CueServer 2 User's Manual

23.7.26 — Last update: Jul 27, 2023

Interactive Technologies, Inc.

Distributor:

Techni-Lux

Phone: 407-857-8770
Fax: 407-857-8771
Email: sales@techni-lux.com
www.techni-lux.com

Table of Contents

Getting Started

Welcome to CueServer.

Sections are being added to this User’s Manual on a regular basis.

Current Version

On July 25, 2023, version **5.0.7** of CueServer Studio was released. Please download the latest version here:

- CueServer Studio Downloads

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

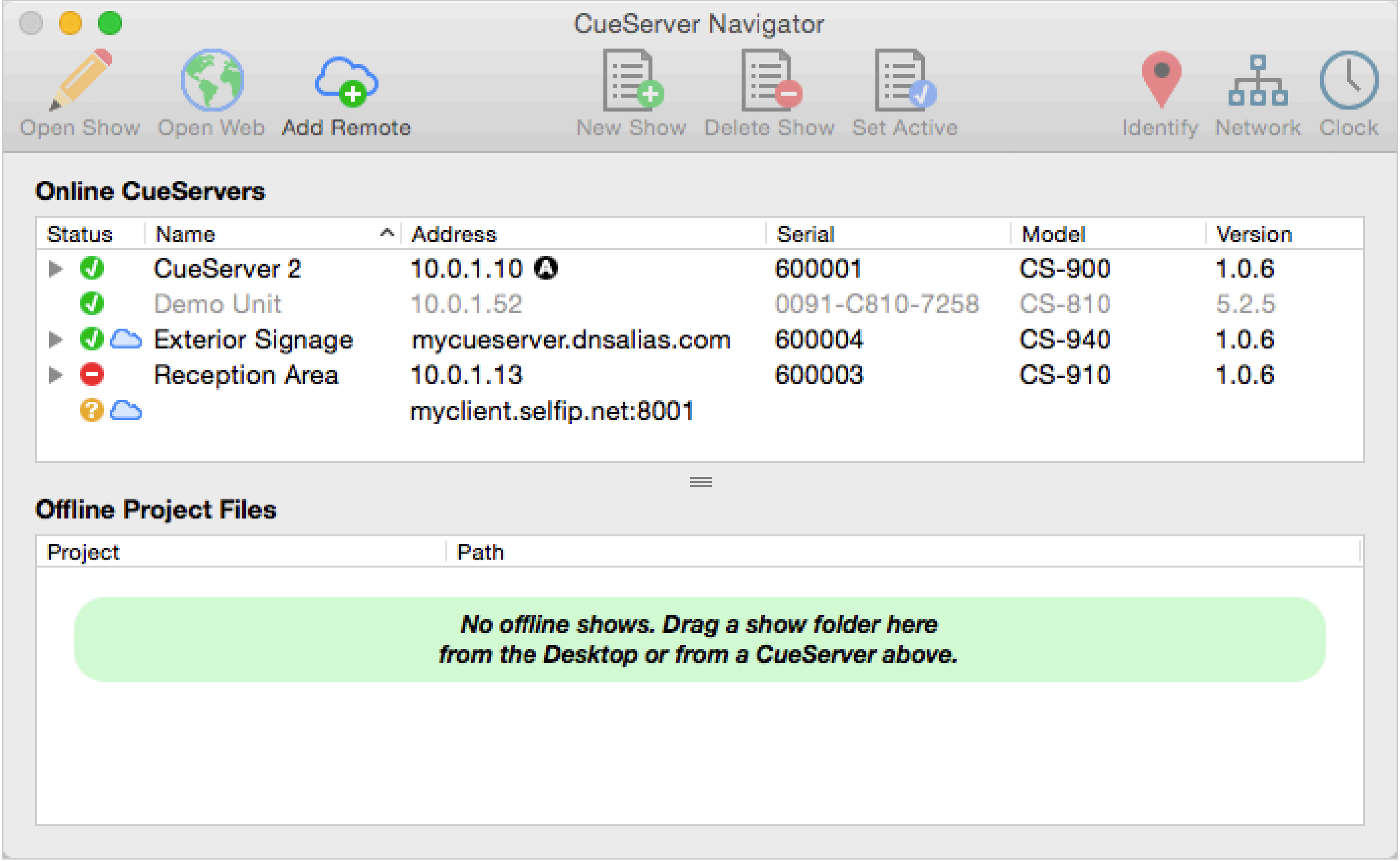
CueServer Studio is the desktop application used to program, configure, locate and operate CueServer 2 and CueServer 3 devices. It is available for both Mac OS X and Windows. You can download the current version of CueServer Studio here:

- CueServer Studio Downloads

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.



Online CueServers

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			

Offline Project Files

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.

For CueServer devices that are configured to use separate LANs for management and lighting data, the Address column will show which LAN port on the CueServer was used to connect to the device. These icons only appear when the CueServer is configured to use Dual-LANs:

ⒶLAN Interface A
ⒷLAN Interface B

If a CueServer is discovered, but it is not reachable on the local network because of a mismatch between the computer’s local subnet and the CueServer’s IP address, then a warning icon (⚠) will appear next to its address. Try using Network Settings to change the CueServer’s IP address to one that is reachable on the local network.

Older CS-800 series CueServers will also appear in the list of online devices. CueServer Studio 2 can not directly edit these devices, and they appear in the list in gray text. If they are opened, CueServer Studio will simply open the device’s web interface.

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 (🖧) 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.

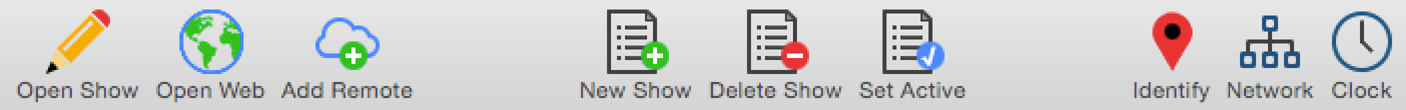
Maintenance

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.

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.

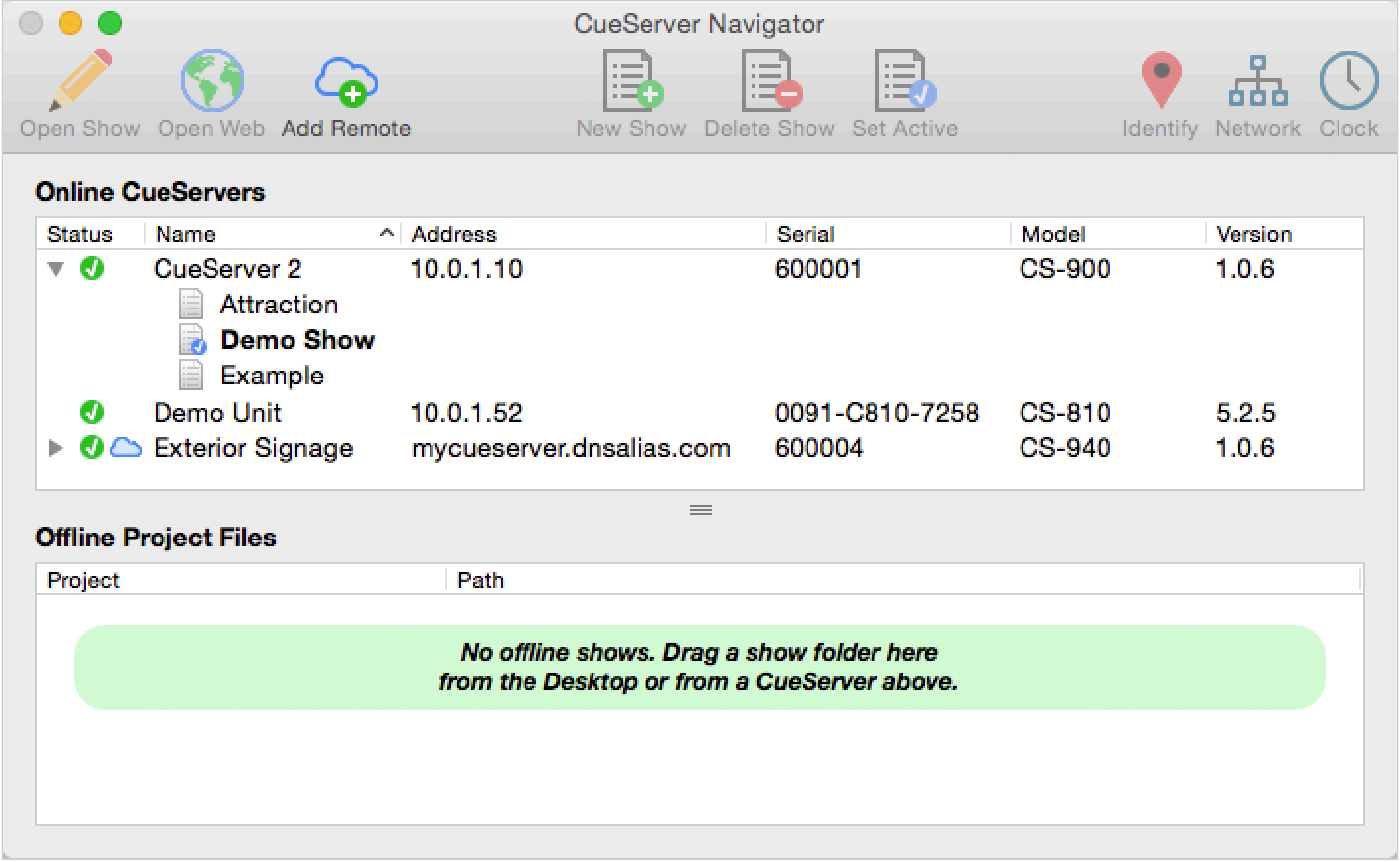
Clock
Displays a dialog window that allows the clock settings of the selected CueServer to be changed. Use this option to change the time zone, automatic and/or manual time settings of a CueServer.

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.



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

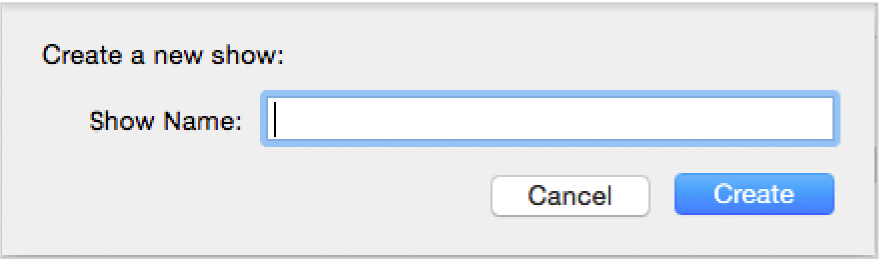
No offline shows. Drag a show folder here from the Desktop or from a CueServer above.

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 ( ). You can also find the **New Show** command in the contextual menu or the CueServer menu.

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.

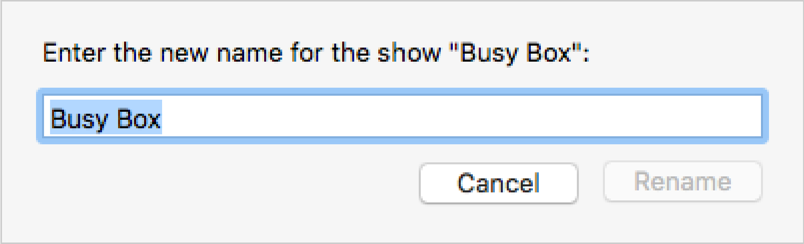
Changing the Active Show

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 ( ). You can also find the **Set Active Show** command in the contextual menu or the CueServer menu.

Renaming a Show

To rename a show, right-click on the show and choose **Rename Show** from the contextual menu. You can also find the **Rename Show** command in the CueServer menu.

A dialog window will appear that allows you to rename the show:



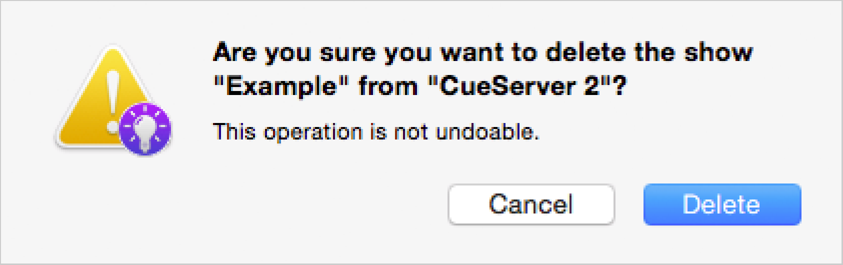
Enter the new name for the show "Busy Box":

Busy Box

Deleting a Show

To delete a show, click on the show file and then click on the Delete Show toolbar item ( ). You can also find the **Delete Show** command in the contextual menu or the CueServer menu.

A confirmation dialog will appear:



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

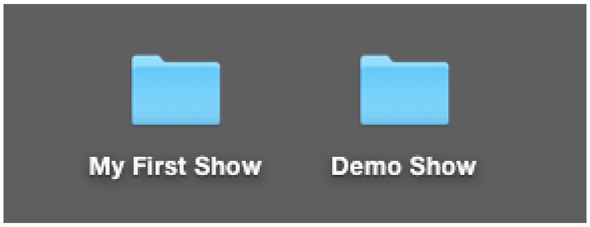
This operation is not undoable.

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

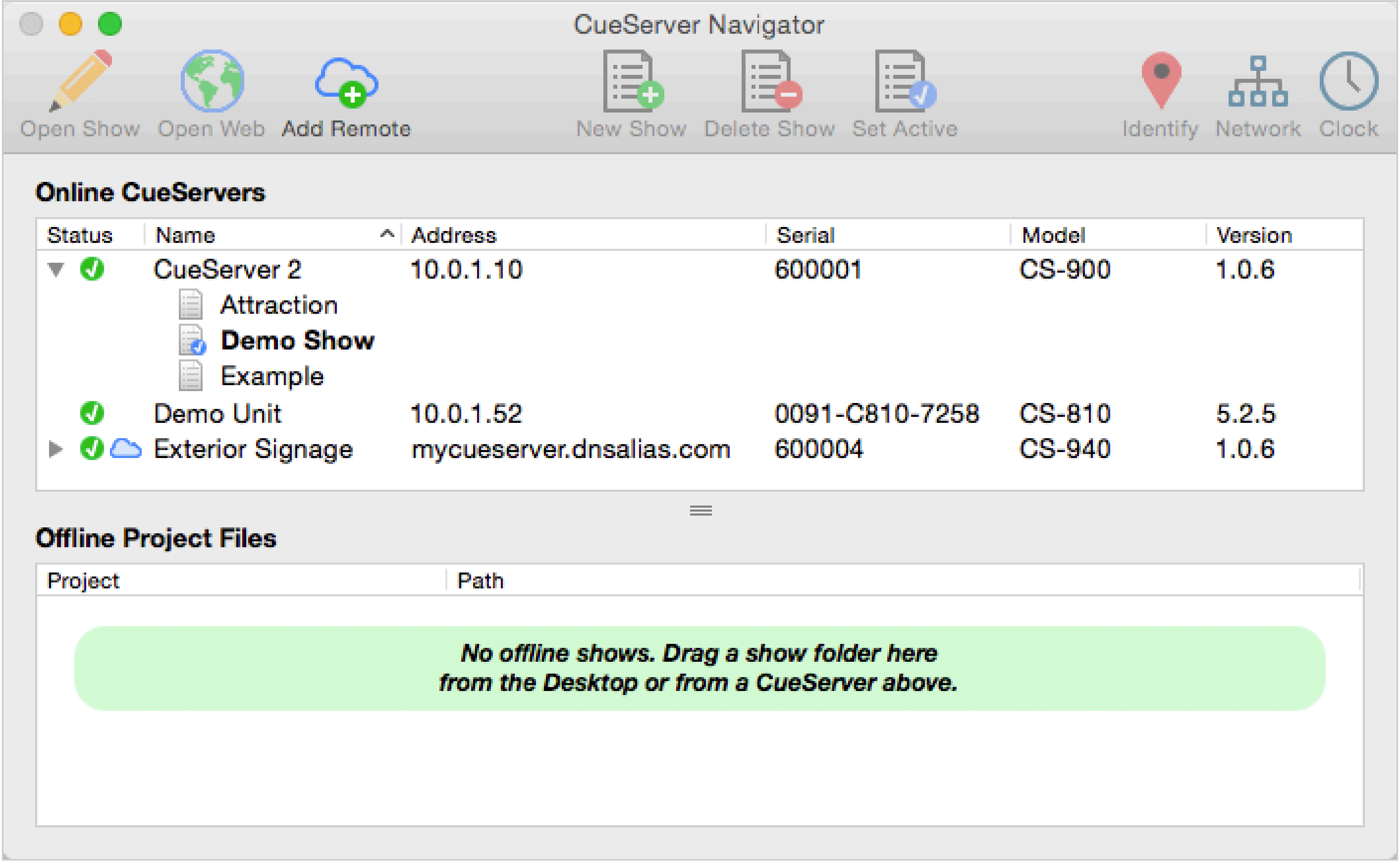
✻ You cannot delete the currently active show. If you want to delete the active show, first switch to another show (or create a new one).

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.

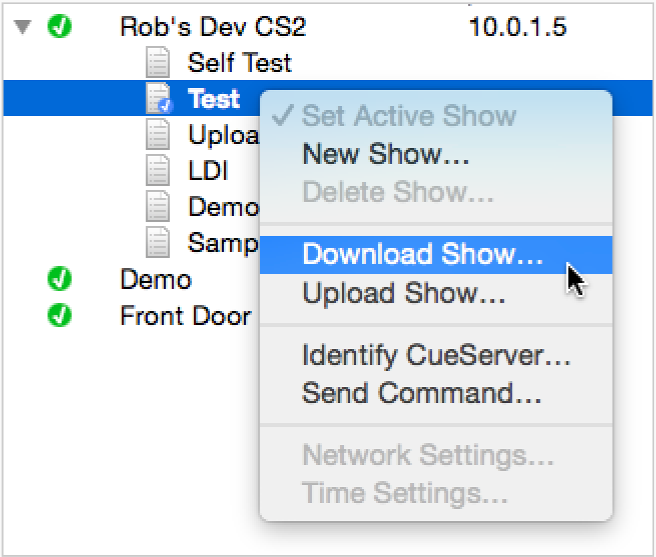
Downloading a Show File from CueServer to Computer

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

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.



Using the Download Show contextual menu item.

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.

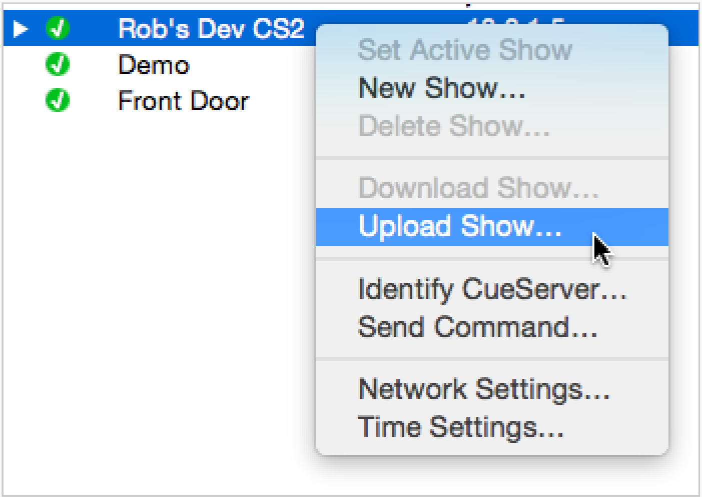
Uploading a Show File from Computer to CueServer

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

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.



Using the Upload Show contextual menu item.

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.

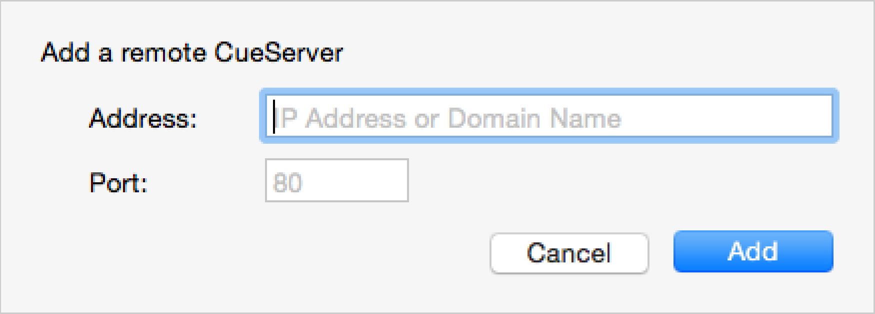
Once the name and location are given, CueServer Studio will create the new show file and add it to the *Offline Project Files* list so the offline show file can be opened and edited.

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:

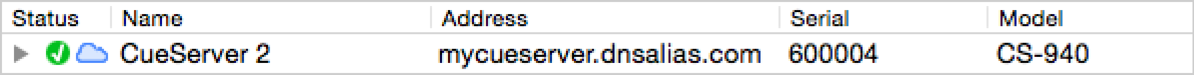
Address
This field can accept either an IP Address (for example: 50.167.102.1), or a domain name (such as: mycueserver.dnsalias.com).

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

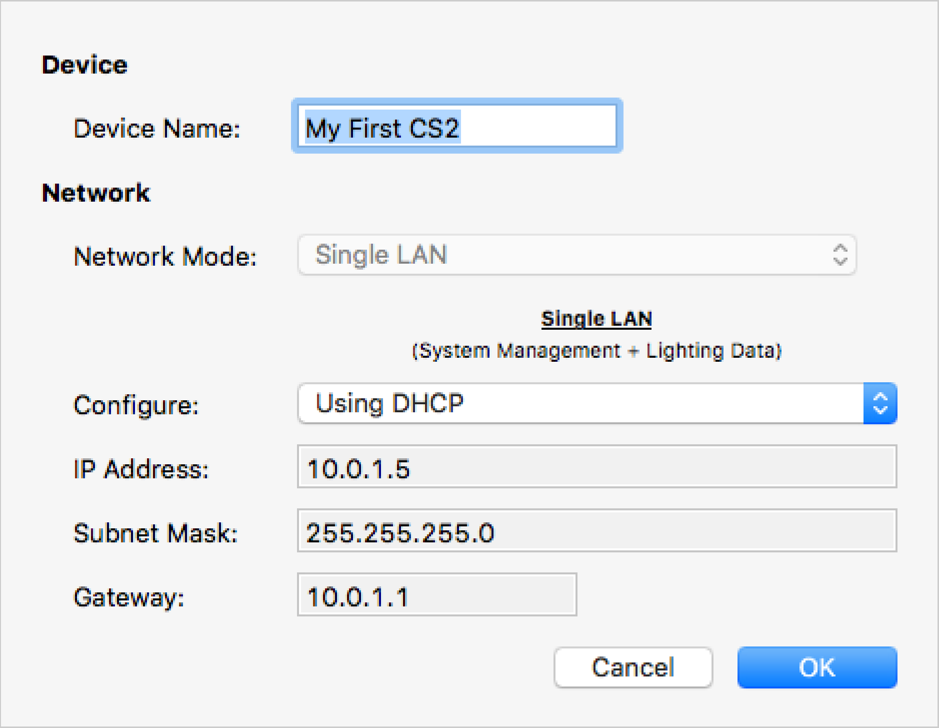
Simply select the Remote CueServer, and then press the **Delete** key on your keyboard.

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

Network Mode: Single LAN

Single LAN
(System Management + Lighting Data)

Configure: Using DHCP

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.

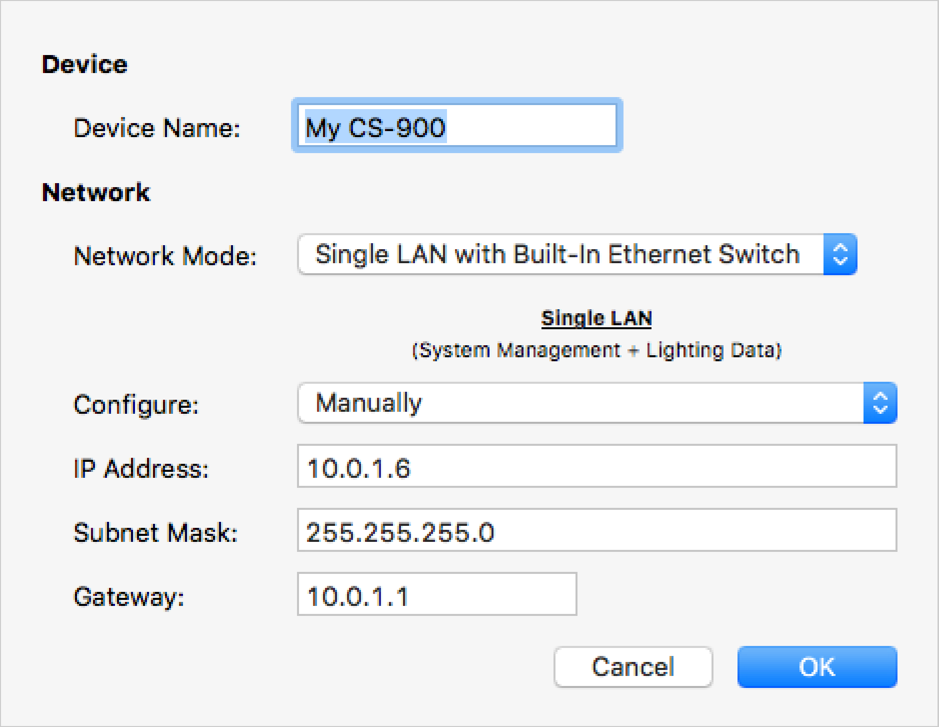
Network Mode

On CueServers with only a single Ethernet jack, this option is fixed to "Single LAN".

On CueServers with two Ethernet jacks, two different options are available in this menu:

Option 1: Single LAN with Built-In Ethernet Switch

When this mode is selected (which is the factory default), the two Ethernet jacks are simply two ports of a built-in Ethernet switch, both of which are connected to the CueServer. In this configuration, either one of the two jacks can be connected to the local network, and the other jack can be used as an extra port for connecting a laptop, DMX node, CueStation Hub or any other network device. All of the CueServer management and lighting control data flows over this single LAN connection.



Device

Device Name: My CS-900

Network

Network Mode: Single LAN with Built-In Ethernet Switch

Single LAN

(System Management + Lighting Data)

Configure: Manually

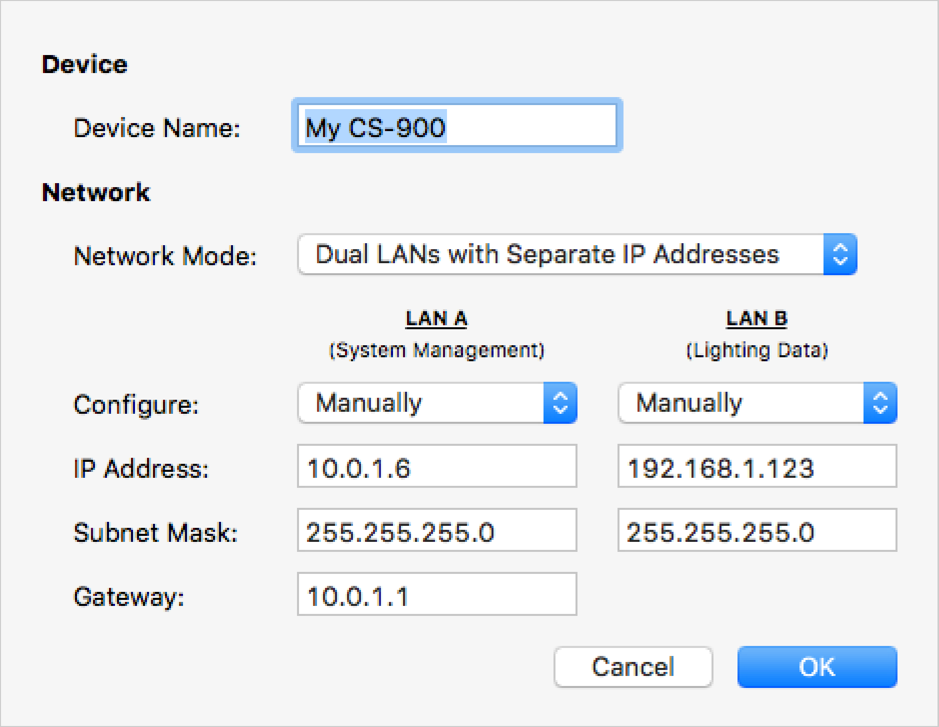
IP Address: 10.0.1.6

Subnet Mask: 255.255.255.0

Gateway: 10.0.1.1

Option 2: Dual LANs with Separate IP Addresses

When this mode is selected, the two Ethernet jacks become two separate LAN ports, each of which have their own network settings. The jack marked “A” is used to connect to a device management network. LAN “A” can be used for device discovery, configuration editing, and has access to the web interface. The jack marked “B” will have a different IP address and is used for lighting control data. LAN “B” is where DMX-over-Ethernet protocols such as sACN, Art-Net, and KiNET are flowing. LAN “B” can also be used for device discovery, configuration editing, and has access to the web interface.



Device

Device Name: My CS-900

Network

Network Mode: Dual LANs with Separate IP Addresses

	LAN A (System Management)	LAN B (Lighting Data)
Configure:	Manually	Manually
IP Address:	10.0.1.6	192.168.1.123
Subnet Mask:	255.255.255.0	255.255.255.0
Gateway:	10.0.1.1	

! Please note that when changing the network mode, the device will need to reboot for the changes to take effect. Please remember to reconfigure the physical network connections when changing modes, especially if the mode is being changed from a Dual LAN to a Single LAN configuration. In this case, it’s likely that there were two separate networks attached to the two ports on the CueServer and after the mode is changed the built-in Ethernet switch would attempt to bridge these separate networks into one, which will certainly cause unintended network problems.

Network Address

CueServer allows the Network Address to be set manually or automatically. If the CueServer is configured for Dual LANs, then each network (A and B) can have it’s own network settings.

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 when using DHCP.

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, or if a specific network configuration is desired that uses a static address for the CueServer.

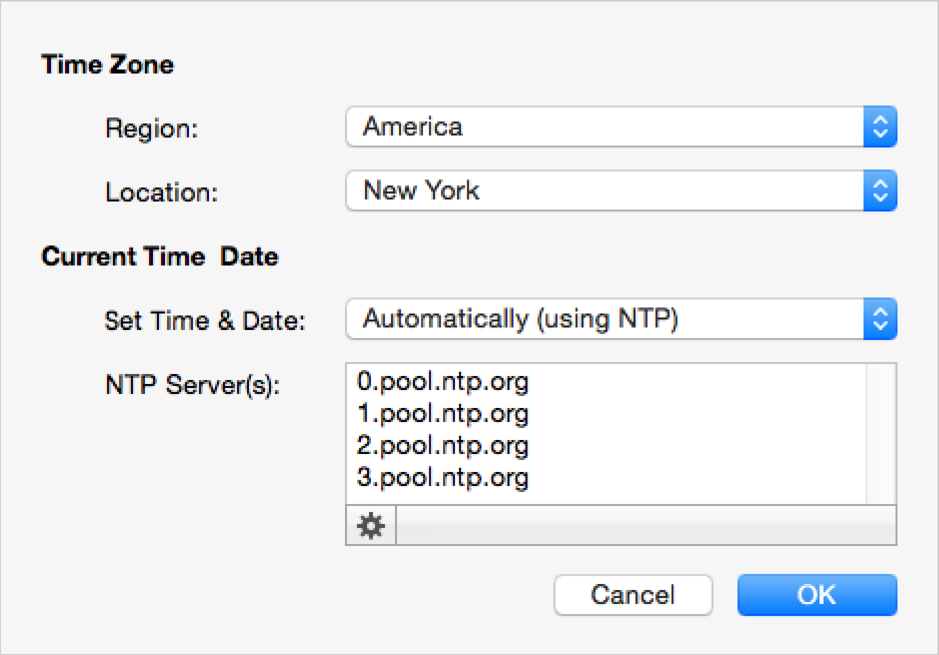
> ✻ Please note that if the CueServer is configured to use Dual LANs, the second network (LAN “B”) does not have a field for setting a default gateway. This is because all network traffic that would need to use a gateway to reach the external Internet will flow through LAN “A”.

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.

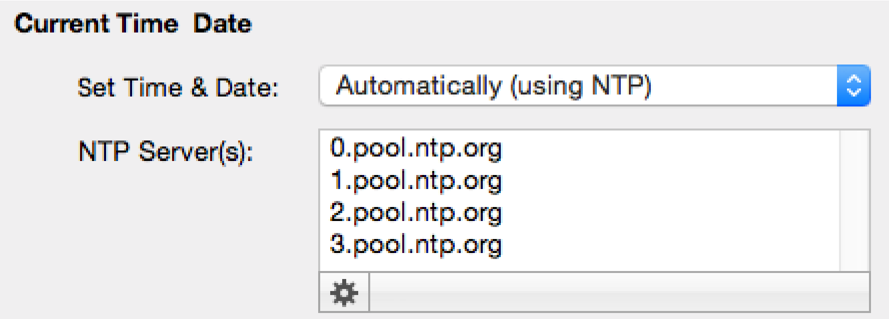
CueServer’s timezone database is derived from the standard Linux distribution and includes over 400 distinct regional locations. See the timezone listing for a complete list of available timezones.

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

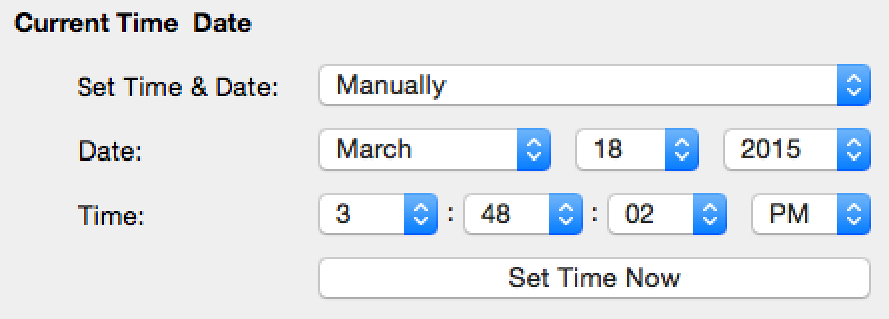
Set Time & Date: Automatically (using NTP)

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.

Available Timezones

The following list shows the time zones available for CueServer.

Africa/Abidjan

Africa/Accra

Africa/Addis Ababa

Africa/Algiers

Africa/Asmara

Africa/Asmera

Africa/Bamako

Africa/Bangui

Africa/Banjul

Africa/Bissau

Africa/Blantyre

Africa/Brazzaville

Africa/Bujumbura

Africa/Cairo

Africa/Casablanca

Africa/Ceuta

Africa/Conakry

Africa/Dakar

Africa/Dares Salaam

Africa/Djibouti

Africa/Douala

Africa/El Aaiun

Africa/Freetown

Africa/Gaborone

Africa/Harare

Africa/Johannesburg

Africa/Juba

Africa/Kampala

Africa/Khartoum

Africa/Kigali

Africa/Kinshasa

Africa/Lagos

Africa/Libreville

Africa/Lome

Africa/Luanda

Africa/Lubumbashi

Africa/Lusaka

Africa/Malabo

Africa/Maputo

Africa/Maseru

Africa/Mbabane
Africa/Mogadishu
Africa/Monrovia
Africa/Nairobi
Africa/Ndjamena
Africa/Niamey
Africa/Nouakchott
Africa/Ouagadougou
Africa/Porto-Novo
Africa/Sao Tome
Africa/Timbuktu
Africa/Tripoli
Africa/Tunis
Africa/Windhoek
America/Anchorage
America/Anguilla
America/Antigua
America/Aruba
America/Bahia
America/Barbados
America/Belize
America/Bogota
America/Boise
America/Buenos Aires
America/Cambridge Bay
America/Campo Grande
America/Cancun
America/Caracas
America/Catamarca
America/Cayenne
America/Cayman
America/Chicago
America/Chihuahua
America/Coral Harbour
America/Cordoba
America/Costa Rica
America/Cuiaba
America/Curacao
America/Danmarkshavn
America/Denver
America/Detroit
America/Dominica
America/Edmonton
America/Eirunepe

America/El Salvador

America/Fort Wayne

America/Grenada

America/Guadeloupe

America/Guatemala

America/Guyana

America/Halifax

America/Havana

America/Hermosillo

America/Indianapolis

America/Inuvik

America/Jamaica

America/Juneau

America/La Paz

America/Lima

America/Los Angeles

America/Louisville

America/Martinique

America/Mendoza

America/Mexico City

America/Monterrey

America/Montreal

America/Nassau

America/New York

America/Nome

America/Panama

America/Phoenix

America/Port-au-Prince

America/Puerto Rico

America/Regina

America/Santiago

America/Santo Domingo

America/Sao Paulo

America/St Johns

America/St Kitts

America/St Lucia

America/St Thomas

America/St Vincent

America/Tijuana

America/Toronto

America/Vancouver

America/Winnipeg

Antarctica/Casey

Antarctica/Davis

Antarctica/DumontDUrville
Antarctica/Macquarie
Antarctica/Mawson
Antarctica/McMurdo
Antarctica/Palmer
Antarctica/Rothera
Antarctica/South Pole
Antarctica/Syowa
Antarctica/Troll
Antarctica/Vostok
Asia/Aden
Asia/Almaty
Asia/Amman
Asia/Anadyr
Asia/Aqtau
Asia/Aqtobe
Asia/Ashgabat
Asia/Ashkhabad
Asia/Baghdad
Asia/Bahrain
Asia/Baku
Asia/Bangkok
Asia/Beirut
Asia/Bishkek
Asia/Brunei
Asia/Calcutta
Asia/Choibalsan
Asia/Chongqing
Asia/Chungking
Asia/Colombo
Asia/Dacca
Asia/Damascus
Asia/Dhaka
Asia/Dili
Asia/Dubai
Asia/Dushanbe
Asia/Gaza
Asia/Harbin
Asia/Hebron
Asia/Ho Chi Minh
Asia/Hong Kong
Asia/Hovd
Asia/Irkutsk
Asia/Istanbul

Asia/Jakarta
Asia/Jayapura
Asia/Jerusalem
Asia/Kabul
Asia/Kamchatka
Asia/Karachi
Asia/Kashgar
Asia/Kathmandu
Asia/Katmandu
Asia/Khandyga
Asia/Kolkata
Asia/Krasnoyarsk
Asia/Kuala Lumpur
Asia/Kuching
Asia/Kuwait
Asia/Macao
Asia/Macau
Asia/Magadan
Asia/Makassar
Asia/Manila
Asia/Muscat
Asia/Nicosia
Asia/Novokuznetsk
Asia/Novosibirsk
Asia/Omsk
Asia/Oral
Asia/Phnom Penh
Asia/Pontianak
Asia/Pyongyang
Asia/Qatar
Asia/Qyzylorda
Asia/Rangoon
Asia/Riyadh
Asia/Saigon
Asia/Sakhalin
Asia/Samarkand
Asia/Seoul
Asia/Shanghai
Asia/Singapore
Asia/Taipei
Asia/Tashkent
Asia/Tbilisi
Asia/Tehran
Asia/Tel Aviv

Asia/Thimbu
Asia/Thimphu
Asia/Tokyo
Asia/Ujung Pandang
Asia/Ulaanbaatar
Asia/Ulan Bator
Asia/Urumqi
Asia/Ust-Nera
Asia/Vientiane
Asia/Vladivostok
Asia/Yakutsk
Asia/Yekaterinburg
Asia/Yerevan
Atlantic/Azores
Atlantic/Bermuda
Atlantic/Canary
Atlantic/Cape Verde
Atlantic/Faeroe
Atlantic/Faroe
Atlantic/Jan Mayen
Atlantic/Madeira
Atlantic/Reykjavik
Atlantic/South Georgia
Atlantic/St Helena
Atlantic/Stanley
Australia/ACT
Australia/Adelaide
Australia/Brisbane
Australia/Broken Hill
Australia/Canberra
Australia/Currie
Australia/Darwin
Australia/Eucla
Australia/Hobart
Australia/LHI
Australia/Lindeman
Australia/Lord Howe
Australia/Melbourne
Australia/NSW
Australia/North
Australia/Perth
Australia/Queensland
Australia/South
Australia/Sydney

Australia/Tasmania

Australia/Victoria

Australia/West

Australia/Yancowinna

Brazil/Acre

Brazil/DeNoronha

Brazil/East

Brazil/West

Canada/Atlantic

Canada/Central

Canada/East-Saskatchewan

Canada/Eastern

Canada/Mountain

Canada/Newfoundland

Canada/Pacific

Canada/Saskatchewan

Canada/Yukon

Chile/Continental

Chile/EasterIsland

Etc/GMT

Etc/GMT+1

Etc/GMT+2

Etc/GMT+3

Etc/GMT+4

Etc/GMT+5

Etc/GMT+6

Etc/GMT+7

Etc/GMT+8

Etc/GMT+9

Etc/GMT+10

Etc/GMT+11

Etc/GMT+12

Etc/GMT-1

Etc/GMT-2

Etc/GMT-3

Etc/GMT-4

Etc/GMT-5

Etc/GMT-6

Etc/GMT-7

Etc/GMT-8

Etc/GMT-9

Etc/GMT-10

Etc/GMT-11

Etc/GMT-12

Etc/Greenwich

Etc/UCT

Etc/UTC

Etc/Universal

Etc/Zulu

Europe/Amsterdam

Europe/Andorra

Europe/Athens

Europe/Belfast

Europe/Belgrade

Europe/Berlin

Europe/Bratislava

Europe/Brussels

Europe/Bucharest

Europe/Budapest

Europe/Busingen

Europe/Chisinau

Europe/Copenhagen

Europe/Dublin

Europe/Gibraltar

Europe/Guernsey

Europe/Helsinki

Europe/Isle of Man

Europe/Istanbul

Europe/Jersey

Europe/Kaliningrad

Europe/Kiev

Europe/Lisbon

Europe/Ljubljana

Europe/London

Europe/Luxembourg

Europe/Madrid

Europe/Malta

Europe/Mariehamn

Europe/Minsk

Europe/Monaco

Europe/Moscow

Europe/Nicosia

Europe/Oslo

Europe/Paris

Europe/Podgorica

Europe/Prague

Europe/Riga

Europe/Rome

Europe/Samara

Europe/San Marino

Europe/Sarajevo

Europe/Simferopol

Europe/Skopje

Europe/Sofia

Europe/Stockholm

Europe/Tallinn

Europe/Tirane

Europe/Tiraspol

Europe/Uzhgorod

Europe/Vaduz

Europe/Vatican

Europe/Vienna

Europe/Vilnius

Europe/Volgograd

Europe/Warsaw

Europe/Zagreb

Europe/Zaporozhye

Europe/Zurich

Indian/Antananarivo

Indian/Chagos

Indian/Christmas

Indian/Cocos

Indian/Comoro

Indian/Kerguelen

Indian/Mahe

Indian/Maldives

Indian/Mauritius

Indian/Mayotte

Indian/Reunion

Mexico/BajaNorte

Mexico/BajaSur

Mexico/General

Pacific/Apia

Pacific/Auckland

Pacific/Chatham

Pacific/Chuuk

Pacific/Easter

Pacific/Efate

Pacific/Enderbury

Pacific/Fakaofo

Pacific/Fiji

Pacific/Funafuti

Pacific/Galapagos
Pacific/Gambier
Pacific/Guadalcanal
Pacific/Guam
Pacific/Honolulu
Pacific/Johnston
Pacific/Kiritimati
Pacific/Kosrae
Pacific/Kwajalein
Pacific/Majuro
Pacific/Marquesas
Pacific/Midway
Pacific/Nauru
Pacific/Niue
Pacific/Norfolk
Pacific/Noumea
Pacific/Pago Pago
Pacific/Palau
Pacific/Pitcairn
Pacific/Pohnpei
Pacific/Ponape
Pacific/Port Moresby
Pacific/Rarotonga
Pacific/Saipan
Pacific/Samoa
Pacific/Tahiti
Pacific/Tarawa
Pacific/Tongatapu
Pacific/Truk
Pacific/Wake
Pacific/Wallis
Pacific/Yap
US/Alaska
US/Aleutian
US/Arizona
US/Central
US/East-Indiana
US/Eastern
US/Hawaii
US/Indiana-Starke
US/Michigan
US/Mountain
US/Pacific
US/Pacific-New

US/Samoa

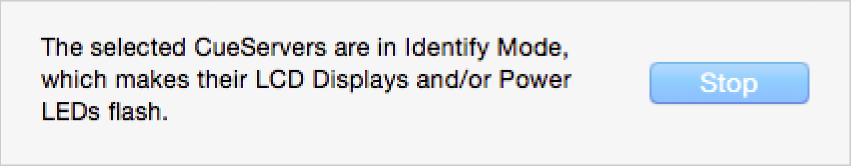
Identifying CueServers

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.

