CueServer 2 User's Manual

Manual

1.0.8 — Last update: 2015/04/28

Interactive Technologies, Inc.

Table of Contents

Getting Started

Welcome to CueServer 2.

This User’s Manual is under construction.

New versions of this document are being published weekly.

Current Version

Note that on Monday, April 27, version **1.0.8** of CueServer Studio was released. Please download this latest version here:

- http://interactive-online.com/products/cueserver/downloads

CueServer Studio can be downloaded as a .dmg file for Macs and a .zip file for Windows.

Whenever you update to a new version of CueServer Studio, it is likely that you will also need to update the firmware in your CueServer. If a firmware update is needed, a yellow caution icon (⚠) will appear next to the CueServer’s firmware version in the Navigator window. To update your CueServer, choose the *Update Firmware…* menu command in the *CueServer* menu to update your device.

CueServer Studio 2

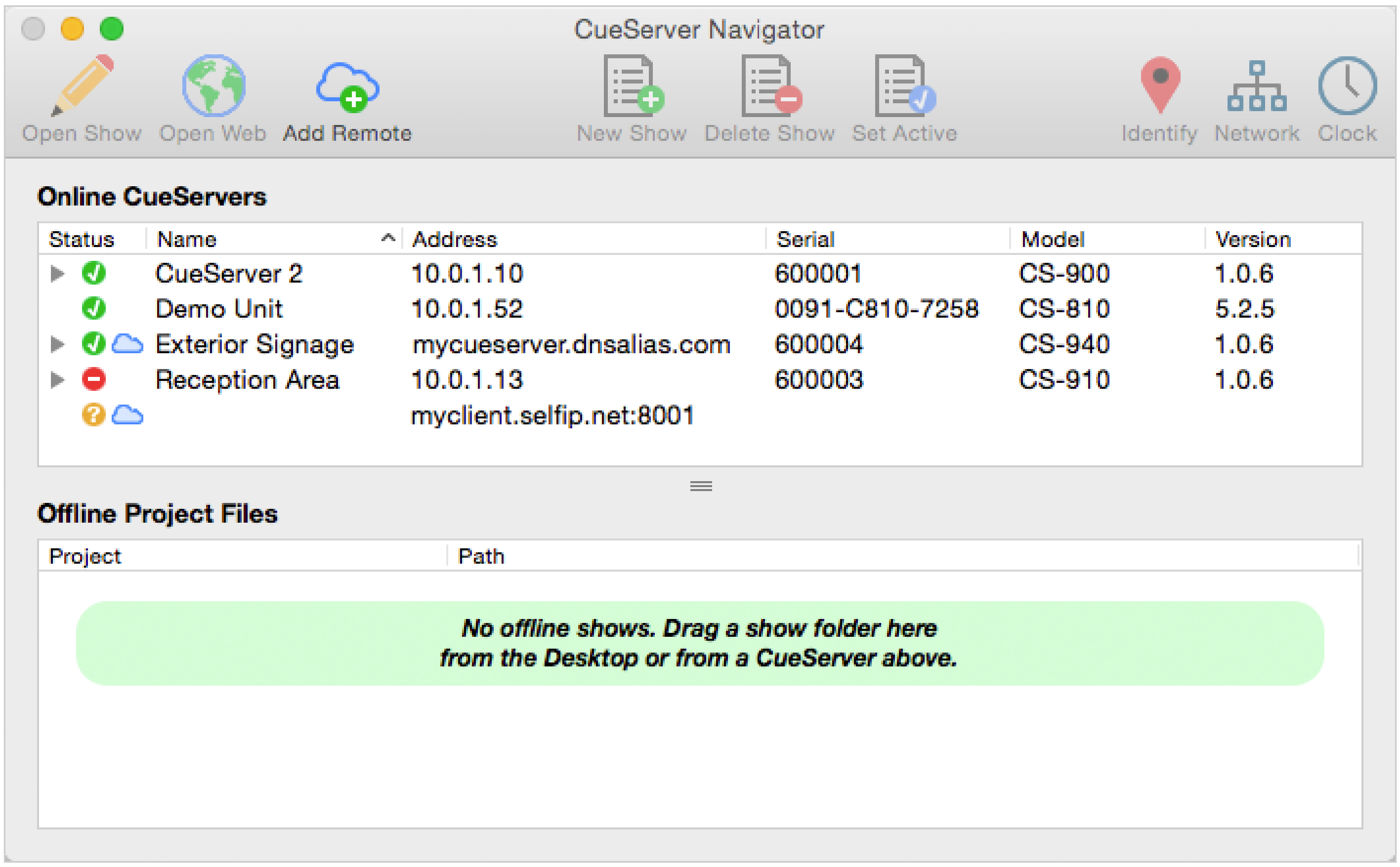
CueServer Studio 2 is the desktop application used to program, configure, locate and operate CueServer 2 devices. It is available for both Mac OS X and Windows. You can download the current version of CueServer Studio 2 here:

- http://interactive-online.com/products/cueserver/downlaods

Navigator Window

Overview

The *Navigator Window* appears when CueServer Studio opens. Use the Navigator Window to view available CueServers, manage basic settings, change active shows, identify individual devices, update firmware and more.



Status	Name	Address	Serial	Model	Version
✓	CueServer 2	10.0.1.10	600001	CS-900	1.0.6
✓	Demo Unit	10.0.1.52	0091-C810-7258	CS-810	5.2.5
✓	Exterior Signage	mycueserver.dnsalias.com	600004	CS-940	1.0.6
⛔	Reception Area	10.0.1.13	600003	CS-910	1.0.6
?		myclient.selfip.net:8001			

Project	Path

No offline shows. Drag a show folder here from the Desktop or from a CueServer above.

The top pane of this window displays both local and remote CueServers along with their online status, name, address, model and firmware version. The bottom pane is used for working with offline project files.

Working With Online CueServer Devices

The Navigator Window constantly scans the local network and displays any CueServers that are available. These devices will automatically appear in the upper list and will have a green status icon (✓).

Remote CueServers can also be added to the upper list manually. These CueServers will appear with a cloud icon (☁) as part of their status. See Working With Remote CueServers for more information.

The Status column shows various icons depending on the current state of a device in the list:

- ✓ The CueServer is online.
- ? The CueServer is being contacted.
- ⛔ The CueServer is offline.

Editing Online CueServers

Double clicking a CueServer or clicking on the Open Show icon (✎) opens that CueServer's Editor Window, which is used to program and configure the CueServer. See the Editor Window section for more information.

Opening the listbox under a CueServer reveals the available and active show file in the CueServer. Options are available to manage the active show, and to create new, delete and rename shows. See Working With Shows for more information.

Working With Offline Show Files

The bottom pane of the window is used as a working area to hold offline show files.

This pane makes it easy to open and edit show files that are on the local computer, or to copy shows between a CueServer and the local computer.

See the section on Working With Offline Shows for more details.

Setting Network or Clock Parameters

When a CueServer is selected, its Network and Clock parameters can be set using options from the CueServer menu, or by right-clicking (or control-clicking) the CueServer to get a contextual menu. Also, a Network button (品) and a Clock button (🕒) are available in the toolbar for easy access to these functions.

See the sections on Setting Network Parameters or Setting Clock Parameters for more information.

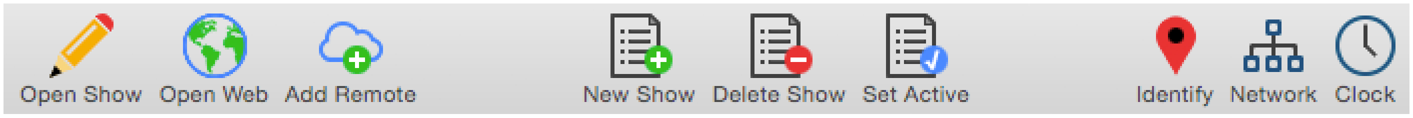
Maintenance

If the firmware of a CueServer is out-of-date, a warning icon (⚠) will appear next to its firmware version. See the Updating Firmware section for details.

If there are multiple CueServers on the network at the same time, it can sometimes be useful to identify which CueServer is which. See the Identifying CueServers section to learn how to activate the Identify function.

Toolbar

The toolbar in the Navigator Window contains several controls for managing CueServers.



Each of the toolbar items are described below:

Open Show
Opens the currently selected CueServer's Editor Window. The Editor Window is used for programming and configuration of a CueServer.

Open Web
Opens the currently selected CueServer's web page in the default web browser. If the **Option** / **Alt** key is held down when clicking this item, a Telnet session is opened to the currently selected CueServer.

Add Remote
Displays a dialog window that allows a remote CueServer to be added to the Navigator Window. This option is used to add CueServers that are not available on the local network, and are published on the Internet via a router's port-forwarding settings. See Working With Remote CueServers for more information.

New Show
Creates a new show file for the selected CueServer.

Delete Show
Removes the selected show file from a CueServer. Please note that the currently active show file cannot be deleted.

Set Active
Makes the selected show file the *active show*. The active show appears in the list in bold with a blue checkmark besides it.

Identify
Activates the selected CueServer's *Identify Mode*. When a CueServer is in Identify Mode, it's LCD

Display and Power LED will flash. Use this feature to help identify which CueServer is which in a complicated setup with multiple CueServer devices. See Identifying CueServers for more information.

Network
Displays a dialog window that allows the network settings of the selected CueServer to be changed. Use this option to change the IP Address, DHCP setting, and Device Name of a CueServer.

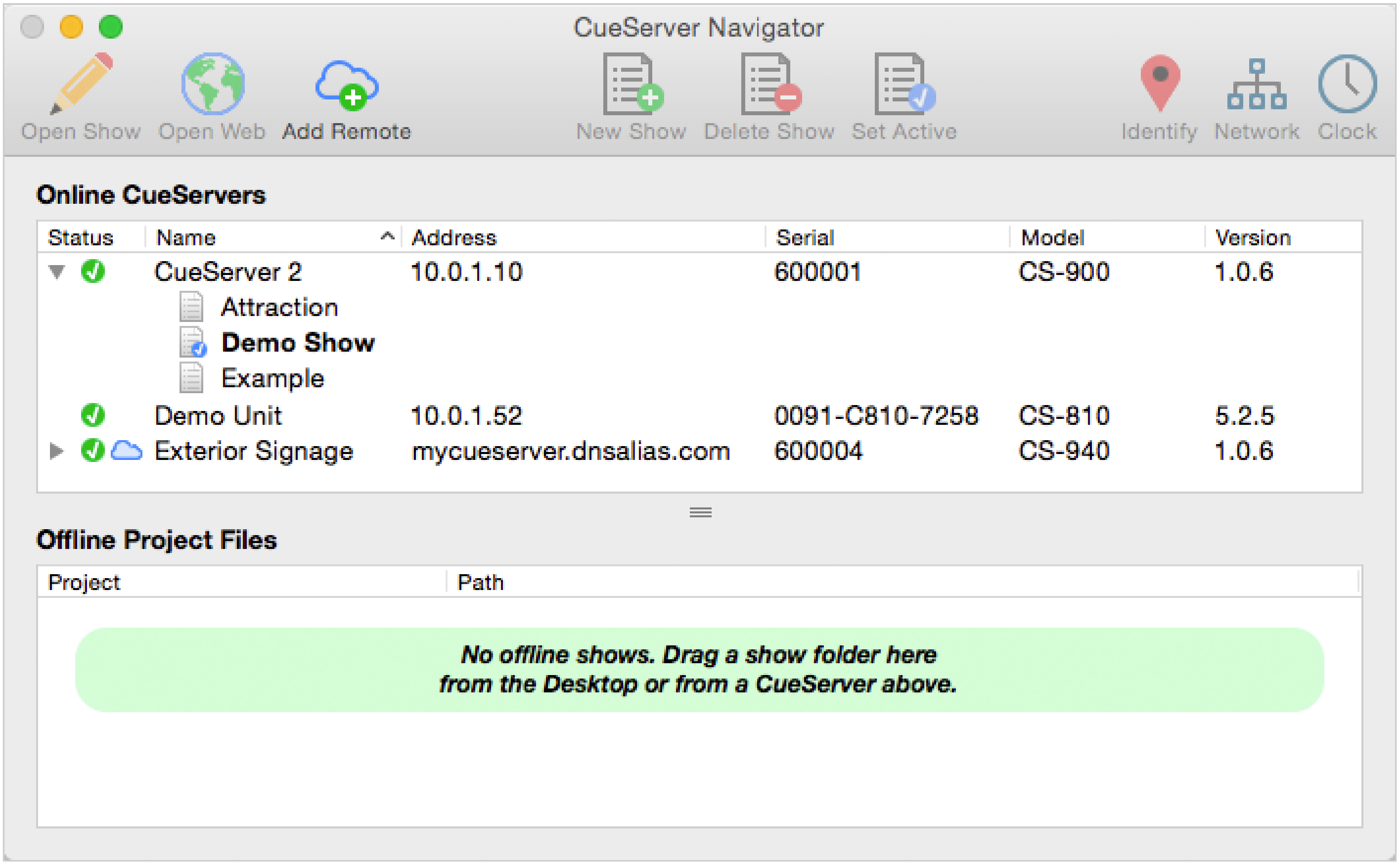
Clock
Displays a dialog window that allows the clock settings of the selected CueServer to be changed. Use this option to change the time zone, automatic and/or manual time settings of a CueServer.

Working With Shows

About Shows

All of the programming and configuration in a CueServer is stored in a *show file*. CueServer show files contain Cues, Groups, Macros, Sounds, Web Pages, Stations, Timers, Rules, Configuration Data and more. The memory card in CueServer can hold one or more show files, however only one show can be active at a time.

The shows available on a CueServer's memory card are displayed by opening the hierarchical list under the CueServer in the Navigator Window.



Status	Name	Address	Serial	Model	Version
▼ ✓	CueServer 2	10.0.1.10	600001	CS-900	1.0.6
	Attraction				
	Demo Show				
	Example				
✓	Demo Unit	10.0.1.52	0091-C810-7258	CS-810	5.2.5
▶ ✓	Exterior Signage	mycueserver.dnsalias.com	600004	CS-940	1.0.6

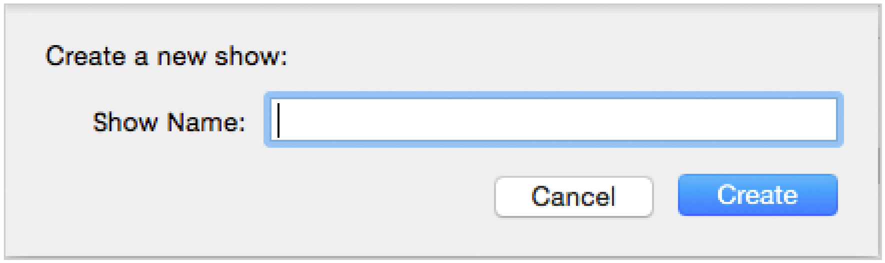
Project	Path

No offline shows. Drag a show folder here from the Desktop or from a CueServer above.

In the above example, the device named CueServer 2 contains three shows. The show marked in bold and with the blue checkmark icon ( ) next to it is the currently active show in the CueServer.

Creating a New Show

To create a new show, click on the New Show toolbar item ( ). A window will appear asking for a new show name:


Create a new show:

Show Name:

Enter a unique show name and press **Create** to create the new show.

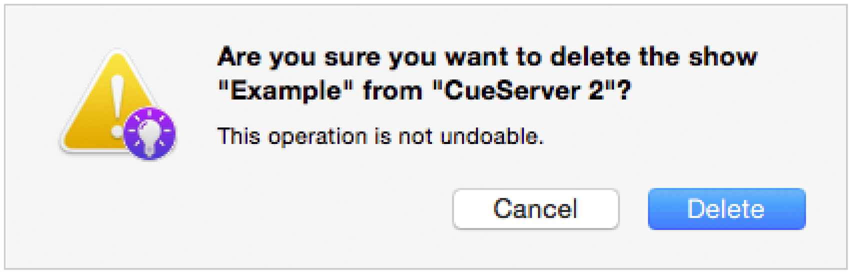
Changing the Active Show

To change the currently active show, click on a show file and then choose the **Set Active Show** menu item or click on the Set Active toolbar item ( ).

Deleting a Show

To delete a show, click on the show file and then click on the Delete Show toolbar item ( ).

A confirmation dialog will appear:


Are you sure you want to delete the show "Example" from "CueServer 2"?

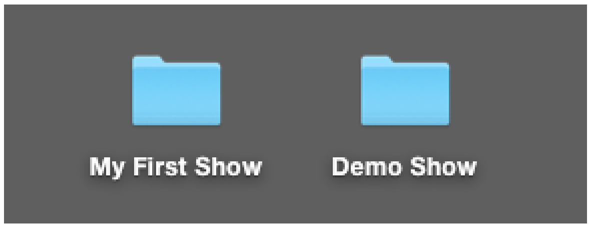
This operation is not undoable.

To proceed with deleting the show, choose the **Delete** button.

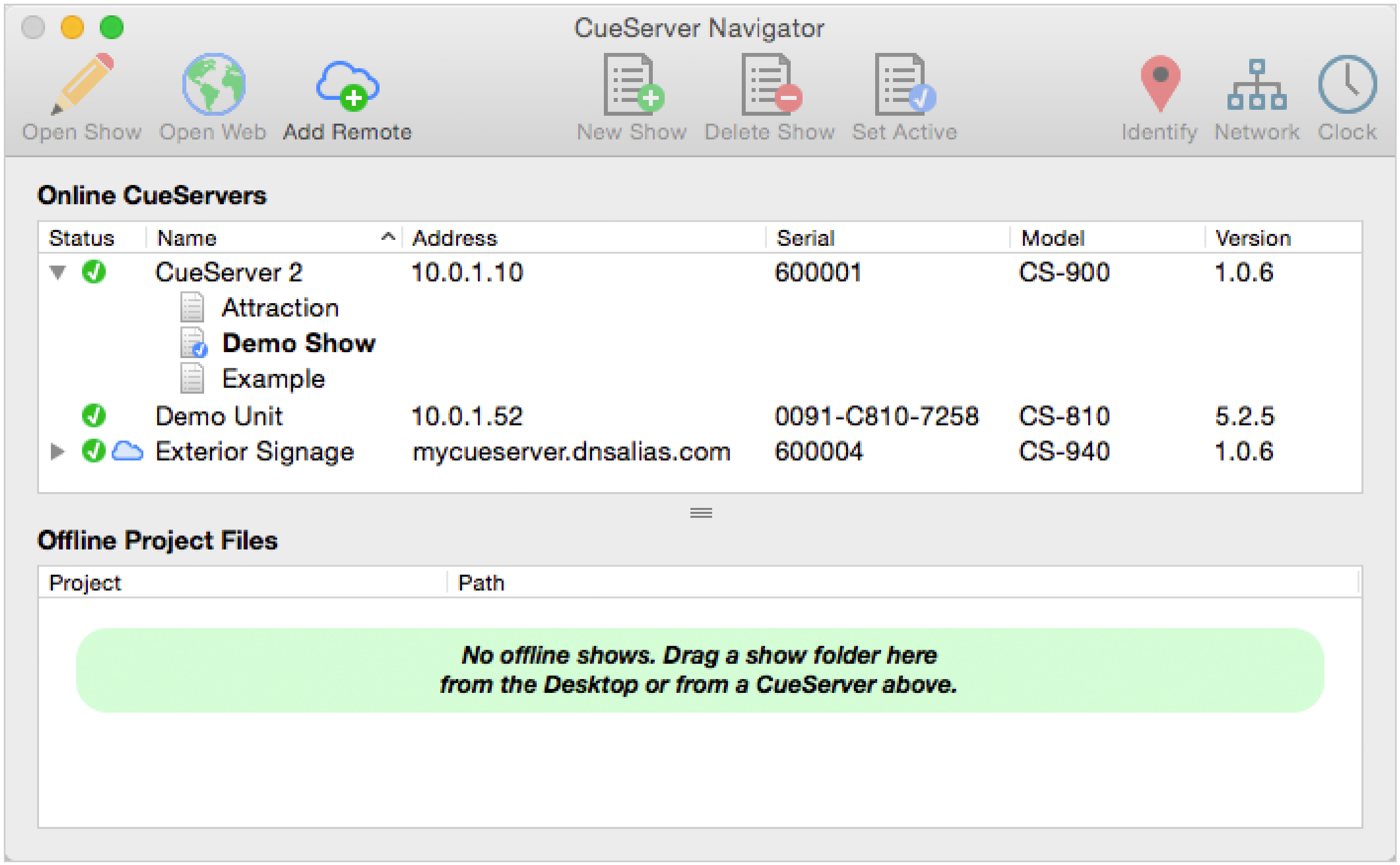
You cannot delete the currently active show. If you want to delete the active show, first switch to another show (or create a new one).

Working With Offline Shows

A *Show File* is a directory that contains the data stored in the show. The contents of the Show File directory are individual binary files and subdirectories for each object in the show, including Cues, Macros, Rules, Timers, Sounds, Web Content and more.



Since a Show File is actually a directory, it can't be opened on the computer like a regular data file. If you double-click on a Show File directory on your desktop, it will just open like any regular folder. Because of this, CueServer Studio has tools for working with Show File directories that make it easier to edit them.



Online CueServers

Status	Name	Address	Serial	Model	Version
✓	CueServer 2	10.0.1.10	600001	CS-900	1.0.6
	Attraction				
	Demo Show				
	Example				
✓	Demo Unit	10.0.1.52	0091-C810-7258	CS-810	5.2.5
✓	Exterior Signage	mycueserver.dnsalias.com	600004	CS-940	1.0.6

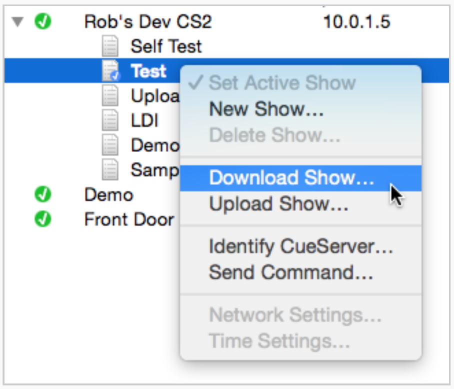
Offline Project Files

Project	Path

No offline shows. Drag a show folder here from the Desktop or from a CueServer above.

Downloading a Show File from CueServer to Computer

There are several ways to download a show file from a CueServer to the computer.



Using the Download Show contextual menu item.

Option 1: Use the *Download Show…* menu item available in the CueServer menu.

Option 2: Use the *Download Show…* contextual menu item available by right clicking (or control-clicking) on the show file in the CueServer.

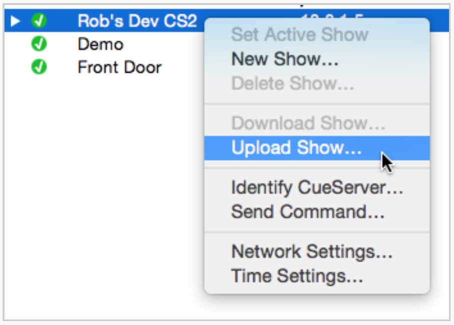
When using either Option 1 or 2, a standard file chooser dialog will appear asking where to place the downloaded show file. Once a destination folder is chosen, CueServer Studio will download the show file into the location chosen.

Option 3: Drag the show file directly from the CueServer in the top panel to the *Offline Shows* panel at the bottom of the window.

When dragging a show from the online panel to the offline panel, CueServer Studio will automatically download the show file from the CueServer to the computer's desktop and add the item to the offline projects list.

Uploading a Show File from Computer to CueServer

There are several ways to upload a show file from a computer to a CueServer.



Using the Upload Show contextual menu item.

Option 1: Use the *Upload Show…* menu item available in the CueServer menu.

Option 2: Use the *Upload Show…* contextual menu item available by right clicking (or control-clicking) on a CueServer.

When using either Option 1 or 2, a standard file chooser dialog will appear asking to choose the show file to upload. Once a show file is chosen, CueServer Studio will upload the show file to the selected CueServer.

