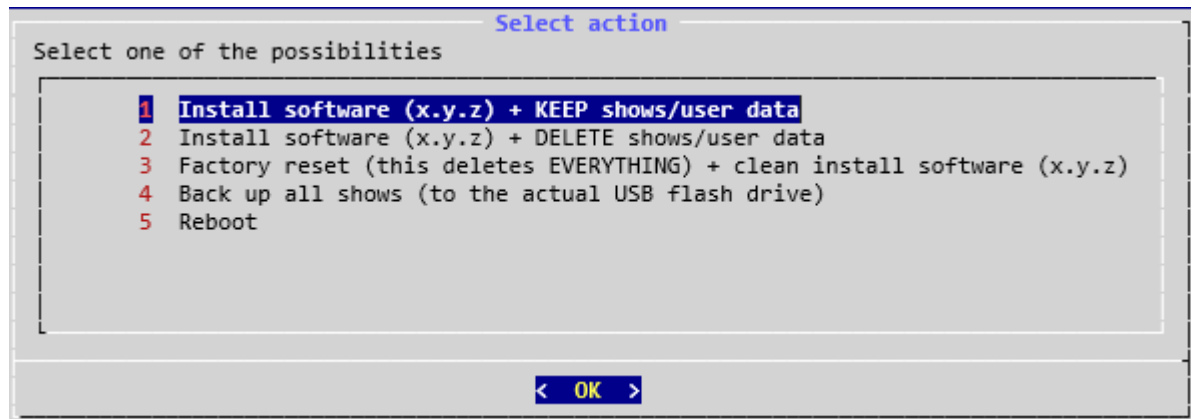
1. Turn off the grandMA3 device.
2. Insert the USB flash drive in a USB port.
3. For devices without an integrated keyboard (e.g. grandMA3 compact console), connect an external keyboard with a USB port.
4. Turn on the grandMA3 device.
5. Press the key **8/F8** on the internal or the external keyboard several times.
The **Boot Manager** opens.
6. Scroll down to EFI USB Device using the arrow keys.
7. Press **Enter** on the (external) keyboard.
The console starts to boot. The EULA screen opens.
8. Accept the EULA.



Important:

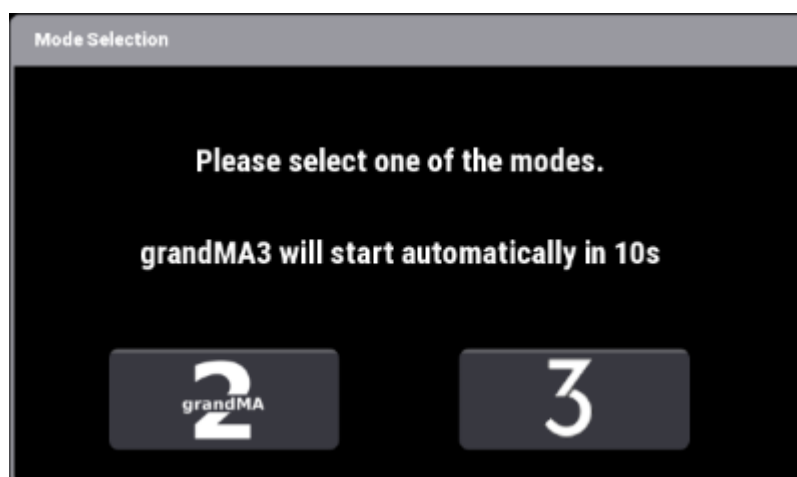
If the USB flash drive contains more than 1 version, select the version you would like to install first.

The Install Selector dialog appears:



- Select option 1 to keep the shows and user data.
 - Select option 2 to delete the shows and user data.
 - Select option 3 for a complete factory reset to clean the system before installing the software.
 - Select option 4 to save the shows on the USB flash drive.
 - Select option 5 to reboot the grandMA3 device.
1. Press **Enter** on the (external) keyboard.
Wait for completion. The grandMA3 device reboots.
 2. Remove the USB flash drive.

The Mode Selection dialog appears.



Select mode

3. Select one of the modes. For more information about grandMA3 mode2, read the **Mode2 topic** in the section grandMA3 Mode2 of the **grandMA2 User Manual**.
If no selection is made, the device automatically starts in grandMA3 mode.



4. Screens 1, 2, 3 are initializing.
The letterbox screens go into self-test mode (red, green, blue, white, and black color changer).
The command screens stay black.
It can take several seconds for them to start initializing.
5. The installation is complete.

47.2. Update grandMA3 Nodes



Important:

We recommend formatting with every update.
Whenever formatting, the IP address will be reset back to default (DHCP).

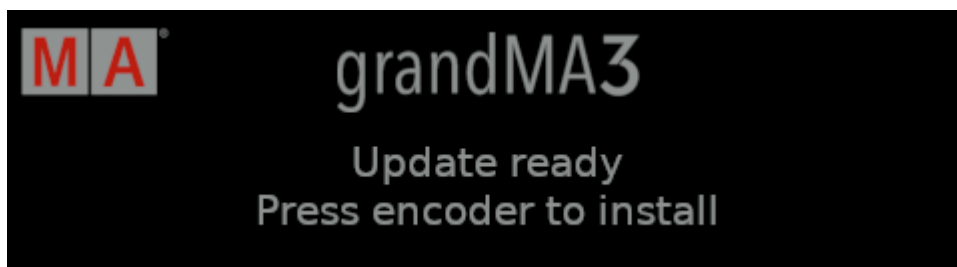


Important:

-The folders EFI, ma, and the update.scr file have to be directly accessible on the flash drive and must not be located in an extra folder.
-The USB flash drive's data system has to be FAT32.

Requirement:

1. To update a grandMA3 device, download the latest software version from www.malighting.com.
The required installer is called grandMA3 Software x.x.x.x.
2. Extract the zip file "grandMA3_stick_v.x.x.x.x.zip" and copy the folders EFI, ma, and the update.scr file into the root directory of your USB flash drive.
3. Insert the USB flash drive in the device's USB port.
4. Turn off the grandMA3 device.
5. Turn on the grandMA3 device.
6. Follow the onscreen instructions during the update process.

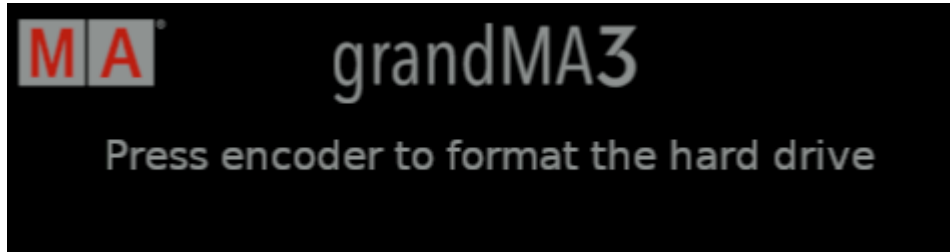


Update notification

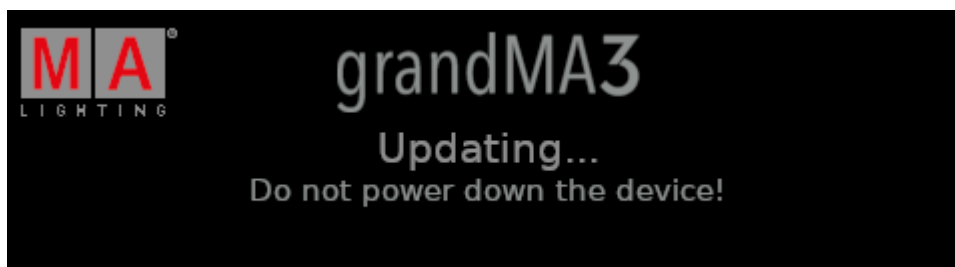
- Press the rotary knob.



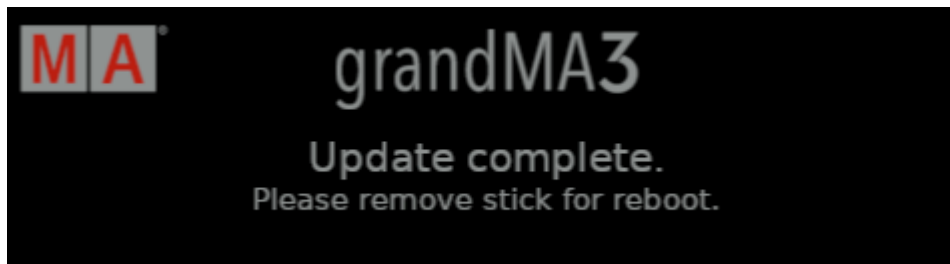
Corrupted installer package



Format the hard drive



grandMA3 xPort Node is updating



Update completed

47.3. Update grandMA3 onPC Windows Hardware

Updating the onPC Software

Updating the grandMA3 onPC software for the grandMA3 onPC windows hardware like command wing XT or rack-unit requires a normal grandMA3 onPC software update.

For more information, see the [Network update topic](#).



Important:

Select the Windows installer version!

Reset to Factory Defaults

To reset the grandMA3 onPC command wing XT or rack-unit to factory defaults, it is necessary to install the full software image onto the hard drive including all operating system and grandMA3 onPC software information.

The installer package always installs the latest version of the onPC software and the Windows updates!

- To update, download the latest software version from www.malighting.com.
The required installer is called grandMA3 onPC Windows Hardware Image x.x.x.x.
- Save the full software image on your hard drive.
- Extract the entire data from the zip file and copy it onto a USB flash drive (FAT32 formatted, minimum 8 GB).
- Insert the USB flash drive.
- Power up the grandMA3 command wing XT or rack-unit.
- If necessary, tap F8 to choose Boot device and select boot from USB.

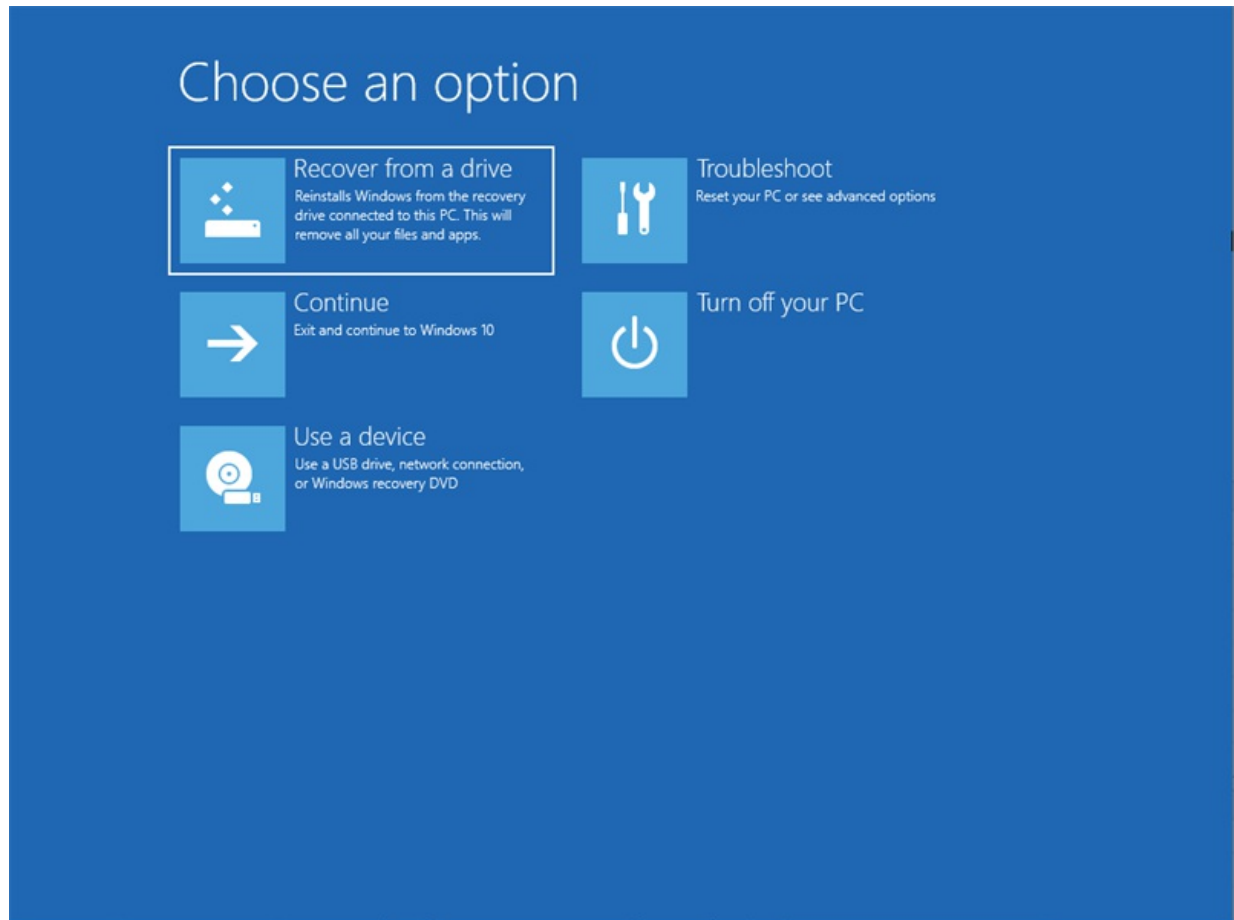


Choose your keyboard layout

- US
- Albanian
- Arabic (101)
- Arabic (102)
- Arabic (102) AZERTY
- Armenian Eastern (Legacy)
- Armenian Phonetic
- Armenian Typewriter
- Armenian Western (Legacy)
- Assamese - INSCRIPT
- Azeri Cyrillic
- Azeri Latin

[See more keyboard layouts](#)

- Choose a keyboard layout: (US).



- Choose an option: Recover from a drive.



Hint:

If the "Recover from a drive" icon does not appear, the USB flash drive may be faulty. Try another USB flash drive and make sure that the files are properly unzipped.

Recover from a drive

Do you want to fully clean your drive? When you remove your files, you can also clean the drive so that the files can't be recovered easily. This is more secure, but it takes much longer.



Just remove my files

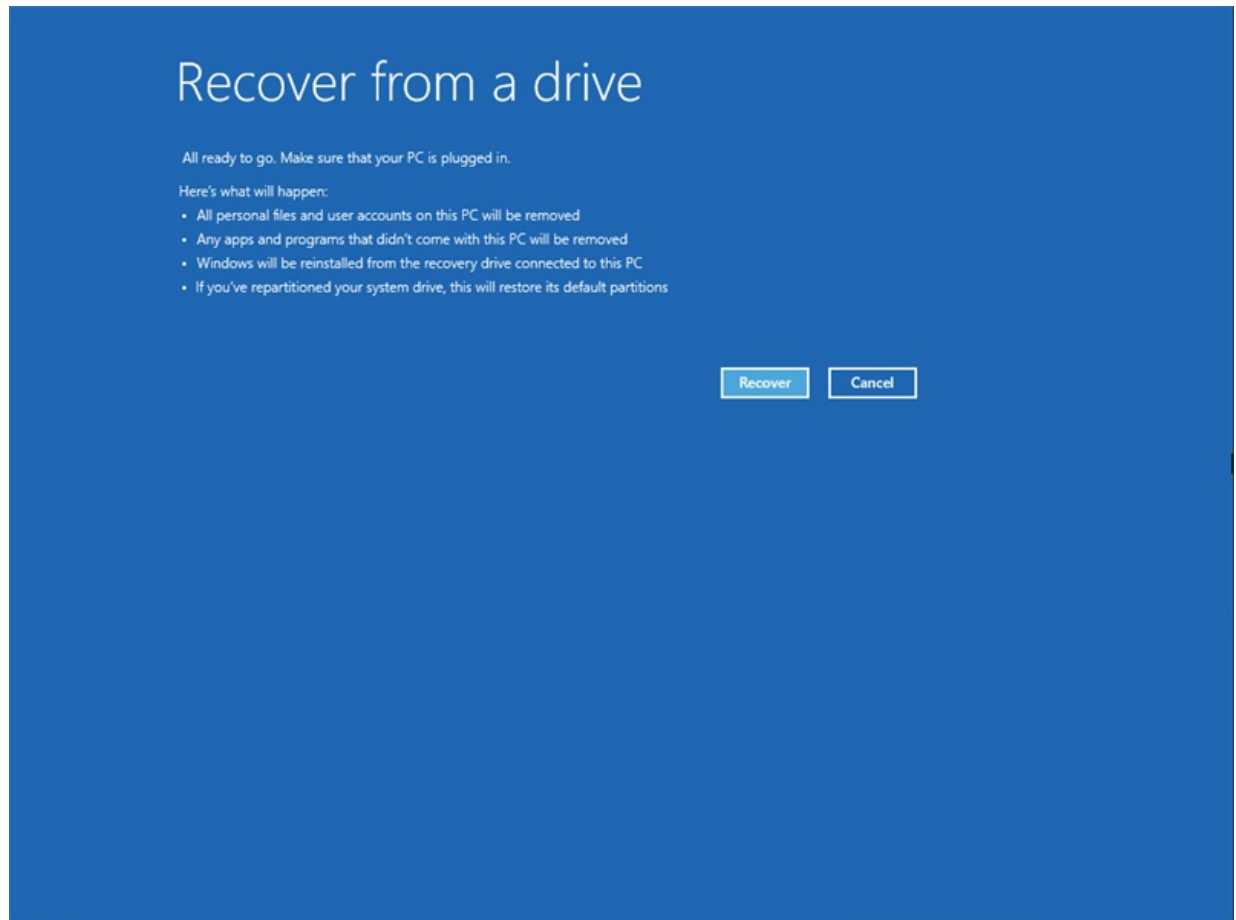
Use this if you're keeping your PC.



Fully clean the drive

Use this if you'll recycle the PC. This can take several hours.

- Select Recover from a drive: Just remove my files.

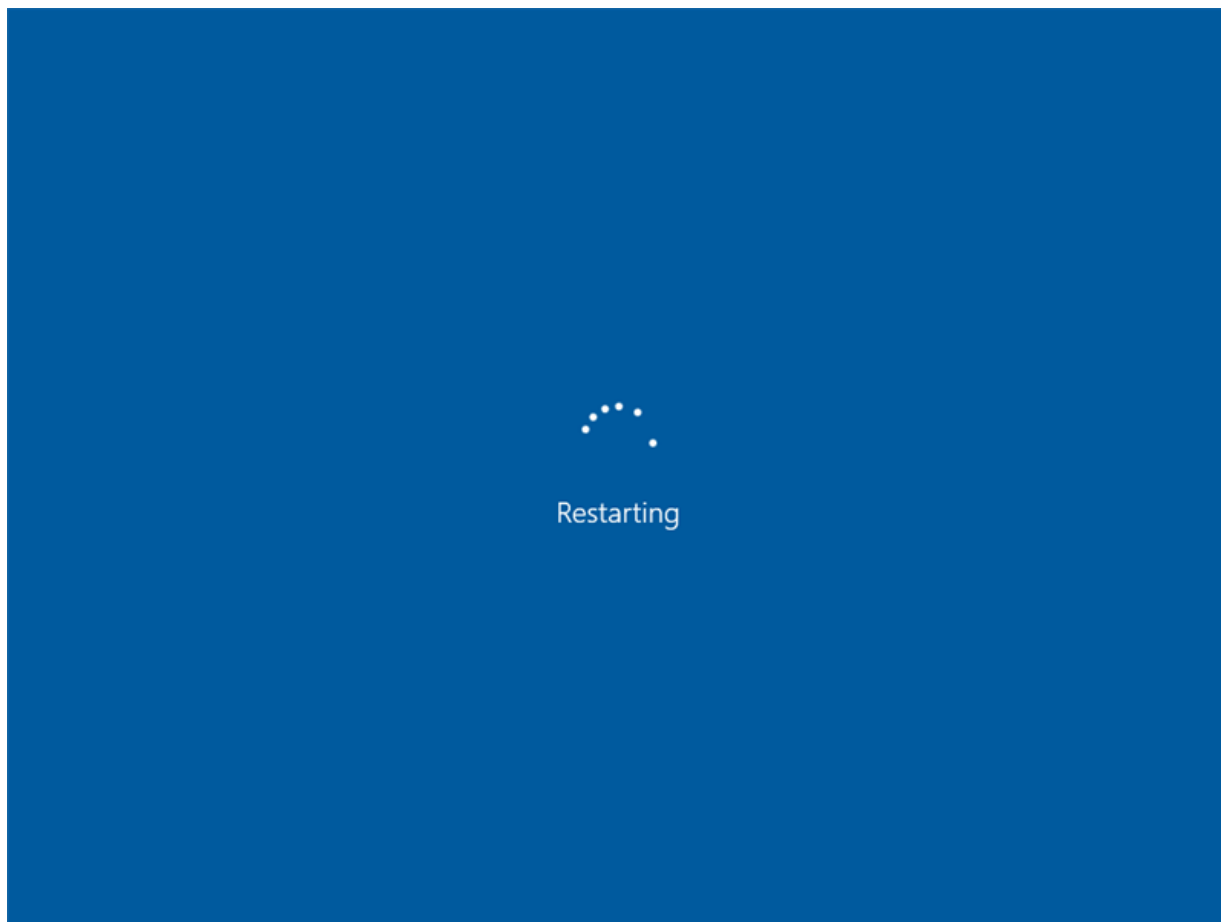


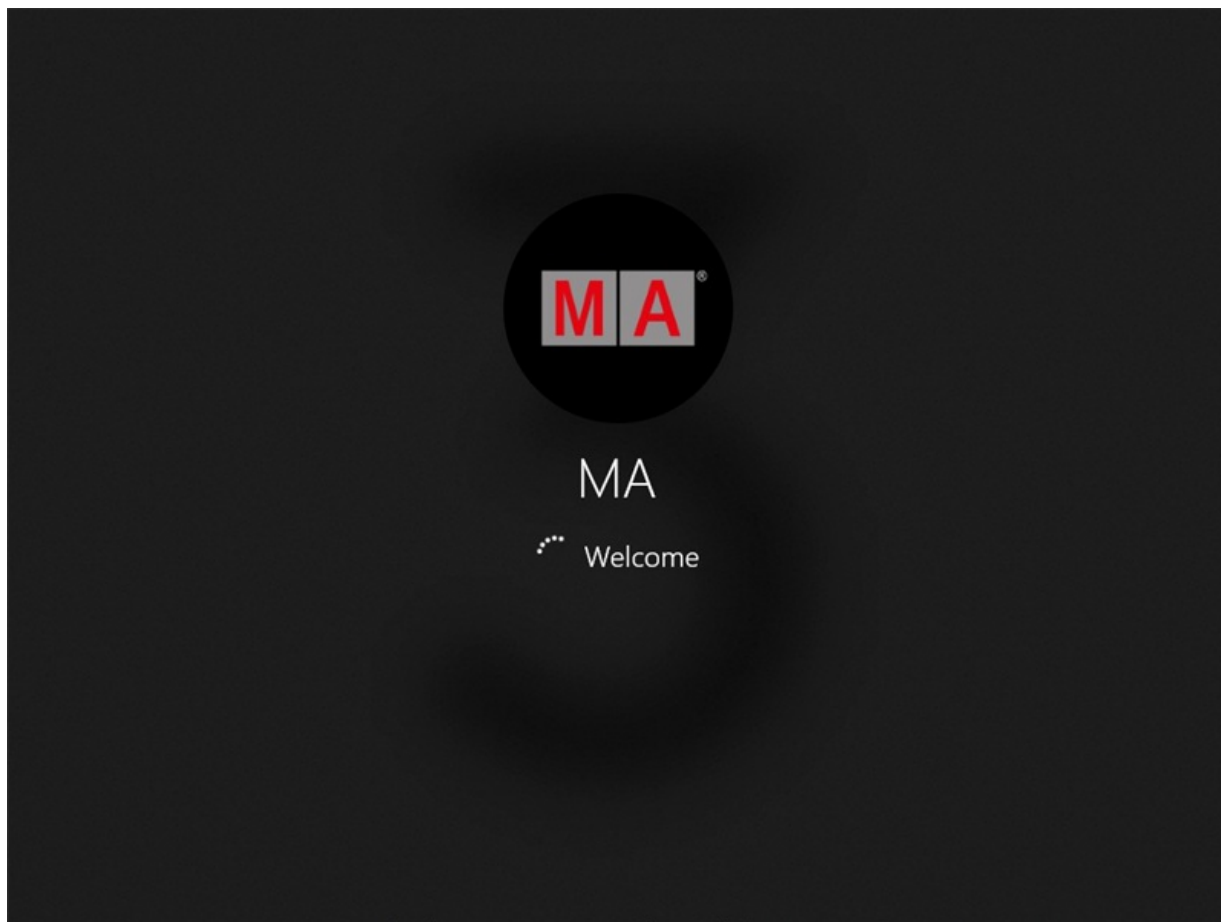
- Recover from a drive: Select Recover.



- Win 10 Licence Agreement: Click Accept.

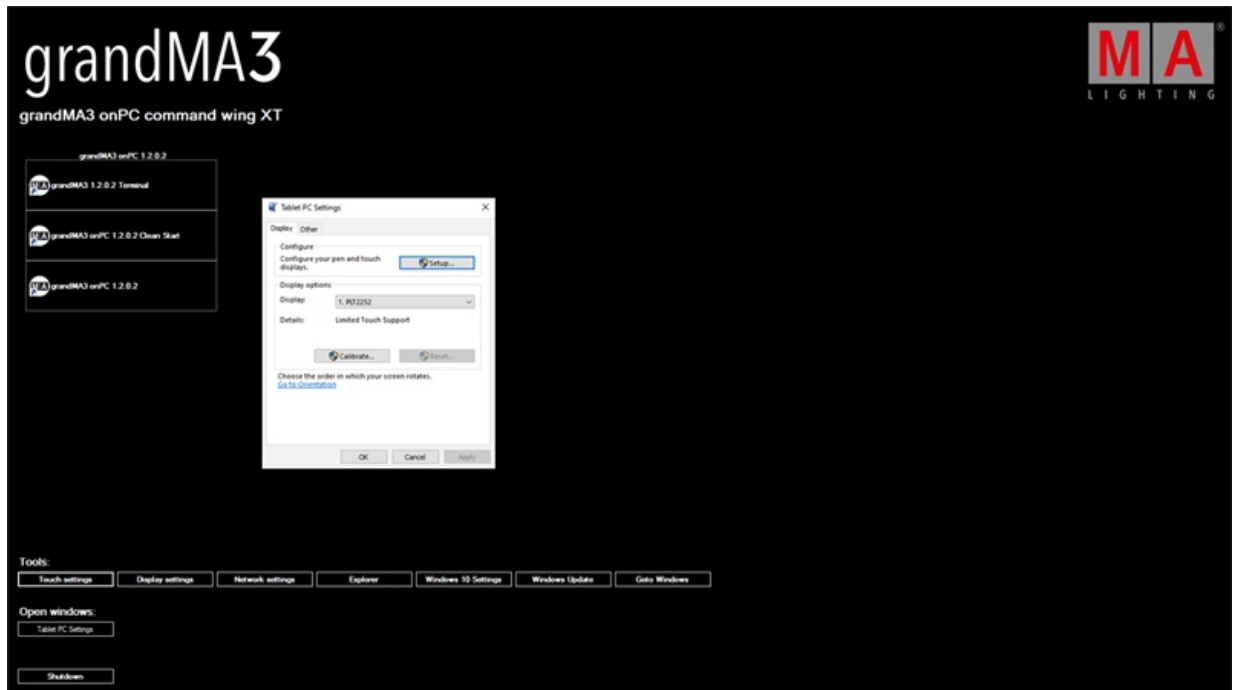
The system restarts several times.