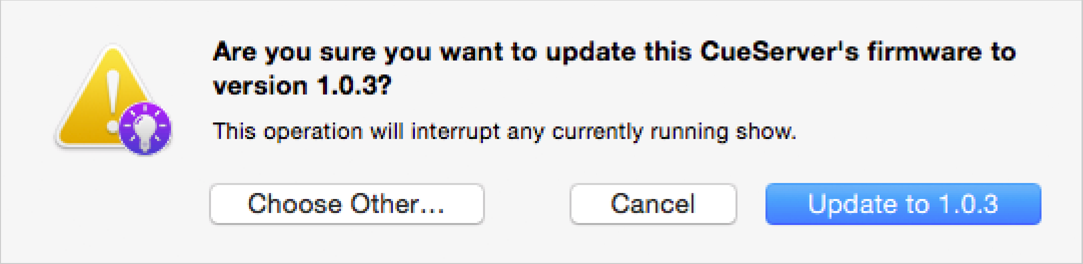
Updating Firmware

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:



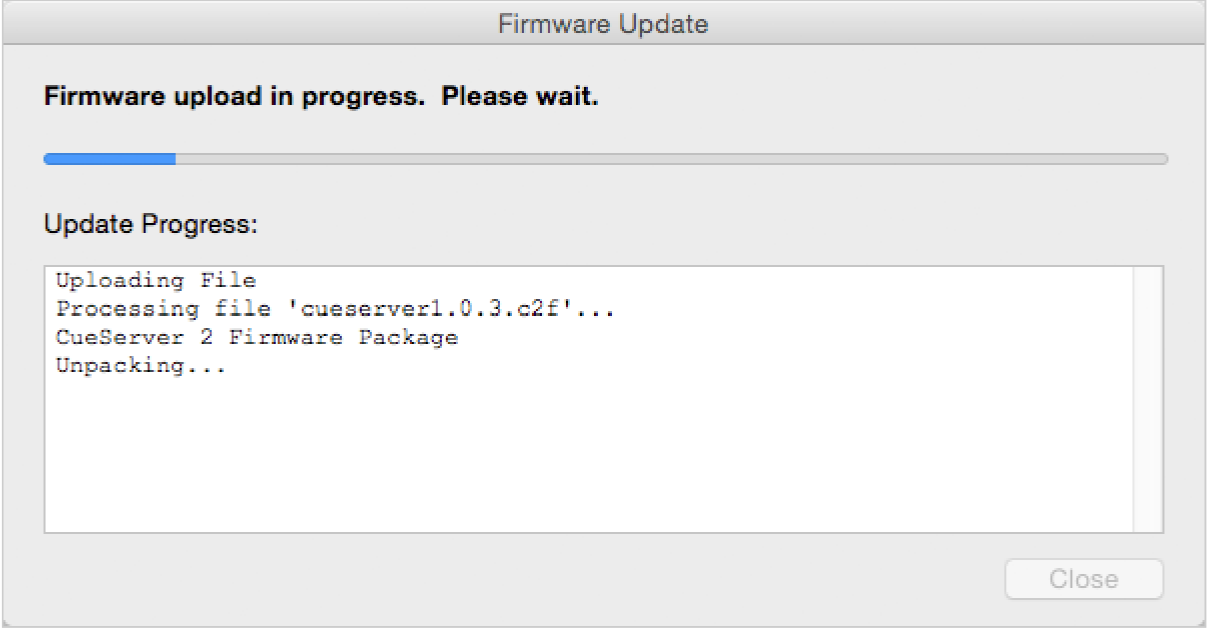
Are you sure you want to update this CueServer's firmware to version 1.0.3?

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:

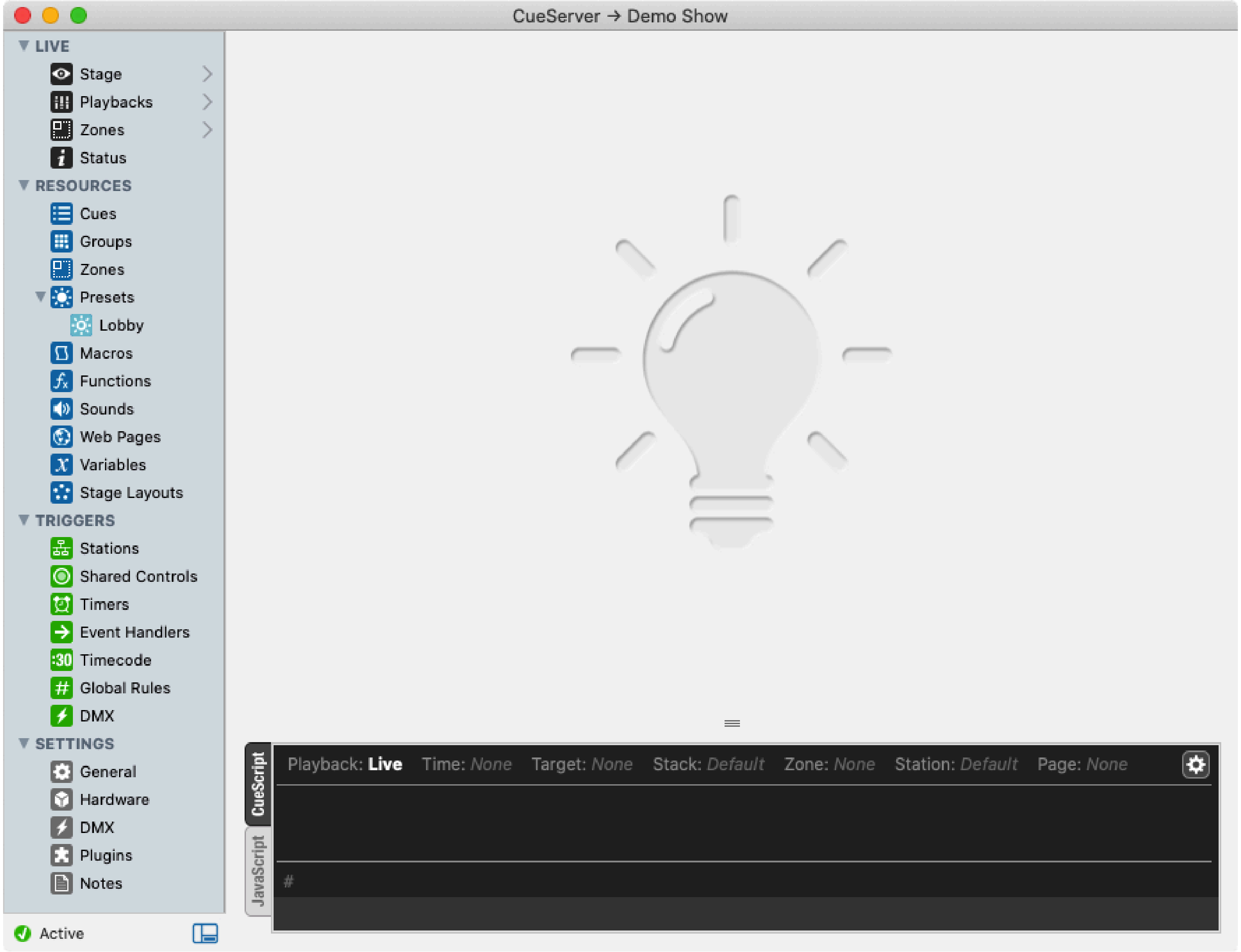
```
Uploading File
Processing file 'cueserver1.0.3.c2f'...
CueServer 2 Firmware Package
Unpacking...
```

The progress of the update is shown in the window and on the LCD screen of the CueServer. 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.



Playback: **Live** Time: *None* Target: *None* Stack: *Default* Zone: *None* Station: *Default* Page: *None*

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
  - Zones – for viewing Zones and Zone states
  - Status – for viewing the live status of various CueServer subsystems.

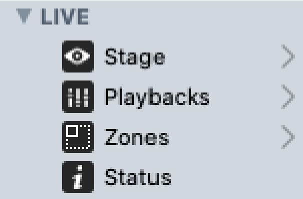
- Resources – various content types for CueServer projects
    - Cues – scenes and timeline based streams
    - Groups – definitions of groups of channels and fixtures
    - Zones – listing of zones
    - Presets – listing of presets
    - Macros – user-defined scripts
    - Functions – user-defined CueScript or JavaScript functions
    - Sounds – audio clips
    - Web Pages – custom web pages for the project
    - Variables – view and update variables
    - Stage Layouts – edit layouts for the live stage
- Triggers – definitions for incoming system events
    - Stations – setup for stations, buttons, contact-closures and more
    - Shared Controls – a global set of buttons
    - Timers – setup for timers
    - Event Handlers – handlers for plugin events
    - Timecode – SMPTE timecode events
    - Global Rules – a global list of rules
    - DMX – trigger actions or events based on incoming DMX values
- Settings – system preferences
    - General – general purpose settings
    - Hardware – model, audio, LCD display and DMX port config
    - DMX – DMX and fixture related settings
    - Plugins – javascript plugins
    - Notes – optional details about the show file

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.

> The command line’s visibility can be toggled by clicking on the console toggle button (⊟) in the lower-left corner of the editor window.

Live

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



The following sections describe these views in more detail:

- Stage – for viewing DMX channels
- Playbacks – for viewing playback faders
- Zones – for viewing Zones and Zone states
- Status – for viewing the live status of various CueServer subsystems.

Stage

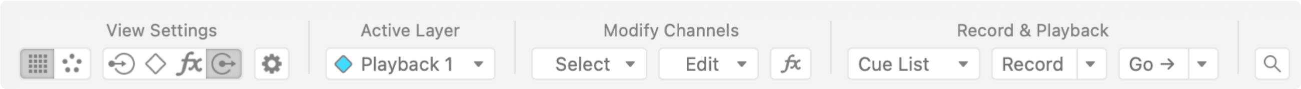
Overview

The stage is a real-time view into CueServer's input, output, and layers between. The interface provides tools and routines to quickly record, play-back and create new content.

At the heart of the stage is a grid of channels. Each channel displays identifying information, the current value, and information about its current state. Channel values are color-coded to indicate the playback from which their value is sourced, and can be displayed in various formats.

Depending on the selected View Settings, the stage will display information from the input (⟲), active playback (◇), or composite output (⊖).

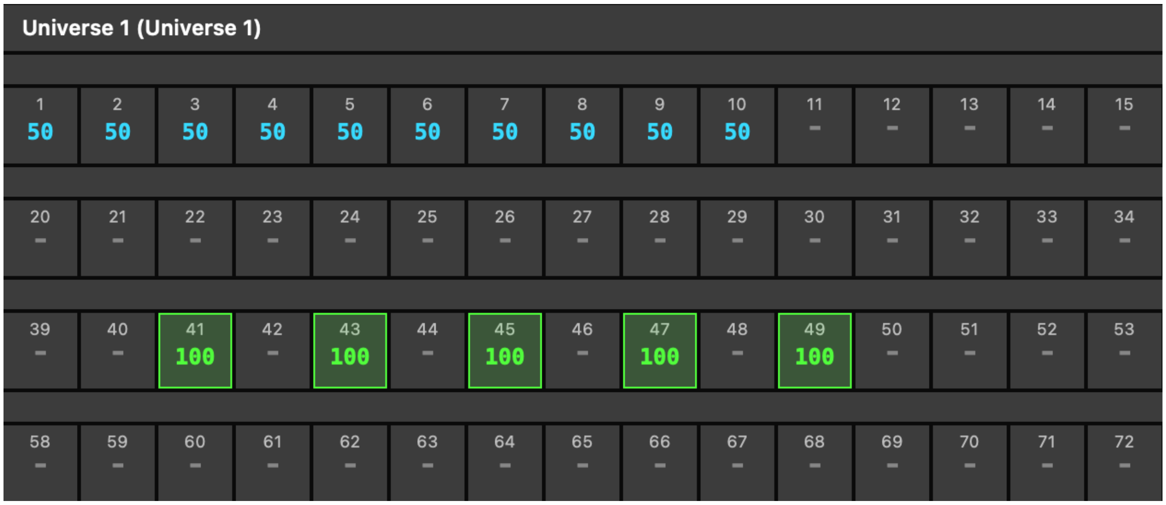
The control ribbon along the top of the stage view provides easy access to view controls, available playbacks, selection tools, and other quick actions *(covered below)*.



Channels

Each channel in the grid displays its current value and information about its current state. The color used to display a channel's value indicates the playback from which that channel's value is sourced.

In the example below, channels 1>10 appear with blue text indicating their values originate in **Playback 1**, while channels 41, 43, 45, 47, & 49 have green text indicating they are sourced from **Playback 2**.



Universe 1 (Universe 1)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
50	50	50	50	50	50	50	50	50	50	—	—	—	—	—
20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
39	40	41	42	43	44	45	46	47	48	49	50	51	52	53
—	—	100	—	100	—	100	—	100	—	100	—	—	—	—
58	59	60	61	62	63	64	65	66	67	68	69	70	71	72
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—

The current selection is also depicted in the stage and can be modified via CueScript, or by selecting channels and fixtures directly. The selection is color-coded to the current playback to indicate where modifications will take effect *(seen as a green highlight around the selected channels in the image above)*. Actions, such as adjusting channel values, can be performed directly from within the stage interface.

Channels may have a visually distinct background or text color indicating that they are in a specific state:

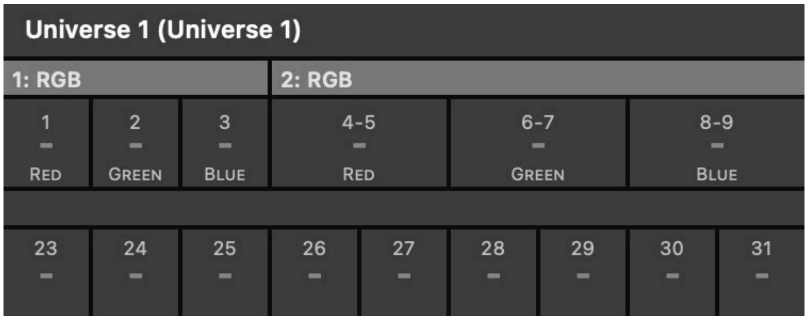
3 50	Parked channels appear with a red background
2 50	Locked channels appear with all-yellow text
1 50	Disabled channels appear with a darkened appearance
101 –	Masked channels appear with a crosshatch

✻ Locked and Disabled states only appear in the active layer of the playback they are locked or disabled in. Parked and Masked channels are visible from all layers.

Fixtures

When fixtures are patched, they will appear above the channels to which they are patched in the stage. Information about patched channel’s function will be visible within the channel, if provided by the fixture patch.

In cases where channels are combined in 16, 24 or 32 bit configurations, those channels will be stitched together into a single contiguous block to indicate that they are operated as a single unit.

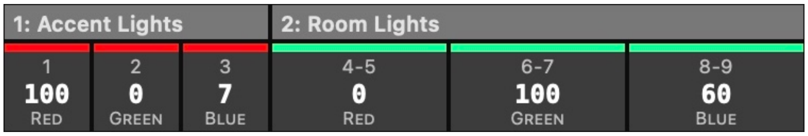

Universe 1 (Universe 1)

1: RGB			2: RGB		
1 – RED	2 – GREEN	3 – BLUE	4-5 – RED	6-7 – GREEN	8-9 – BLUE

23 –	24 –	25 –	26 –	27 –	28 –	29 –	30 –	31 –

Fixtures with color attributes will display the current approximate fixture output color based upon the current channel values in the visible layer (*this setting can be disabled via the gear menu*).

Fixture titles used in the stage view come from the fixture profile, however, a custom name can be specified for each fixture in the fixture patch, which if specified, will be displayed instead.

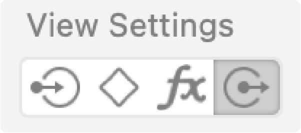


1: Accent Lights			2: Room Lights		
1	2	3	4-5	6-7	8-9
100	0	7	0	100	60
RED	GREEN	BLUE	RED	GREEN	BLUE

When one or more fixtures are selected, and the fixture panel is enabled (*via the gear menu*), a panel of controls for the selected fixtures will appear in a panel to the right. If channels are selected and are patched to a fixtures, the appropriate controls for the selected channels will appear.

Choosing the Visible Layer

Use the **View Settings** selector to choose which layer of the DMX composition is being shown:

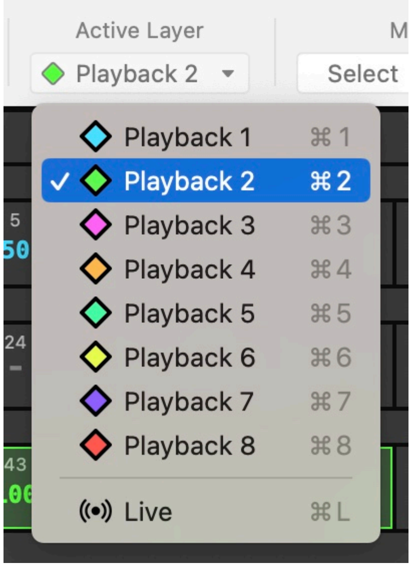


The view options are:

- **Input** (⇥) – This view shows any DMX values that are being input into the device.
- **Active** (◇) – This view shows DMX values that are present in the active Playback Fader.
- **Active + Effects** (*fx*) – This view shows DMX values that are present in the active Playback Fader + the output of the Playback's combined effects.
- **Output** (↦) – This view shows the final composite DMX values that are being output from the device.

Choosing the Active Layer

Use the **Active Layer** menu to choose which Playback is currently Active:



The layer options are:

- **Playback *n*** – Sets the selected playback as active.
- **Live** – Sets the Live playback as active *(default)*.

Any actions taken, via the UI or CueScript, will be executed within the **Active Layer**. When changed from this menu, any current selection will be retained, but will target the the newly active Playback.

Choose Stage Settings

Use the ⚙ menu to choose how the stage displays information.

The display options are:

- **View as Percent** – This mode shows channel levels as a percentage. Values range from 0 to 99, and then FL (meaning Full, or 100%).
- **View as Decimal** – This mode shows channel levels in decimal format. Values range from 0 to 255.
- **View as Hexadecimal** – This mode shows channel levels in hexadecimal format. Values range from 00 to FF.
- **View as Bar Graph** – This mode shows channel levels in a bar-graph format.
- **Enable Fader Panel** – Shows or hides the fader wheel panel.
- **Enable Fixture Panel** – Shows or hides the fixture controls panel.*
- **Additive Selection** – Enables/Disables the additive selection mode.
- **Channel Selection** – Enables/Disables channel selection.*
- **Local Channel Numbers** – Enables/Disables local channel numbers.
- **Show Unpatched Channels** – Shows/Hides channels that are not patched to fixtures.*
- **Show Fixture Colors** – Shows/Hides the approximate color based on a fixtures attributes.*
- **Fixture Layout** – Additional visual layout configurations for how fixtures are laid out.*

*only available while fixtures are patched

Using the Select Menu

Use the Select ▾ menu to make quick selection modifications.

The **Select** menu will operate differently depending on the current selection. If there is an existing selection, this menu operates as a filter for that selection. If there is not and existing selection, this menu operates as a selection operation.

The selection operations are:

- **All** – Selects all channels or fixtures.
- **Active** – Selects active channels or fixtures.
- **Even** – Selects even channels or fixtures
- **Odd** – Selects odd channels or fixtures.
- **Every 3rd** – Selects every 3rd channel or fixture.
- **Every 4th** – Selects every 4th channel or fixture.
- **Random** – Selects an assortment of channels or fixtures at random.
- **Next** – Shift the current selection 1 place forward.
- **Previous** – Shift the current selection 1 place backward.
- **Group *n*** – Selects the channels or fixtures contained in the selected Group.
- **From Cue *n*** – Selects the channels or fixtures contained in the selected Cue.
- **Clear Selection** – Clears the current selection.
- **Undo Selection** – Undo the most-recent selection change.
- **Redo Selection** – Redo the most-recent undo operation.
- **Select Effects** – Selects the active playback effects and opens the effects panel.*

*only available while fixtures are patched

Using the Edit Menu

Use the Edit ▾ menu to take quick actions on the current selection.

The operations are:

- **Copy / Paste** – Copy or paste the selected channel or fixture values.
- **Release** – Released the selected fixtures or channels.
- **Park / Unpark** – Parks or Unparks the selected channels or fixtures (*only visible in live or output layer*).
- **Lock / Unlock** – Locks or Unlocks the selected channels or fixtures (*only visible in active layer*).
- **Enable / Disable** – Enables or disabled the selected channels or fixtures (*only visible in active layer*).

- **Rem Dim** – Remainder DIM turns all but the current selection off.
- **Assert** – Raises the priority of selected LTP channels or fixtures in the active playback fader

Using the Effects Button

Use the fx button to open the effects panel for the active playback.

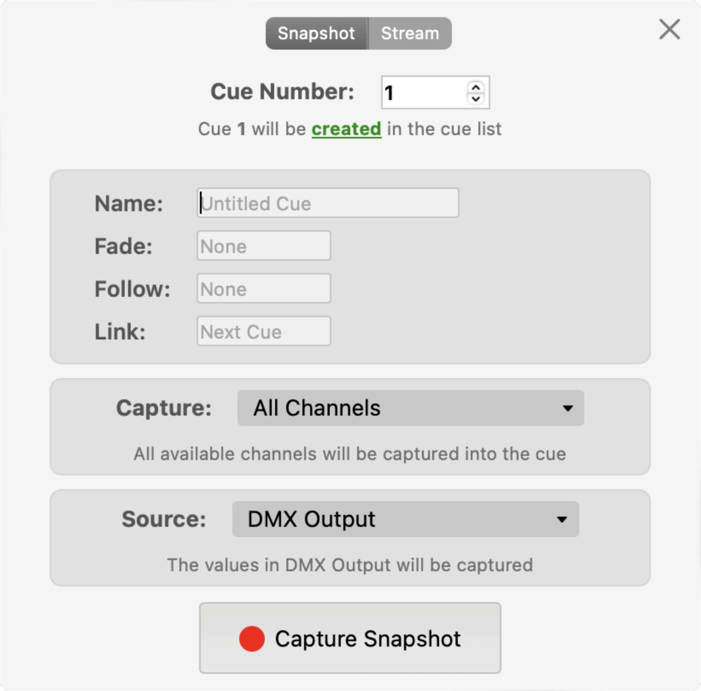
Record and Playback Tools

Use the Cue List ▾ menu to choose the currently active **Cue List**.

The active **Cue List** defines where Cues will be played from, captured to, or updated. These options include any defined Zones, Cue Stacks, and the standard *Cues* list (*default*).

Use the Record ▾ button to open the record panel, or update an existing Cue in the active **Cue List**.

The record panel allows you to provide, or update, Cue properties such as the name, number, fade, follow, and the link cue in addition to capture or re-capture a Cue’s contents. The **Capture** and **Source** menus can be used to refine what gets captured into the Cue, and from where.



Cue Number: 1

Cue 1 will be created in the cue list

Name: Untitled Cue

Fade: None

Follow: None

Link: Next Cue

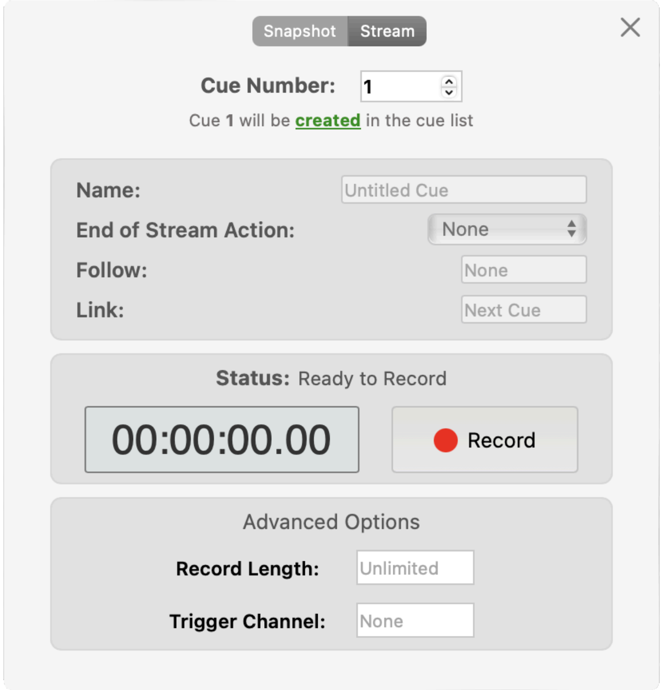
Capture: All Channels ▾

All available channels will be captured into the cue

Source: DMX Output ▾

The values in DMX Output will be captured

If capturing or updating a streaming Cue, you can choose the **End of Stream Action** and define a **Record Length** or **Trigger Channel** to ensure you capture (*or re-capture*) a precise number of frames.



Cue Number: 1

Cue **1** will be **created** in the cue list

Name: Untitled Cue

End of Stream Action: None

Follow: None

Link: Next Cue

Status: Ready to Record

00:00:00.00

Advanced Options

Record Length: Unlimited

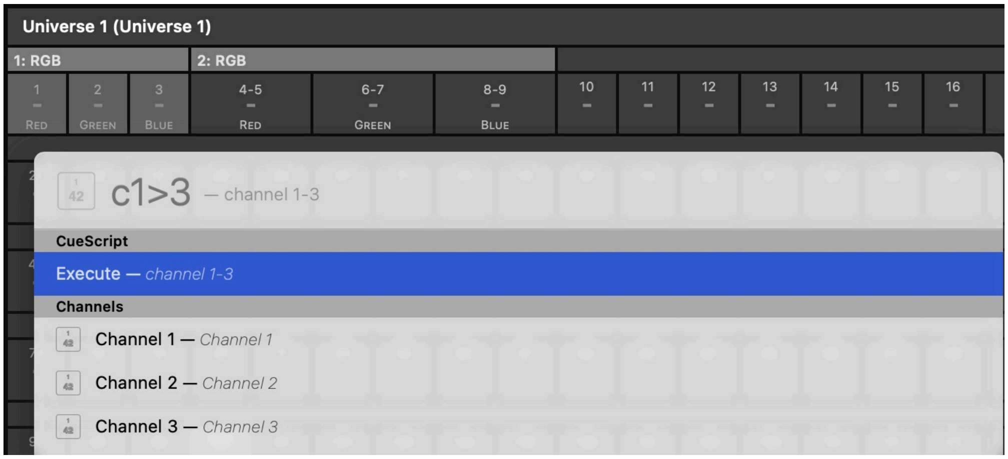
Trigger Channel: None

Use the Go → ▾ button to execute the next Cue, or launch a specific Cue from the active **Cue List**.

Using the Search Feature

Use the 🔍 feature to find items or execute CueScript selectors.

The **Search** panel highlights channels or fixtures that match your query as you type and provides a listing of the matching items which can be selected to highlight and bring them into view. You can also execute your query as CueScript directly from the search panel.



c1>3 — channel 1-3

CueScript

Execute — *channel 1-3*

Channels

Channel 1 — *Channel 1*

Channel 2 — *Channel 2*

Channel 3 — *Channel 3*

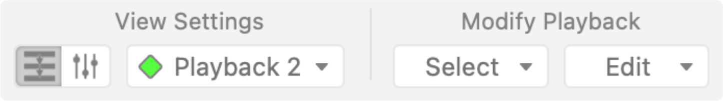
The Stage view can also be accessed from a web browser by navigating to `http://<ip-of-CueServer>/stage`

Playbacks

Overview

The *Playbacks View* shows the current state and properties of the Input, Playback Faders, and Output layers and can be displayed in two formats. This view provides a plethora of mechanisms for controlling Playbacks and the Outputs of CueServer.

The control ribbon along the top of the playback view provides easy access to playback selection and controls.



Use the view toggle (☰ 🎚) to switch between view styles.

The Flowchart View (☰) is arranged as a uni-directional array of sources, starting with Input, flowing through each of the Playbacks, and then finally to the Output layer. This represents the true flow of DMX content through CueServer. Each layer's output is combined with the next.

The Faders View (🎚) is arranged as a panel of vertical submaster faders. Each with playback controls and readouts for current state, next state, and Playback status.

Using the Select Menu

Use the Select ▾ menu to select one ore more Playbacks.

Selection Options:

- **All Playbacks** – selects all Playbacks
- **Playback (*n*)** – select a specific Playback
- **Next Playback** – select the next Playback
- **Previous Playback** – select the previous Playback
- **Empty Playbacks** – select only the empty Playbacks
- **Non-Empty Playbacks** – select only the active Playbacks

Using the Edit Menu

Use the Edit ▾ take quick actions on all the selected Playbacks.

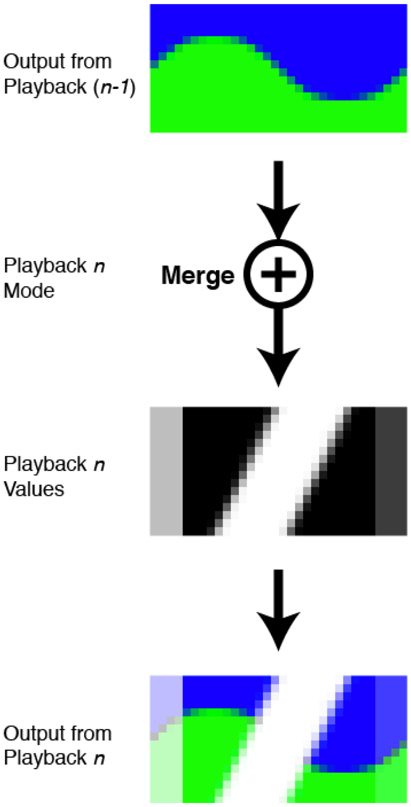
Edit Options:

- **Go** – play the next Cue in selected Playbacks.
- **Go Cue…** – play a selected Cue from the selected Playback’s active Cue List.
- **Go First Cue** – play the first Cue in the selected Playback’s active Cue List.
- **Set Next Cue…** – set a selected Cue as the next Cue for selected Playbacks.
- **Clear Stack…** – un-assign Cue Stack’s from the selected Playbacks (if any are assigned).
- **Set Cue Stack…** – assign a Cue Stack to the selected Playbacks (if any are defined).
- **Release** – release the selected Playback’s channels.
- **Clear** – clear the selected Playbacks.
- **Assert** – raise the priority of the selected Playback’s LTP (Latest Takes Precedence) channels.
- **Lock Selected Channels** – lock the currently selected channels in the active Playback.
- **Unlock All Channels** – unlock all channels in the selected Playbacks.
- **Enabled** – toggle the selected Playback’s enabled states.
- **Stopped** – toggle the selected Playback’s stopped states.
- **Merge** – set the selected Playbacks to **Merge** mode (*default*).
- **Override** – set the selected Playbacks to **Override** mode.
- **Scale** – set the selected Playbacks to **Scale** mode.
- **Pin** – set the selected Playbacks to **Pin** mode.
- **Crossfade** – set the selected Playbacks to **Crossfade** mode.
- **Mask** – set the selected Playbacks to **Mask** mode.

Playback Modes

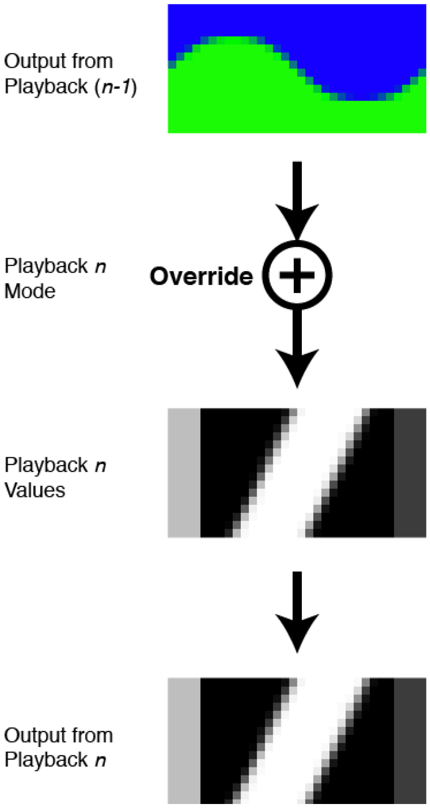
Each Playback’s values flow into, and are combined with, the next in sequential order. The mechanics of how Playback combinations occur is dictated by each Playback’s mode property.

By default, playbacks operate in **Merge** mode.



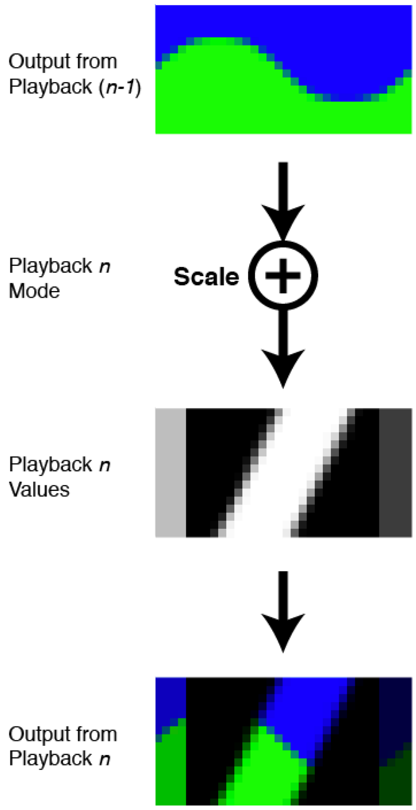
Merge combines a playback’s channel values with the previous playback in a Highest Takes Precedence (HTP) manner. This yields the highest value between the Playback’s input values and the Playback’s channel values.

The transformation can be correlated to the *Lighten* blend mode.



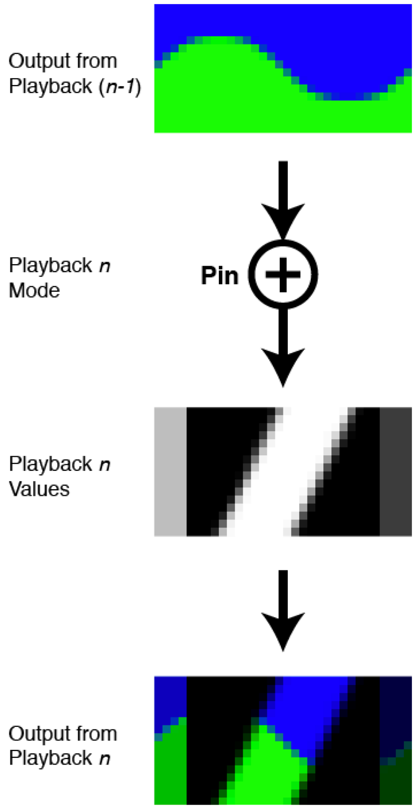
Override ignores the Playback’s input. This yields the Playback’s contents unchanged.

This transformation can be correlated to the *Normal* blend mode.



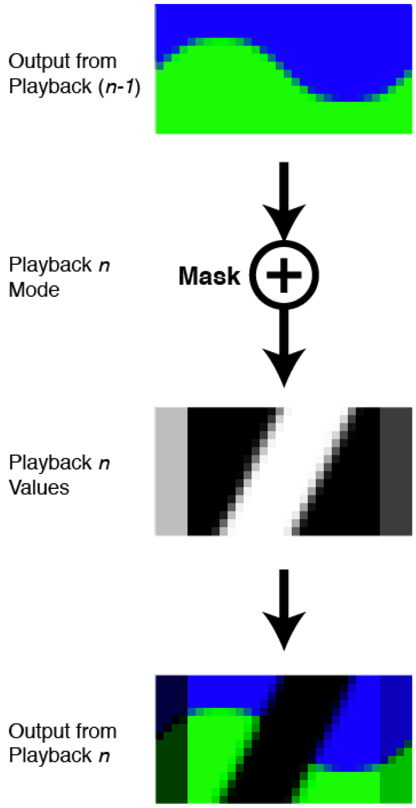
Scale multiplies the Playback’s channel values with its input values. This yields a scaled version of the Playback’s input.

This transformation can be correlated to the *Multiply* blend mode.



Pin clamps the Playback’s input values to its channel values, where each channel value cannot be higher than the playback’s value. This yields the inverse of **Merge** mode.

This transformation can be correlated to the *Darken* blend mode.

 Output from Playback (*n-1*) Playback *n* Mode Mask Playback *n* Values Output from Playback *n*	**Mask** multiplies the inverse of the Playback’s channel values with its input values, resulting in an inverted scaled version of the input. This yields the inverse of the **Scale** mode. This transformation can be correlated to the *Subtract* blend mode.

The Playbacks view can also be accessed from a web browser by navigating to `http://<ip-of-CueServer>/playbacks`

Flowchart

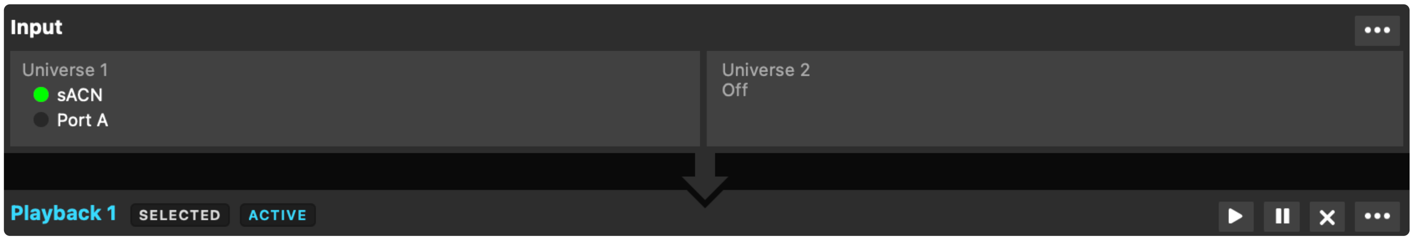
Overview

The flowchart depicts the input, each of the Playbacks, and the output including information about the current and future states of each. The chart is arranged as a uni-directional array of sources, beginning with Input, flowing through each of the Playbacks, and then finally to the Output layer. This model represents the true flow of DMX content through CueServer.

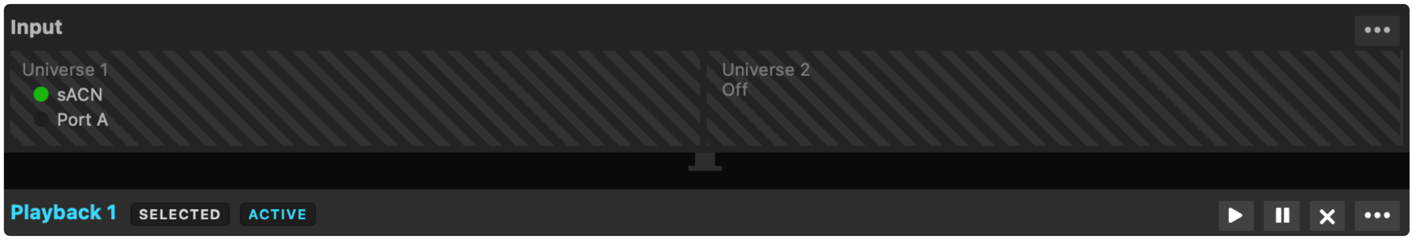
The input layer shows an overview of the CueServer inputs and universes, including the states of each input and whether or not input is enabled. The Playback layers provide an overview of each playback, including its current status, next status, and various properties. The output layers displays the an overview of the CueServer's outputs and universes, along with the states of each.

Input

This layer shows the patched universes and the inputs assigned to each. Next to each input source is a status indicator that conveys the current state of that source. Solid green indicates that a signal is being received from the source, while black indicates that no signal is being detected. The status indicator may also blink red or yellow if there is a problem with the source.