Option 3: Drag a show folder directly from the *Offline Shows* panel at the bottom of the window to an online CueServer.

Option 4: Drag a show folder directly from the computer’s Desktop to an online CueServer.

When dragging a show from the offline panel or Desktop to an online CueServer, CueServer Studio will automatically upload the show file from the computer to the CueServer device.

Creating An Offline Show

To create a show file for offline editing, first click in the *Offline Project Files* list to select it.

Then, click the *New Show* toolbar item ( ).

A standard file save dialog window will appear, asking for a name and location to save the new show file.

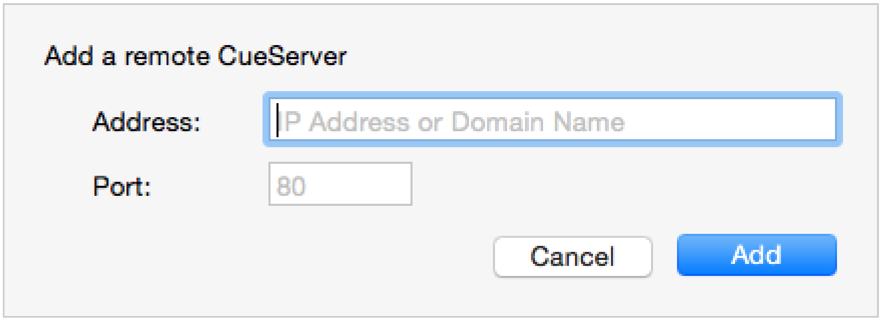
Once the name and location are given, CueServer Studio will create the new show file and add it to the *Offline Project Files* list so the offline show file can be opened and edited.

Working With Remote CueServers

Adding a Remote CueServer

To add a CueServer to the Navigator Window that is “across the Internet” (i.e., not on the local network), choose **Add Remote CueServer…** from the CueServer Menu, or click the **Add Remote** button ( ) in the toolbar.

The **Add Remote CueServer** window will appear:


Add a remote CueServer

Address: IP Address or Domain Name

Port: 80

The fields in this window are described below:

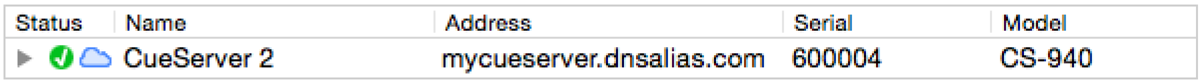
Address
This field can accept either an IP Address (for example: 50.167.102.1), or a domain name (such as: mycueserver.dnsalias.com).

Port
This field is used to specify the *port number* of the remote CueServer. If left blank, the default port 80 will be used. Valid port numbers range from 1 to 65535.

To add a remote CueServer (after the fields are filled out properly), click **Add**.

Viewing Remote CueServers in the Navigator Window

Once a Remote CueServer has been added to the Navigator Window, it will appear in the CueServer list with a small cloud icon ( ) next to the status icon. For example:



Status	Name	Address	Serial	Model
▶ ✓ ☁	CueServer 2	mycueserver.dnsalias.com	600004	CS-940

The cloud icon shows that the CueServer in the list is a Remote CueServer.

The following icons can appear in the status column for Remote CueServers:

- ✓ The CueServer is online.
- ? The CueServer is being contacted.
- ⊖ The CueServer is offline.

Remote CueServers that connect properly are automatically saved in the application’s preferences. Each time the application is launched, the added Remote CueServers will re-appear. If an added Remote CueServer cannot be contacted, it will not be saved in the preferences.

Removing Remote CueServers from the Navigator Window

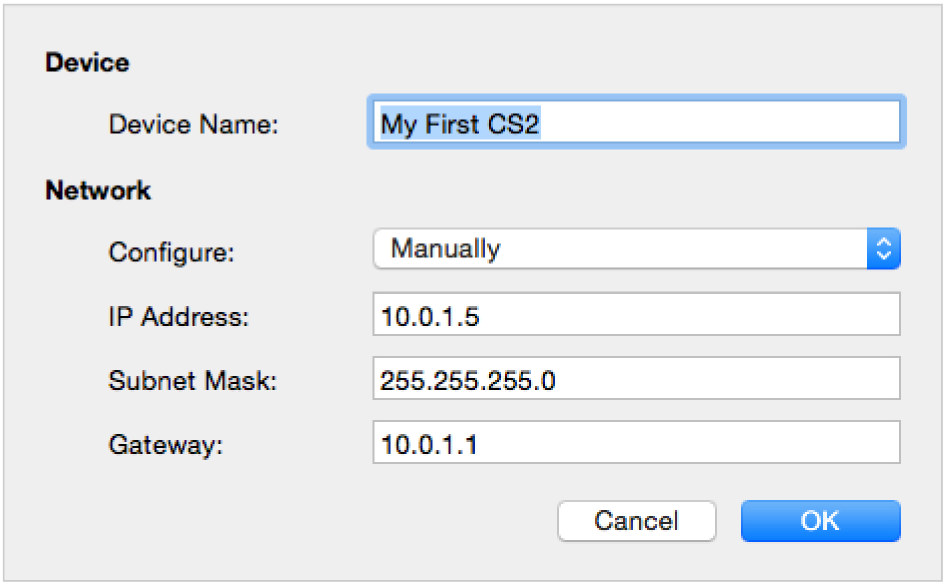
Simply select the Remote CueServer, and then press the **Delete** key on your keyboard.

Setting Network Parameters

When a CueServer is selected in the Navigator Window, it’s various network parameters can be changed by clicking on the Network Toolbar Item ( ), or by selecting the *Network Settings…* menu item in the CueServer menu.

These parameters include the device’s network name, DHCP settings, IP Address, Subnet and Gateway addresses.

A dialog window similar to the following will appear:


Device

Device Name: My First CS2

Network

Configure: Manually

IP Address: 10.0.1.5

Subnet Mask: 255.255.255.0

Gateway: 10.0.1.1

Device Name

This is the name of the device on the network (sometimes called the *hostname*). The device name can be set to any practical name that can be used to identify the CueServer on the network.

Network Address

CueServer allows the Network Address to be set manually or automatically. If the CueServer is on a network with a DHCP server or Router (which is common in buildings, offices and home networks), this setting can be set to *Using DHCP*.

Using DHCP

When *Using DHCP* is chosen, the IP Address fields become disabled. This is because the CueServer will fetch these address parameters from the network automatically. There is no need to set these parameters manually.

Manually

When *Manually* is chosen, the IP Address fields can be entered with a static IP Address, Subnet and Gateway address.

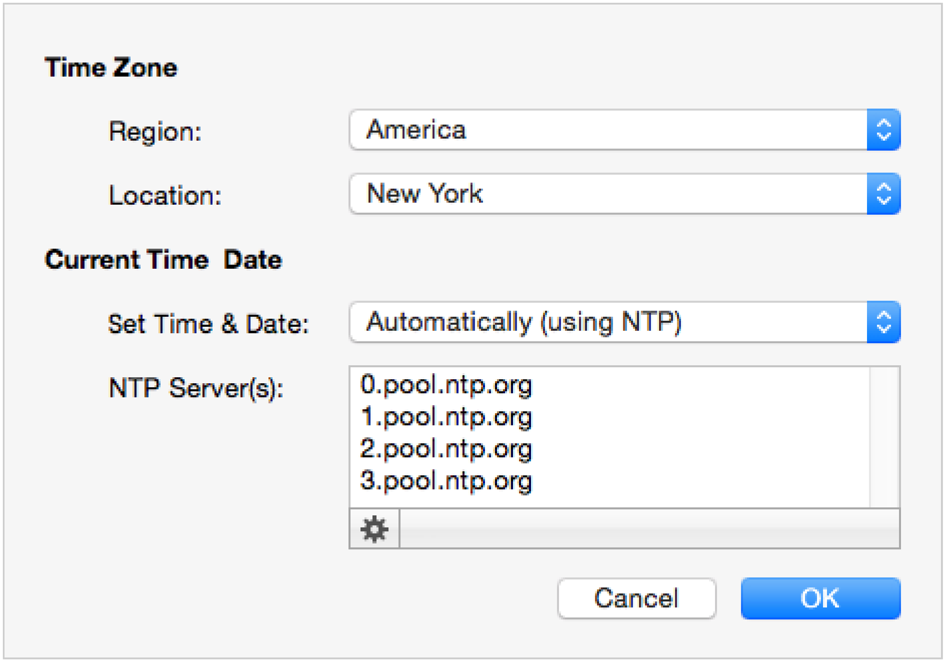
It is best to use this option if the CueServer is not connected to a network, or if the network is known to not have a DHCP server or Router.

Setting Clock Parameters

When a CueServer is selected in the Navigator Window, it’s various clock parameters can be changed by clicking on the Clock Toolbar Item (🕓), or by selecting the *Time Settings…* menu item in the CueServer menu.

These parameters include the timezone the unit is located within, network time protocol (NTP) server configuration, and/or manual date and time settings.

A dialog window similar to the following will appear:



Time Zone

Region: America

Location: New York

Current Time Date

Set Time & Date: Automatically (using NTP)

NTP Server(s):
0.pool.ntp.org
1.pool.ntp.org
2.pool.ntp.org
3.pool.ntp.org

Timezone

The top section of this window allows the timezone of the CueServer to be set. Use the *Region* menu first to select a general region from around the globe. Options exist for America, Asia, Australia, Canada, Europe, Pacific, US and others.

Once a region is chosen, use the *Location* menu to choose a specific timezone location within the region.

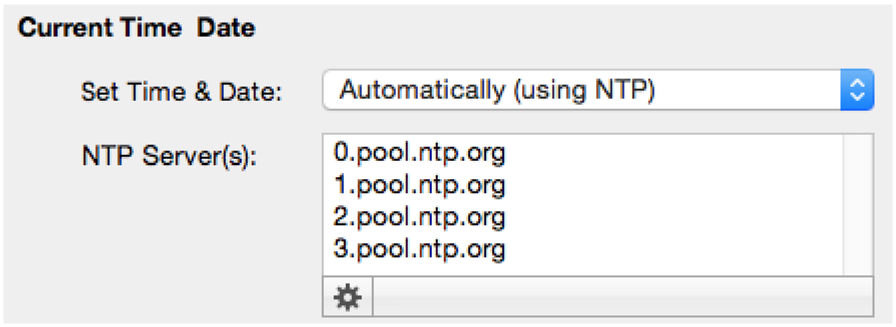
CueServer’s timezone database is derived from the standard Linux distribution and includes over 400 distinct regional locations. See the timezone listing for a complete list of available timezones.

Current Time & Date

CueServer allows the Time and Date to be set manually or automatically. If the CueServer has a network connection where it can reach the Internet, or if the network has a network time server, then the Set Time & Date option can be set to *Automatically*.

Automatically Set Time & Date

When *Automatically* is chosen, a text field appears that allows one or more NTP time server addresses to be entered. Put one time server per line.



Current Time Date

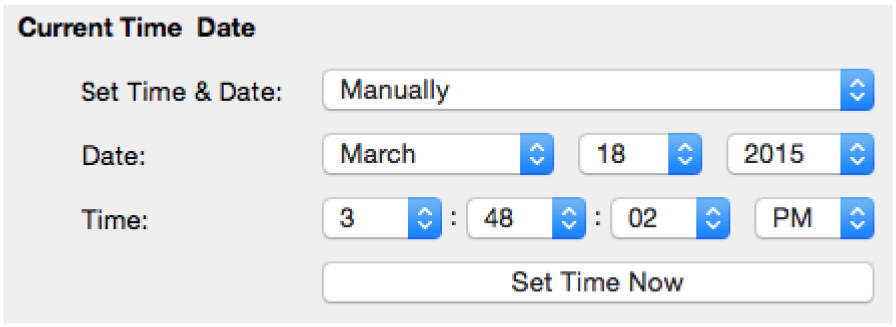
Set Time & Date: Automatically (using NTP)

NTP Server(s):
0.pool.ntp.org
1.pool.ntp.org
2.pool.ntp.org
3.pool.ntp.org

The gear button (⚙) can be clicked to pop up a menu that includes several popular choices of publicly available Network Time (NTP) Servers. Choosing one of these options will automatically fill the server list with one of these sets of options.

Manually Set Time & Date

When *Manually* is chosen, the time and date can be set manually.



Current Time Date

Set Time & Date: Manually

Date: March 18 2015

Time: 3 : 48 : 02 PM

Use the popup menus to choose the Time and Date. Before any of the menus are clicked, they show the current time of the computer. Once a menu is clicked, the time and date can be adjusted independently from the computer. Once the desired time is chosen, click on the **Set Time Now** button to set the clock in the CueServer.

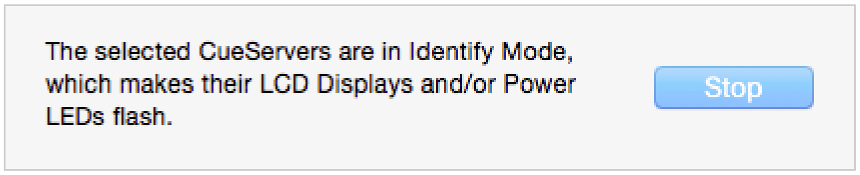
Identifying CueServers

When working with multiple CueServers, sometimes it may be useful to be able to positively identify which CueServer is which.

A CueServer's *Identify Mode* can be activated, which causes it's LCD Display and Power LED to flash. This function makes it easy to match a CueServer listed in the Navigator Window with a physical device on the network.

To activate the Identify Mode, select a CueServer in the list, then choose the **Identify…** item in the CueServer Menu, or click on the Identify toolbar icon (📍).

The CueServer will begin flashing, and the following window will appear:


The selected CueServers are in Identify Mode, which makes their LCD Displays and/or Power LEDs flash.

To exit the Identify Mode, click on the **Stop** button.

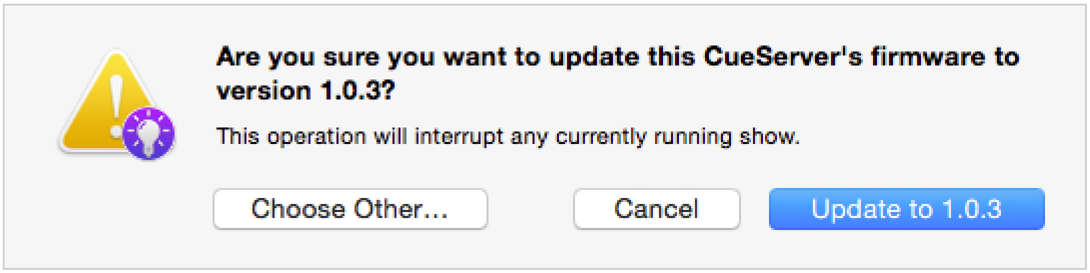
Updating Firmware

When new features or bug fixes become available for CueServer 2, a new version of CueServer Studio will be released. With each software release, CueServer Studio will check to make sure that the CueServer devices have the most up-to-date software version.

If a CueServer’s firmware is out of date, it will appear in the Navigator Window with a warning icon (⚠) in front of the firmware version number.

CueServer Studio can update the firmware in connected CueServers by choosing the **Update Firmware…** menu item in the CueServer menu.

The following dialog window appears:



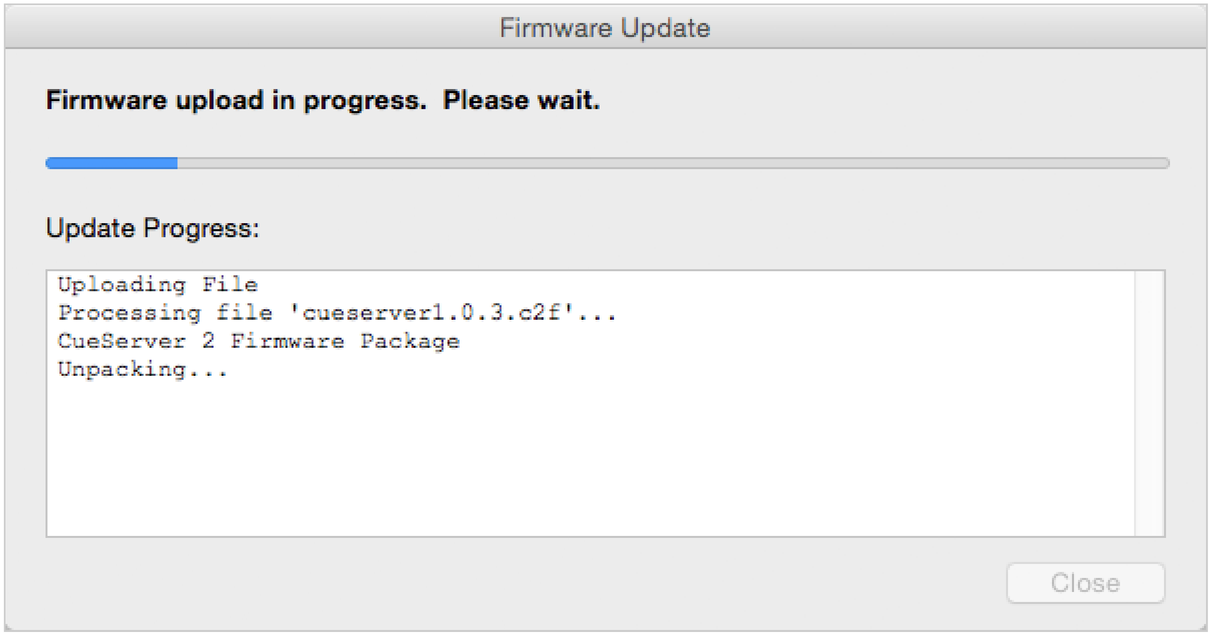
Are you sure you want to update this CueServer's firmware to version 1.0.3?

This operation will interrupt any currently running show.

In this example, CueServer Studio is recommending that the device be upgraded to version 1.0.3. This firmware image is embedded in the CueServer Studio application itself. Simply click on the **Update** button to perform the update.

If you want to update the CueServer to a different version of firmware, click on the **Choose Other…** button. A file chooser window will appear that will allow a different firmware version to be loaded. CueServer firmware files have the file extension **.c2f**.

When the firmware update process is running, a progress window appears:



Firmware upload in progress. Please wait.

Update Progress:

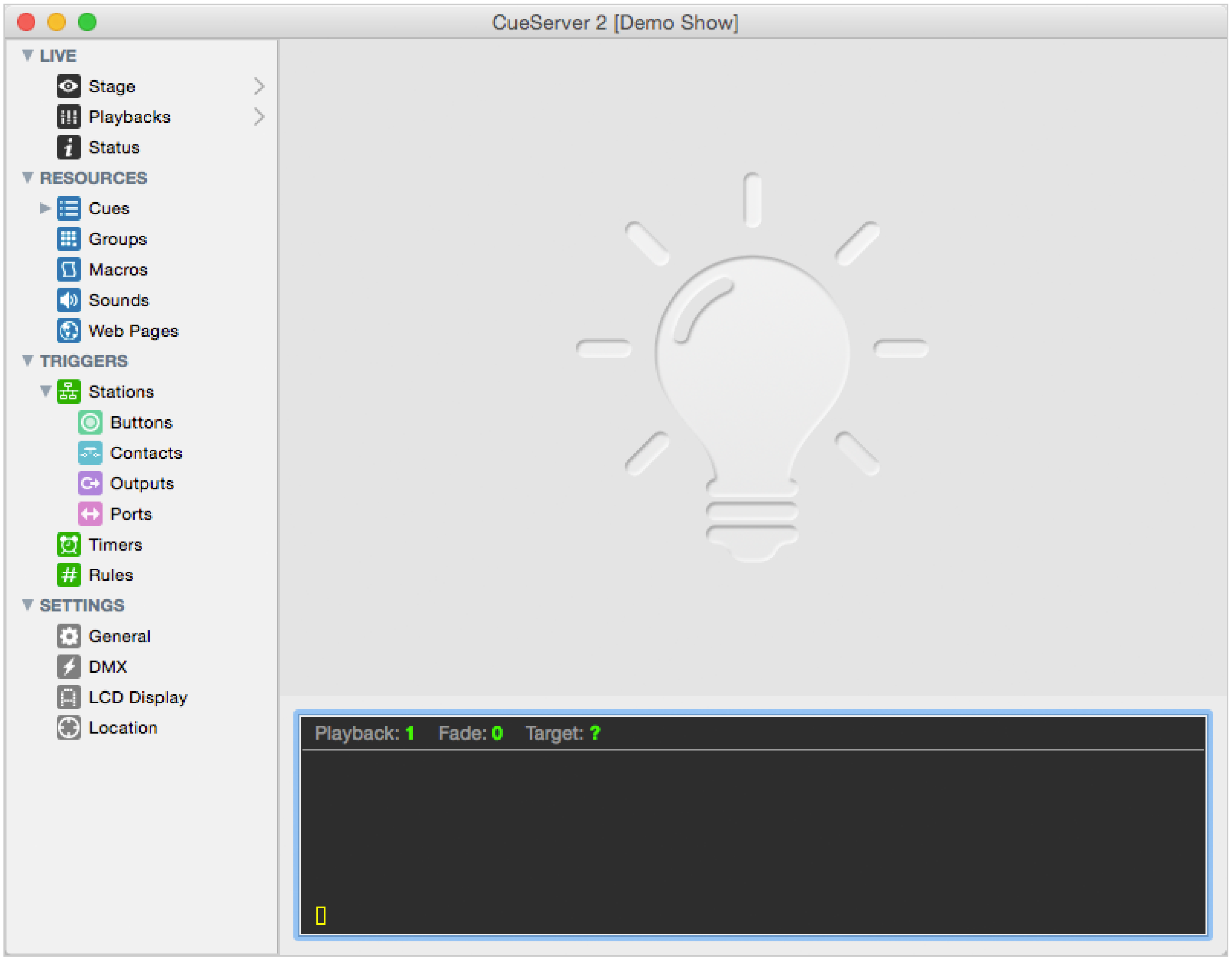
```
Uploading File
Processing file 'cueserver1.0.3.c2f'...
CueServer 2 Firmware Package
Unpacking...
```

The progress of the update is shown in the window. When the update is complete, the CueServer will reboot and the **Done** button can be clicked to dismiss the window.

Editor Window

Overview

The *Editor Window* is the primary window used to interact with, program and configure CueServer.



Use the Editor Window to view the “live” operation, edit resources and triggers, and set various configuration properties of a CueServer show.

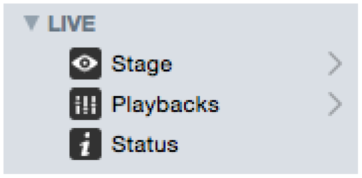
The panel on the left of the window contains numerous views into the CueServer, such as Stage, Cues, and Location. The following manual sections describe the details of each of these CueServer editor views:

- Live – live views of CueServer operation
    - Stage – for viewing DMX channels
    - Playbacks – for viewing playback faders
    - Status – for viewing the front-panel of the CueServer
- Resources – various content types for CueServer projects
    - Cues – scenes and timeline based streams
    - Groups – definitions of groups of channels
    - Macros – user-defined scripts
    - Sounds – audio clips
    - Web Pages – custom web pages for the project
- Triggers – definitions for incoming system events
    - Stations – setup for stations, buttons, contact-closures and more
    - Timers – setup for timers
    - Rules – a global list of rules
- Settings – system preferences
    - General – general purpose settings
    - DMX – DMX related settings
    - LCD Display – customization of the LCD display
    - Location – location settings for astronomical time

The panel at the bottom of the window is a live command line that allows the user to directly enter CueScript commands to cause the CueServer to perform operations. Note that this command line is only visible if you are editing the active show file in an “online” CueServer.

Live

The *Live* section of the navigator contains views that show the Stage, Playback Operation, and System Status of the CueServer. Each of these views show dynamic screens that are updating “live” as the CueServer is performing it’s operations.