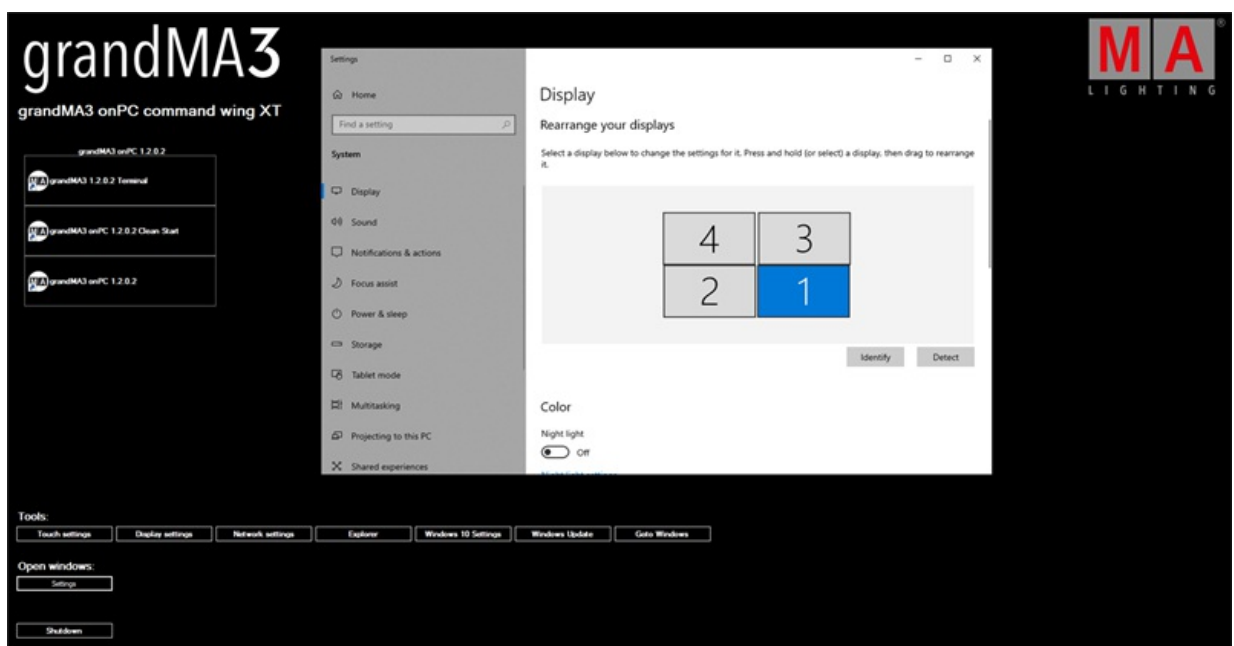


The MA Shell Launcher starts for the first time.

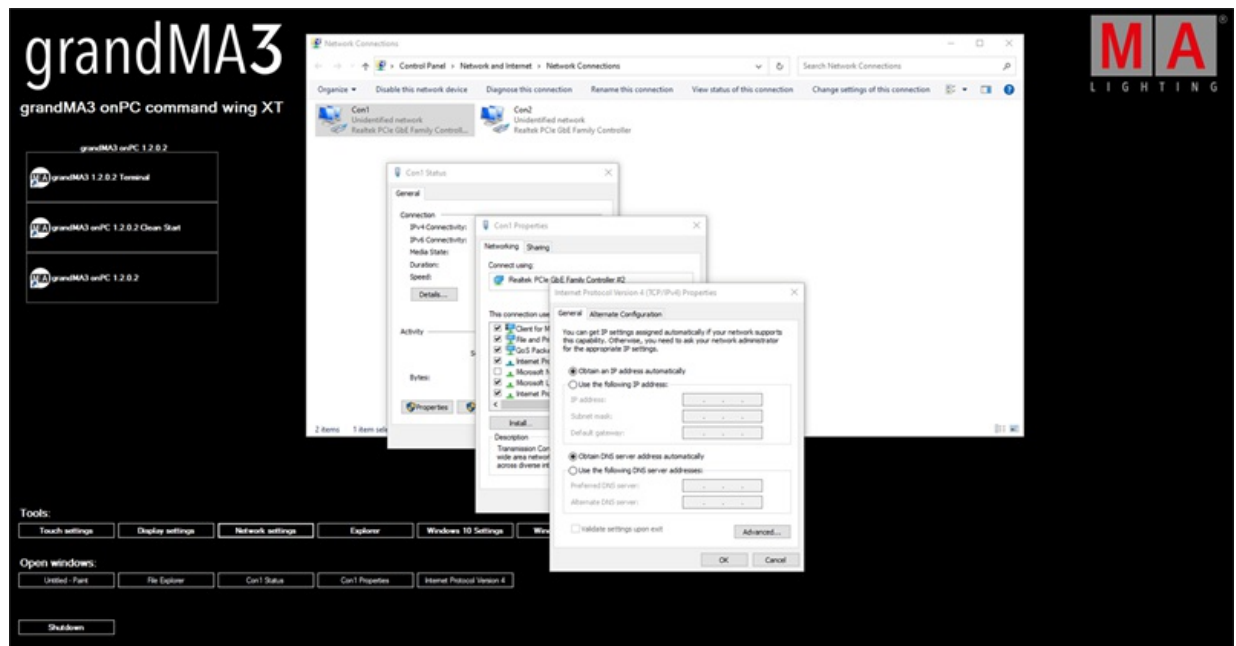
Configure the Settings



- Configure the touch screens.



- Set the desired screen configuration.

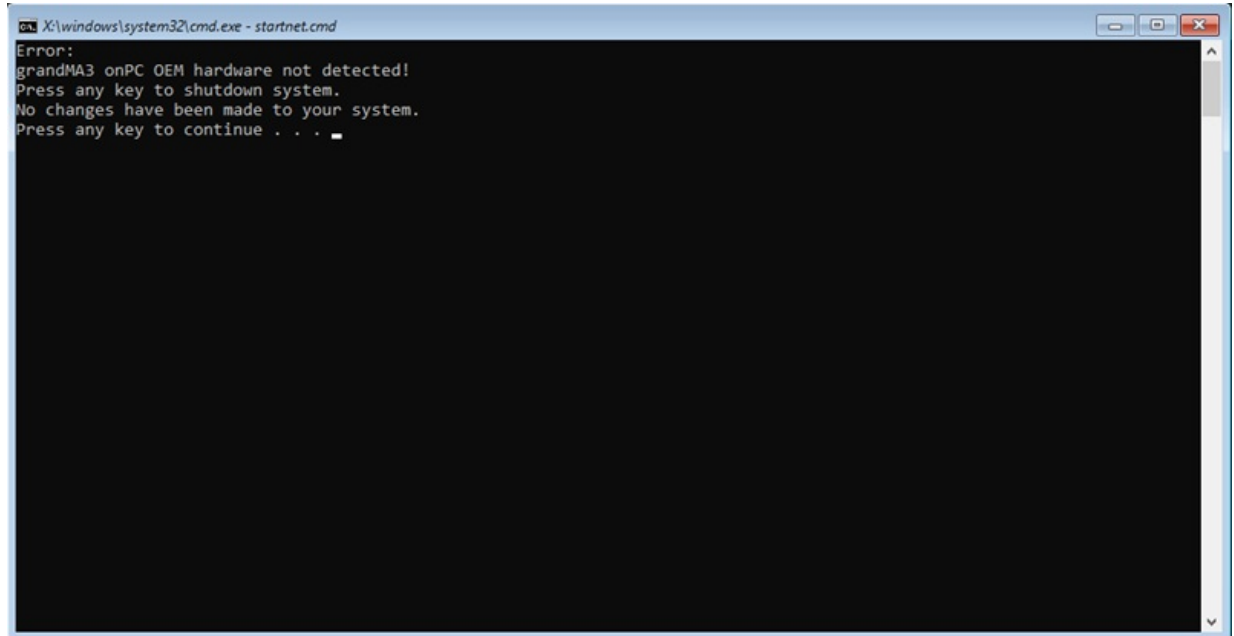


- Set the desired IP address configuration.



Hint:

If a wrong mainboard is installed, the following error message pops up.



47.4. Update grandMA3 viz-key



Important:

The grandMA3 onPC software includes the grandMA3 viz-key software. There is no additional installation required to use the grandMA3 viz-key.

In case that the grandMA3 onPC software is not installed on the computer that is running the visualizer, the separate grandMA3 viz-key software needs to be installed.



Hint:

If the grandMA3 viz-key hardware is connected to a grandMA3 onPC station or visualizer, the running software version is sent to the grandMA3 viz-key hardware.

- Download the latest grandMA3 viz-key software version from www.malighting.com. For more information, see [update the software](#).
The required installer is called grandMA3 Software x.x.x.x for grandMA3 viz-key.


Update Standalone Third-Party Visualizer Using a USB Stick

Requirement: The USB flash drive's data system has to be FAT32.

1. Extract the zip file grandMA3_viz_key_vx.x.x.x.zip and copy the ma folder into the root directory of the USB flash drive.
2. Insert the USB flash drive in the grandMA3 onPC or console USB port.

Follow the next steps from point 3.

Update Standalone Third-Party Visualizer Using the Local Hard Drive

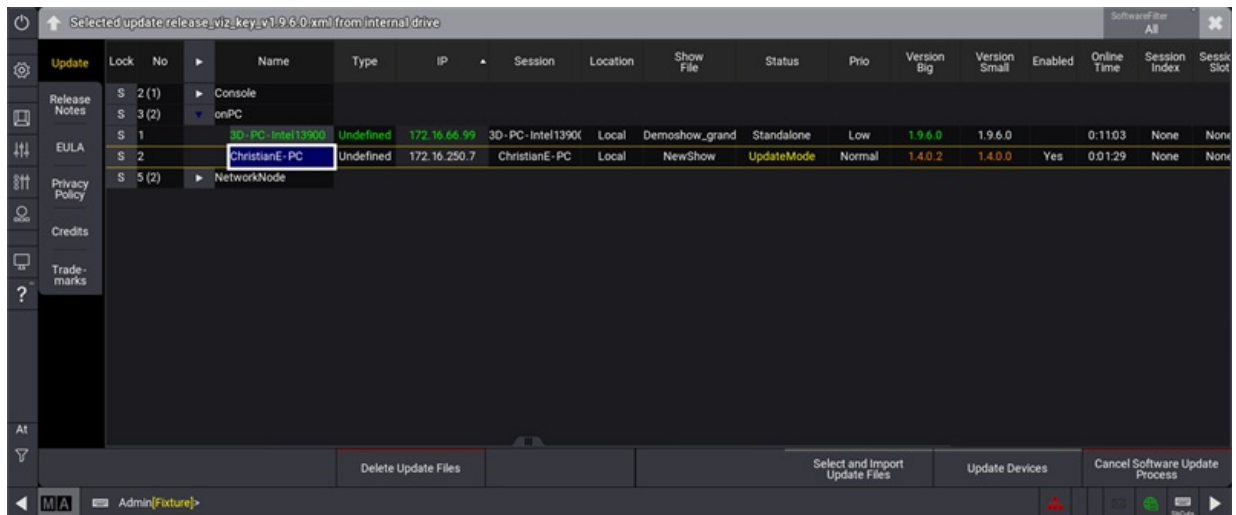
1. Extract the zip file grandMA3_viz_key_vx.x.x.x.zip.
2. For Windows systems, copy the files from the ma folder into the directory C:\ProgramData\MALightingTechnology\installation_packages.
For macOS systems, copy the files from the ma folder into the directory ~/MALightingTechnology/installation_packages. For more information, see [Folder Structure](#).
3. To access **Software Update**, tap .
4. Tap **Settings**.
5. Tap **Software Update**.



Important:

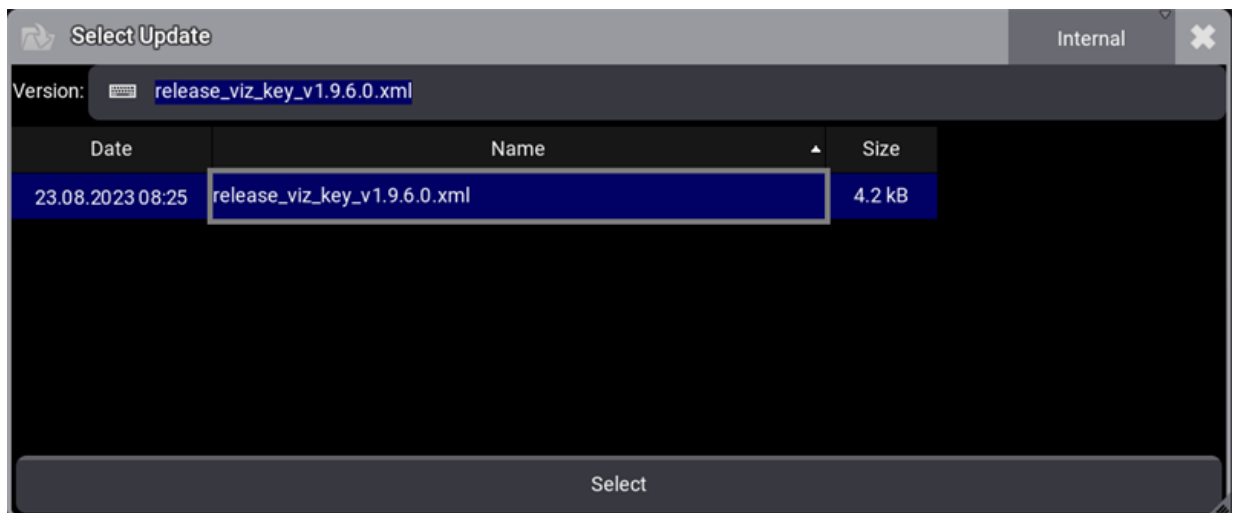
The third-party visualizer must be running in the update mode. For more information, see the manual of the third-party software.

The Software Update window opens:



Software Update Window with selected update file

6. Tap **Select and Import Update Files** .
The Select Update window opens.



Select Update pop-up

8. Select the location that contains the update files (internal or any plugged-in external device). Select the release_viz_key_vx.x.x.x.xml file.
9. Tap **Select** .
10. The pop-up closes and the End User License Agreement (EULA) opens.
Confirm the End User License Agreement (EULA).
The selected update file is displayed at the title bar of the software update window.
11. Select the third-party visualizer with viz-key support. The selected devices turns into bright blue.
12. Tap **Update Devices** .
13. The software update starts copying files.



14. Once the file is transferred, restart the third-party visualizer with the viz-key support software.



Hint:

To learn more on how to connect, visit <https://www.malighting.com/viz-key>.

47.5. Network Update

Use **Software Update** to update one or more stations on a network.



Important:

Make sure that all devices are in the same network (check network adapter).
To learn more about network settings, read the Networking, **Interfaces and IP** topic.


Preliminary Procedures

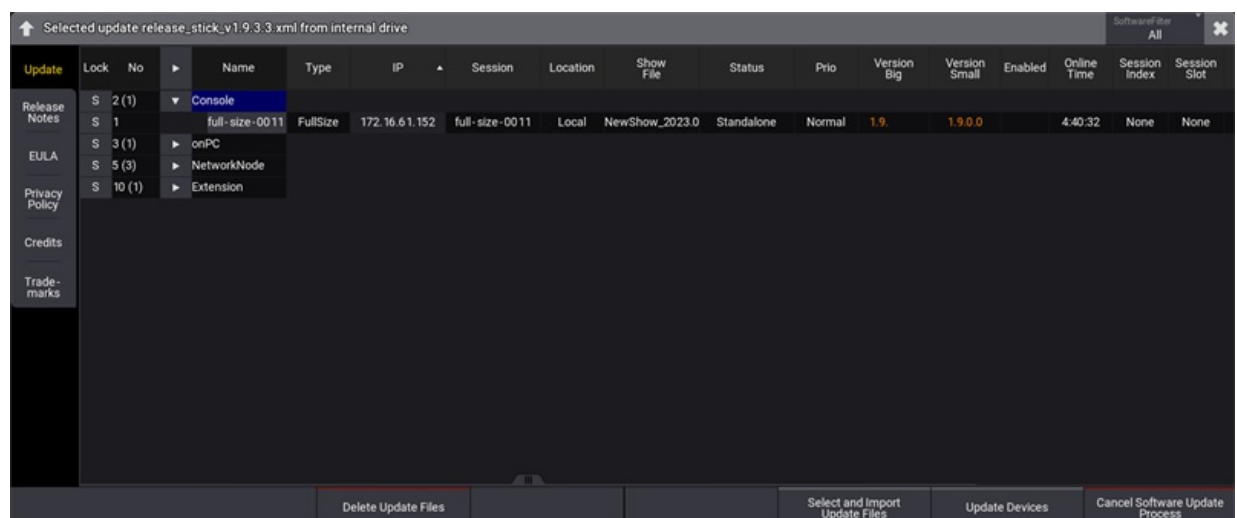
- Connect all grandMA3 devices to a network with an etherCON/RJ45 connector.
- Download the required installation packages from www.malighting.com, **Downloads**.
- Copy all installation packages for grandMA3 software into the root directory of your USB flash drive. For more information, see **Update the Software**.
- Insert the USB flash drive in the device's USB port.
- For Windows systems, copy the zipped files from the ma folder into the directory C:\ProgramData\MALightingTechnology\installation_packages.
- For macOS systems, copy the files from the ma folder into the directory ~/MALightingTechnology/installation_packages.

Procedure

To update the grandMA3 device, follow the onscreen instructions that appear during the update.

To access **Software Update** :

1. Press **Menu** or tap .
2. Tap **Settings** and then tap **Software Update**. The Software Update window opens, as shown in the image below:



Software Update window

3. To select a file, tap **Select and Import Update File**. The Select Update pop-up opens:



Select Update pop-up

4. Make sure the correct update file location is selected (Internal or external device). Select the desired update file.
5. Tap **Select**. The pop-up closes and the End User License Agreement (EULA) opens.
6. Scroll down to read the complete EULA. The button **I agree** in the upper right corner of the pop-up turns white. To close the EULA, tap **I agree**.
7. Select the desired device(s). The selected devices are marked in blue.
8. Tap **Update Devices**. The software update starts copying files.



Hint:

The selected update file will be displayed in the title bar of the software update menu.

- **Cancel Software Update Process**: Tap to cancel the current update process. For more information, see **CancelSoftwareUpdate keyword**.

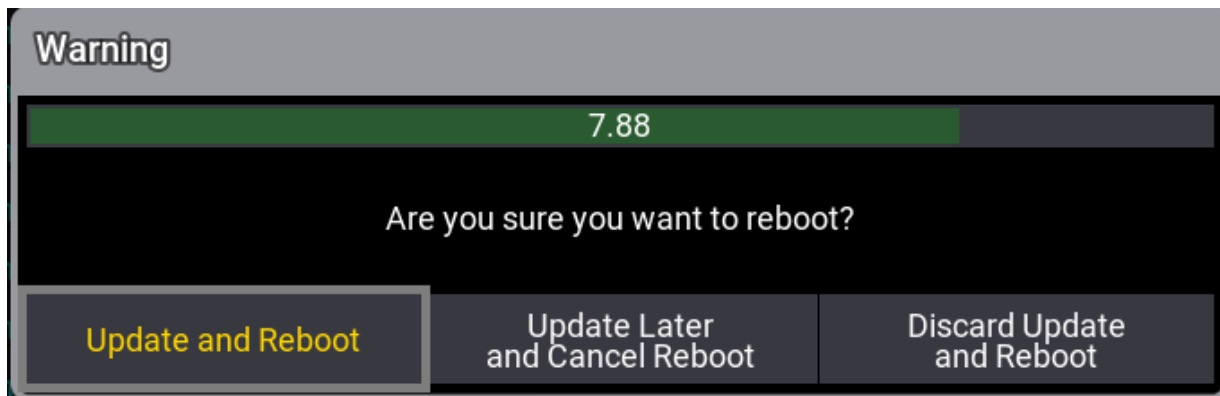


Hint:


It is also possible to update the grandMA3 **onPC**, **consoles**, **xPort Nodes**, **onPC command wing XT**, **processing unit** and **onPC rack-unit** directly.

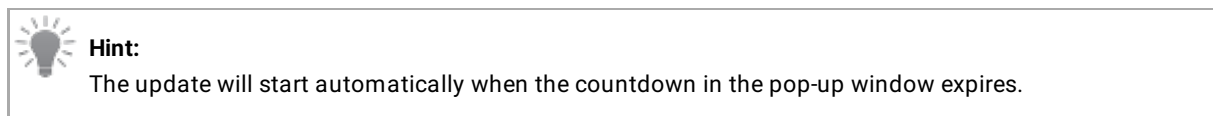
Update Confirmation


After copying the files to the grandMA3 device, a warning pop-up will appear:

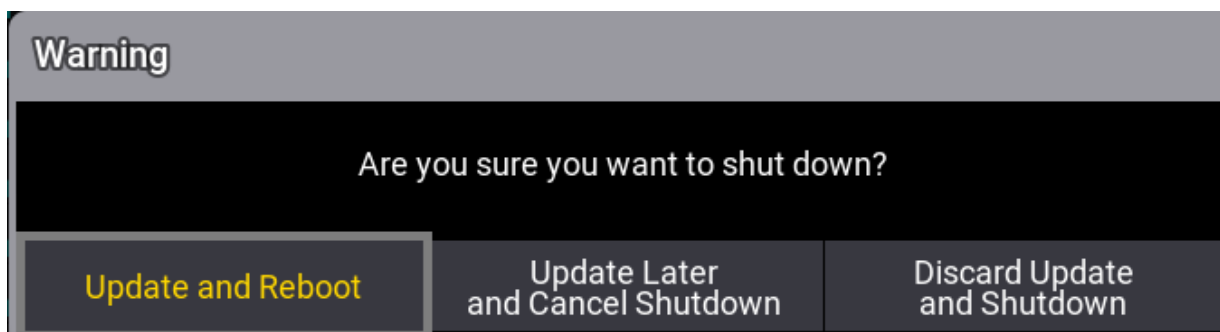


Pop-up in the update process


- **Update and Reboot:** Updates and reboots the device.
- **Update Later and Cancel Reboot:** Delays the update and leaves a red indicator in the control bar .
- **Discard Update and Reboot:** Aborts the update and reboots the device.



Tap  in the Shutdown Menu to install delayed updates. This will open a second warning pop-up.



Shutdown pop-up

- **Update and Reboot:** Updates and reboots the device.
- **Update Later and Cancel Shutdown:** Delays the update and leaves a red indicator in the control bar .
- **Discard Update and Shutdown:** Aborts the update and shuts down the device.


For more information about shutting down, see [Shut Down the System](#).

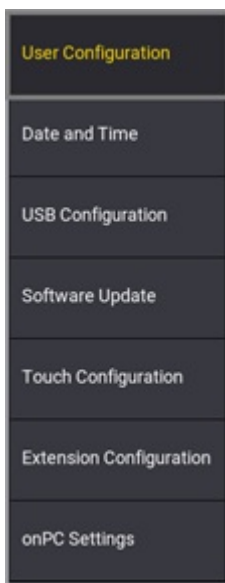


47.6. Delete Update Files

To save storage on the hard drive, we recommend to delete not used update files from the hard drive.

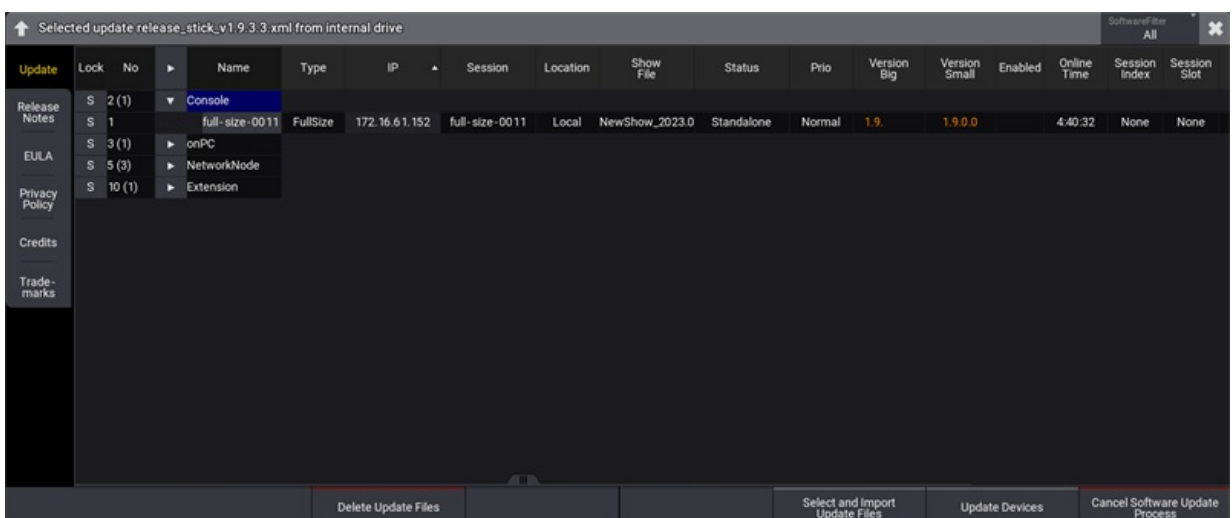
To delete update files from the hard drive:

1. To access the software update window, tap .
 2. Tap **Settings**.
- The settings drop-down menu opens.



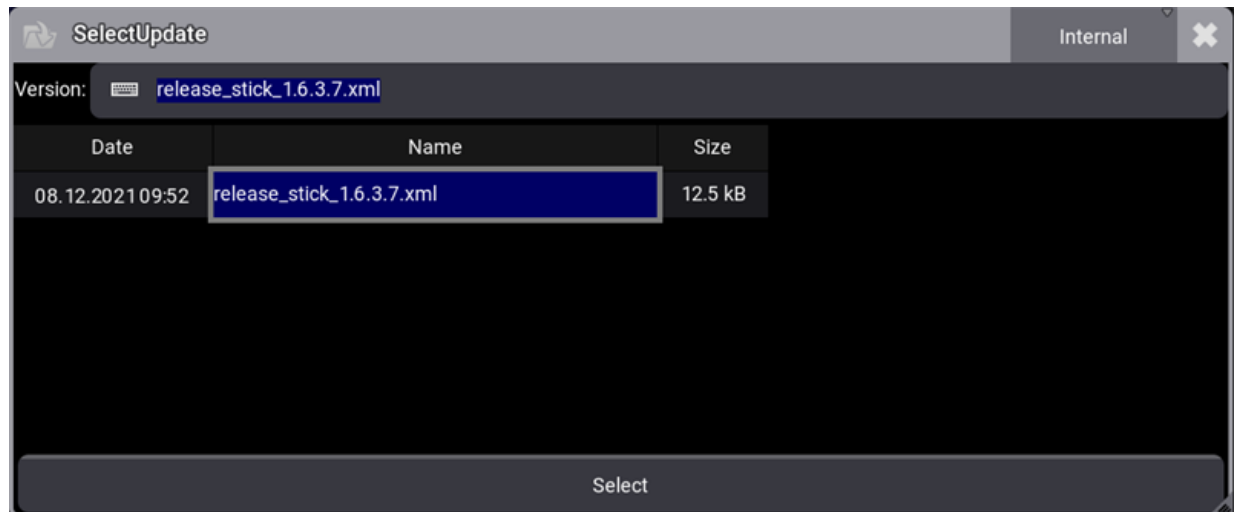
Settings drop-down

3. Tap **Software Update**.
- The Software Update window opens.



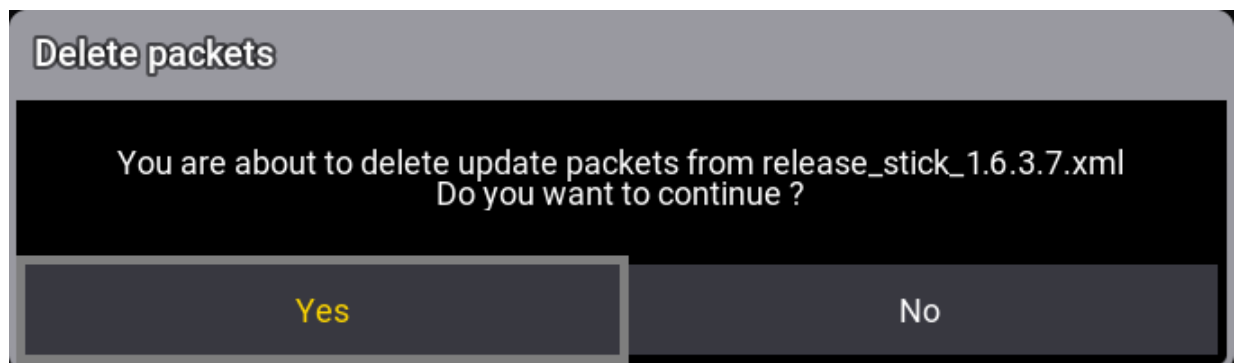
Software update window

4. Tap **Delete Update Files** .
The Select Update pop-up opens.



Select update pop-up

5. Select the to be deleted update file.
6. Tap **Select** .
The Delete packets pop-up opens.



Delete packets pop-up

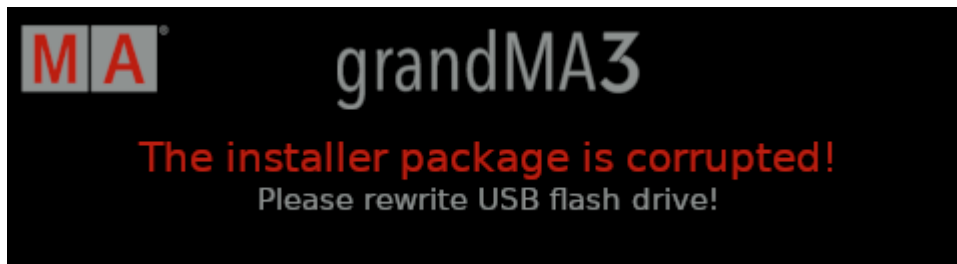
6. Tap **Yes** .