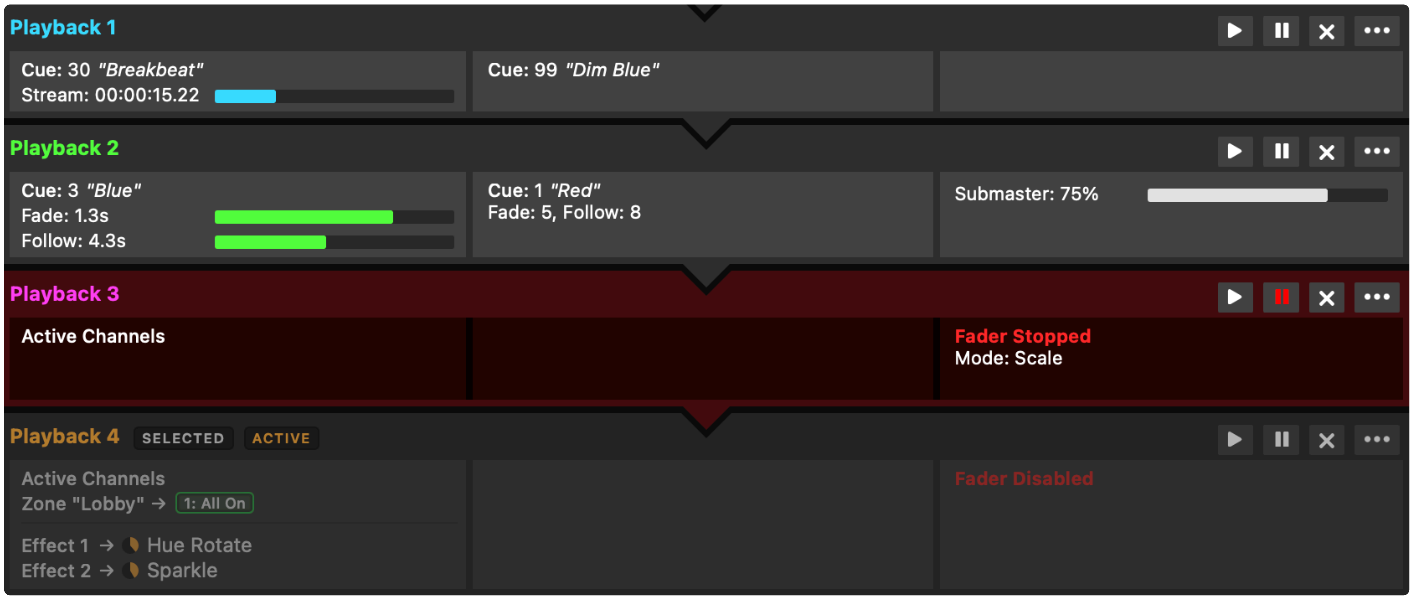
Click the ellipsis (•••) button in the top-right to enable or disable the input layer. When input is disabled CueServer will not pass input values through to the playbacks, the input layer's arrow will be broken, and a crosshatch pattern will cover its universe blocks.



More information about indicator states can be found in the DMX Ports section.

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: 00:00:15.22

Cue: 99 *"Dim Blue"*

Playback 2

Cue: 3 *"Blue"*
Fade: 1.3s
Follow: 4.3s

Cue: 1 *"Red"*
Fade: 5, Follow: 8

Submaster: 75%

Playback 3

Active Channels

Fader Stopped
Mode: Scale

Playback 4 SELECTED ACTIVE

Active Channels
Zone "Lobby" → 1: All On

Effect 1 → Hue Rotate
Effect 2 → Sparkle

Fader Disabled

In the example above, Playback 1 is currently playing back Cue 30, which is a streaming cue called "Breakbeat". It is currently 15.22 seconds into the stream. The next cue in Playback 1 is Cue 99, which is called "Dim Blue". Playback 2 is currently fading into Cue 3 "Blue". The fade has 1.3 seconds remaining, and a follow timer is running with 4.3 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%. Playback 3 has manually set "active" DMX channels in it and no next cue. Playback 3 is "stopped", meaning that fade and follow timing is disabled, and its layer mode is set to "Scale". Playback 4 is the active playback, it has Preset 1, named "All On", active in the zone "Lobby". Playback 4 also has two effects running, "Hue Rotate" and "Sparkle", but is currently disabled, so it will not contribute to the DMX output.

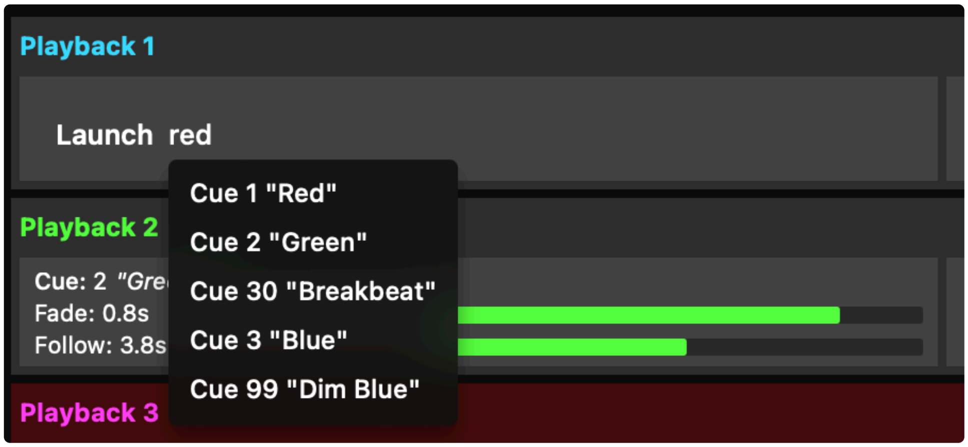
The Current Pane (Left Side)

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 progress bar and numerical countdown show the fade time remaining.

- **Follow (*time*)** – Shown when the Playback is counting down to an auto-follow event. A progress bar and numerical countdown show the follow time remaining.
- **Stream (*time*)** – Shown when a Streaming Cue is being played back. A progress bar and numerical countdown show the stream time remaining.

Left-click on the pane to type in a Cue number, name, or keyword. Typing a number followed by the enter key will immediately launch that Cue. Typing a name or keyword will bring up a list of matching Presets and Cues to select from. You can click an item in the list, or use the arrow keys to traverse the options and then press enter on the option you want. Click outside of the pane, or press the escape key, to cancel.



Launch red

Cue: 2 *"Gre*
Fade: 0.8s
Follow: 3.8s

Right-click on the current pane to access a quick-launch menu with options to clear the Playback, or select a Cue/Preset from a list.

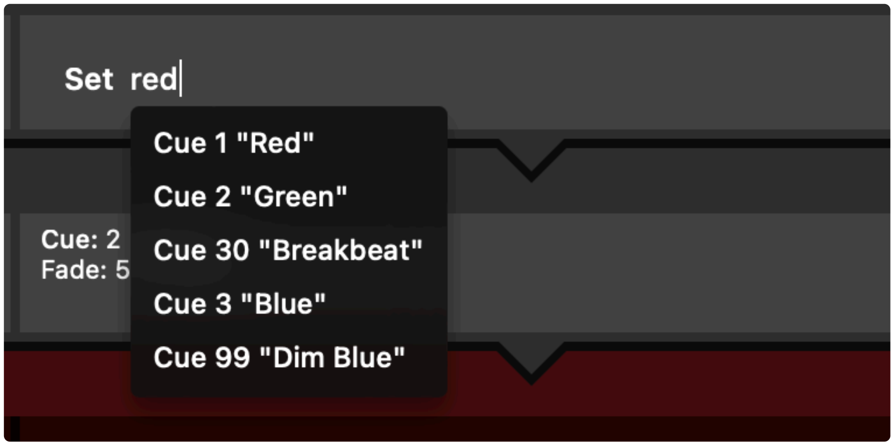
Click on the Playback title, or name if defined, to quickly set the Playback as active.

The Next Pane (Center)

The panel in the center of each Playback shows what is queued to be “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*.

Left-click on the pane to type in a Cue number, name, or keyword. Typing a number followed by the enter key will set that Cue as next. Typing a name or keyword will bring up a list of matching Presets and Cues to select from. You can click an item in the list, or use the arrow keys to traverse the options and then press enter on the option you want. Click outside of the pane, or press the escape key, to cancel.



Set red

Right-click on the next pane to access a quick-set menu with options to clear the next pane, or set a Cue/ Preset from a list.

The Properties Pane (Right Side)

The panel on the right-hand side of each Playback shows additional properties for the Playback Fader. Right-clicking on the properties pane will present a quick-action menu to enable/disable the playback or change the playback mode.

The properties that appear in the status pane are:

- **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 its 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 Locked** – Shown in a yellow color when channels in the Playback are locked. Locked channels retain their current values and cannot be modified by executing cues or by using the Channel, At, Release, or Clear commands. Locked channels must either be Unlocked, or the CueServer can be Reset.
- **Submaster (*level*)** – Shown when the Playback's submaster level is not at 100%. A white progress bar shows the submaster percentage. Click and drag on the progress bar to set its level.
- **Mode (*mode*)** – Shown if the Playback's combine mode is set to anything other than the default "Merge" mode. Options include **Override**, **Scale**, **Pin**, **Crossfade**, and **Mask**.

At the top-right of this pane you will find a set of 4 buttons:

- **Play Button** (▶) – play the next Cue
- **Pause Button** (⏸) – start/stop the Playback fader
- **Clear Button** (☒) – clear the Playback fader
- **Ellipsis Button** (•••) – open the Playback menu

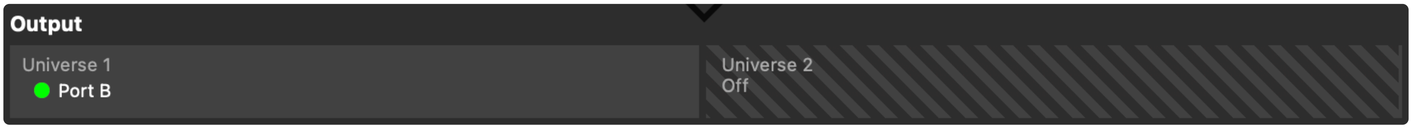
The playback menu contains the following options:

- **Go** – play the next Cue.
- **Go Cue…** – play a selected Cue from the Playback's active Cue List.
- **Go First Cue** – play the first Cue in the Playback's active Cue List.
- **Set Next Cue…** – set a selected Cue from the Playback's active Cue List as the next Cue.
- **Set Cue Stack…** – assign a Cue Stack to the Playback (if any are defined).
- **Release** – release the Playback's channels.
- **Clear** – clear the Playback.
- **Assert** – raise the priority of the Playback's LTP (Latest Takes Precedence) channels.
- **Active** – set the Playback as active.
- **Previous Playback** – set the previous Playback as active.
- **Next Playback** – set the next Playback as active.
- **Lock Selected Channels** – lock the currently selected channels in the Playback.
- **Unlock All Channels** – unlock all channels in the Playback.
- **Submaster…** – set the Playback's submaster level from a list of options.
- **Enabled** – toggle the Playback's enabled state.
- **Stopped** – toggle the Playback's stopped state.
- **Merge** – set the Playback to **Merge** mode (*default*).
- **Override** – set the Playback to **Override** mode.
- **Scale** – set the Playback to **Scale** mode.
- **Pin** – set the Playback to **Pin** mode.
- **Crossfade** – set the Playback to **Crossfade** mode.
- **Mask** – set the Playback to **Mask** mode.

Output

This layer shows the patched universes and the outputs assigned to each. Next to each output is a status indicator that conveys the current state of that output. Solid green indicates that a signal is being sent, while black indicates that no signal is being sent. The status indicator may also blink red or yellow if there is a problem with the output.

Left-click on a universe block to enable or disabled it. When a universe is disabled, it will appear with a crosshatch pattern and no longer output data via sACN, Art-Net, or KiNet.



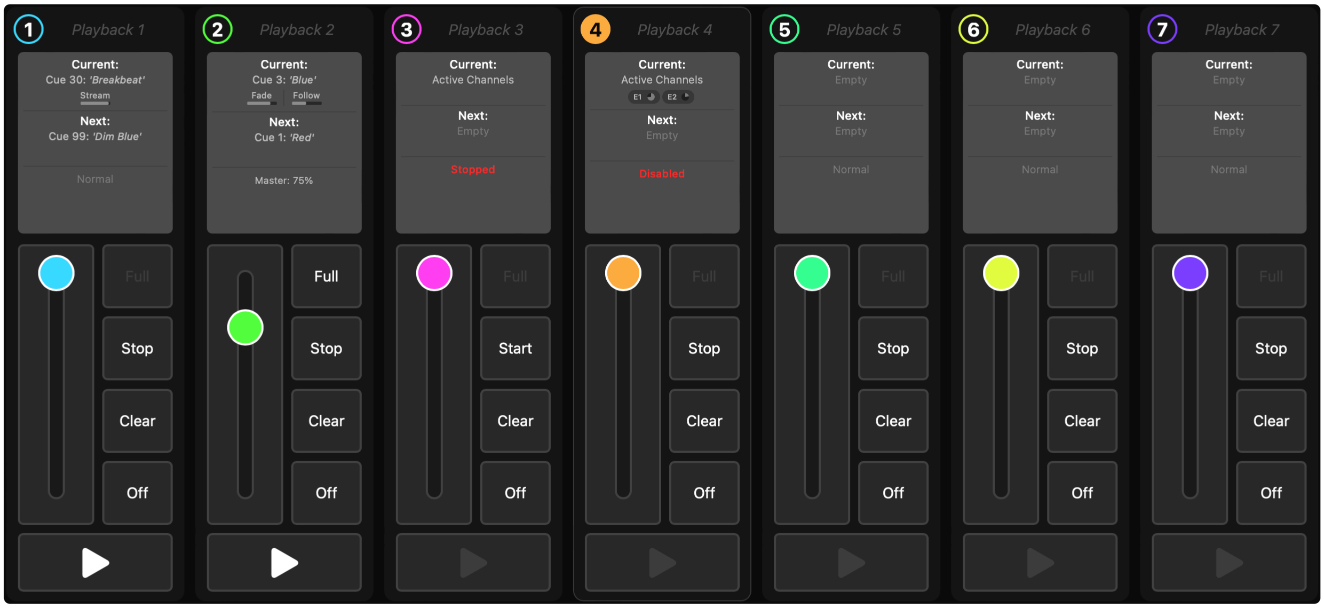
More information about indicator states can be found in the DMX Ports section.

Faders

Overview

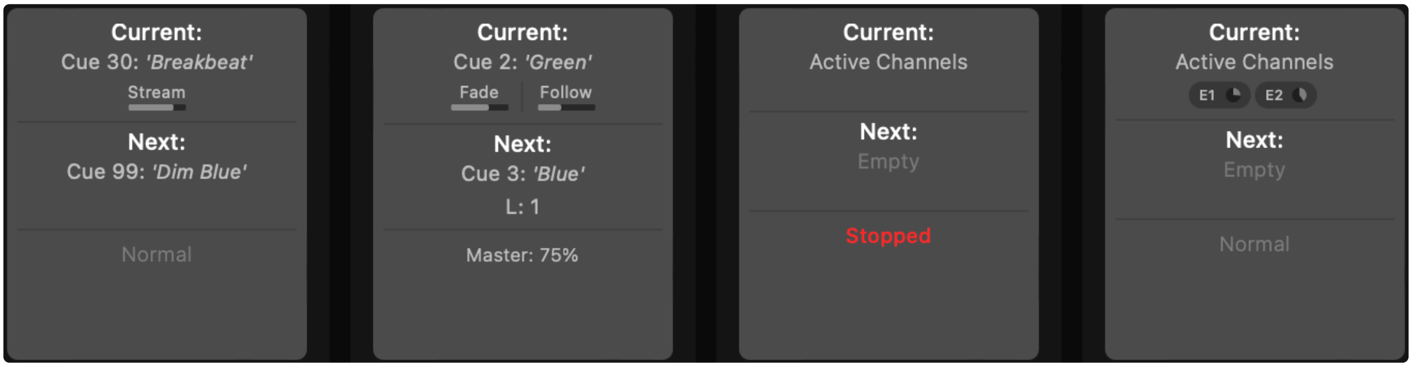
The fader view focuses on playbacks alone, with a set of fader controls laid out horizontally. Each control displays the Playback name and number, current and next state, and current properties in a condensed format.

Large submaster sliders are provided for easy adjustment of submaster levels, as well as large buttons for common shortcuts like Go, Stop, Clear, Full and Off. The active playback will appear with it’s number circle filled-in and a semi-transparent white border. Click on a Playback’s number field, or make adjustments using the controls within, to set the playback as active.



Information Panel

The information panel gives a brief overview of what the playback is doing. This panel shows three sections; Current, Next, and Properties.



Current Section

The Current section shows what Cues, Presets, and Effects are active, and the fade, follow, or stream progress.

Next Section

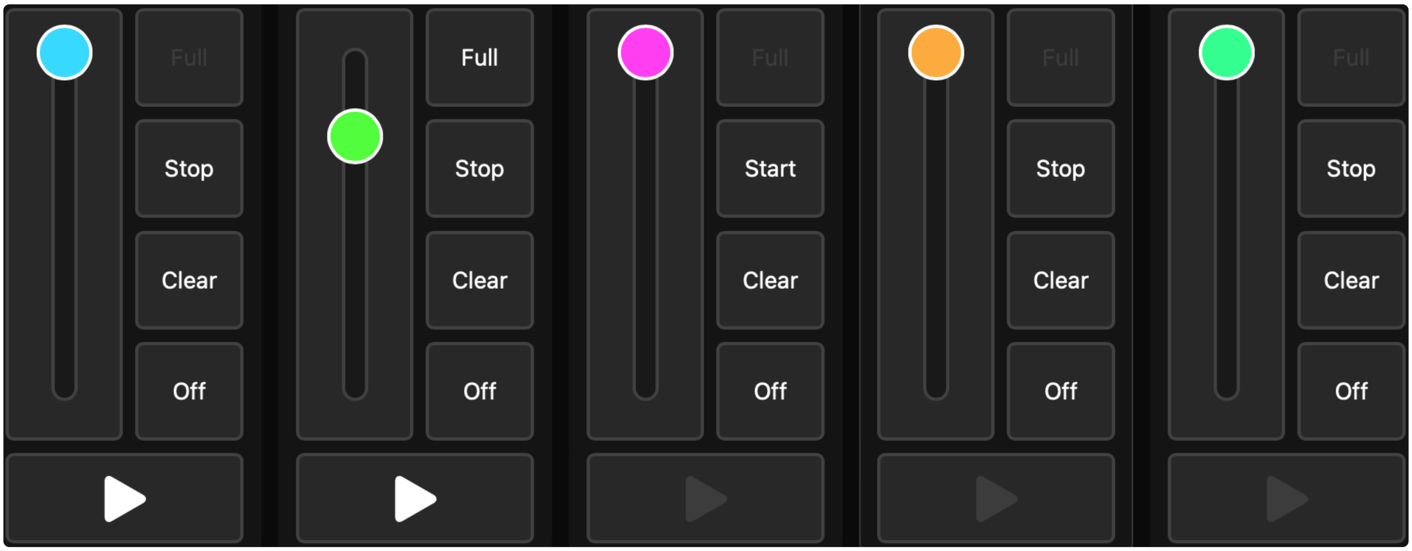
The Next section shows whether the playback has a Cue on-deck to play next, and whether that cue has a Link cue. If there is a current Cue, and it has a follow time, the cue in the Next section will launch after the current Cue's follow timer expires. If there is not a current Cue, or the current Cue has no follow time, the Cue in the Next section will run the next time a "go" command is issued.

Properties Section

The properties section shows states that affect the Playback, like the submaster level and whether the Playback is stopped or disabled. If all of the Playback's properties are set to defaults, this section reports "Normal".

Fader Controls

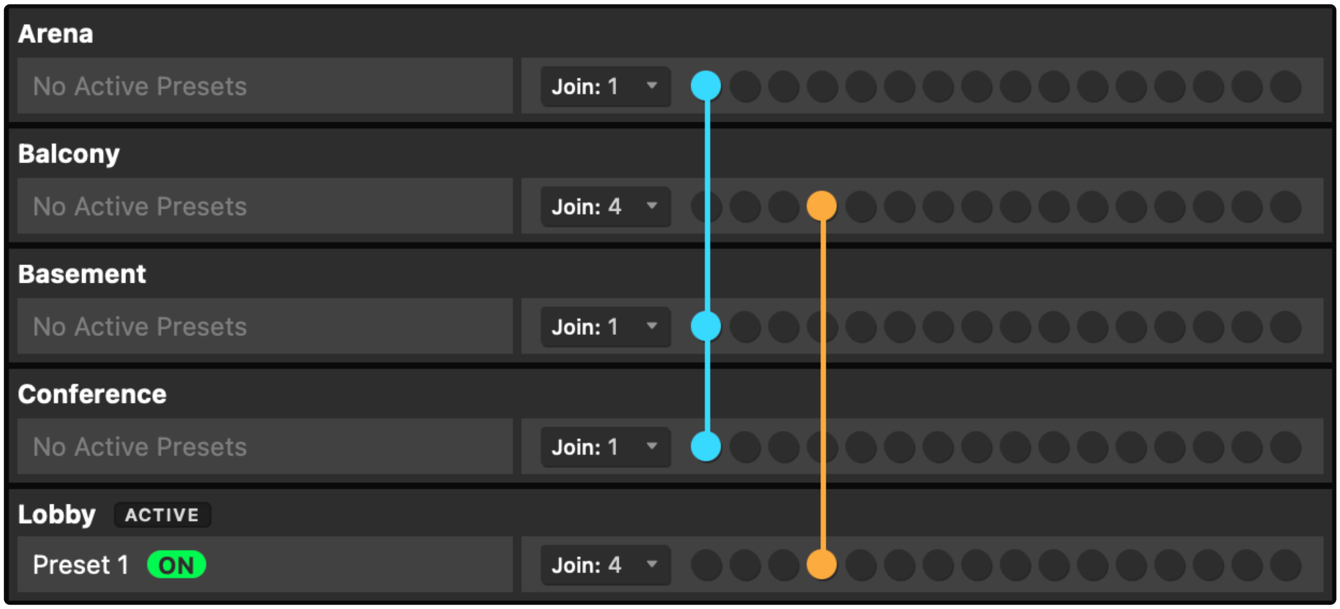
At the bottom of each Playback fader is has a set of buttons and controls to manipulate the Playback. The slider on the left-hand side controls the submaster level. This slider is designed to operate instantaneously, meaning that if a global time (fade time) is set this slider will not adhere to it. To the right of the slider are 4 buttons: Full which will immediately set the submaster level to full, Stop which will stop or start the Playback fader, Clear which will clear the Playback, and Off which will immediately set the submaster to 0. Finally, at the bottom of each Playback fader is a Go button which will step to the next cue. If no next cue is defined, this button is disabled.



Zones

Overview

The zones view provides a quick overview of the current Zones, and is structured vertically as a series of panels with two sections.



Arena
No Active Presets	Join: 1

Balcony
No Active Presets	Join: 4

Basement
No Active Presets	Join: 1

Conference
No Active Presets	Join: 1

Lobby ACTIVE
Preset 1 ON	Join: 4

Zone Status

The left side shows the state of each defined Zone, including which zone is active (if any) and the state of each Zone’s presets. In the example above, you can see that “Lobby” is the only Zone with active presets, and it’s also the active Zone.

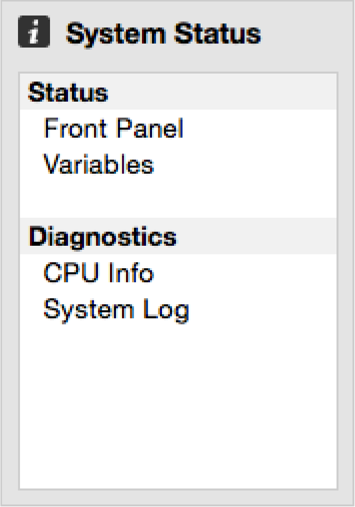
Zone Joins

The right side shows the current Join group for each Zone (if in a join group). This is depicted by a row of circles, each representing a different join number. Each join number is distinguished by color and horizontal position. When a Zone is in a join group, the circle for that join is turned on. If multiple Zones are joined to the same group, the circle representing that join number will light up in each of the joined Zones, aligning vertically, and a line in the color of that join group will be drawn visually connecting them together.

You can set a Zone’s join group by using the drop-down selector, or by clicking on the circle that represents the desired join group.

Status

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



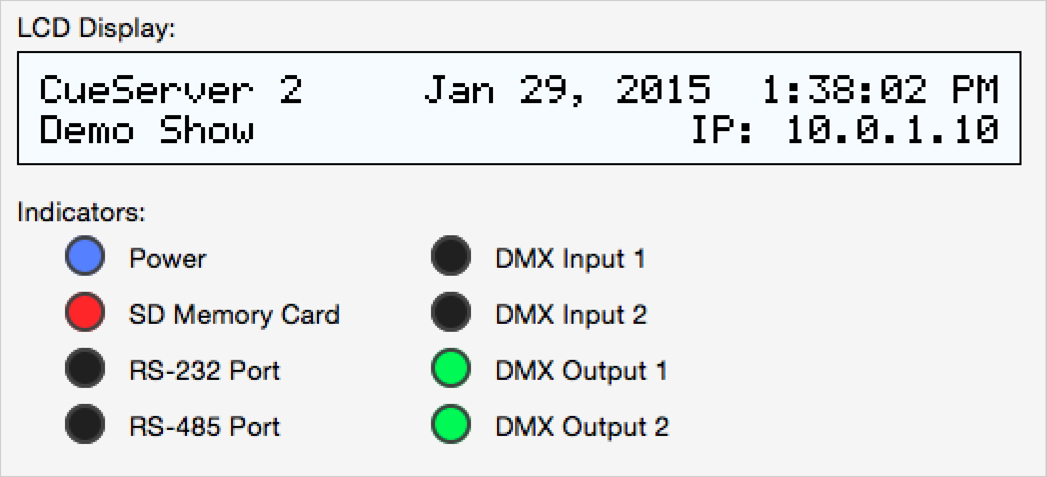
The following status views are available:

- 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.



LCD Display:

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

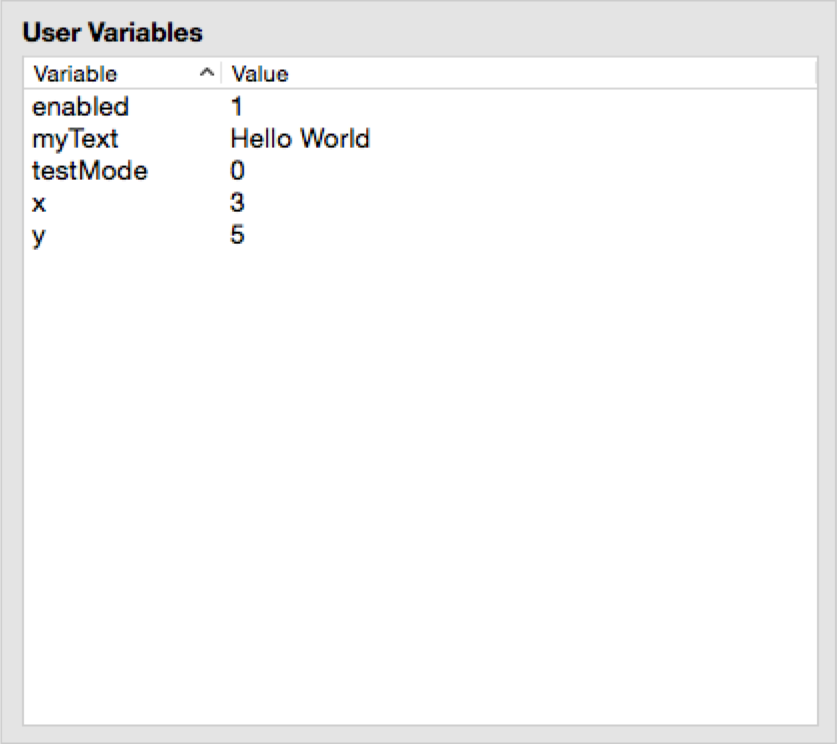
Indicators:

🔵	Power	⚫	DMX Input 1
🔴	SD Memory Card	⚫	DMX Input 2
⚫	RS-232 Port	🟢	DMX Output 1
⚫	RS-485 Port	🟢	DMX Output 2

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

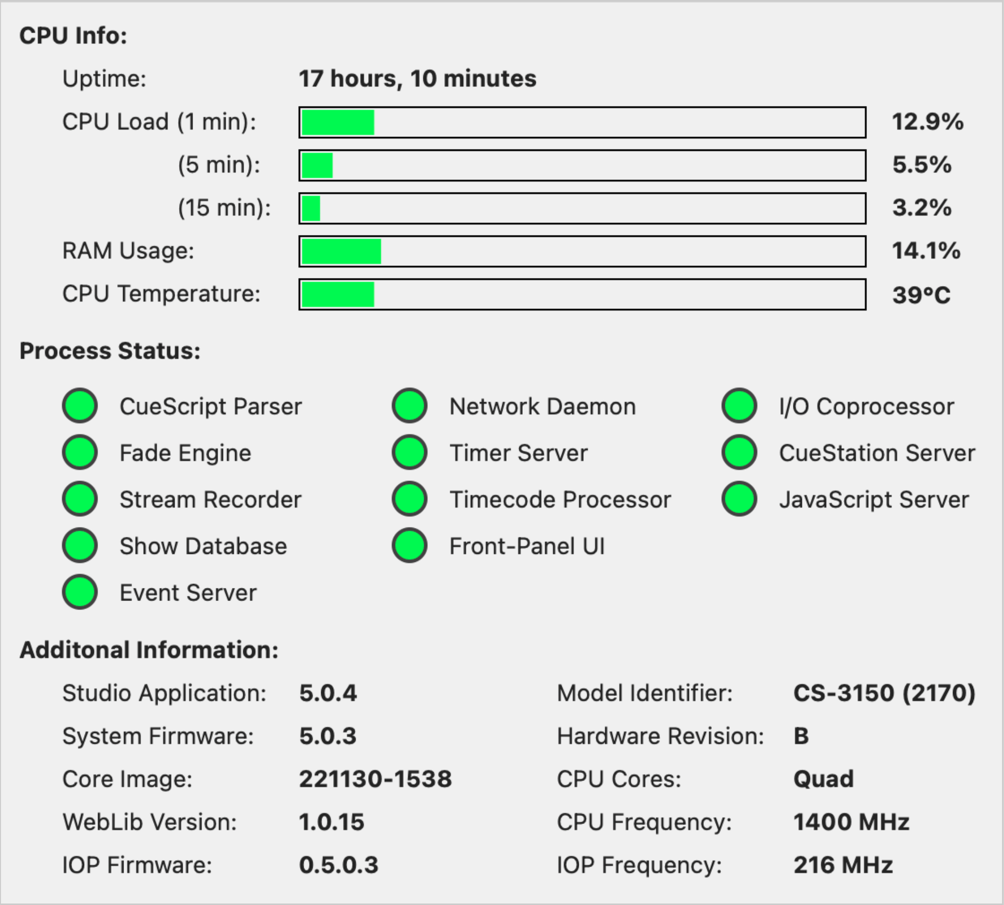
Variable	Value
enabled	1
myText	Hello World
testMode	0
x	3
y	5

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:	**17 hours, 10 minutes**
CPU Load (1 min):	**12.9%**
(5 min):	**5.5%**
(15 min):	**3.2%**
RAM Usage:	**14.1%**
CPU Temperature:	**39°C**

Process Status:

🟢 CueScript Parser	🟢 Network Daemon	🟢 I/O Coprocessor
🟢 Fade Engine	🟢 Timer Server	🟢 CueStation Server
🟢 Stream Recorder	🟢 Timecode Processor	🟢 JavaScript Server
🟢 Show Database	🟢 Front-Panel UI	
🟢 Event Server		

Additonal Information:

Studio Application:	**5.0.4**	Model Identifier:	**CS-3150 (2170)**
System Firmware:	**5.0.3**	Hardware Revision:	**B**
Core Image:	**221130-1538**	CPU Cores:	**Quad**
WebLib Version:	**1.0.15**	CPU Frequency:	**1400 MHz**
IOP Firmware:	**0.5.0.3**	IOP Frequency:	**216 MHz**

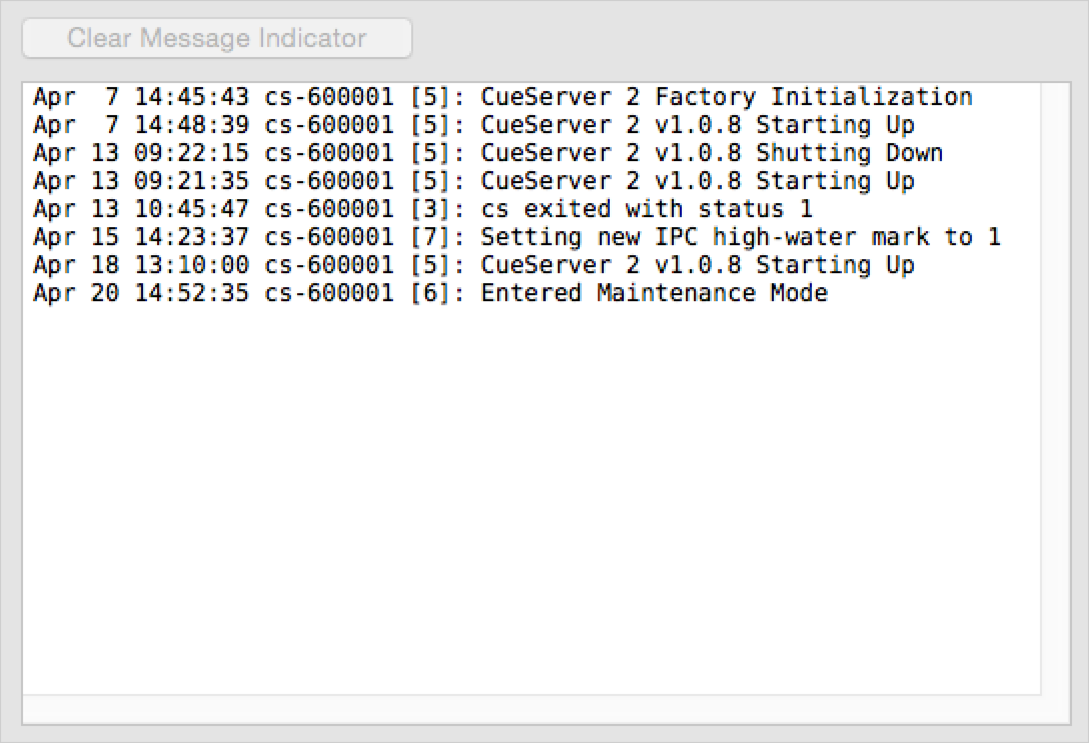
The following information is displayed:

- **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.

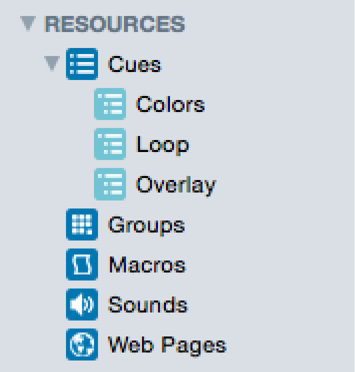
Also, user-defined messages can be added to the System Log by using the **Log** CueScript command.

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.



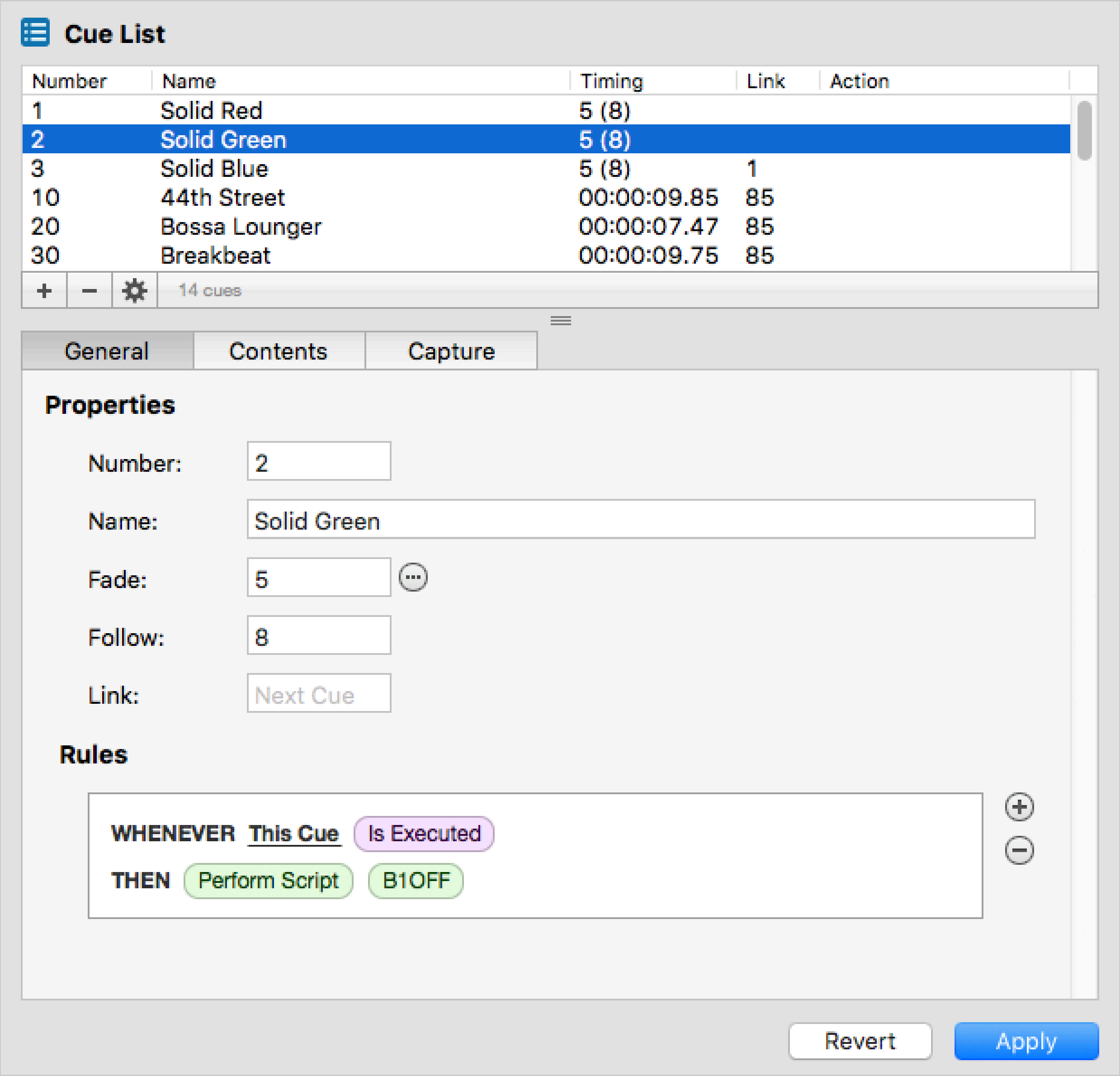
The following sections describe these views in more detail:

- Cues – scenes and timeline based streams
- Groups – definitions of groups of channels and fixtures
- Zones – definitions of zones
- Presets – definitions of presets
- Macros – user-defined scripts
- Functions – user-defined CueScript or JavaScript functions
- Sounds – audio clips
- Web Pages – custom web pages for the project
- Variables – view and update variables
- Stage Layouts – custom layouts

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	Solid Red	5 (8)		
2	Solid Green	5 (8)		
3	Solid Blue	5 (8)	1	
10	44th Street	00:00:09.85	85	
20	Bossa Lounger	00:00:07.47	85	
30	Breakbeat	00:00:09.75	85	

14 cues

Properties

Number: 2

Name: Solid Green

Fade: 5

Follow: 8

Link: Next Cue

Rules

WHENEVER This Cue Is Executed

THEN Perform Script B1OFF

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, rules, and contents 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.

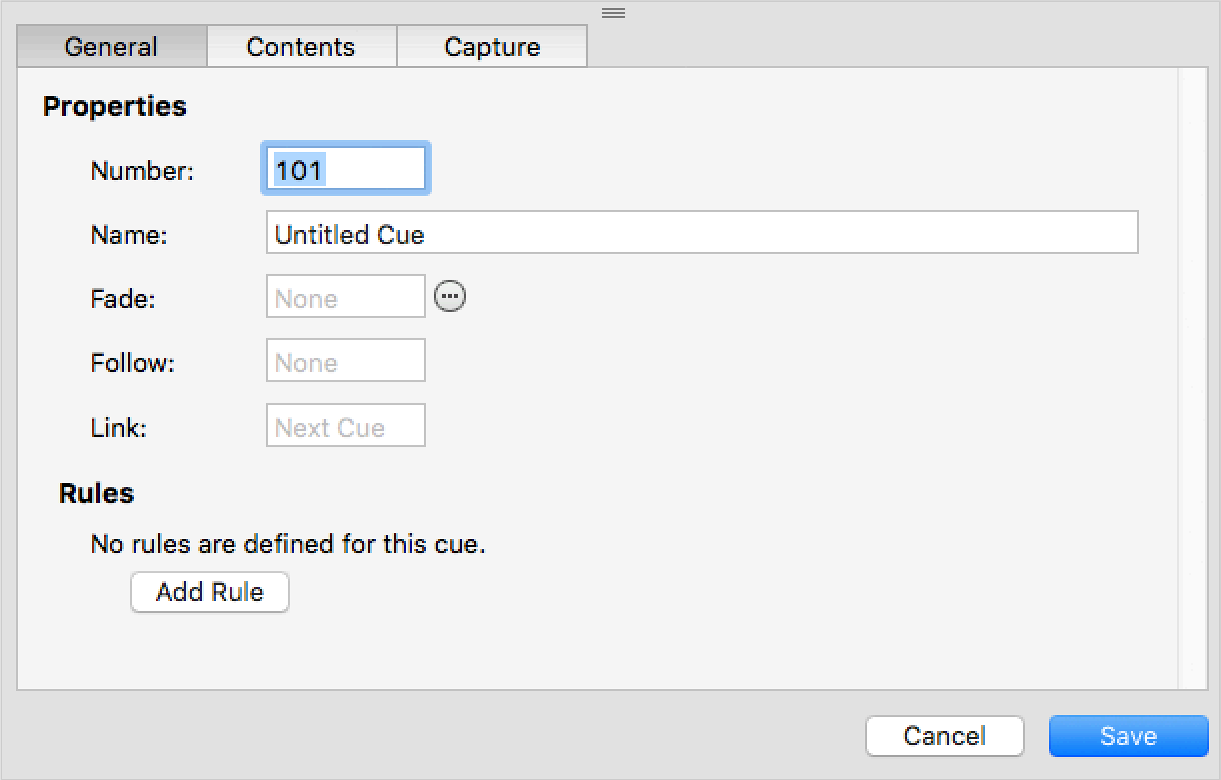
An analogy can be made between a streaming cue in CueServer, and a “tape recorder” for audio. When a streaming cue is captured in CueServer, every change to a DMX channel during the capture is saved. Then, when the streaming cue is played back, the changes occur in real-time just the same way that it 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. Or, choose the **New Cue…** item from the File menu.