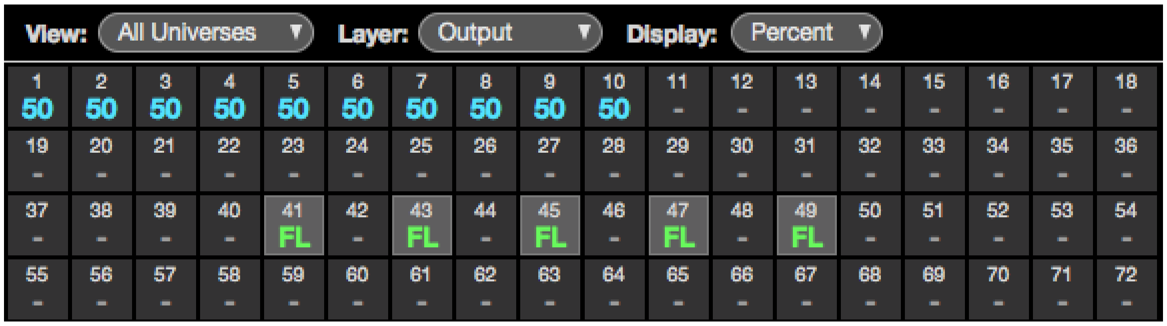
The following sections describe these views in more detail:

- Stage – for viewing DMX channels
- Playbacks – for viewing playback faders
- Status – for viewing the front-panel of the CueServer

Stage

Overview

The *Stage View* shows the output channels of the CueServer. This view is arranged in a grid of channels. Controls within the window change the visible layer of the channel grid between the device’s Output, one of the Playbacks, or the Input. Various colors indicate the source of each channel value and/or the state of the channel.



View: All Universes Layer: Output Display: Percent

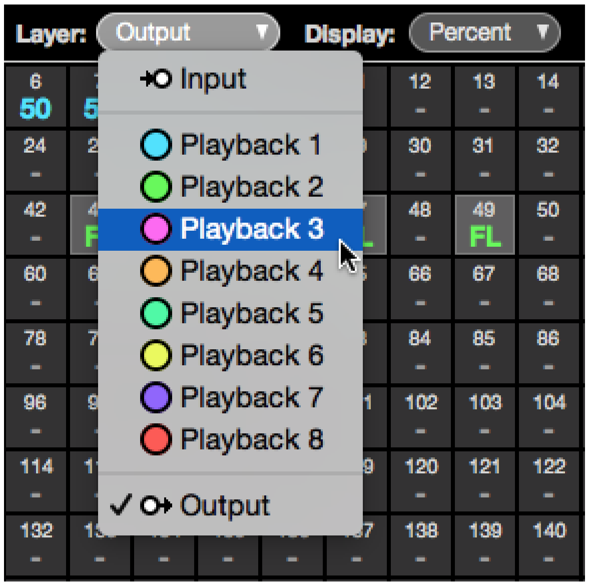
1 50	2 50	3 50	4 50	5 50	6 50	7 50	8 50	9 50	10 50	11 -	12 -	13 -	14 -	15 -	16 -	17 -	18 -
19 -	20 -	21 -	22 -	23 -	24 -	25 -	26 -	27 -	28 -	29 -	30 -	31 -	32 -	33 -	34 -	35 -	36 -
37 -	38 -	39 -	40 -	41 FL	42 -	43 FL	44 -	45 FL	46 -	47 FL	48 -	49 FL	50 -	51 -	52 -	53 -	54 -
55 -	56 -	57 -	58 -	59 -	60 -	61 -	62 -	63 -	64 -	65 -	66 -	67 -	68 -	69 -	70 -	71 -	72 -

While cues are running and/or channels are fading, they will update live within this view. The channels are colored to match the display color for each Playback Fader. In the example above, the Blue channels are coming from Playback 1 and the Green channels are coming from Playback 2.

The area behind the odd-numbered channels from 41 through 49 are shaded in Gray to indicate that these channels are currently selected.

Choosing the View Layer

Use the **Layer** popup menu to choose which layer of the DMX composition is being shown:

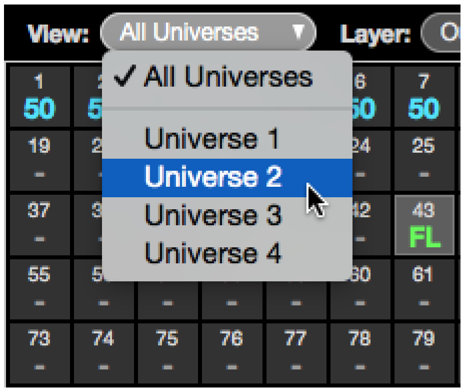


The view options are:

- **Input** – This view shows any DMX values that are being input into the device.
- **Playback** – This view shows DMX values that are present in a specific Playback Fader. The colored circle shows the color of the channels for the given Playback Fader.
- **Output** – This view shows the final composite DMX values that are being output from the device.

Choosing the Visible Universes

Use the **View** popup menu to choose which universe(s) are being shown:

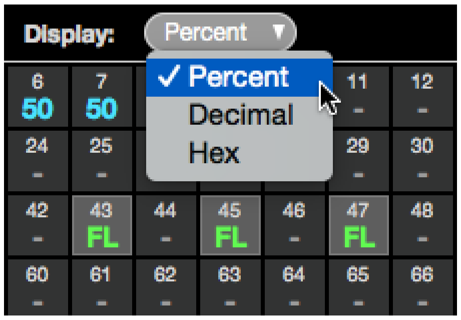


The view options are:

- **All Universes** – This view shows all Universes in one continuous table.
- **Universe *n*** – This view focuses the display on only the chosen Universe.

Choosing a Display Mode

Use the **Display** popup menu to choose how the values in the channel grid are shown:



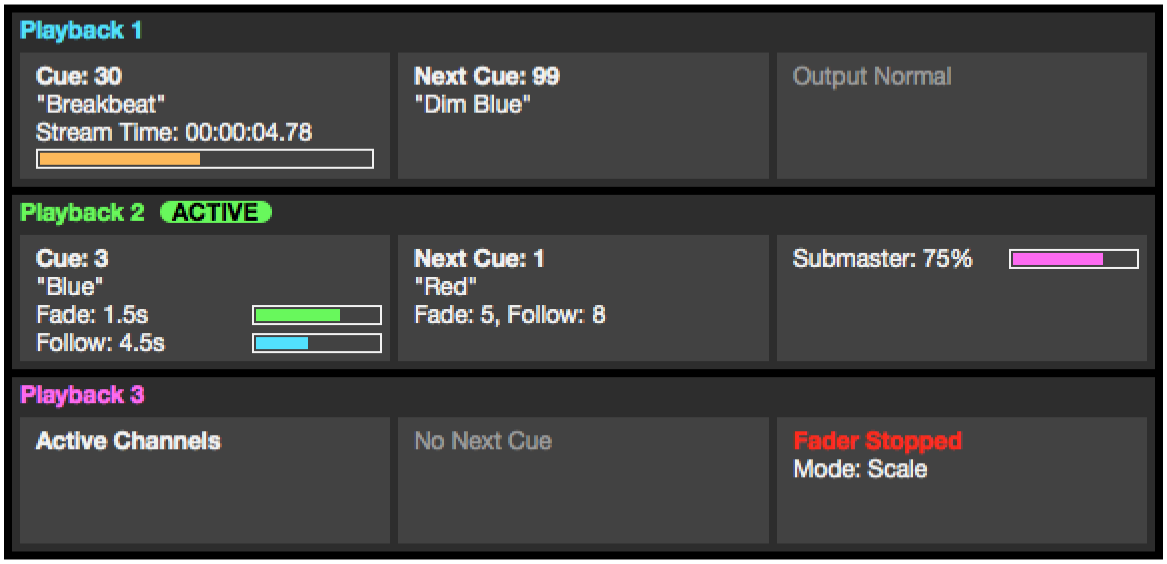
The display options are:

- **Percent** – This mode shows channel levels as a percentage. Values range from 0 to 99, and then FL (meaning Full, or 100%).
- **Decimal** – This mode shows channel levels in decimal format. Values range from 0 to 255.
- **Hexadecimal** – This mode shows channel levels in hexadecimal format. Values range from 00 to FF.

Playbacks

Overview

The *Playbacks View* shows the current state and properties of the Playback Fader layers of the CueServer. This view is arranged in a stack of Playbacks. Each Playback has three panes, the left-hand pane shows what is currently loaded in the Playback, the center pane shows what's coming up next, and the right-hand pane shows additional properties for the Playback. While cues are running and/or channels are fading, bar graphs appear that show the progress of the cues, fades, streams, etc.


Playback 1
Cue: 30
"Breakbeat"
Stream Time: 00:00:04.78
Next Cue: 99
"Dim Blue"
Output Normal
Playback 2 ACTIVE
Cue: 3
"Blue"
Fade: 1.5s
Follow: 4.5s
Next Cue: 1
"Red"
Fade: 5, Follow: 8
Submaster: 75%
Playback 3
Active Channels
No Next Cue
Fader Stopped
Mode: Scale

In the example above, Playback 1 is currently playing back Cue 30, which is a streaming cue called “Breakbeat”. It is currently 4.78 seconds into the stream. The next cue in Playback 1 is Cue 99, which is called “Dim Blue”. Playback 2 is the active playback, it is currently fading into Cue 3 “Blue”. The fade has 1.5 seconds remaining, and a follow timer is running with 4.5 seconds remaining. The next cue in Playback 2 is Cue 1 “Red”, and that cue will have a Fade Time of 5 seconds, and a Follow Timer of 8 seconds. Also, Playback 2’s submaster has been lowered to 75%. Finally, Playback 3 has manually set “active” DMX channels in it and no next cue. Playback 2 is “stopped”, meaning that fade and follow timing is disabled, and it’s layer mode is set to “Scale”.

The Current Pane (Left Side)

The pane on the left-hand side of each Playback shows what is currently loaded in the Playback.

- **Empty** – Shown if the playback has no active channels. An empty Playback has no effect on the DMX output.
- **Active Channels** – Shown when the Playback has active channels (not originating from a Cue).
- **Cue (*n*)** – Shown when the Playback is loaded with the channels from a particular Cue.
- **Cue (*n*) + Changes** – Shown when the Playback was loaded with a Cue, and then manual channel values were changed.
- **Fade (*time*)** – Shown when the Playback is actively fading channels. A green progress bar ( ) shows the fade time remaining.
- **Follow (*time*)** – Shown when the Playback is counting down to an auto-follow event. A blue progress bar ( ) shows the follow time remaining.
- **Stream (*time*)** – Shown when a Streaming Cue is being played back. An orange progress bar ( ) shows the stream time remaining.

The Next Pane (Center)

The panel in the center of each Playback shows what is queued to be "next".

- **Next Cue (*n*)** – Shown if the Playback has a *next cue* that will execute upon a Go command or auto-follow.
- **Fade (*time*)** – Shown to indicate the fade time of the *next cue*.
- **Follow (*time*)** – Shown to indicate the follow time of the *next cue*.
- **Link (*n*)** – Shown to indicate the link of the *next cue*.
- **No Next Cue** – Shown if the Playback does not have a *next cue*.

The Properties Pane (Right Side)

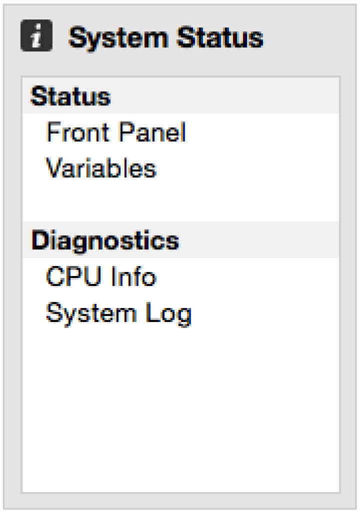
The panel on the right-hand side of each Playback shows additional properties for the Playback Fader.

- **Output Normal** – Shown if the Playback has no overridden properties. All values are normal.
- **Stack (*name*)** – Shown if the Playback has a cue stack assigned to it.
- **Fader Stopped** – Shown in Red color when the Playback is stopped. A stopped Playback has it's timing overridden, meaning that setting channel levels or executing cues always appear immediately (they do not fade), the follow timer does not run, and streaming cues are paused.
- **Channels Parked** – Shown in Red color when channels in the Playback are parked. Parked channels retain their current values and cannot be modified by executing cues or by using the Channel, At, Release, or Clear commands. Parked channels must either be Unparked, or the CueServer can be Reset.
- **Submaster (*level*)** – Shown when the Playback's submaster level is not at 100%. A pink progress bar ( ) shows the submaster percentage.

- **Mode (*mode*)** – Shown if the Playback’s combine mode is set to anything other than the default “Merge” mode. Options include **Override**, **Scale**, and **Pin**.

Status

The *Status* page provides several views that show live status of various CueServer subsystems.



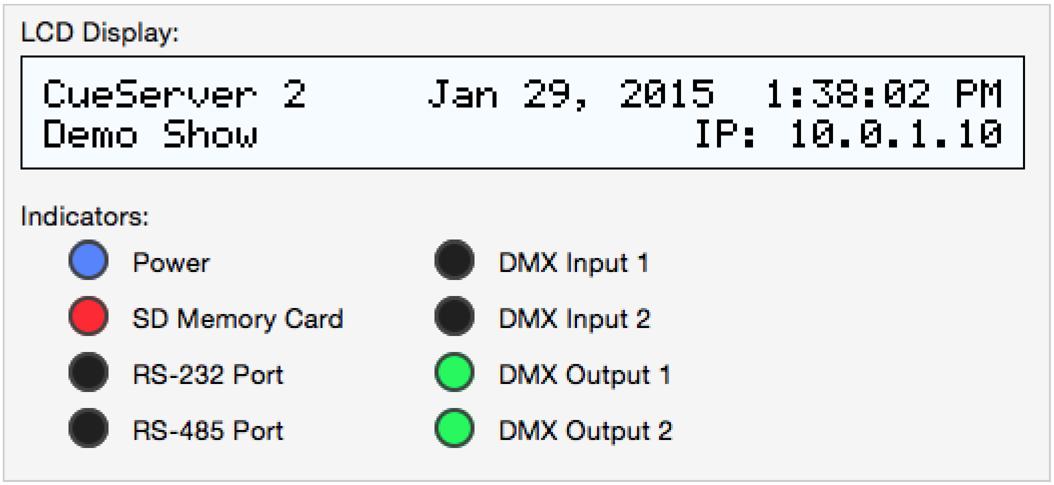
The following status views are available:

- Front Panel – a live view of the front-panel of the CueServer.
- Variables – a live listing of user-defined variables.
- CPU Info – a live view of the hardware status.
- System Log – the current system log.

Note that if any of the status views has an important condition that needs to be shown to the user, the caution icon (⚠) will appear to the right of the corresponding line in the list of status views.

Front Panel

The *Front Panel View* shows the current state of the physical CueServer. The CueServer's LCD display and LED indicators are visible in this view.

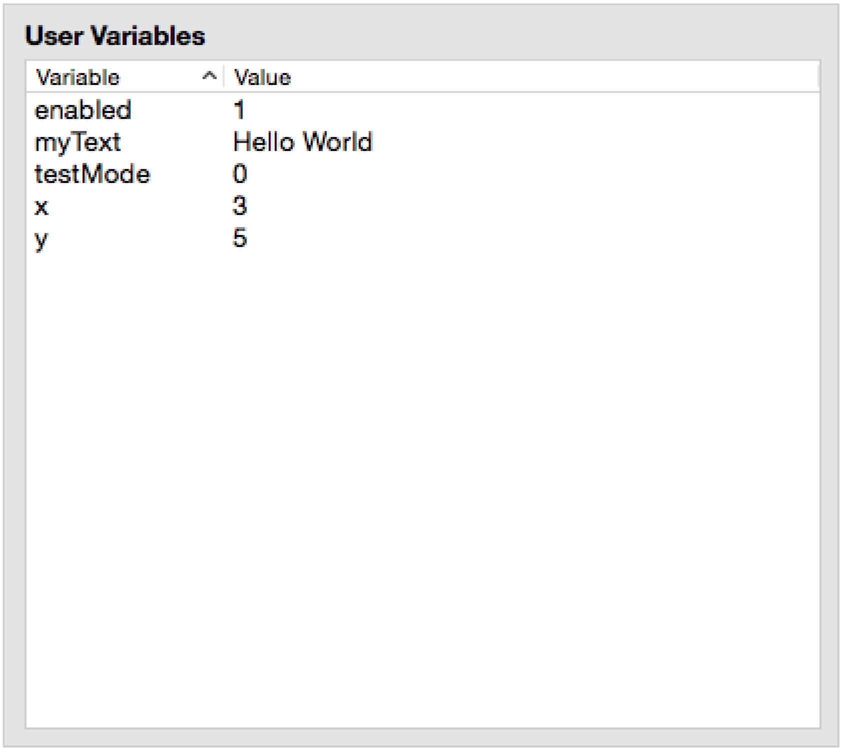


```
CueServer 2      Jan 29, 2015  1:38:02 PM
Demo Show                 IP: 10.0.1.10
```

As the LCD display and/or LED indicators on the physical CueServer changes, they are updated live on this view.

Variables

The *Variables View* shows any currently defined user variables.


User Variables

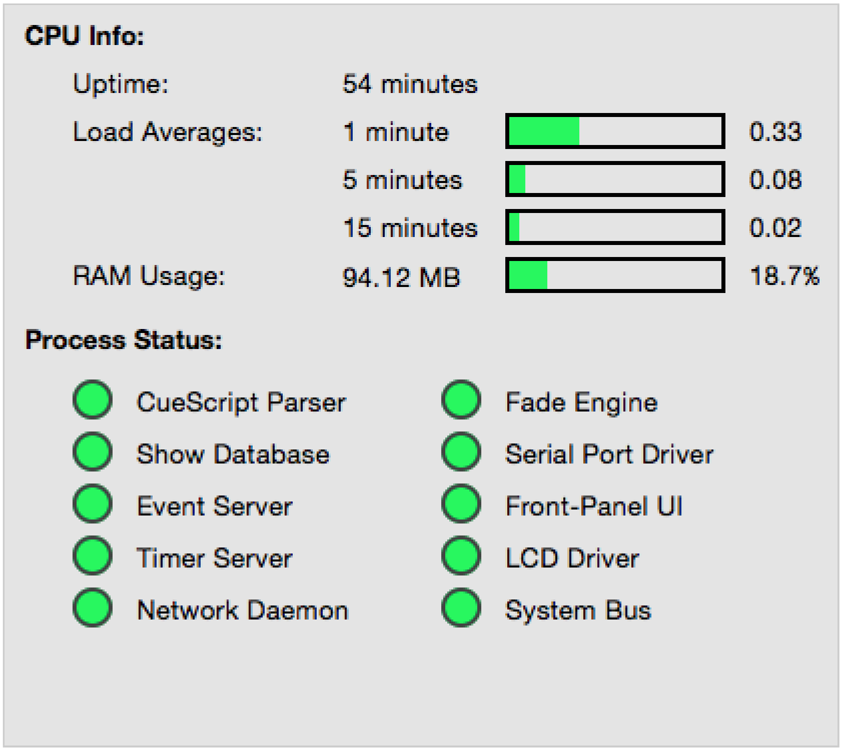
Variable	Value
enabled	1
myText	Hello World
testMode	0
x	3
y	5

Whenever any CueScript statements are used to define or update the value of a user variable, this view will show those values “live”.

For more information about using variables in scripts, see the Variables section of the CueScript Language chapter.

CPU Info

The *CPU Info View* shows the status of the CueServer hardware.



CPU Info:

Uptime:	54 minutes		
Load Averages:	1 minute		0.33
	5 minutes		0.08
	15 minutes		0.02
RAM Usage:	94.12 MB		18.7%

Process Status:

CueScript Parser	Fade Engine
Show Database	Serial Port Driver
Event Server	Front-Panel UI
Timer Server	LCD Driver
Network Daemon	System Bus

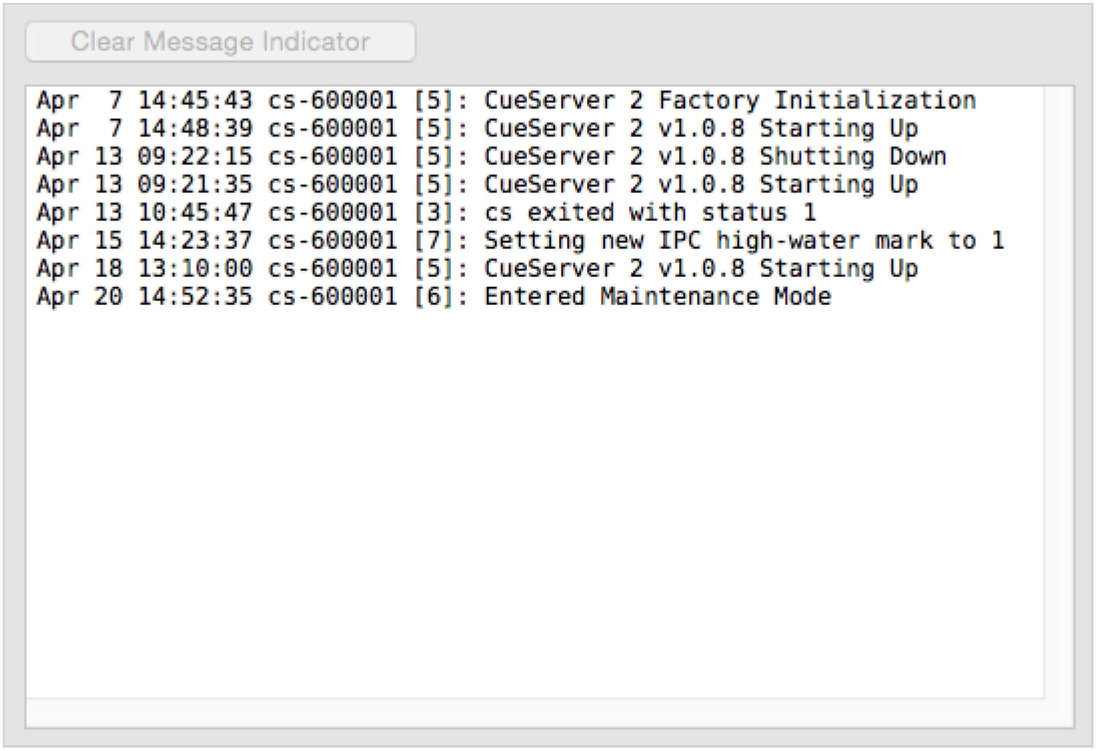
The following information is displayed:

- **Uptime** – shows the number of days, hours, and minutes since the CueServer was powered-on.
- **Load Averages** – shows the CPU load, averaged over the last 1, 5 and 15 minutes.
- **RAM Usage** – shows how much system RAM is being used. Note that this is not the memory on the SD Card.
- **Process Status** – shows the running state of each of CueServer's internal processes. Green means that the service is running, Red means that an error has occurred.

Note that if any of the processes in the CPU Info view require attention, a warning icon (⚠) will appear next to the CPU Info line in the status list.

System Log

The *System Log* shows internal system messages posted by CueServer's operating system and related software.



```
Apr  7 14:45:43 cs-600001 [5]: CueServer 2 Factory Initialization
Apr  7 14:48:39 cs-600001 [5]: CueServer 2 v1.0.8 Starting Up
Apr 13 09:22:15 cs-600001 [5]: CueServer 2 v1.0.8 Shutting Down
Apr 13 09:21:35 cs-600001 [5]: CueServer 2 v1.0.8 Starting Up
Apr 13 10:45:47 cs-600001 [3]: cs exited with status 1
Apr 15 14:23:37 cs-600001 [7]: Setting new IPC high-water mark to 1
Apr 18 13:10:00 cs-600001 [5]: CueServer 2 v1.0.8 Starting Up
Apr 20 14:52:35 cs-600001 [6]: Entered Maintenance Mode
```

Most messages in the System Log are only useful for diagnosing problems, however other informational messages can appear in the System Log as well.

For instance, the System Log shows each time the system is rebooted.