The selected update file is deleted from the hard drive.

47.7. Troubleshooting Update Process

During the update process, different problems may occur.

Corrupted Files on USB Flash Drive



Corrupted installer package

If the installer package is corrupted, try the following:

- Remove the USB flash drive from the grandMA3 device.
- Shutdown the grandMA3 device.
- Format the USB flash drive (FAT32).
- Copy the folders EFI, ma, and the update.scr file into the root directory of your USB flash drive.
- Update the grandMA3 device with the USB flash drive.

If the error message shows up again, try the following:

- Use a different USB flash drive.
- Format the USB flash drive (FAT32).
- Start a new download of the latest software version from www.malighting.com.
- Extract the zip file before copying the folders EFI, ma, and the update.scr file into the root directory of your USB flash drive.

Corrupted Show File

If the grandMA3 device does not stop updating, the show file saved on the grandMA3 device may be corrupted.

If the show file is corrupted, try the following:

- Remove the USB flash drive from the grandMA3 device.
- Reboot the grandMA3 device.
- Save the corrupted show file to another USB flash drive.
- Delete the corrupted show file from the grandMA3 device.
- Update the grandMA3 device with the other USB flash drive.



Individual Hardware Freezes

If the grandMA3 device starts updating, but fails to update one of the hardware sections, try the following:

- Remove the USB flash drive from the grandMA3 device.
- Reboot the grandMA3 device.

If the measures above do not help, try the following:

- Save any show file to another USB flash drive.
- Delete any show file from the grandMA3 device.
- Update the grandMA3 device with the other USB flash drive.



48. Fixture Types

Fixture types are used to visualize and control real-life fixtures on stage.

A fixture type is a footprint of a real fixture, smoke machine, laser, media server, dimmer, or anything to be controlled and visualized with a grandMA3 console.

A fixture type is described with the following parameters:

- modes
- channels
- physical properties

You can find a substantial amount of fixture types and manufacturers in the grandMA3 library.

Fixture types are also available for download from MA Fixture Share and GDTF Share webpages.

Beyond that, it is possible to build a fixture from scratch with its own parameters and attributes using the fixture builder! The grandMA3 software includes its own fixture builder.

Using the GDTF Builder is also an option.

Links

For more information on Fixture Share, see <https://www.malighting.com/training-support/fixture-shares/>.

For more information on GDTF Share, see <https://gdtf-share.com/>.

For more information on GDTF Builder, see <https://fixturebuilder.gdtf-share.com/>



48.1. Import Fixture Types


Fixture Types are used to import fixtures for a later use.


It is possible to import fixture types from these libraries:

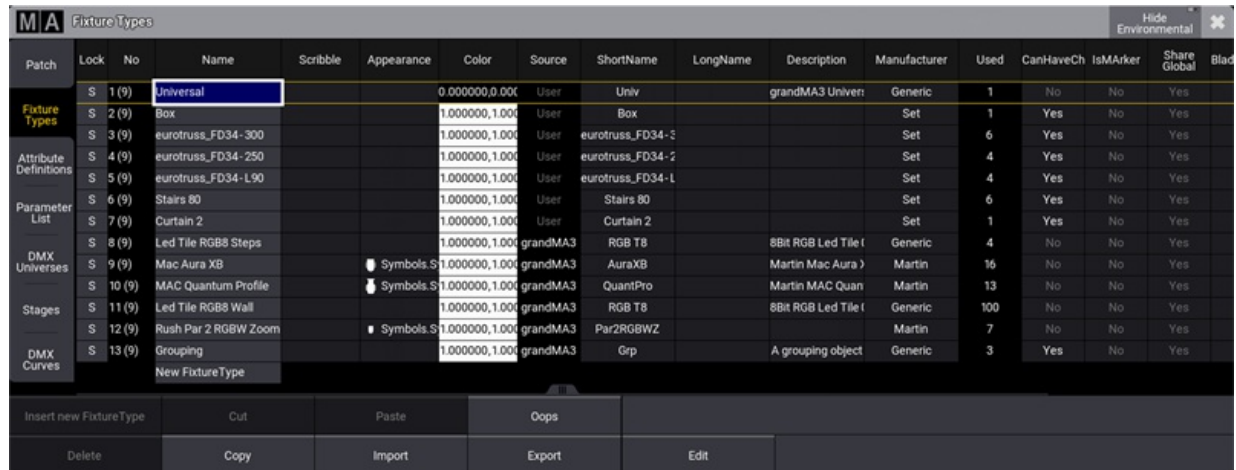
- **grandMA3** fixtures
- Converted **grandMA2** fixtures
- Fixtures using **GDTF** format

Fixture Types Menu

1. To access Patch menu, press **Menu** and then tap **Patch**.
2. Tap **Fixture Types** in the bar on the left of the patch dialog. The Fixture Types menu opens as shown in the image below:



Hint:
When the Patch Menu is opened in a new show, the Insert New Fixtures window will open automatically. Close the window by tapping  in the top-right corner.



Patch	Lock	No	Name	Scribble	Appearance	Color	Source	ShortName	LongName	Description	Manufacturer	Used	CanHaveCh	IsMarker	Share Global	Blad
S	1 (9)		Universal			0.000000,0.000000	User	Univ	grandMA3 Univer		Generic	1	No	No	Yes	
Fixture Types	S	2 (9)	Box			1.000000,1.000000	User	Box			Set	1	Yes	No	Yes	
	S	3 (9)	eurotruss_FD34-300			1.000000,1.000000	User	eurotruss_FD34-3			Set	6	Yes	No	Yes	
Attribute Definitions	S	4 (9)	eurotruss_FD34-250			1.000000,1.000000	User	eurotruss_FD34-2			Set	4	Yes	No	Yes	
	S	5 (9)	eurotruss_FD34-L90			1.000000,1.000000	User	eurotruss_FD34-L			Set	4	Yes	No	Yes	
Parameter List	S	6 (9)	Stairs 80			1.000000,1.000000	User	Stairs 80			Set	6	Yes	No	Yes	
	S	7 (9)	Curtain 2			1.000000,1.000000	User	Curtain 2			Set	1	Yes	No	Yes	
DMX Universes	S	8 (9)	Led Tile RGB8 Steps			1.000000,1.000000	grandMA3	RGB T8	8Bit RGB Led Tile f		Generic	4	No	No	Yes	
	S	9 (9)	Mac Aura XB		Symbols.S	1.000000,1.000000	grandMA3	AuraXB	Martin Mac Aura X		Martin	16	No	No	Yes	
	S	10 (9)	MAC Quantum Profile		Symbols.S	1.000000,1.000000	grandMA3	QuantPro	Martin MAC Quan		Martin	13	No	No	Yes	
Stages	S	11 (9)	Led Tile RGB8 Wall			1.000000,1.000000	grandMA3	RGB T8	8Bit RGB Led Tile f		Generic	100	No	No	Yes	
	S	12 (9)	Rush Par 2 RGBW Zoom			1.000000,1.000000	grandMA3	Par2RGBWZ			Martin	7	No	No	Yes	
DMX Curves	S	13 (9)	Grouping			1.000000,1.000000	grandMA3	Grp	A grouping object		Generic	3	Yes	No	Yes	

Fixture Types menu

The menu is separated in different columns and rows. This is a short explanation of the different columns:

- **Lock, No, Name:**
See **Parameter List** topic.
- **Scribble, Appearance:**
See **Scribble** topic and **Appearances** topic.
- **Color:**
Changes the color value. The selected color value is linked to the Fixture Types column in the Split View of the Patch window. It is also the image color of the used appearance.

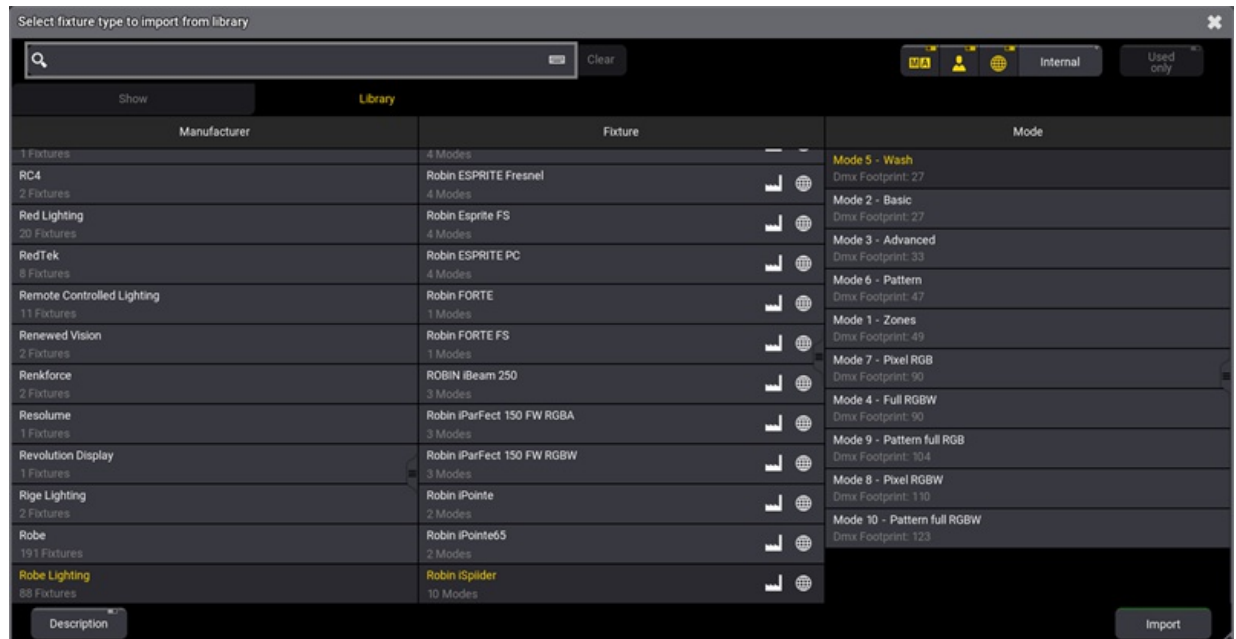


- **Source:**
Displays the origin of the fixture type. The source can be grandMA3, grandMA2 or User. User includes fixture types from GDTF share and from imported grandMA3 fixture types.
- **ShortName:**
Defines the name suggestion of fixtures that the user wants to patch with this fixture type.
- **LongName:**
Displays the full name of the fixture type/product.
- **Description:**
Adds a description to a fixture. It is automatically filled out, if a description is deposited in the library.
- **Manufacturer:**
Shows the name of the fixture type manufacturer.
- **Used:**
Displays how many fixtures in the show use this fixture type.
- **CanHaveChildren:**
This setting defines if a fixture can have children or not. Default is set to No for most fixture types.
- **IsMarker:**
Is set to No by default, but can be toggled between Yes and No. It is set to Yes when using a Marker fixture type. IsMarker cannot be changed in the Live Patch.
- **ShareGlobal:**
When the same fixture type (attributes, their order, and their physical from and physical to values) is used several times within a show file, the patched fixtures of these fixture types can use the global presets that are only stored for one of these fixture types. ShareGlobal is set to Yes by default. ShareGlobal can not be changed in the Live Patch.
- **Blade Mirror:**
Set the fixture type in this column to Yes to mirror the angle of a blade. It is a visual feature in the 3D window.
- **BladesPerPrismBeam:**
Defines if the shaper is located in front of the prism or behind the prism. If set to Yes, the shaper is visually used on each beam generated by the prism (shaper after prism). If set to No, the shaper is used on the whole spot (shaper in front of prism). This is a visual feature in the 3D window.
- **Version:**
Displays the grandMA3 version number the fixture type can be used with. The running grandMA3 version has to be the same or higher than the version number of the fixture type.
- **XYZ:**
XYZ can be enabled for a fixture type that supports this feature. Yes indicates that every mode of the fixture type has XYZ enabled. Partial indicates that at least one mode of the fixture type has XYZ enabled. XYZ is set to No per default. To enable XYZ, see Activating XYZ for Fixture Types.

Select and Import Fixture Types

Requirements: Steps 1 and 2 from the chapter above.

3. To import fixture types from the library, tap **Import**. The pop-up Select fixture type to import opens.



Select Fixture Type to Import From Library pop-up.

4. Select a fixture type and then tap **Import** in the bottom-right corner of the Select fixture type window.

The fixture is imported to the fixture type list and can now be patched.

For a detailed description of Insert new Fixture window, see [Add fixtures to the show](#).



Hint:

The **Show** -tab is only available in the Insert New Fixtures pop-up. The **Used only** button is only available in the **Shows** -tab.



48.1.1. Import GDTF

In grandMA3, it is possible to import a GDTF file (General Device Type Format) that contains the description of a fixture type.



The GDTF file is a zip file, with an .gdtf ending, containing:

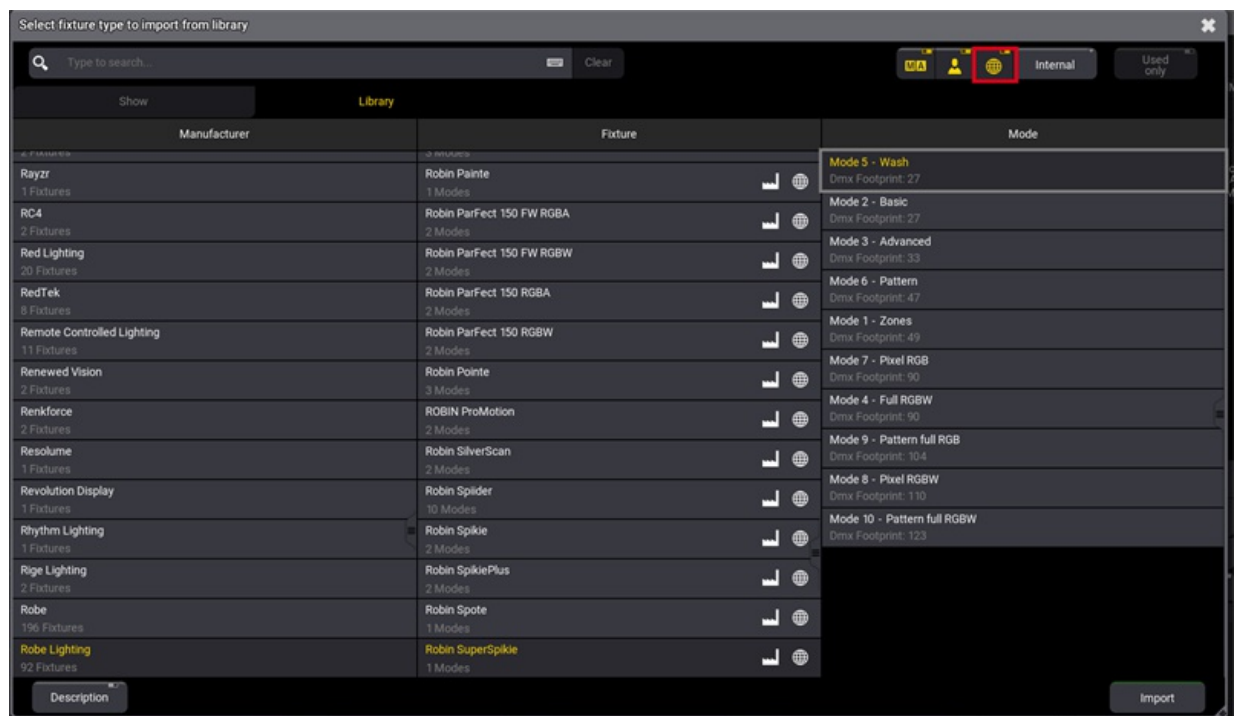
- Description
- Geometry data
- Gobo images

For a detailed description of GDTF, see the DIN specification 15800:2022 (<https://www.beuth.de/de/technische-regel/din-spec-15800/349717520>).

Import GDTF from the Console or Directly from World Server

When there is an active connection to a World Server (for more information see [World server](#)), it is possible to access **grandMA3 fixture share** and **GDTF** libraries directly:

1. Open the patch menu by pressing **Menu**.
2. Tap **Patch**.
3. On the left side of the window, tap **Fixture Types**.
4. Tap **Import** at the bottom of the window. The Select Fixture Type to Import pop-up opens.
5. Enable the share button  to list all fixture types sourced from the GDTF share library. The share button  is also displayed in the image below:





Select Fixture to Import from Library pop-up

5. Select a fixture type and then tap **Import**.

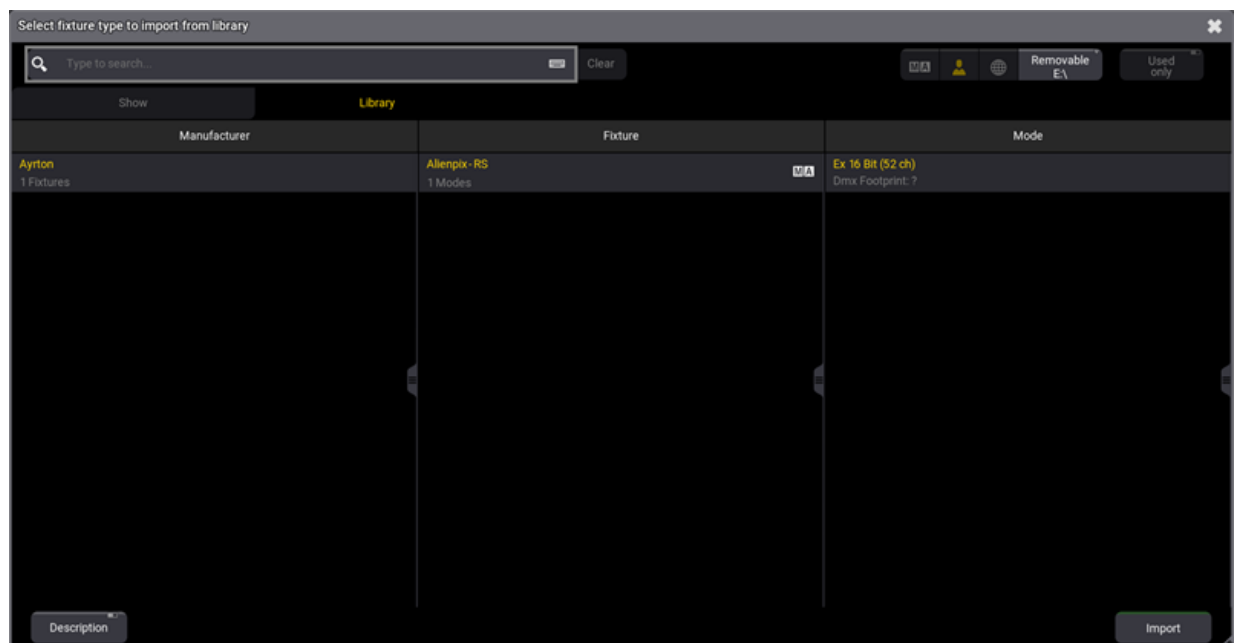
The fixture is imported to the fixture type list and can now be patched.

Import GDTF from a Flash Drive

Fixtures can be exported to a USB flash drive and the fixtures can be imported to a show file even when there is no access to the World Server.

1. Download the GDTF file on www.gdtf-share.com or export it to the USB flash drive (and skip to point 4).
2. The file is located in the download folder.
3. Copy the GDTF file to a USB flash drive. The default path for GDTF fixture types is:
/grandMA3/gma3_library/fixturetypes.
4. Insert the USB flash drive into the console.
5. Open the patch dialog and tap **Fixture Types**.
6. Tap **Import**.

The **Select Fixture Type to Import pop-up** opens:



Select Fixture to Import from Library pop-up with USB drive selected

7. In the upper right corner, select the USB flash drive that contains the GDTF files.
8. Select the desired fixture type.
9. Tap **Import**.

The fixture is imported to the fixture type list and can now be patched.

For more information on how to patch, see [Add Fixtures to the Show](#).

48.1.2. Conflicts in Fixture Types

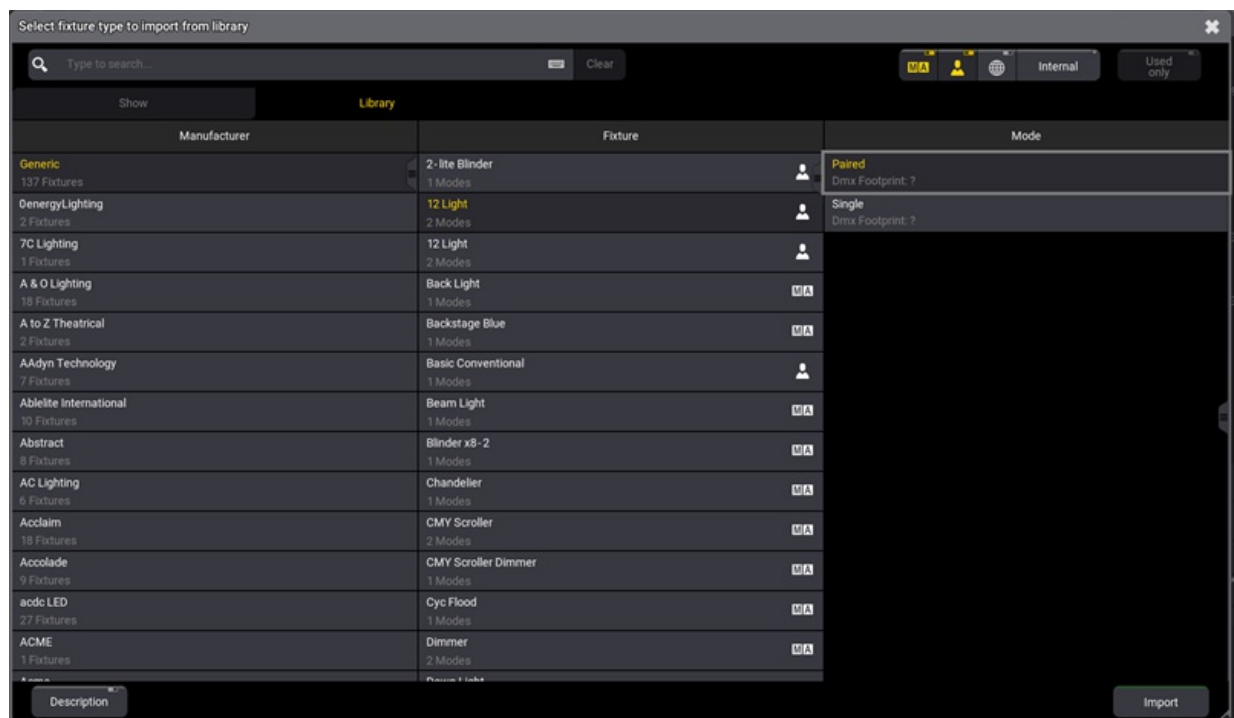
When importing or editing fixture types, it can happen that fixtures are displayed in a specific font color. The specific font color is displaying an existing conflict. The button **Show Conflicts** provides more information about this conflict. A conflict can be, for example, a DMX channel overlap!

Fixture Type with Orange Font Color

- When importing a fixture type from the library, a fixture is shown in an orange font color. This indicates a warning.
- Fixture types with an orange font color are accessible, but may not function properly.

Example:

1. Open the Fixture Types menu, as shown in **Import fixture types**.
2. To select a fixture from the library, tap **Import**.
3. Select **Generic**, **12 Light**, **Paired** from the library as shown in the image below:



A selection of a specific fixture type for a example of a warning

4. Tap **Import**, the select fixture type library closes. The selected fixture is displayed in an orange font color:



Patch	Lock	No	Name	Scribble	Appearance	Color	Source	ShortName	LongName
Fixture Types	S	1 (9)	Universal			0.000000,0.000000	User	Univ	
	S	2 (9)	Box			1.000000,1.000000	User	Box	
	S	3 (9)	eurotruss_FD34-300			1.000000,1.000000	User	eurotruss_FD34-300	
Attribute Definitions	S	4 (9)	eurotruss_FD34-250			1.000000,1.000000	User	eurotruss_FD34-250	
	S	5 (9)	eurotruss_FD34-L90			1.000000,1.000000	User	eurotruss_FD34-L90	
Parameter List	S	6 (9)	Stairs 80			1.000000,1.000000	User	Stairs 80	
	S	7 (9)	Curtain 2			1.000000,1.000000	User	Curtain 2	
DMX Universes	S	8 (9)	Led Tile RGB8 Steps			1.000000,1.000000	grandMA3	RGB T8	8Bit RGB
	S	9 (9)	Mac Aura XB	Symbols S		1.000000,1.000000	grandMA3	AuraXB	
Stages	S	10 (9)	MAC Quantum Profile	Symbols S		1.000000,1.000000	grandMA3	QuantPro	
	S	11 (9)	Led Tile RGB8 Wall			1.000000,1.000000	grandMA3	RGB T8	8Bit RGB
DMX Curves	S	12 (9)	Rush Par 2 RGBW Zoom	Symbols S		1.000000,1.000000	grandMA3	Par2RGBWZ	
	S	13 (9)	Grouping			1.000000,1.000000	grandMA3	Grp	
	S	14 (9)	12 Light			1.000000,1.000000	User	12 Light	12 Light

The example is displayed in the Fixture Types menu.