A new empty cue will appear in the window:



Properties

Number: 101

Name: Untitled Cue

Fade: None

Follow: None

Link: Next Cue

Rules

No rules are defined for this cue.

CueServer Studio will automatically create the cue with the next available cue number already chosen. This number can be changed before saving the cue to use a different cue number.

The cue’s name, fade and follow times, link and rules can all be set by clicking into these fields.

The newly created cue does not have any channels recorded into it. To add content to this cue, click on the Capture button (Capture...). See the Cue Contents section for information about how to capture scenes and/or streams into cues.

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

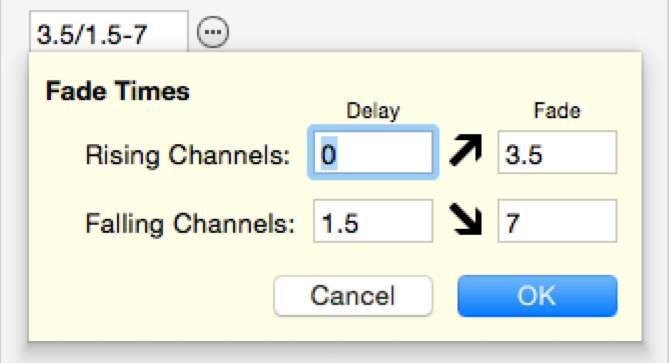
A cue may be given a descriptive name.

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.

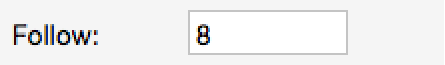
Mode (streaming cues only)

Mode: Follow

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

- **None** – When the stream finishes, playback stops and the last channel values remain active.
- **Loop** – When the stream reaches its last frame, it will seamlessly loop back to its beginning.
- **Follow** – When the stream finishes, the next cue automatically follows.
- **Release** – When the stream finishes, channels in the stream are released.

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.

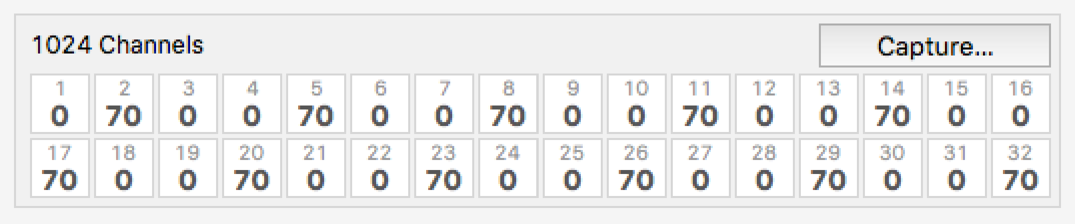
Cue Contents

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



1024 Channels

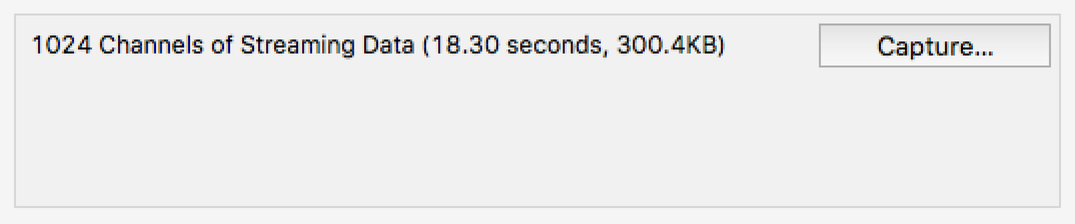
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	70	0	0	70	0	0	70	0	0	70	0	0	70	0	0
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
70	0	0	70	0	0	70	0	0	70	0	0	70	0	0	70

Normal cues contain a single snapshot of 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.

To capture a DMX snapshot, see the section Capturing DMX Snapshots.

Streaming Cue Data



1024 Channels of Streaming Data (18.30 seconds, 300.4KB)

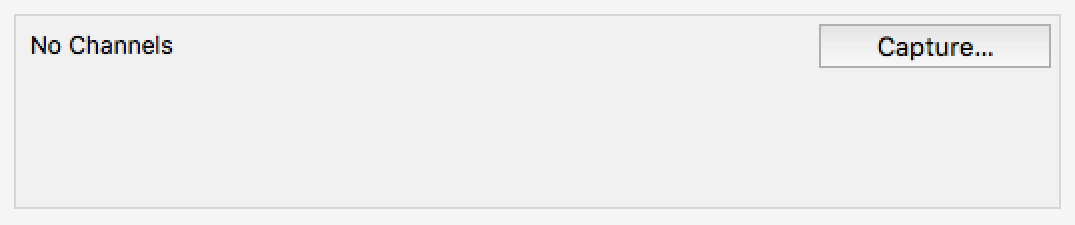
A streaming cue contains a recording of DMX data over a period of time.

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.

To capture a DMX stream, see the section Capturing DMX Streams.

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.

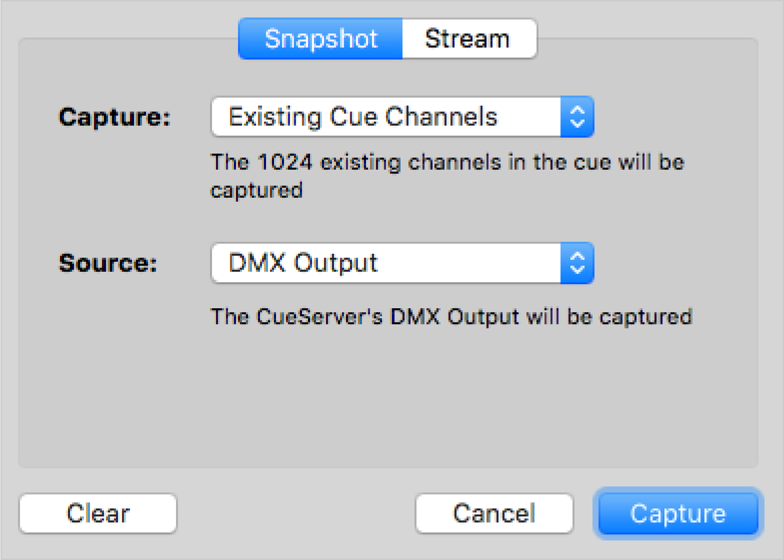
To clear a cue’s contents (creating an empty cue), see the section Clearing Cue Contents.

Capturing Content into a Cue

To record, change or clear the contents of a cue, click on the Capture button (Capture...).

Capturing DMX Snapshots

The **Snapshot** tab of the Capture window is used to capture a single snapshot of DMX channels into a cue:



Capture: Existing Cue Channels

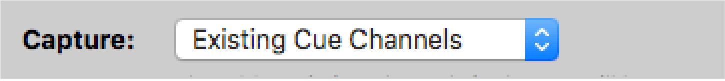
The 1024 existing channels in the cue will be captured

Source: DMX Output

The CueServer's DMX Output will be captured

This window has pop-up menus for choosing what DMX channels will be captured into the cue and from what source the channels will be captured.

Capture Menu



Capture: Existing Cue Channels

This menu has several options to specify which channels will be recorded into the cue:

- **All Channels** – Every channel in the system will be recorded. If the CueServer is configured with two universes of DMX then 1,024 channels will be captured.
- **Active Channels** – Only channels that have non-zero values will be recorded into the cue.
- **Selected Channels** – Only the currently selected channels will be recorded into the cue. The selected channels are the ones previously selected using the Channel and Group commands.
- **Existing Cue Channels** – Only the channels that are currently recorded in the cue will be re-recorded. If the cue previously contained channels 101 through 199, then those channels are the only ones that will be re-recorded.

Source Menu

Source: DMX Output

This menu has several options to specify the source of the DMX channel values that will be recorded into the cue:

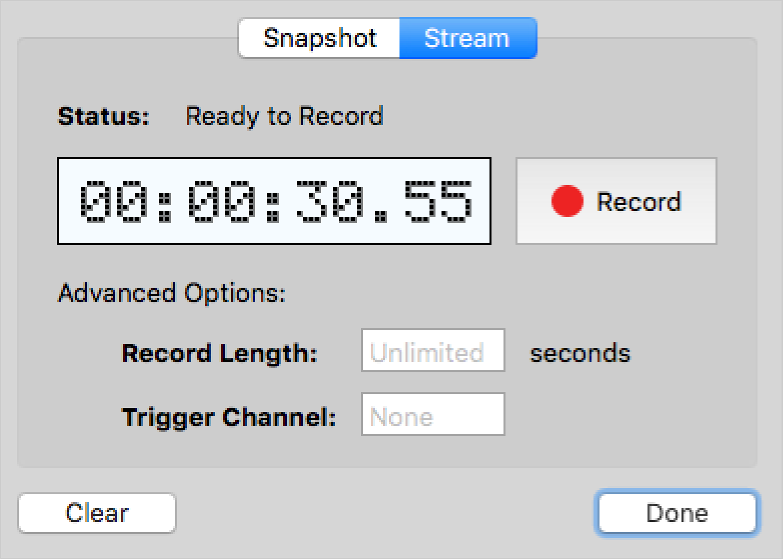
- **DMX Input** – The DMX channel values being input into the CueServer will be recorded. None of the values in the playbacks or being output will be recorded.
- **Playback n** – The DMX channel values in Playback *n* will be recorded. Neither the DMX input or output will be recorded.
- **DMX Output** – The DMX channel values being output from the CueServer will be recorded. This is the default option.

Capturing Channel Values

Once the appropriate options have been selected, click on the **Capture** button to record the current DMX values into the cue. The Capture window will close and the cue will be updated to show the newly captured channels.

Capturing DMX Streams

The **Stream** tab of the Capture window is used to capture a stream of changing DMX channel values into the cue.



Status: Ready to Record

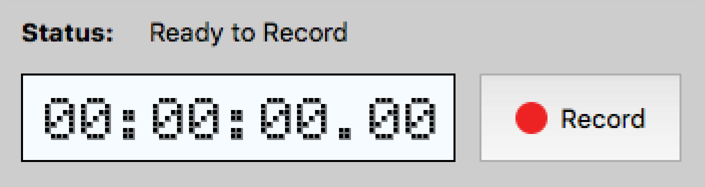
00:00:30.55

Record Length: Unlimited seconds

Trigger Channel: None

This window has controls for starting/stopping the stream recording and additional advanced options for controlling the length or external triggering of the stream recording.

Recording Controls



Status: Ready to Record

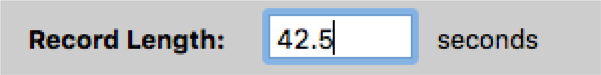
00:00:00.00

At the top of the window, there is a time display readout and a **Record** button.

The time display shows the current duration of stream recording that is in the cue. For new cues, this will show `00:00:00.00`. For cues with existing streaming data, the cue's current duration will appear in this display.

To start recording, press the **Record** button. The button will change to **Stop** and the time display will begin counting. Recording of DMX channel values will continue until the **Stop** button is pressed.

Record Length Option



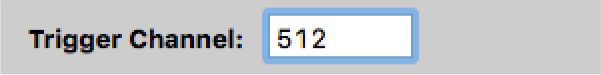
Record Length: 42.5 seconds

This field can be used to limit the length of the recording to a specific number of seconds. Any number of seconds may be entered down to 1/100th second precision.

When a Record Length has been specified, the stream recording will automatically stop after the length has been reached.

If this field is empty, recording will continue until the **Stop** button is pressed.

Trigger Channel Option

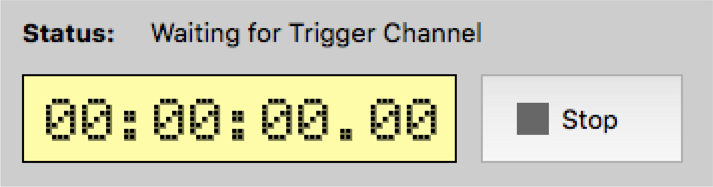


Trigger Channel: 512

This field can be used to specify a channel number that the CueServer should watch to automatically start and stop the stream recording. When the input channel rises above zero, the recording will start. Then, when the input channel falls back to zero, the recording will stop.

The typical use for this feature is to allow the external console that is sending DMX data to be able to start and stop the CueServer's recording by raising and lowering this "trigger channel". Any channel can be chosen, but it is typical to use a channel that is not being used by a dimmer or fixture.

When a trigger channel is specified, press the **Record** button to begin waiting for the trigger channel to rise above zero. As long as the trigger channel is being received as zero, the time display will wait to start recording:



Status: Waiting for Trigger Channel

00:00:00.00

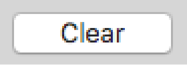
As soon as the external console raises the trigger channel above zero, the recording will begin automatically. Then, when the trigger channel falls back to zero, recording will stop.

CueServer very precisely monitors the value of the trigger channel and will begin recording a stream on the very first DMX frame that has a non-zero trigger channel value. The recording continues until the trigger channel becomes zero again. The last frame recorded is the frame received *just before* a frame arrives with a zero value trigger channel.

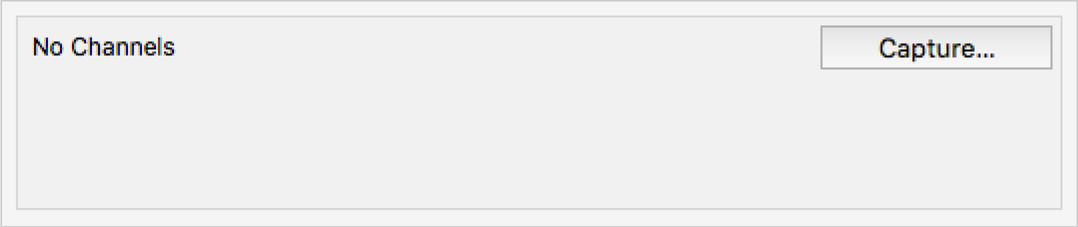
Clearing Cue Contents

Sometimes it may be desirable to create a cue that does not have any DMX channel values recorded in it. This is called an empty cue. An empty cue can be useful to provide additional timing steps in a list of cues, or that may have automation rules without affecting DMX channels, etc.

A cue with channel values can be cleared by clicking on the **Capture** button, and then clicking on **Clear** in the lower left corner of the capture window.



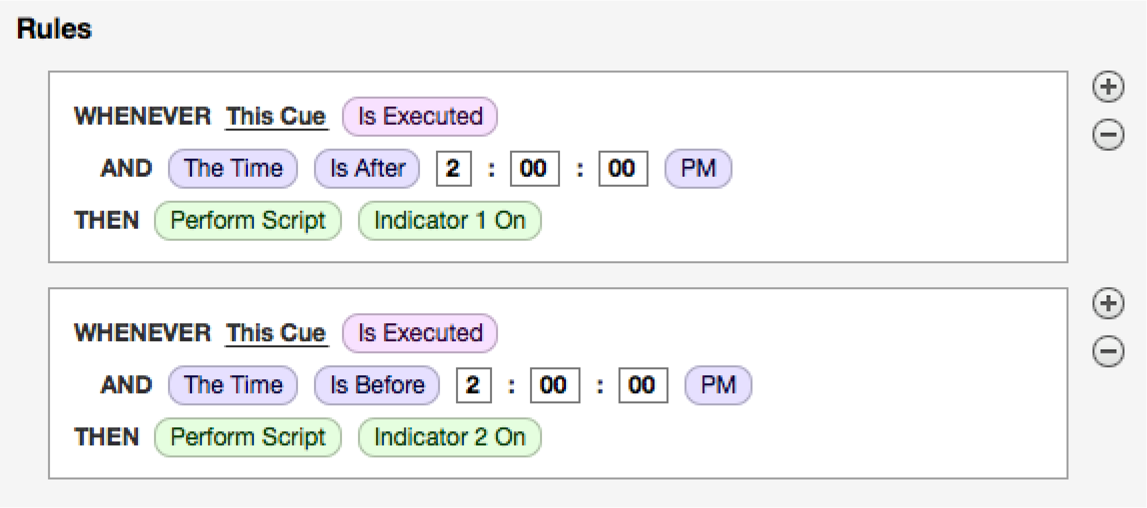
After **Clear** is clicked, the capture window will close and the cue will show that it no longer contains any channel values:


No Channels

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 (⊕).

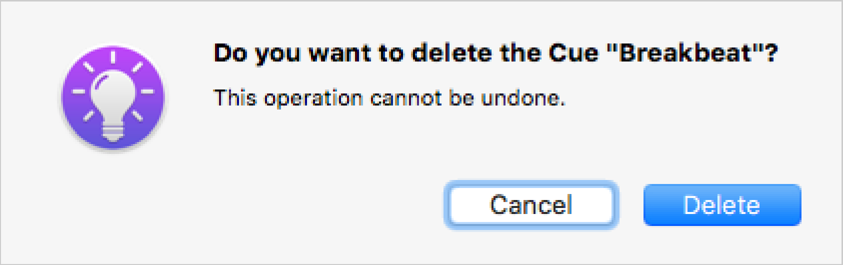
Then, click on the various “bubble” buttons (Choose...) in the rule to build an event, conditions and action that the rule will execute.

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

Deleting Cues

To remove a cue from the cue list, click the minus button (−) at the lower-left corner of the cue list.

A confirmation dialog will appear:



Do you want to delete the Cue "Breakbeat"?

This operation cannot be undone.

After confirmation of the delete operation, the cue will be removed from the cue list.

✻ You can also use the **Delete** or **Backspace** keys on your keyboard. To avoid the confirmation dialog, you can hold the **Option** or **Alt** key.

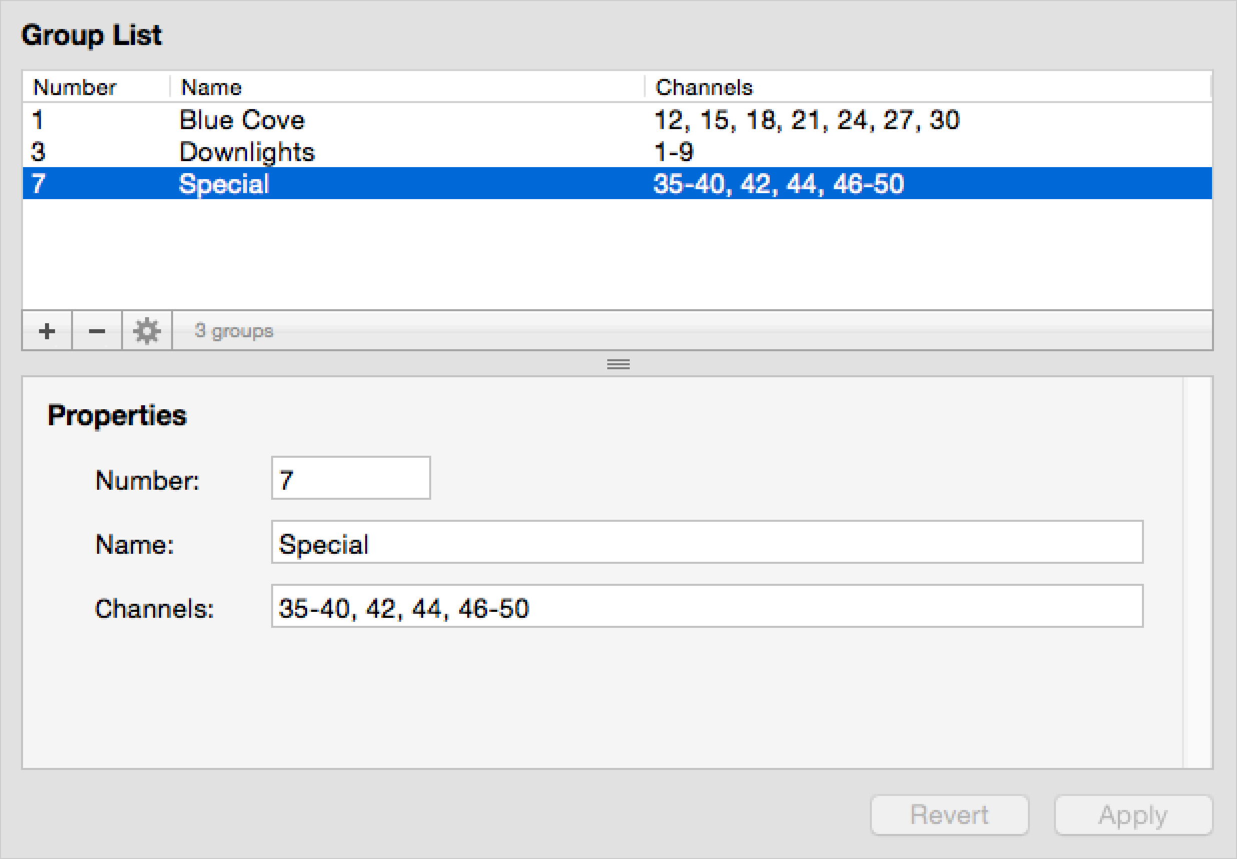
Groups

Overview

A group is a collection of one or more channels. Groups are used as shortcuts for recalling a specific set of channels. Groups can be used to select channels, set channel values and more.

The *Groups* editor is where groups are created, edited and removed from the project.

For information on using groups, see the Group command.



Group List

Number	Name	Channels
1	Blue Cove	12, 15, 18, 21, 24, 27, 30
3	Downlights	1-9
7	Special	35-40, 42, 44, 46-50

3 groups

Properties

Number: 7

Name: Special

Channels: 35-40, 42, 44, 46-50

The Groups Editor is divided into several sections. The top panel shows the list of Groups. Click on a group to have it appear in the lower panel. Once selected, a Group’s properties can be viewed or modified.

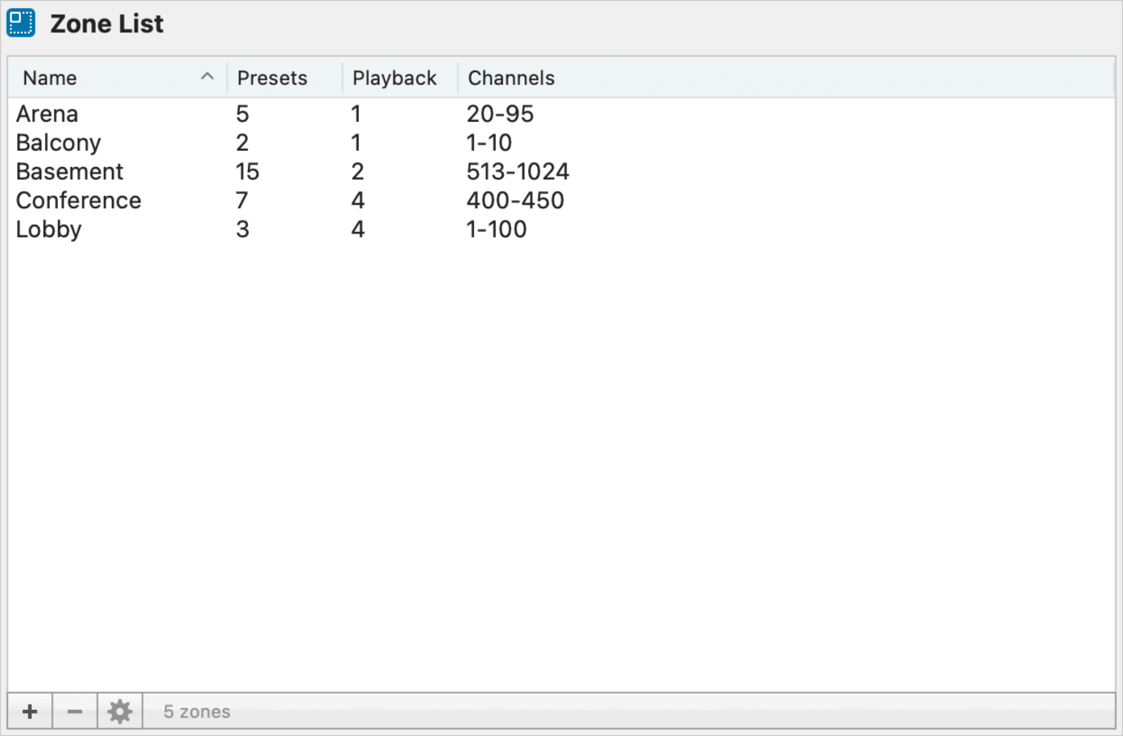
Zones

Overview

Zones are used to separate different areas of a particular project. Zones are also “parent containers” for presets, meaning that any preset must be a member of a zone.

The *Zones* editor is where zones are created and removed from the project.

For information on using zones, see the Zone command.



Zone List

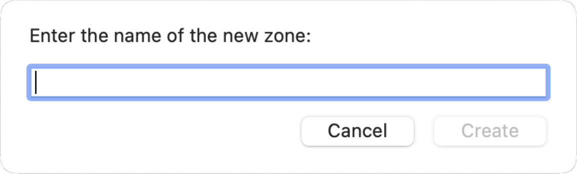
Name	Presets	Playback	Channels
Arena	5	1	20-95
Balcony	2	1	1-10
Basement	15	2	513-1024
Conference	7	4	400-450
Lobby	3	4	1-100

5 zones

Add a Zone

To add a zone, select the plus button (+).

A prompt will appear asking for a Zone name.



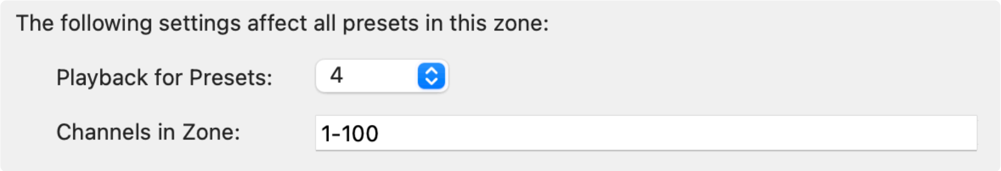
Enter the name of the new zone:

Enter the desired Zone name and then click **Create**.

Edit a Zone

To edit a zone, double-click on the zone you want to modify and it will bring you to the zone settings.

The zone settings define what channels are in a zone and which playback the zone's presets are executed in.



The following settings affect all presets in this zone:

Playback for Presets: 4

Channels in Zone: 1-100

Click on **Apply** in the bottom right of the window to save any changes.

Remove a Zone

To remove a zone, select the zone you want to remove and then select the minus button (−).

A popup will appear asking if you want to delete the zone. Click **Delete**.

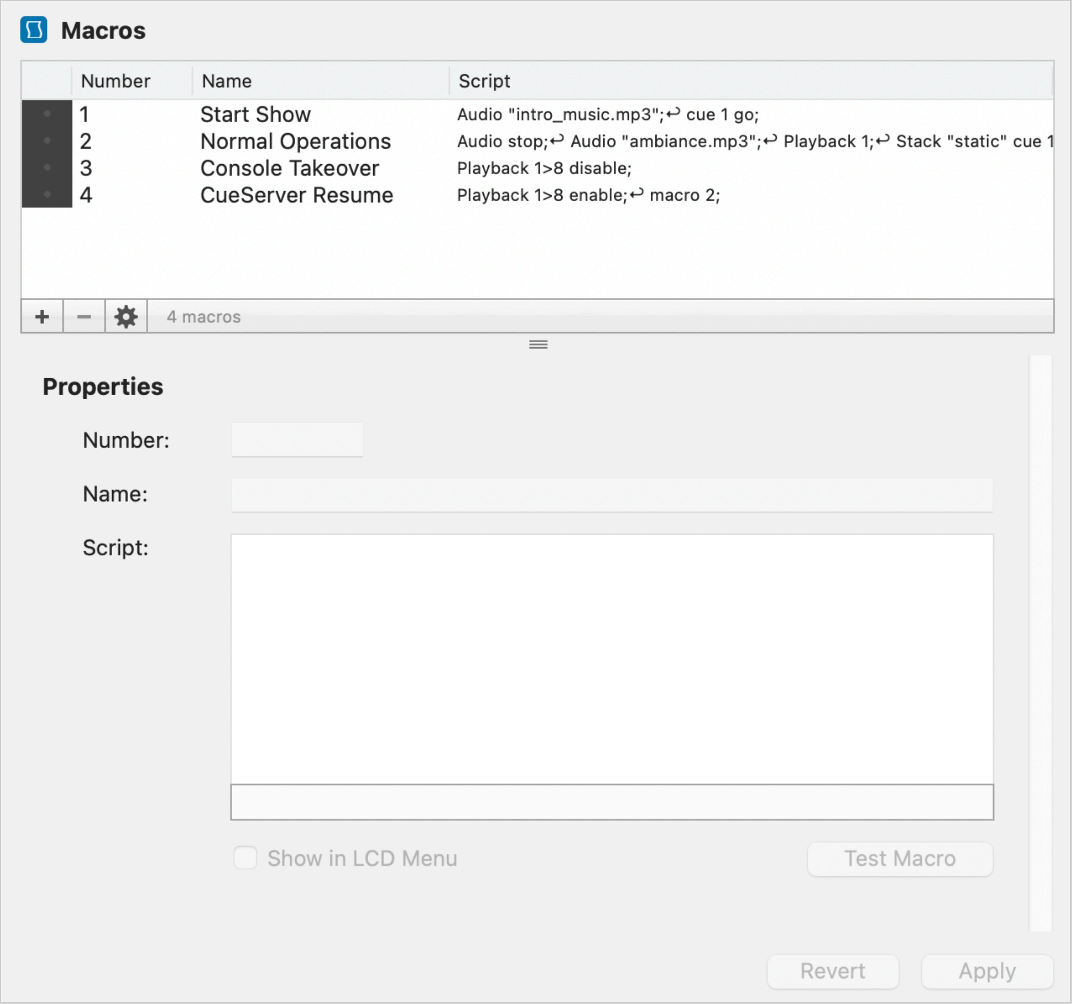
Macros

Overview

A macro is a container for a set of commands to perform a particular task. When the macro command is executed, all of the commands defined in the macro are executed in its place.

The *macros* editor is where macros are created, edited and removed from the project.

For information on using macros, see the Macro command.



Macros

Number	Name	Script
1	Start Show	Audio "intro_music.mp3";↩ cue 1 go;
2	Normal Operations	Audio stop;↩ Audio "ambiance.mp3";↩ Playback 1;↩ Stack "static" cue 1
3	Console Takeover	Playback 1>8 disable;
4	CueServer Resume	Playback 1>8 enable;↩ macro 2;

Properties

Number:

Name:

Script:

□ Show in LCD Menu

Add a Macro

To add a macro, select the plus button (+).

A new macro will appear in the list and the properties panel will become active.

Remove a Macro

To remove a macro, select the macro in the list, and then select the minus button (−), or use the delete/backspace key.

A popup will appear asking if you want to delete the macro. Click **Delete**.

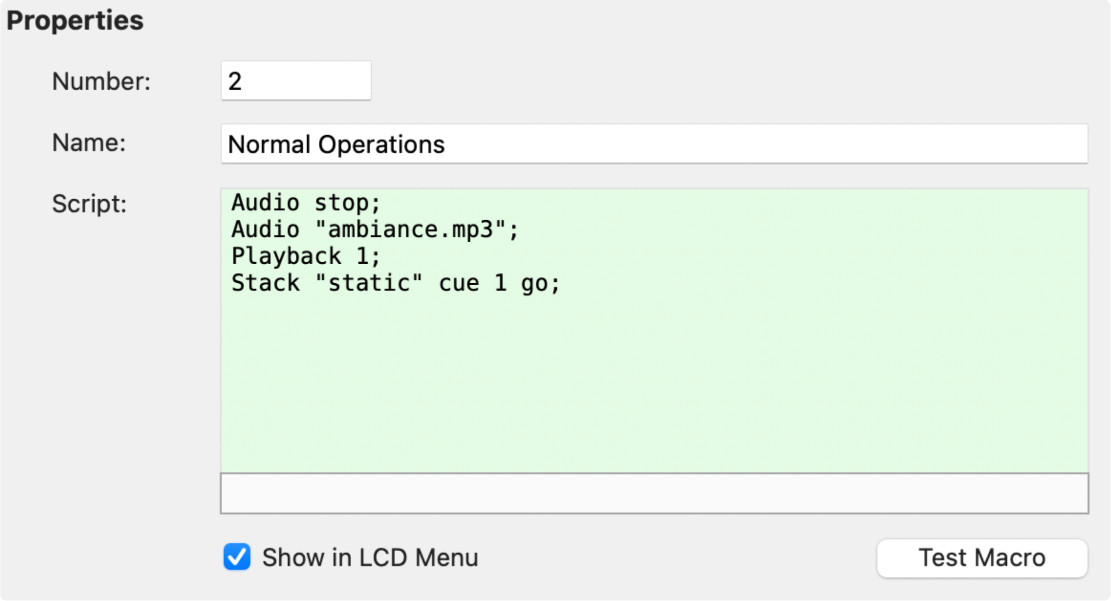
Properties Panel

After creating or selecting a macro, the properties panel will become active and you can edit the macro's properties.

The **Number** field defines the macro number. The macro number is used as the handle to reference a macro. Upon creation, the next available macro number will be assigned automatically. You can use the **Number** field to change it, if desired.

The **Name** field assigns the macro a name. The macro name is used to easily identify macros in the list and/or LCD menu. Upon creation, the default name "Untitled Macro" will be assigned. You can use the **Name** field to change it, if desired.

The **script** field contains the CueScript to be executed upon execution of the macro.



Properties

Number: 2

Name: Normal Operations

Script:

```
Audio stop;
Audio "ambiance.mp3";
Playback 1;
Stack "static" cue 1 go;
```

☑ Show in LCD Menu

If your CueServer model has an LCD display, you can optionally check the box next to **Show in LCD Menu** to make the macro appear as an executable option in the macros section of the LCD menu.

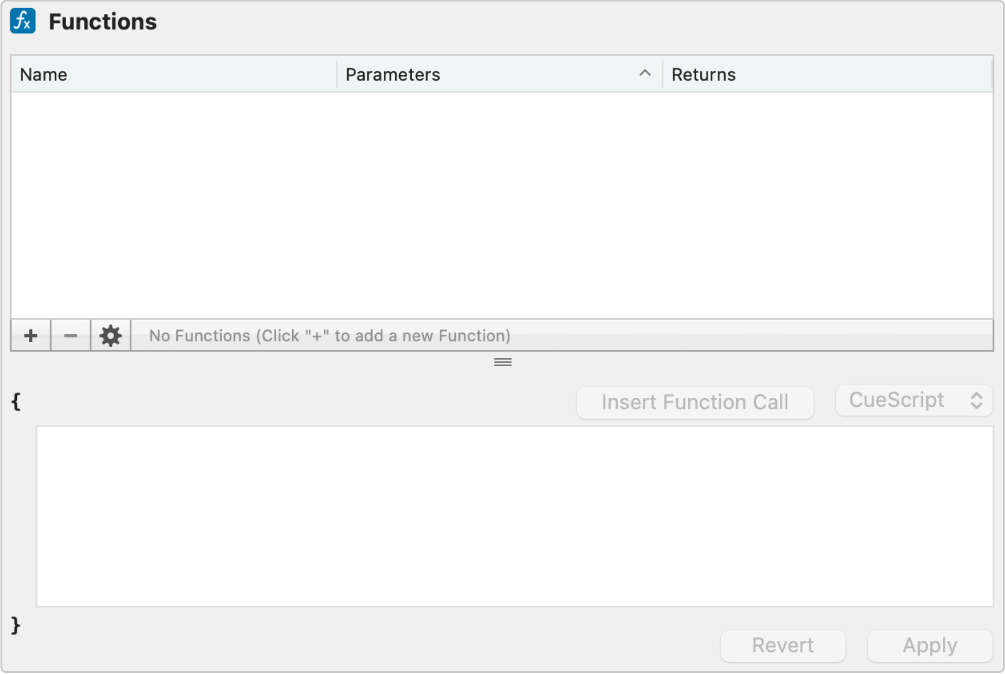
Use the **Test Macro** button (Test Macro) to test and/or execute the macro immediately.

Functions

Overview

Functions are user-definable code blocks that can be executed to accomplish a task, similar to macros. Unlike macros, Functions can have parameters, return values, and be written in JavaScript as well as CueScript. Functions can be called directly from CueScript, optionally with arguments, and can interact with JavaScript Plugins or Event Handlers.

The Function editor is where you add, modify, or adjust user-defined CueScript or JavaScript functions.

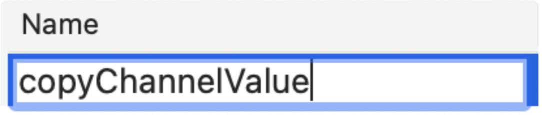


No Functions (Click "+" to add a new Function)

Add a Function

Click the plus button (+) to add a new function. A new row will appear with the **Name** field focused.

Enter the name that will be used to invoke the function.

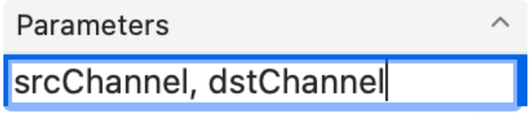


Name

copyChannelValue

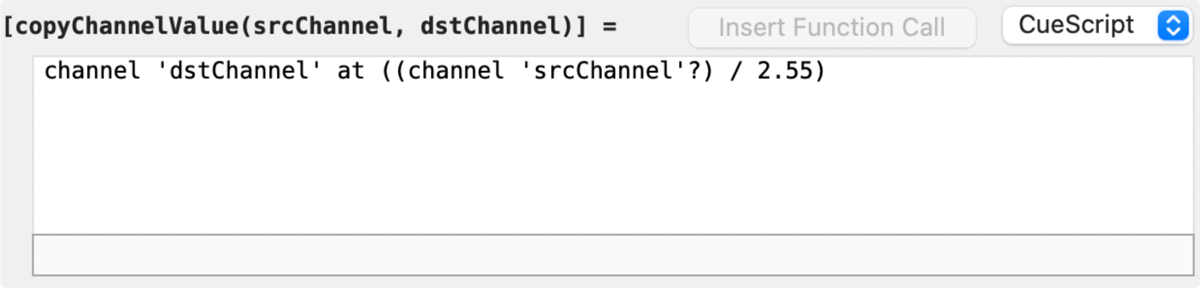
If the function will accept arguments, use the tab key to move the focus to the **Parameters** field.

Enter the name of each parameter, separated by commas.

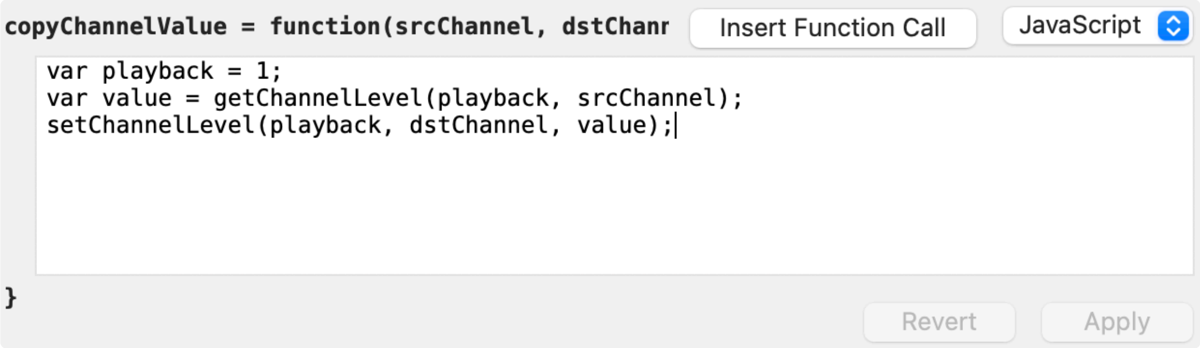


srcChannel, dstChannel

Enter the script that the function will execute in the script box at the bottom.



```
[copyChannelValue(srcChannel, dstChannel)] =
  channel 'dstChannel' at ((channel 'srcChannel'?) / 2.55)
```



```
copyChannelValue = function(srcChanr
  var playback = 1;
  var value = getChannelLevel(playback, srcChannel);
  setChannelLevel(playback, dstChannel, value);
}
```

Click **Apply** to save the function.

! Function and Parameter names must begin with a letter and contain no spaces or special characters.

Remove a Function

To remove a function, select the function in the list, and then select the minus button (−), or use the delete/backspace key.

A popup will appear asking if you want to delete the function. Click **Delete**.

Function Types

CueServer functions can be written using CueScript or JavaScript syntax.

Use the type menu (JavaScript) to define which syntax a function uses.

CueScript function

CueScript functions are similar to macros, but can utilize parameters and return values. CueScript functions have access to all of the normal CueScript syntax, can access user variables, and can query, capture, and use the results of CueScript commands.