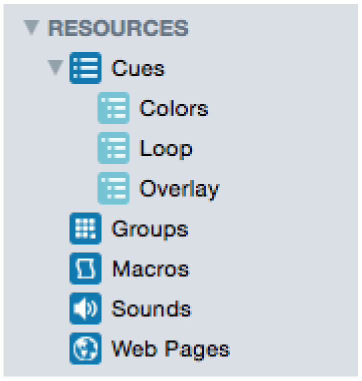
Also, user-defined messages can be added to the System Log by using the **Log** CueScript command.

When a message is added to the System Log that indicates a serious condition, the Power LED will begin to blink. This is called the "Message Indicator". It means that the System Log contains an important message. To clear this indication, click on the "Clear Message Indicator" button.

If a new important message is currently showing, a warning icon (⚠) will appear next to the System Log line in the status list.

Resources

The *Resources* section of the navigator contains views that edit Cues, Groups, Macros, Sounds and Web Pages in the CueServer project.



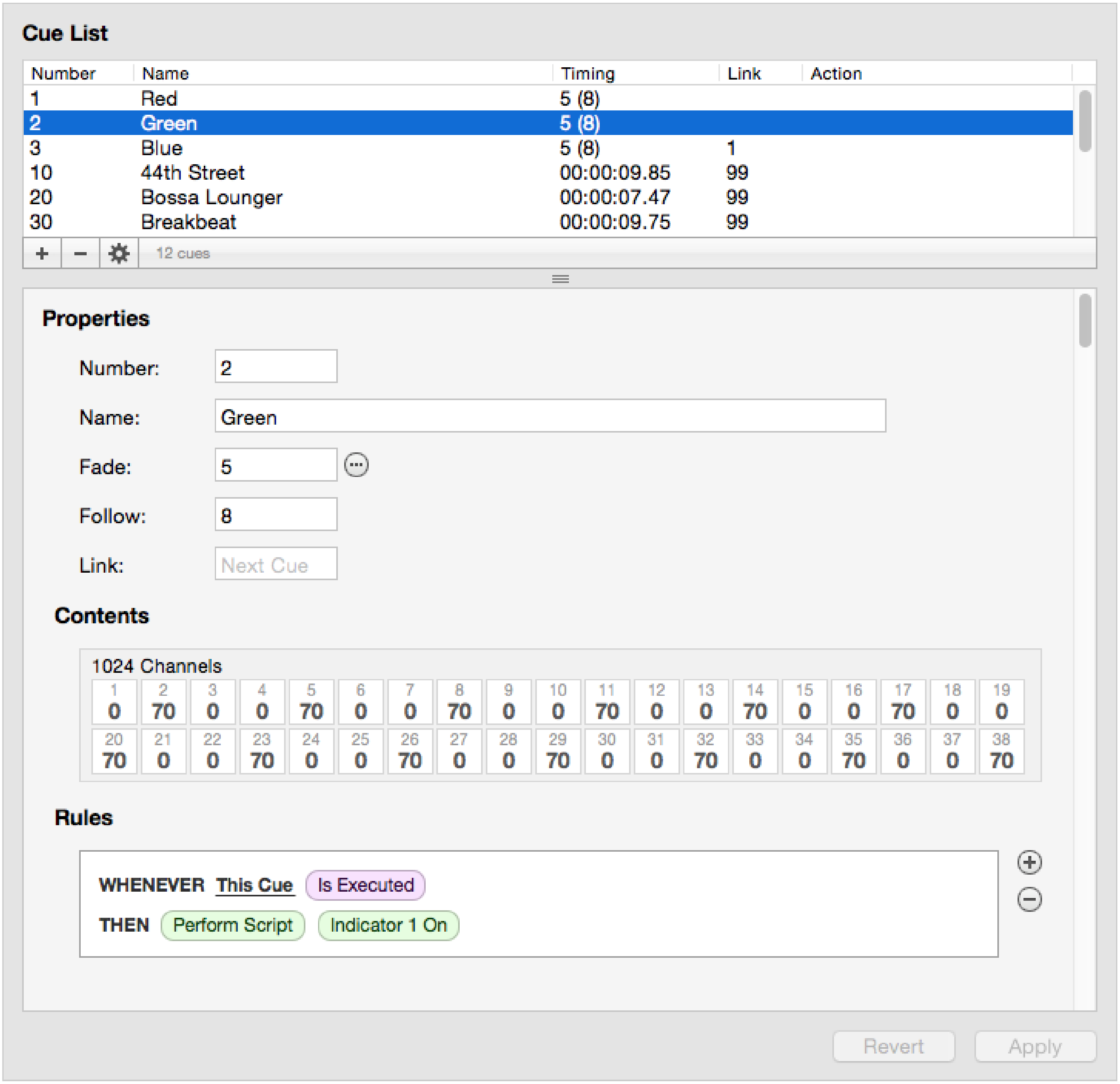
The following sections describe these views in more detail:

- Cues – scenes and timeline based streams
- Groups – definitions of groups of channels
- Macros – user-defined scripts
- Sounds – audio clips
- Web Pages – custom web pages for the project

Cues

Overview

The *Cues* editor shows the Cue List, and allows for the creation, capture, modification and removal of cues from the project.



Cue List

Number	Name	Timing	Link	Action
1	Red	5 (8)		
2	Green	5 (8)		
3	Blue	5 (8)	1	
10	44th Street	00:00:09.85	99	
20	Bossa Lounger	00:00:07.47	99	
30	Breakbeat	00:00:09.75	99	

12 cues

Properties

Number:	2
Name:	Green
Fade:	5
Follow:	8
Link:	Next Cue

Contents

1024 Channels

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	70	0	0	70	0	0	70	0	0	70	0	0	70	0	0	70	0	0
20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38
70	0	0	70	0	0	70	0	0	70	0	0	70	0	0	70	0	0	70

Rules

WHENEVER This Cue Is Executed
THEN Perform Script Indicator 1 On

The Cues Editor is divided into several sections. The top panel shows the list of Cues. Click on a cue to have it appear in the lower panel. Once selected, a Cue’s properties, contents and rules can be viewed or modified.

For details about different aspects of creating and modifying cues, see the following topics:

- Cue Types – discusses the differences between normal and streaming cues.
- Adding Cues – to learn how to add cues to a project.
- Cue Properties – for a description of the various properties of a cue.
- Cue Contents – to see how the contents of a cue are displayed.
- Cue Rules – for how to add automation rules to a cue.

Cue Types

There are two cue types available to CueServer.

Normal Cues

A “normal” cue is similar to the type of cue used on traditional lighting consoles. A cue of this type stores a single scene (or part of a scene).

In CueServer, a normal cue stores an array of DMX channel values, which will be recalled when the cue is executed. The cue may contain *all*, *some*, or *none* of the available DMX channels in the system. Normal cues have extra parameters such as fade and follow times, an optional linked cue, and automation rules.

Generally speaking, when playing back (executing) normal cues, the output of the CueServer will crossfade to a new scene. Again, a normal cue may only include *some* of the DMX channels, so only part of a scene may be affected by playing back a normal cue.

Streaming Cues

A “streaming” cue is a different type of cue that stores DMX channels and their changes over a period of time.

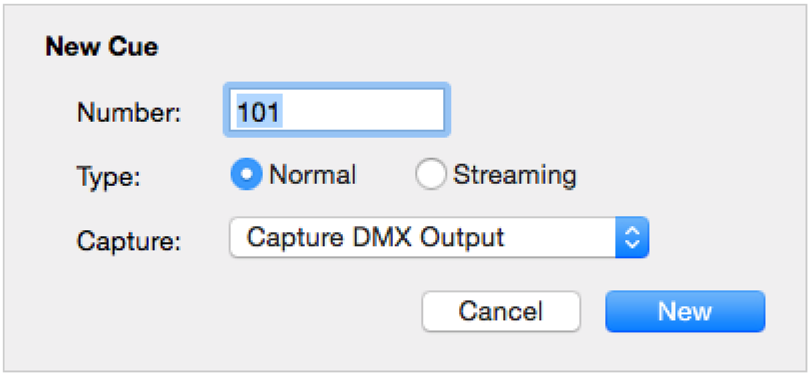
An analogy can be made between a streaming cue in CueServer, and a “tape recorder” for audio. When a streaming cue is captured in CueServer, every change to a DMX channel during the capture is saved. Then, when the streaming cue is played back, the changes occur in real-time just the same way that it was recorded.

Streaming cues have extra parameters such as playback mode, follow time, an optional linked cue, and automation rules.

Adding Cues

To add a new cue to the cue list, click the plus button (+) at the lower-left corner of the cue list.

The *New Cue* dialog window will appear:


New Cue

Number: 101

Type: [✓ Normal] Streaming

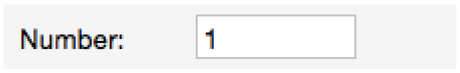
Capture: Capture DMX Output

The process of recording new cues will be improved significantly in the next version of CueServer Studio. Check back soon for more details.

Cue Properties

Each cue has a number of properties that may be edited:

Number

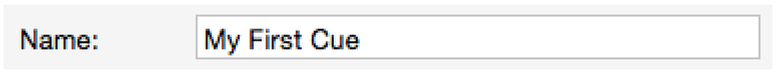


Number: 1

By convention, every cue in a cue list has a number. Valid cue numbers range from 0 through 999999. Optionally, up to two digits can be used after a decimal point (for example, Cue 1.23).

Once a cue is recorded, it’s number can be changed by entering a new number into this field.

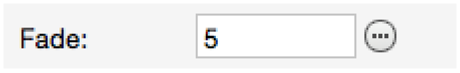
Name



Name: My First Cue

A cue may be given a descriptive name.

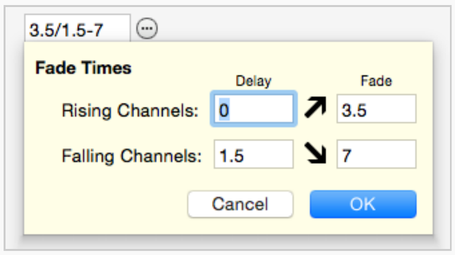
Fade (normal cues only)



Fade: 5

A normal cue has a fade time (expressed in seconds) that is used to specify how quickly the cue’s channels will crossfade from their previous values to the ones recorded in the cue. Fade times from 0 (no fade) to 86400 seconds (24 hours) may be specified.

Fade times can be split into separate times for channels fading up and channels fading down, and delays can be introduced to the up-fading and down-fading channels.



3.5/1.5-7

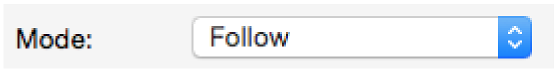
Fade Times

	Delay		Fade
Rising Channels:	0	↗	3.5
Falling Channels:	1.5	↘	7

Fade details window.

Click on the More button ( ) next to the fade field to display a window to enter advanced fade time parameters.

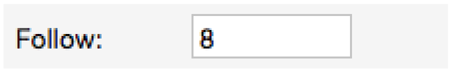
Mode (streaming cues only)


Mode: Follow

A streaming cue can be set to play back with one of four modes:

- **Follow** – When the stream finishes, the next cue automatically follows.
- **Loop Indefinitely** – The stream will loop each time it reaches it's end.
- **Hold Last Channels** – When the stream reaches its last frame, playback stops and the last channel values remain active.
- **Blackout** – When the stream reaches its last frame, playback stops and all channels are set to zero.

Follow


Follow: 8

Cues have an *auto follow timer* that begins when the cue is executed, as specified by this field (in seconds). When the timer expires, the playback fader automatically executes a *Go* to advance to the next cue in the cue list (or whatever cue the current cue is linked to).

This field can be left blank to allow cues to advance in regular numerical order.

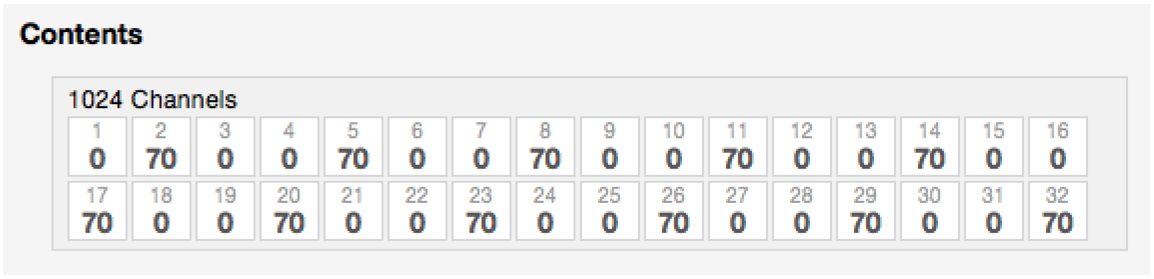
Cue Contents

Each cue may contain DMX channels, or streaming data, or may be empty.

The contents of the cue is displayed in the **Contents** section of the Cue Editor panel.

One of three types of content will be displayed:

Normal DMX Channels



Contents

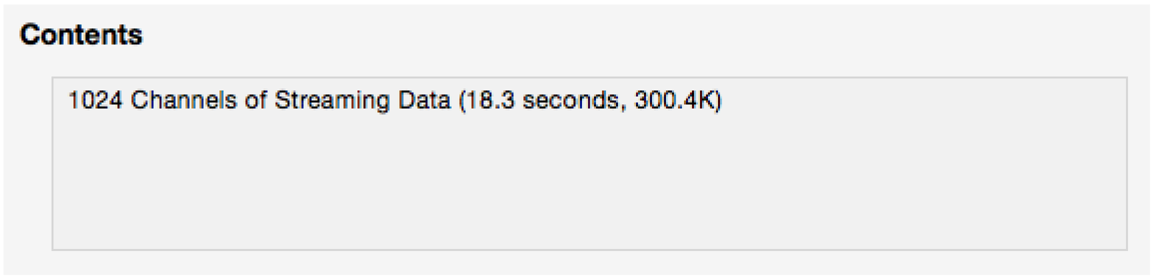
1024 Channels

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	70	0	0	70	0	0	70	0	0	70	0	0	70	0	0
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
70	0	0	70	0	0	70	0	0	70	0	0	70	0	0	70

Normal cues contain DMX channels. The cue might have been recorded with all DMX channels in it, or only a subset of available channels (selected channels).

When a cue with DMX channels is executed, those channel values will appear in the active playback fader. If the cue has a fade time of zero (no fade time), the channel values will appear immediately. If the cue has a fade time, then the channels will crossfade from their previous values to the ones in the cue.

Streaming Cue Data



Contents

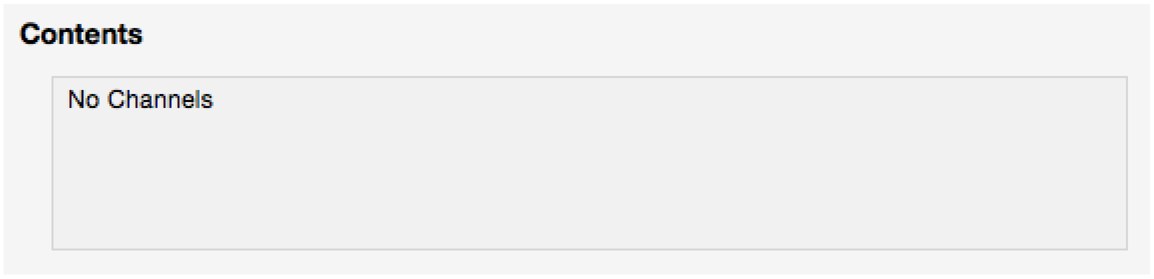
1024 Channels of Streaming Data (18.3 seconds, 300.4K)

A streaming cue contains a recording of DMX data over a period of time.

When a cue with streaming DMX data is executed, the recorded channel data plays back over time matching the changes that were occurring when it was recorded.

Recording and playing back streaming cues is similar to using a tape recorder to store and then play back an audio recording. Streaming cues do a similar thing with DMX lighting data.

Empty Cues



A cue can be recorded with **no** DMX channels. This type of cue does not directly affect any DMX channels when it is executed.

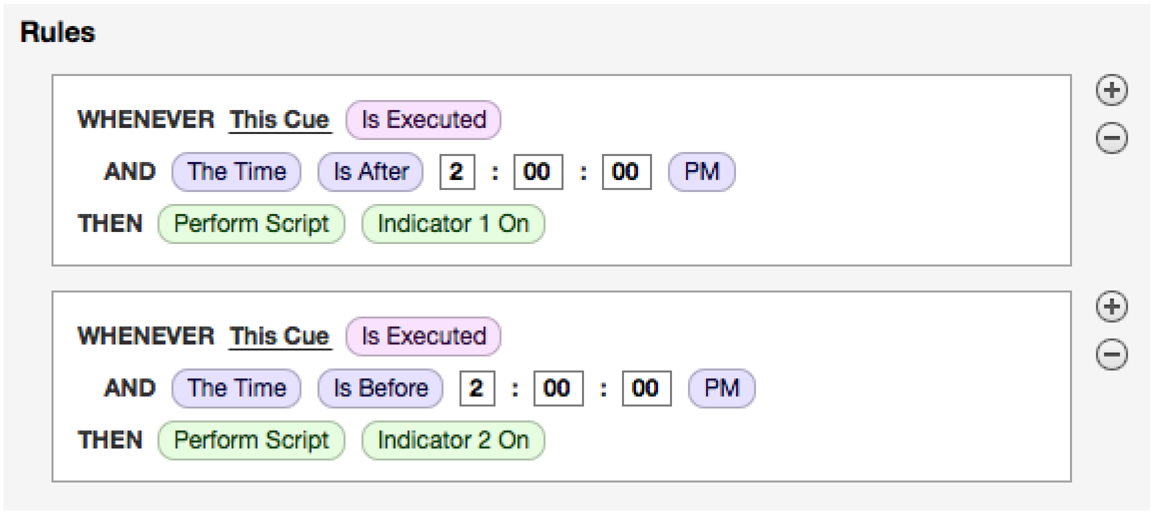
An empty cue will still observe it’s follow timing and it will also evaluate any rules in the cue, but it will not change any DMX channel values.

In the current version of CueServer Studio, the only way to edit the channel values in cues is to either Record or Update a cue. A future version of the software will improve the Contents panel of this window to allow channel values to be directly editable.

Cue Rules

Rules can be added to a cue to allow it to automate certain tasks when it is executed.

The rules for a cue might look like this:


Rules
WHENEVER This Cue Is Executed
AND The Time Is After 2 : 00 : 00 PM
THEN Perform Script Indicator 1 On
WHENEVER This Cue Is Executed
AND The Time Is Before 2 : 00 : 00 PM
THEN Perform Script Indicator 2 On

To add a rule to a cue, click on the “plus” button (⊕).

Then, click on the various “bubble” buttons (*Choose...*) in the rule to build an event, conditions and action that the rule will execute.

For more information about building rules, see the Rules topic.

CueScript Language

CueServer uses a command language called ***CueScript*** as the basis of nearly all of CueServer's control and automation capabilities. You will use CueScript to make CueServer perform actions. If you need CueServer to start playing a cue, you can enter `Cue 1 Go` on the command line. If you want CueServer to fade up a DMX channel, enter `Time 5 Channel 1 At FL`.

Not only can CueScript be used to enter live commands into CueServer, but CueScript is used throughout the system to perform all kinds of automation tasks. Advanced logic can be added to a CueServer project using CueScript to orchestrate lighting cues with button presses, contact closure inputs, serial port strings, LCD messages, digital outputs, and much more.

CueScript was created with the following in mind:

- It must be easy to use – the language reads easily in English.
- It must be familiar to lighting professionals – commands like `Group 1 Release` are very "console-like".
- It has a short-hand abbreviation system to make it easier to type – Instead of typing `Channel 1 At 100`, you can type `C1A100`.

The following sections describe the language in more detail.

CueScript Overview

The following topics describe the details of the language:

- Executing Commands
- Command Syntax
- Expressions
- Command Context
- Levels

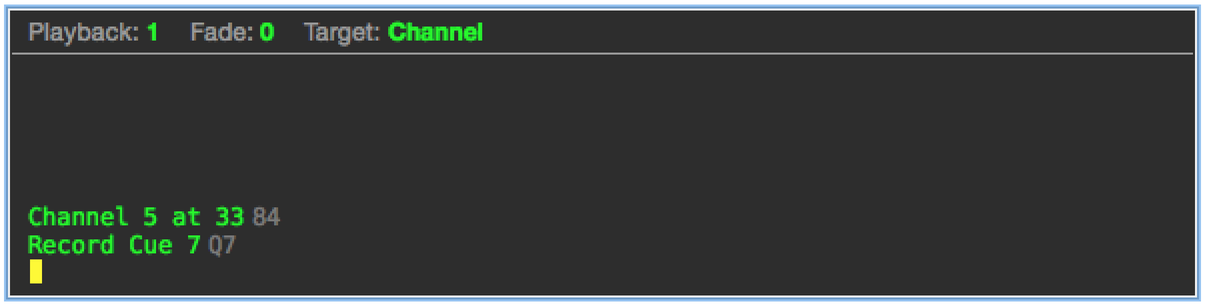
The specific commands available are detailed in the following sections:

- Selection Commands
- Action Commands
- Logic Commands

Executing Commands

There are several places where CueScript commands are used within the system.

Command Line

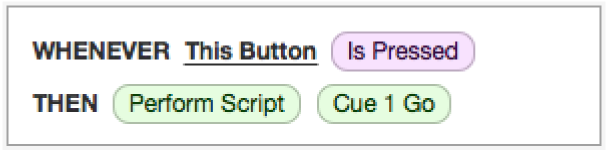


```
Channel 5 at 33 84
Record Cue 7 Q7
```

When working with CueServer Studio on a live CueServer, a command line appears at the bottom of the window. This command line allows CueScript commands to be executed at any time.

Enter a CueScript command (like `Channel 5 at 33` or `Record Cue 7`) and CueServer performs the requested task. Whenever a command is entered, the CueServer replies with a value (which is shown in gray text after the command).

Rules

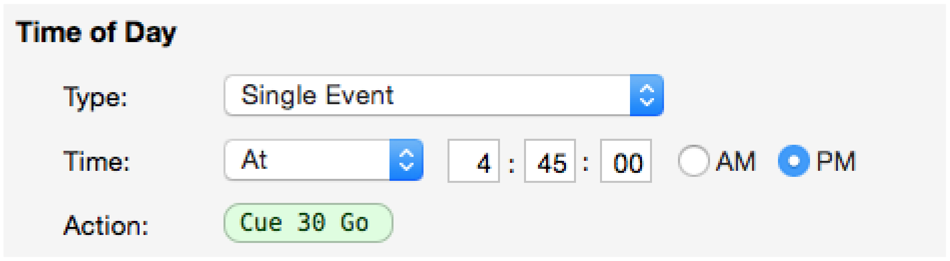


CueServer uses the concept of *rules* to define automation tasks throughout the system. Using CueServer Studio, you can define global rules that are always being monitored for triggering, or you can assign local rules to individual cues, buttons, contacts, and other objects within the system.

A rule takes the form of **Whenever** *something happens* … **Then** *do something*

One of the options in the *then do something* clause of a rule is to perform a CueScript. In the example above, *whenever this button is pressed, execute the command* `Cue 1 Go`.

Actions


Time of Day

Type: Single Event

Time: At 4 : 45 : 00 AM [✓ PM]

Action: Cue 30 Go

Some objects in the system (such as Timers and Macros) are programmed with CueScript actions.

When editing a Timer or Macro, an action field appears, allowing a CueScript command to be entered as the object's action. Whenever the Timer or Macro is triggered, the programmed action is performed.

External Commands

CueScript commands can also be sent to CueServer from an external source by one of the methods listed below:

- via UDP packets
- via HTTP requests
- via RS-232 Serial strings
- via Telnet session

Command Syntax

To make it easy to understand, CueScript uses simple human readable nouns, verbs and objects. These pieces are put together into commands such as `Time 5`, which sets the current fade-time to 5 seconds.

Multiple commands can be strung together to make more complex requests. For example, to change the fade time and set a DMX channel to 50% at the same time, the command `Time 5 Channel 3 at 50` is used.