Fixture Type with Red Font Color

- Fixture types that have errors are displayed with a red font color in the Fixture Types-tab of the Patch menu.
- When a fixture type has errors, the corresponding fixtures are not displayed in the rest of the software, because they cannot function correctly.

Show Conflicts

Next to the font colors, it is also possible to get a more detailed description of the error / warning.

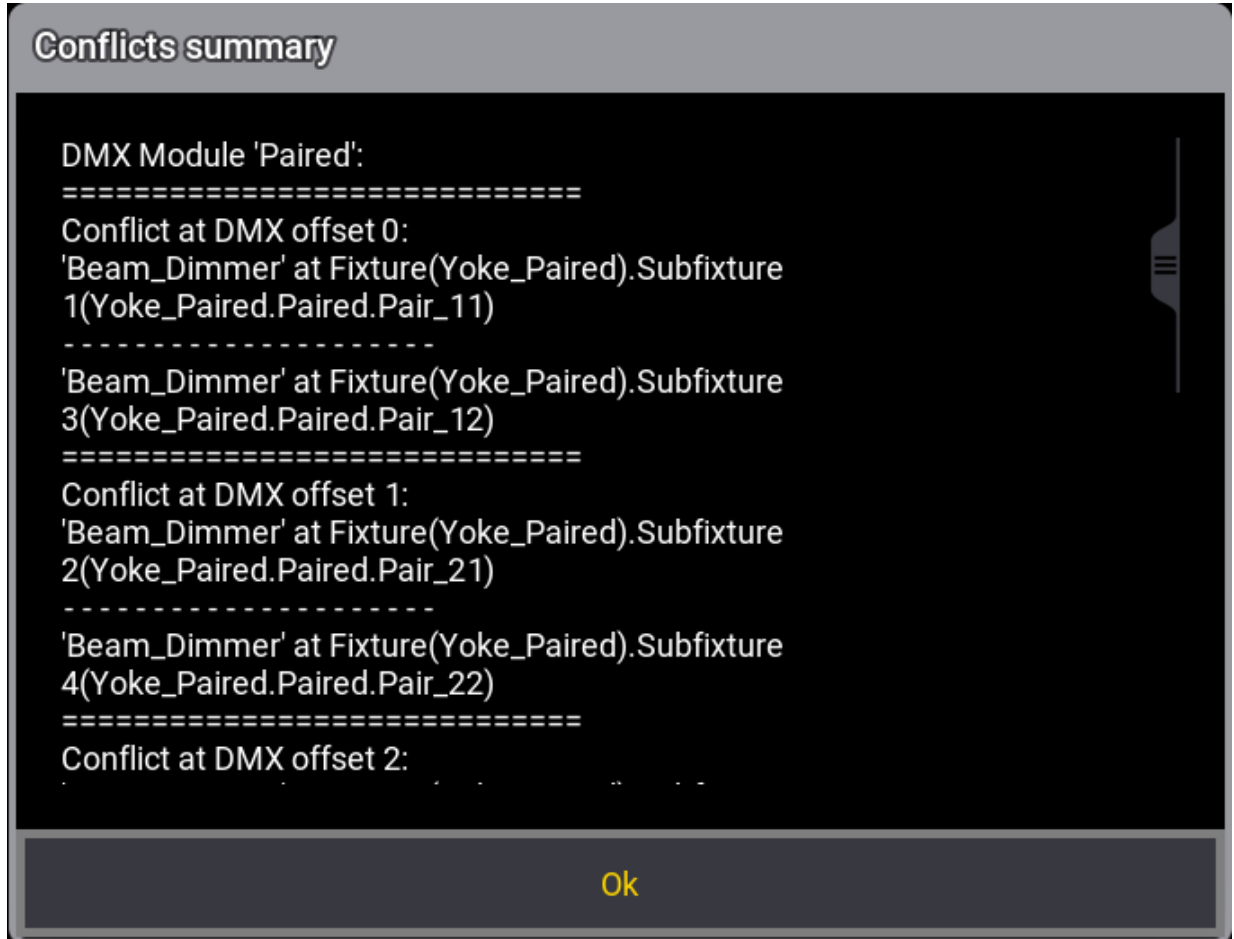
To get a more detailed summary of the error / warning:

1. Select the colored fixture type with a red or orange font color. To do this, go to the **Fixture Types**-tab in the Patch menu.
2. To open the editor, tap **Edit**. The Edit FixtureType Editor opens.
3. A button called **Show Conflicts (x)** is colored in a red font. The number in the parentheses shows the number of conflicts. See image below:

Insert new DMXMode	Cut	Paste	Oops	DMXReadout Percent	Select up	New object line
Delete	Copy	Import	Export	Show Conflicts (8)	Select down	Merge children

Show Conflicts button is only visible when an error / warning is emerging.

4. To open the Conflicts summary pop-up, tap **Show Conflicts**. The Conflicts summary pop-up opens:



Conflicts pop-up.

48.2. Build Fixture Types

grandMA3 allows building your own fixture types. Therefore, the grandMA3 software benefits of using a Fixture Type Editor. In this editor, it is possible to set up a custom made fixture type using different inputs. For example:

- DMXModes
- Geometries
- PhysicalDescriptions

In the following topics it is displayed how to use the Fixture Type Editor. There is also a simple example for how a standard fixture type can look like.



Important:

Creating fixture types can be necessary to project real-live fixtures that have not been build yet.



Important:

Instead of creating a new fixture type it can be easier and faster to take an existing fixture type and adjust only the differences!




Hint:

It is also possible to build fixtures using the GDTF Builder and import them to grandMA3. For more information see [**Import GDTF**](#).



48.2.1. Insert Fixture Types

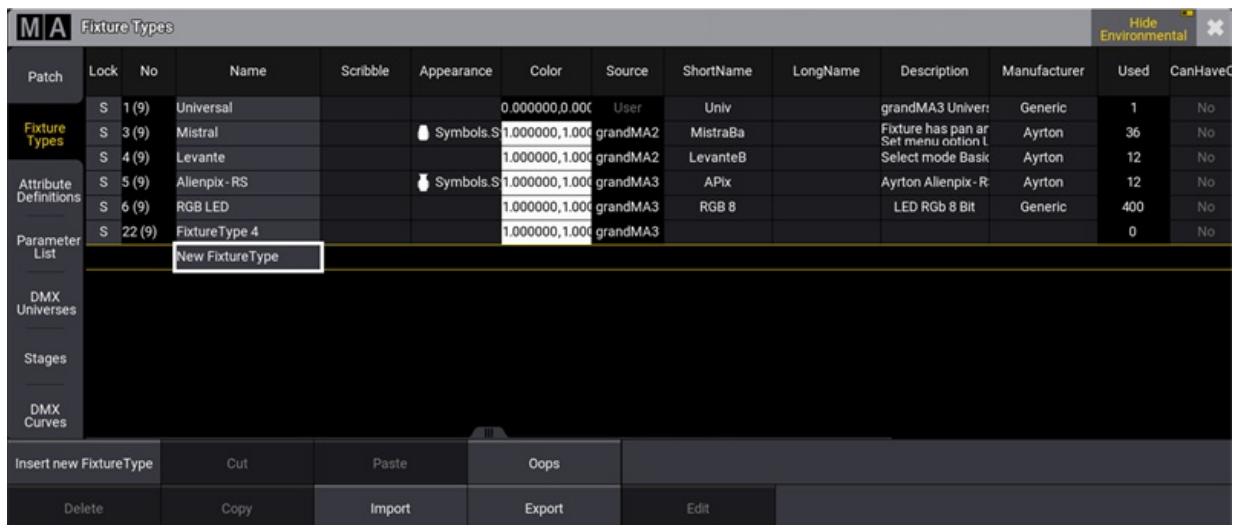
First of all, a fixture type has to be inserted.

 **Important:**
 Always make sure to save the settings you made; otherwise, they will get lost.
 How to save settings in Fixture Types:

1. Leave the Patch.
2. The **pop-up Leaving the patch** appears.
3. Tap **Ok**.
4. Save the show file.

For more information, see [SaveShow Keyword](#).

1. Open the Fixture Types-tab in the Patch menu. To exit Fixture Types-tab, see [Import fixture types](#) topic.
2. In the **Name** column, tap **New Fixture Type**.
3. Tap **Insert New Fixture Type** in the bottom left corner. A new fixture type row is inserted into the sheet. In the example below, FixtureType 4.



Patch	Lock	No	Name	Scribble	Appearance	Color	Source	ShortName	LongName	Description	Manufacturer	Used	CanHaveC
	S	1 (9)	Universal			0.000000,0.000	User	Univ		grandMA3 Univer:	Generic	1	No
Fixture Types	S	3 (9)	Mistral		Symbols.S	1.000000,1.000	grandMA2	MistraBa		Fixture has pan ar Set menu option L	Ayrton	36	No
	S	4 (9)	Levante			1.000000,1.000	grandMA2	LevanteB		Select mode Bask	Ayrton	12	No
Attribute Definitions	S	5 (9)	Alienpix-RS		Symbols.S	1.000000,1.000	grandMA3	APix		Ayrton Alienpix-R	Ayrton	12	No
	S	6 (9)	RGB LED			1.000000,1.000	grandMA3	RGB 8		LED RGB 8 Bit	Generic	400	No
Parameter List	S	22 (9)	FixtureType 4			1.000000,1.000	grandMA3					0	No
			New FixtureType										

4. To edit or enter the basic data of the fixture type, fill out the following cells by right-clicking or tap and hold the cell:

- Name
- Scribble
- Appearance
- ShortName
- Description
- Manufacturer



Hint:

To see a detailed description of every cell in the Fixture Types-tab, see [Insert fixture types](#) topic.

48.2.2. Insert DMX Modes

The second step is to open up the Fixture Type Editor and inserting values to the different columns and cells. Notice that the Editor will automatically jump to the DMX Modes column after opening it.

Requirement:

- **Insert fixture type** first.

Description of DMX Modes:

- A DMX Mode consists of one or several DMX channels.
- The DMX Mode is the parent, and the DMX Channel is its child.



Important:

The build of a fixture type is based on the hierarchic structure of parent-child. A parent comes first, and the child follows its parent. There may be several children.



Hint:

Many manufacturers provide DMX charts that define channels. It is possible to download the DMX charts for a specific fixture type from the manufacturer website in most cases. Use a DMX chart to enter the channels in the fixture type table.


Preparation

- Example for a basic moving head

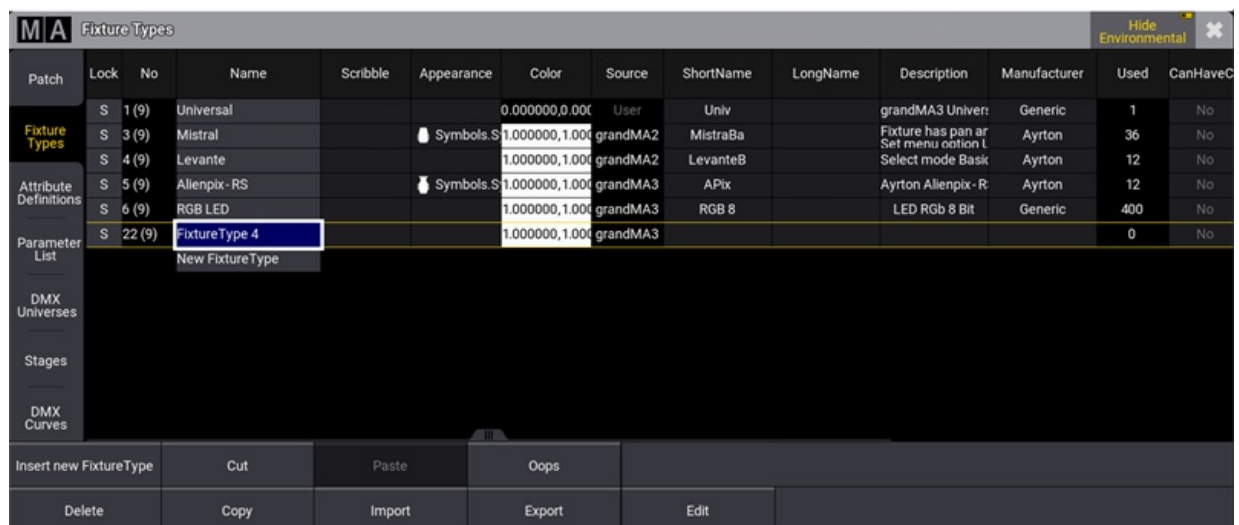


This basic moving head is based on the following DMX chart, showing a relative patch address and the related function:

Relative Patch Address	Function
1	Pan (8 bit) Pan movement by 540°
2	Pan Fine (16 bit)
3	Tilt (8 bit) Tilt movement by 270°
4	Tilt Fine (16 bit)
5	Dimmer
6	Red
7	Green
8	Blue

 **Important:**
 -To back up data during or after the build of fixture types, leave the **Fixture Type Editor** and tap **Export** in the **Fixture Types** menu.
 An xml. file is exported. It is possible to import this file into the show file.
 -Or save the show file. Leave the **Patch** menu and save all changes.

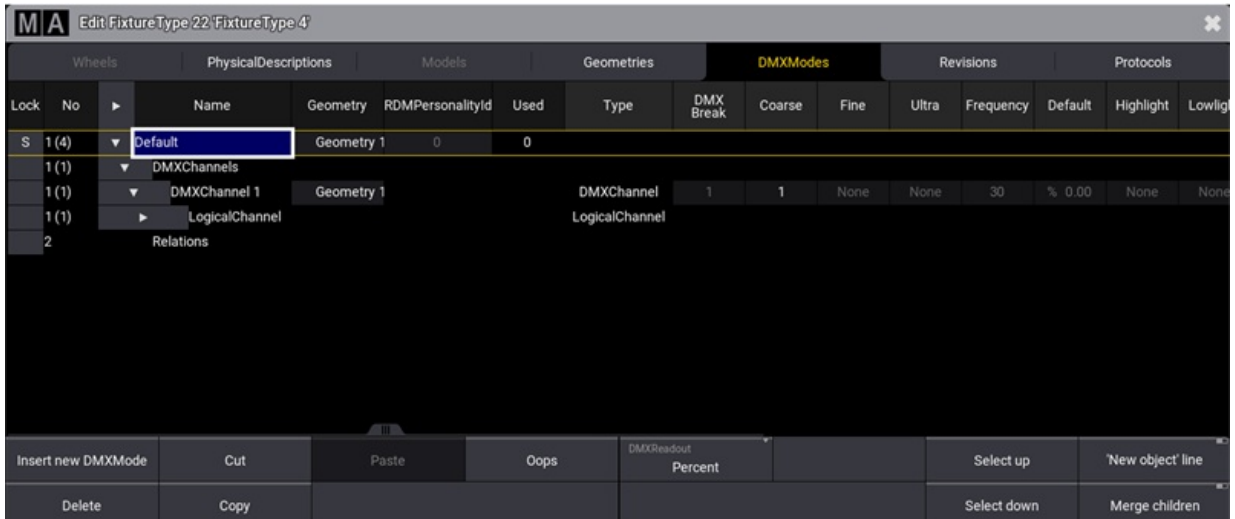
1. Select the column FixtureType 4 in the Fixture Types-tab, as shown in the image below:



Selecting Fixture Type 4 which was inserted in the Insert Fixture Types topic.



2. To open the Fixture Type Editor, tap **Edit**. The window Edit FixtureType opens, as shown in the image below:



The editor for building a fixture.

3. To change the name of the cell, tap and hold or right-click the cell labeled **Default**. A text editor opens.
 4. Rename Default to Mode 1 as shown in the image below:



This is the text editor

Example for a Basic Moving Head

Before starting with an example of a basic moving head, take a look at the most important columns and the characteristics of it:




- **Coarse, Fine, Ultra, Default, Highlight:**
 See [Parameter List](#) topic.
- **Attribute:**
 This displays the functionality for the DMX-Channel.
- **Master:**
 This defines if the value of the channel is affected by a group master or the grand master.



- **Physical From:**
This displays the physical value of a fixture. Physical From can not be changed in the Live Patch.
- **Physical To:**
This displays the physical value of a fixture. Physical To can not be changed in the Live Patch.
- **DMX From:**
Sets the DMX value from the lower end of the DMX range on the channel function. DMX From can not be changed in the Live Patch.
- **DMX To:**
DMX To in the channel function is automatically calculated depending on the DMX From value of the following channel function or the end of the DMX channel.

Enter DMXChannel 1 – Pan

To enter the first channel:

1. Expand Mode 1:
-Tap .
2. In the row DMXChannel 1, enter:
-**Coarse:** 1
-**Fine:** 2
-**Default:** 50 %
-**Highlight:** none
3. Make sure the **Merge children** button is disabled.
4. Expand DMXChannel 1 for further entries in LogicalChannel:
-Tap  in the row DMXChannel 1.
5. In the row LogicalChannel select:
-**Attribute:** Pan
6. The DMXChannel 1 is renamed to Geometry 1_Pan.
-**Master:** none
7. Expand LogicalChannel, now called Pan, for further entries in ChannelFunction:
-Tap  in the row LogicalChannel.
8. In the row ChannelFunction select:
-**Attribute:** Pan
-**Physical From:** -270
-**Physical To:** 270



Hint:

As soon as you select an attribute, the DMX Channel is renamed after the attribute – **Geometry_Attribute.**

DMXChannel1 is entered.



Enter DMXChannel 2 – Tilt

To enter the second channel:

1. Tap Geometry 1_Pan.
2. To enable NewDMXChannel, tap 'New object' line button.
3. Tap New DMXChannel and tap **Insert new DMXChannel** .
New DMXChannel is renamed to DMXChannel 2.
4. To enter the second channel, enter in the row DMXChannel 2:
 - Coarse**: 3
 - Fine**: 4
 - Default**: 50 %
 - Highlight**: none-Open LogicalChannel. For more information, see [step 3](#) in the type LogicalChannel.
5. -**Attribute**: Tilt
6. DMXChannel 2 is renamed after the attribute.
 - Master**: none
7. Open ChannelFunction. For more information, see [step 5](#) in the type ChannelFunction.
 - Attribute**: Tilt
 - Physical From**: -135
 - Physical To**: 135

DMXChannel 2 is entered.

Enter DMXChannel 3 – Dimmer

To enter the third channel:

1. Start with steps **1 to 3**, as described in DMXChannel 2.
2. To enter the third channel, enter in the row DMXChannel 3:
3. **-Coarse:** 5
-Default: 0 %
-Highlight: 100 %
-Open LogicalChannel. For more information see **step 3** in the type LogicalChannel.
4. **-Attribute:** Dimmer
5. DMXChannel 3 is renamed after the attribute.
6. To control the value of the attribute using the Grand Master, tap in the column **Master**.
-The **pop-up Select Master** opens



1. -Tap Grand.
2. Open ChannelFunction. For more information, see **step 5** in the type ChannelFunction.
-Attribute: Dimmer
-Physical From: 0
-Physical To: 1

DMXChannel 3 is entered.



Enter DMXChannel 4 – Red

1. Start with steps **1 to 3**, as described in DMXChannel 2.
2. To enter the fourth channel, enter in the row DMXChannel 4:
3. **-Coarse:** 6
 - Default:** 100 %
 - Highlight:** 100 %
 - Open LogicalChannel. For more information, see **step 3** in the type LogicalChannel.
4. **-Attribute:** ColorRGB_R
5. DMXChannel 4 is renamed after the attribute.
6. **Master:** none
7. Open ChannelFunction. For more information see **step 5** in the type ChannelFunction.
 - Attribute:** ColorRGB_R
 - Physical From:** 0
 - Physical To:** 1

DMXChannel 4 is entered.



Enter DMXChannel 5 – Green

1. Start with steps **1 to 3**, as described in DMXChannel 2.
2. To enter the fifth channel, enter in the row DMXChannel 5:
3. **-Coarse:** 7
 - Default:** 100 %
 - Highlight:** 100 %
 - Open LogicalChannel. For more information, see **step 3** in the type LogicalChannel.
4. **-Attribute:** ColorRGB_G
5. DMXChannel 5 is renamed after the attribute.
6. **Master:** none
7. Open ChannelFunction. For more information see **step 5** in the type ChannelFunction.
 - Attribute:** ColorRGB_G
 - Physical From:** 0
 - Physical To:** 1

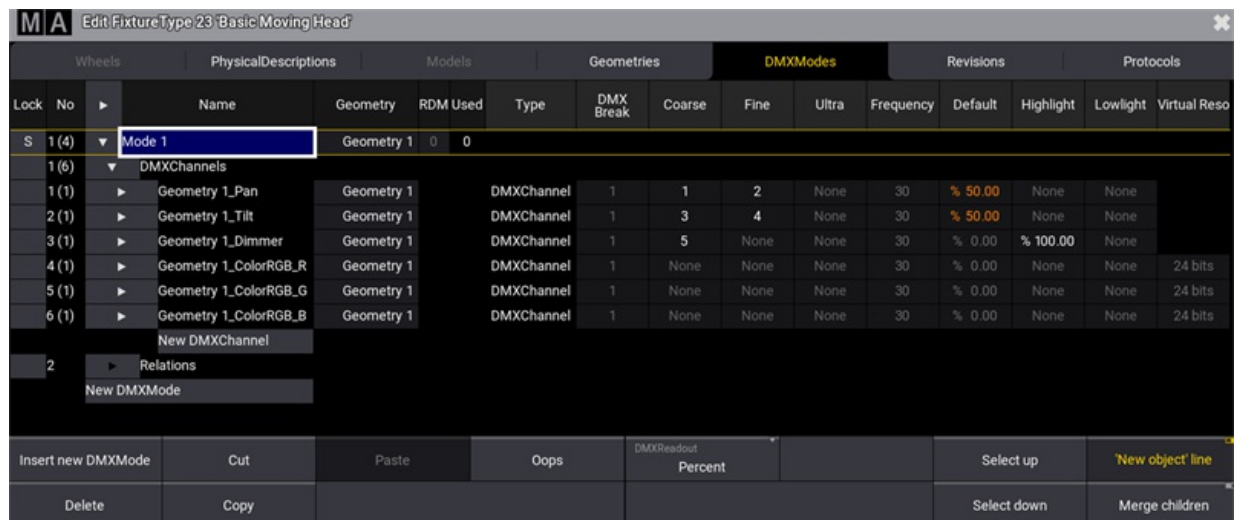
DMXChannel 5 is entered.



Enter DMXChannel 6 – Blue

1. Start with steps **1 to 3**, as described in DMXChannel 2.
2. To enter the sixth channel, enter in the row DMXChannel 6:
3. **-Coarse:** 8
 -Default: 100 %
 -Highlight: 100 %
 -Open LogicalChannel. For more information, see **step 3** in the type LogicalChannel.
4. **-Attribute:** ColorRGB_B
5. DMXChannel 6 is renamed after the attribute.
6. **Master:** none
7. Open ChannelFunction. For more information, see **step 5** in the type ChannelFunction.
 -Attribute: ColorRGB_B
 -Physical From: 0
 -Physical To: 1

DMXChannel 6 is entered, and DMX Mode 1 is inserted.



48.2.3. Insert Geometries

Geometry is the physical description of parts of the device.

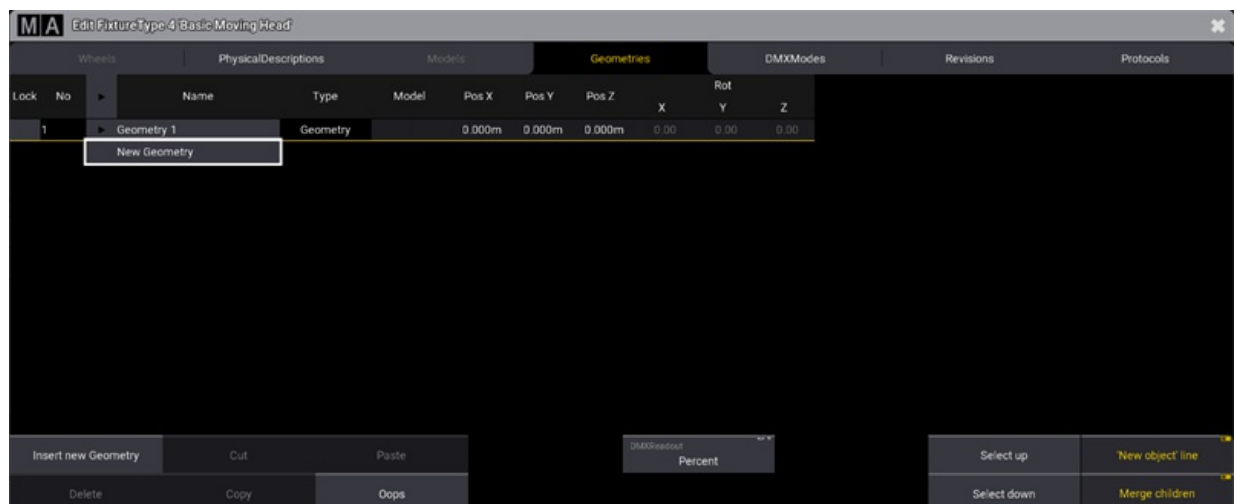
Requirement:

- **Insert DMX Modes.**

Geometries are built upon DMX Modes.

1. Tap the tab **Geometries**.

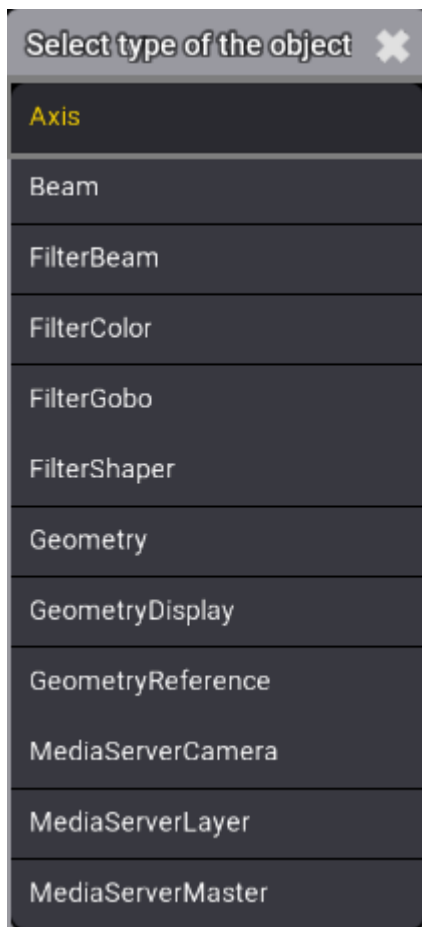
The window Geometries opens.



Open Geometries

2. Select Geometry 1 and rename it Base.
3. Tap to enable 'New Object' line .
4. Open Base and select the Base's child New Geometry.
5. Tap **Insert new Geometry** .

The pop-up Select type of the object opens.