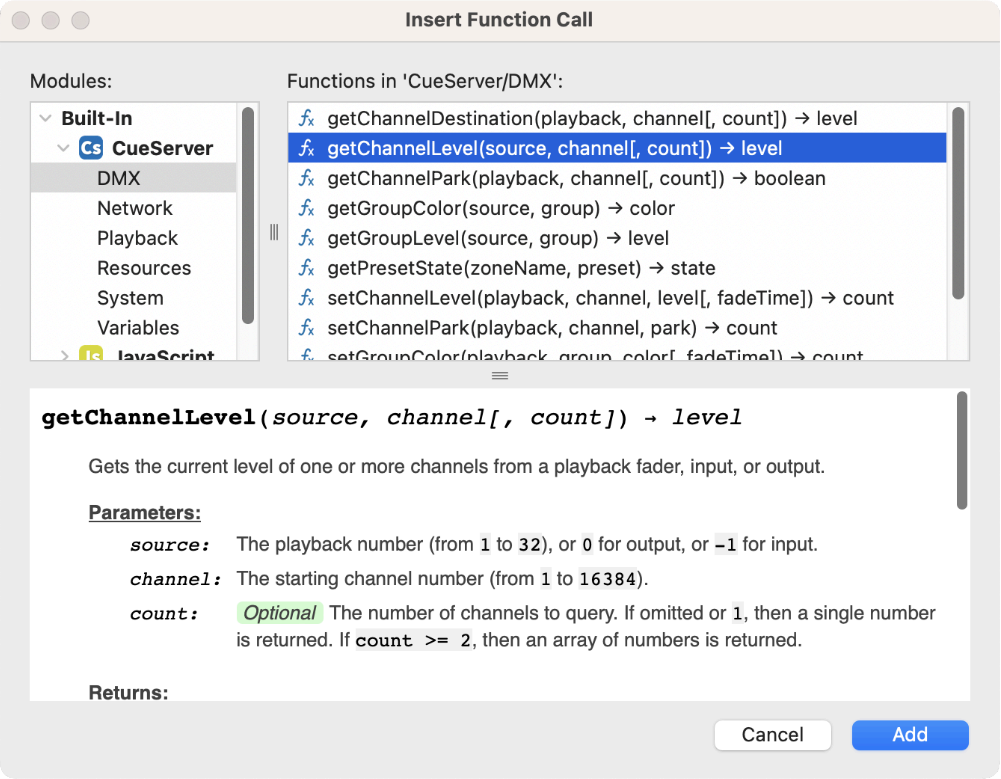
Arguments are accessible in the scope of the function and can be accessed in the same way as a global or system variable. If a parameter is used with the same name as a global variable, the provided argument value will take precedence within the scope of the function. For information on how to use CueScript variables, see the variables section.

CueScript functions can return a value with the Return command.

Javascript function

CueServer's embedded JavaScript engine supports ECMAScript E5 syntax. In addition to the most of the standard JavaScript API's, functions also have access to a host of CueServer-specific JavaScript API's.

To access the library of functions and their documentation, use the **Insert Function Call** button (Insert Function Call) to open the functions panel. Choosing a function and then clicking the **Add** button will insert a template of that function into the script.



Functions in 'CueServer/DMX':

getChannelDestination(playback, channel[, count]) → level
getChannelLevel(source, channel[, count]) → level
getChannelPark(playback, channel[, count]) → boolean
getGroupColor(source, group) → color
getGroupLevel(source, group) → level
getPresetState(zoneName, preset) → state
setChannelLevel(playback, channel, level[, fadeTime]) → count
setChannelPark(playback, channel, park) → count
setGroupColor(playback, group, color[, fadeTime]) → count

getChannelLevel(*source, channel[, count]*) → *level*

Gets the current level of one or more channels from a playback fader, input, or output.

Parameters:

source: The playback number (from `1` to `32`), or `0` for output, or `-1` for input.

channel: The starting channel number (from `1` to `16384`).

count: *Optional* The number of channels to query. If omitted or `1`, then a single number is returned. If `count >= 2`, then an array of numbers is returned.

Returns:

Arguments are accessible as normal variables in the local scope of the function.

A value can be returned with the JavaScript reserved word **return**.

Using Functions

To call a CueScript or JavaScript function, wrap the call in square brackets (`[]`), with the arguments in parentheses:

```
[ add(2, 3) ]
```

If a value is returned, it can be stored in a variable or used inline with commands:

> `"x" = [ add(2, 3) ]`
> Sets the variable *x* to the number `5`.

> `"x" = ([ add(2, 3) ] + 5)`
> Sets the variable *x* to the number `10`.

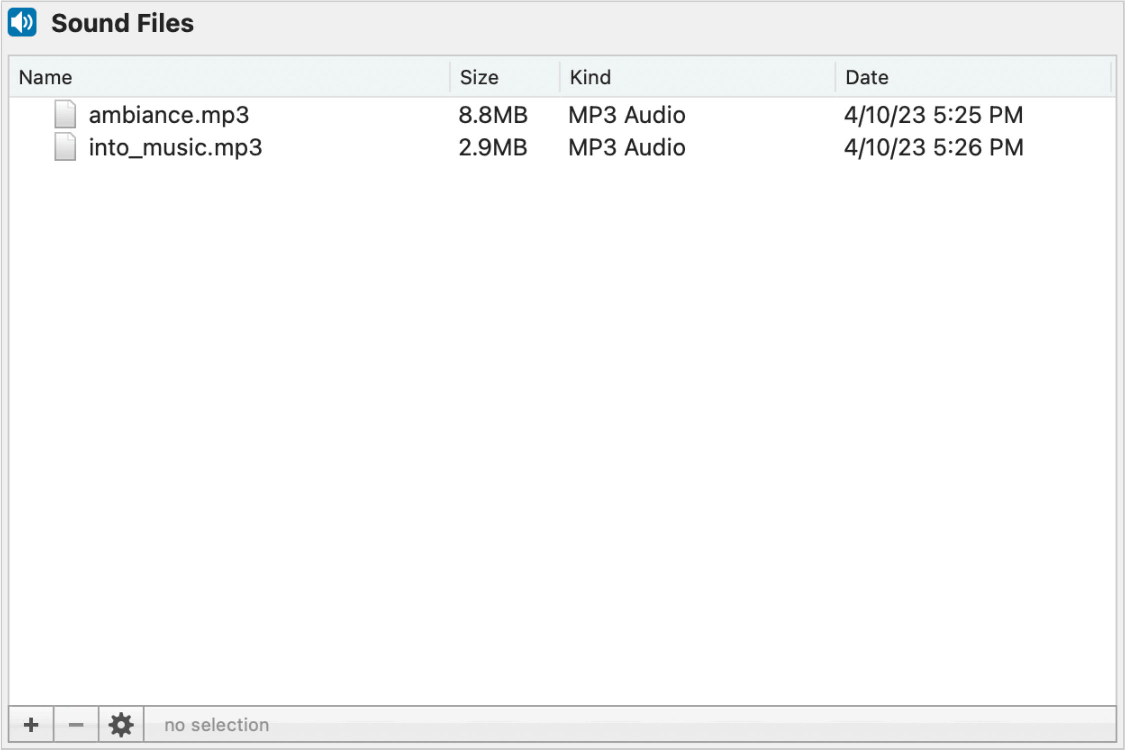
> `cue [ add(2, 3) ] go`
> Sets Cue `5` as the next Cue and executes it.

Sounds

Overview

The **Sounds** panel displays a list of sound files that have been added to the show.

For information on playing sound files, see the Audio command.



Sound Files

Name	Size	Kind	Date
ambiance.mp3	8.8MB	MP3 Audio	4/10/23 5:25 PM
into_music.mp3	2.9MB	MP3 Audio	4/10/23 5:26 PM

Add Sounds

To add a sound, click on the plus button (+) and then select the sound file in the file-picker window that appears.

You can also add sound files by dragging and then dropping them onto the list.

The supported audio formats include: .aif, .mp3, .ogg, .snd, and .wav

Remove Sounds

To remove a sound, select it from the list and then select the minus button (−), or use the delete/backspace key on your keyboard.

A popup will appear asking if you want to delete the file. Click **Delete**.

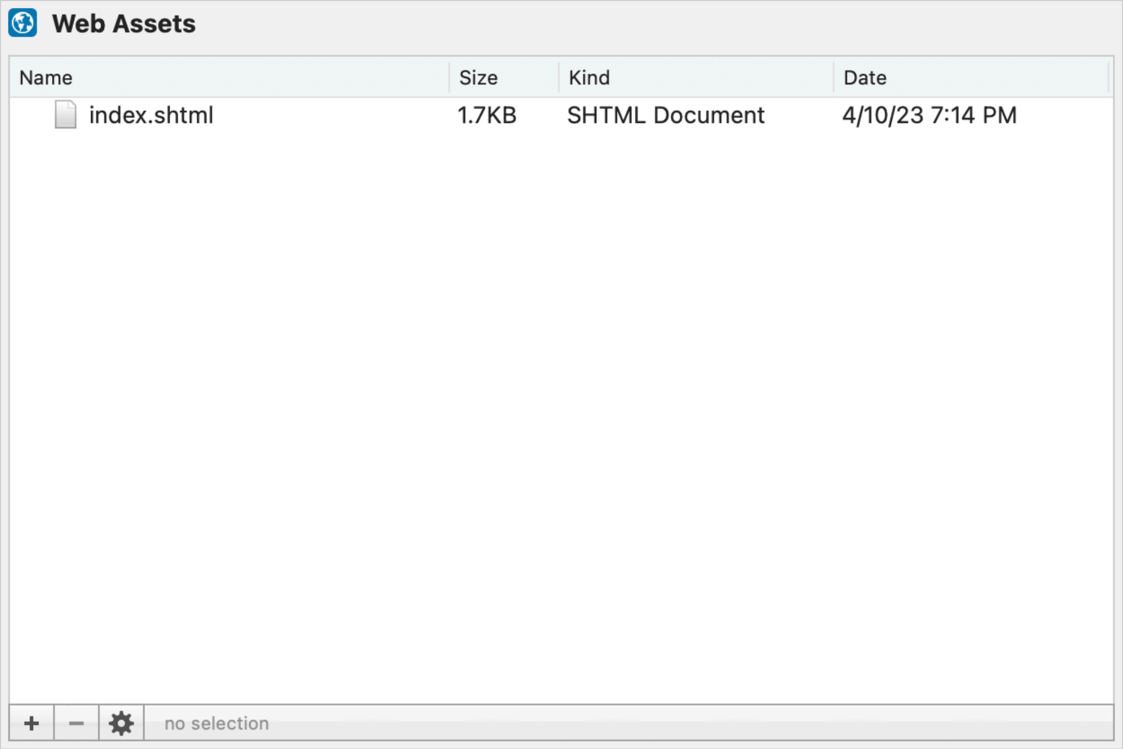
Web Pages

Overview

The **Web Pages** panel shows a list of files that have been added to the show.

You can access files stored within this panel from a web browser at the CueServer's root address.

For example, the default **index.shtml** file, which is included in every new show, can been seen by visiting the URL `http://<ip-of-CueServer>/`.



Name	Size	Kind	Date
index.shtml	1.7KB	SHTML Document	4/10/23 7:14 PM

If an **index.html** or **index.shtml** file are included in this list, they will be served when navigating to the CueServer's IP address in a browser. If the HTML file used is not named “index”, you will need to add the full file name to the end of the URL in order to access it.

For information about SSI files and variables, see the Environmental Variables section.

! CueServer uses the paths **layouts**, **playbacks**, **stage**, **station** and **weblib** for web-accessible views. Directories or files with these names located at the web root will not be reachable.

Add Files and Folders

To add a file, click on the plus button (+) and then select the file in the file-picker window that appears.

You can also add files or folders by dragging and then dropping them onto the list.

Remember to use directory and file names that can be used in a URL.

Remove Files

To remove a file or folder, select it in the list and then select the minus button (−), or use the delete/backspace key on your keyboard.

A popup will appear asking if you want to delete the file or folder. Click **Delete**.

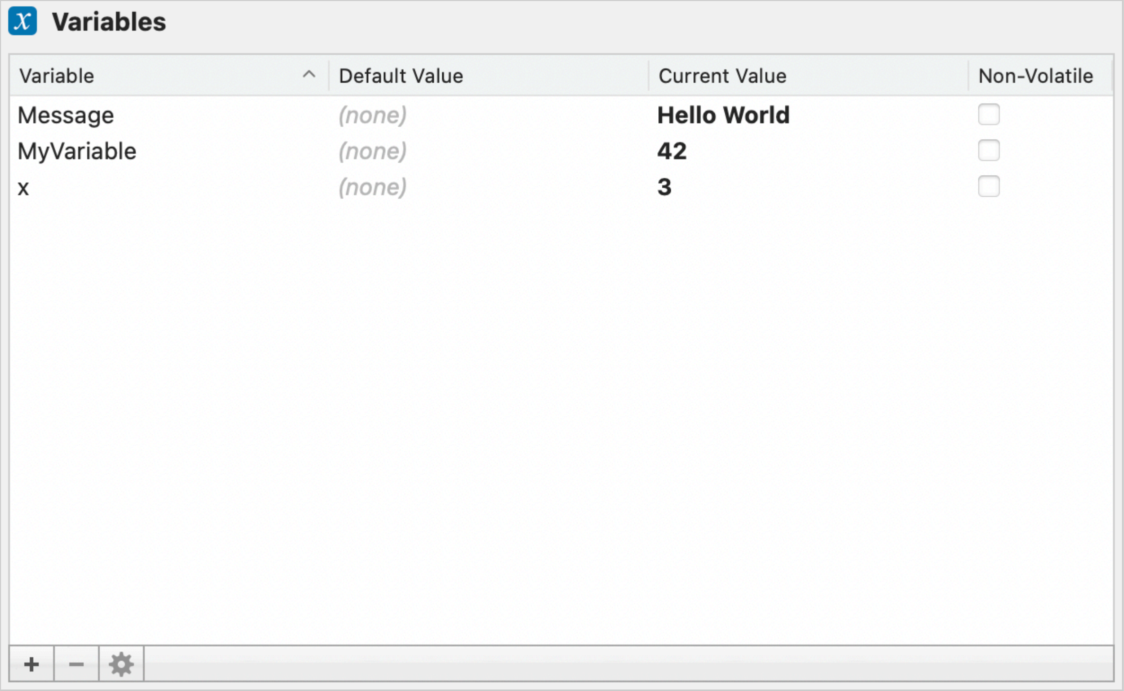
Variables

Overview

A variable is a symbol that holds and represents a value. Variables can be used to hold arbitrary values, states, or even CueScript commands.

The *Variables* editor can be used to view, create, edit, and remove variables from the project.

For information on using variables, see the Variables section of CueScript Expressions.



Variable	Default Value	Current Value	Non-Volatile
Message	*(none)*	**Hello World**	□
MyVariable	*(none)*	**42**	□
x	*(none)*	**3**	□

By default, variables are temporary and are cleared when the system is rebooted, loses power, or a new show is loaded. There are two options that allow you to adjust this behavior in cases where persistence is important:

Default Value

If a default value is provided, the variable will be saved to the show file and re-initialized, with the default value, any time the show is loaded.

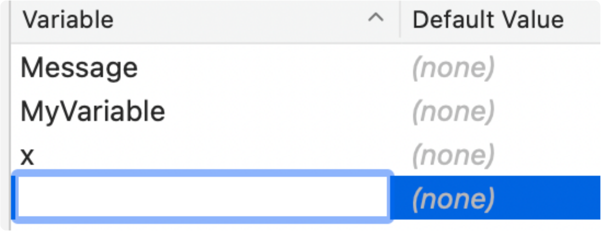
Non-Volatile

If a variable is marked Non-Volatile, the variable is stored in non-volatile RAM and it’s last-known value will persist through a reboot, power-loss, or show change.

Add a variable

To add a variable, click on the plus button (+).

A new row will appear with the *Variable* field focused.



Variable	Default Value
Message	(none)
MyVariable	(none)
x	(none)
	(none)

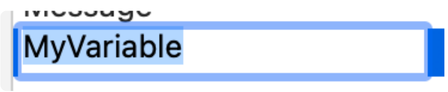
Type in the new variable name, then click outside the row to save it.

Alternatively, you can use the tab key after keying in a variable name to focus the next field.

Update a variable

To update a variable's name, or its current or default value, double-click on the field you wish to update.

The field will then become focused and you can modify or clear its contents.



MyVariable

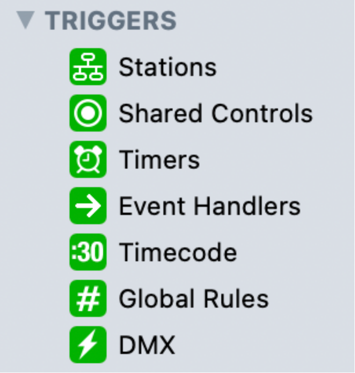
When you are done making changes, click outside of the field, or use the enter key to save your changes. Alternatively, you can use the tab key to focus the next field.

Remove a variable

To remove a variable, select the variable in the list and then select the minus button (−).

Triggers

The *Triggers* section of the navigator contains views that deal with definitions for incoming system events.



The following sections describe these views in more detail:

- Stations – setup for stations, buttons, contact-closures and more.
- Shared Controls – a global set of buttons.
- Timers – setup for timers.
- Event Handlers – setup for plugin events.
- Timecode – SMPTE timecode events.
- Global Rules – a global list of rules.
- DMX – trigger actions or events based on incoming DMX values.

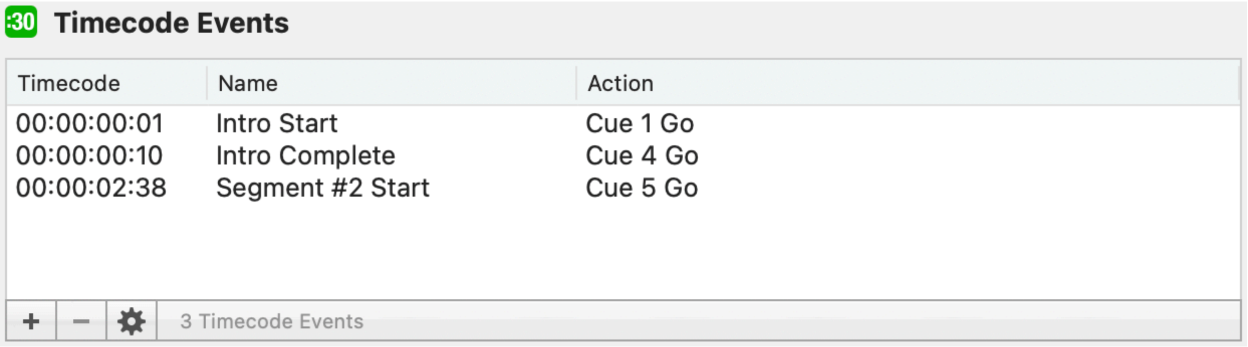
Timecode

CueServer offers the ability to trigger actions based on internal or external SMPTE Timecode.

Timecode events are configured using the **Timecode** section of the **TRIGGERS** group within CueServer Studio.

:30 Timecode

When chosen, the top of the editor panel will show the Timecode events listing:



:30 Timecode Events

Timecode	Name	Action
00:00:00:01	Intro Start	Cue 1 Go
00:00:00:10	Intro Complete	Cue 4 Go
00:00:02:38	Segment #2 Start	Cue 5 Go

Each column of the list is described below:

- **Timecode** – The timecode of the event.
- **Name** – The name given to each event.
- **Action** – The action that fires at that timecode.

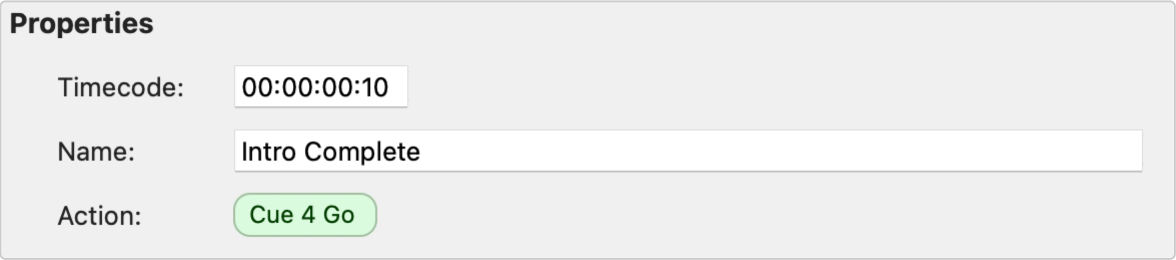
The + and − buttons at the bottom of the list will add a new event, or remove a selected event.

The ⚙ button displays a menu of options, including:

- **Duplicate Timecode Event…** – Used to make a copy of an existing timecode event.
- **Refresh** – Used to reload the list of timecode events.

Properties

When creating or editing a selected timecode event, three fields will appear in the properties pane below the list.



Properties

Timecode: 00:00:00:10

Name: Intro Complete

Action: Cue 4 Go

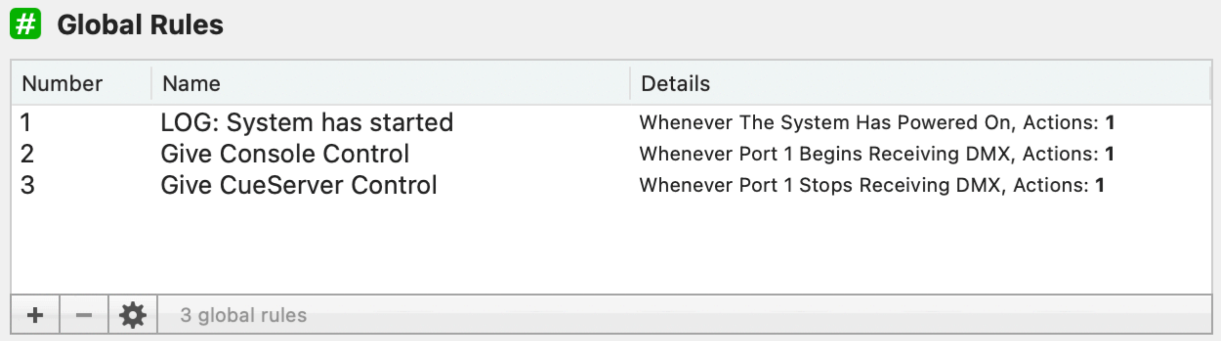
Click on the action bubble (None) to add the CueScript to be executed at the selected timecode.

Global Rules

CueServer offers the ability to trigger actions based on Global events, such as a button press, system startup, or a DMX signal being detected or lost.

Global Rules are configured using the **Global Rules** section of the **TRIGGERS** group within CueServer Studio.

When chosen, the top of the editor panel will show the Global Rules listing:



Global Rules

Number	Name	Details
1	LOG: System has started	Whenever The System Has Powered On, Actions: **1**
2	Give Console Control	Whenever Port 1 Begins Receiving DMX, Actions: **1**
3	Give CueServer Control	Whenever Port 1 Stops Receiving DMX, Actions: **1**

Each column of the list is described below:

- **Number** – The numerical order of the list.
- **Name** – The name given to each Global Rule.
- **Details** – A summary of the defined trigger and action.

The + and − buttons at the bottom of the list will add a new rule, or remove a selected rule.

The ⚙ button displays a menu of options, including:

- **Duplicate Rule…** – Used to make a copy of an existing rule.
- **Refresh** – Used to reload the list of Global Rules.

Properties

When creating or editing a selected Global Rule, three fields will appear in the properties pane below the list:

- **Number** – The rule number is used to order the rule in the list above. If creating a rule, this field will automatically be populated with the next sequential number available.
- **Name** – The name property is to help identify each rule's purpose at a glance.

- **Rule** – The rule describes the event that acts as the rule’s trigger, and the action(s) that should take place.

Building a Rule

A rule is structured to describe a sequence:

WHENEVER *something happens* → **AND** *a condition is met* → **THEN** *do something*

In the case of global rules, there are 4 general components:

- **Entity** – the thing that something happens to
- **Event / Trigger** – the something that happens
- **Condition** – optional condition(s) that must be met
- **Action** – the something to be done

Entity

Click on the Choose... button to select an entity. The available options are:

- The System
- The Show
- Button
- Contact
- Control
- DMX Universe
- DMX Port

In some cases, such as with Buttons, there may be multiple instances. In these cases, a field will appear to define the specific instance.

Event

Choose the event / trigger for the rule. The available events depend on the chosen entity.

Entity	Events
The System	Has Powered On Is Powering Off
The Show	Has Loaded Will Be Unloaded

Button (*n*)	Is Pressed Is Held Is Released
Contact (*n*)	Is Opened Is Closed Is Held
Control (*n*)	Is Pressed Is Held Is Released
DMX Universe (*n*)	Begins Receiving Data Stops Receiving Data
DMX Port (*n*)	Begins Receiving Data Stops Receiving Data

Condition(s)

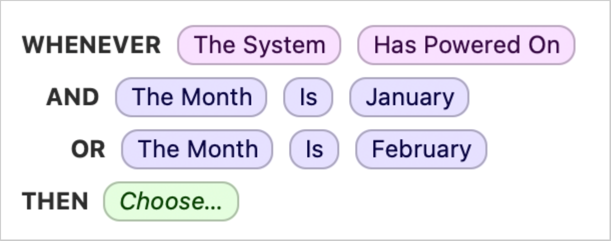
One ore more conditions can be added to an event. Hover over the end of a line the in the rule list to show the add AND button (+AND).

The following options are available:

- The Time
- The Hour
- The Minute
- The Second
- The Date
- The Month
- The Day
- The Year
- The Sun Brightness
- Button
- Contact
- Indicator
- Output
- Variable

Each of the above options contain various configurations and/or fields to describe the condition.

Additionally, conditions can have **OR** statements. For example, **AND** The Month Is January **OR** The Month Is February. To add an **OR** statement to a condition, hover over the end of the condition and press the add OR button (+OR).



WHENEVER The System Has Powered On
AND The Month Is January
OR The Month Is February
THEN Choose...

✻ In cases where the condition options above do not cover a project's needs, CueScript can be used to define custom logic using If..Then statements.

Action(s)

Use the Choose... button to define an Action.

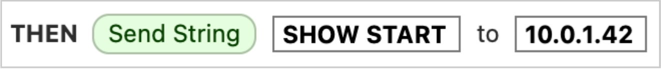
The following actions are available:

- Perform CueScript
- Perform Macro
- Execute Cue
- Next Cue
- Update Cue
- Select Playback
- Clear Playback
- Set Submaster
- Set Channel
- Set Group
- Release Channel
- Release Group
- Activate Preset
- Toggle Preset
- Deactivate Preset
- Update Preset
- Set Indicator
- Set Output
- Set Variable
- Send String

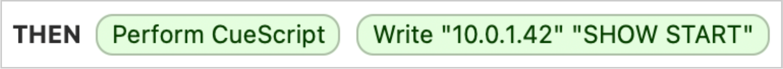
Rules can also have more than one action and are executed in the order that they are listed. Hover over the end of an action in the rule list to show the add THEN button (+THEN).

In some cases you may need to change a rule to CueScript in order to accomplish a more-complex action, or add special cases.

You can convert an action into CueScript automatically by hovering over the end of the action, selecting the menu button (▼), and then selecting **Convert to CueScript** from the menu.



THEN Send String SHOW START to 10.0.1.42



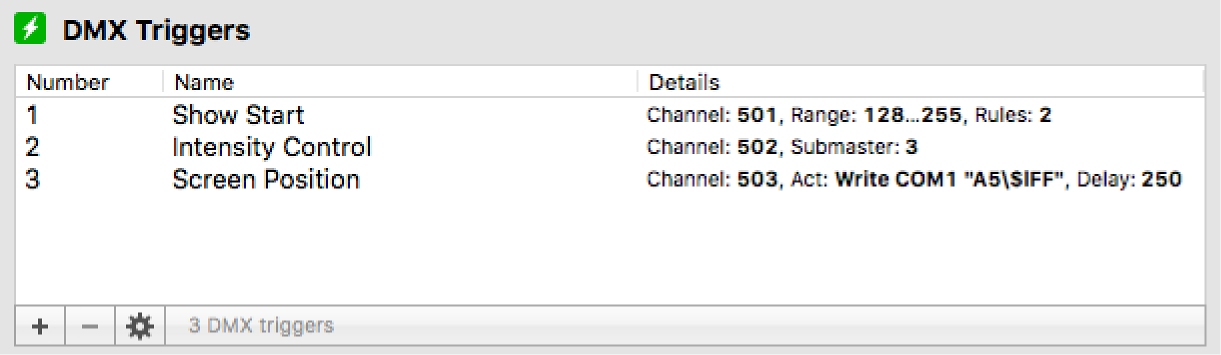
THEN Perform CueScript Write "10.0.1.42" "SHOW START"

DMX

CueServer offers the ability to trigger actions or events based on the live incoming value of DMX channels.

DMX Triggers are configured using the **DMX** section of the **TRIGGERS** group within CueServer Studio.

When chosen, the top of the editor panel will show the DMX Triggers listing:



DMX Triggers

Number	Name	Details
1	Show Start	Channel: **501**, Range: **128...255**, Rules: **2**
2	Intensity Control	Channel: **502**, Submaster: **3**
3	Screen Position	Channel: **503**, Act: **Write COM1 "A5\$IFF"**, Delay: **250**

Each column of the list is described below:

- **Number** – The numerical order of the list.
- **Name** – The name given to each DMX Trigger.
- **Details** – A summary of the properties of each DMX Trigger.

The + and − buttons at the bottom of the list will add a new trigger, or remove a selected trigger.

The ⚙ button displays a menu of options, including:

- **Duplicate DMX Trigger…** – Used to make a copy of an existing trigger.
- **Refresh** – Used to reload the list of DMX triggers.

DMX Trigger Types

When creating or editing a DMX Trigger, a type menu appears that chooses what type of behavior the trigger will have:

Type: Enter/Exit Range

There are three types (or modes) of DMX Triggers. Each type of trigger requires different properties and exhibits a different behavior. The available trigger types include:

- **Enter/Exit Range** – A trigger based on a DMX channel either entering or exiting a specific range of values.
- **Submaster Control** – A trigger that automatically links the given DMX channel value to a Playback fader’s submaster.
- **Act on Change** – A trigger that performs a CueScript action whenever a DMX channel value changes.

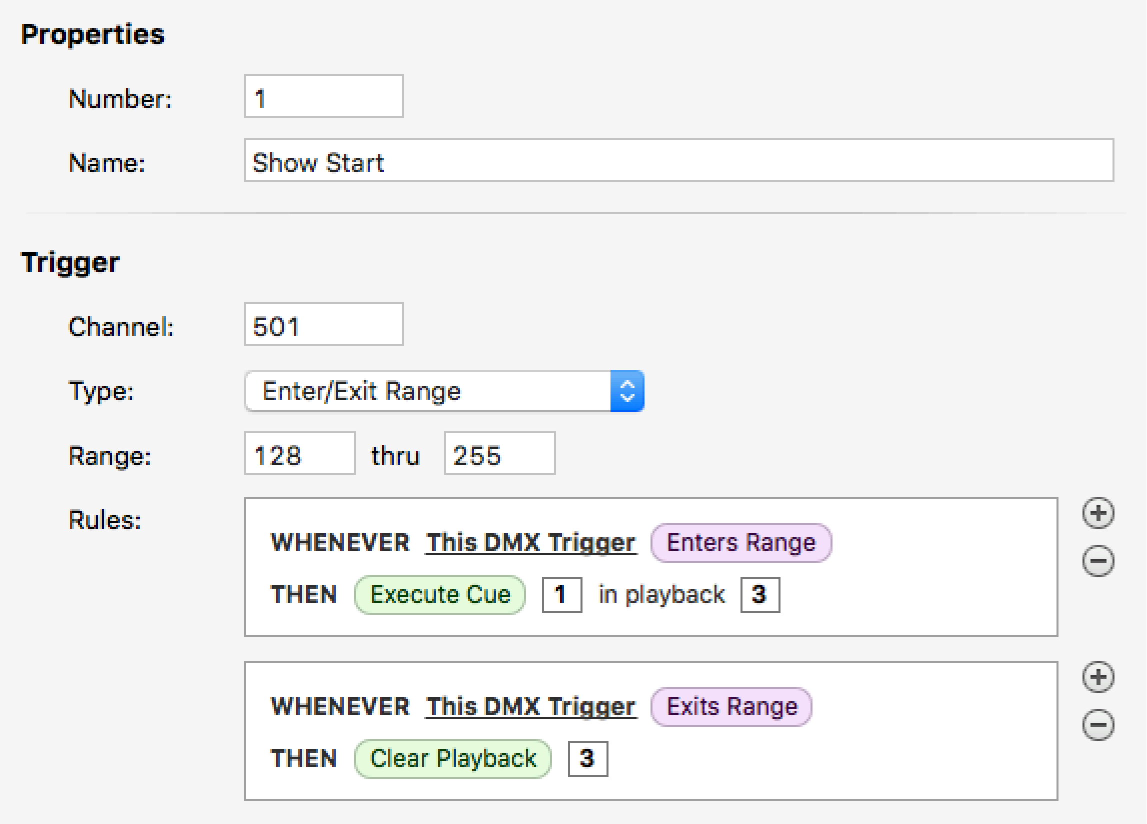
The following sections describe each of these DMX Trigger types.

Enter/Exit Range Trigger

When a DMX Trigger is set to the **Enter/Exit Range** type, various rules can be added to the trigger that fire whenever the channel value either enters or exits a particular range of values.

This type of trigger is best used to activate certain events within CueServer based on an input channel being raised or lowered into or out of a range of values.

When editing an **Enter/Exit Range** trigger, the editor panel will appear similar to this example:



Properties

Number: 1

Name: Show Start

Trigger

Channel: 501

Type: Enter/Exit Range

Range: 128 thru 255

Rules:

WHENEVER This DMX Trigger Enters Range
THEN Execute Cue 1 in playback 3

WHENEVER This DMX Trigger Exits Range
THEN Clear Playback 3

The following properties can be set for an **Exter/Exit Range** trigger:

- **Properties**
  - **Number** – The numerical order of the trigger in the list. This number can be changed to replace an existing trigger or to organize triggers numerically.
  - **Name** – A name for the trigger. This field can be freely set to any text.
- **Trigger**
  - **Channel** – The channel that is being observed for the trigger. A value from `1` to `16384` may be entered into this field. The channel number corresponds to the *global* channel number within CueServer, not a local channel number in a single universe.
  - **Type** – The type of the DMX Trigger. Set to **Enter/Exit Range** for this type of trigger.
  - **Range** – A range of *decimal* DMX values for this trigger. Each field may be from `0` to `255`.
  - **Rules** – One or more rules configured to trigger whenever the input DMX channel either enters

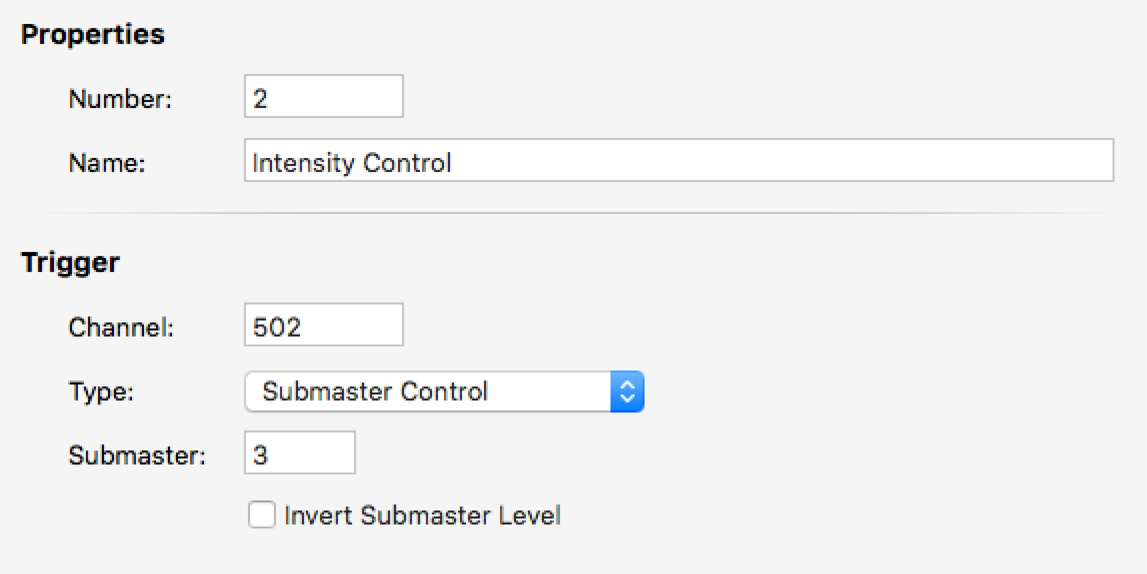
or exits that range. Each rule may contain any actions or conditions as permitted by rules.

Submaster Control Trigger

When a DMX Trigger is set to the **Submaster Control** type, the incoming DMX channel value is directly mapped to control the Submaster of a Playback Fader.

This type of trigger is best used when it is desired to have an external console directly control the Submaster level of one of CueServer's Playback Faders.

When editing an **Submaster Control** trigger, the editor panel will appear similar to this example:



Properties

Number: 2

Name: Intensity Control

Trigger

Channel: 502

Type: Submaster Control

Submaster: 3

□ Invert Submaster Level

The following properties can be set for an **Submaster Control** trigger:

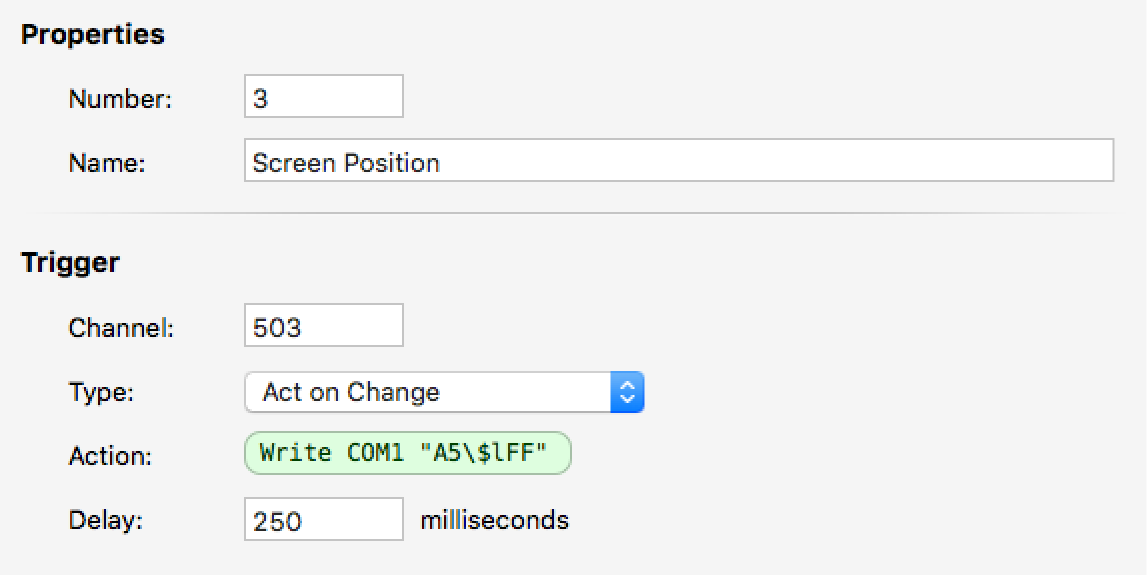
- **Properties**
  - **Number** – The numerical order of the trigger in the list. This number can be changed to replace an existing trigger or to organize triggers numerically.
  - **Name** – A name for the trigger. This field can be freely set to any text.
- **Trigger**
  - **Channel** – The channel that is being observed for the trigger. A value from `1` to `16384` may be entered into this field. The channel number corresponds to the *global* channel number within CueServer, not a local channel number in a single universe.
  - **Type** – The type of the DMX Trigger. Set to **Submaster Control** for this type of trigger.
  - **Submaster** – The number of the Playback Fader's Submaster to control. A value from `1` to `32` may be in this field.
  - **Invert Submaster Level** – Normally, the DMX value is passed directly to the Submaster value. When this checkbox is selected, the Submaster value will be inverted from the DMX value. For example, when the DMX value is at zero, the Submaster will be at full.

Act on Changes Trigger

When a DMX Trigger is set to the **Act on Changes** type, any time the incoming DMX value changes, a CueScript action is executed.

This type of trigger is best used to create custom actions that take the input value of a DMX channel and perform some kind of operation upon it. For example, a trigger of this type can output a string via a serial port or by UDP messages that passes the DMX channel value to another device. Another example would be to control a group of channels from the input of a single channel.

When editing an **Act on Changes** trigger, the editor panel will appear similar to this example:



Properties

Number: 3

Name: Screen Position

Trigger

Channel: 503

Type: Act on Change

Action: `Write COM1 "A5\$lFF"`

Delay: 250 milliseconds

The following properties can be set for an **Act on Changes** trigger:

- **Properties**
  - **Number** – The numerical order of the trigger in the list. This number can be changed to replace an existing trigger or to organize triggers numerically.
  - **Name** – A name for the trigger. This field can be freely set to any text.
- **Trigger**
  - **Channel** – The channel that is being observed for the trigger. A value from `1` to `16384` may be entered into this field. The channel number corresponds to the *global* channel number within CueServer, not a local channel number in a single universe.
  - **Type** – The type of the DMX Trigger. Set to **Act on Changes** for this type of trigger.
  - **Action** – A CueScript action that is executed each time the input DMX value changes. See below for examples.
  - **Delay** – Each time this trigger is activated by a change in DMX value, this delay temporarily *disarms* the trigger from firing again. After the delay expires, the trigger is re-armed and will fire again if the channel value has changed since the previous triggering. This is particularly useful when using this trigger type to send serial strings to external devices that cannot handle

updates as fast as DMX in coming into the CueServer (approximately 40Hz).

Note: String Substitution Codes can be used to retrieve the DMX value and/or other live information into strings when using this type of DMX Trigger. See the section on Strings for more information.

Examples

Controlling Groups

To create a scenario where a group of channels in CueServer is controlled by an input DMX channel, string substitution can be used. First assign a variable to the DMX value, and then use that variable to adjust the level of the group. The CueScript action would be:

```
"level" = "#\#l"; Group 1 at `level`
```

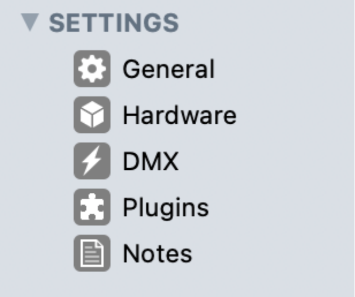
Controlling Window Blinds

Popular window blind systems use a RS-232 serial protocol to communicate with the motors. To create a DMX trigger that takes the value of a DMX channel and passes it to the window blind controller, a string such as `"FCF0FF<group-address><motor-id>FB<position>FFFF<checksum>"` must be sent out the serial port. To do this, the CueScript action would be:

```
Write COM1 "FCF0FF808080ABCDEFFB\$lFFFF\$S"
```

Settings

The *Settings* section of the editor window contains views that configure how the show operates.