White spaces in a command (spaces, tabs, new lines, etc.) are ignored by CueServer and are used to simply make the commands more readable. Also, the semicolon (;) can optionally be used between commands on a single line to make commands more readable. CueScript is not case-sensitive, meaning that it doesn't matter if you use upper or lower case letters in a command. All of the following commands are equivalent:

- `Time 5 Channel 3 at 50`
- `time5channel3at50`
- `Time 5; Channel 3 at 50`

-
  ```
  Time 5
  Channel 3 at 50
  ```

Using Abbreviations

Also, to make CueScript more efficient to type and/or send, most CueScript command words may be abbreviated. For example, the `Time` command may be abbreviated as just `T`, `Channel` as `C` and `At` as `@`. For example, the previous example may be abbreviated as:

- `T5;C3@50`

Only a few commands can be abbreviated as a single letter. For instance, the `Cue` command shares the same first letter as the `Channel` command. As documented in the descriptions of each command, the shortest abbreviation for `Channel` is `C`, but the shortest abbreviation for `Cue` is `Cu`. However, some commands also have aliases – the `Cue` command can also be invoked by the single letter `Q`. Therefore, the command `Cue 1 Go` may be abbreviated as `Q1G`.

Expressions

An expression is a combination of symbols including numbers, operators, variables and groupings that are used to specify a mathematical function. Expressions result in a numerical value.

The following are examples of expressions:

- `5`
- `3 + 7`
- `'x' + 4`
- `('x' + 5) - 'y'`
- `(3 + (('x' - 'y') * 12)) - 1`
- `'x' > 9`
- `('x' > 3) and ('y' < 5)`
- `(('myShow' + 1) > 5) or 'maintenanceMode'`

These examples show the use of operators (such as +, -, >, and *and*), variables (such as ‘x’, ‘y’, and ‘maintenanceMode’) and groupings (using parenthesis).

The following sections explain each of these expression components in detail:

- Operators
- Variables
- Grouping

Operators

The CueScript language allows for operators to be used in expressions. Operators are symbols that appear in-between two values that "operate" on those values. Common operators include mathematical functions such as **+** and **–** for addition and subtraction, and boolean functions such as **And**, and **Or**.

Mathematical Operators

The following operators are mathematic, meaning that they perform functions on numbers:

Operator	Function	Example	Result
+	Addition	3 + 5	8
–	Subtraction	5 – 3	2
*	Multiplication	3 * 7	21
/	Division	18 / 3	6

Boolean Operators

The following operators are boolean, meaning that they compare two values in a true or false context. Note that the result of a boolean operator will always be either 0 (meaning false) or 1 (meaning true).

Operator	Function	Examples	Result
=	Equality	5 = 5 3 = 5	1 0
>	Greater Than	5 > 3 3 > 5	1 0
<	Less Than	3 < 5 5 < 3	1 0
and	Logical And	0 and 0 0 and 1 1 and 0 1 and 1	0 0 0 1
or	Logical Or	0 or 0 0 or 1 1 or 0 1 or 1	0 1 1 1

It is important to note that when using boolean operators, any value that is zero is interpreted to mean “false”, and any value that is non-zero is interpreted to mean “true”. Given that any non-zero value is “true”, then the expression `5 and 3` would evaluate to `1`, because both sides of the **and** are both true.

Variables

A variable is a symbol that holds and represents a value. Variable symbols are names such as *x*, *MyVariable*, or *lcd.backlight*. Variables can hold numbers (such as *3* or *12.7*) or strings (such as *Hello World*).

CueServer uses two different kinds of variables: User Variables and System Variables. User variables can be any combination of printable letters, numbers, the underscore (_) or hyphen (-). System variables are similar, but must contain a dot (.) character. The dot character is how the CueServer distinguishes between User and System variables.

Assigning Values to Variables

There are two ways to assign a value to a variable. The first is with the **Assign** command. Here are a few examples:

```
"x" = 3
"MyVariable" = 42
"Message" = "Hello World"
"y" = ('x' + 3)
```

The first line assigns the number *3* to the variable *x*. The second assigns the number *42* to the variable *MyVariable*. The third assigns the string *Hello World* to the variable *Message*. The last example assigns the result of the expression *'x'+3* to the variable *y*.

The second way to assign a value to a variable is with the **Set** command. Here are a few examples:

```
Set x 3
Set MyVariable 42
Set Message "Hello World"
Set y ('x' + 3)
```

These examples are the same as above, except that the **Set** command is used instead of using the **Assign** command.

Using Variable Values

To use variables in CueScript commands, enclose the variable name in single quotes (`'MyVariable'`).

For example, using the variable values set from above, the following variable substitutions would be made:

`Cue 'x' Go`	Executes Cue 3
`Macro 'MyVariable'`	Runs Macro 42
`Set lcd.top 'Message'`	Displays “Hello World” on the top line of the LCD

Using System Variables

Special *System Variables* are used to set the properties of hardware devices, or to change internal behaviors of the CueServer. All system variables include a dot (`.`) in their name, for example `lcd.backlight`, or `universe.priority`.

See the section on System Variables for a description of how to use these built-in variables.

Grouping

Parenthesis are used for grouping expressions. Expression grouping is useful when multiple expressions are strung together in a line and the normal order of operations must be overridden.

The CueScript, operators are always interpreted from left to right. Parenthesis can be inserted into a command string to force different groupings of expressions to be evaluated in a different order.

The following examples illustrate how to use parenthesis to get different results. For these examples, assume *x* = 3 and *y* = 7.

Expression	Result
4 + 2 * 3	18
4 + (2 * 3)	10
Channel ‘x’ + ‘y’	Selects channel 3 and channel 7
Channel (‘x’ + ‘y’)	Selects channel 10

Command Context

CueServer keeps track of the “context” of the currently executing string of CueScript commands, which allows multiple commands which operate on a single object to be split into completely separate requests.

When the user types `Channel 1 At 100`, the user is actually executing two separate commands. The first command, `Channel 1` tells CueServer to select DMX channel 1. The second command, `At 100` tells CueServer to set the currently selected objects (DMX channel 1) to 100%.

The selected objects (in this case, DMX channel 1) are part of the saved command context.

If the user then enters the command `At 75`, CueServer still has DMX channel 1 selected, so channel 1 will be set to 75%.

The command context stores the selected objects (channels, buttons, outputs, etc.), which playback fader is chosen, timing parameters such as fade and follow times and more.

CueServer uses separate command contexts internally to keep the user who is using the live command line in CueServer Studio operating in a different environment from other asynchronous actions that are occurring elsewhere in the system. For instance, if an external process is sending UDP messages to CueServer, these messages get their own command context so they don’t interfere with others using the system. Also, if a timer or button executes in-between when the user selected a channel and set it’s level, this process won’t be disturbed, because each of these asynchronous actions occur in their own context.

Levels

The **At** command and several other methods set levels. Levels are an expression of a quantity from lowest possible value (zero) to highest possible value (full). CueServer allows levels to be expressed in four primary ways, by percentage (the default), or by decimal, hexadecimal or binary notation.

Percentage

By default, when setting DMX channel values, levels are specified by percentage numbers (0, 1, 2, … 98, 99, 100).

For example, to turn a channel completely off, the command `Channel 1 At 0` may be used. To turn a channel completely on, the command `Channel 1 At 100` may be used. Any percentage number in-between 0 and 100 can set a channel to the corresponding level.

For convenience, a percent sign (%) may be added to the number for clarity. For instance, `Channel 1 At 50%`. Using the percent sign is **optional**.

Also for convenience, when specifying a level of 100%, either a value of `100` can be entered or `FL` can be used (meaning “Full”).

Decimal

In some instances, it may be appropriate to use decimal numbers to set DMX channel values (such as when setting levels for moving lights, matching colors or adjusting a level more precisely than percentage levels allow).

Decimal numbers use values from 0 to 255 to specify the range from zero to full.

To use decimal numbers while specifying levels, use a pound sign *before* the level. For example, `Channel 1 At #253`.

Decimal numbers may be used in arrays, such as `Group 1 At {#255, #192, #134}`.

Hexadecimal

In some instances, it may be appropriate to use hexadecimal numbers to set DMX channel values (such as when setting levels for moving lights, matching colors or adjusting a level more precisely than percentage levels allow).

Hexadecimal numbers use digits 0 through 9 and A through F and values from 00 to FF to specify the complete range from zero to full.

To use hexadecimal numbers while specifying levels, use a dollar sign *before* the level. For example, `Channel 1 At $A5`.

Hexadecimal numbers may be used in arrays, such as `Group 1 At {$FF, $C0, $86}`. Note that when specifying hexadecimal numbers to CueServer, always use 2 digits. For example, use $00, $01, $02, not $0, $1, $2, for the single-digit hexadecimal values.

Binary (On/Off)

Some devices being controlled by CueServer only have two states, on and off. In order to simplify their operation, the CueScript language has two extra values named `On` and `Off`. These are used as a convenience to mean the same as 0% and 100%.

Any place that a percentage value can be used in a command, the `On` and `Off` keywords can be used instead. For example, `Channel 1 On`, `Button 2 Off`, `Group 3 On`, `Output * On` are all valid binary-value commands.

Selection Commands

A *selection command* is a type of CueScript command that is used to refer to objects in the system.

Selection commands can be used in conjunction with action commands to perform actions, or selection commands can be used by themselves to query an object's value.

Selecting Objects To Perform Actions

As CueScript is being interpreted by the system, selection commands are used in conjunction with action commands to get things done. First, one or more objects are *selected* by using a selection command, and then one or more *action commands* are used to operate on those selected objects.

For instance, the following CueScript does two things. First, it selects a button. Second, it performs the `On` action.

```
Button 1 On
```

Note that the On action turns "on" the currently selected objects, which in this case happens to be Button 1.

The next CueScript selects playback number 3 with a selection command, then the action command `At 75` sets the playback's submaster to 75%.

```
Playback 3 At 75
```

More than one action can be performed on a selected object. The following example shows the selection command `Channel 1` followed by four action commands: `Time 0`, `At 100`, `Time 5`, and `At 0`. In other words, Channel 1 is selected, then the fade time is set to zero, then Channel 1's value is set to 100%, then the fade time is set to 5 seconds, then Channel 1's value is set to 0%.

```
Channel 1 Time 0 At 100 Time 5 At 0
```

Stringing multiple actions together that refer to the same selected object is a powerful way to express compound actions that you want to apply to one or more objects.

Referring To Objects To Determine Their Value

Another powerful way to use *selection commands* is to refer to one or more object to retrieve their value.

For instance, by executing the command:

```
Channel 1
```

CueServer will not only select Channel 1, but it will also reply with the current value of Channel 1.

Being able to ask CueServer the value of an object is very useful for evaluating expressions. Consider the following command:

```
If (Channel 1 > 50) Then Cue 1 Go
```

The **If .. Then** statement is used with the expression **Channel 1 > 50** to make a decision based on the current value of Channel 1. If the value is greater than 50, then **Cue 1 Go** will occur.

All of the Selection Commands, such as Button, Channel, Contact, Group, Indicator, Output, Playback, and Universe all reply with the current value of their objects.

Referring To Multiple Objects With Different Values

When referring to multiple objects at once, if *all* of the objects have the same value, their shared value will be returned. For instance, if channels 1 through 10 are all set to 50, then the following command will return 50.

```
Channel 1>10
```

But, if the values of channels 1 through 10 have *mixed* values, then the value `-1` will be returned. This special value indicates that the selected objects' values are *mixed*.

Button

Syntax

Command	Description	Return Value
`Button <number> [<range...>]`	Select one or more buttons	The pressed state of the selected button(s)
`Button <station>.<number> [<range...>]`	Select one or more buttons on a specific station	The pressed state of the selected button(s)
`Button ?`	Return the current selection	A selection string

Abbreviation

`B`

Description

Selecting Buttons

The **Button** command selects one or more buttons in the system. Buttons are typically physical pushbuttons on the front of a CueServer or individual buttons on a connected button station. Use the **Button** command in conjunction with an action command like **At**, **On**, **Off**, **Set**, **Enable** or **Disable** to change the properties of one or more buttons. When used alone or in logic expressions, the **Button** command returns the current state of the specified button(s).

Either a single button number can be specified, or a range of buttons can be specified using the various selection operators like +, -, > and ~.

The wildcard character `*` can be used as the button number to mean *all* buttons for a particular station.

Working With Stations

When no station number specified, the default station is assumed. The **Station** command can be used to change the default station. Unless changed by the **Station** command, the default station is typically Station 0, which corresponds to the built-in buttons on the CueServer itself. When a station number is specified as part of the **Button** command, that station number will be used for the selection.

Determining Which Buttons Are Selected

The question mark `?` can be used to ask what the current selection is. A selection string will be returned, which will consist of a single number (like `3`) or a range (like `5>7+9`), or if no buttons are selected, `0` will be returned.

> Note that CueServer treats buttons and indicators very similarly. Buttons have indicators. Buttons are the physical switch that is being pressed by the user and Indicators are the pilot light that shows a button's status. Setting the value of a button or an indicator both sets the indicator's value (turning the indicator on or off). However, getting the value of a button returns the physical switch state (opened or closed), and getting the value of an indicator returns the current state of the indicator (on or off).

Examples

```
Button 1
```
Selects button 1. Future action commands will be directed towards button 1. Also returns `0`, or `1` to indicate if the button is currently unpressed or pressed.

```
Button 1>5 On
```
Turns the LED indicators of buttons 1 thru 5 on.

```
Button 1>3+5>8 Off
```
Turns the LED indicators of buttons 1 through 3 and 5 through 8 off.

```
Button 2.3>5 Enable
```
Enables buttons 3 through 5 on station 2.

```
Button 3.1
Set Button.OnColor {100,50,0}
Set Button.Flash 4
On
Disable
```
Selects button 1 of station 3, then sets the button's *OnColor* property to Orange (RGB color (100,50,0)), then sets the button's *Flash* property to 4, then turns the LED indicator on, then disables button presses from the button.

```
Button 1.* Off
```
Turns the LED indicators of all buttons on station 1 off.

```
Station 5
Button 7 Enable
```

Enables button 7 of station 5.

```
Button ?
```

Returns the current button selection in the format of a single number like `3`, or a range like `5>7+9`.

See Also

- Selection Operators
- **At**, **Disable**, **Enable**, **Off**, **On**, **Set**

Channel

Syntax

Command	Description	Return Value
`Channel <number> [<range...>]`	Select one or more DMX channels	The selected channels' value
`Channel ?`	Return the currently selected DMX channels	A selection string

Abbreviation

`C`

Description

Selecting Channels

The **Channel** command selects one or more DMX channels in the currently active playback fader. DMX channels are the individual control levels sent out of the CueServer to operate connected DMX lighting fixtures. Use the **Channel** command in conjunction with an action command like **At**, **On**, **Off**, **Enable**, **Disable**, **Park**, **Unpark** or **Release** to set channel levels, change the enable or parked state of channels, or release them. When used alone or in logic expressions, the **Channel** command returns the current value of the specified channel(s).

Either a single channel number can be specified, or a range of channels can be specified using the various selection operators like +, -, > and ~.

The wildcard character `*` can be used as the channel number to mean *all* channels in the active playback fader.

Determining Which Channels Are Selected

The question mark `?` can be used to ask what the current selection is. A selection string will be returned, which will consist of a single number (like `3`) or a range (like `5>7+9`), or if no channels are selected, `0` will be returned.

Examples

`Channel 1`
Selects channel 1. Future action commands will be directed towards channel 1. Also returns the channel’s current value between `0` and `255`, or `-1` if the channel is released.

`Channel 1>5 At 33`
Sets channels 1 through 5 to 33%.

`Channel 1>3+5>8 On`
Sets channels 1 through 3 and 5 through 8 to 100%.

@Channel 2+5 Park
Parks channels 2 and 5.

```
Channel 100
Time 0
At 75
Time 5
At 0
```
Selects channel 100, then sets the fade time to 0 (immediate), then sets the channel (100) to 75%, then sets the fade time to 5 (seconds), then sets the channel (100) to 0%.

`Channel 33 At #253`
Sets channel 33 to decimal value 253.

`Channel 44 at $FA`
Sets channel 44 to hexadecimal value $FA.

`Channel ?`
Returns the current channel selection in the format of a single number like `3`, or a range like `5>7+9`.

See Also

- Selection Operators
- **At**, **Disable**, **Enable**, **Off**, **On**, **Park**, **Release**, **Unpark**

Contact

Syntax

Command	Description	Return Value
`Contact <number> [<range...>]`	Select one or more contacts	The closed state of the selected contact(s)
`Contact <station>.<number> [<range...>]`	Select one or more contacts on a specific station	The closed state of the selected contact(s)
`Contact ?`	Return the current selection	A selection string

Abbreviation

`CO`

Description

Selecting Contacts

The **Contact** command selects one or more contacts in the system. Contacts are typically the hard-wired contact closure inputs on a CueServer or external I/O board. Use the **Contact** command in conjunction with an action command like **Enable** or **Disable** to change the enabled state of contacts. When used alone or in logic expressions, the **Contact** command returns the current state of the specified contact(s).

Either a single contact number can be specified, or a range of contacts can be specified using the various selection operators like +, -, > and ~.

The wildcard character `*` can be used as the contact number to mean *all* contacts for a particular station.

Working With Stations

When no station number specified, the default station is assumed. The **Station** command can be used to change the default station. Unless changed by the **Station** command, the default station is typically Station 0, which corresponds to the built-in contacts on the CueServer itself. When a station number is specified as part of the **Contact** command, that station number will be used for the selection.

Determining Which Contacts Are Selected

The question mark `?` can be used to ask what the current selection is. A selection string will be returned, which will consist of a single number (like `3`) or a range (like `5>7+9`), or if no contacts are selected, `0` will be returned.

Examples

```
Contact 1
```

Selects contact 1. Future action commands will be directed towards contact 1. Also returns `0`, or `1` to indicate if the contact is currently opened or closed.

```
Contact 1>5 Disable
```

Disables processing of events on contacts 1 thru 5.

```
Contact 1>3+5>8 Enable
```

Enables processing of events on contacts 1 through 3 and 5 through 8.

```
Station 5
Contact 7 Enable
```

Enables contact 7 of station 5.

```
Contact ?
```

Returns the current contact selection in the format of a single number like `3`, or a range like `5>7+9`.

See Also

- Selection Operators
- **Disable**, **Enable**

Group

Syntax

Command	Description	Return Value
`Group <number> [<range...>]`	Select one or more channel groups	The selected channels' value

Abbreviation

`GR` or `U`

Description

Selecting Groups

The **Group** command selects one or more DMX channels in the currently active playback fader that were stored in the specified group resource. Use the **Group** command in conjunction with an action command like **At**, **On**, **Off**, **Enable**, **Disable**, **Park**, **Unpark** or **Release** to set channel levels, change the enable or parked state of channels, or release them. When used alone or in logic expressions, the **Group** command returns the current value of the specified channel(s).

Either a single group number can be specified, or a range of groups can be specified using the various selection operators like +, -, > and ~.

Examples

`Group 1`
Selects the channels in group 1. Future action commands will be directed towards these channels. Also returns the selected channel's current value between `0` and `255`, or `-1` if the channels are released and/or mixed in value.

`Group 1+5 At 33`
Sets the channels in groups 1 and 5 to 33%.

@Group 2+5 Park
Parks the channels in groups 2 and 5.

```
Group 100
Time 0
At 75
Time 5
At 0
```

Selects the channels in group 100, then sets the fade time to 0 (immediate), then sets the selected channels to 75%, then sets the fade time to 5 (seconds), then sets the selected channels to 0%.

See Also

- Selection Operators
- **At**, **Disable**, **Enable**, **Off**, **On**, **Park**, **Release**, **Unpark**

Indicator

Syntax

Command	Description	Return Value
`Indicator <number> [<range...>]`	Select one or more indicators	The on state of the selected indicator(s)
`Indicator <station>.<number> [<range...>]`	Select one or more indicators on a specific station	The on state of the selected indicator(s)
`Indicator ?`	Return the current selection	A selection string

Abbreviation

`IND`

Description

Selecting Indicators

The **Indicator** command selects one or more indicators in the system. Indicators are typically the LED indicators of pushbuttons on the front of a CueServer or individual indicators on a connected button station. Use the **Indicator** command in conjunction with an action command like **At**, **On**, **Off** or **Set** to change the indication state of one or more indicators. When used alone or in logic expressions, the **Indicator** command returns the current state of the specified indicator(s).

Either a single indicator number can be specified, or a range of indicators can be specified using the various selection operators like +, -, > and ~.

The wildcard character `*` can be used as the indicator number to mean *all* indicators for a particular station.

Working With Stations

When no station number specified, the default station is assumed. The **Station** command can be used to change the default station. Unless changed by the **Station** command, the default station is typically Station 0, which corresponds to the built-in indicators on the CueServer itself. When a station number is specified as part of the **Indicator** command, that station number will be used for the selection.

Determining Which Indicators Are Selected

The question mark `?` can be used to ask what the current selection is. A selection string will be returned, which will consist of a single number (like `3`) or a range (like `5>7+9`), or if no indicators are selected, `0` will be returned.

Note that CueServer treats buttons and indicators very similarly. Buttons have indicators. Buttons are the physical switch that is being pressed by the user and Indicators are the pilot light that shows a button's status. Setting the value of a button or an indicator both sets the indicator's value (turning the indicator on or off). However, getting the value of a button returns the physical switch state (opened or closed), and getting the value of an indicator returns the current state of the indicator (on or off).

Examples

```
Indicator 1
```
Selects indicator 1. Future action commands will be directed towards indicator 1. Also returns `0`, or `1` to indicate if the indicator is currently off or on.

```
Indicator 1>5 On
```
Turns indicators 1 thru 5 on.

```
Indicator 1>3+5>8 Off
```
Turns indicators 1 through 3 and 5 through 8 off.

```
Indicator 3.1
Set Indicator.OnColor (100,50,0)
Set Indicator.Flash 4
On
```
Selects indicator 1 of station 3, then sets the indicator's *OnColor* property to Orange (RGB color (100,50,0)), then sets the indicator's *Flash* property to 4, then turns the indicator on.

```
Indicator 1.* Off
```
Turns indicators of all buttons on station 1 off.

```
Station 5
Indicator 7 On
```
Turns indicator 7 of station 5 on.

`Indicator ?`

Returns the current indicator selection in the format of a single number like `3`, or a range like `5>7+9`.

See Also

- Selection Operators
- **At**, **Off**, **On**, **Set**

Output

Syntax

Command	Description	Return Value
`Output <number> [<range...>]`	Select one or more outputs	The state of the selected output(s)
`Output <station>.<number> [<range...>]`	Select one or more outputs on a specific station	The state of the selected output(s)
`Output ?`	Return the current selection	A selection string

Abbreviation

`O`

Description

Selecting Outputs

The **Output** command selects one or more outputs in the system. Outputs are typically the hard-wired digital outputs on a CueServer or external I/O board. Use the **Output** command in conjunction with an action command like **On**, **Off**, or **At** to change the output state of one or more outputs. When used alone or in logic expressions, the **Output** command returns the current state of the specified output(s).

Either a single output number can be specified, or a range of outputs can be specified using the various selection operators like +, -, > and ~.

The wildcard character `*` can be used as the output number to mean *all* outputs for a particular station.

Working With Stations

When no station number specified, the default station is assumed. The **Station** command can be used to change the default station. Unless changed by the **Station** command, the default station is typically Station 0, which corresponds to the built-in outputs on the CueServer itself. When a station number is specified as part of the **Output** command, that station number will be used for the selection.

Determining Which Outputs Are Selected

The question mark `?` can be used to ask what the current selection is. A selection string will be returned, which will consist of a single number (like `3`) or a range (like `5>7+9`), or if no outputs are selected, `0` will be returned.

Examples

`Output 1`
Selects output 1. Future action commands will be directed towards output 1. Also returns `0`, or `1` to indicate if the output is currently off or on.

`Output 1>5 On`
Turns on outputs 1 thru 5.

`Output 1>3+5>8 Off`
Turns off outputs 1 through 3 and 5 through 8.

`Output 7 At 50`
Sets the level of output 7 to 50% (any non-zero level turns an output on).