Select type

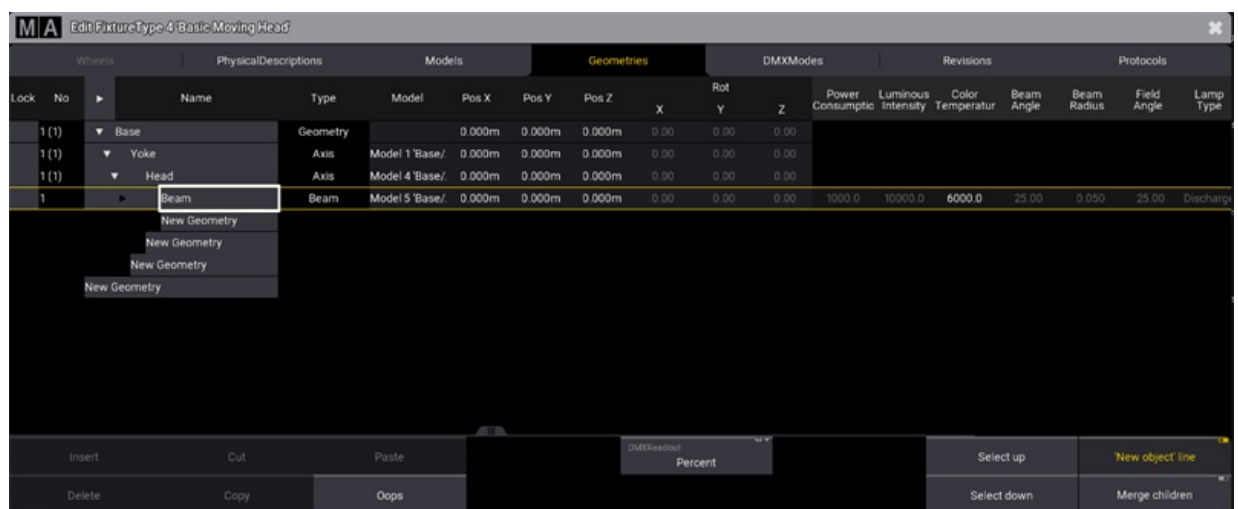
- Select **Axis**.
- Axis is displayed in the column Type.
 6. In the column Name, rename Axis 1 to Yoke.
- Press **Edit** and tap Axis 1, or press and hold Axis 1.
- The virtual keyboard opens.
- Enter Yoke.
 7. Expand the cell Yoke.
- Yoke's child New Geometry opens.

- Repeat steps **5 to 6** and rename Axis 1 to Head.

9. Expand the cell Head.

- Head's child New Geometry opens.
- Repeat steps **5 to 6** selecting Beam.
- Rename Beam 1 to Beam.

Geometries are inserted, as shown in the image below:



Inserted geometry and its children



48.2.4. Insert Models

Each geometry has a separate description of the model.

A mesh is a .3ds file (3D image format used by Autodesk 3D Studio) or a .gltf/.glb file (open standard 3D image format).

Meshes are stored in the Meshes pool after they have been imported to the show.

- For more information about importing meshes, see [Meshes](#).



Important:

After they have been imported, meshes must be linked to a model of a fixture type.



Hint:

Meshes are automatically added to the mesh pool after importing an MVR file or when a fixture type is imported to the show. For more information, see [MVR](#).

Requirement:

- [Insert Geometries](#).

Models are built upon Geometries.

1. Tap the tab **Models**.

The menu Models opens.

Wheels		PhysicalDescriptions		Models			Geometries		DMXModes	Revisions	Protocols
Lock	No	Name	Length	Width	Height	Dimensions follow Ratio	CastShadow	Mesh			
	1	Yoke	0.400m	0.100m	0.300m	No	Yes	5 [yoke_3ds]			
	2	Head	0.250m	0.250m	0.300m	No	Yes	6 [head_3ds]			
	3	Beam	0.080m	0.080m	0.010m	No	Yes	3 [cylinder_3ds]			
		New Model									
Insert new Model		Cut		Paste		DMXModecast Percent		Select up		New object line	
Delete		Copy		Oops				Select down		Merge children	

Open Models tab in the Fixture Types Editor

2. Select New Model and tap **Insert new Model**.

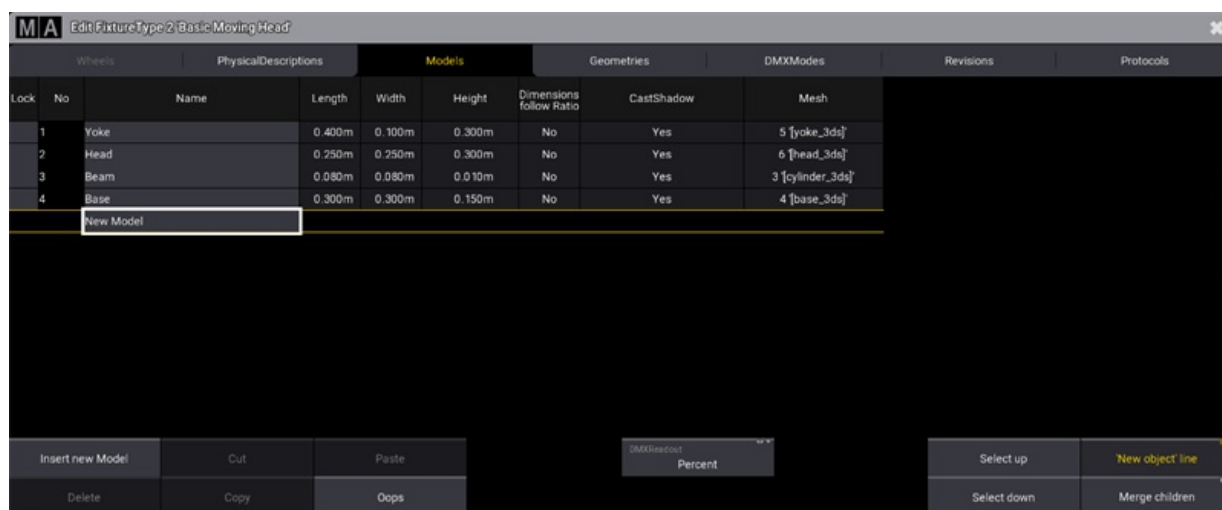
- Model 4 is added.
 3. Rename Model 4 into Base and set its **Mesh** to [base_3ds].
 4. Rename Base/Axis1 into Yoke and set its **Mesh** to [yoke_3ds].
 5. Rename Base/Yoke/Axis 1 into Head and set its **Mesh** to [head_3ds].
 6. Rename Base/Yoke/Head/Beam 1 into Beam and set its **Mesh** to [cylinder_3ds].
 7. Enter the measurements of the fixture, as shown in the table below.



Important:

To avoid the models to automatically resize themselves, set the **Dimensions follow Ratio** column to **No** before entering the measurements.

Name	Length	Width	Height
Yoke	0.4 m	0.1 m	0.3 m
Head	0.25 m	0.25 m	0.3 m
Beam	0.08 m	0.08 m	0.01 m
Base	0.3 m	0.3 m	0.150 m



Insert models and the measurements

The models are automatically taken over to the Geometries in the column Model, except for Base as it was inserted directly in the models.

- Link Base to the model Base.

For more information, see [Link Models to Geometries](#).

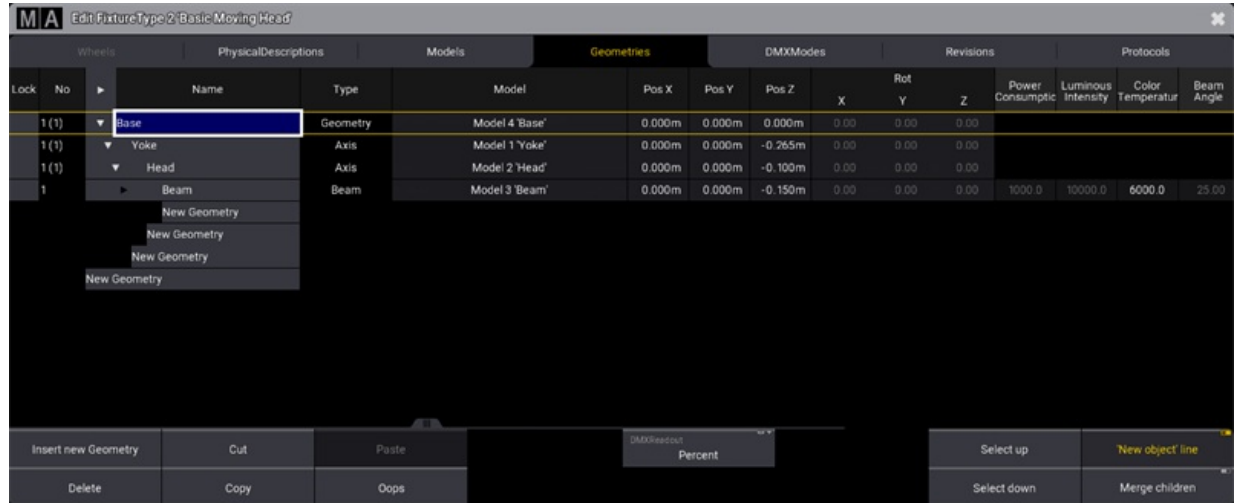


48.2.5. Link Models to Geometry

The offset of the position depends on the measurements of the models.

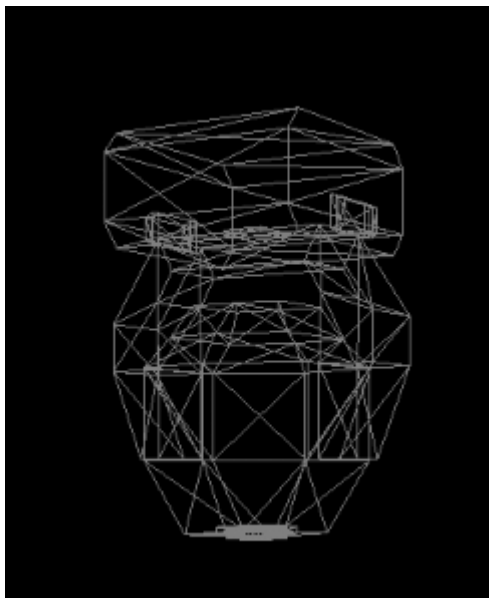
Requirement:

- **Insert Geometries** and **Insert Models**.
 1. Tap the tab **Geometries**.
 2. To link the model Base to geometries, tap, and hold the cell in the column Model.
- Pop-up Select Model opens.
- Select Base.
 3. Set the offset of position in single geometries.
- In Base, leave the default value of Pos Z.
- In Yoke, set Pos Z to -0.265 m.
- In Head, set Pos Z to -0.100 m.
- In Beam, set Pos Z to -0.150 m.



Set the offset of Pos Z

This is the result in the 3D window.



Basic moving head displayed in wireframe in the 3D window



Hint:

To display the basic moving head in wireframe in the 3D window, see [3D](#) topic.



Basic moving head rendered in the 3D window

48.2.6. Link DMX Modes to Geometries

After linking models to geometries, link DMX channels and DMX modes to geometries.

Requirement:

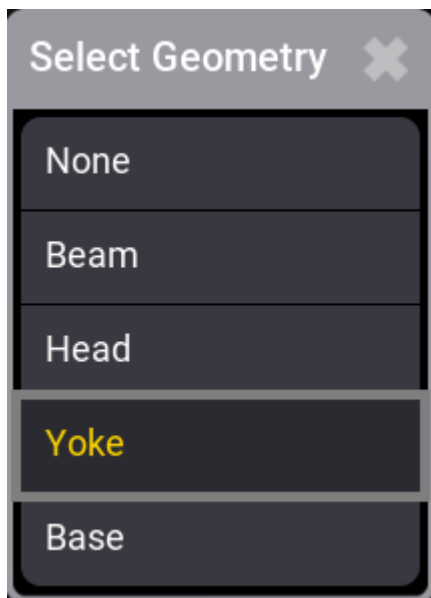
- **Link Models to Geometries.**

1. Tap the tab **DMXModes**.

Edit the column Geometry:

2. In the row Yoke_Pan, tap and hold or press **Edit** and tap the cell.
Select geometry Yoke.

The **pop-up Select Geometry** opens.



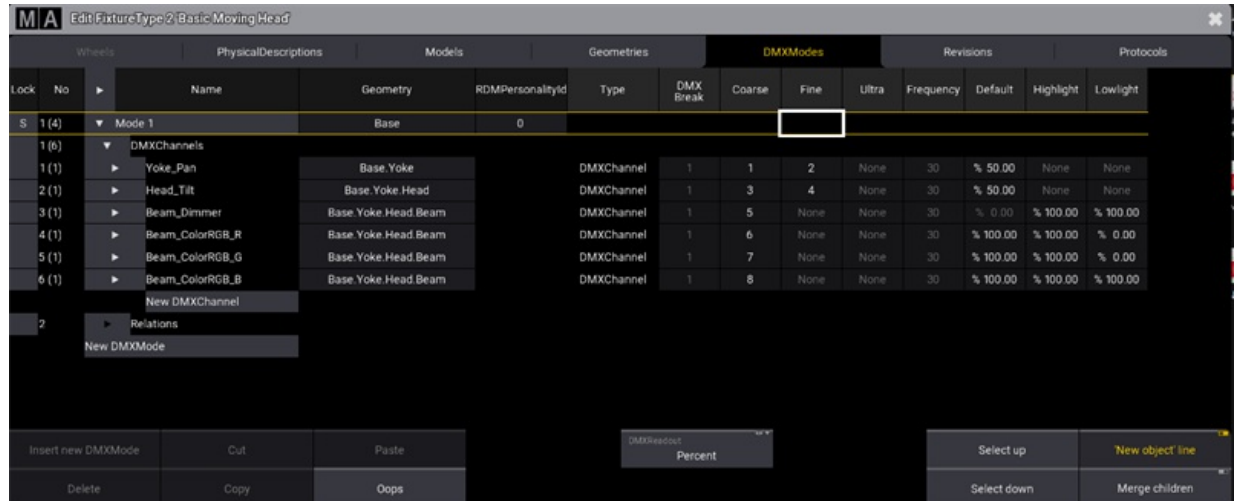
Select geometry

3. In the row Head_Tilt, tap and hold the cell.
Select geometry Head.
4. In the row Beam_Dimmer, tap and hold the cell.
Select geometry Beam.
5. In the rows Base_ColorRG_R, Base_ColorRGB_G, and Base_ColorRGB_B, tap and hold the cell.
Select geometry Beam.



- In row Mode 1, tap and hold the cell.
 Select geometry Base.

DMX channels and DMX modes are now linked to geometries.



Link geometries

48.2.7. Environmental Fixture Types


A way to make a stage more realistic is to use environmental fixture types. These fixture types are anything other than lighting fixtures, such as stages, trusses, seating, or scenery.

Environmental fixture types are part of the fixture library in the Patch menu. The environmental fixture types can be found in the "Set" manufacturer folder of the Insert New Fixtures overlay.

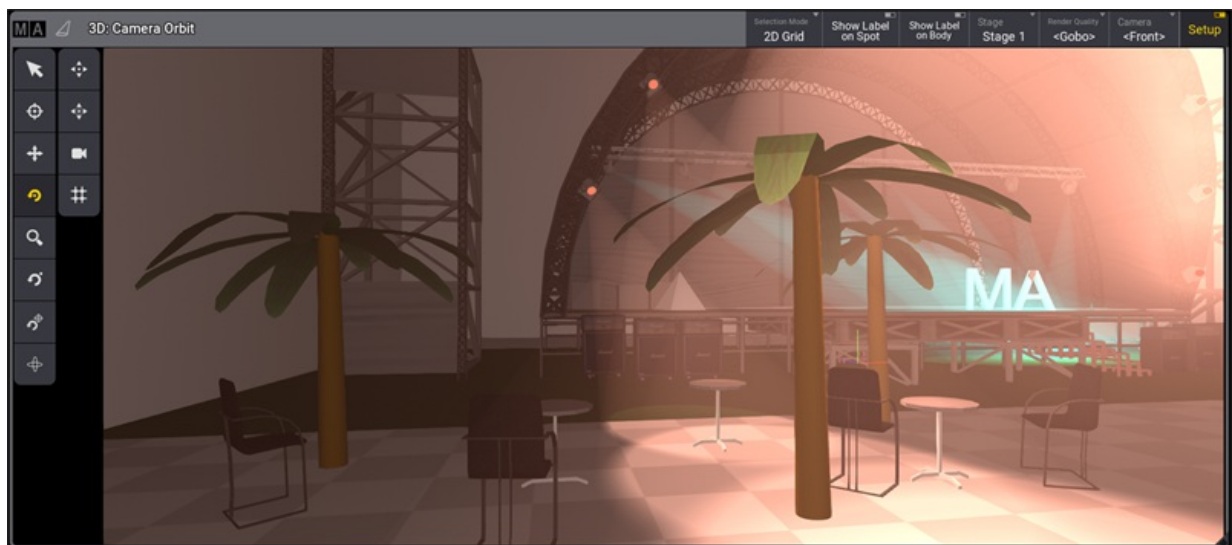
For more information on how to add new fixtures, see [Add fixtures to the show](#).

Environmental fixtures do not have DMX addresses.

The **Hide Environmental** button hides all fixture types that do not have at least one DMX address assigned to a DMX channel. This button can be found in the Patch menu, the Live Patch menu, the Fixture Type section and the Insert New Fixture dialog.



Hint:
www.gdtf-share.com contains even more environmental fixture types.



A stage with environmental fixtures displayed in the 3D window.

48.3. Export Fixture Types

It is possible to export the fixture types to the following locations:

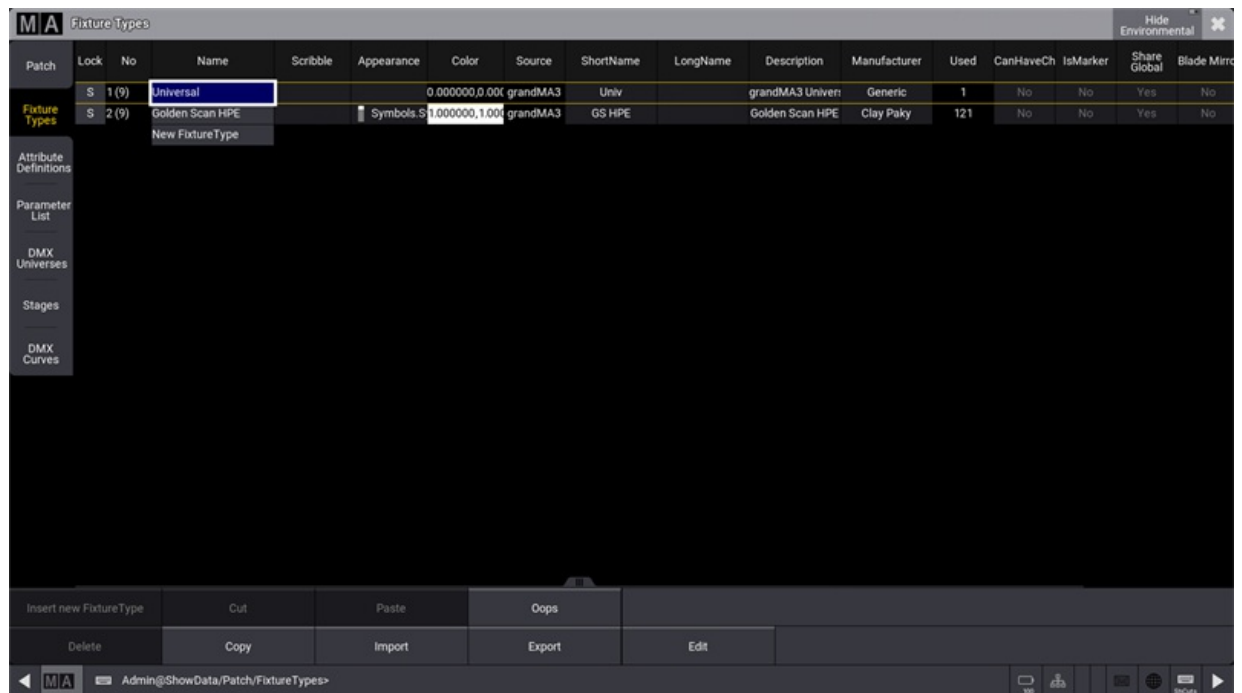
- Internal drive
- External USB drive

It is possible to export the fixture types as the following file formats:

- grandMA3 format
- GDTF

Requirement: Open the Patch.

1. Tap **Fixture Types** in the bar on the left of the patch dialog.
The **Fixture Types** menu opens.



Open Fixture Types

2. Tap at the to be exported fixture type in the list.
3. Tap **Export**.

The **Export Fixture Type** pop-up opens.



Export a fixture type

4. To select the destination drive, tap at **Internal** in the title bar.
If an USB stick is inserted, the USB stick can be chosen.
5. To export the fixture type as grandMA3 format for only grandMA3 use case, tap at **User**.
To export the fixture type as GDTF to use the fixture type within different programs that can read GDTF, tap at **GDTF**.
6. Enter a name.



Hint:

If no name is entered, the fixture type file will be exported as manufacturer@name.xml or manufacturer@name.gdtf.

7. Tap **Export**.

The selected fixture type is exported to the selected drive.

When the selected drive is internal, the fixture type is exported to the computer local drive **(C:)\ProgramData\MALightingTechnology\gma3_library\fixturetypes**.

When the selected drive is external, the fixture type is exported to the external USB Drive **(E:)\grandMA3\library\fixturetypes**.



49. File Management

In addition to saving and loading complete show files, the grandMA3 software supports the exchange and portability of smaller portions of show data; for example, a collection of macros or even a single preset.

To maintain a predictable organizational structure, the software uses well-defined sets of folders within the hard drive of the console, the hard drive of a computer running grandMA3 onPC, and any USB drive connected to either a console or an onPC station.

The following topics cover the export and import functions available within the graphical user interface of the grandMA3 software, as well as the default folder structure created and utilized by the software. For more information about the handling of complete show files, see the [Show File Handling](#) topic.



Important:

The file name must not contain the following characters: \ " \$ * ? ^ | / : < > `

49.1. SFTP Connection to a Console

The consoles can be accessed using an SFTP client program.

One such program could be FileZilla Client from the [FileZilla Project](#) (external link to internet webpage).

The SFTP client needs to be installed on a computer in the same network as the console.

To access the folder structure of the console, the **IP address** of the device is needed. A user name and password is also required. SFTP uses TCP port **22**.

To access the grandMA3 area of the consoles hard drive please use **madata** as both username and password.



Hint:

The grandMA2 area of the grandMA3 consoles can also be accessed using SFTP. Please use **data** as both username and password to access this part.



WARNING:

Be very careful about making any changes in the folders or files. This is direct access to the device. Deleting or moving files may cause serious harm to the device.

Once the connection is made, it is possible to browse through the folders inside the device. The first level of folders has three folders. The "actual" folder gives access to the current software version. The "gma3_library" folder contains exported objects including screenshots. The "installation_packages" contains zip files with necessary installation files.

Learn more about the folder structure in the [Folder Structure topic](#). Read more about screenshots in the [Screenshots topic](#).

49.2. Import / Export Menu

The **Import / Export** menu presents a graphical workflow for exporting objects from the current show file as a smaller file with a minimum of additional show data.




Hint:

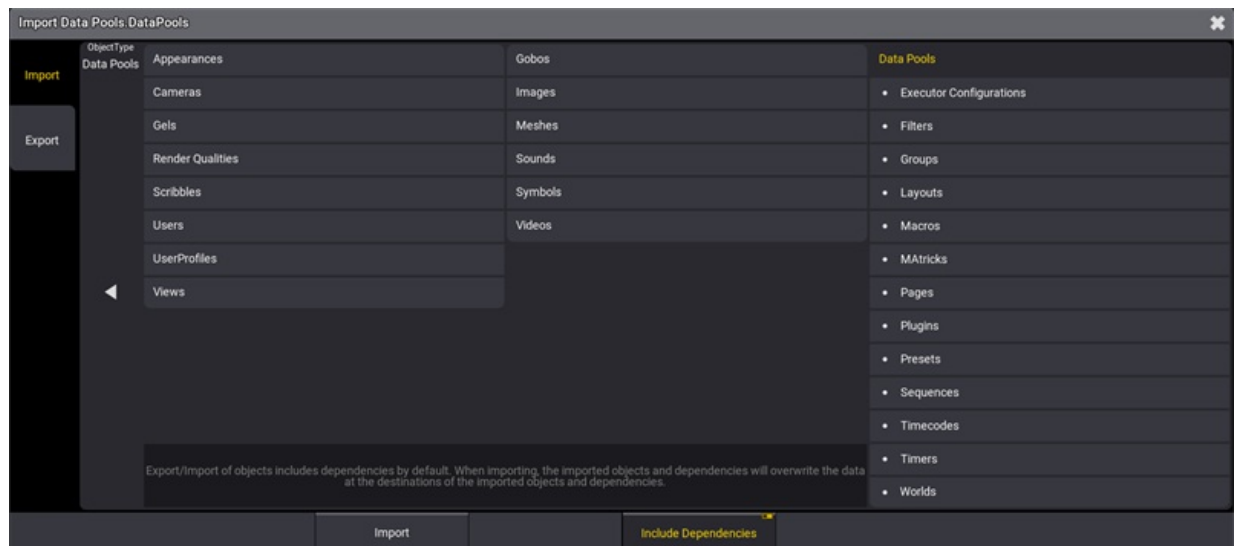
When exporting objects that reference other objects in the show, the referenced objects are automatically included in the export. For example, when exporting a macro with a defined appearance, the exported file includes the referenced appearance.

This menu also allows the import of objects from these smaller files into the current show file. For example, this menu provides an easy way to copy a user profile or selection of macros from one show file to another.

Open the **Import / Export** menu using the following steps:

1. Press the **Menu** key or tap the  icon near the top of the **Control Bar**.
2. Tap **Import / Export** along the bottom row of buttons in the **Menu** pop-up.

Once the main **Import / Export** menu is open, tap the **Import** or **Export** tab in the upper-left corner to access the desired interface.



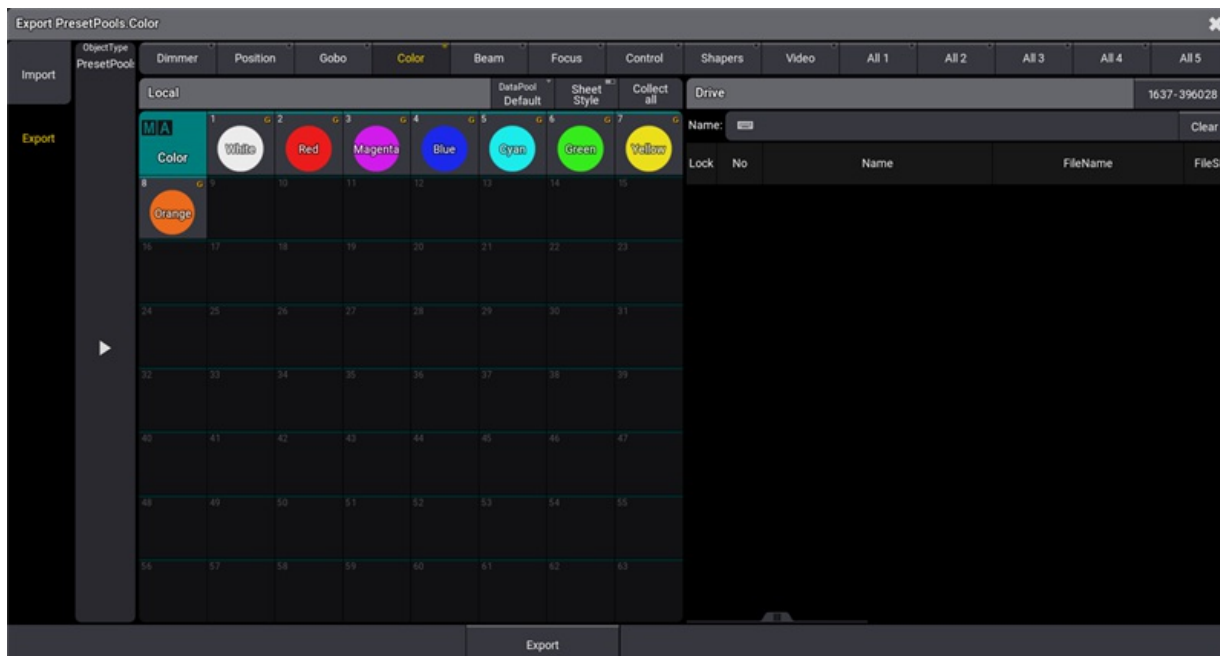
Import / Export menu showing Object Type Data Pools



Object Types

These menus allow for the import and export of a wide variety of object types. Tap **ObjectType** on the left side of the main area to open or close the full list of available object types:

- **Appearances**
- **Cameras**
- **Data Pools**
- **Executor Configurations**
- **Filters**
- **Gels**
- **Gobos**
- **Groups**
- **Images**
- **Layouts**
- **Macros**
- **MAtricks**
- **Meshes**
- **Pages**
- **Plugins**
- **Presets**
- **Render Qualities**
- **Scribbles**
- **Sequences**
- **Sounds**
- **Symbols**
- **Timecodes**
- **Users**
- **UserProfiles**
- **Views**
- **Videos**
- **Worlds**



Export menu prepared to export color presets

The main area of both the **Export** and **Import** menus shows the source of the data on the left and the destination on the right. Therefore, the **Export** menu displays the specified object types from the current show file in the area on the left, in an area titled **Local**, and the contents of the relevant folder of the selected drive on the right, in an area titled **Drive**. In the **Import** menu, these two areas appear on opposite sides.