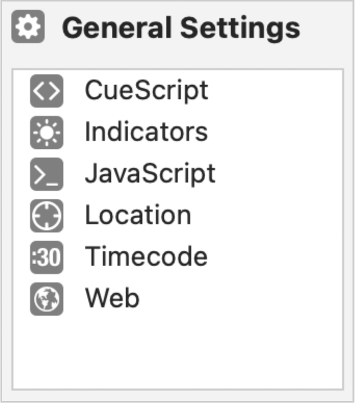


The following sections describe these views in more detail:

- General – general purpose settings.
    - CueScript – configure CueScript console server and inbound UDP execution.
    - Indicators – configure default indicator colors and flash patterns.
    - JavaScript – configure JavaScript console server.
    - Location – set geographical location for astronomical functionality.
    - Timecode – configure SMPTE timecode behavior.
    - Web – configure Apache CORS headers.
- Hardware – model, audio, LCD display and DMX port config.
    - Audio – audio output volume.
    - LCD Display – configure LCD content and brightness.
    - DMX Ports / Modules – configure hardwire DMX ports and modules.
- DMX – DMX and fixture related settings.
    - Resources – control the number of available channels and playbacks.
    - Universe Patch – control DMX universes and DMX-over-Ethernet.
    - Fixture Patch – patch fixtures to DMX addresses.
    - Playback [n] – configure playback options.
- Plugins – Javascript plugins.
- Notes – optional details about the show file.

General

The *General Settings* page provides several views that control how CueServer is configured.



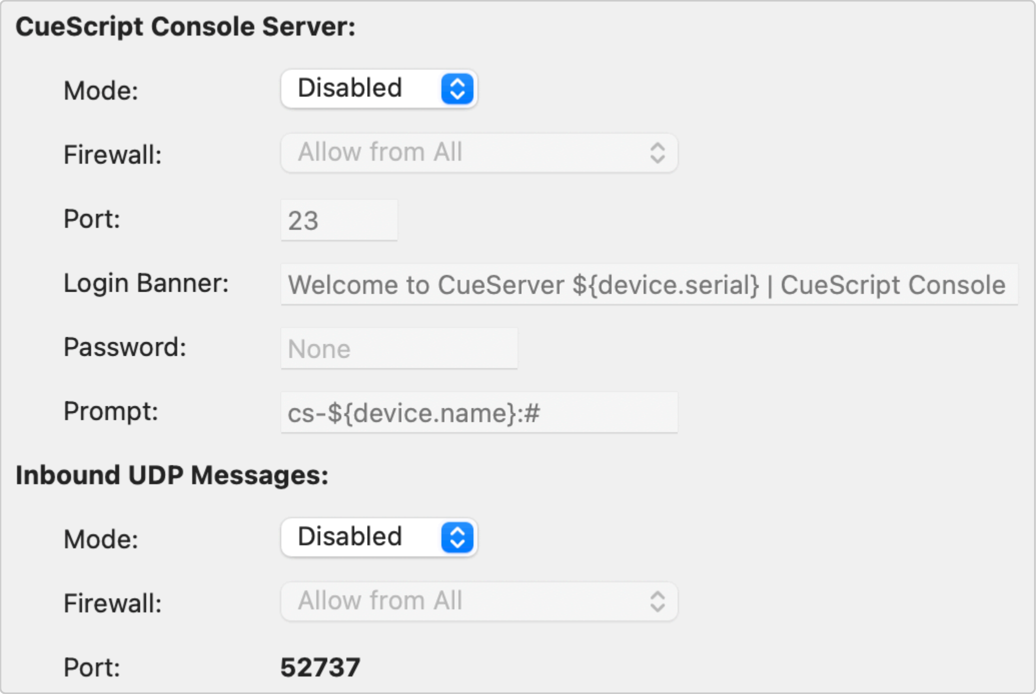
The following General Settings views are available:

- CueScript – configure CueScript console server and inbound UDP execution
- Indicators – configure default indicator colors and flash patterns
- JavaScript – configure JavaScript console server
- Location – set geographical location for astronomical functionality
- Timecode – configure SMPTE timecode behavior
- Web – configure Apache CORS headers

CueScript

Overview

The CueScript settings panel allows you to configure the CueScript console server and Inbound UDP message execution rules.



CueScript Console Server:

Mode: Disabled

Firewall: Allow from All

Port: 23

Login Banner: Welcome to CueServer ${device.serial} | CueScript Console

Password: None

Prompt: cs-${device.name}:#

Inbound UDP Messages:

Mode: Disabled

Firewall: Allow from All

Port: **52737**

CueScript Console Server

The CueScript console server is a network terminal that accepts and executes CueScript.

The following options are available:

- **Mode** – disables or sets the type of server used, Telnet or Raw TCP (*disabled by default*).
- **Firewall** – determines the network addresses that are allowed to connect to the server.
- **Port** – defines the port on which the console will accept connections.
- **Login Banner** – the “welcome message” sent to connecting devices.
- **Password** – an optional password that must be provided to use the terminal.
- **Prompt** – the line prefix that prompts users to input CueScript.

Inbound UDP Messages

This section controls whether the CueServer should execute CueScript commands sent via UDP to port 52737.

The following options are available:

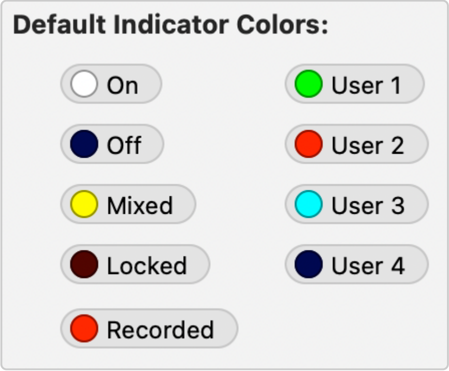
- **Mode** – enables or disables UDP command execution (*disabled by default*).
- **Firewall** – determines the network addresses that CueServer should accept UDP commands from.

> **!** Prior to v3.0.0 of CueServer firmware, the option to accept CueScript commands was not available. All CueServers would accept commands over port 52737. Starting with v3.0.0 new shows have this option disabled by default, however, shows made before v3.0.0 that are later updated to v3.0.0+ will maintain the ability to accept commands for compatibility.

Indicators

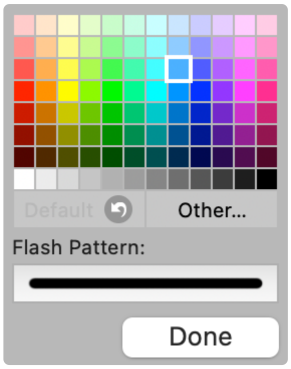
Overview

The Indicator settings panel allows you to configure the default, or top-level, indictor settings for **Buttons** and **Shared Controls**.



Each indicator state is listed here with a colored circle displaying the state’s current configuration. Indicator states can have both a color and a flash pattern.

Select an indicator state to open the indicator palette and change its settings.



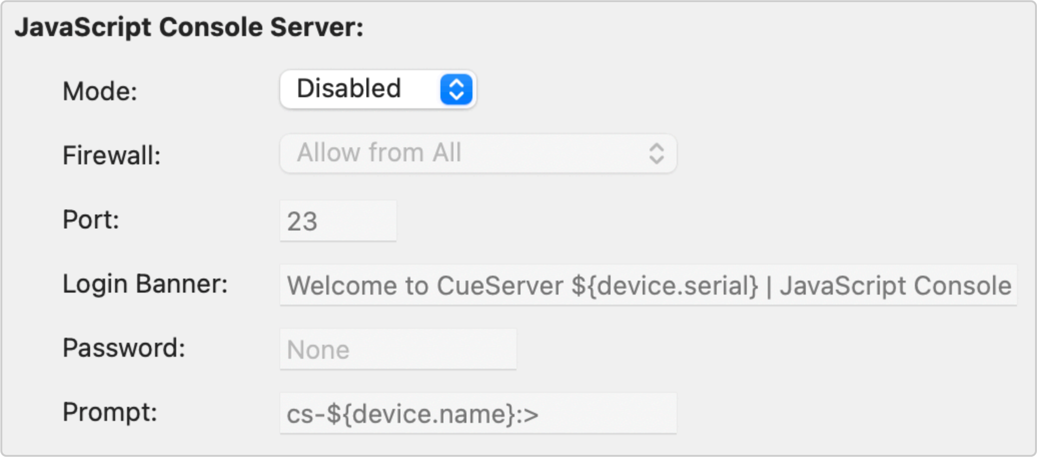
Select a color in the color palette to set a new indicator color, or select the flash pattern box to choose from a list of patterns.

The settings here are the global defaults, however, indicator settings can also be adjusted at the Station or Button level. The setting closest to the Button or Shared Control will apply. For example, if the On state is set to green in this settings panel but set to pink in a button or that button’s station settings, pink will be used.

JavaScript

Overview

The JavaScript settings panel allows you to configure the JavaScript console server settings.



JavaScript Console Server:

Mode: Disabled

Firewall: Allow from All

Port: 23

Login Banner: Welcome to CueServer ${device.serial} | JavaScript Console

Password: None

Prompt: cs-${device.name}:>

The JavaScript console server is a network terminal that accepts and executes JavaScript on the CueServer.

The following options are available:

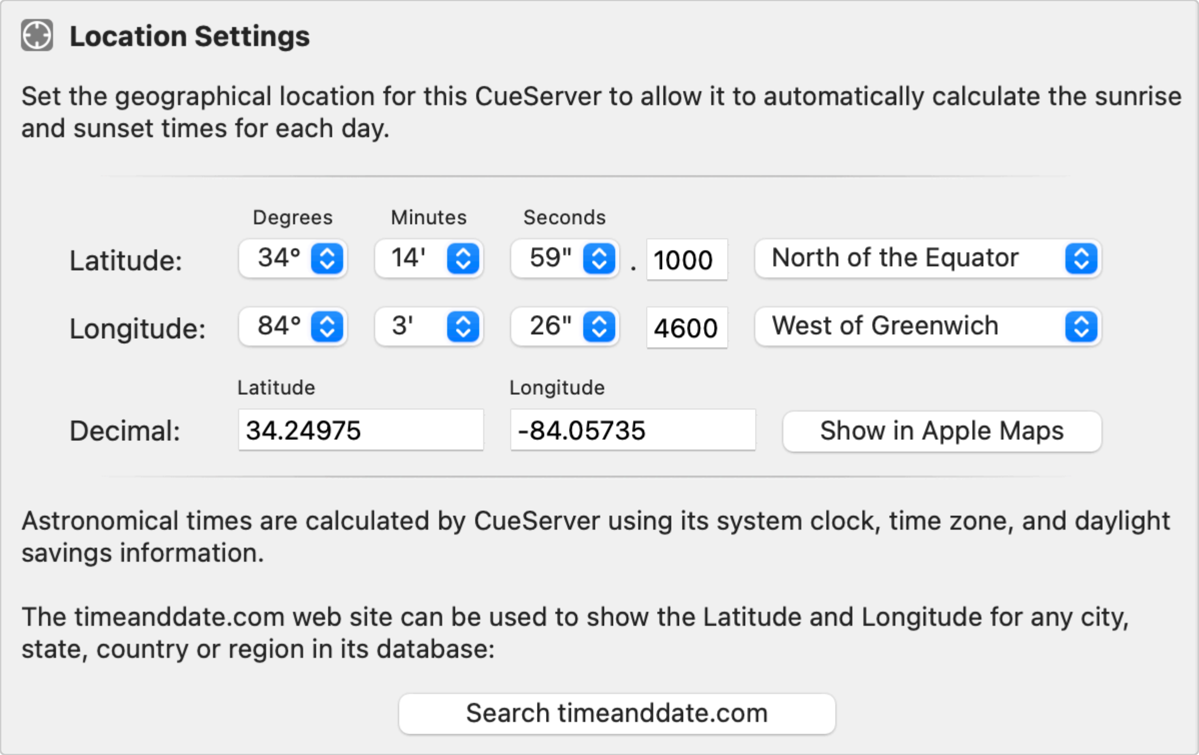
- **Mode** – disables or sets the type of server used, Telnet or Raw TCP (*disabled by default*).
- **Firewall** – determines the network addresses that are allowed to connect to the server.
- **Port** – defines the port on which the console will accept connections.
- **Login Banner** – the “welcome message” sent to connecting devices.
- **Password** – an optional password that must be provided to use the terminal.
- **Prompt** – the line prefix that prompts users to input JavaScript.

Location

Overview

The Location settings panel is where the CueServer’s geographical location is set.

This is the location used in the determination of sunrise and sunset times.



Location Settings

Set the geographical location for this CueServer to allow it to automatically calculate the sunrise and sunset times for each day.

	Degrees	Minutes	Seconds		
Latitude:	34°	14'	59"	. 1000	North of the Equator
Longitude:	84°	3'	26"	4600	West of Greenwich

	Latitude	Longitude	
Decimal:	34.24975	-84.05735	Show in Apple Maps

Astronomical times are calculated by CueServer using its system clock, time zone, and daylight savings information.

The timeanddate.com web site can be used to show the Latitude and Longitude for any city, state, country or region in its database:

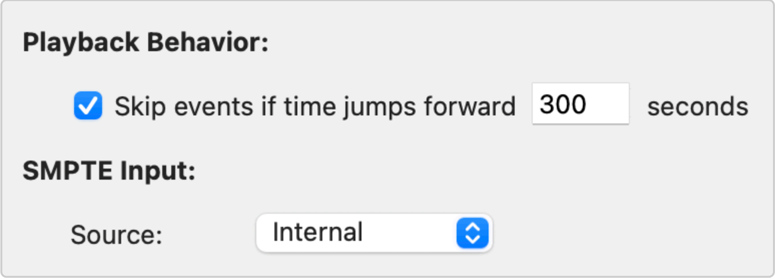
Use the **Show in Apple Maps** button to show the currently input coordinates on the map.

Timecode

Overview

The Timecode settings panel is where SMPTE Timecode options are configured.

See the SMPTE Command for information on using internal SMPTE Timecode.



Playback Behavior:

☑ Skip events if time jumps forward 300 seconds

SMPTE Input:

Source: Internal

There are two SMPTE Timecode configuration options:

Skip Events if time jumps forward *n* seconds.

This setting prevents events from firing that were “skipped” by the time adjusting forward by a defined number of seconds.

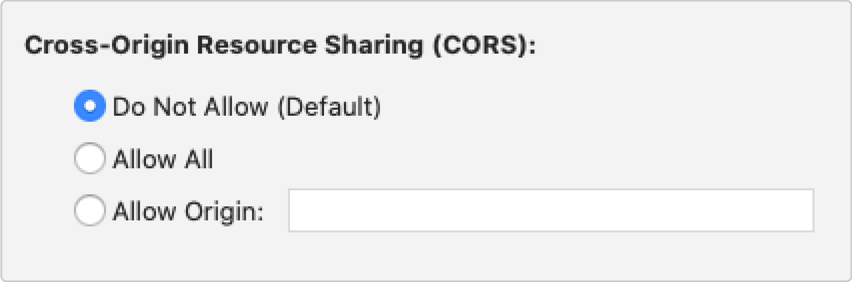
SMPTE Source

This determines what SMPTE Timecode source to use. Internal timecode is controlled and used solely by CueServer, external timecode is created outside of CueServer and received through the Audio Input port.

Web

Overview

The Web settings panel allows you to configure the Cross-Origin Resource Sharing (CORS) headers sent by Apache.



Cross-Origin Resource Sharing (CORS):

● Do Not Allow (Default)
○ Allow All
○ Allow Origin:

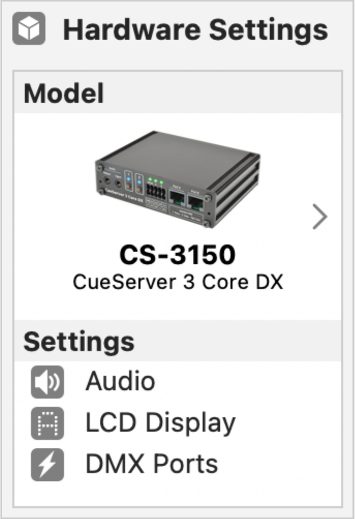
Cross-Origin Resource Sharing is a security feature that instructs web browsers to prevent code from accessing a server's responses from a remote origin. If not allowed, CORS *may* prevent external web applications not hosted on the CueServer from accessing the CueServer's files or API's. In most cases, CORS should be left disabled unless specifically needed.

The configuration options are:

- **Do Not Allow** – deny all cross-origin requests to CueServer (*default*).
- **Allow All** – allow all cross-origin requests, from any origin.
- **Allow Origin** – deny all cross-origin requests, except from the specified origin (i.e. 'localhost', or '192.168.2.100').

Hardware

The *Hardware Settings* page provides several views that control how CueServer hardware is configured.



The following Hardware Settings views are available:

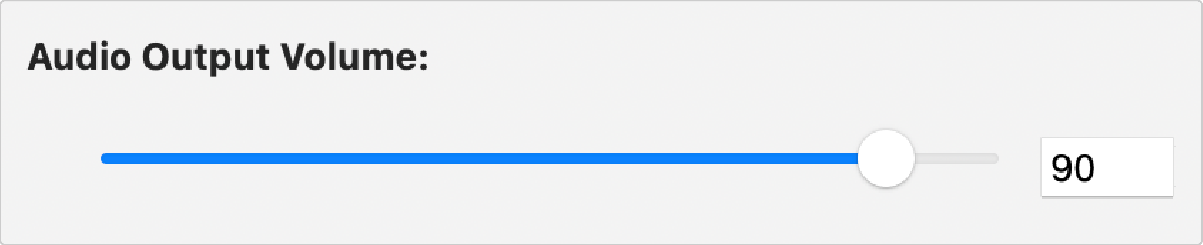
- Audio – audio output volume.
- LCD Display – configure LCD layout and brightness.
- DMX Ports / Modules – configure hardwire DMX ports and modules.

Note that if any of the Hardware 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 hardware views.

Audio

Overview

The Audio panel is where CueServer’s audio output volume is adjusted (if equipped).



Audio Output Volume:

90

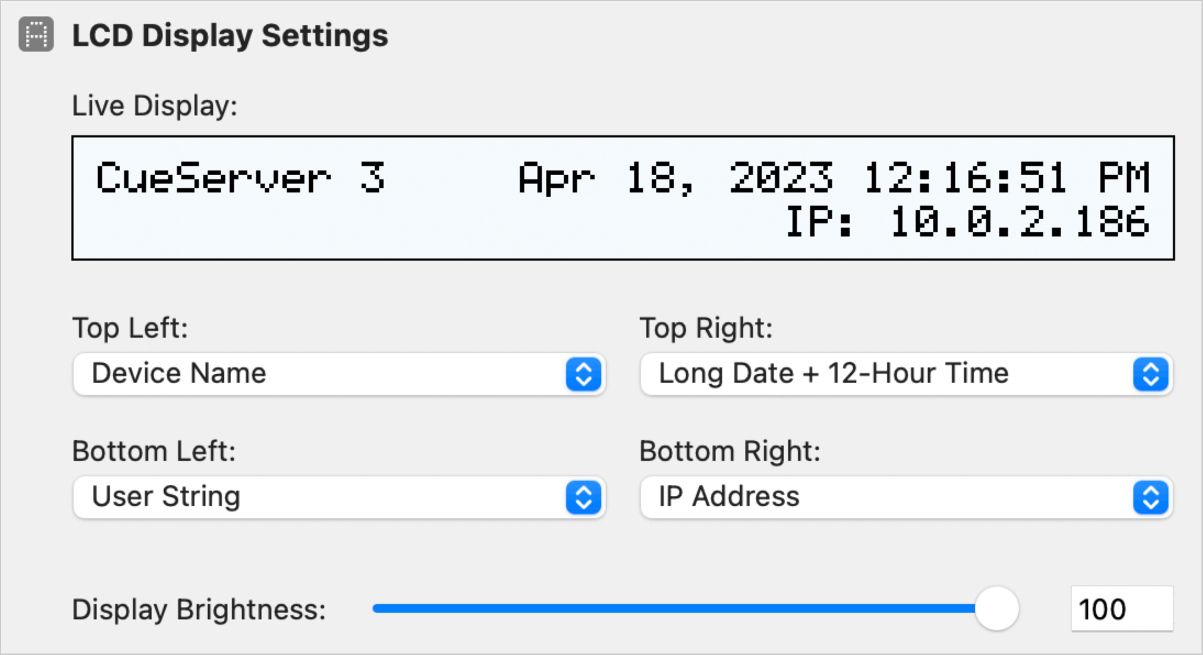
See the **Audio** section of System Variables for information on changing volume with CueScript.

The volume control in the Audio panel reflects the default volume for the edited show. If the volume is later adjusted using the applicable System Variable, that adjustment will not be reflected here and will be reset to this value after a reboot or show change. Changes in this panel **will** take effect immediately when applied.

LCD Display

Overview

The LCD Display settings panel configures the CueServer LCD display (if equipped).



LCD Display Settings

Live Display:

```
CueServer 3      Apr 18, 2023 12:16:51 PM
                          IP: 10.0.2.186
```

Top Left: Device Name

Top Right: Long Date + 12-Hour Time

Bottom Left: User String

Bottom Right: IP Address

Display Brightness: 100

Display Contents

The LCD is organized into into four individual sections: **Top Left**, **Top Right**, **Bottom Left**, and **Bottom Right**.

Each section can be assigned a built-in display option:

- **None** – leave the section blank.
- **Device Name** – the CueServer name defined in the network panel of the Navigator window.
- **User String** – the value of the LCD user string (see the Write command).
- **Show Name** – the name of the currently active show file.
- **Show Directory** – the file path to the currently active show file.
- **IP Address** – the CueServer's current IP address.
- **Timecode** – the current SMPTE timecode value.
- **IO Status** – graphical representation of each contact closure and digital output.
- **CPU Load** – real-time CPU load.
- **Long Date + 12-Hour Time** – alphanumerical date and time in 12-hour format (i.e. Apr 18, 2023 1:30:00 PM).
- **Short Date + 12-Hour Time** – numerical date and time in 12-hour format (i.e. 4/18/2023 1:30:00 PM).
- **Long Date + 24-Hour Time** – alphanumerical date and time in 24-hour format (i.e. Apr 18, 2023

13:30:00).

- **Short Date + 24-Hour Time** – numerical date and time in 24-hour format (i.e. 4/18/2023 13:30:00).
- **Long Date Only** – alphanumerical date (i.e. Apr 18, 2023).
- **Short Date Only** – numerical date (i.e. 4/18/2023).
- **12-Hour Time Only** – current time in 12-hour format (i.e. 1:30:00 PM).
- **24-Hour Time Only** – current time in 24-hour format (i.e. 13:30:00).

In addition to static configurations defined here, each section (or combination of sections) can be set with CueScript.

See the LCD Display section of System Variables.

Display Brightness

Use the Brightness slider to set the brightness of the LCD Display's backlight.

Brightness can also be set using CueScript, via the `lcd.backlight` System Variable.

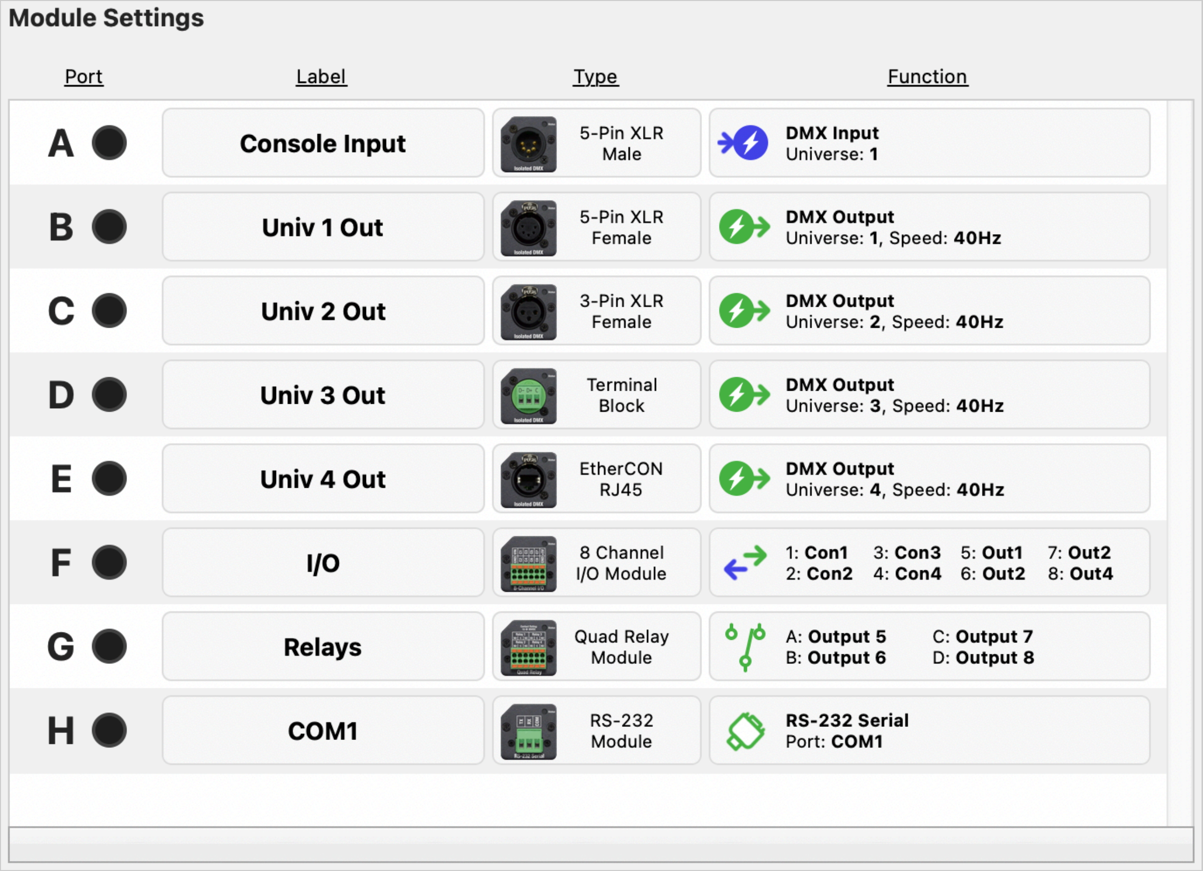
✻ The brightness control in the LCD Display panel reflects the default brightness for the edited show. If the brightness is later adjusted using the applicable System Variable, that adjustment will not be reflected here and will be reset to this value after a reboot or show change. Changes in this panel **will** take effect immediately.

DMX Ports / Modules

Overview

The DMX Ports / Modules editor is where built-in ports and module slots are configured.

For information about using protocols such as sACN, Art-Net, and KiNET, see the Universe Patch section.



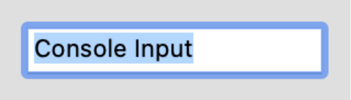
Module Settings

Port	Label	Type	Function
A ●	Console Input	5-Pin XLR Male	**DMX Input** Universe: **1**
B ●	Univ 1 Out	5-Pin XLR Female	**DMX Output** Universe: **1**, Speed: **40Hz**
C ●	Univ 2 Out	3-Pin XLR Female	**DMX Output** Universe: **2**, Speed: **40Hz**
D ●	Univ 3 Out	Terminal Block	**DMX Output** Universe: **3**, Speed: **40Hz**
E ●	Univ 4 Out	EtherCON RJ45	**DMX Output** Universe: **4**, Speed: **40Hz**
F ●	I/O	8 Channel I/O Module	1: **Con1** 3: **Con3** 5: **Out1** 7: **Out2** 2: **Con2** 4: **Con4** 6: **Out2** 8: **Out4**
G ●	Relays	Quad Relay Module	A: **Output 5** C: **Output 7** B: **Output 6** D: **Output 8**
H ●	COM1	RS-232 Module	**RS-232 Serial** Port: **COM1**

Port Label

Each port is labeled with a letter, such as “Port A”, but a custom label can be provided to help with identifying its function.

To add a label, click on the label block and enter a name in the resulting input field.



Console Input

Use the enter key or click outside of the label block to confirm the label.

Port Type

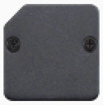
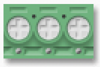
Each port is configured with a port type. The configured type should match the physical hardware.

To change a port’s type, select the type block and choose the desired type from the resulting menu.

Not all models support all types, the menu will display the supported types for the current model at the top, with unsupported types in a separate list at the bottom.

Note that if a chosen type is incompatible with the current CueServer model, a caution icon (⚠) will appear next to that port’s type.

Below are the available types:

Type	Hardware
 Empty	–
RJ45	**Fixed RJ45 Jack**
 Terminal Block	**Fixed Terminal Block**
EtherCON RJ45	**SM-DMX-RJ45** **CS-MOD-RJ45**
Terminal Block	**SM-DMX-TB** **CS-MOD-TB-ST**
Terminal Block	**SM-DMX-IDC** **CS-MOD-TB-IDC**
5-Pin XLR Female	**SM-DMX-X5F** **CS-MOD-X5F**
5-Pin XLR Male	**SM-DMX-X5M** **CS-MOD-X5M**

3-Pin XLR Female	**SM-DMX-X3F** **CS-MOD-X3F**
3-Pin XLR Male	**SM-DMX-X3M** **CS-MOD-X3M**
8 Channel I/O Module	**SM-IO8**
Quad Relay Module	**SM-RELAY4**
RS-232 Module	**SM-RS232**

Port Function

The port function defines how the port operates. Depending on the selected port type, various configuration options are available.

Select a port’s function block to change its configuration.

DMX Ports and Modules

The following port types are included in this category: **Fixed Terminal Block**, **Fixed RJ45**, **X5M**, **X5F**, **X3M**, **X3F**, **Terminal Block**, **IDC**, **RJ45**.

The following options are available:

Function	Additional Options
DMX Input	**Universe** – the internal CueServer universe incoming DMX data is mapped to.
DMX Output	**Universe** – the Internal CueServer universe to output. **Rate** – the DMX Update rate.
Station Bus	**Hub ID** – the hub ID used for the station network.
RS-485 Serial	**Port** – the assigned COM port (1-8) on station 0.

I/O Module

Each of the eight I/O ports can be individually configured with the following options:

Function	Additional Options
Off	–
Contact-Closure	**Number** – the assigned contact number on station 0.
Digital Output	**Number** – the assigned output number on station 0.

Relay Module

Each of the four relays can be individually configured with the following options:

Function	Additional Options
Off	–
Digital Output	**Number** – the assigned output number on station 0.

RS-232 Module

This module has the following options:

Function	Additional Options
Off	–
RS-232 Serial	**Port** – the assigned COM port (1-8) on station 0.

Extended Functionality

When assigning an I/O, Serial or Relay module, the port configuration determines which resources those items point to. The additional configuration options or rules are defined in the resources themselves.

RS-232 & RS-485

Modules with serial functions use the **COM (*n*)** parameter to define which COM port the module is assigned to on the Built-In station. Options such as protocol, baud rate, and data format are configured in the corresponding port's settings.

The number of available COM ports on the Built-In station is defined in the **Station Settings** panel under **Resources**. Once added, ports appear in the list on the left, below Buttons, Contacts, and Outputs.

I/O & Relay modules

Modules that are assigned Contact Closures or Digital Outputs are configured in the Built-In station. Functions and rules for each are configured in the corresponding contact or output settings.

The number of available Contacts or Outputs on the Built-In station is defined in the **Station Settings** panel under **Resources**. Once added, the contacts and outputs appear in the list on the left, below Buttons.

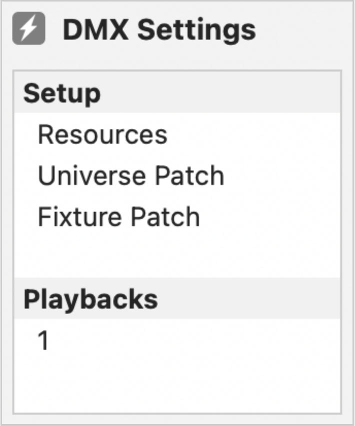
Station Bus

In this mode, DMX modules (or fixed RJ45) can be used to directly connect up to 24 individual 5-Wire CueStations (such as Mystique or Ultra stations) to a CueServer without the need for a separate station controller hub.

Once connected to the module (and external 24VDC power supply), stations can be added from the Stations panel using the station address and module Hub ID.

DMX

The *DMX* page provides several views that control how CueServer interprets and handles DMX data.


DMX Settings

Setup

Resources

Universe Patch

Fixture Patch

Playbacks

1

The following DMX views are available:

- Resources – control the number of available channels and playbacks.
- Universe Patch – control DMX universes and DMX-over-Ethernet configuration.
- Fixture Patch – patch fixtures to DMX addresses.
- Playback [n] – configure playback options.

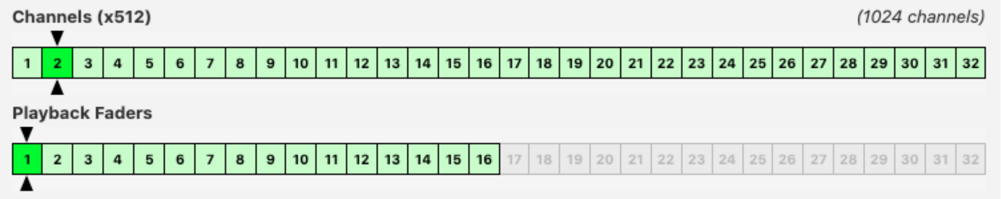
Note that if any of the DMX views has an important condition that needs to be shown to the user, the caution icon (⚠) will appear to the right of the corresponding line in the list of DMX views.

Resources

Overview

The resources editor lets you choose how many channels and playbacks are available.

Adding more of one reduces the possible amount of the other.



Below are the maximum available playbacks for each channel tier:

Universe Count	Channel Count	Playbacks Available
1	512	32
2	1024	16
3	1536	10
4	2048	8
5	2560	6
6	3072	5
7 – 8	3584 – 4096	4
9 – 10	4608 – 5120	3
11 – 16	5632 – 8192	2
17 – 32	8704 – 16384	1

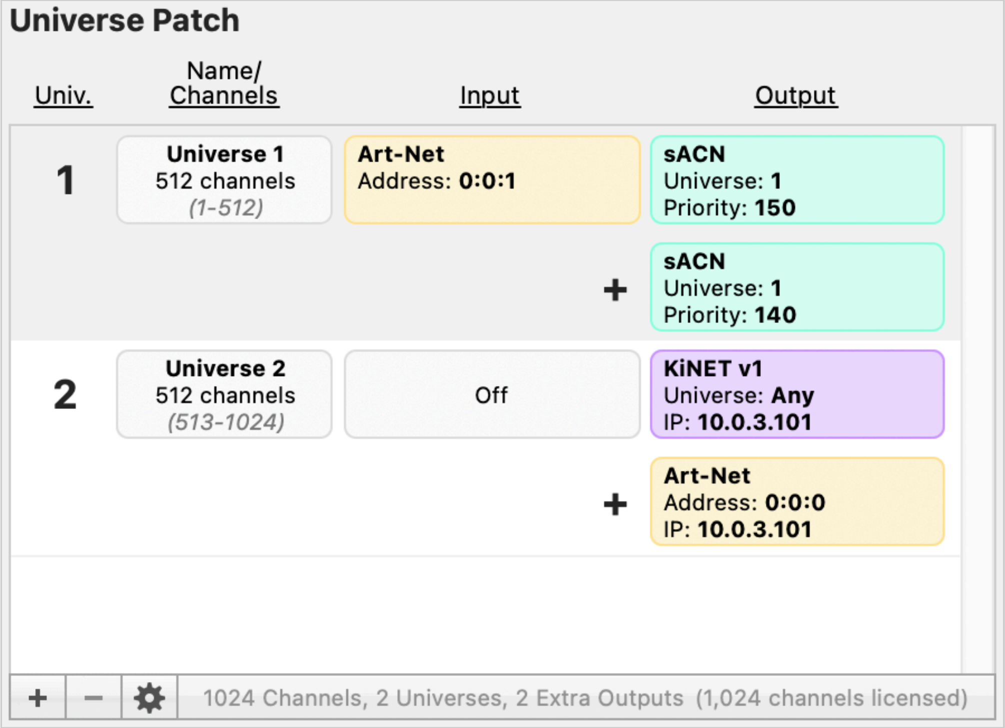
✻ This constraint is affected by channel count only. For example, If 32 universes are patched, but each universe is pared down to contain 10 channels, then all 32 playbacks would remain available, as the total number of channels would be 320 and fit in a single standard DMX universe.

Universe Patch

Overview

The universe patch editor is where CueServer universes are added, removed, and edited.

Each universe can have a name, number of channels, or DMX-over-Ethernet settings defined.



Universe Patch

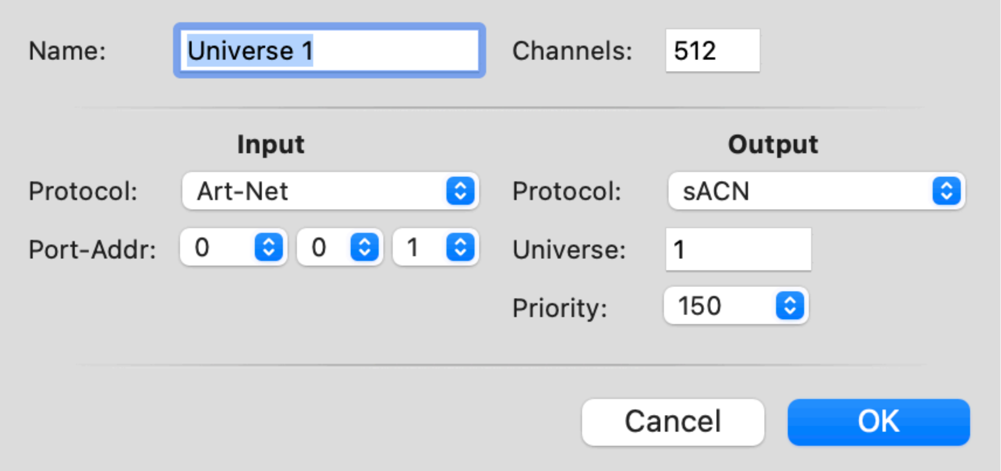
Univ.	Name/ Channels	Input	Output
1	**Universe 1** 512 channels *(1-512)*	**Art-Net** Address: **0:0:1**	**sACN** Universe: **1** Priority: **150**
		+	**sACN** Universe: **1** Priority: **140**
2	**Universe 2** 512 channels *(513-1024)*	Off	**KiNET v1** Universe: **Any** IP: **10.0.3.101**
		+	**Art-Net** Address: **0:0:0** IP: **10.0.3.101**

1024 Channels, 2 Universes, 2 Extra Outputs (1,024 channels licensed)

The universe patch is displayed as a series of rows, each with four columns **Universe Number**, **Name / Channels**, **Input** and **Output**. Rows without information in the first three columns are additional outputs for the closest universe above them. Across the bottom of the universe patch is a textual synopsis of the current configuration.

Edit a Universe

Each universe can be configured to have 1-512 channels, a custom name, and a DMX-over-Ethernet protocol configured as an Input and one or more outputs. These options can be configured in the properties panel of each universe.



Name: Universe 1
Channels: 512

Input
Protocol: Art-Net
Port-Addr: 0 0 1

Output
Protocol: sACN
Universe: 1
Priority: 150

There are three ways to access the properties panel:

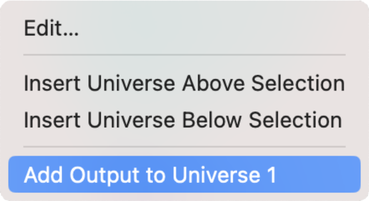
- Double-click on the target universe.
- Right-click on the target universe and select **Edit**.
- Select the target universe row and use the gear menu (⚙) at the bottom to select **Edit**.

After making changes, click **OK** to close the properties panel, and then click **Apply** in the bottom-right to save the changes.

> ✱ The priority fields in the Universe Patch reflect the default priority for each input and output in the edited show. If the priority of a universe is later adjusted using applicable System Variables, those adjustments will not be reflected here and will be reset to these values after a reboot or show change. Changes in this panel **will** take effect immediately.

Additional Outputs

Universes can have extra DMX-over-ethernet outputs to send the content from a universe to multiple destinations or over multiple protocols. To add an output to a universe, right-click on the universe or select the universe and then use the gear (⚙) menu, and choose **Add Output to Universe** from the list.


Edit...
Insert Universe Above Selection
Insert Universe Below Selection
Add Output to Universe 1

Once an additional output has been added, it will appear directly below the universe (or previous additional output) and the output can then be configured in the same manor as with a regular universe.

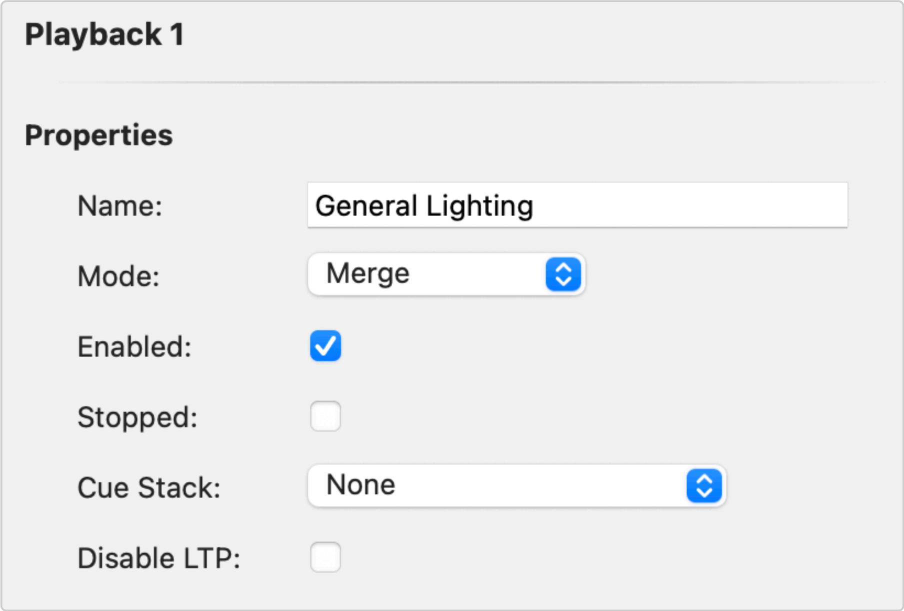
After adding or removing universes or outputs, click **Apply** in the bottom-right to save the changes.