`Output ?`
Returns the current output selection in the format of a single number like `3`, or a range like `5>7+9`.

See Also

- Selection Operators
- **At**, **Off**, **On**

Playback

Syntax

Command	Description	Return Value
`Playback <number>`	Change the active playback fader	The new playback number
`Playback ?`	Return the currently active playback fader	The current playback number

Abbreviation

`P`

Description

Choosing The Active Playback

The **Playback** command changes the currently active playback fader and/or allows a playback fader's properties to be changed. Playback faders are the functional units in the DMX output stack that control the playback of cues, streams and maintain the fade progress and timing of linked cues. Each playback fader operates as an independent *layer* of control in the DMX output stack and has properties that control how each playback layer is merged with the preceding layer, and the overall intensity of the channels in the layer. Use the **Playabck** command to change the active playback fader, or in conjunction with an action command like **At**, **On**, **Off** or **Set** to change the playback's submaster level, or to set the layer properties. When used alone or in logic expressions, the **Playback** command returns the currently active playback fader.

Determining The Active Playback

The question mark `?` can be used to ask what the currently active playback is. A number will be returned, from `1` to `32` indicating which playback is currently active.

Many commands operate on the currently active playback fader, such as **Channel**, **Clear**, **Cue**, **Fade**, **Follow**, **Go**, **Group**, **Link**, **Stack**, **Start** and **Stop**. Since each playback fader maintains its own set of DMX channels and properties, such as the current and next cue, fade and follow times, cue link, and more, it is important to make sure that you use the **Playback** command to specify which playback you are targeting when using the above commands.

Although CueServer can have a maximum of 32 playback faders, your configuration may have fewer, depending on the combination of playbacks and DMX universes that you have chosen. If you select a playback that is not available, the subsequent commands sent to that playback will have no effect.

Examples

```
Playback 1
```
Makes playback 1 active. All future playback related commands will be directed to playback 1.

```
Playback 2 At 75
```
Makes playback 2 active and sets the playback's submaster to 75%.

```
Playback 3
Cue 1 Go
```
Makes playback 3 active, then executes cue 1 in playback 3.

```
Playback 4
set Playback.Mode "Override"
```
Makes playback 4 active, then sets the playback's layer mode to "override".

See Also

- **At**, **Off**, **On**, **Set**

Station

Syntax

Command	Description	Return Value
`Station <number>`	Select one or more stations	The selected stations
`Station ?`	Return the current selection	The selected stations

Abbreviation

`STAT`

Description

When selecting Buttons, Contacts, Indicators, or Outputs on a connected station, the **Station** command can be used to specify a station number instead of specifying it as part of the Button, Contact, Indicator or Output number. The **Station** command actually sets the default station(s) to be used with any of the other selection commands when a station number is not used with the selection command. See the examples below for clarification.

Two Ways To Select Station Objects

There are two ways to specify a station object. The first is to use dotted-notation with the selection command. For example, to select Button 7 of Station 3, the command `Button 3.7` can be used. The the “dot” is used to separate the station number from the object number.

The **Station** command provides a second way to select station objects. Using this method, the **Station** command is used first to change which station (or stations) are the default, and then use the object command to select the individual object. For example, to select Button 7 of Station 3, the command `Station 3 Button 7` can be used.

Selecting The Same Object On Multiple Stations

The **Station** command allows the same object on multiple stations to be selected at the same time.

For example, to select Button 3 of Stations 1>10, the command `Station 1>10 Button 3` can be used.

Examples

```
Station 3 Contact 4
```

Sets Station 3 as the default station, and then selects Contact 4 on the default station (Station 3).

```
Station 4 Output 1>10
```

Sets Station 4 as the default station, and then selects Outputs 1 through 10 on the default station (Station 4).

```
Station 1>5 Button 8 Off
```

Sets Stations 1 through 5 as the default stations, and then selects Button 8 on the default stations (Stations 1>5), and then turns these button indicators off.

```
Station 4
Button 1 On
Button 2 Off
Button 3 On
```

Sets Station 4 as the default station, then turns Button 1 On, Button 2 Off and Button 3 On (all on Station 4).

See Also

- Selection Operators
- **Button**, **Contact**, **Indicator**, **Output**

Universe

Syntax

Command	Description	Return Value
`Universe <number> [<range...>]`	Select one or more universes	*None*
`Universe ?`	Return the current selection	A selection string

Abbreviation

`UNIV`

Description

Selecting Universes

The **Universe** command selects one or more universes in the system. Universes are the logical blocks of 512 DMX channels that are sent and/or received by the CueServer across the Ethernet network. Use the **Universe** command in conjunction with an action command like **Enable**, **Disable**, or **Set** to enable/disable the universe or to change the universe's properties (such as it's broadcast priority). When used alone or in logic expressions, the **Universe** command returns `0` or `1` to indicate is the universe is disabled or enabled.

Either a single universe number can be specified, or a range of universes can be specified using the various selection operators like +, -, > and ~.

The wildcard character `*` can be used as the universe number to mean *all* universes.

Determining Which Universes Are Selected

The question mark `?` can be used to ask what the current selection is. A selection string will be returned, which will consist of a single number (like `3`) or a range (like `5>7+9`), or if no universes are selected, `0` will be returned.

Examples

`Universe 1`

Selects universe 1. Future action commands will be directed towards universe 1. Also returns `0`, or `1` to indicate if the universe is currently disabled or enabled.

`Universe 3 Disable`

Disables the transmission of universe 3.

`Universe 1>3+5>8 Enable`

Enables universes 1 through 3 and 5 through 8.

```
Universe 2
Set Universe.Priority 150
```

Sets the “priority” property of Universe 2 to 150.

See Also

- Selection Operators
- **Disable**, **Enable**, **Set**

Selection Operators (+, -, >, ~)

Most of the selection commands allow more than one object to be selected at once. To select more than one of a particular object, you can use the **Plus**, **Minus**, **Thru** and **Invert** operators.

These operators work on most of the selection commands, including:

- Button
- Channel
- Contact
- Group
- Indicator
- Output
- Universe

Plus (+)

Use the **Plus** (+) operator to add additional objects to your selection.

For example:

```
Channel 1+3+5+7
```

Selects channels 1, 3, 5 and 7.

```
Channel 1>10+20>30
```

Selects channels 1 thru 10 and 20 thru 30.

```
Group 1+5
```

Selects the channels in Group 1 and Group 5.

Minus (-)

Use the **Minus** (-) operator to remove objects from your selection.

For example:

```
Channel 1>10-5
```

Selects channels 1 thru 10, except for channel 5 (or, in other words, channels 1 thru 4 and 6 thru 10).

```
Group 1-3
```

Selects the channels in Group 1 that are not in Group 3.

Thru (>)

Use the **Thru** (>) operator to add a range of objects to your selection.

For example:

```
Channel 1>10
```

Selects channels 1 thru 10.

```
Channel 1>50-20>30
```

Selects channels 1 thru 50, except for channels 20 thru 30 (or, in other words, channels 1 thru 19 and 31 thru 50).

Invert (~)

Use the **Invert** (~) operator to invert which objects are selected.

For example:

```
Button 3
On
~
Off
```

Selects button 3, then turns it’s indicator on, then inverts the selection (selecting all buttons except for button 3), then turns those indicators off.

Using Wildcards

When selecting objects, a *wildcard* operator is available as a shortcut for selecting *all* objects of a particular type.

The wildcard operator is an asterisk character (`*`).

This character can be inserted in most places that a selection range is required, which means to select *all* of a particular object.

The following table shows how the wildcard operator can be used:

Command	**Result**
Channel * Release	Releases all channels in the selected playback
Button * On	Turns on all button indicators on the default station
Button 3.* Off	Turns off all button indicators on Station 3
Output * Off	Turns off all outputs on the default station
Universe * Enable	Enabled all universe outputs

Action Commands

Action commands perform actions. Some action commands operate on the current selection (as set by the Selection Commands), and some action commands perform a global action that does not depend on selected objects.

For example, the **At** command operates on selected channels, buttons, playbacks, outputs and more. In order to properly use the **At** command, one of these objects must be selected first. The following examples show some of the proper uses of **At**:

- `Channel 1 At 75`
- `Button 1>8 At 0`
- `Playback 3 At FL`
- `Group 1+3+5+7 At 95`

Other action commands, such as the **Audio** command do not depend on other objects being selected first. The following examples show how the **Audio** command can be used to start and stop playing sounds.

- `Audio "Chime.wav"`
- `Audio "Breakbeat.mp3"`
- `Audio Stop`

All of the available action commands are detailed in the following sections.

Assign (=)

Syntax

Command	Description	Return Value
`<variable> = <value>`	Sets the value of the variable	The value the variable was set to

- `<variable>`
    - A user variable or system variable name. Must be enclosed in double quotes.
- `<value>`
    - A string (a combination of characters enclosed in quotes, such as “Hello World”).
    - A number (a whole number, or a decimal number, such as *123* or *12.7*).

Alternate Syntax

See: Set command.

Description

Setting Values

The **Assign** command sets the value of a variable. The variable can be user defined (such as *xyz*, *LoopCount*, or *IsMyShowEnabled*), or it may be a system variable (such as *button.onColor* or *lcd.backlight*. System variables always contain a “dot” character (`.`). User variables must not contain a “dot” character, otherwise, they will be interpreted as a system variable, and they will not be stored properly.

User variables can be defined on the fly, simply by assigning a value to a variable. There is no need to pre-define variables.

See the **Assign** command for an alternate syntax for assigning variable values.

See the **Variables** section for how to use variables in the script language.

See the **System Variables** section for a complete list of available system variables.

Examples

`"x"=3`
Sets the variable *x* to the number `3`.

`"text"="Hello World"`
Sets the variable *text* to the string `Hello World`.

`"lcd.backlight"=25`
Sets the system variable *lcd.backlight* to 25%.

`"y"=('x' + 1)`
Sets the variable *y* to the result of the expression `'x' + 1`.

See Also

- **Set**, **System Variables**

Audio

Syntax

Command	Description	Return Value
`Audio "<filename>"`	Plays the given sound file	The file name being played
`Audio Stop`	Stops playing sound immediately	Always returns `0`

- `<filename>`
  - The file name of a sound resource loaded into *Sounds*
  - Sound formats recognized include: .aif, .mp3, .ogg, .snd, and .wav

Abbreviation

None

Description

Playing Sounds

The **Audio** command will play a given sound resource file to the Audio Output jack. The sound plays asynchronously, meaning that the command returns immediately while the sound plays in the background until it is finished, or it is interrupted by the **Audio Stop** command.

If a sound was already playing, issuing another **Audio** command will immediately stop playing the previous sound and begin playing the new sound.

Stopping Sounds

The **Audio Stop** command will immediately stop any sound that is currently playing.

Examples

`Audio "Sound Effect.wav"`
Begins playing the *Sound Effect.wav* sound resource file.

`Audio "Background Music.mp3"`

Begins playing the *Background Music.mp3* sound resource file.

`Audio Stop`

Stops playing sound immediately.

At

Syntax

Command	Description	Return Value
`At <value>`	Set the value of the selected object(s)	The value the object(s) were set to
`At Cue <cue>`	Sets the selected channels to the values in Cue *cue*	The number of channels set
`At Playback <playback>`	Sets the selected channels to the values in Playback *playback*	The number of channels set

- `<value>`
    - A percentage from `0` to `100`. When specifying percentages, the value can optionally be followed by the `%` sign.
    - A decimal number from `#0` to `#255`. When specifying decimal numbers, the value must be proceeded with a `#` sign.
    - A hexadecimal number from `$00` to `$FF`. When specifying hexadecimal numbers, the value must be proceeded with a `$` sign.
    - `FL` (Full) or `On` can be used as a shortcut that means `100%`
    - `Off` can be used as a shortcut that means `0%`
- `<cue>`
    - Any whole number from `0` to `99999`
    - May optionally contain decimal numbers from `.00` to `.99`
- `<playback>`
    - Any whole number from `1` to `32`

Abbreviation

`A` or `@`

Description

Setting Values

The **At** command sets the currently selected object(s) values. The **At** command can be used with many types of objects, including **Buttons**, **Channels**, **Groups**, **Outputs**, and **Playbacks**.

The following table shows how the **At** command affects each of these object types:

Object	Result Of At Command
Buttons	Any non-zero value turns the button's LED indicator on
Channels	The channel is set to the specified value
Groups	The channels in the group are set to the specified value
Outputs	Any non-zero value turns the output on
Playbacks	The playback's submaster is set to the specified value

Setting Values With Timing

A playback fader's **Time** parameter will cause the value of **Channels**, **Groups** and **Playbacks** to fade to the desired value. A time of 0 (zero) causes the value to be set immediately. Any non-zero time will cause the value to gradually change at a speed that will cause it to reach the desired value in the number of seconds set by the **Time** command.

Recalling Values From A Cue

Using the **At Cue** command allows the data from a given Cue to be used to set the currently selected channel values (instead of specifying a single fixed value). This is useful to only recall parts of a Cue without affecting other channels.

Recalling Values From A Playback

Using the **At Playback** command allows the channels from a given Playback to be used to set the currently selected channel values (instead of specifying a single fixed value). This is useful to copy components of a scene from one playback fader to another.

Examples

```
Channel 1 At 33
```
Sets the value of channel 1 to 33%.

```
Channel 1>3+5>8 On
```
Sets channels 1 through 3 and 5 through 8 to 100%.

```
Channel 1>10
Time 5
```

`At FL`

Selects channels 1 through 10, then changes the fade time to 5 seconds, then begins fading the channels to 100% (Full).

`Group 5 On`

Sets the channels in group 5 to 100%.

`Channel 100>200 At Cue 44`

Sets channels 100 through 200 to the channel levels recorded in Cue 44.

`Group 7 At Playback 8`

Sets the channels in Group 7 to the channel levels currently in Playback 8.

See Also

- **Button**, **Contact**, **Group**, **Output**, **Playback**

Clear

Syntax

Command	Description	Return Value
`Clear`	Clears the active playback fader	The playback number cleared

Abbreviation

`CL`

Description

The **Clear** command clears the active playback fader.

Clearing a playback fader has the following effect:

- Releases all DMX channels (including parked channels)
- Removes the current selection
- Sets the current and next cue to *none*
- Aborts any fade or follow timers in progress
- Zeros the current fade, follow and link properties
- Returns the playback's submaster to 100%

The **Clear** command also clears parked channels. To clear channels without clearing parked channels, use the **Release** command instead.

Examples

```
Playback 1 Clear
```
Clears playback 1.

See Also

- **Playback**, **Release**

Cue

Syntax

Command	Description	Return Value
`Cue <cue number>`	Sets the active playback fader's next cue	The cue number set
`Cue ?`	Returns the current cue in the active playback fader	The current cue number

- `<cue number>`
    - Any whole number from `0` to `99999`
    - May optionally contain decimal numbers from `.00` to `.99`

Abbreviation

`Q` or `CU`

Description

Setting The Next Cue

Use the **Cue** command to set the *next cue* in the active playback fader. Whenever the playback fader receives a **Go** command, it will advance to this next cue. The **Cue** command is frequently used in conjunction with the **Go** command. For instance, the commands `Cue 1 Go` are typically used together, even though they are two distinct commands. The first command, **Cue 1** sets which cue is "next", and then the **Go** proceeds to execute it.

Setting The Next Cue With Overrides

When the **Cue** command is executed, not only is the next cue number placed into the playback fader, but the cue's fade and follow times and link cue are also loaded into the playback. This allows for manually overriding the timing or link before the **Go** command is executed. For instance, the commands `Cue 1 Fade 5 Follow 10 Go` would first load cue 1 as the next cue for the playback, then the fade time would be changed to 5 seconds, then the follow time would be changed to 10 seconds, and then the cue would be executed with the new timing substituted into place of the default values for the cue.

Working With Cue Stacks

By default, all cues are loaded from the main cue list. Additionally, the show file may contain one or more *cue stacks*. The **Stack** command is used to change which cue stack the playback fader is using. Once the stack has been changed on a playback fader, all cues on that playback will be loaded from that cue stack.

Determining The Current Cue

Use the **Cue** command with the question mark (?) to return the current cue number of the active playback fader. For instance, if playback 3 currently executing cue 7, then executing the commands `Playback 3 Cue ?` will return `7`.

If a playback isn't loaded with a cue, the return value will be negative (less than zero).

Examples

```
Cue 1
```
Sets cue 1 as the next cue in the active playback fader.

```
Cue 7 Go
```
Executes cue 7 in the active playback fader by first loading cue 7 then executing it.

```
Playback 3 Cue 100.5 Go
```
Executes cue 100.5 in playback 3.

```
Cue 999.99 Fade 5 Go
```
Loads cue 999.99, then overrides the fade time to 5 seconds, then executes it.

```
Playback 5
Stack "Intro"
Cue 1 Go
```
Sets playback 5 as the active playback, then switches the playback to use the stack named “Intro”, then executes Cue 1 from the “Intro” stack.

See Also

- **Fade**, **Follow**, **Go**, **Link**, **Playback**, **Stack**

Delete

TODO

Disable

Syntax

Command	Description	Return Value
`Disable`	Disables the selected object(s)	The number of objects disabled

Abbreviation

`DIS`

Description

The **Disable** command disables the currently selected object(s). The **Disable** command can be used with many types of objects, including **Buttons**, **Channels**, **Contacts**, **Groups**, and **Universes**. **Disable** has the opposite effect as **Enable**.

The following table shows the various effect of enabling or disabling an object:

Object	When Enabled	When Disabled
Buttons	Responds to presses normally	Does not trigger any actions
Channels	Normal output from playback	No output from playback
Contacts	Responds to closures normally	Does not trigger any actions
Groups	Normal output from playback	No output from playback
Universes	Normal broadcast from universe	No broadcast from universe

Examples

`Button 1 Disable`
Disables button 1.

`Channel 1>10 Disable`
Disables channels 1 thru 10.

```
Universe 1+7 Disable
```

Disables universes 1 and 7.

See Also

Button, **Channel**, **Contact**, **Group**, **Universe**

Enable

Syntax

Command	Description	Return Value
`Enable`	Enables the selected object(s)	The number of objects enabled

Abbreviation

`ENA`

Description

The **Enable** command enables the currently selected object(s). The **Enable** command can be used with many types of objects, including **Buttons**, **Channels**, **Contacts**, **Groups**, and **Universes**. **Enable** has the opposite effect as **Disable**.

The following table shows the various effect of enabling or disabling an object:

Object	When Enabled	When Disabled
Buttons	Responds to presses normally	Does not trigger any actions
Channels	Normal output from playback	No output from playback
Contacts	Responds to closures normally	Does not trigger any actions
Groups	Normal output from playback	No output from playback
Universes	Normal broadcast from universe	No broadcast from universe

Examples

`Button 1 Enable`
Enables button 1.

`Channel 1>10 Enable`
Enables channels 1 thru 10.

`Universe 1+7 Enable`

Enables universes 1 and 7.

See Also

Button, **Channel**, **Contact**, **Group**, **Universe**

Fade

Syntax

Command	Description	Return Value
`Fade <cue fade time>`	Sets the active playback fader’s cue fade time	The cue fade time set
`Fade ?`	Returns the current cue fade time of the active playback fader	The current cue fade time

- `<cue fade time>`
  - A decimal number of seconds (optionally using decimal digits for fractions of seconds)
  - `0` means no fade time (or the channels are set immediately without fading)
  - Optionally may include a slash (`/`) which indicates a split (up/down) fade
  - Optionally may include a dash (`-`) which indicates a delayed fade

Abbreviation

`FA`

Description

Setting The Cue Fade Time

Use the **Fade** command to set the *cue fade time* for the active playback fader. This time is used to crossfade the channels of the next cue whenever the **Go** command is executed. The cue fade time is automatically set by cues being loaded into the playback fader, but the **Fade** command can be used to override the cue fade time.

Using Split Fade Times

A *split fade time* is used when it is desired to have channels that are fading up occur at a different rate than channels fading down. To specify a split fade time, use a slash character in between two fade times. For instance, the command `Fade 3.5/7.5` will cause any channel that is fading up to occur in 3.5 seconds, and any channel that is fading down to occur in 7.5 seconds.

Using Fade Delays

Normally, whenever a cue is executed, the fade begins immediately. A delay can be inserted that would cause the fade to be delayed before starting to change value. To specify a fade delay, use a delay time and dash character before the fade time. For instance, the command `Fade 5.5-10` will cause the fade to delay 5.5 seconds before beginning a 10 second fade.

Using Both Fade Delays And Split Fade Timing

Both fade delays and split fades can be combined. For instance, the command `Fade 1-2/3-4` would cause any channels fading up to be delayed 1 second before fading over 2 seconds, while the downward fading channels would be delayed 3 seconds before fading over 4 seconds.

Determining The Current Cue Fade Time

Use the **Fade** command with the question mark (?) to return the current cue fade time. A cue fade time such as `7.21` or `12/3` will be returned.

> Note that the *Cue Fade Time* is different from the *Global Fade Time*. The cue fade time effects the **Go** command. The global fade time effects the **At** command. The cue fade time is set with the **Fade** command.

Examples

`Fade 1`
Sets the cue fade time to 1 second.

`Fade 1.35/7.2`
Sets the cue fade time to 1.35 seconds for upward fading channels and 7.2 seconds for downward fading channels.

`Cue 22 Fade 5 Go`
Loads cue 22, then overrides it’s fade time to 5 seconds before executing it.

See Also

- **Cue**, **Go**, **Time**

Follow

Syntax

Command	Description	Return Value
`Follow <cue follow time>`	Sets the active playback fader’s cue follow time	The cue follow time set
`Follow Clear`	Clears the active playback fader’s cue follow time	*None*
`Follow ?`	Returns the current cue follow time of the active playback fader	The current cue follow time

- `<cue follow time>`
  - A decimal number of seconds (optionally using decimal digits for fractions of seconds)
  - `0` means no follow time

Abbreviation

`FO`

Description

Setting The Cue Follow Time

Use the **Follow** command to set the *cue follow time* for the active playback fader. Whenever a **Go** occurs, this time is used to start a timer that will automatically execute another **Go** as soon as the timer expires. The cue follow time is automatically set by cues being loaded into the playback fader, but the **Follow** command can be used to override the cue follow time.

Clearing The Cue Follow Time

Sometimes, it may be useful to cancel the follow timer in a playback fader. Use the **Follow Clear** command to clear any currently running follow timer in the active playback fader.

Determining The Current Cue Follow Time

Use the **Follow** command with the question mark (?) to return the current cue follow time. A cue follow time such as `1` or `7.21` will be returned.

Examples

```
Follow 1
```
Sets the cue follow time to 1 second.

```
Cue 22 Follow 5 Go
```
Loads cue 22, then overrides it’s follow time to 5 seconds before executing it.

```
Follow Clear
```
Clears any currently running follow timer in the active playback fader.

See Also

- **Cue**, **Go**

Go

Syntax

Command	Description	Return Value
`Go`	Executes the next cue in the active playback fader	The cue number executed

Abbreviation

`G`

Description

Going To The Next Cue

Use the **Go** command to execute the *next cue* in the active playback fader. The next cue is typically the cue with the next numerically higher number in the cue list, but the next cue can be overridden by an optional link in the cue or executing the **Cue** command.

After each **Go** occurs, the properties of the next cue are loaded into the playback fader. These properties include the next cue's fade and follow times and the cue's link.

Timing For Normal Cues

When the next cue is executed with the **Go** command, the playback's Fade time is used to crossfade to the channels recorded in the cue. The playback's Follow time is used to start a timer that, when expired, will automatically "follow" to the next cue by automatically executing another **Go**.

It is important to note that a cue's Fade and Follow times are started at the same time. For instance, if a cue has a fade of 3 seconds and a follow of 4 seconds, then the fade will complete 3 seconds after the cue started, and the follow will occur 4 seconds after the cue started (or 1 second after the fade completes). This means that if the follow time is shorter than the fade time, the fade will not fully complete before the follow occurs.