Drive Selection

To choose a different destination drive for the exported data, tap the **Internal** button at the right side of the title bar of the **Drive** section or swipe to open a list of available drives. In the **Import** menu, this drive setting defines which drive the console uses as the data source.

Local Section Title Bar Tools

The title bar of the **Local** section includes a few buttons with helpful tools.

- Tap **DataPool** to cycle through available data pools within the current show file or swipe to open a list of data pools. This option is only available when importing or exporting an **ObjectType** which is incorporated into a **DataPool** within the show file. These **ObjectTypes** are listed below **DataPools** in the right column of the **ObjectType** section.
- Tap **Sheet Style** to display objects in the **Local** section as a sheet with relevant details for each object. Tap **Sheet Style** again to view objects as they appear in their pools. Some objects, like UserProfiles, never appear in pools. Therefore they will always appear in the sheet format regardless of the state of the **Sheet Style** button.
- Tap **Collect all** to select all objects of the current type for export. Once tapped, this button changes to read **Uncollect all**. Tap this button to deselect all objects. These functions do not appear in the **Import** menu.

Object Sub-Types

Some entries in the **ObjectType** selection include several collections of objects, which appear as different pools. These include:

- **PresetPools**: The **Import / Export** menu handles and displays each preset pool individually.
- **GelPools**: The **Import / Export** menu handles and displays each gel book as an individual pool.

With the **PresetPools** or **GelPools** **ObjectType** selected, a row of radio buttons appears across the top of the main area showing all of the relevant object sub-types. Tap one of these buttons to display the desired pool below.

Multiple Object Selection

Tap any object in the displayed pool to select it for export. A brown border appears around each selected object. To select multiple objects in a pool, tap each of the desired objects. Tap any selected object to deselect it.

When **Sheet Style** is enabled, select multiple adjacent objects by drawing a lasso over all desired objects. To select just one object, tap only the desired object. To toggle selection for any objects in the sheet without changing the selection status of any other objects, hold **Ctrl** on the keyboard while tapping the desired objects in the sheet.

Export Workflow

1. Tap the **Export** tab in the upper-left corner.
2. Set the desired **ObjectType**.
3. Set the desired destination drive.
4. Select all of the desired objects to export from the **Local** area.
5. Enter the desired file name in the Name field of the **Drive** area.
6. Tap **Export** at the bottom of the menu.

Import Workflow

1. Tap the **Import** tab in the upper-left corner.
2. Set the desired **ObjectType**.
3. Set the desired source drive.
4. Select all of the desired files to import from the **Drive** area.
5. Select the desired destination in the pool in the **Local** area.
6. Tap **Import** at the bottom of the menu.



Hint:

Certain object types (including **Gobos**, **Images**, **Symbols**, and **Videos**) can be included in **Appearances**. To automatically create new **Appearances** based on these objects as they are importing, toggle on the **Create Appearances** option before tapping **Import**.



Import and Export Using Command Line Syntax

For information about exporting and importing show data using command line syntax without the use of these menus, see the [Export keyword](#) and [Import keyword](#) topics.

Include Dependencies

The Import and Export taps both have an [Include Dependencies](#) button.

Enabling [Include Dependencies](#) will import and export dependencies.

For more information, see [/NoDependencies keyword](#).

49.3. Folder Structure

The grandMA3 software uses a default set of folders to help maintain a predictable structure when saving data to any internal or external drives.



Hint:

This topic, and other topics in the manual, lists a particular folder name as **gma3_x.y.z**. In this name, the “x.y.z” portion takes the place of the actual installed version number. If the installed version is v1.4.2.1, the name of this folder is **gma3_1.4.2**.

Main Folder

The main folder where all grandMA3 user data is stored by default appears using different labels depending upon the type of drive. This folder can be found under the following names and locations:

On USB Drives:

The main folder on any USB drive connected to either a console or a computer running grandMA3 onPC appears at the root of the drive with the name **grandMA3**.

The full folder structure is created once the drive is connected to the station and selected in the system. There are many ways to select the drive, including the Backup menu, any menu allowing import or export, or the **Select Drive [Drive Number]** command. For more information on the **Drive** keyword, see the [Drive keyword](#) topic.

On Internal Drives of Computers Running grandMA3 onPC:

The full folder structure is created as part of the software installation process.

The main folder on the internal drive of any computer running grandMA3 onPC appears with the name **MALightingTechnology**.

On computers running macOS, the main folder is found at the following location:

[System HD]/Users/[User Name]/MALightingTechnology

On computers running Windows, the main folder is found at the following location:

C:\ProgramData\MALightingTechnology

On the Internal Drive of the Console:

When logged into the console via SFTP, the console presents the main grandMA3 folder as the root directory, using the name /. For more information about SFTP, see the [SFTP Connection](#) topic.

Shows and Backups

Show files and backup show files are saved in dedicated folders, which appear in the following locations:



On USB Drives:

grandMA3/shared/shows
grandMA3/shared/backups

On Internal Drives of Computers Running grandMA3 onPC:

MALightingTechnology/gma3_x.y.z/shared/shows
MALightingTechnology/gma3_x.y.z/shared/backups

On the Internal Drive of the Console:

/actual/shared/shows
/actual/shared/backups



Warning:

Moving, altering, or deleting any files other than show files or backup show files within in the **gma3_x.y.z** folder of an onPC station or the **actual** folder of a console may make the software unstable or render the system unable to boot.

Library

When exporting smaller portions of data from the show file (including objects such as fixture profiles, user profiles, and macros), these files are saved into the **gma3_library** folder of the selected drive unless an alternate destination is specified during the export. Most object types have dedicated default sub-folders within the library.

The complete list of automatically generated library folders and sub-folders includes:

- agendas
- appearances
- certificates
- colorthemes



- datapools
 - executorconfigurations
 - executorpages
 - filters
 - groups
 - layouts
 - macros
 - matricks
 - plugins
 - presets
 - sequences
 - timecodes
 - worlds
- fixturetypesresources
 - gobos
 - meshes
- fixturetypes
- inout
 - artnet
 - dcremotes
 - dmxremotes
 - midiremotes
 - osc
 - outputconfigurations
 - sacn
- media
 - images
 - sounds
 - symbols
 - videos
- mvr
- netkeys
- patch
 - dmxcurves
 - stages
- scribbles

- userprofiles
 - cameras
 - renderqualities
 - screenconfigurations
 - viewbuttons
 - views
- users

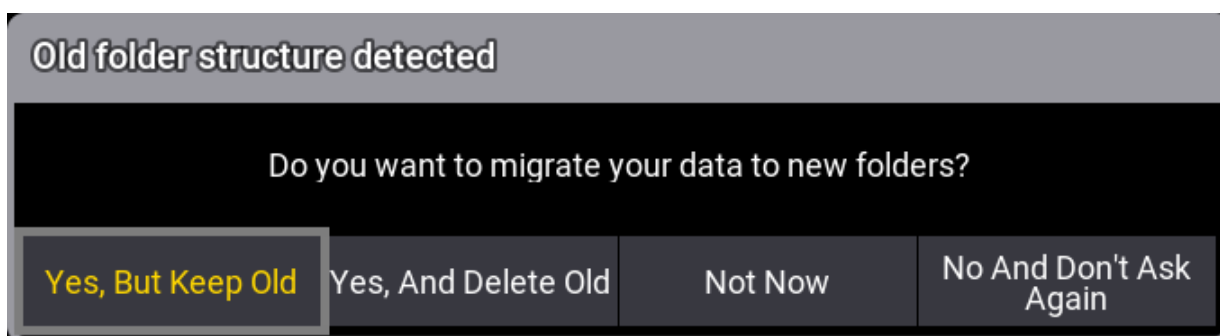
Exceptions to the Default Library Structure

Exporting any objects, which don't have a dedicated sub-folder within the `gma3_library`, saves the files directly to the `gma3_library` folder unless an alternate destination is specified during the export.

Exporting either the parent of multiple objects, which each have their own dedicated folders, (for example, exporting DMXProtocols as one file rather than separate exports of Art-Net and sACN) or children of objects where those children do not have their own dedicated folder (for example, exporting one or more cues rather than a whole sequence), automatically adds a descriptive sub-extension to the file name while saving the file to the most appropriate folder.

Automatic Folder Structure Migration

The default folder structure used by version 1.5 and later of the grandMA3 software is different than the structure used in previous versions. When the software recognizes a USB drive, which includes the older structure but not the newer structure, the software presents a pop-up with options for automatically migrating the existing data to the new structure.



Old folder structure pop-up

Options in the pop-up include:

- **Yes, But Keep Old**: Tap to copy files to the new structure while also maintaining copies in the old structure. This allows both older and newer software versions to continue using the drive.
- **Yes, And Delete Old**: Tap to move files to the new structure and delete the old structure.




- **Not Now** : Tap to make no changes to the current structure. The next time the USB drive is recognized by the software, the same pop-up will appear.
- **No And Don't Ask Again** : Tap to make no changes to the current structure. The software will not present this pop-up when the USB drive is recognized in the future.

50. Shut down the System

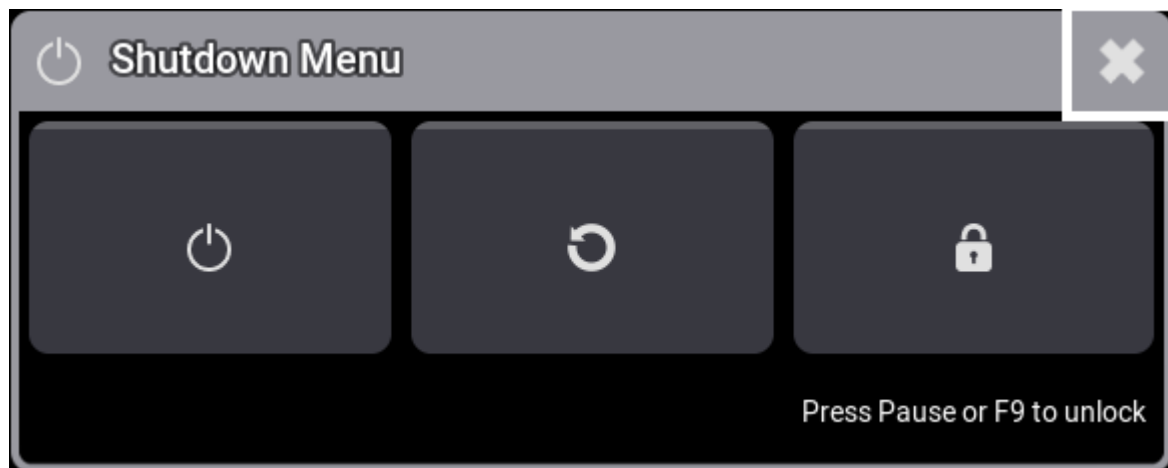
Before shutting down the system, save the show file via the **Backup menu**.




To start the shutdown procedure, use one of the following options:

Use the Software UI

- Tap the  icon at the top of the control bar.

The shutdown menu opens:



- To shut down, tap the  icon.
- To restart, tap the  icon.
- To lock the desk, tap the  icon. For more information, see **desk lock**.

Use the Power Key

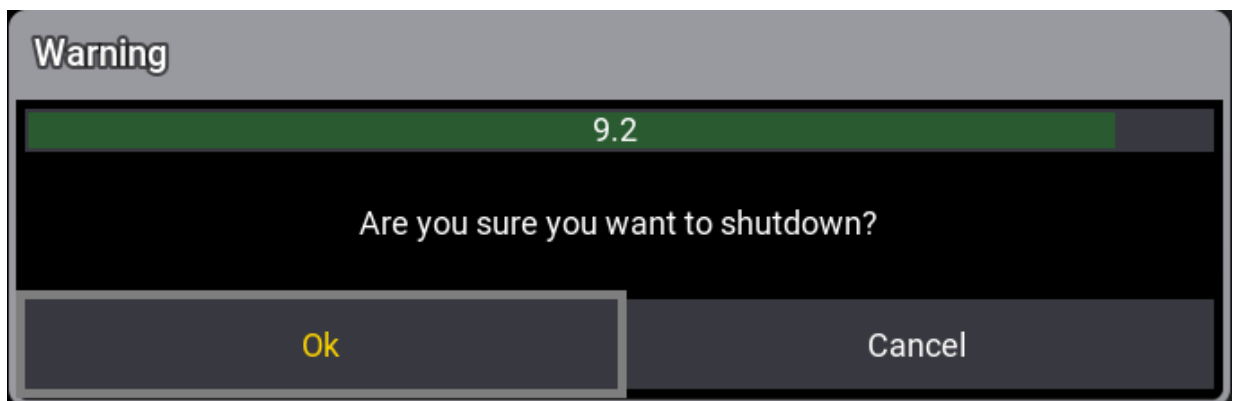
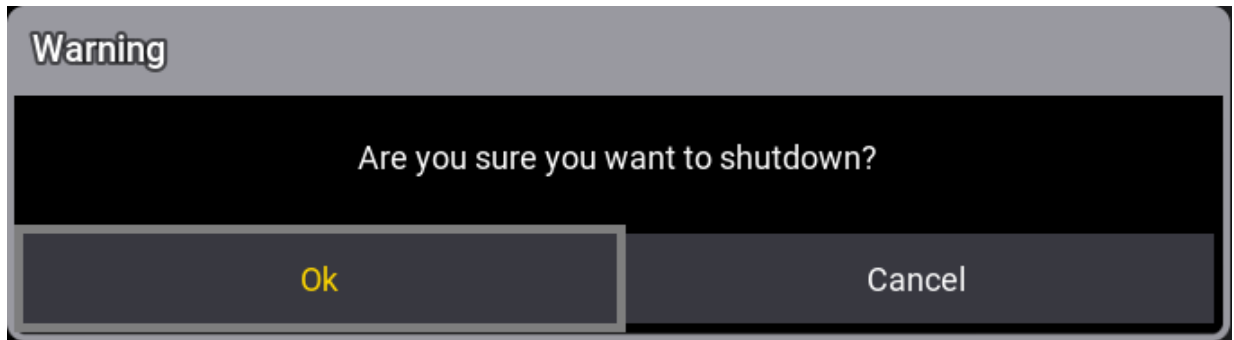
- Press the **power key** on the front panel.

Use a Keyword

- **Reboot keyword**
- **Restart keyword**
- **ShutDown keyword**


Shutdown Warning Pop-Up


One of the following warning pop-ups appears:



- Tap **OK**.

The grandMA3 device shuts down.

 **Hint:**
When using the **ShutDown keyword**, you can choose an option without a shutdown pop-up or without timer.

 **Hint:**
If an attempt is made to shut down before saving the show file, a save show file pop-up will appear. For more information, see **Save Show File**.

- To shut down the current station without confirmation, type:

```
MA User name[Fixture]> ShutDown /nc
```




- To shut down the current station without timer, type:

```
MA User name[Fixture]> ShutDown /noautoclose
```



51. Troubleshooting

In case your grandMA3 device does not work as expected, the topics below include helpful general troubleshooting steps as well as detailed solutions to some possible issues.

51.1. Clean Start

A clean start clears the current show and all user settings as the grandMA3 software boots. This may be a helpful troubleshooting step in cases where the software boots normally but encounters an issue when the show file loads.

Clean Start a grandMA3 console, Replay Unit, or Processing Unit



Important:

This process relies on the **Ctrl** or **control** key commonly found at or near the bottom-left or bottom-right corner of an alpha-numeric keyboard.

For devices without an integrated keyboard, an external USB keyboard is required.

To perform a clean start on a grandMA3 console, replay unit, or processing unit:

1. Make sure the integrated keyboard is functional or connect an external USB keyboard.
2. Boot or reboot the device, watching for the mode selection pop-up.
3. When the mode selection pop-up appears, press and hold the **Ctrl** key.
4. While holding the **Ctrl** key, tap the desired mode in the mode selection pop-up.
5. Once the mode selection pop-up disappears, release the **Ctrl** key.
6. The device completes the boot process with no show file loaded and all user settings reset to factory defaults.

Clean Start grandMA3 onPC

- To perform a clean start on a Windows system, run the grandMA3 onPC x.x.x.x Clean Start app from the MA Lighting folder.
- To perform a clean start on a macOS system, press and hold the **option** key on the keyboard and then open the grandMA3 onPC application. An alert pop-up appears, confirming the clean start operation. Click **Continue** to proceed with the clean start. Otherwise, click **Cancel**.

The software starts with no show file loaded and all user settings reset to factory defaults.

51.2. Update Does Not Work

There may be several reasons for a software update to fail:

Broken Installer Package



Restriction:

The installer package zip file may be broken after the download.

1. Delete the old installer package.
2. Download a new installer package.

Installation from the Zip File



Restriction:

Do not install the software from the zip file.

1. Unzip the file to your computer's hard drive.
2. Install the software from the uncompressed folder.

Broken USB Flash Drive



Restriction:

Older USB flash drives may be worn and cannot be read correctly.

- Format the USB flash drive.
- Replace the USB flash drive with a new one.



Hint:

New out-of-the-box USB flash drives may come without a partition table.

1. Check the USB flash drive by copying and pasting other files and folders from one computer to another.
2. Copy the installation packages if the USB flash drive works perfectly.
3. Perform software update.

Broken USB Port



Restriction:

The USB port may be broken on the grandMA3 device or on the computer.

- Use a different USB port.



- Update the USB port driver.
- Replace the broken USB port.

Broken USB Cable



Restriction:

The USB cable may be broken.

- Use a different USB cable.



51.3. Station Does Not Connect

If a station or a wing does not connect, reset the station settings with a **Clean Start**.

If the command wing or fader wing fails to connect to the onPC software, try switching the product off and then on again.

Reasons that can prevent an onPC command wing or fader wing communication with a computer:

- A damaged USB port on the command wing or fader wing.
- A damaged USB port on the computer.
- A damaged USB cable.
- Recently installed operating system updates or power settings.
- An incomplete or corrupted onPC software installation.

51.4. Panic Macro



Important:

Make sure to save the show file before executing the Panic macro.



Restriction:

Do not use the Panic macro during a live show.



Important:

Be aware that the use of the panic macro can have a severe impact on the show.

In different cases, for example, if a show file is not working as expected before a show, it can be useful to use the "Panic" macro.

The panic macro will restore and reset general settings to default. These general settings are listed below:

1. Clear the Programmer (ClearAll)
2. Change Command Line back to root destination
3. Switch off Web Remote
4. Switch off Remote HID
5. Switch off RDM
6. Switch off Agenda
7. Switch off DC/ MIDI/ DMX Remotes
8. Switch off OSC Input
9. Switch off OSC Output
10. Switch off PSN Input
11. Go to World 1
12. Select Filter 1
13. Switch off Freeze
14. Switch off Preview
15. Switch off Plugins
16. Switch off Sequences
17. Switch off Timecodes
18. Switch off all Executors
19. Switch off Timecode Slots
20. Switch off Master Mode of Groups
21. Set the Master Fader of all Groups to 100%
22. Switch Off DMX Tester
23. Set the Grand Master to 100%
24. Set the World Master to 100%
25. Set the Highlight Master to 100%
26. Switch off Highlight
27. Set the Lowlight Master to 100%



28. Switch off Lowligh
29. Set Solo Master to 100%
30. Switch off Solo
31. Set Grand Rate to 1
32. Set Grand Speed to 60 BPM
33. Set Programmer Time to 0s
34. Switch off Programmer Time
35. Set Programmer X Fade to 0s
36. Switch off Programmer X Fade
37. Set Executor Time to 0s
38. Switch off Executor Time
39. Set Executor X Fade to 0s
40. Switch off Executor X Fade
41. Set all Speed Masters to 60 BPM
42. Set the Blind Master to 100%
43. Switch off Blind
44. Enable Sync for Phasers in the Programmer
45. Disable SingleStep for Phasers in the Programmer
46. Switch off Align
47. Set Value Readout to Natural
48. Set Wheel Mode to Additive
49. Request all Universes
50. Set Merge Mode of all Universes to HTP
51. Switch off Art-Net and sACN Output
52. Switch off Art-Net and sACN Input
53. Clear Programmer for all Users
54. Switch to first Feature Group
55. Reload the User Interface
56. Switch off all Macros



Hint:

To see the list from above in the software, press **Edit** and then tap the **Panic** macro in the macro pool.

- To learn more about macros, see [Macros](#).

Import and Execute Panic Macro

1. Search for Panic in the import list. For more information about importing macros, see [Create Macros](#).
2. After import, tap **Panic** in the Macros pool window. The text editor opens.
3. To confirm the panic macro, press **Please** **Please**.
4. The panic macro is executed.





52. Glossary

The following collection of abbreviations, acronyms and fundamental lighting terms can be found in the grandMA3 user manual.

A

ACN - Architecture for Control Networks

Is a suite protocol. It uses a lot of elements that are currently not supported by grandMA3. But the ACN protocols also have a version for transporting DMX data. For more information on ACN, see [sACN menu](#).

ANSI - American National Standards Institute

Art-Net

Art-Net 4 Ethernet communication standard is developed by Artistic Licence. For information about Art-Net in the grandMA3 software, see [Art-Net menu](#).

Attributes

Attributes are the building blocks of fixture types. For more information, see [Attribute definitions](#).

B

bit - binary digit

bpm - beats per minute

C

CIE - Commission internationale de l'éclairage

CLI - Command-line interaction

Command Line

The syntax to control the console can be entered in the Command Line. For more information, see [Command Line](#).

Conventional lights

Lighting device with dimmer only.

CPU - Central Processing Unit

Cue

Cues a stored values. A set of cues is a Sequence. For more information, see [Cues and Sequences](#).



D

dBu - decibel unloaded

DC - Direct Current

DHCP - Dynamic Host Control Protocol

Is a system where IP addresses are distributed from a DHCP server. For more information on how to set IP addresses, see [Interfaces and IP](#).

DMX - Digital Multiplex

Communication protocol to send a digital signal from a controller to a receiver. A single DMX cable can transmit one Universe.

DMX Channel

A single DMX Channel has a value between 0 and 255.

DMX Universe

A DMX Universe contains 512 DMX channels.

Dual Encoder

The five dual encoders on a console or a command wing are used to adjust the different attributes of the fixtures or other functions. For more information, see [Dual Encoders](#).

E

Encoder Bar

Displays the attributes or functions linked to the encoders and includes shortcuts for quick access to a selection of user profile settings and programming tools. For more information, see [Encoder Bar](#).

Executors

Executors (Exec) are physical keys, knobs, and faders on the grandMA3 hardware. They can also be represented as on-screen virtual executors. For more information, see [Executors](#).

Extension

Allows to extend the amount of physical executor handles for specific grandMA3 products. For more information, see [Connect grandMA3 extension](#).



F

Fader Wing

An additional fader for the grandMA3 onPC system. For more information, see [grandMA3 onPC fader wing](#).

Fixture

A complete lighting unit including at least a housing and a lamp.

Fixture Type

Fixture types are used to visualize and control real-life fixtures on stage. For more information, see [Fixture Types](#).

fps - frames per second

G

GB - Gigabyte

GDTF - General Device Type Format

Is a file format used to describe devices. For information about GDTF, see [GDTF Share](#).

GlobalMaster

This is the session master station in all networked systems. For more information, see [Locally Networked Devices](#).

GPI - General Purpose Interface

Grand Master

Is used to limit the output of the intensity of all the fixtures patched in a show. For more information, see [Grand Master](#).

Groups

Groups contain a selection of fixtures. This includes the order and grid position of the selection. This can be used as a programming tool, where it is a fast way to select the fixtures. For more information about Groups, see [Groups](#).

H

Hex - Hexadecimal

Highlt - Highlight

See [Highlt key](#).

HTP - Highest Takes Precedence

The highest value will be output, mostly used with Dimmer. For more information about priorities, see [LTP](#).

Hz - Hertz



I

IdleMaster

When a console is ready to be connected to another station, the console is in IdleMaster mode. For more information, see [Standalone Device](#).

IEC - International Electrotechnical Commission

IP - Internet Protocol

K

Keywords

Defined word in the software with a specific usage.

L

LED - Light Emitting Diode

An intelligent light with a LED source.

Level Wheel

The level wheel on the right side of a console or command wing is always assigned to dimmer attributes.

log - logarithm

Lua

Lua is a scripting language designed to support general procedural programming. For more information, see [What is Lua](#).

LTC - Linear Time Code

LTP - Latest Takes Precedence

Most recent value will be output, mostly used for all attributes besides Dimmer. For more information about priorities, see [DMX port configuration](#).



M

mA - milliampere

Macro

Macros are commands stored in a pool object. For more information, see [Macros](#).

Master

Masters are physical representations of various timing and level overrides that exist in the software. For more information, see [Masters](#).

MB - Megabyte

MIB - Move In Black

For more information, see [Move In Black](#).

MIDI - Musical Instrument Digital Interface

A standard communication protocol. For more information about MIDI, see [MIDI](#).

Moving Light

Lighting instrument with additional attributes other than dimmer. It is possible to control the position of a Moving Light from a console.

MSC - MIDI Show Control

Sends functions via MIDI signal from an executor to another device, for example a console or MIDI sequencer.

Multi-Instance Fixture

A lighting instrument with multiple individually controllable sections.

MVR - My Virtual Rig

MVR is a file format that is used to share data for a patch. For more information, see [MVR](#).

O

OS - Operating System

OSC - Open Sound Control

OSC is a client and server system that defines a message address pattern used to address elements in the receiving server. For more information, see [OSC](#).



P

Pan

Horizontal movement axis of a fixture.

Parameters

A parameter can use several DMX channels. For more information, see [Parameters](#).

Phasers

Dynamically changes the output for attributes using a set of information in two or more steps. For more information, see [Phasers](#).

PoE - Power over Ethernet

Playback

Recalling stored information physically (for example, fader) or graphically (user interface) on a console. A preset, a sequence with cues, or a timecode show can be played back. For more information, see [Playback Window](#).

Presets

Presets are containers for values. For example, when the same values are used in different cues several times. For more information, see [Presets](#).

Prev - Previous

See [Prev key](#).

Programmer

The programmer is a temporary memory, where the active values are placed. The values can then be stored or released. For more information, see [Programmer](#).

Prvw - Preview

See [Prvw key](#).

PSN - PosiStageNet

grandMA3 stations can receive PSN data. For more information, see [PSN](#).