Playback (n)

Overview

The playback settings panel is where a playback's default configuration is managed.

The settings defined here will take effect following a reboot or show change.



Playback 1

Properties

Name: General Lighting

Mode: Merge

Enabled: ☑

Stopped: □

Cue Stack: None

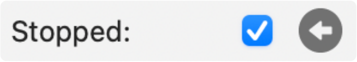
Disable LTP: □

Playbacks have the following configuration options:

- **Name** – the display name for the playback.
- **Mode** – the default mode for the playback.
- **Enabled** – the default enabled state of the playback.
- **Stopped** – the default stopped/started state of the playback.
- **Cue Stack** – the default Cue Stack assigned to the playback.
- **Disable LTP** – whether the playback adheres to Latest Takes Precedence (LTP).

The **Mode**, **Enabled**, **Stopped**, and **Cue Stack** options can be changed during a show's operation.

If an option does not match the playback's current state, an arrow (⬅) will appear to indicate the conflicting value.

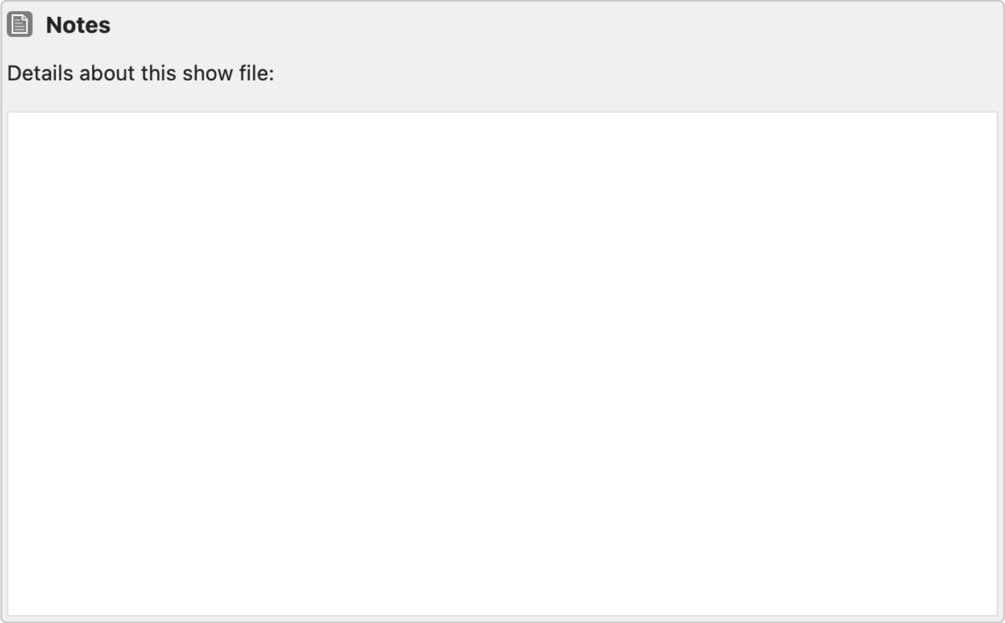


Stopped: ☑ ⬅

To update a configured option the to match that of the playbacks's current state, click the arrow next to the option and then click **Apply** to save the changes.

Notes

The notes view provides a place to store general information about the show file and its operation.



Details about this show file:

Hardware

Overview

This chapter describes the models of CueServer 2 available along with the various physical features, differences between models, specifications and explanation of indicators and displays.

For a description of the available models of CueServer 2, see these sections:

- Models
    - CS-900 CueServer 2 Pro
    - CS-920 CueServer 2 Mini
    - CS-940 CueServer 2 DIN
    - CS-950 CueServer 2 DIN

For explanations and specifications for the physical features of CueServer 2, see these sections:

- Power Input
- Ethernet Ports
- DMX Ports
- Audio Ports
- USB Ports
- LCD Display
- Function Buttons
- Contact Closures
- Digital Outputs
- Serial Ports
- Memory Card
- Reset Button

Models

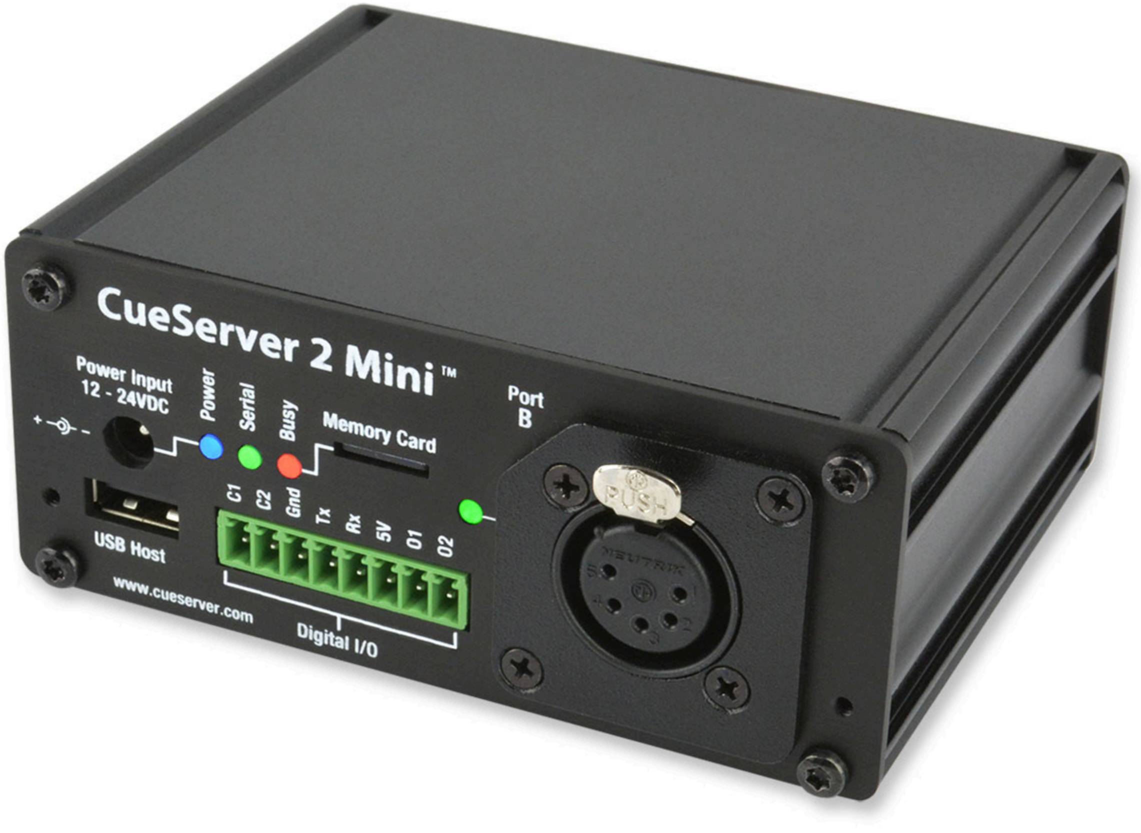
There are currently three CueServer 2 models available. A fourth CueServer 2 model (the CS-940) was replaced by the more capable CS-950.

CS-900 CueServer 2 Pro

CueServer 2 Nov 21, 2015 9:42:00 AM
Show: Main Ballroom IP: 10.0.1.234
CueServer 2
1
2
3
4
5
6
7
8

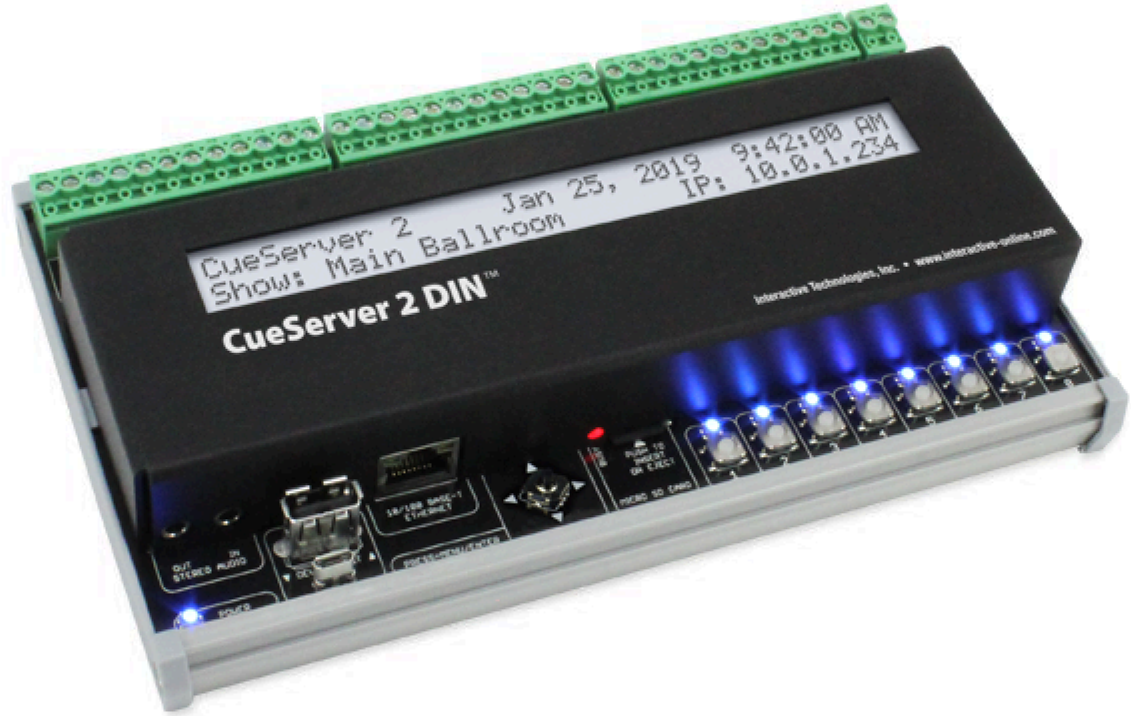
The CueServer 2 Pro is housed in an enclosure with removable brackets suitable for either 19” rack-mounting or desktop use. It features dual LAN ports, four field-replaceable DMX module slots, and eight user definable pushbutton inputs with customizable front-panel button caps. See the **CS-900 CueServer 2 Pro** section for more details for this model.

CS-920 CueServer 2 Mini



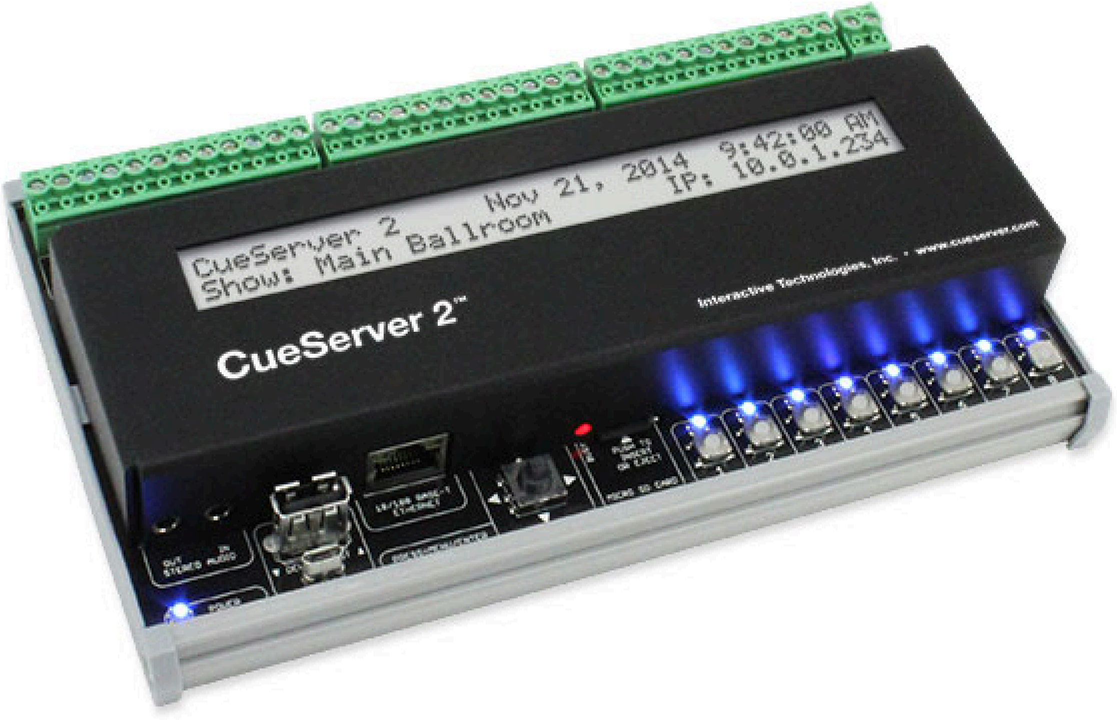
The CueServer 2 Mini is housed in a small enclosure suitable for desktop use. Optional brackets allow the CS-920 to be DIN Rail mounted, or attached to a flat surface, or hung from truss. It features a single LAN port, two field-replaceable DMX module slots, and two user definable pushbutton inputs. See the **CS-920 CueServer 2 Mini** section for more details for this model.

CS-950 CueServer 2 DIN


CueServer 2   Jan 25, 2019   9:42:00 AM
Show: Main Ballroom   IP: 10.0.1.234
CueServer 2 DIN™
Interactive Technologies, Inc. • www.interactive-online.com

The CueServer 2 DIN is housed in an enclosure with replaceable side brackets suitable for DIN-Rail or surface mounting. It features a single LAN port, four bi-directional DMX ports, and eight user definable pushbutton inputs. See the CS-950 CueServer 2 DIN section for more details for this model.

CS-940 CueServer 2 DIN (Discontinued: See CS-950)



The CueServer 2 DIN is housed in an enclosure with replaceable side brackets suitable for DIN-Rail or surface mounting. It features a single LAN port, two DMX input ports, two DMX output ports, and eight user definable pushbutton inputs. See the **CS-940 CueServer 2 DIN** section for more details for this model.

CS-900 CueServer 2 Pro

The CueServer 2 Pro (CS-900) is housed in a sturdy 1U rack-mount enclosure with removable brackets.

CueServer 2 Nov 21, 2015 9:42:00 AM
Show: Main Ballroom IP: 10.0.1.234
CueServer 2
1
2
3
4
5
6
7
8

CueServer 2 Pro features dual LAN ports for splitting Ethernet-based lighting and management data onto separate networks if desired.

CueServer 2 Pro boasts an innovative modular DMX port system. Four bi-directional DMX ports on the back of the unit are user-configurable with any of seven available port modules. These interchangeable modules allow CueServer 2 Pro to be customized for different installation environments eliminating the need for external adaptors.

The front-panel of CueServer 2 Pro has eight customizable function buttons. Each button has fully controllable RGB backlighting and field-replaceable legends for project personalization. A navigation keypad is used to operate the onboard LCD menu for basic system settings, show selection, and macro execution.

Features

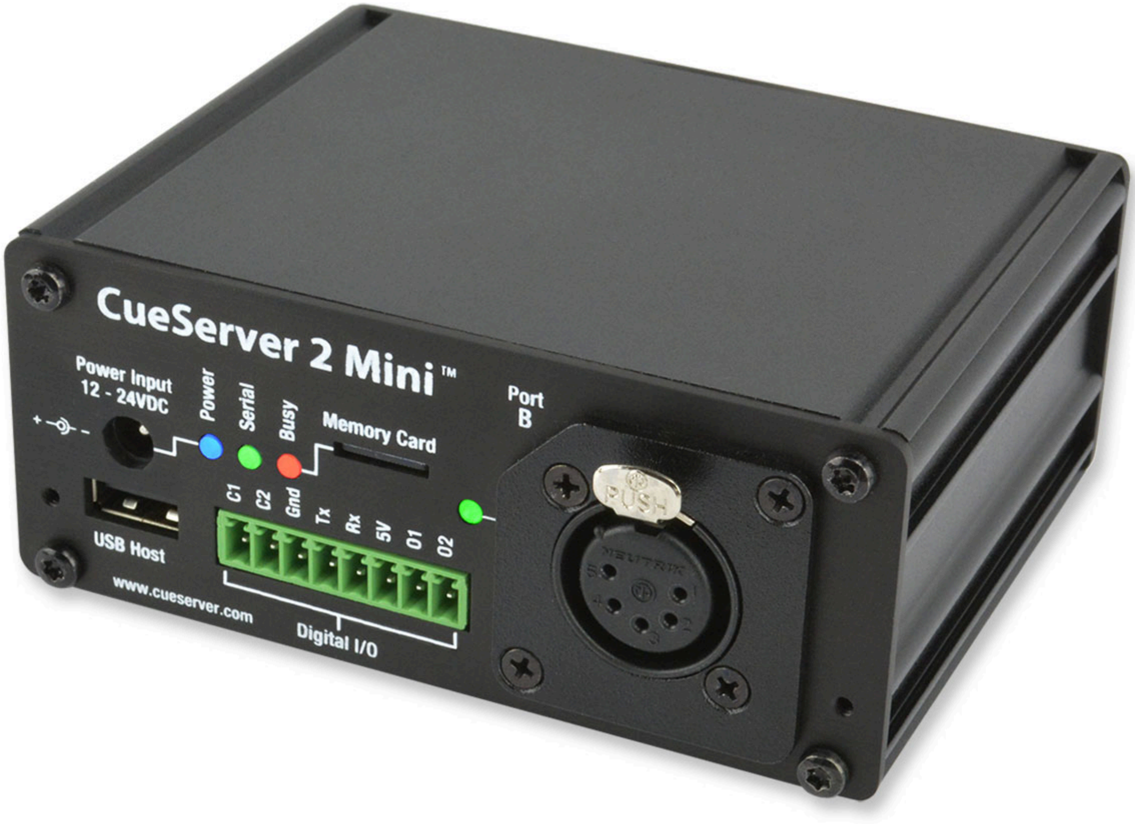
- Completely self-contained lighting playback, architectural processor, and DMX fade engine
- Seamless handling of Cue Lists, Presets, and Streams
- Control of up to 32 universes of DMX or 32 independent playback timelines
- Dynamic patching of up to 16,384 channels to 128 separate sACN, Art-Net, or KiNET universes
- Dual Ethernet ports for separate configurable lighting data and management networks
- Flexible module-based bi-directional DMX ports for custom jack configurations
- Front-panel configurable function buttons with RGB backlighting and field- replaceable legends
- Creation of lighting scenes directly or capture from external sources
- Powerful CueScript scripting language
- Real-Time clock with astronomical and calendar events
- Built-in web server for hosting custom interactive web pages
- Multi-show storage on removable microSD memory card
- System integration via Ethernet, Serial, Digital I/O, and Audio
- Compatible with CueStation buttons and CueTouch touchscreens
- Easy interfacing with Crestron, AMX, Vantage, Control 4, Medialon, Savant and other automation

systems

- Native programming environment for both Mac and Windows
- 1U rack-mounted enclosure with removable brackets

CS-920 CueServer 2 Mini

The CueSevrer 2 Mini (CS-920) is the smallest of the CueServer 2 models and is housed in a rugged anodized aluminum enclosure suitable for desktop use or panel, DIN, or truss mounting using optional bracket kits.



The CueServer 2 Mini can output shows utilizing up to 16,384 channels and features two built-in modular DMX slots that are user-configurable with any of seven available port modules. These interchangeable modules allow CueServer 2 Mini to be customized for different installation environments eliminating the need for external adapters.

CueServer 2 Mini also features two user-definable function buttons with RGB indicators, two contact closure inputs, two low-voltage digital outputs, a serial port, and stereo audio output.

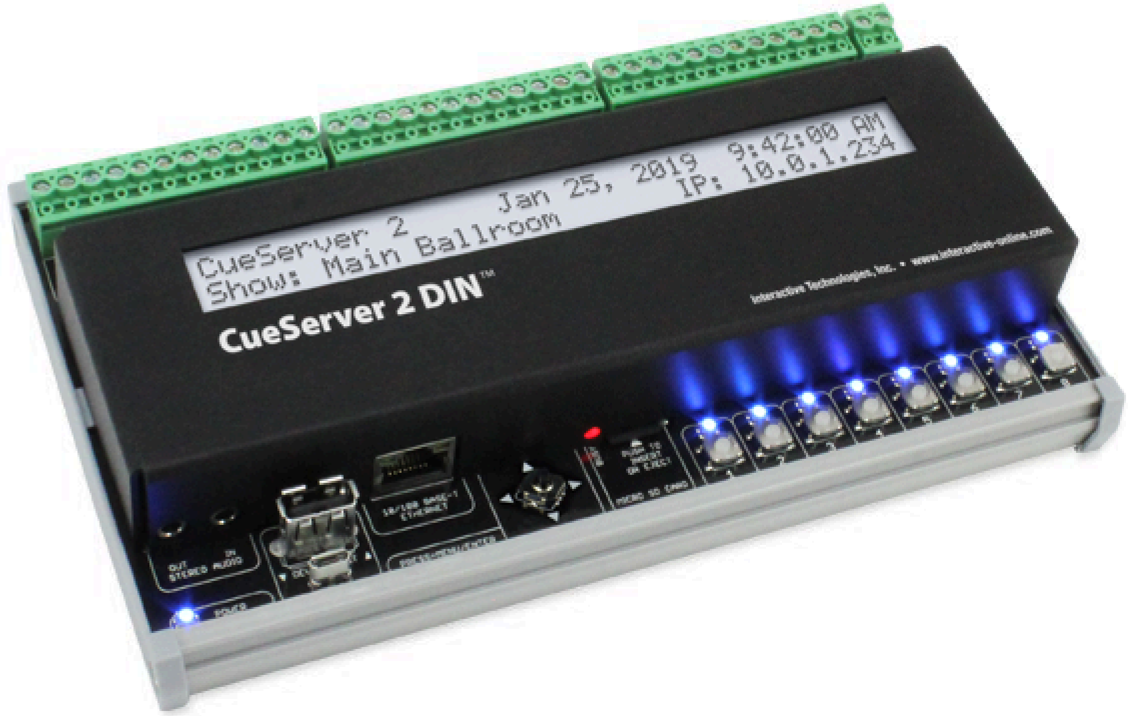
Features

- Completely self-contained lighting playback, architectural processor, and DMX fade engine
- Seamless handling of Cue Lists, Presets, and Streams
- Control of up to 32 universes of DMX or 32 independent playback timelines
- Dynamic patching of up to 16,384 channels to 128 separate sACN, Art-Net, or KiNET universes
- Flexible module-based bi-directional DMX ports for custom jack configurations
- Front-panel configurable function buttons with RGB indicator LEDs
- Creation of lighting scenes directly or capture from external sources
- Powerful CueScript scripting language
- Real-Time clock with astronomical and calendar events

- Built-in web server for hosting custom interactive web pages
- Multi-show storage on removable microSD memory card
- System integration via Ethernet, Serial, Digital I/O, and Audio
- Compatible with CueStation buttons and CueTouch touchscreens
- Easy interfacing with Crestron, AMX, Vantage, Control 4, Medialon, Savant and other automation systems
- Native programming environment for both Mac and Windows
- Small anodized aluminum enclosure with optional brackets

CS-950 CueServer 2 DIN

The CueServer 2 DIN (CS-950) is housed in an enclosure suitable for DIN rail, surface, or panel mounting. The DIN rail brackets accomodate standard 35mm rail. Mounting flanges are included for surface or panel mounting.


CueServer 2 Jan 25, 2019 9:42:00 AM
Show: Main Ballroom IP: 10.0.1.234
CueServer 2 DIN™
Interactive Technologies, Inc. • www.interactive-online.com

Connections for power, DMX, contact closures, and digital outputs are made using removable terminal blocks across the top edge of the unit. Ethernet, USB, and stereo audio are connected along the bottom edge.

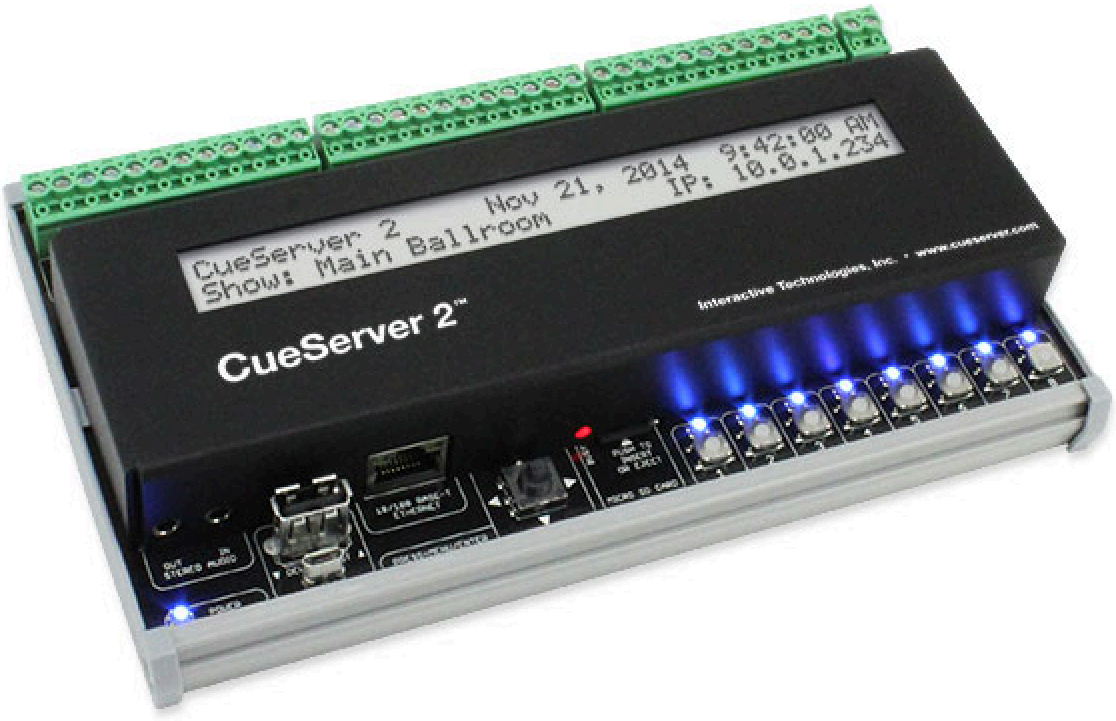
CueServer 2 DIN’s front panel has eight customizable function buttons. Each button has a fully controllable RGB indicator. A navigation joystick is used to operate the onboard LCD menu for basic system settings, show selection, and macro execution.

Features

- Completely self-contained lighting playback, architectural processor, and DMX fade engine
- Seamless handling of Cue Lists, Presets, and Streams
- Control of up to 32 universes of DMX or 32 independent playback timelines
- Dynamic patching of up to 16,384 channels to 128 separate sACN, Art-Net, or KiNET universes
- Front-panel configurable function buttons with RGB indicator LEDs
- Creation of lighting scenes directly or capture from external sources
- Powerful CueScript scripting language

- Real-Time clock with astronomical and calendar events
- Built-in web server for hosting custom interactive web pages
- Multi-show storage on removable microSD memory card
- System integration via Ethernet, Serial, Digital I/O, and Audio
- Compatible with CueStation buttons and CueTouch touchscreens
- Easy interfacing with Crestron, AMX, Vantage, Control 4, Medialon, Savant and other automation systems
- Native programming environment for both Mac and Windows
- Standard DIN-Rail mounting or surface/panel mounting

CS-940 CueServer 2 DIN


CueServer 2 Nov 21, 2014 9:42:00 AM
Show: Main Ballroom IP: 10.0.1.234
CueServer 2™
Interactive Technologies, Inc. • www.cueserver.com
OUT IN STEREO AUDIO
10/100 BASE-T ETHERNET
PUSH TO INSERT OR EJECT
MICRO SD CARD
POWER

Please Note: The CS-940 has been discontinued. It has been replaced by the more capable CS-950.

The CueServer 2 DIN (CS-940) is housed in an enclosure suitable for DIN rail, surface, or panel mounting. The DIN rail brackets accomodate standard 35mm rail. Mounting flanges are included for surface or panel mounting.

Connections for power, DMX, contact closures, and digital outputs are made using removable terminal blocks across the top edge of the unit. Ethernet, USB, and stereo audio are connected along the bottom edge.

CueServer 2 DIN’s front panel has eight customizable function buttons. Each button has a fully controllable RGB indicator. A navigation joystick is used to operate the onboard LCD menu for basic system settings, show selection, and macro execution.

Features

- Completely self-contained lighting playback, architectural processor, and DMX fade engine
- Seamless handling of Cue Lists, Presets, and Streams
- Control of up to 32 universes of DMX or 32 independent playback timelines

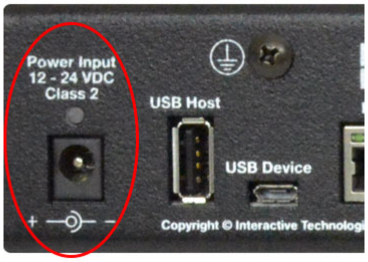
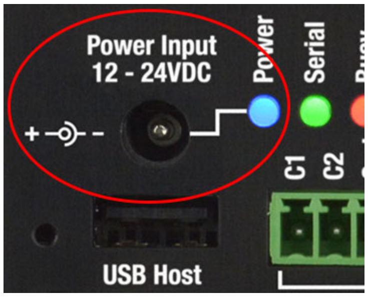
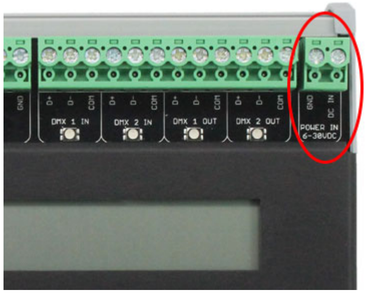
- Dynamic patching of up to 16,384 channels to 128 separate sACN, Art-Net, or KiNET universes
- Front-panel configurable function buttons with RGB indicator LEDs
- Creation of lighting scenes directly or capture from external sources
- Powerful CueScript scripting language
- Real-Time clock with astronomical and calendar events
- Built-in web server for hosting custom interactive web pages
- Multi-show storage on removable microSD memory card
- System integration via Ethernet, Serial, Digital I/O, and Audio
- Compatible with CueStation buttons and CueTouch touchscreens
- Easy interfacing with Crestron, AMX, Vantage, Control 4, Medialon, Savant and other automation systems
- Native programming environment for both Mac and Windows
- Standard DIN-Rail mounting or surface/panel mounting

Power Input

All models of CueServer 2 can be powered by a 12 to 24 VDC Class 2 input.

Although the power requirements are similar for the various models, their connectors and current requirements are different. The rack-mounted CS-900 and the miniature CS-920 both have a standard DC power input jack. The surface-mounted CS-940/950 uses screw terminals that are suitable for hardwire connections to DC power.

Specifications

	CS-900	CS-920	CS-940/950
Power Input	12-24 VDC	12-24 VDC	12-24 VDC
Minimum Power Supply Wattage	9 Watts	7 Watts	8 Watts
Connector	2.1mm DC Power Jack	2.1mm DC Power Jack	Screw Terminals
			

Pinout	Center = DC Input (V+)
	Barrel = Ground
		Center = DC Input (V+)
		Barrel = Ground
			1 = DC Input (V+)
			2 = Ground

Indicators

Color & Pattern	Description
● Solid Blue	Power is on, all systems normal
☀ Slowly Flashing Blue	Device is in the process of starting up
☀ Slowly Flashing Magenta	Device is in Bootloader Mode (contact Technical Support)

Slowly Alternating Red/Yellow	Device is writing firmware or boot information (do not unplug)
Slowly Alternating Blue/Magenta	The System Log has a new message
Quickly Alternating Blue/Magenta	The System Log has an *important* new message
Slowly Flashing Red	Device has shut down (must power cycle to reboot)
Off	Device has no power

Ethernet Ports

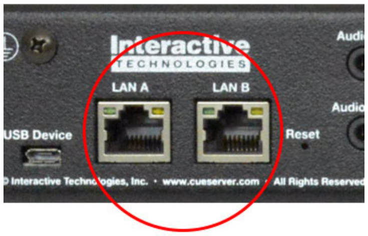
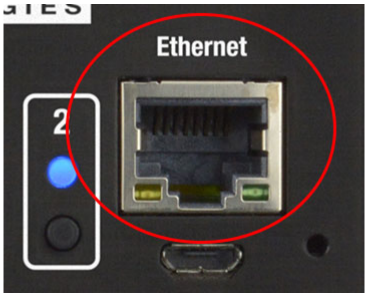
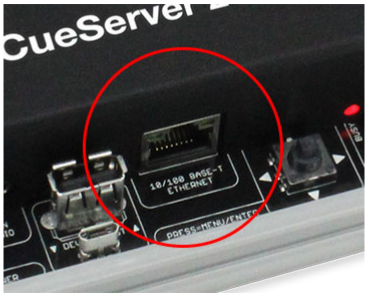
CueServer 2 is an Ethernet-based product. An Ethernet connection to a local network is required to program CueServer using the CueServer Studio 2 software. Additionally, Ethernet is required if a DMX-over-Ethernet protocol (such as sACN, Art-Net, or KiNET) is going to be used to input or output DMX over Ethernet, or to connect to button stations, use the CuePad iOS application or to remotely manage the device. Only if the CueServer has already been programmed and no Ethernet protocols are needed to run the show can the CueServer be used without a network connection.

The rack-mounted CS-900 has two Ethernet ports, the miniature CS-920 and the surface-mounted CS-940/950 only have a single Ethernet port.

Units with two Ethernet ports can be configured in one of two modes. The first mode is to provide only a single network on both ports using a built-in Ethernet Switch. The second mode is to separate the two ports into different LANs, with management data on LAN A and lighting data on LAN B. In this second mode, each port will have separate IP addresses.

See the Ethernet Protocols section for a description of the protocols supported by CueServer.

Specifications

	CS-900	CS-920	CS-940/950
Ethernet Ports	2	1	1
Network Mode(s)	Single Network with Built-In Switch or Two Separate LANs	Single Network	Single Network
Factory Default Settings	Single Network DHCP Enabled Fallback IP 10.0.1.234	DHCP Enabled Fallback IP 10.0.1.234	DHCP Enabled Fallback IP 10.0.1.234
	 Interactive TECHNOLOGIES LAN A LAN B USB Device Reset Audio Interactive Technologies, Inc. • www.cueserver.com • All Rights Reserved		

Indicators

Left LED	Description

● Off	No Link, bad cable, or no connection on opposite end
🟢 Solid Green	Ethernet link is established
Right LED	**Description**
● Off	No Link, bad cable, or no connection on opposite end
🟠 Solid Amber	No data activity
🔆 Flashing Amber	Data activity

Ethernet Protocols

CueServer 2 supports several *Ethernet Protocols* for the transmission and reception of DMX lighting control data, remote control of the CueServer, synchronization of network time, web services and more.

See the following sections for additional information about each Ethernet Protocol as implemented by CueServer:

- sACN (Streaming ACN) Protocol
- Art-Net Protocol
- KiNET Protocol
- CueScript Protocol
- CueStation Protocol
- HTTP Protocol
- DHCP Protocol
- NTP Protocol

sACN (Streaming ACN) Protocol

sACN (or Streaming ACN) is a preferred method of sending and receiving DMX-over-Ethernet to and/or from a CueServer.

The following table lists the general specifications for the CueServer implementation of sACN:

- Compliant with ANSI Standard E1.31-2009 (sACN)
- May send and/or receive up to 128 universes of sACN simultaneously
- May send and/or receive sACN packets with an arbitrary universe number between 1 and 63999
- sACN is sent at a maximum rate of 40Hz for each universe when channel values are changing
- sACN is sent at a minimum rate of 1Hz for each universe when channel values are static
- Supports the sending of user-defined priority levels for each universe
- Received sACN is merged with hardwired DMX designated for the same universe
- Ignores incoming data marked as Preview Data
- Performs immediate stream termination when a Stream Terminated packet is received
- Maintains proper sequence number transmission separately for each output universe
- Sends the universe’s name as the sACN source name for each universe
- Ignores packets with start code 0xDD (used for slot-by-slot priority)
- Each universe times out after 2.5 seconds when no packets for that universe are not received
- CueServer does not receive its own sACN output

For more information about sACN, please visit the **ESTA Technical Standards Program** at tsp.esta.org.

Art-Net Protocol

Art-Net is a method of sending and receiving DMX-over-Ethernet to and/or from a CueServer.

Art-Net is owned and copyright by Artistic Licence Holdings Ltd. Artistic Licence has published the specification and made it available for anyone to use on a royalty-free basis.

The following table lists the general specifications for the CueServer implementation of Art-Net:

- Compliant with the Art-Net 3 Specification by Artistic License
- May send and/or receive up to 128 universes of Art-Net simultaneously
- May send and/or receive Art-Net packets with an arbitrary port address (network/sub-net/universe) from 0:0:0 thru 127:F:F
- Art-Net is sent at a maximum rate of 40Hz for each universe when channel values are changing
- Art-Net is sent at a minimum rate of 1Hz for each universe when channel values are static
- Art-Net may be sent to the limited broadcast address, directed broadcast address or unicast to a specific IP address
- Received Art-Net is merged with hardwired DMX designated for the same universe
- Maintains proper sequence number transmission separately for each output universe
- Each universe times out after 6 seconds when no packets for that universe are not received
- CueServer’s implementation of Art-Net does not yet support “automatic” IP configuration via the ArtPoll method
- CueServer does not receive its own Art-Net output

For more information about Art-Net, please visit the **Art-Net Home Page** at art-net.org.uk.

KiNET Protocol

KiNET is an alternate method of sending DMX lighting control values to from a CueServer to lighting fixtures and/or power supplies that support the proprietary Philips/Color Kinetics KiNET protocol.

The following table lists the general specifications for the CueServer implementation of KiNET:

- Compliant with the v1 and v2 versions of the KiNET protocol
- May send and/or receive up to 128 universes of KiNET simultaneously
- A universe outputting KiNET v1 may be sent to any arbitrary IP address
- A universe outputting KiNET v2 may specify a port number and DMX range
- KiNET is sent at a maximum rate of 40Hz for each universe when channel values are changing
- KiNET is sent at a minimum rate of 1Hz for each universe when channel values are static

For more information about KiNET, please visit **Color Kinetics** at colorkinetics.com.

Interactive Technologies is a KiNET licensee by Philips/ColorKinetics.

CueScript Protocol

CueScript Protocol is a method of sending CueScript commands to a CueServer over Ethernet.

CueServer listens for incoming UDP packets on port 52737 that contain a valid CueScript command string.

CueScript packets may be sent to CueServer by:

- Unicast to the CueServer’s IP Address
- Multicast to the CueServer group address 239.255.204.2

The payload of the packet can be any valid CueScript command, such as:

- Cue 1 Go
- Q1G
- Macro 7
- M7
- Channel 1>10 At FL
- C1>10AFL
- Button 1 On; Wait 3.5; Button 1 Off
- If (‘x’ = 1) Then Playback 7 Clear

CueStation Protocol

CueStation Protocol is the method of communication between a CueServer and the CueStation Hub.

The following table lists the general specifications for the CueServer implementation of CueStation protocol:

- Can communicate with one or more CueStation Hubs
- Supports HUB IDs from 1..254 for unique identification of multiple hubs on a single network
- Uses the CueStation Multicast group 239.255.204.3
- Uses multicast traffic only for configuration-free setup

HTTP Protocol

Hypertext Transfer Protocol (HTTP) is an network protocol for “hypermedia information systems”. HTTP is the foundation of data communication for the World Wide Web.

CueServer uses HTTP for a variety of purposes.

CueServer uses its embedded HTTP web server to allow custom web pages to be served by the active project file. This allows a project to be set up in CueServer that includes its own customized web based content. A project in CueServer can host a “home page” that acts as a landing page for the project, and nearly any other pages, images, documents, and other content as necessary. CueServer also has the ability to interact with this web content, making the web pages interact live with the running CueServer lighting control and automation.

CueServer uses HTTP to communicate with CueServer Studio. All of the transactions between CueServer Studio and the CueServer device are occurring over HTTP. This allows CueServer Studio to be able to remotely control a CueServer using nothing other than TCP Port 80 access over the Internet.

CueServer uses HTTP to communicate with various companion “apps”, such as CuePad and our touchscreen options.

CueServer exposes an open Application Programming Interface (API) through HTTP for software developers to use to interact with the device. Custom applications can be written in nearly any computer language that communicates with CueServer via HTTP.

For more information about HTTP, please visit the * Hypertext Transfer Protocol Wikipedia Page* at wikipedia.org/wiki/Hypertext_Transfer_Protocol.

DHCP Protocol

Dynamic Host Configuration Protocol (DHCP) is a network protocol used to automatically configure devices on the network. With DHCP, devices request IP addresses and networking parameters automatically from a DHCP server, reducing the need for a network administrator or a user to configure these settings manually.

CueServer can optionally use DHCP to automatically set its network parameters (such as IP Address, Subnet, Gateway, etc.) without requiring the user to adjust these settings manually.

By default from the factory, CueServer has DHCP turned on, which means that it will attempt to find a DHCP server and automatically configure itself as the CueServer is powered on. CueServer can be configured to have DHCP turned off, which would allow manually assigned (static) network parameters to be used.

Some CueServer models may be configured to have more than one LAN connection. On models configured this way, DHCP may be used separately on each LAN.

For more information about DHCP, please visit the **Dynamic Host Configuration Protocol Wikipedia Page** at wikipedia.org/wiki/Dynamic_Host_Configuration_Protocol.

NTP Protocol

Network Time Protocol (NTP) is a networking protocol for clock synchronization between computer systems over Ethernet.

CueServer uses NTP to keep its clock as accurate as possible without requiring the user to adjust the time manually.