Timing For Streaming Cues

Streaming cues do not have a fade time, but they do use a follow time. If a follow time is specified, the stream will only play until the follow time is reached and then will automatically “follow” to the next cue by automatically executing another **Go**.

Streams have various playback modes that affects what action is taken when the end of the stream is reached. These modes include Follow, Loop, Hold and Blackout. Please refer to the section about streaming cues for more details.

Links

If a cue has an optional link, then when it is loaded into a playback fader, the playback’s Link property is set. Whenever a **Go** occurs on that playback, if the playback has a link set, then instead of advancing to the next sequential cue, the linked cue will be loaded.

Examples

`Go`
Advances to the next cue in the active playback fader.

`Cue 7 Go`
Executes cue 7 in the active playback fader by first loading cue 7 then executing it.

`Playback 3 Cue 100.5 Go`
Executes cue 100.5 in playback 3.

See Also

- **Cue**, **Fade**, **Follow**, **Link**, **Playback**, **Stack**

Input

Syntax

Command	Description	Return Value
`Input Disable`	Disables the DMX Input into the Playback Faders	`0`
`Input Enable`	Enables the DMX Input into the Playback Faders	`1`
`Input ?`	Returns the current enable state of the DMX Input	`0` or `1`

Abbreviation

`IN`

Description

The **Input** command is used to either enable or disable the DMX Input layer into the Playback Fader stack. By disabling the DMX Input, no incoming DMX channels from Ethernet or hardwired DMX will flow into the Playback Fader stack. Use the **Input Disable** command to ignore DMX Input. Use the **Input Enable** command to resume the reception of DMX Input.

Use the **Input ?** command to return the current enable state of DMX Input.

Examples

`Input Disable`
Disables the DMX Input into the Playback Fader stack.

`Input Enable`
Resumes normal DMX Input into the Playback Fader stack.

`Input ?`
Returns either `0` or `1`, indicating if the DMX Input is currently disabled or enabled.

Link

Syntax

Command	Description	Return Value
`Link <cue number>`	Sets the active playback fader’s linked cue number	The linked cue
`Link Clear`	Clears the active playback fader’s linked cue	*None*
`Link ?`	Returns the current linked cue of the active playback fader	The current linked cue

- `<cue number>`
    - Any whole number from `0` to `99999`
    - May optionally contain decimal numbers from `.00` to `.99`

Abbreviation

`L`

Description

Setting The Linked Cue

Use the **Link** command to set the *linked cue* for the active playback fader. Whenever a **Go** occurs, this link is used to override the normal sequential execution of cues. If no link is set, cues execute in numerical order. If the link is set to a cue, then this cue will become the next cue after the **Go**.

Clearing The Linked Cue

Use the **Link Clear** command to clear any linked cue in the active playback fader. Without a linked cue, future **Go** commands will execute cues in numerical order.

Determining The Current Linked Cue

Use the **Link** command with the question mark (?) to return the current linked cue. A cue number such as `1` or `100.5` will be returned. If no cue is linked, `-1` is returned.

Examples

`Link 1`

Sets the linked cue to cue 1.

`Cue 22 Link 1 Go`

Loads cue 22, then overrides it’s linked cue to cue 1 before executing it.

`Link Clear`

Clears any currently linked cue from the active playback fader.

See Also

- **Cue**, **Go**

Load

TODO

Log

Syntax

Command	Description	Return Value
`Log <string>`	Writes a message to the system log	The message is returned
`Log Clear`	Clears the new message indicator	The number of messages cleared
`Log ?`	Returns the current new message count	The number of new messages pending

Abbreviation

None

Description

The **Log** command writes a message to the system log. When new messages are written to the system log, the device's power/status LED will blink with a magenta color.

Using **Log Clear** will acknowledge new messages by clearing the new message indicator.

Using **Log ?** will return the number of new messages in the system log.

Examples

```
Log "This is a test"
```

Writes the string “This is a test” to the system log. The power/status indicator will begin to blink, indicating that new messages have been added to the system log.

```
Log Clear
```

Clears the “new message” indicator.

```
Log ?
```

Returns the number of new messages in the system log.

Macro

Syntax

Command	Description	Return Value
`Macro <number>`	Executes the CueScript commands stored in the specified macro	The result of the last command in the macro

Abbreviation

`M`

Description

The **Macro** command executes the CueScript instructions stored in a macro. A macro is a single command that expands automatically into a set of commands to perform a particular task. Macros are defined within CueServer Studio. When the macro command is executed, all of the commands defined in the macro are executed in it's place.

For instance, if Macro 1 is defined to include the commands `Time 5 At 100`, then from somewhere else the command `Channel 7 Macro 1` were executed, the result would be to select channel 7, then change the fade time to 5 seconds and set channel 7's level to 100%.

Macros can contain an arbitrary number of CueScript commands, and may even call upon other macros. When macros call upon other macros, this is called "nesting".

Take care to not create infinite loops by having one macro call upon another macro, which in turn calls upon the first macro. This will create an "infinite loop".

A common mistake is to use the **Go** command in conjunction with the **Macro** command (for example: `Macro 1 Go`). The **Macro** command does not need **Go** in order to function. The CueServer will interpret `Macro 1 Go` as two separate commands, the first will be to execute Macro 1, the second will be to make the active playback fader step to the next cue, which is a valid combination of commands, but usually not intended.

Examples

```
Macro 3
```

Executes the CueScript commands stored in macro 3.

New

TODO

Off

Syntax

Command	Description	Return Value
`Off`	Turn an object off	`0`

Abbreviation

None

Description

The `Off` command sets the currently selected object(s) values to the minimum. In other words, it turns the object(s) “off”.

Note that `Off` is simply an alias for the command `At 0%`.

Examples

`Button 1>5 Off`
Turns the LED indicators of buttons 1 thru 5 off.

`Channel 1>3+5>8 Off`
Sets channels 1 through 3 and 5 through 8 to 0%.

`Group 5 Off`
Sets the channels in group 5 to 0%.

See Also

Button, **Contact**, **Group**, **Output**, **Playback**

On

Syntax

Command	Description	Return Value
`On`	Turn an object on	`255`

Abbreviation

None

Description

The `On` command sets the currently selected object(s) values to the maximum. In other words, it turns the object(s) “on”.

Note that `On` is simply an alias for the command `At 100%`.

Examples

```
Button 1>5 On
```
Turns the LED indicators of buttons 1 thru 5 on.

```
Channel 1>3+5>8 On
```
Sets channels 1 through 3 and 5 through 8 to 100%.

```
Group 5 On
```
Sets the channels in group 5 to 100%.

See Also

Button, **Contact**, **Group**, **Output**, **Playback**

Park

Syntax

Command	Description	Return Value
`Park`	Parks the selected channel(s)	*None*

Abbreviation

None

Description

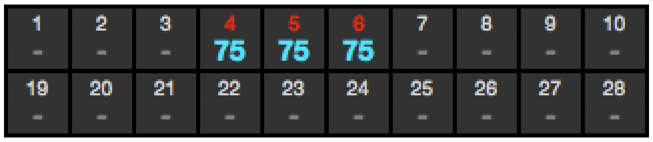
The **Park** command locks the current value of the selected channel(s) in the active playback fader. Parked channels cannot be changed by running cues or using the **At** command. Use the **Park** command to freeze one or more channels so future cues and/or commands have no effect on the channel level.

The **Unpark** command has the opposite effect as the **Park** command.

Channels are parked in individual playback faders (not globally). Beware that if a channel is parked in one playback, it is still possible for another playback to cause the output of that channel to change.

Visual Representation

In CueServer Studio, parked channels appear in the stage view with their channel numbers in red. Note that you can only see parked channels if the Stage View is set to display a specific playback fader. You will not see parked channels in either the input or output stage views.



Examples

```
Park
```
Parks the currently selected channels in the currently active playback fader.

```
Channel 1>3+5>8 Park
```

Parks channels 1 through 3 and 5 through 8 in the currently active playback fader.

```
Playback 3 Channel * Park
```

Parks all channels in playback 3.

See Also

- **At**, **Channel**, **Playback**, **Unpark**

Press

TODO

Random

Syntax

Command	Description	Return Value
`Random <value>`	Generates a random number from 0 to *value*	A random number
`Random {<value1>, <value2>}`	Generates a random number from *value1* to *value2*	A random number

Abbreviation

`RAND`

The **Random** command to generate a random number. Use the **Random** command in CueScript expressions or commands to introduce randomness.

The **Random** command comes in two forms. If a single number is specified, a random number from 0 through that number (inclusive) will be returned. If an array of two numbers are specified, a random number from the first number through the second number (inclusive) will be returned.

When using the **Random** command as a substitution for a single parameter to another command, it must be enclosed in parenthesis. This is because the random command needs to be evaluated as if it is an expression, so the result of the expression is substituted into the outer command properly. See the examples below for clarification.

Examples

`Random 5`
Returns a random number from 0 through 5.

`Random {10,20}`
Returns a random number from 10 through 20.

`Macro (Random{5,8})`
Executes a random Macro from 5 through 8.

`Cue (Random{1,4}) Go`
Executes a random Cue from 1 through 4.

```
Channel 1 At (Random{50,100})
```

Sets Channel 1 to a random value from 50 through 100.

Reboot

Syntax

Command	Description	Return Value
`Reboot`	Reboot the CueServer	Always returns `1`

Abbreviation

None

Description

Causes the CueServer to reboot immediately.

Any show or playback occurring will be interrupted, and the hardware will gracefully shut down and then reboot.

Examples

```
Reboot
```
Causes the CueServer to reboot.

Record

Use the **Record** command to record/create/store Cues, Streams and Groups.

There are several variants of the **Record** command:

- **Record Cue**
- **Record Group**
- **Record Stream**
- **Record Stop**

Record Cue

Syntax

Command	Description	Return Value
`Record [option] Cue <cue number>`	Records cue *cue number*	The cue number recorded

- `<cue number>`
  - Any whole number from `0` to `99999`
  - May optionally contain decimal numbers from `.00` to `.99`
- `[option]`
  - `Empty` causes no DMX channels to be recorded into the cue
  - `Selected` causes only the current selected DMX channels to be recorded into the cue

Abbreviation

`R` [`EMPTY`, `SEL`] `Q`

Description

Recording Cues

The **Record Cue** command records (or re-records) a normal cue.

By default, all channels being output from the CueServer are captured into the new cue.

If no cue with the cue number exists, a new cue will be created. If a cue with the cue number already exists, it will be deleted first and replaced with an entirely new cue. To re-record just the DMX channels without affecting the other cue parameters, use the **Update Cue** command instead.

Record Cue Options

Several options are available to change how a cue is recorded.

Empty
If this option is used, then the new cue will be created containing no DMX channels. When this cue is

played back, it will have no affect on any DMX channels, but it will still behave like a normal cue with follow timing and automation rules.

Selected
If this option is used, then the new cue will be created containing only the currently selected DMX channels. Using this option, cues can be created that only affect certain channels when being played back.

Examples

```
Record Cue 1
```
Records the current output from the CueServer as Cue 1.

```
Record Empty Cue 2
```
Records Cue 2 with no DMX channels.

```
Record Selected Cue 3
```
Records the currently selected DMX channels as Cue 3.

```
Channel 1>10
Record Selected Cue 4
```
Records Channels 1 through 10 as the only channels in Cue 4.

See Also

- **Update Cue**

Record Group

Syntax

Command	Description	Return Value
`Record Group <group number>`	Records group *group number*	The group number recorded

- `<group number>`
    - Any whole number from `0` to `99999`

Abbreviation

`R U` or `R GR`

Description

The **Record Group** command creates a new group from the currently selected channels.

If no group with the group number exists, a new group will be created. If a group with the group number already exists, it will be deleted first and replaced with an entirely new group. To re-record just the selected channels without affecting the other group parameters, use the **Update Group** command instead.

Examples

```
Record Group 1
```

Records the currently selected channels as Group 1.

```
Channel 1>10
Record Group 2
```

Records Channels 1 through 10 as Group 2.

See Also

- **Update Group**

Record Stream

Syntax

Command	Description	Return Value
`Record Stream <cue number>`	Records streaming cue *cue number*	The cue number recorded

- `<cue number>`
    - Any whole number from `0` to `99999`
    - May optionally contain decimal numbers from `.00` to `.99`

Abbreviation

`R STR`

Description

Recording Streaming Cues

The **Record Stream** command begins recording (or re-recording) a streaming cue. As soon as this command is executed, a stream recording of the CueServer's DMX output will begin. Use the **Record Stop** command to stop recording the stream.

If no cue with the cue number exists, a new cue will be created. If a cue with the cue number already exists, it will be deleted first and replaced with an entirely new cue.

Examples

`Record Stream 1`
Begins recording the current output from the CueServer into Streaming Cue 1.

`Record Stop`
Stops recording the Streaming Cue.

See Also

- Record Stop

Record Stop

Syntax

Command	Description	Return Value
`Record Stop`	Stops recording a streaming cue	The cue number recorded

Abbreviation

`R STO`

Description

The **Record Stop** command stops recording any currently recording streaming cue. Use this command in conjunction with the **Record Stream** command.

Examples

`Record Stream 1`
Begins recording the current output from the CueServer into Streaming Cue 1.

`Record Stop`
Stops recording the Streaming Cue.

See Also

- Record Stream

Release

Syntax

Command	Description	Return Value
`Release`	Releases channels from the active playback fader	*None*

Abbreviation

`REL`

Description

Released channels have no effect on the DMX output. One can think of released channels as being “transparent”. Before any channels are set or cues executed, all of the channels of a playback fader are *released*.

Releasing Selected Channels

The **Release** command releases the currently selected channels in the active playback fader. If the **Release** command is executed when channels are selected, those channels are released (they become transparent) immediately. After the channels are released, the selection is cleared.

Releasing All Channels

If the **Release** command is executed when *no* channels are selected, then *all* channels in the active playback fader are released. It is common practice to execute the release command twice (`Release Release`) when one wants to be sure to release all channels in the active playback fader.

The **Release** command does not release parked channels. To release all channels, including parked channels, use the **Clear** command instead.

Examples

`Release`
Releases selected channels in the active playback fader.

`Channel 1>10 Release`
Releases channels 1>10 in the active playback fader.

`Playback 3 Release Release`
Releases all channels in playback 3.

See Also

- **Clear**, **Playback**

Reset

Syntax

Command	Description	Return Value
`Reset`	Resets the device to it's entirely cleared state	`0`

Abbreviation

`RESET`

Description

The **Reset** command clears all running show information and returns the device to it's cleared state.

Reset performs the following actions:

- Clears all playback faders (including parked channels)
- Kills any pending Wait commands
- Resets the command context

The **Reset** command does not honor parked channels. They will be cleared too.

Examples

`Reset`
Entirely resets the device to it's cleared state.

Set

Syntax

Command	Description	Return Value
`Set <variable> <value>`	Sets the value of the variable	The value the variable was set to

- `<variable>`
  - A user variable or system variable name.
- `<value>`
  - A string (a combination of characters enclosed in quotes, such as “Hello World”).
  - A number (a whole number, or a decimal number, such as *123* or *12.7*).

Alternate Syntax

See: **Assign** command.

Description

Setting Values

The **Set** command sets the value of a variable. The variable can be user defined (such as *xyz*, *LoopCount*, or *IsMyShowEnabled*), or it may be a system variable (such as *button.onColor* or *lcd.backlight*. System variables always contain a “dot” character (`.`). User variables must not contain a “dot” character, otherwise, they will be interpreted as a system variable, and they will not be stored properly.

User variables can be defined on the fly, simply by assigning a value to a variable. There is no need to pre-define variables.

See the **Assign** command for an alternate syntax for assigning variable values.

See the **Variables** section for how to use variables in the script language.

See the **System Variables** section for a complete list of available system variables.

Examples

`Set x 3`
Sets the variable *x* to the number `3`.

`Set text "Hello World"`
Sets the variable *text* to the string `Hello World`.

`Set lcd.backlight 25`
Sets the system variable *lcd.backlight* to 25%.

`Set y ('x' + 1)`
Sets the variable *y* to the result of the expression `'x' + 1`.

See Also

- **Assign**, **System Variables**

Stack

Syntax

Command	Description	Return Value
`Stack "<stack name>"`	Sets the active playback fader's cue stack	The number of the first cue in the stack
`Stack Clear`	Sets the active playback fader to use the main cue list	The number of the first cue in the main cue list
`Stack ?`	Queries the current stack name	The name of the current cue stack

- `<stack name>`
    - A name of the desired cue stack

Abbreviation

None

Description

Use the **Stack** command to change what cue stack the active playback fader is using.

By default, all cues are loaded from the main cue list. Additionally, the show file may contain one or more *cue stacks*. The **Stack** command is used to change which cue stack the playback fader is using. Once the stack has been changed on a playback fader, all cues on that playback will be loaded from that cue stack.

Use the **Stack Clear** command to return the active playback fader back to the main cue list. Optionally **Stack “”** can be used to accomplish the same thing.

When switching cue stacks, the first cue in the new stack will automatically become the playback fader’s *next cue*. Because the **Stack** command selects the first cue in a cue stack, the Go command can be used to run the first cue in the stack.

Examples

```
Stack "Surprise"
```

Sets the active playback fader's cue stack to the stack named "Surprise".

```
Stack "Intro" Go
```

Sets the active playback fader to use the "Intro" stack and executes the first cue in that stack.

```
Stack Clear
```

Sets the active playback fader to use the main cue list.

See Also

- **Cue**, **Go**

Start

Syntax

Command	Description	Return Value
`Start`	Resumes normal timing operation of the active playback fader	The playback number started

Abbreviation

`STA`

Description

The **Start** command resumes normal timing operation of the active playback fader. **Start** has the opposite effect as the **Stop** command.

When a playback fader is stopped, it’s timing operation is temporarily suspended in the following ways:

- Using the **Go** command does not crossfade, the channel levels of the cue appear immediately
- Using the **At** command does not crossfade, the levels appear immediately
- The auto-follow timer stops counting down
- Streaming cues are paused

Examples

`Start`
Resumes normal timing operation of the active playback fader.

`Playback 2 Start`
Resumes normal timing operation of playback 2.

See Also

- **Playback**, **Stop**

Stop

Syntax

Command	Description	Return Value
`Stop`	Suspends normal timing operation of the active playback fader	The playback number stopped

Abbreviation

`STO`

Description

The **Stop** command suspends normal timing operation of the active playback fader. **Stop** has the opposite effect as the **Start** command.

When a playback fader is stopped, it's timing operation is temporarily suspended in the following ways:

- Using the **Go** command does not crossfade, the channel levels of the cue appear immediately
- Using the **At** command does not crossfade, the levels appear immediately
- The auto-follow timer stops counting down
- Streaming cues are paused

Examples

`Stop`
Suspends normal timing operation of the active playback fader.

`Playback 2 Stop`
Suspends normal timing operation of playback 2.

See Also

- **Playback**, **Start**

Time

Syntax

Command	Description	Return Value
`Time <fade time>`	Sets the global fade time	The global fade time set
`Time ?`	Returns the current global fade time	The current global fade time

- `<fade time>`
  - A decimal number of seconds (optionally using decimal digits for fractions of seconds)
  - `0` means no fade time (or the channels/values are set immediately without fading)
  - Optionally may include a slash (`/`) which indicates a split (up/down) fade
  - Optionally may include a dash (`-`) which indicates a delayed fade

Abbreviation

`T`

Description

Setting The Global Fade Time

Use the **Time** command to set the *global fade time*. This time is used to crossfade channels or values whenever the **At** command is executed. The global fade time is used when setting channels, or a playback's submaster value.

Using Split Fade Times

A *split fade time* is used when it is desired to have channels that are fading up occur at a different rate than channels fading down. To specify a split fade time, use a slash character in between two fade times. For instance, the command `Time 3.5/7.5` will cause any channel that is fading up to occur in 3.5 seconds, and any channel that is fading down to occur in 7.5 seconds.

Using Fade Delays

Normally, whenever a channel level is set, the fade begins immediately. A delay can be inserted that would cause the fade to be delayed before starting to change value. To specify a fade delay, use a delay time and dash character before the fade time. For instance, the command `Time 5.5-10` will cause the fade to delay 5.5 seconds before beginning a 10 second fade.

Using Both Fade Delays And Split Fade Timing

Both fade delays and split fades can be combined. For instance, the command `Time 1-2/3-4` would cause any channels fading up to be delayed 1 second before fading over 2 seconds, while the downward fading channels would be delayed 3 seconds before fading over 4 seconds.

Determining The Current Global Fade Time

Use the **Time** command with the question mark (?) to return the current global fade time. A fade time such as `7.21` or `12/3` will be returned.

Note that the *Global Fade Time* is different from the *Cue Fade Time*. The global fade time effects the **At** command. The cue fade time effects the **Go** command. The cue fade time is set with the **Fade** command.

Examples

`Time 1`
Sets the global fade time to 1 second.

`Time 1.35/7.2`
Sets the global fade time to 1.35 seconds for upward fading channels and 7.2 seconds for downward fading channels.

`Channel 1>10 Time 5 At 50`
Selects channels 1 thru 10, then sets the fade time to 5 seconds, then sets the channels to 50%.

`Playback 1 Time 3.5 At 25`
Selects playback 1, then sets the fade time to 3.5 seconds, then sets the playback's submaster to 25%.

See Also

- **Cue**, **Fade**, **Go**

Toggle

Syntax

Command	Description	Return Value
`Toggle <value>`	Toggles the value of the selected object(s)	The value the object(s) were set to

- `<value>`
  - A percentage from `0` to `100`. When specifying percentages, the value can optionally be followed by the `%` sign.
  - A decimal number from `#0` to `#255`. When specifying decimal numbers, the value must be proceeded with a `#` sign.
  - A hexadecimal number from `$00` to `$FF`. When specifying hexadecimal numbers, the value must be proceeded with a `$` sign.
  - `FL` (Full) or `On` can be used as a shortcut that means `100%`
  - `Off` can be used as a shortcut that means `0%`

Abbreviation

`TOG`

Description

Toggling Values

The **Toggle** command flip-flops the currently selected object(s) values between a fixed value and zero. In other words, if the selected value is already set to the toggle value, the value is set to zero. But, if the selected value is not equal to the toggle value, then the value is set to the toggle value. This flip-flop behavior creates a situation where each time the **Toggle** command is executed, the selected value(s) alternate between zero and the toggle value.

The **Toggle** command can be used with many types of objects, including **Buttons**, **Channels**, **Groups**, **Outputs**, and **Playbacks**.

Other than the alternating behavior, the **Toggle** command otherwise behaves similarly to the **At** command.

Examples

```
Channel 1 Toggle 100
```

On each execution, toggles the value of channel 1 between 0% and 100%.

```
Group 3 Toggle 33
```

On each execution, toggles the value of the channels in group 3 between 0% and 33%.

See Also

- **Button**, **Channel**, **Contact**, **Group**, **Output**, **Playback**

Unpark

Syntax

Command	Description	Return Value
`Unpark`	Unparks the selected channel(s)	*None*

Abbreviation

None

Description

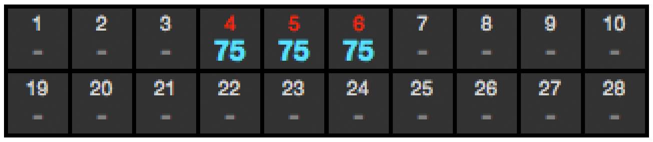
The **Unpark** command unlocks the current value of the selected channel(s) in the active playback fader. Parked channels cannot be changed by running cues or using the **At** command. Use the **Unpark** command to un-freeze one or more channels so future cues and/or commands will effect the channel level(s).

The **Park** command has the opposite effect as the **Unpark** command.

Channels are parked in individual playback faders (not globally). Beware that if a channel is parked in one playback, it is still possible for another playback to cause the output of that channel to change.

Visual Representation

In CueServer Studio, parked channels appear in the stage view with their channel numbers in red. Note that you can only see parked channels if the Stage View is set to display a specific playback fader. You will not see parked channels in either the input or output stage views.



Examples

```
Unpark
```
Unparks the currently selected channels in the currently active playback fader.