PWM - Pulse Width Modulation



R

RAM - Random Access Memory

RCA - Radio Corporation of America

RDM - Remote Device Management

Permits intelligent bidirectional communication between devices from multiple manufacturers using a modified DMX512 data link. For more information, see **RDM**.

RPM - Rounds per Minute

S

S/PDIF - Sony/Philips Digital Interconnect Format

sACN - Streaming Architecture for Control Networks

For more information, see **ACN**.

SelfFix - SelectFixture

See **SelfFix key**.

Sequ - Sequence

See **Sequ key**.

SMPTE - Society of Motion Picture and Television Engineers

SSD - Solid-State Drive

SSE - Streaming SIMD Extensions

STP - Shielded Twisted Pair

Syntax

The command line syntax is used to create valid commands. For more information, see **Syntax**.



T

TC - Timecode

Timecode is a time signal used to let off the recordings of a playback or trigger cues. For more information, see [Timecode](#).

TCP - Transmission Control Protocol

Tilt

Vertical movement axis of a fixture.

Tracking

Tracking is the principle to store only the changes in the cues. For more information, see [Tracking](#).

U

UDP - User Datagram Protocol

UI - User Interface

See [User Interface](#).

V

V - Volt

VRAM - Video Random Access Memory

W

Web remote

See [Web remote](#).

X

XFade - Crossfade

For more information, see [Assign object to an executor](#).

XLR - External Line Return

XML - Extensible Markup Language

52. Index

- [Minus] key [2.15.4.](#)
- [Minus] Keyword [8.3.7.](#)
- % [Percent] Keyword [8.3.9.](#)
- * [Asterisk] Character [8.3.8.](#)
- . [Dot] Character [8.3.3.](#)
- / [Slash] | * [Asterisk/Multiply] key [2.15.6.](#)
- / [Slash] Character [8.3.2.](#)
- /Active option keyword [8.4.1.](#)
- /ActiveForSelected Option Keyword [8.4.2.](#)
- /AddNewContent Option Keyword [8.4.3.](#)
- /All Option Keyword [8.4.4.](#)
- /AllForSelected Option Keyword [8.4.5.](#)
- /Ask Option Keyword [8.4.6.](#)
- /Auto Option Keyword [8.4.7.](#)
- /CopyCueDestination Option Keyword [8.4.8.](#)
- /CopyCueScr Option Keyword [8.4.9.](#)
- /CreateReferenceObject Option Keyword [8.4.10.](#)
- /CreateSecondCue option keyword [8.4.11.](#)
- /CueOnly option keyword [8.4.12.](#)
- /Date option keyword [8.4.13.](#)
- /Default Option Keyword [8.4.14.](#)
- /DiscardChanges Option Keyword [8.4.15.](#)
- /DMX Option Keyword [8.4.16.](#)
- /DMXProtocols Option Keyword [8.4.17.](#)
- /Embed Option Keyword [8.4.18.](#)
- /Enumerate Option Keyword [8.4.19.](#)
- /File Option Keyword [8.4.20.](#)
- /ForceGlobal Option Keyword [8.4.21.](#)
- /Gaps Option Keyword [8.4.23.](#)
- /GDTF Option Keyword [8.4.22.](#)
- /Global Option Keyword [8.4.24.](#)
- /GridMergeMode Option Keyword [8.4.25.](#)
- /Indirect Option Keyword [8.4.26.](#)
- /InputFilter Option Keyword [8.4.27.](#)
- /KeepActivation Option Keyword [8.4.28.](#)
- /LocalSettings Option Keyword [8.4.29.](#)



[/Look Option Keyword 8.4.30.](#)
[/MAtricks Option Keyword 8.4.31.](#)
[/Merge Option Keyword 8.4.32.](#)
[/MergeHighPriority Option Keyword 8.4.33.](#)
[/MergeLowPriority Option Keyword 8.4.34.](#)
[/MoveValues Option Keyword 8.4.35.](#)
[/NoAutoClose Option Keyword 8.4.36.](#)
[/NoConfirmation Option Keyword 8.4.37.](#)
[/NoDependencies Option Keyword 8.4.38.](#)
[/NoOops Option Keyword 8.4.39.](#)
[/NoRefresh Option Keyword 8.4.40.](#)
[/NoSave Option Keyword 8.4.41.](#)
[/NoShowData Option Keyword 8.4.42.](#)
[/NoSubFolders Option Keyword 8.4.43.](#)
[/OriginalContentOnly Option Keyword 8.4.44.](#)
[/Output Option Keyword 8.4.45.](#)
[/OutputStations Option Keyword 8.4.46.](#)
[/Overwrite Option Keyword 8.4.47.](#)
[/PatchOffset Option Keyword 8.4.48.](#)
[/Path Option Keyword 8.4.49.](#)
[/PhaserData Option Keyword 8.4.50.](#)
[/Programmer Option Keyword 8.4.51.](#)
[/Release Option Keyword 8.4.52.](#)
[/Remove Option Keyword 8.4.53.](#)
[/Save Option Keyword 8.4.54.](#)
[/Screen Option Keyword 8.4.55.](#)
[/Selection Option Keyword 8.4.56.](#)
[/Selective Option Keyword 8.4.57.](#)
[/Single Option Keyword 8.4.58.](#)
[/Tab Option Keyword 8.4.59.](#)
[/Type Option Keyword 8.4.60.](#)
[/Universal Option Keyword 8.4.61.](#)
[/Wrap Option Keyword 8.4.62.](#)
[; \[Semicolon\] Character 8.3.1.](#)
[+ \[Plus\] key 2.15.5.](#)
[+ \[Plus\] Keyword 8.3.10.](#)
[<<< GoFastBackward | Black key 2.15.2.](#)



<<< GoFastBackward Keyword [8.3.5.](#)
>>> [GoFastForward] | Flash key [2.15.3.](#)
>>> GoFastForward Keyword [8.3.6.](#)
3D Viewer [13.10.](#)
Absolute [2.15.59.](#)
Absolute Keyword [8.3.11.](#)
Acceleration keyword [8.3.12.](#)
Action keyword [8.3.13.](#)
activation group [13.15.1.](#)
ActivationGroup Keyword [8.3.14.](#)
agenda [35.](#) [35.1.](#) [35.2.](#) [35.4.](#)
agenda - edit - entry [35.3.](#)
Agenda Keyword [8.3.15.](#)
align [14.8.](#)
align bar [14.8.](#)
Align key [2.15.9.](#)
Align Keyword [8.3.16.](#)
AlignTransition Keyword [8.3.17.](#)
alwayson [30.](#)
appearance [21.](#)
Appearance [2.15.67.](#)
appearance - assign [21.2.](#)
appearance - create [21.1.](#)
appearance - delete [21.3.](#)
appearance - use [21.2.](#)
Appearance Keyword [8.3.18.](#)
arrange [38.7.](#)
Art-Net - configuration [11.2.1.](#)
Art-Net - transmit example [11.2.3.](#)
assign - layout - multipatch - fixture [38.2.](#)
Assign key [2.15.8.](#)
Assign Keyword [8.3.19.](#)
At Filter [25.1.](#)
at filter menu [2.15.7.](#)
At key [2.15.7.](#)
At Keyword [8.3.20.](#)
attribute [2.15.48.](#)



[attribute - definitions 13.15.](#)
[Attribute Keyword 8.3.21.](#)
[audio device 5.7.](#)
[AutoCreate keyword 8.3.22.](#)
[backup shows 6.](#)
[Black Keyword 8.3.23.](#)
[blank 6.3.](#)
[blind 14.2.](#)
[Blind key 2.15.10.](#)
[Blind Keyword 8.3.24.](#)
[Block Keyword 8.3.25.](#)
[BPM 43.](#)
[BPM keyword 8.3.26.](#)
[Call 2.15.40.](#)
[Call Keyword 8.3.27.](#)
[camera 13.12.](#)
[Camera Keyword 8.3.29.](#)
[CancelSoftwareUpdate Keyword 8.3.28.](#)
[Capture Keyword 8.3.30.](#)
[ChangeDestination Keyword 8.3.31.](#)
[Channel key 2.15.12.](#)
[Channel Keyword 8.3.32.](#)
[ChannelSet Keyword 8.3.33.](#)
[Chat Keyword 8.3.34.](#)
[ChatJoin Keyword 8.3.35.](#)
[ChatLeave Keyword 8.3.36.](#)
[Circle 32.3.](#)
[Circle Phaser 32.3.](#)
[Clean start 51.1.](#)
[Clean Start 2.15.14.](#)
[CleanUp Keyword 8.3.37.](#)
[Clear key 2.15.11.](#)
[Clear Keyword 8.3.38.](#)
[ClearActive 2.15.11.](#)
[ClearActive Keyword 8.3.39.](#)
[ClearAll 2.15.11.](#)
[ClearAll Keyword 8.3.40.](#)



[ClearSelection 2.15.11.](#)
[ClearSelection Keyword 8.3.41.](#)
[Clone 2.15.64.](#)
[Clone keyword 8.3.42.](#)
[Collect Keyword 8.3.44.](#)
[Collection Keyword 8.3.45.](#)
[Color 7.12.2.](#)
[Color Keyword 8.3.46.](#)
[color picker 14.5.](#)
[color theme 42.3.](#)
[Color Theme 7.12.3.](#)
[color wheel only 14.5.](#)
[ColorDefinition Keyword 8.3.47.](#)
[colors 7.12.](#)
[colors – system colors 7.12.1.](#)
[ColorTheme Keyword 8.3.48.](#)
[Command section 2.16.1.](#)
[command syntax - extended 8.6.](#)
[Command Syntax and Keywords 8.](#)
[command syntax and keywords – syntax rules 8.1.](#)
[Command Wing Bar 7.11.](#)
[CommandDelay Keyword 8.3.43.](#)
[compare 7.12.4.](#)
[Configuration keyword 8.3.49.](#)
[configure display 9.1.](#)
[Conflicts in Fixture Types 48.1.2.](#)
[Connect grandMA3 fader wing 4.15.](#)
[Connect MIDI 4.10.](#)
[connector 2.1.](#)
[connector pin assignment 2.17.](#)
[Console Keyword 8.3.50.](#)
[Content Sheet 27.3.](#)
[context sensitive help 1.1.](#)
[Control elements 2.16.](#)
[Control Other MA Devices 46.](#)
[Cook 2.15.57.](#)
[Cook Keyword 8.3.51.](#)



[Copy key 2.15.13.](#)
[Copy Keyword 8.3.52.](#)
[countdown 36.2.](#)
[Ctrl key 2.15.14.](#)
[cue - break 27.1.2.](#)
[cue - copy 27.7.](#)
[cue - playback 27.10.](#)
[cue - recipe 27.8.](#)
[cue - renumber 27.13.](#)
[cue - store 27.5.](#)
[cue - timing 27.12.](#)
[Cue key 2.15.15.](#)
[Cue Keyword 8.3.53.](#)
[CueAbsolute Keyword 8.3.54.](#)
[CueDelay 2.15.59.](#)
[CueDelay Keyword 8.3.55.](#)
[CueFade 2.15.59.](#)
[CueFade Keyword 8.3.56.](#)
[CueInDelay Keyword 8.3.57.](#)
[CueInFade Keyword 8.3.58.](#)
[CueOutDelay Keyword 8.3.59.](#)
[CueOutFade Keyword 8.3.60.](#)
[CuePart 2.15.15.](#)
[CueRelative Keyword 8.3.61.](#)
[cues 27.](#)
[cues - delete 27.14.](#)
[CueUpdate keyword 8.3.62.](#)
[CurrentEnvironment Keyword 8.3.63.](#)
[CurrentUser Keyword 8.3.64.](#)
[CurrentUserProfile Keyword 8.3.65.](#)
[custom section 2.1. 2.16.3.](#)
[Custom Section 28.4.](#)
[Custom/Master Section 28.4.](#)
[Cut Keyword 8.3.66.](#)
[data pool 41.](#)
[DataPool Keyword 8.3.67.](#)
[DC Remotes 44.1.](#)



[deactivation group 13.15.3.](#)
[Deceleration Keyword 8.3.68.](#)
[Decimal16 keyword 8.3.70.](#)
[Decimal24 Keyword 8.3.71.](#)
[Decimal8 Keyword 8.3.69.](#)
[Default Keyword 8.3.72.](#)
[Delay 2.15.59.](#)
[Delay Keyword 8.3.73.](#)
[Delete key 2.15.17.](#)
[Delete keyword 8.3.74.](#)
[DeleteGlobalVariable keyword 8.3.76.](#)
[DeleteOtherVersions Keyword 8.3.75.](#)
[DeleteUserVariable keyword 8.3.77.](#)
[demofiles 6.](#)
[desk light 42.3.](#)
[desk light - connect 4.5.](#)
[Desk lock 7.1.2.](#)
[device - position 4.3.](#)
[device - unpack 4.1.](#)
[device overview 2.](#)
[Dismiss Keyword 8.3.78.](#)
[Display Configuration 7.1.1.](#)
[Display Keyword 8.3.79.](#)
[DMX 11.](#)
[DMX - connect 4.8.](#)
[DMX - Ethernet 11.2.](#)
[DMX - input 11.1.](#)
[DMX - output 11.1.](#)
[DMX - via network 11.2.](#)
[DMX curves 13.17.](#)
[DMX Remotes 44.3.](#)
[DMX sheet 13.6.](#)
[DMX tester 13.6.](#)
[DMX universe 13.7.](#)
[DMXAddress Keyword 8.3.80.](#)
[DMXAdress 2.15.71.](#)
[DMXLayer Keyword 8.3.81.](#)



[DMXReadout Keyword 8.3.82.](#)
[DMXUniverse 2.15.71.](#)
[DMXUniverse Keyword 8.3.83.](#)
[dot key 2.15.1.](#)
[DoubleSpeed Keyword 8.3.84.](#)
[Down key 2.15.16.](#)
[Down Keyword 8.3.85.](#)
[Drive Keyword 8.3.87.](#)
[DrowOwnership Keyword 8.3.88.](#)
[dual encoder key 2.16.4.](#)
[dual encoder section 2.1.](#)
[dual encoders 2.16.4.](#)
[DumpLog Keyword 8.3.86.](#)
[Echo Keyword 8.3.89.](#)
[Edit key 2.15.19.](#)
[Edit Keyword 8.3.90.](#)
[edit layout 38.3.](#)
[Edit layout elements 38.6.](#)
[Edit layout view 38.5.](#)
[EditSetting Keyword 8.3.91.](#)
[Effect Keyword 8.3.93.](#)
[Eject Keyword 8.3.92.](#)
[encoder 38.7.](#)
[EncoderPage Keyword 8.3.94.](#)
[encoder-resolution-multiplier 14.4.](#)
[EndIf 2.15.31.](#)
[EndIf keyword 8.3.95.](#)
[End-user License Agreement \(EULA\) 4.16.](#)
[entry 35.2.](#)
[Environment Keyword 8.3.96.](#)
[Environmental Fixture Types 48.2.7.](#)
[Equal character 8.3.4.](#)
[Esc key 2.15.18.](#)
[examples 44.4.1.](#)
[Exchange 2.15.36.](#)
[Exchange Keyword 8.3.97.](#)
[Exec 2.15.79.](#)



[exec config 28.2.](#)
[executor 28.](#)
[executor - assign object 28.1.](#)
[executor configurations 28.2.](#)
[executor elements 2.16.7.](#)
[Executor Keyword 8.3.98.](#)
[Executor Time 30.2.1.](#)
[Export 49.2.](#)
[Export Keyword 8.3.99.](#)
[Extension Keyword 8.3.101.](#)
[external keyboard - connect 4.7.](#)
[external sources 37.9.](#)
[extract 2.15.7.](#)
[Extract keyword 8.3.100.](#)
[Fade 2.15.59.](#)
[Fade Keyword 8.3.102.](#)
[FaderCrossFade Keyword 8.3.103.](#)
[FaderCrossFadeA Keyword 8.3.104.](#)
[FaderCrossFadeB keyword 8.3.105.](#)
[FaderMaster keyword 8.3.106.](#)
[FaderRate Keyword 8.3.107.](#)
[FaderSpeed Keyword 8.3.108.](#)
[FaderTemp Keyword 8.3.109.](#)
[FaderTime Keyword 8.3.110.](#)
[Fancy 13.11.](#)
[feature 13.15.2.](#)
[feature groups 13.15.2.](#)
[FeatureGroup Keyword 8.3.111.](#)
[FeatureGroupe 2.15.48.](#)
[File Management 49.](#)
[Filter 2.15.23. 25.](#)
[Filter - Create 25.3.](#)
[Filter - Delete 25.6.](#)
[filter - use 25.4.](#)
[Filter Keyword 8.3.112.](#)
[first steps 4.](#)
[first steps – connect Audio Remote In 4.9.](#)

[first steps – connect Ethernet 4.12.](#)
[first steps – connect external screens 4.6.](#)
[first steps – connect GPI 4.13.](#)
[first steps – connect grandMA3 extension 4.14.](#)
[first steps – connect LTC 4.11.](#)
[Fix 2.15.43.](#)
[Fix Keyword 8.3.113.](#)
[fixture 14.3.](#)
[fixture - marker 33.2.](#)
[fixture - patch 13.](#)
[fixture - setup 13.](#)
[fixture graphic 14.3.1.](#)
[Fixture key 2.15.22.](#)
[Fixture Keyword 8.3.116.](#)
[fixture selection 14.1.](#)
[fixture sheet 14.3. 14.3.1.](#)
[fixture types 48.](#)
[fixture types – build 48.2.](#)
[fixture types – export 48.3.](#)
[fixture types – import 48.1.](#)
[fixture types – import fixture types – import GDTF 48.1.1.](#)
[fixture types – insert 48.2.1.](#)
[fixture types – insert dmx modes 48.2.2.](#)
[fixture types – insert geometries 48.2.3.](#)
[fixture types – insert models 48.2.4.](#)
[fixture types – link DMX modes to geometries 48.2.6.](#)
[fixture types – link models to geometries 48.2.5.](#)
[FixtureClass Keyword 8.3.114.](#)
[FixtureLayer Keyword 8.3.115.](#)
[fixtures - add to show 13.2.](#)
[fixtures - delete 13.8.](#)
[fixtures - position in 3D 13.9.](#)
[fixtures - remove from show 13.8.](#)
[fixtures - what is it 13.1.](#)
[FixtureType Keyword 8.3.117.](#)
[Flash Keyword 8.3.119.](#)
[Flip Keyword 8.3.118.](#)



[Fog Keyword 8.3.120.](#)
[Folder Structure 49.3.](#)
[framework 2.14.1.](#)
[freeze 14.2.](#)
[Freeze key 2.15.20.](#)
[Freeze Keyword 8.3.121.](#)
[front panel 2.1.](#)
[Full key 2.15.21.](#)
[Full Keyword 8.3.122.](#)
[full package 47.](#)
[gel 2.15.48.](#)
[Gel 14.6.](#)
[Gel Keyword 8.3.123.](#)
[Gels 14.6.](#)
[gestures 7.2.](#)
[GetGlobalVariable Keyword 8.3.124.](#)
[GetUserVariable Keyword 8.3.125.](#)
[glossary 52.](#)
[Go- 2.15.28.](#)
[Go- \[large\] key 2.15.26.](#)
[Go- |Top key 2.15.28.](#)
[Go- Keyword 8.3.127.](#)
[Go+ \[large\] key 2.15.25.](#)
[Go+ | Temp button 2.15.27.](#)
[Go+ Keyword 8.3.126.](#)
[gobos 19.](#)
[Goto key 2.15.24.](#)
[Goto Keyword 8.3.128.](#)
[Grand 30.2.](#)
[grand master 2.16.6.](#)
[grandMA3 compact 2.1.6.](#)
[grandMA3 compact XT 2.1.5.](#)
[grandMA3 extension 2.2.](#)
[grandMA3 full-size 2.1. 2.1.1.](#)
[grandMA3 full-size CRV 2.1.2.](#)
[grandMA3 I/O Node DIN-Rail 2.8.](#)
[grandMA3 I/O Nodes 2.7.](#)



[grandMA3 light 2.1.3.](#)
[grandMA3 light CRV 2.1.4.](#)
[grandMA3 Network Switch 46.2.](#)
[grandMA3 Nodes 46.1.](#)
[grandMA3 Nodes DIN-Rail 2.6.](#)
[grandMA3 onPC 5.](#)
[grandMA3 onPC - macOS installation 5.4.](#)
[grandMA3 onPC - Optimize macOS 5.5.](#)
[grandMA3 onPC - Optimize Windows 5.3.](#)
[grandMA3 onPC – system requirements 5.1.](#)
[grandMA3 onPC - Windows installation 5.2.](#)
[grandMA3 onPC command wing 2.10.](#)
[grandMA3 onPC command wing XT 2.9.](#)
[grandMA3 onPC fader wing 2.11.](#)
[grandMA3 onPC rack-unit 2.12.](#)
[grandMA3 onPC Terminal App 5.6.](#)
[grandMA3 processing units 2.4.](#)
[grandMA3 replay unit 2.3.](#)
grandMA3 User Manual
[grandMA3 xPort Nodes 2.5.](#)
[Grid 2.15.66.](#)
[Grid Keyword 8.3.129.](#)
[GridPosition Keyword 8.3.130.](#)
[group 37.3.](#)
[Group key 2.15.23.](#)
[Group Keyword 8.3.131.](#)
[groups 23.](#)
[groups - create 23.1.](#)
[groups - delete 23.3.](#)
[groups - edit 23.2.](#)
[groups - master 23.4.](#)
[HalfSpeed Keyword 8.3.132.](#)
[help 1.1. 1.4.](#)
[help in the command line 1.1.](#)
[Help key 2.15.29.](#)
[Help Keyword 8.3.133.](#)
[help pages 1.2.](#)



[HelpLua Keyword 8.3.134.](#)
[Hex16 Keyword 8.3.136.](#)
[Hex24 Keyword 8.3.137.](#)
[Hex8 Keyword 8.3.135.](#)
[Highlight Keyword 8.3.138.](#)
[Hight \[Highlight\] key 2.15.30.](#)
[HookObjectChange 39.5.93.](#)
[Houselights Keyword 8.3.139.](#)
[Hz Keyword 8.3.140.](#)
[If key 2.15.31.](#)
[If Keyword 8.3.141.](#)
[IfActive 2.15.31.](#)
[IfActive Keyword 8.3.142.](#)
[IfOutput 2.15.31.](#)
[IfOutput Keyword 8.3.143.](#)
[IfProg 2.15.31.](#)
[IfProgrammer Keyword 8.3.144.](#)
[image 16.](#)
[image - delete 16.](#)
[image - import 16.](#)
[image - screenshot 16.1.](#)
[Image Keyword 8.3.145.](#)
[image pool 16.](#)
[Import 39.5.97. 49.2.](#)
[Import Keyword 8.3.146.](#)
[include dependencies 49.2.](#)
[index 52.](#)
[Index Keyword 8.3.147.](#)
[Info 42.6.](#)
[inner ring 2.16.4.](#)
[Insert 2.15.13. 2.15.36.](#)
[Insert Keyword 8.3.148.](#)
[installation packages 47.](#)
[integrate 2.15.7.](#)
[Integrate Keyword 8.3.149.](#)
[Interface Keyword 8.3.150.](#)
[introduction – about the help 1.](#)



[Invert Keyword 8.3.151.](#)
[Invite Keyword 8.3.152.](#)
[ip address 10.1.](#)
[IP Keyword 8.3.153.](#)
[JoinSession Keyword 8.3.154.](#)
[Key Keyword 8.3.155.](#)
[keyboard - connect 4.7.](#)
[Keyboard Keyword 8.3.156.](#)
[KeyboardShortcut Keyword 8.3.157.](#)
[Keys 2.15.](#)
[keys - keyboard shortcuts 2.14.](#)
[keys - shortcuts 2.14.](#)
[keyword help 1.1.](#)
[keywords - general 8.3.](#)
[Kill 2.15.28.](#)
[Kill Keyword 8.3.158.](#)
[KnockIn Keyword 8.3.159.](#)
[KnockOut Keyword 8.3.160.](#)
[Label 2.15.8.](#)
[Label Keyword 8.3.161.](#)
[layout 38.7.](#)
[Layout 2.15.67.](#)
[layout - create 38.1.](#)
[Layout Keyword 8.3.162.](#)
[layout view - settings 38.4.](#)
[Layouts 38.](#)
[Learn | Rate1 key 2.15.32.](#)
[LearnRate key 2.15.32.](#)
[LearnSpeed Keyword 8.3.163.](#)
[LeaveSession Keyword 8.3.164.](#)
[level wheel 2.16.5.](#)
[Library Keyword 8.3.165.](#)
[Link 2.15.65.](#)
[list 42.6.](#)
[List 2.15.33.](#)
[List key 2.15.33.](#)
[List Keyword 8.3.166.](#)



ListOwnership Keyword [8.3.167.](#)

ListRef [2.15.33.](#)

ListReference Keyword [8.3.168.](#)

live patch [13.5.](#)

Load [2.15.24.](#)

Load Keyword [8.3.169.](#)

load show [6.](#) [6.1.](#)

Loaded Keyword [8.3.170.](#)

LoadShow Keyword [8.3.171.](#)

Lock Keyword [8.3.172.](#)

LogIn Keyword [8.3.173.](#)

LogOut Keyword [8.3.174.](#)

Lowlight Keyword [8.3.175.](#)

Lua Keyword [8.3.176.](#)

LuaFile Keyword [8.3.177.](#)

MA key [2.15.34.](#)

macro [34.](#) [34.2.](#)

Macro [2.15.77.](#)

macro - assign [34.4.](#)

macro - create - import [34.1.](#)

macro - edit [34.3.](#)

macro - examples [34.6.](#)

Macro Keyword [8.3.178.](#)

marker [33.2.](#)

Marker [7.12.2.](#)

MArker Keyword [8.3.179.](#)

Master [30.](#) [30.1.](#) [30.2.](#) [30.3.](#) [30.4.](#)

master area [2.16.2.](#)

Master Keyword [8.3.180.](#)

Master Modes [30.](#)

Master Section [28.4.](#)

master slave [8.2.](#)

Masters [30.](#)

matricks [26.](#)

MATricks [2.15.54.](#)

MAtricks - blocks [26.1.](#)

MAtricks - groups [26.2.](#)



[MAtricks - shuffle 26.5.](#)
[MAtricks - width 26.4.](#)
[MAtricks - wings 26.3.](#)
[MAtricks Keyword 8.3.181.](#)
[MATricksBlocks 2.15.54.](#)
[MATricksInterleave 2.15.54.](#)
[Measure Keyword 8.3.182.](#)
[Media Keyword 8.3.183.](#)
[MemoryInfo keyword 8.3.184.](#)
[menu 9. 9.7.](#)
[menu - change location 9.8.](#)
[Menu key 2.15.35.](#)
[Menu keyword 8.3.185.](#)
[Mesh Keyword 8.3.186.](#)
[meshes 18.](#)
[MIB 27.11.](#)
[Midi 44.6.](#)
[MIDI 5.7.](#)
[MIDI Remotes 44.2.](#)
[mix color only 14.5.](#)
[modes 35.1.](#)
[more – clock 42.2.](#)
[more – system info 42.4.](#)
[move grid cursor 23.2.](#)
[move in black 27.11.](#)
[Move key 2.15.36.](#)
[Move Keyword 8.3.187.](#)
[multipatch 13.3.](#)
[Multipatch Keyword 8.3.188.](#)
[Multi-Select 2.15.14.](#)
[MVR 13.4.](#)
[my virtual rig 13.4.](#)
[MyRunningMacro Keyword 8.3.189.](#)
[MyRunningPreset Keyword 8.3.190.](#)
[MyRunningSequence Keyword 8.3.191.](#)
[MyRunningTimer Keyword 8.3.192.](#)
[Natural Keyword 8.3.193.](#)



[navigate in the help 1.4.](#)
[NDI 17.](#)
[NDI Keyword 8.3.194.](#)
[network 10.](#)
[Network - Design 10.4.](#)
[Network - Design - WLAN 10.4.2.](#)
[network - interface 10.1.](#)
[network - session - master selection 10.2.7.](#)
[NetworkNode Keyword 8.3.199.](#)
[new 6.3.](#)
[NewShow Keyword 8.3.195.](#)
[Next key 2.15.37.](#)
[Next Keyword 8.3.196.](#)
[Next Step 2.15.37.](#)
[NextY Keyword 8.3.197.](#)
[NextZ Keyword 8.3.198.](#)
[NonDim Keyword 8.3.200.](#)
[normal 2.15.7.](#)
[Normal Keyword 8.3.201.](#)
[Note Keyword 8.3.202.](#)
[notes 22.](#)
[Numeric keys | arrows 2.15.38.](#)
[Object ownership 12.3.](#)
[ObjectList 39.5.109.](#)
[Off key 2.15.41.](#)
[Off Keyword 8.3.203.](#)
[off menu 28.3.](#)
[Off menu 2.15.41.](#)
[Offset Keyword 8.3.204.](#)
[On key 2.15.40.](#)
[On Keyword 8.3.205.](#)
[onPC Keyword 8.3.206.](#)
[onPC package 47.](#)
[onPC settings 5.7.](#)
[Oops key 2.15.39.](#)
[Oops Keyword 8.3.208.](#)
[Open 2.15.21.](#)



[open stage control 44.4.4.](#)
[operate fixtures 14.](#)
[Option keywords 8.4.](#)
[order show file 6.5.](#)
[OSC 44.4.](#)
[OSC Keyword 8.3.207.](#)
[osc protocol viewer 44.4.6.](#)
[other – date and time 42.1.](#)
[outer ring 2.16.4.](#)
[OutputLayer Keyword 8.3.209.](#)
[packing material 4.1.](#)
[Page 2.15.78.](#)
[Page- key 2.15.45.](#)
[Page Keyword 8.3.210.](#)
[page pool 28.](#)
[Page+ key 2.15.44.](#)
[panic macro 51.4.](#)
[parameter 3.4. 3.4.1.](#)
[parameter - expansion 3.4.2.](#)
[parameter list 13.16.](#)
[parent child 8.2.](#)
[Park 2.15.43.](#)
[Park Keyword 8.3.212.](#)
[Part Keyword 8.3.211.](#)
[Paste 2.15.13.](#)
[Paste Keyword 8.3.213.](#)
[patch - add fixtures 13.2.](#)
[patch - classes 13.14.](#)
[patch - layers 13.14.](#)
[patch - stage 13.13.](#)
[Patch Keyword 8.3.214.](#)
[Pause 2.15.43.](#)
[Pause \[large\] key 2.15.42.](#)
[Pause | Fix key 2.15.43.](#)
[Pause Keyword 8.3.215.](#)
[pdf 1.3.](#)
[pdf help 1.3.](#)



[pdf manual 1.3.](#)
[Percent keyword 8.3.216.](#)
[PercentFine keyword 8.3.217.](#)
[Phase keyword 8.3.218.](#)
[Phaser 2.15.76.](#)
[phaser - circle around preset 32.4.](#)
[phaser - create color rainbow 32.5.](#)
[phaser - create dimmer phaser 32.2.](#)
[phaser - editor 32.1.](#)
[phaser - simple circle \(absolute values\) 32.3.](#)
[Phaser Editor 32.2.](#)
[Phaser Layers 32.](#)
[phasers 32.](#)
[phasers - what is it 32.](#)
[Physical keyword 8.3.219.](#)
[physical units 13.15.](#)
[Playback 30.4.](#)
[Playback Bar 7.10.](#)
[playback window 28.](#)
[Please key 2.15.46.](#)
[Plugin 2.15.77.](#)
[plugin - Lua - function - object - Addr 39.6.1.](#)
[plugin - Lua - function - object - AddrNative 39.6.2.](#)
[plugin - Lua - function - object - Children 39.6.3.](#)
[plugin - Lua - function - object - Dump 39.6.4.](#)
[plugin - Lua - function - object - Export 39.6.5.](#)
[plugin - Lua - function - object - GetAttributeByUIChannel 39.5.47.](#)
[plugin - Lua - function - object - GetChannelFunctionIndex 39.5.52.](#)
[plugin - Lua - function - object - GetClassDerivationLevel 39.5.53.](#)
[plugin - Lua - function - object - GetRTChannelCount 39.5.72.](#)
[plugin - Lua - function - object - GetUIChannelCount 39.5.85.](#)
[plugin - Lua - function - object - GetUIChannelIndex 39.5.86.](#)
[plugin - Lua - function - object - HasActivePlayback 39.6.6.](#)
[plugin - Lua - function - object - Import 39.6.7.](#)
[plugin - Lua - function - object - ToAddr 39.6.8.](#)
[plugin - Lua - function - objectfree - AddonVars 39.5.2.](#)
[plugin - Lua - function - objectfree - BuildDetails 39.5.3.](#)

plugin - Lua - function - objectfree - ClassExists [39.5.6.](#)
plugin - Lua - function - objectfree - CloseAllOverlays [39.5.7.](#)
plugin - Lua - function - objectfree - CloseUndo [39.5.8.](#)
plugin - Lua - function - objectfree - Cmd [39.5.9.](#)
plugin - Lua - function - objectfree - CmdIndirect [39.5.10.](#)
plugin - Lua - function - objectfree - CmdIndirectWait [39.5.11.](#)
plugin - Lua - function - objectfree - CmdObj [39.5.12.](#)
plugin - Lua - function - objectfree - ConfigTable [39.5.15.](#)
plugin - Lua - function - objectfree - Confirm [39.5.16.](#)
plugin - Lua - function - objectfree - CreateUndo [39.5.19.](#)
plugin - Lua - function - objectfree - CurrentEnvironment [39.5.20.](#) [39.5.21.](#)
plugin - Lua - function - objectfree - CurrentProfile [39.5.22.](#)
plugin - Lua - function - objectfree - CurrentUser [39.5.23.](#)
plugin - Lua - function - objectfree - DataPool [39.5.24.](#)
plugin - Lua - function - objectfree - DefaultDisplayPositions [39.5.25.](#)
plugin - Lua - function - objectfree - DelVar [39.5.26.](#)
plugin - Lua - function - objectfree - DeskLocked [39.5.27.](#)
plugin - Lua - function - objectfree - DeviceConfiguration [39.5.28.](#)
plugin - Lua - function - objectfree - DrawPointer [39.5.31.](#)
plugin - Lua - function - objectfree - Echo [39.5.33.](#)
plugin - Lua - function - objectfree - ErrEcho [39.5.34.](#)
plugin - Lua - function - objectfree - ErrPrintf [39.5.35.](#)
plugin - Lua - function - objectfree - Export [39.5.36.](#)
plugin - Lua - function - objectfree - ExportCSV [39.5.37.](#)
plugin - Lua - function - objectfree - ExportJson [39.5.38.](#)
plugin - Lua - function - objectfree - FromAddr [39.5.45.](#)
plugin - Lua - function - objectfree - GetApiDescriptor [39.5.46.](#)
plugin - Lua - function - objectfree - GetAttributeCount [39.5.48.](#)
plugin - Lua - function - objectfree - GetAttributeIndex [39.5.49.](#)
plugin - Lua - function - objectfree - GetChannelFunction [39.5.51.](#)
plugin - Lua - function - objectfree - GetDisplayByIndex [39.5.54.](#)
plugin - Lua - function - objectfree - GetDisplayCollect [39.5.55.](#)
plugin - Lua - function - objectfree - GetDMXUniverse [39.5.56.](#)
plugin - Lua - function - objectfree - GetDMXValue [39.5.57.](#)
plugin - Lua - function - objectfree - GetExecutor [39.5.59.](#)
plugin - Lua - function - objectfree - GetFocus [39.5.60.](#)
plugin - Lua - function - objectfree - GetFocusDisplay [39.5.61.](#)



plugin - Lua - function - objectfree - GetObjApiDescriptor [39.5.62.](#)
plugin - Lua - function - objectfree - GetPath [39.5.63.](#)
plugin - Lua - function - objectfree - GetPathSeparator [39.5.65.](#)
plugin - Lua - function - objectfree - GetRTChannel [39.5.71.](#)
plugin - Lua - function - objectfree - GetRTChannels [39.5.73.](#)
plugin - Lua - function - objectfree - GetSample [39.5.75.](#)
plugin - Lua - function - objectfree - GetSelectedAttribute [39.5.76.](#)
plugin - Lua - function - objectfree - GetShowFileStatus [39.5.77.](#)
plugin - Lua - function - objectfree - GetSubfixture [39.5.78.](#)
plugin - Lua - function - objectfree - GetSubfixtureCount [39.5.79.](#)
plugin - Lua - function - objectfree - GetTokenName [39.5.80.](#)
plugin - Lua - function - objectfree - GetTokenNameByIndex [39.5.81.](#)
plugin - Lua - function - objectfree - GetUIChannels [39.5.87.](#)
plugin - Lua - function - objectfree - GetVar [39.5.89.](#)
plugin - Lua - function - objectfree - GlobalVars [39.5.90.](#)
plugin - Lua - function - objectfree - HandleToInt [39.5.91.](#)
plugin - Lua - function - objectfree - HandleToStr [39.5.92.](#)
plugin - Lua - function - objectfree - HostOS [39.5.94.](#)
plugin - Lua - function - objectfree - HostSubType [39.5.95.](#)
plugin - Lua - function - objectfree - HostType [39.5.96.](#)
plugin - Lua - function - objectfree - IncProgress [39.5.98.](#)
plugin - Lua - function - objectfree - IntToHandle [39.5.99.](#)
plugin - Lua - function - objectfree - IsObjectValid [39.5.101.](#)
plugin - Lua - function - objectfree - KeyboardObj [39.5.103.](#)
plugin - Lua - function - objectfree - masterpool [39.5.105.](#)
plugin - Lua - function - objectfree - MessageBox [39.5.106.](#)
plugin - Lua - function - objectfree - MouseObj [39.5.108.](#)
plugin - Lua - function - objectfree - Patch [39.5.111.](#)
plugin - Lua - function - objectfree - Printf [39.5.115.](#)
plugin - Lua - function - objectfree - Pult [39.5.118.](#)
plugin - Lua - function - objectfree - ReleaseType [39.5.120.](#)
plugin - Lua - function - objectfree - Root [39.5.122.](#)
plugin - Lua - function - objectfree - SelectedSequence [39.5.124.](#)
plugin - Lua - function - objectfree - Selection [39.5.125.](#)
plugin - Lua - function - objectfree - SelectionCount [39.5.129.](#)
plugin - Lua - function - objectfree - SelectionFirst [39.5.130.](#)
plugin - Lua - function - objectfree - SelectionNext [39.5.131.](#)

[plugin - Lua - function - objectfree - SerialNumber 39.5.135.](#)
[plugin - Lua - function - objectfree - SetBlockInput 39.5.136.](#)
[plugin - Lua - function - objectfree - SetProgress 39.5.141.](#)
[plugin - Lua - function - objectfree - SetProgressRange 39.5.142.](#)
[plugin - Lua - function - objectfree - SetProgressText 39.5.143.](#)
[plugin - Lua - function - objectfree - SetVar 39.5.144.](#)
[plugin - Lua - function - objectfree - showdata 39.5.145.](#)
[plugin - Lua - function - objectfree - ShowSettings 39.5.146.](#)
[plugin - Lua - function - objectfree - StartProgress 39.5.147.](#)
[plugin - Lua - function - objectfree - StopProgress 39.5.148.](#)
[plugin - Lua - function - objectfree - StrToHandle 39.5.149.](#)
[plugin - Lua - function - objectfree - TextInput 39.5.151.](#)
[plugin - Lua - function - objectfree - Time 39.5.152.](#)
[plugin - Lua - function - objectfree - TouchObj 39.5.156.](#)
[plugin - Lua - function - objectfree - UserVars 39.5.159.](#)
[plugin - Lua - function - objectfree - Version 39.5.160.](#)
[plugin - Lua - handle 39.2.](#)
[plugin - Lua - interface 39.3.](#)
[plugin - Lua - interface - ProgressBar 39.3.1.](#)
[Plugin - Lua - Object API 39.6.](#)
[Plugin - Lua - Object-Free API 39.5.](#)
[plugin - Lua - variables 39.4.](#)
[plugin - programmerpart 39.5.117.](#)
[Plugin keyword 8.3.220.](#)
[Plugins 39.](#)
[plugins - lua 39.1.](#)
[plugins - what is lua 39.1.](#)
[pool - create 9.9.1.](#)
[pool - general information 9.9.](#)
[pool - select object 9.9.](#)
[pool - settings 9.5.](#)
[pool - swipeys 9.9.](#)
[pool object - copy 9.9.5.](#)
[pool object - delete 9.9.7.](#)
[pool object - insert 9.9.4.](#)
[pool object - label 9.9.2.](#)
[pool object - lock 9.9.6.](#)



[pool object - move 9.9.3.](#)
[pool object - unlock 9.9.6.](#)
[PopupInput 39.5.113.](#)
[position 38.7.](#)
[power - connect 4.4.](#)
[Power key 2.15.47.](#)
[powerCON TRUE1 4.4.](#)
[preset - recipe 24.3.](#)
[Preset key 2.15.48.](#)
[Preset keyword 8.3.221.](#)
[presets 24.](#)
[presets – create 24.2.](#)
[presets - edit 24.5.](#)
[presets – preset pools 24.1.](#)
[presets - update 24.5.](#)
[presets - use 24.4.](#)
[PresetUpdate keyword 8.3.222.](#)
[Press Keyword 8.3.223.](#)
[Prev 2.15.50.](#)
[Prev \[Previous\] key 2.15.50.](#)
[preview 14.2.](#)
[Preview Executor 2.15.27.](#)
[Preview key 2.15.49.](#)
[Preview Keyword 8.3.224.](#)
[PreviewEdit 2.15.49.](#)
[Previous 2.15.50.](#)
[Previous keyword 8.3.225.](#)
[PreviousY Keyword 8.3.226.](#)
[PreviousZ Keyword 8.3.227.](#)
[ProcessingUnit Keyword 8.3.230.](#)
[Program Time 30.2.1.](#)
[programmer 14.2.](#)
[Programmer 2.15.15.](#)
[Programmer keyword 8.3.228.](#)
[Property Keyword 8.3.229.](#)
[protocols 10.4.1.](#)
[Prvw \[Preview\] key 2.15.49.](#)



Prvw key [2.15.49](#).
PSN [44.5](#).
Pyro keyword [8.3.231](#).
qlab [44.4.3](#).
Quality [13.11](#).
quickey [40.2](#).
Quickey [2.15.77](#).
Quickey Keyword [8.3.232](#).
quickeys [40](#). [40.1](#).
quickeys example [40.3](#).
Rate1 key [2.15.32](#).
Rate1 keyword [8.3.233](#).
RDM [45](#).
RDM keyword [8.3.234](#).
Readout keyword [8.3.235](#).
RealtimeChannel keyword [8.3.251](#).
Reboot keyword [8.3.236](#).
Recast Keyword [8.3.237](#).
recipe - cue [27.8](#).
recipes [31](#).
Record Keyword [8.3.238](#).
red fixture sheet [14.2](#).
red lock [12.3](#).
red window [14.2](#).
Redo [2.15.39](#).
reference [42.6](#).
Regulations and Standards [10.5](#).
Relative [2.15.59](#).
Relative Keyword [8.3.239](#).
release [2.15.17](#).
Release Keyword [8.3.240](#).
Release Notes [4.16](#).
ReloadAllPlugins keyword [8.3.241](#).
ReloadUI keyword [8.3.242](#).
Remote Device Management [45](#).
remote inputs [44](#).
Remote keyword [8.3.243](#).



RemoteCommand keyword [8.3.245.](#)
RemoteHID [46.3.](#)
RemoteHID keyword [8.3.244.](#)
remove [2.15.17.](#)
Remove Keyword [8.3.246.](#)
Render Quality [13.11.](#)
RenderQuality keyword [8.3.247.](#)
Reset keyword [8.3.248.](#)
Restart keyword [8.3.249.](#)
Root keyword [8.3.250.](#)
running playbacks [28.3.](#)
Running Timer Keyword [8.3.256.](#)
RunningMacro keyword [8.3.252.](#)
RunningPreset keyword [8.3.253.](#)
RunningSequence keyword [8.3.254.](#)
RunningTimecode keyword [8.3.255.](#)
sACN menu [11.2.2.](#)
Sample [2.15.20.](#)
save [6.2.](#)
save show [6.](#)
SaveShow keyword [8.3.257.](#)
scope of delivery [4.2.](#)
screen [2.13.](#)
Screen Layout [7.1.1.](#)
ScreenConfiguration keyword [8.3.258.](#)
ScreenContent [2.15.70.](#)
ScreenContent keyword [8.3.259.](#)
screenshots [16.1.](#)
Scribble [2.15.67.](#)
Scribble keyword [8.3.260.](#)
scribbles [15.](#)
scribbles – assign [15.3.](#)
scribbles - create [15.1.](#)
scribbles - delete [15.4.](#)
search [1.4.](#)
Seconds keyword [8.3.261.](#)
Select key [2.15.51.](#)



Select keyword [8.3.262](#).
Selected [30.1](#).
select-fixtures [14.1](#).
SelectFixtures keyword [8.3.264](#).
selection [2.15.22](#).
selection bar [14.7](#).
selection grid [14.9](#).
Selection keyword [8.3.263](#).
Selecton [2.15.51](#).
SelFix [SelectFixture] key [2.15.52](#).
SendMIDI keyword [8.3.266](#).
SendOSC keyword [8.3.267](#).
Sequ [2.15.53](#).
Sequ [Sequence] key [2.15.53](#).
Sequence [2.15.53](#).
sequence - settings [27.4](#).
sequence edit toolbar [27.2](#).
Sequence keyword [8.3.265](#).
sequence sheet [27.2](#).
sequences [27](#).
session [10.2](#).
session - create [10.2.1](#).
session - dismiss [10.2.5](#).
session - invite [10.2.4](#).
session - join [10.2.2](#).
session - keys [10.2.6](#).
session - leave [10.2.3](#).
Set [2.15.8](#).
Set key [2.15.54](#).
Set keyword [8.3.268](#).
SetGlobalVariable keyword [8.3.269](#).
SetUserVariable keyword [8.3.270](#).
SFTP connection [49.1](#).
showfile [6](#).
shows [6.4](#).
Shuffle Keyword [8.3.271](#).
Shut down the System [50](#).



ShutDown Keyword [8.3.272.](#)
Sinus Dimmer Phaser [32.2.](#)
smart view [14.10.](#)
SnapDelay Keyword [8.3.273.](#)
SoftwareImport keyword [8.3.274.](#)
SoftwareUpdate keyword [8.3.275.](#)
Solo key [2.15.55.](#)
Solo keyword [8.3.277.](#)
Sound [43.](#)
Sound Window [43.](#)
SoundChannel keyword [8.3.276.](#)
space [13.13.](#)
Special Executor [28.4.](#)
SpecialExec keyword [8.3.278.](#)
Speed [30.3.](#)
Speed keyword [8.3.279.](#)
Speed1 keyword [8.3.280.](#)
SpeedMaster Keyword [8.3.281.](#)
sribbles - edit [15.2.](#)
stage [13.13.](#)
stage - space [13.13.](#)
Stage keyword [8.3.282.](#)
Station does not connect [51.3.](#)
Station keyword [8.3.283.](#)
StationSettings Keyword [8.3.284.](#)
Step [2.15.68.](#)
Step keyword [8.3.285.](#)
Steps [32.](#)
Stomp [32.](#)
Stomp key [2.15.56.](#)
Stomp keyword [8.3.286.](#)
store - options [27.9.](#)
store - preferences [27.9.](#)
Store key [2.15.57.](#)
Store keyword [8.3.287.](#)
subfixtures [14.1.](#)
sub-fixtures [14.1.](#)



Swap keyword [8.3.290](#).
Swatch Book [14.6](#).
SwitchTograndMA2Software keyword [8.3.288](#).
SwitchTograndMA3Software Keyword [8.3.289](#).
symbol [20.1](#). [20.2](#).
symbol-delete [20.2](#).
symbol-import [20.2](#).
symbol-pool [20.1](#). [20.2](#).
symbols [20](#).
Sync [32](#).
system [42](#).
system - local network system [3.2](#).
system - overview [3](#).
system - parameter count [3.4.1](#).
system - standalone device [3.1](#).
system monitor [42.5](#).
TC [2.15.69](#).
Temp keyword [8.3.291](#).
Tester encoder bar [13.6](#).
third-party software agreement [4.16](#).
Thru key [2.15.58](#).
Thru keyword [8.3.292](#).
Time Control [30.2.1](#).
Time key [2.15.59](#).
Time keyword [8.3.293](#).
time ranges-events [37.4](#).
timecode [37](#).
Timecode [2.15.69](#).
timecode – edit a timecode show [37.8](#).
timecode – record a timecode show [37.6](#).
timecode – record an external timecode show [37.7](#).
timecode – settings [37.2](#).
timecode – toggle view mode [37.5](#).
timecode – what are timecode slots [37.1](#).
Timecode keyword [8.3.294](#).
TimecodeSlot keyword [8.3.295](#).
timer [36.1](#).



[Timer 39.5.153.](#)
[timer - countdown 36.2.](#)
[timer - stopwatch 36.1.](#)
[timer keyword 8.3.296.](#)
[timers 36.](#)
[Title Bar Configuration 9.6.](#)
[Toggle keyword 8.3.297.](#)
[toolbar 35.4.](#)
[Top 2.15.28.](#)
[Top keyword 8.3.299.](#)
[TopUp keyword 8.3.298.](#)
[touch osc 44.4.2.](#)
[track 37.3.](#)
[tracking 27.1.](#)
[tracking - distance 27.1.](#)
[tracking distance 27.1.1.](#)
[trackpad 7.1.8.](#)
[Transform 26.6.](#)
[Transition keyword 8.3.300.](#)
[troubleshooting 51.](#)
[Troubleshooting Update Process 47.7.](#)
[turn on the console 4.16.](#)
[type keyword 8.3.301.](#)
[U1 key 2.15.62.](#)
[U2 key 2.15.63.](#)
[UIChannel 8.3.302.](#)
[UIGridSelection keyword 8.3.303.](#)
[Unblock Keyword 8.3.304.](#)
[undo 7.4.](#)
[Universal keyword 8.3.305.](#)
[Unlock Keyword 8.3.306.](#)
[Unpark 2.15.27.](#)
[Unpark Keyword 8.3.307.](#)
[Unpress Keyword 8.3.308.](#)
[Up key 2.15.60.](#)
[Up keyword 8.3.309.](#)
[update cue 27.6.](#)



[Update does not work 51.2.](#)
[update files 47.6.](#)
[Update grandMA3 command wing XT 47.3.](#)
[update grandMA3 consoles 47.1.](#)
[Update grandMA3 viz-key 47.4.](#)
[update grandMA3 xport nodes 47.2.](#)
[Update key 2.15.61.](#)
[Update keyword 8.3.310.](#)
[update the software 47.](#)
[Update the software - network update 47.5.](#)
[UpdateContent Keyword 8.3.311.](#)
[ups battery 2.18.](#)
[USB - connect 4.7.](#)
[user - rights 12.2.](#)
[user - create 12.1.](#)
[user - edit 12.1.](#)
[user - multi user setup 12.](#)
[user - profile settings 12.2.](#)
[user - settings 12.2.](#)
[user - single user setup 12.](#)
[User keyword 8.3.314.](#)
[User1 keyword 8.3.312.](#)
[User2 keyword 8.3.313.](#)
[UserProfile keyword 8.3.315.](#)
[Value 2.15.59.](#)
[Variables 34.5.](#)
[Version keyword 8.3.317.](#)
[video 17.](#)
[Video Keyword 8.3.316.](#)
[video pool 17.](#)
[view 9. 9.3.](#)
[View 2.15.70.](#)
[view - assign to view button 9.3.](#)
[view - recall a view 9.3.](#)
[view - store a view 9.3.](#)
[view - update 9.3.](#)
[View keyword 8.3.318.](#)



[view modes 35.1.](#)
[view pool 9.3.](#)
[ViewButton 2.15.70.](#)
[Viewbutton keyword 8.3.319.](#)
[Visualizer 13.10.](#)
[web 1.2.](#)
[web help 1.2.](#)
[web remote 10.3.](#)
[wheel mix mode 14.5.](#)
[Width keyword 8.3.320.](#)
[window 9.](#)
[window - adding window 9.1.](#)
[Window - At Filter 25.1.](#)
[window - move 9.2.](#)
[window - pools 9.9.](#)
[window - rearrange 9.2.](#)
[window - remove 9.4.](#)
[window - resize 9.2.](#)
[window - settings 9.5.](#)
[work space – encoder bar – encoder toolbar 7.8.2.](#)
[workspace 7.](#)
[workspace – allocation of displays 7.7.](#)
[workspace – calculator 7.9.](#)
[workspace – command area 7.3.](#)
[workspace – encoder bar 7.8.](#)
[workspace – encoder bar – feature group control bar 7.8.1.](#)
[workspace – master controls 7.5.](#)
[workspace – playback controls 7.6.](#)
[workspace – tables in general 7.1.7.](#)
[workspace – user interface 7.1.](#)
[workspace – user interface – command line 7.1.4.](#)
[workspace – user interface – control bar 7.1.5.](#)
[workspace – user interface – user-defined area 7.1.3.](#)
[workspace – user interface – view buttons 7.1.6.](#)
[World 2.15.23. 25.](#)
[world - create 25.2.](#)
[world - delete 25.5.](#)



[world - use 25.4.](#)
[World keyword 8.3.321.](#)
[World Server 3.3.](#)
[X1 2.15.64.](#)
[X1 | Clone key 2.15.64.](#)
[X10 key 2.15.73.](#)
[X11 key 2.15.74.](#)
[X12 key 2.15.75.](#)
[X13 2.15.76.](#)
[X13 | Phaser key 2.15.76.](#)
[X14 2.15.77.](#)
[X14 | Macro key 2.15.77.](#)
[X15 2.15.78.](#)
[X15 | Page key 2.15.78.](#)
[X16 2.15.79.](#)
[X16 | Exec key 2.15.79.](#)
[X2 2.15.65.](#)
[X2 | Link key 2.15.65.](#)
[X3 2.15.66.](#)
[X3 | Grid key 2.15.66.](#)
[X4 2.15.67.](#)
[X4 | Layout key 2.15.67.](#)
[X5 2.15.68.](#)
[X5 | Step key 2.15.68.](#)
[X6 2.15.69.](#)
[X6 | TC key 2.15.69.](#)
[X7 2.15.70.](#)
[X7 | View key 2.15.70.](#)
[X8 2.15.71.](#)
[X8 | DMX key 2.15.71.](#)
[X9 key 2.15.72.](#)
[XKeys key 2.15.80.](#)
[XYZ 33.](#)
[XYZ - active for fixture type 33.1.](#)
[yellow fixture sheet 14.2.](#)
[yellow window 14.2.](#)
[zactrack 44.4.5.](#)



zero [2.15.1.](#)

Zero keyword [8.3.322.](#)