`Channel 1>3+5>8 Unpark`
Unparks channels 1 through 3 and 5 through 8 in the currently active playback fader.

`Playback 3 Channel * Unpark`
Unparks all channels in playback 3.

See Also

- **Channel**, **Playback**, **Park**

Update

Use the **Update** command to change what’s stored in Cues or Groups without affecting the other parameters of the object.

There are several variants of the **Update** command:

- **Update Cue**
- **Update Group**

Update Cue

Syntax

Command	Description	Return Value
`Update [option] Cue <cue number>`	Updates the channels in cue *cue number*	The cue number updated

- `<cue number>`
  - Any whole number from `0` to `99999`
  - May optionally contain decimal numbers from `.00` to `.99`
- `[option]`
  - `Empty` causes no DMX channels to be recorded into the cue
  - `Selected` causes only the current selected DMX channels to be recorded into the cue

Abbreviation

`UP` [`EMPTY`, `SEL`] `Q`

Description

Updating Cues

The **Update Cue** command updates the DMX channels of a normal cue. Use this command to change the DMX channels stored in a cue without loosing any of the other properties recorded in the cue (such as the fade and follow time, link, and automation rules).

By default, all channels being output from the CueServer are captured into the updated cue.

If no cue with the cue number exists, a new cue will be created. If a cue with the cue number already exists, as if it had been recorded with the **Record Cue** command.

Update Cue Options

Several options are available to change how a cue is updated.

Empty
If this option is used, then the updated cue will contain no DMX channels. When this cue is played back, it will have no affect on any DMX channels, but it will still behave like a normal cue with follow timing and automation rules.

Selected
If this option is used, then the updated cue will contain only the currently selected DMX channels. Using this option, cues can be created that only affect certain channels when being played back.

Examples

```
Update Cue 1
```
Stores (updates) the current output from the CueServer into Cue 1.

```
Update Empty Cue 2
```
Removes the DMX channels from Cue 2.

```
Record Selected Cue 3
```
Stores (updates) the currently selected DMX channels into Cue 3.

```
Channel 1>10
Update Selected Cue 4
```
Stores (updates) Channels 1 through 10 as the only channels into Cue 4.

See Also

- **Record Cue**

Update Group

Syntax

Command	Description	Return Value
`Update Group <group number>`	Updates group *group number*	The group number updated

- `<group number>`
  - Any whole number from `0` to `99999`

Abbreviation

`UP U` or `UP GR`

Description

The **Update Group** command updates the channels in a group to the currently selected channels.

If no group with the group number exists, a new group will be created. Use this command to change the DMX channels stored in a group without loosing any of the other properties recorded in the group (such as the name).

If no group with the group number exists, a new group will be created. If a group with the group number already exists, as if it had been recorded with the **Record Group** command.

Examples

```
Update Group 1
```
Stores (updates) the currently selected channels into Group 1.

```
Channel 1>10
Update Group 2
```
Stores (updates) Channels 1 through 10 into Group 2.

See Also

- **Record Group**

Wait

Syntax

Command	Description	Return Value
`Wait <time>`	Causes the execution of the current script to be suspended for a given number of seconds	An *id number* to identify the remaining commands
`Wait Clear`	Causes **any** commands that are currently waiting to be cancelled	The number of cleared commands
`Wait ?`	Returns the number of currently waiting commands	A number of processes

- `<time>`
  - A decimal number of seconds (optionally using decimal digits for fractions of seconds)

Abbreviation

`W`

Description

The **Wait** command causes the current command to be suspended for a given number of seconds. Use **Wait** to cause a delay between script steps.

The **Wait Clear** command cancels **any** currently waiting commands. If more than one command is currently in a waiting state, all of them will be cleared simultaneously.

Examples

`Channel 1 At FL; Wait 5; At 0`
Sets Channel 1 to FL, then waits 5 seconds, then sets Channel 1 to 0.

`Cue 1 Go; Wait 2.5; Clear`
Executes Cue 1, then waits 2.5 seconds, then clears the playback fader.

`Wait Clear`
Clears any currently waiting commands.

Write

Syntax

Command	Description	Return Value
`Write <port> <string>`	Writes the given string to the specified port	The number of characters written

- `<port>`
  - `COM1` refers to the built-in RS-232 port.
  - `COM2` refers to the built-in RS-485 port.

Abbreviation

`WR`

Writes (or sends) the given string to the specified serial port.

Use the **Write** command to send strings to other devices via one of the serial ports.

Examples

`Write COM1 "Hello World"`
Sends the string “Hello World” to the RS-232 port.

`Write COM2 "This is a test"`
Sends the string “This is a test” to the RS-485 port.

Logic Commands

CueScript contains several logic commands that are used to modify the normal execution of commands.

The **If..Then..Else** command is used to conditionally execute commands depending on the result of a conditional expression.

The **Break** command is used to force early termination of execution of a command string.

These commands are described in detail in the following sub-sections.

Break

Syntax

Command	Description	Return Value
`Break`	Stops executing the current command string	*None*

Abbreviation

`BR`

Description

The **Break** command stops executing the current command string. Use **Break** in situations where a condition requires that all of the subsequent commands should be ignored.

Examples

```
Cue 1 Go; Break; Button 1 On
```

Executes Cue 1, then stops execution of subsequent commands. The **Button 1 On** phrase will never be executed.

```
Cue 1 Go
If ('myVariable' > 5) Then Break
Button 1 On
Playback 3 At 50
```

Executes Cue 1, then checks to see if *myVariable* is greater than 5. If it is, then none of the remaining commands will execute. If *myVariable* is less than or equal to 5, then Button 1 will be turned on and Playback 3’s submitter will be set to 50%.

If..Then..Else

Syntax

Command	Description	Return Value
`If (<expression>) Then <action> [Endif]`	Tests *expression* and performs *action* if true	The result of *action*
`If (<expression>) Then <action1> Else <action2> [Endif]`	Tests *expression* and performs *action1* if true or *action2* if false	The result of *action1* or *action2*

Abbreviation

None

Description

The **If .. Then .. Else** statements are used to conditionally execute commands based on the value of an expression.

Consider this command:

```
If ('x'=1) Then Cue 1 Go
```

The above example first checks the value of the variable *x*, and if it is equal to 1, then Cue 1 is executed. On the other hand, if *x* is not equal to 1, then nothing will happen.

Using Else

The **Else** keyword can be used to execute commands if the expression is false. Consider this example:

```
If ('mode'>5) Then Cue 1 Go Else Cue 2 Go
```

In the above example, if the value of *mode* is greater than 5, then Cue 1 will execute, but if *mode* is 4 or less, then Cue 2 will execute.

Using Endif

In the basic form of the **If .. Then** statements, *all* of the commands after the **Then** will be executed if the expression is true. In the case where additional non-conditional commands are needed after the **If .. Then** statement, use the **Endif** keyword to end the conditional part of the **If .. Then** statement.

For example, in the following command, Cue 1 will execute *and* Playback 2 will be cleared if the *showEnabled* variable is 1:

```
If ('showEnabled'=1) Then Cue 1 Go; Playback 2 Clear
```

But, by inserting the **Endif** keyword, the script can be changed to have Cue 1 execute only if the *showEnabled* variable is 1, but Playback 2 is always cleared:

```
If ('showEnabled'=1) Then Cue 1 Go Endif Playback 2 Clear
```

Using Multiple Lines

The **If .. Then .. Else** statements can also be used across multiple lines of code, which is particularly useful when the script becomes more complex:

```
Playback 1

If ('testMode'=1) Then
    Cue 1 Go
Else
    Cue 2 Go
EndIf

Playback 2 Clear
```

Nesting Multiple If .. Then Statements

For more complex logic scenarios, you can put **If .. Then** statements *inside* of other **If .. Then** statements. For example:

```
Playback 1

If ('testMode'=1) Then
    Cue 1 Go
Else
    If ('eStop'=1) Then
        Cue 99 Go
    Else
        Cue 2 Go
    EndIf
EndIf

Playback 2 Clear
```

System Variables

CueServer uses System Variables to allow CueScript commands to change properties or behaviors of various system related objects. Setting a system variable has immediate effect, causing the referenced object to change appearance or behavior. For example, to immediately change the brightness of the LCD Backlight, the commands `Set lcd.backlight 25` or `"lcd.backlight"=25` can be used.

The following sections list the available system variables:

Buttons

Sets the color and flashing patterns for the built-in user defined function buttons.

Before setting or retrieving one of the button variables, make sure that one or more buttons are selected first. For example, use the **Button** command to specify which button(s) you want to change a property for.

`button.flash`	Sets the flash pattern for buttons. Available patterns range from `0` to `15`. A value of `0` means "no flash". The remaining 15 values produce various combinations of flashing when the button indicator is turned on.
`button.onColor` `button.offColor`	Sets the "on" and "off" colors for buttons. The value can be a single number from `0` to `100`, meaning off (black) to full-on (white), or it may be a 3-element array representing an RGB color. For example the array `{100,50,0}` would produce an Orange color.

Example:

```
Button 1
Set button.onColor {100,0,50}
Set button.flash 4
On
```

The example above first selects button 1, then sets it's color to a rose color, then sets it's flash pattern to a fast blink. Then, it turns the button's indicator "on".

LCD Display

Sets the backlight brightness and various string fields for the LCD display.

`lcd.backlight`	Sets the brightness of the LCD Backlight. Brightness values range from `0` to `100`.

`lcd.top`
`lcd.bottom`
`lcd.topLeft`
`lcd.topRight`
`lcd.bottomLeft`
`lcd.bottomRight`
	Sets a temporary overlay string that replaces the top or bottom lines, or quadrant of the display. Set this value to an empty string (`""`) to remove the temporary overlay.

Example:

```
Set lcd.backlight 25
Set lcd.top "Hello World"
Set lcd.bottom ""
```

The example above first sets the LCD Backlight brightness to 25%, then writes a temporary string to the top line that says `Hello World`, then removes any temporary string from the bottom line.

Playbacks

Changes properties of a Playback fader.

`playback.mode`	Sets the combine mode of a Playback fader. Available modes include `"Merge"`, `"Override"`, `"Scale"`, and `"Pin"`.

Example:

```
Playback 1
Set playback.mode "Override"
Playback 2
Set playback.mode "Scale"
```

The example above first sets the combine mode of Playback 1 to Override, then sets the combine mode of Playback 2 to Scale.

Random Numbers

Sets the seed for the random number generator.

`random.seed`	Sets the random number generator’s seed value. The random seed is an unsigned 32-bit value from `0` to `4294967295`.

Example:

```
Set random.seed 42
```

The example above sets the random seed to 42.

Universes

Sets properties of the DMX Universes.

`universe.priority`	Sets the priority level of the universe. Available values range from `0` to `200`.

Example:

```
Universe 7
Set universe.priority 150
```

The example above sets the priority of Universe 7 to 150.

Release Notes

The following list shows a revision history of the software releases for CueServer 2:

Version 1.0.8 (4/27/2015)

- **CueServer Studio 2**
    - Added colored icons to the Stage View's "view" menu to make it easier to identify which playback is being selected.
    - Added a new "View" menu to the Stage View that allows all universes, or only a specific universe to appear in the display.
    - Added user-assigned names to the playback faders in the Playbacks view.
    - The command line and live views are now only available from the active show editor window.
    - The Editor Window now shows "[ACTIVE]" in its title bar when viewing an active show.
    - Resource and Trigger editor panels now remember what state they were in when switching between panels.
    - Resource editor panels now refresh automatically when an object is recorded or updated by CueScript commands.
    - Fixed a bug introduced in 1.0.7 that crashed when opening offline shows.
    - Fixed a problem with the Stage View that would only allow the first eight playback faders to be selected in the View menu.
    - Fixed an issue that would cause the editor for Stations or Buttons to disappear when changes were applied.
    - Fixed a problem where the entire device list in the Navigator window could get a green background when dragging a project into the list.
    - Renamed the previous View menu to Layer in the Stage View for consistency.
    - Fixed a problem with the Editor Window that would reload the active editor if the currently selected editor was clicked on.
    - Updated the default index.shtml file in the new show template.
    - Addressed a problem that could cause the reported uptime to be blank.
    - Added missing category icons for several editor panels.
    - Added "Refresh" menu item to pop-up gear menu for Cues, Groups, Macros, Sounds and Web Pages.
    - Updated compilers resulting in more compact Windows builds.
    - Fixed a Drag & Drop highlighting problem with the Web and Sound file browsers.
    - Fixed a problem that caused folders to not be able to be dragged into the Web and Sound browsers.

  - Addressed problems with creating/deleting stations when editing offline show files.
  - Addressed a problem that could cause the station panel to crash when changing the selected station.
  - Addressed a problem with text fields that would not properly select the entire field on mouse click entry.
  - Improved the text entry interaction with hours/minutes/seconds entered into timers.
  - Fixed an issue with improperly formatted data being stored for the “only specific days” type of timer scheduling.
  - Windows: Fixed a problem with offline show paths appearing with slashes instead of backslashes.
  - Windows: Fixed a problem that prevented Drag & Drop to the Web and Sound file browsers from the Desktop.
  - Windows: Fixed a problem with field validation that sometimes caused the insertion point to move unexpectedly.
  - Windows: Fixed a problem in the Specific Month and Year dialogs that would cause the checkboxes to not draw properly in certain circumstances.
  - Windows: Fixed a problem where the System Log may not display the entire log file.
- **Firmware**
  - Added the INPUT ENABLE/DISABLE syntax to enable or disable the DMX Input layer of the playback stack.
  - Added automatic updating of playback fader user preferences for combine modes when loading or switching shows.
  - Added ability to specify a wider range of weeks of the month when picking date ranges for timers (i.e.: 5th Friday, or 2nd from Last Wednesday, etc.).
  - Added the ability to query variable values to the get.cgi API.
  - Addressed a problem with CueScript parsing timeouts being raised too quickly.
  - Fixed an issue that would cause the CueScript parser to erroneously report itself as shut down after executing a WAIT command.
  - Fixed a problem introduced in 1.0.7 that caused the WAIT CLEAR command to raise an exception.
  - Fixed a problem where incomplete CueScript strings would silently fail without reporting an error.
  - Fixed a problem that could cause CueStations to not respond after switching active shows.
  - Addressed a problem that could cause timers to fail to trigger that are set for “nth weekday of the month” in certain circumstances.
  - Improved error descriptions for unrecognized commands.

Version 1.0.7 (4/7/2015)

- **CueServer Studio 2**
    - Added a display of the current Stack Name to the Playback view.
    - Added an indicator to the Universe Settings panel to show if a universe is receiving input data.
    - Added Variables sub-view to the Status panel that shows any currently defined user variables in the system.
    - Added CPU Info sub-view to the Status panel that shows the running status of the various CueServer processes, average CPU load and memory usage.
    - Added System Log sub-view to the Status panel that shows the system log file and allows the system message indicator to be cleared.
    - Added a warning indicator to the navigator panel in the Editor Window that shows when an important message is available in one of the sub-panels.
    - Addressed a problem with the format of the query string when CueServer Studio attempts to fetch the current version of software.
    - Addressed a problem with the progress indicators in the Stations, Timers and Rules panels not moving properly when the window is resized.
    - Changed the global fade time label in the command field to “Time”.
    - Fixed a spelling mistake in the Clock Settings window.
    - Adjusted the minimum allowable size for the Navigator Window.
- **Firmware**
    - Added the AT CUE syntax for selectively recalling specific channels from a cue.
    - Added the AT PLAYBACK syntax for selectively recalling specific channels from a playback fader.
    - Added syntax for adding or subtracting groups to the current group selection using GROUP x + y – z.
    - Changed the behavior of testing rule condition variables to interpret a null-string (“”) as being equal to zero (0).
    - Addressed a problem that caused CueServer to not communicate properly with sACN or CueStation nodes if there was no router on the network.
    - Addressed a problem that could cause only one universe of sACN data to be received as input into the system.
    - Addressed a problem that would leak socket resources when sending CueStation Hub indicator changes.
    - Addressed a problem with the Stack command that caused empty stacks to produce an error.
    - Addressed a problem that prevented a Playback Fader to have it’s stack name cleared by setting the stack name to the empty string.
    - Addressed a problem that caused the Clear command to not clear a Playback Fader’s stack property.
    - Addressed a problem that could prevent setting of static IP address parameters via the LCD Menu.

    - Addressed a problem that would cause the device to not be discoverable when booted on a network without a router.
    - Addressed several issues with the get.cgi API for compatibility with the CuePad iOS app. CueServer 2 requires CuePad v2.2 or greater.
    - Addressed a problem that prevented CueScript commands to be able to be unicast to the CueServer's IP Address.
    - Added additional error checking and reporting to the various daemon processes.

Version 1.0.6 (3/13/2015)

- **CueServer Studio 2**
    - Addressed a problem introduced in 1.0.5 that caused the Stations editor panel to not appear properly if an external station was edited immediately after editing the built-in station.
- **Firmware**
    - Addressed a problem introduced in 1.0.5 that caused buttons and contacts on external stations to not trigger properly.

Version 1.0.5 (3/11/2015)

- **CueServer Studio 2**
    - Show project files can now be downloaded/uploaded to/from your computer.
    - Offline project files can now be edited without the CueServer hardware.
    - Added a second list view to the main Navigator window to make is easier to work with offline project files.
    - Added the ability to create new projects from within CueServer Studio.
    - Drag and Drop has been added to move show project files between a CueServer and your computer.
    - Added Network Settings window to remotely change a CueServer's network settings.
    - Added Time Settings window to set manual or automatic time settings, including definitions for over 400 time zones.
    - A warning dialog now appears when you try to edit a CueServer that has outdated firmware.
    - Changed CueScript buttons to show newlines as semicolons (;) to be more consistent with syntax rules.
    - Addressed a problem that caused device discovery to only work on the host's default Ethernet interface.
    - Windows: Enabled the main window's close box.
    - Windows: Reduced the flickering of the Playback and Status panels.
    - Windows: Addressed a problem that would cause CueScript buttons to not display multiline text properly.

  - Windows: Addressed a problem that caused the Control-C shortcut for “Copy” to not work properly in the CueScript popup editor window.
  - Windows: Addressed a problem that caused the popup menu controls in the Stage view to display incorrect labels.
  - Windows: Addressed a problem that would cause the main window to not open properly if reopened after the app was closed while minimized.
- **Firmware**
  - Added the ability for Buttons and Contacts to be enabled/disabled.
  - Implemented the Update Cue and Update Group commands.
  - Pressing and holding the Up/Down navigation buttons while editing values on the LCD Display now continuously adjusts the value.
  - Changed the behavior of the “=” command to dynamically act as either Assign or Equals, depending on the context of the parameters.
  - Addressed a problem that caused sACN to not receive data properly from certain consoles.
  - Improved sACN receive logic to deal with transmitters that do not properly terminate
  - Addressed problems with setting the Network Settings using the LCD Display that would cause unexpected results.
  - Addressed a problem with the Fade and Time commands that caused them to not be able to receive their values from variables.
  - Addressed a problem with the LCD Display that could cause it to freeze if the system time was adjusted in certain circumstances.
  - Addressed a problem with the sACN protocol not properly supplying a valid CID field for transmit packets.
  - Added additional network diagnostics support to csctl.

Version 1.0.4 (2/9/2015)

- **CueServer Studio 2**
  - Added the ability to open local offline show files.
  - Added an Cue Fade Times popup window that provides direct access to extended fade time attributes.
  - Added cursor and history control to the command line using the up/down/right/left arrow keys.
  - Added contextual menu to Web Pages that includes option to open files in the user’s web browser.
  - Added new rule conditions for testing indicators and outputs.
  - Added a watermark that appears when no editor panel is selected.
  - Added support for Rev. B hardware.
  - The navigator window now remembers it’s preferred size and column widths.
  - Improved the description and input validation of the Add Remote CueServer window.

- Addressed a problem that caused the Fade/Follow/Link fields in the Cue panel that could make it difficult to remove unwanted values or enter decimal numbers.
    - Addressed a cosmetic problem with the Month/Day/Year popup menus in the Active Days section of the Timers panel.
    - Addressed a problem with Timers set to trigger between two dates that would cause the Weekdays field to have an invalid default value.
    - Addressed a problem that caused the default value of the Sun Brightness Rule Condition to be undefined.
    - Addressed a problem that could cause a crash if a CueScript popup editor button was double-clicked.
    - Windows: Fixed the titles of several dialog box windows.
    - Windows: Select entire text field when entering the Rename Cue Stack window.
    - Windows: Change the Sounds and Web Pages panels to display file paths with the correct path delimiters.
- **Firmware**
    - Changed the rule execution behavior to execute rules in two passes (check eligibility first, then perform actions), which solves a multi-rule race condition problem.
    - Added a new default index.shtml file to the Web Resources of new shows.
    - Exported several show and device related environment variables in the Apache virtualhost for use with web pages’ CGI and SSI scripting.
    - Addressed a problem with switching shows not causing Apache to properly switch the served web pages.
    - Addressed a problem that caused the Fade command to fail to modify the next cue’s execution time in certain circumstances.
    - Addressed a problem that caused the Toggle command to not work with button indicators or digital outputs.
    - Addressed a problem that caused shows with spaces or other special characters in their name to not be able to be deleted.
    - Addressed a problem that caused Telnet sessions to hang in certain situations.

Version 1.0.3 (1/22/2015)

- **CueServer Studio 2**
    - First version available as both OS X and Windows builds.
    - Added the ability to add and remove remote devices to the CueServer Navigator window.
    - Added firmware version column to the Navigator window.
    - Added a built-in firmware image that matches the release version of Studio.
    - Addressed a problem that caused the command line text to appear very small on Retina displays.

    - Addressed a problem that could allow automatic text substitutions to occur in the script editor popup window.
    - Addressed a problem that would cause the app to not launch properly if the splash screen was clicked.
- **Firmware**
    - Addressed a problem that caused remote devices to not return the correct ping data via HTTP.

Version 1.0.2 (1/9/2015)

- **CueServer Studio 2**
    - Added rules to cues. Previously, cues only had a single action field. Now each cue can have an arbitrary number of rules associated with them.
    - Added a cue “contents preview” to the Cue editor panel. This panel shows the first channels and/or information about the stream.
    - Added resizable divider between panels in the Cues, Groups, Macros, Timers and Rules panels.
    - Added an automatic software version check when the application is launched.
    - The Editor Window now remembers it’s preferred size.
    - The License Code Details window no longer displays window resize controls.
    - Added drop-down menu arrows to buttons in Stage View.
    - Enabled the File>Close menu item for separate Stage and Playback view windows.
    - The New Cue window now remembers the last used capture mode.
    - Addressed a problem that caused the Capture Selected Channels cue recording mode to fail.
    - Addressed a problem that could cause a crash when exiting from full-screen mode on OS X.
- **Firmware**
    - Added the FOLLOW CLEAR command variant.
    - Addressed a problem that could cause cues with more than about 4000 channels to not play back correctly.
    - Addressed a problem that caused the Universe Loss event to be sent more often than expected to the global rules.
    - Addressed a problem that could cause cue execution to improperly return an error if the cue contained an action.
    - Addressed a problem with the CLEAR command not clearing parked channels.
    - Addressed a problem with the CLEAR command not restoring a playback’s submaster to 100%.
    - Addressed a problem that could cause the RESET command to crash the CueScript parser.
    - Addressed a problem with the RESET command not clearing commands that were in the wait queue.

Version 1.0.1 (12/23/2014)

- **CueServer Studio 2**
  - Added “User’s Manual…”, “Support Website…”, and “Release Notes…” to the Help Menu.
  - Changed the font used for displaying CueScript commands.
  - Addressed a problem on OS X that made it possible to insert “smart quotes” into CueScript fields, which would cause the execution of the script to fail.
  - Addressed a problem with the *Open Web* command that could cause the web page to not open properly.
- **Firmware**
  - Changed the LCD Menu display for System Information.
  - Addressed a problem that could cause show data to not be synchronized with the memory card.
  - Improved error reporting for I2C Bus Daemon.

Version 1.0 (12/18/2014)

- First public release version.
- All versions prior to v1.0 were private.