NTP is intended to synchronize computers to within a few milliseconds of Coordinated Universal Time (UTC). It an algorithm to select accurate time servers and is designed to mitigate the effects of variable network latency. NTP can usually maintain time to within tens of milliseconds over the public Internet, and can achieve better than one millisecond accuracy in local area networks under ideal conditions. Asymmetric routes and network congestion can cause errors of 100 ms or more.

Because a CueServer that intends to use NTP to synchronize its time must be able to reach a NTP Server via Ethernet, this function can only work if the CueServer is on a network that has access to the Internet, or the facility must have an NTP Server on its local network.

For more information about NTP, please visit the **Network Time Protocol Wikipedia Page** at wikipedia.org/wiki/Network_Time_Protocol.

Ethernet Port Numbers

CueServer 2 uses several different Ethernet protocols to communicate with other devices on the network.

The following table lists each of the TCP and UDP port numbers used by CueServer's various features and facilities:

Port	Protocol	Service	Direction	Description
22	TCP	SSH	Incoming	Terminal access to CueServer's internal command shell. This is only necessary for advanced users or factory troubleshooting.
23	TCP	Telnet	Incoming	[Depreciated] Terminal access to CueServer's internal command shell. This is only necessary for advanced users or factory troubleshooting. *No longer active beginning with Firmware Version 3.1*
68	UDP	DHCP	Bidirectional	Used by DHCP service if enabled. Not active if CueServer is given a static IP address.
80	TCP	HTTP	Incoming	CueServer's internal web server. Used for communication with CueServer Studio, virtual touchscreens, Insite touchscreens, user's custom web content, and the CueServer web-based API.
123	UDP	NTP	Bidirectional	Used for polling Network Time Protocol servers. Only active when NTP service is enabled.
5568	UDP	sACN	Outgoing	Streaming ACN (sACN) data packets. This port is only active when configured to receive and/or transmit sACN protocol.
6038	UDP	KiNET	Outgoing	KiNET v1 and v2 data packets. This port is only active when configured to receive and/or transmit KiNET protocol.
6454	UDP	Art-Net	Outgoing	Art-Net data packets. This port is only active when configured to receive and/or transmit Art-Net protocol.
52737	UDP	CueServer	Bidirectional	Used for CueServer auto-discovery, CueScript commands, and CueStation Hub Protocol.

Port Forwarding for CueServer Studio

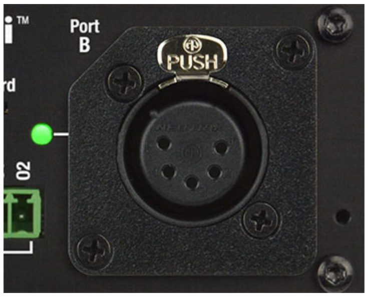
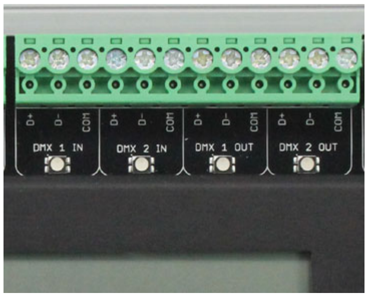
To provide access to a CueServer for *CueServer Studio* remotely, it is only necessary to open Port 80. All communication between *CueServer Studio* and CueServer devices occurs over this port, with the sole exception of auto-discovery. The auto-discovery function cannot work across different networks, so it is necessary to use the “Add Remote” feature in *Studio* to add the CueServer to the list of available devices.

DMX Ports

In addition to being able to transmit and receive DMX-over-Ethernet, CueServer also has built-in DMX ports for hard-wired DMX connections to fixtures, dimmers, consoles and virtually any other DMX compatible devices.

The rack-mounted CS-900 has four replaceable DMX module slots, and the miniature CS-920 has two replaceable DMX module slots, each of which can accept any of seven available DMX modules for input or output of DMX. The surface-mounted CS-940 has two DMX input ports and two DMX output ports that are available on the unit's pluggable terminal block strips. The surface-mounted CS-950 has four bi-directional DMX ports that are available on the unit's pluggable terminal block strips.

Specifications

	CS-900	CS-920	CS-940	CS-950
DMX Ports	4	2	4	4
Bi-Directional Ports	4	2	–	4
Fixed Input Ports	–	–	2	–
Fixed Output Ports	–	–	2	–
Replacable DMX Modules	Yes	Yes	No	No
				

Indicators

Description	CS-900/920/950	CS-940
Port Disabled	● Off	● Off
DMX Output (Active)	🟢 Solid Green	🟢 Solid Green

DMX Input (Active)	🔵 Solid Blue	🟢 Solid Green
DMX Input (No Input)	🟣 Solid Magenta	⚫ Off
Bad Universe	🟡 Slowly Flashing Yellow	🟡 Slowly Flashing Yellow
Bad Port Direction	n/a	🔴 Quickly Flashing Red
Error Condition	🔴 Solid Red	🔴 Solid Red

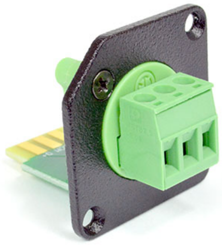
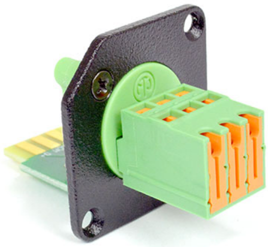
DMX Modules

The **CS-900 CueServer 2 Pro** and the **CS-920 CueServer 2 Mini** use a unique field-replaceable DMX module system for allowing the DMX ports to be customized for each individual project's needs.

The CS-900 ships with four blank plates covering the module slots. The CS-920 ships with two blank plates. Optional modules can be purchased and installed into each of these slots. Using CueServer Studio, each slot can be configured as a DMX input or output.

The following list shows the available modules:

Module	Description	DMX Pinout
	MOD-X5F 5-Pin Female XLR	1 – Common 2 – Data - 3 – Data + 4 – NC 5 – NC
	MOD-X5M 5-Pin Male XLR	1 – Common 2 – Data - 3 – Data + 4 – NC 5 – NC
	MOD-X3F 3-Pin Female XLR	1 – Common 2 – Data - 3 – Data +

	MOD-X5M 3-Pin Male XLR	1 – Common 2 – Data - 3 – Data +
	MOD-RJ45 Ethercon RJ45	1 – Data + (White/Orange) 2 – Data – (Orange) 3 – NC (White/Green) 4 – NC (Blue) 5 – NC (White/Blue) 6 – NC (Green) 7 – Common (White/Brown) 8 – Common (Brown) * Colors use T-568B Standard
	**MOD-TB-ST** Screw Terminals	1 – Data - 2 – Data + 3 – Common
	**MOD-TB-IDC** IDC Terminals	1 – Data - 2 – Data + 3 – Common

Audio Ports

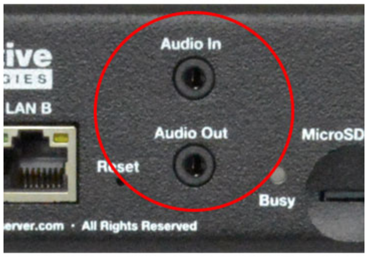
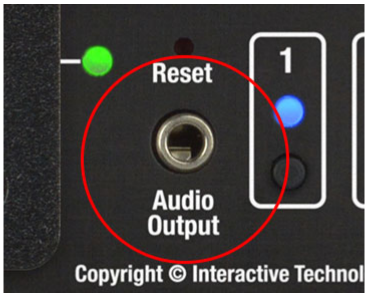
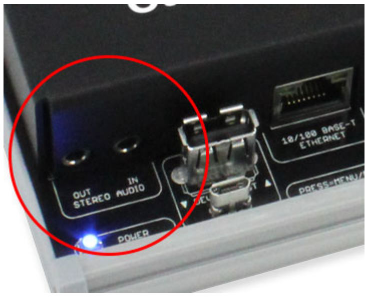
CueServer 2 has built-in stereo audio.

The stereo output is able to play sound effects, music and other audio clips in response to CueScript commands triggered by the active show.

Models that provide an audio input jack currently support the ability for CueServer to receive and decode SMPTE timecode via LTC audio input.

See the section Supported Audio File Formats for a listing of what types of audio files that CueServer 2 can play.

Specifications

	CS-900	CS-920	CS-940/950
Audio Input	1	–	1
Audio Output	1	1	1
Connector	1/8” Phono Plug	1/8” Phono Plug	1/8” Phono Plug
	 Audio In LAN B Audio Out MicroSD Reset Busy erver.com • All Rights Reserved	 Reset 1 Audio Output Copyright © Interactive Technol	

Supported Audio File Formats

CueServer 2 supports the playback of the following common popular audio file formats:

File Extension	Description
`.aif .aifc .aiff .aiffc`	Audio Interchange File Format (Apple)
`.au .snd`	Unix Audio File (Sun, NeXT, UNIX)
`.flac`	Free Lossless Audio (Open-Source)
`.mp2 .mp3`	MPEG Audio File (MP3)
`.ogg .vorbis`	Ogg Vorbis Audio File (Open-Source)
`.wav`	Waveform Audio File (See WAV Sample Formats)

CueServer 2 also contains partial support for the following formats (use cautiously as these have not been tested):

```
8svx amb amr-nb amr-wb anb avr awb cdda cdr cvs cvsd cvu dat dvms gsm gsrt hcom htk ima
ircam lpc lpc10 maud nist prc sds sln smp sndfile sndr sndt sou sox sph txw vms voc vox
wavpcm wv wve xa
```

CueServer 2 **DOES NOT** support the following audio file formats:

File Extension	Description
`.m4a .m4p`	iTunes Advanced Audio Coding (AAC)

WAV Sample Formats

The following sample formats are supported by CueServer 2’s WAV audio playback:

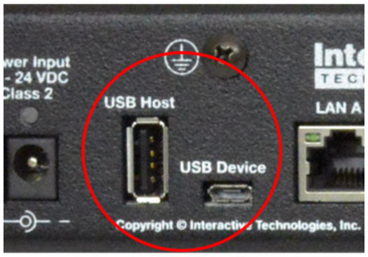
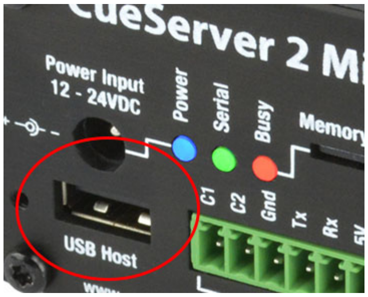
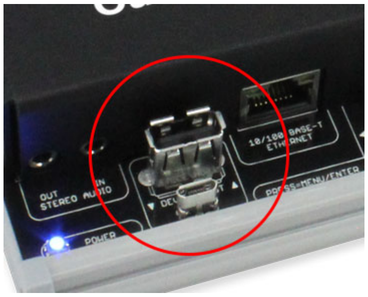
- S8
- U8
- S16_LE
- S16_BE
- U16_LE
- U16_BE
- S24_LE
- S24_BE
- U24_LE
- U24_BE
- S32_LE
- S32_BE
- U32_LE
- U32_BE
- FLOAT_LE
- FLOAT_BE
- FLOAT64_LE
- FLOAT64_BE
- IEC958_SUBFRAME_LE
- IEC958_SUBFRAME_BE
- MU_LAW
- A_LAW
- IMA_ADPCM
- MPEG
- GSM
- SPECIAL
- S24_3LE
- S24_3BE
- U24_3LE
- U24_3BE
- S20_3LE
- S20_3BE
- U20_3LE
- U20_3BE
- S18_3LE
- S18_3BE
- U18_3LE

USB Ports

CueServer 2 has both USB Host and USB Device ports.

At this time, the USB Host port is only used as an alternative way to apply firmware updates to the device. The USB Device port is not used and is reserved for future use.

Specifications

	CS-900	CS-920	CS-940/950
USB Host Ports	1 (Type A)	1 (Type A)	1 (Type A)
USB Device Ports	1 (Micro B)	1 (Micro B)	1 (Micro B)
			

LCD Display

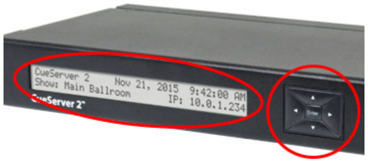
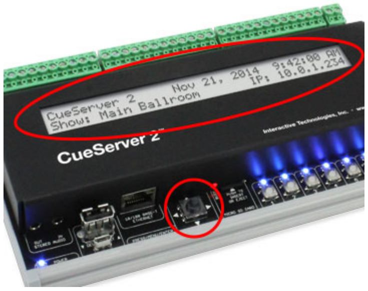
Some CueServer 2 models have a front-panel LCD Display with navigation buttons. This display is used to see the overall operational status of the device and can be used to adjust a small subset of settings and perform basic diagnostics.

The LCD Display features include:

- System Status (Default)
- Main Menu
    - Execute Macros
    - Change the Active Show
    - Adjust the Clock
    - Adjust Network Settings
    - View DMX Channels
    - Display System Information
    - Perform a Self Test

See the LCD Display Modes section for a complete description of the LCD functions available.

Specifications

	CS-900	CS-920	CS-940/950
LCD Display	2-Line x 40 Character Black on White	n/a	2-Line x 40 Character Black on White
Navigation	5-Way Backlit Button Pad	n/a	5-Way Micro Joystick
	 CueServer 2 Show: Main Ballroom Nov 21, 2015 9:42:00 AM IP: 10.0.1.234 CueServer 2		 CueServer 2 Show: Main Ballroom Nov 21, 2014 9:42:00 AM IP: 10.0.1.234 CueServer 2 Interactive Technologies, Inc.

LCD Display Modes

The LCD Display has two modes of operation, the Status Display and the Menu Display:

Status Display

By default, while the CueServer is running normally, the Status Display will be visible. This display typically shows the device’s name, the current time, and the device’s IP address. Although this information is typically shown, the Status Display is entirely customizable by the active show, so the Status Display might appear differently.

```
CueServer 2      Jan 1, 2016  9:42:00 AM
                             IP: 10.0.1.5
```

For a list of available Status Display customizations, see the LCD Status Options section.

Menu Display

By pressing the **Enter** button of the navigation switch, the LCD will change to it’s Menu Display. This display will show a list of **Main Menu** choices. Use the **Up** and **Down** buttons to scroll through the list of available choices. To activate a choice, press **Enter** or **Right**. To return to the previous display, press the **Left** button.

```
Main > [1] Macros
       [2] Shows
       [3] Clock Settings
       [4] Network Settings
       [5] DMX Menu
       [6] System Information
       [7] Self Test
```

See the LCD Menu Functions section for details about each available menu function.

LCD Status Options

The Status Display of the LCD is divided into four quadrants, the top-left, top-right, bottom-left, and bottom-right. Each of these four quadrants can be customized to show a different piece of information about the status of the device or show.

The default Status Display for CueServer 2 has the following quadrant layout:

Device Name	**Long Date + 12-Hour Time**
Show Name	**IP Address**

The following table shows the available options and how they appear on the LCD display:

Status Type	Example	Description
Device Name	`CueServer 2`	The assigned name of the device.
Show Name	`My First Show`	The name of the active show.
Show Directory	`/shows/My First Show/`	The file system directory of the active show.
IP Address	`IP: 10.0.1.234`	The IP Address of the device. If the device is configured with multiple network LANs, the display will alternate between both networks' IP Addresses. If the Ethernet jack is unplugged, a ∅ symbol will appear in front of the address.
Timecode	`TC: 00:00:00:00`	The current timecode within the system.
IO Status	`[C:--*--*--][O:-----**-]`	The current status of the built-in contact closures and digital outputs. A `*` symbol indicates that a contact is closed or an output is active.
CPU Load	`CPU: 7%`	The current 5-minute average CPU load percentage.
Long Date + 12-Hour Time	`Feb 15, 2017 10:34:56 PM`	The long date and 12-hour time combined.
Short Date + 12-Hour Time	`2/15/17 10:34:56 PM`	The short date and 12-hour time combined.
Long Date + 24-Hour Time	`Feb 15, 2017 22:34:56`	The long date and 24-hour time combined.
Short Date + 24-Hour Time	`2/15/17 22:34:56`	The short date and 24-hour time combined.
Long Date	`Feb 15, 2017`	The long date.

Short Date	`2/15/17`	The short date.
12-Hour Time	`10:34:56 PM`	The 12-hour time.
24-Hour Time	`22:34:56`	The 24-hour time.

These settings can be changed by choosing the *Settings > LCD Display* from within CueServer Studio.

LCD Menu Functions

The Main Menu of the LCD appears when the **Enter** or **Right** button of the navigation switch are pressed. Use the **Up** and **Down** buttons to select a menu option, then press **Enter** or **Right** to choose the menu option. To exit a menu option, press **Left**.

The following table shows the **Main Menu** options:

Menu Option	Function
Macros	Displays a list of the available Macros in the current show that have the “Show in LCD Display” option selected. Any macro shown in the list can be activated.
Shows	Displays a list of the available Shows on the memory card. This menu also shows and allows the currently active show to be changed.
Clock Settings	Displays and allows adjustments to the current time and date, including time zone.
Network Settings	Displays and allows adjustments to the current network settings, including DHCP.
DMX Menu	Displays a list of DMX related functions.
System Information	Displays information about the hardware and firmware revisions, including license information.
Self Test	Runs the built-in hardware self test routine. Two confirmation screens will appear before allowing the self test to run. See the Self Test section of this manual for instructions. **Warning:** Executing the self test will interrupt the currently running show.

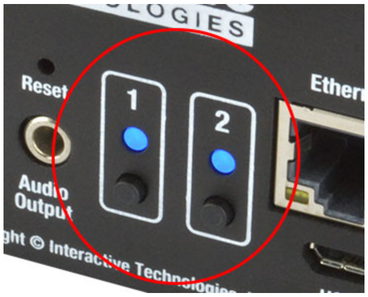
Function Buttons

CueServer 2 provides up to eight customizable front-panel function buttons. These buttons can be customized for the needs of a particular application, for example, they can run presets, start shows, change modes, show operating status, and more.

Each button can be programmed with any of the available CueScript actions and/or rules to automate any kind of action in CueServer. Each function button includes full RGB backlighting that is also controlled by the CueScript programming.

The CS-900 CueServer 2 Pro features removable button caps. Optional blank caps are available that allow for the insertion of printed transparency films to customize the legends for each button.

Specifications

	CS-900	CS-920	CS-940/950
Function Buttons	8	2	8
Backlighting	Full RGB	Full RGB	Full RGB
Replaceable Legends	Yes, Optional	No	No
			

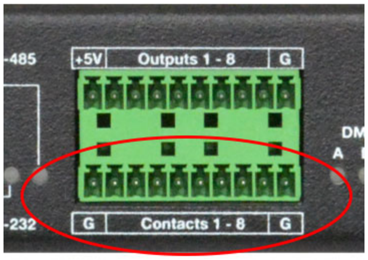
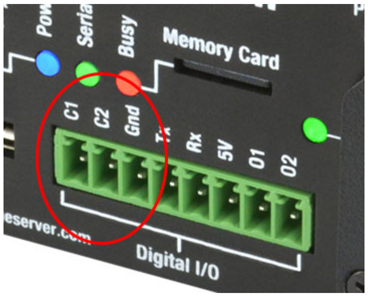
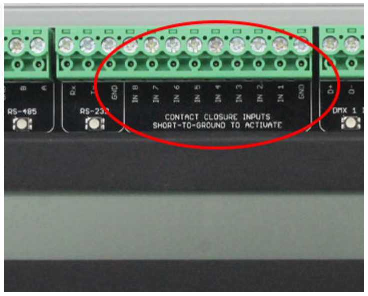
Contact Closures

CueServer 2 provides up to eight low-voltage contact-closure inputs. These inputs are designed for connecting switches, buttons, motion detectors, sensors, and most anything that makes an electrical connection between two conductors.

Each contact-closure input can be programmed with any of the available CueScript actions and/or rules to automate any kind of action in CueServer.

Each input floats up to a voltage around 3.3V DC through a weak pull-up resistor. When that input is shorted-to-ground, the CueServer device detects this drop in voltage as a “contact closure”.

Specifications

	CS-900	CS-920	CS-940/950
Contact Closure Inputs	8	2	8
Connector	10-Position Pluggable Terminal Block	8-Position Pluggable Terminal Block (shared with other I/O)	12-Position Pluggable Terminal Block (shared with RS-232 port)
			
Pinout	1 = Common (Ground) 2 = Contact 1 3 = Contact 2 4 = Contact 3 5 = Contact 4 6 = Contact 5 7 = Contact 6 8 = Contact 7 9 = Contact 8 10 = Common (Ground)	1 = Contact 1 2 = Contact 2 3 = Common (Ground) 4 = RS-232 Transmit 5 = RS-232 Receive 6 = Aux +5VDC Output 7 = Output 1 8 = Output 2	1 = Common (Ground) 2 = Contact 1 3 = Contact 2 4 = Contact 3 5 = Contact 4 6 = Contact 5 7 = Contact 6 8 = Contact 7 9 = Contact 8 10 = RS-232 Common 11 = RS-232 Transmit 12 = RS-232 Receive

Digital Outputs

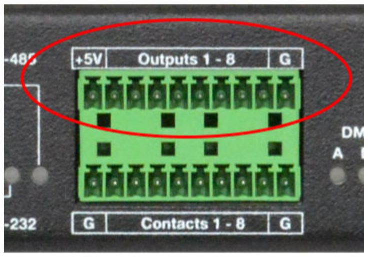
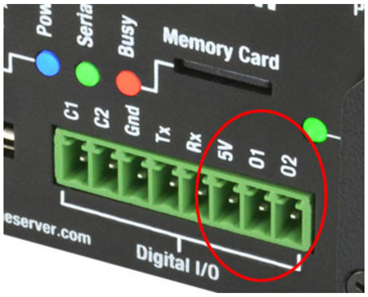
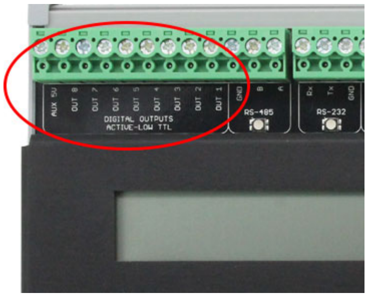
CueServer 2 provides up to eight low-voltage digital outputs. These outputs are designed for connecting LED indicators, small relays, buzzers, pilot lights, and most anything that can be powered from a small DC voltage.

Each digital output can be turned on or off as a response to any of the available CueScript actions and/or rules.

Each output is a TTL “short-to-ground” output. When the output is “on”, the corresponding pin is shorted-to-ground. When an output is “off”, the corresponding pin is an open connection. This means that the output is used to make a complete circuit through the connected accessory device (like an indicator or relay) to ground; the other side of the accessory needs to be connected to a positive voltage. Each digital output can handle a maximum of 500mA.

For convenience, an auxiliary 5VDC output is provided with the digital outputs. This voltage source can optionally be used to provide power to LED indicators, small relays, etc. The maximum current available at the *Aux +5VDC Output* is 200mA.

Specifications

	CS-900	CS-920	CS-940/950
Digital Outputs	8	2	8
Connector	10-Position Pluggable Terminal Block	8-Position Pluggable Terminal Block (shared with other I/O)	12-Position Pluggable Terminal Block (shared with RS-485 port)
	 -485 +5V Outputs 1 - 8 G DM A -232 G Contacts 1 - 8 G	 Pow Serial Busy Memory Card C1 C2 Gnd Tx Rx 5V O1 O2 eserver.com Digital I/O	
Pinout	1 = Aux +5VDC Output 2 = Output 1 3 = Output 2 4 = Output 3 5 = Output 4 6 = Output 5 7 = Output 6	1 = Contact 1 2 = Contact 2 3 = Common (Ground) 4 = RS-232 Transmit 5 = RS-232 Receive 6 = Aux +5VDC Output 7 = Output 1	1 = RS-485 “A” 2 = RS-485 “B” 3 = RS-485 Common 4 = Output 1 5 = Output 2 6 = Output 3 7 = Output 4

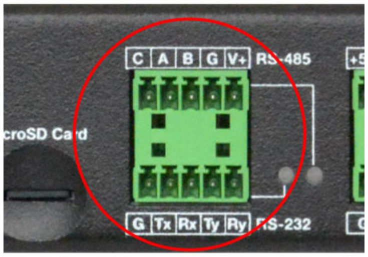
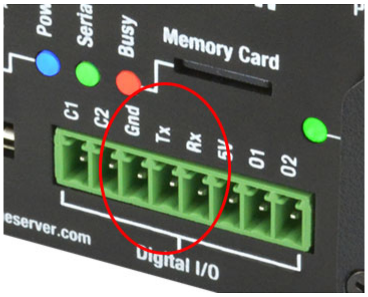
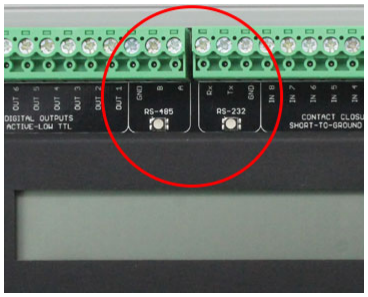
	8 = Output 7 9 = Output 8 10 = Common (Ground)	8 = Output 2	8 = Output 5 9 = Output 6 10 = Output 7 11 = Output 8 12 = Aux +5VDC Output

Serial Ports

CueServer 2 provides up to two serial ports, one RS-232 port and optionally one RS-485 port. These ports are designed to be used to interface with 3rd party devices such as video projectors, automation systems, security panels, motorized window coverings, and most anything else that has either an RS-232 or RS-485 serial interface.

Each serial port can be programmed to output strings of ASCII and/or binary data via CueScript actions and/or rules. Each serial port can be configured to receive and execute CueScript commands or to work with CueStation Hub protocol. Triggering on custom input strings has not been implemented yet.

Specifications

	CS-900	**CS-920**	**CS-940/950**
RS-232 Ports	1	1	1
RS-485 Ports	1	–	1
Connector	2× 5-Position Pluggable Terminal Blocks	8-Position Pluggable Terminal Block (shared with other I/O)	2× 12-Position Pluggable Terminal Blocks (shared with contact closures and digital outputs)
		 Pow Seria Busy Memory Card C1 C2 Gnd Tx Rx 5V O1 O2 eserver.com Digital I/O	 DIGITAL OUTPUTS ACTIVE-LOW TTL RS-485 RS-232 CONTACT CLOSU SHORT-TO-GROUND
Pinout	**Top Row** 1 = RS-485 Common 2 = RS-485 “A” 3 = RS-485 “B” 4 = Ground 5 = V+ **Bottom Row** 1 = RS-232 Common 2 = RS-232 Transmit 3 = RS-232 Receive 4 = Aux Transmit (Do Not	1 = Contact 1 2 = Contact 2 3 = Common (Ground) 4 = RS-232 Transmit 5 = RS-232 Receive 6 = Aux +5VDC Output 7 = Output 1 8 = Output 2	**Contact Closure Terminals** 1-9 = Contact Closures 10 = RS-232 Common 11 = RS-232 Transmit 12 = RS-232 Receive **Digital Output Terminals** 1 = RS-485 “A” 2 = RS-485 “B” 3 = RS-485 Common 4-12 = Digital Outputs

	Use) 5 = Aux Receive (Do Not Use)		

Indicators

Color & Pattern	Description
● Off	No serial data input or output
🟢 Quickly Flashing Green	Transmitting and/or receiving data
🟡 Quickly Flashing Yellow	Received unexpected data (possibly bad protocol)
🔴 Quickly Flashing Red	Received poorly framed data bytes (possibly wrong baud rate, or breaks in data)

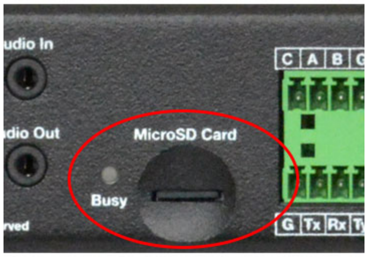
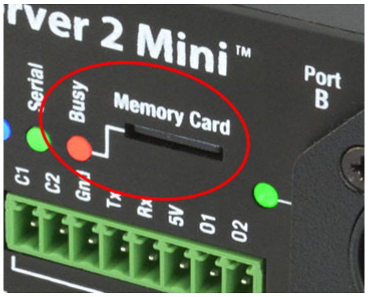
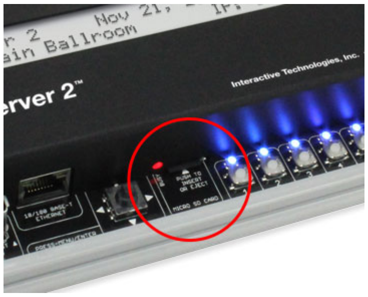
Memory Card

CueServer 2 uses a *microSD* memory card for storage of show files. Units ship from the factory with “Class 10” 16GB cards pre-installed.

At this time, CueServer 2 does not support hot-swapping cards while the system is running. This means that a card must be inserted when the device is turned on, and must remain inserted while it is running. To switch cards, please power the device off before changing cards.

! We have received reports of CueServer 2 show failures on units that have Class 4 or lower cards. All CueServer 2 units ship from the factory with Class 10 (or higher) cards installed. If you plan to use your own card, please make sure that it is Class 10 (or higher) as specified below.

Specifications

	CS-900	CS-920	CS-940/950
Card Type	microSD	microSD	microSD
File System Format	SDHC/FAT32	SDHC/FAT32	SDHC/FAT32
Speed Class	Class 10 (or higher) or UHS Class 1 (U1) (or higher)	Class 10 (or higher) or UHS Class 1 (U1) (or higher)	Class 10 (or higher) or UHS Class 1 (U1) (or higher)
Maximum Card Size	2TB	2TB	2TB
	 udio In C A B G dio Out MicroSD Card Busy G Tx Rx T	 rver 2 Mini™ Port B Serial Busy Memory Card C1 C2 Gnd Tx Rx 5V 01 02	 Nov 21, ain Ballroom rver 2™ Interactive Technologies, Inc.

Indicators

Color & Pattern	Description
● Off	No card inserted

🔴 Solid Red	Card is mounted and in use by system (do not remove)
🔴 Quickly Flashing Red	Card did not mount properly

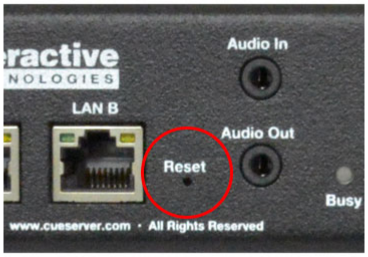
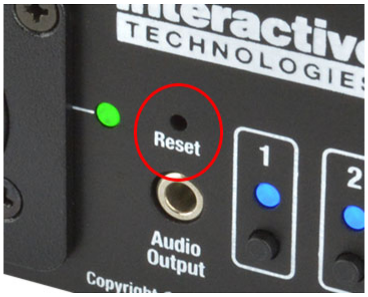
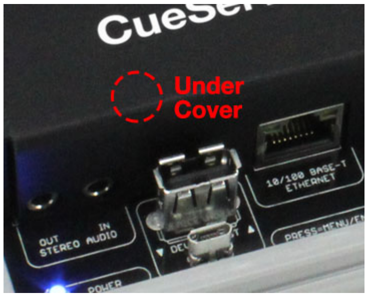
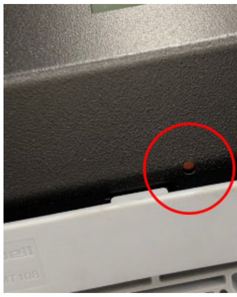
Reset Button

All CueServer 2 models have a “Reset” button. Most models (except for the older CS-940) have a small “pinhole” to access this button with a paperclip or small screwdriver.

The Reset Button is used for multiple maintenance purposes. It can be used to activate the built-in self-test, clear the user password, reset network settings, perform a full factory reset, and enter a special boatloader mode.

See the sections below for detailed instructions for using the Reset Button.

Specifications

	CS-900	CS-920	CS-940	CS-950
Location	“Pinhole” on Rear Panel	“Pinhole” on Front Panel	Internal, Under Cover	“Pinhole” on Right
	 ractive NOLOGIES LAN B Audio In Audio Out Reset Busy www.cueserver.com • All Rights Reserved	 nteractiv TECHNOLOGIES Reset Audio Output 1 2 Copyrigh	 CueSe Under Cover 10/100 BASE-T ETHERNET OUT STEREO IN AUDIO POWER PRESS-MENU	

Entering Bootloader Mode

To enter the special bootloader mode, start with CueServer powered off. When applying power, hold down the Reset Button. Instead of the Power LED blinking Blue, it will blink a Magenta color. This bootloader mode is for factory use only. There is no need to attempt to use bootloader mode except as instructed by Technical Support. If bootloader mode is entered accidentally, simply remove and reapply power without holding down the Reset Button.

Activating Boot Options

To use the Reset Button to activate one of the special boot options, follow these steps:

1. Start with CueServer’s power off.
2. Apply power *without* holding down the Reset Button.
3. After the Power LED begins flashing Blue, press and hold the Reset Button. Note: If you press the Reset Button too soon, the Power LED will flash Magenta which indicates it has entered bootloader

mode. If this happens, simply power off and try these steps again.

4. After approximately 20 seconds, the Power LED will turn off and the LCD Display will show `[RESET] Button Pressed`.
5. Continue to hold the Reset Button down.
6. Every 2 seconds, the Power LED will change colors and the LCD Display will show a different “boot option”. The options and their LED colors are shown in the table below.
7. When the option you want to activate appears, release the button.
8. The display will change to `Press & Release [RESET] to Confirm` and the Power LED will begin flashing rapidly.
9. If you want to activate the chosen option, press and release the Reset Button again to confirm within 5 seconds.
10. If anything goes wrong (you selected the wrong option, or missed the opportunity to choose the desired option), simply don’t press the Reset Button again. The selection of a boot option will timeout and the device will resume starting up normally. Or, you can remove power at any time to try again.

Boot Options and their corresponding LED Colors

As the boot option menu has been activated at startup time, the following LCD Displays and Power LED colors will appear:

Color & Pattern	LCD Display	Description
☀ Slowly Flashing Blue	`Welcome to CueServer`	Device is in the process of starting up, hold down the Reset Button during this time.
● Off	`[RESET] Button Pressed`	Approximately 20 seconds after power-on, if the Reset Button is held, the Power LED will turn off for 1.5 seconds while the LCD indicates that reset options will appear shortly.
● Green	`Start Self Test`	Release the Reset Button to choose to enter the self-test mode.
● Magenta	`Skip Loading Show File`	Release the Reset Button to choose to skip loading the currently active show file during startup.
● Yellow	`Clear User Password`	Release the Reset Button to choose to clear the user password from the device.
● Cyan	`Clear Network Settings`	Release the Reset Button to choose to clear the network settings back to factory defaults.

🔴 Red	`Factory Reset`	Release the Reset Button to choose to perform a full factory reset on the device (all show data will be preserved).

❗ For models without an LCD Display, the user has to rely solely on the appearance of the Power LED.

❗ The LED colors on the CS-940 are different. Because the CS-940 hardware is not capable of generating a full range of colors on the Power LED, each of the boot options appear in Red color. Use the LCD Display to choose an appropriate option, or count the number of Red flashes to choose an option.

Self-Test Function

CueServer 2 has a built-in Self-Test function that tests nearly every subsystem and circuit path in the product. This function can be used if there is a suspicion that the CueServer hardware has a physical fault.

> ! Do not enter the Self-Test mode while a show is in progress. The show will be halted and the DMX output from the device will switch to a test pattern. Also, the only way to exit the Self-Test mode is to power-cycle the device.

This function can be accessed by selecting the “Self Test” menu item from the LCD Display. To enter the Self-Test mode, press the **Enter** button to show the main menu, then scroll down to the “Self Test” item and press **Enter** again. A confirmation dialog will appear on the LCD screen. Move the cursor to the right and press **Enter** again. A second confirmation dialog will appear. Again, move the cursor to the right and press **Enter**. These confirmations appear because the Self-Test function causes the CueServer to halt any currently running show.

When the Self-Test starts, a display similar to the following will appear on the LCD:

```
KEYS | DI | BUS | RTC | DMX | SER |
---- | -- | OK  | OK  | OK  | OK  |
```

While the Self-Test is running, the following functions are performed:

- Continuous display of front-panel switch inputs
- Continuous display of contact closure inputs
- Display of system bus status
- Display of real-time clock status
- Display of DMX loopback test
- Display of Serial Port loopback test
- PCB Indicator test
- Digital Output test

The following sections describe each of these tests and the display in detail.

Indicator and Digital Output Test

While in the Self-Test mode, all of the indicators on the device illuminate and slowly cycle through various colors.

Function Button Indicators

Since the function button indicators have the capability of illuminating in full 24-bit RGB colors, they will demonstrate this by slowly crossfading through the entire RGB spectrum in order: Red, Yellow, Green, Cyan, Blue, Magenta. If any of the button indicators is faulty, they will not match the color of the other button indicators.

General Purpose Indicators
The remaining indicators on the product can only illuminate in primary colors. As the Self-Test is running, they will periodically jump from color to color. On the CS-900, the indicators will show six colors (Red, Yellow, Green, Cyan, Blue, Magenta). On the CS-940, all indicators except the power indicator will show three colors (Red, Yellow, Green) and the power indicator will show three different colors (Red, Magenta, Blue). If any of these patterns are not followed, then one of the indicators may be faulty.

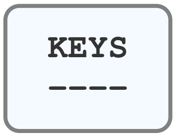
Digital Outputs
The eight digital outputs slowly cycle through one output being on at a time in order from 1 thru 8. If more than one output is on or if an output is skipped in the pattern, the output may be faulty.

Manual Testing Mode
When any of the front-panel buttons and/or contact closure inputs is pressed/closed, then the corresponding function button indicator will illuminate White and the corresponding digital output will turn on and a corresponding general indicator will illuminate. In this mode, all other indicators and outputs will turn off. After the button or contact is released, the regular cycling pattern will be resumed after three seconds. This “manual” mode of testing is useful for diagnosing a broken switch or indicator.

The following table shows how each button, indicator, contact and output is connected in manual testing mode:

Function Button, Contact Closure, or Digital Output	Indicator (CS-900)	Indicator (CS-940)
1	DMX Port D	Power/Status
2	DMX Port C	RS-485
3	DMX Port B	RS-232
4	DMX Port A	Memory Card
5	RS-485	DMX 1 In
6	RS-232	DMX 2 In
7	Memory Card	DMX 1 Out
8	Power/Status	DMX 2 Out

Keyboard and LCD Display Test



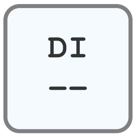
```
KEYS
----
```

The section of the display marked **KEYS** shows which keys on the front-panel are currently depressed. The following table shows the possible values that are shown for this portion of the test:

KEYS	Meaning
`----`	No keys are pressed. This is the normal “good” display when no keys are pressed.
`UP`	The navigation UP button is pressed. Pressing UP also dims the LCD display’s backlight to 50% to test software control of the backlight.
`DOWN`	The navigation DOWN button is pressed. Pressing DOWN also sets the LCD display’s contrast to 50% to test software control of the display contrast.
`RGHT`	The navigation RIGHT button is pressed. Pressing RIGHT also advances to the next self-test page, such as one of the DMX detail pages. Continue pressing LEFT and RIGHT to circle around the available self-test pages.
`LEFT`	The navigation LEFT button is pressed. Pressing LEFT also advances to the previous self-test page, such as one of the DMX detail pages. Continue pressing LEFT and RIGHT to circle around the available self-test pages.
`ENTR`	The navigation ENTER button is pressed. Pressing ENTER also clears any errors for the DMX and Serial loopback tests.
`1` .. `8`	One of the front-panel function buttons is pressed. The number of the button is displayed. Also, the indicator on above the button illuminates White and the corresponding digital output is turned on.
`XXXX`	A four-digit hexadecimal number will appear if more than one button is pressed simultaneously. The hexadecimal number represents the sum of individual “bits” that correspond to each button that is pressed. The following are the bit values of each button: 0001 = Up 0002 = Down 0004 = Right 0008 = Left 0010 = Enter 0100 = Function 1 0200 = Function 2 0400 = Function 3 0800 = Function 4 1000 = Function 5 2000 = Function 6

	4000 = Function 7 8000 = Function 8

Digital Input (Contact Closure) Test

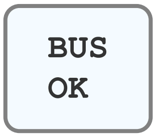


```
DI
--
```

The section of the display marked **DI** shows which digital inputs (contact closures) are currently closed. The following table shows the possible values that are shown for this portion of the test:

DI	Meaning
`--`	No contacts are closed. This is the normal “good” display when no contacts are closed.
`1` .. `8`	One of the contact closure inputs is closed. The number of the contact input is displayed. Also, the corresponding digital output is turned on and the indicator above the corresponding button illuminates White.
`XX`	A two-digit hexadecimal number will appear if more than one contact is closed simultaneously. The hexadecimal number represents the sum of individual “bits” that correspond to each contact that is closed. The following are the bit values of each button: 01 = Contact 1 02 = Contact 2 04 = Contact 3 08 = Contact 4 10 = Contact 5 20 = Contact 6 40 = Contact 7 80 = Contact 8

System Bus Test



```
BUS
OK
```

The section of the display marked **BUS** shows the result of testing the peripherals on the internal system bus. The following table shows the possible values that are shown for this portion of the test:

BUS	Meaning
OK	All tests have passed.
XXX	A three-digit hexadecimal number will flash if one or more system bus tests failed. The hexadecimal number represents the sum of individual “bits” that correspond to each failed test. The following are the bit values of each test: 001 = Board ID Error 002 = NVRAM Error 004 = LCD Contrast Error 008 = Digital IO Error 010 = Audio Codec Error 020 = Keyboard Switch Error 040 = Keyboard Indicator 1-4 Error 080 = Keyboard Indicator 5-8 Error 100 = PCB Indicator A Error 200 = PCB Indicator B Error 400 = Real-Time Clock Error

Real-Time Clock Test

```
RTC
OK
```

The section of the display marked **RTC** shows the result of testing the real-time clock circuitry. The following table shows the possible values that are shown for this portion of the test:

RTC	Meaning
OK	All tests have passed.
OSC	The clock oscillator has failed.
BAT	The clock backup battery has failed.

DMX Transceiver Loopback Test

```
DMX
OK
```

While the Self-Test is running, a DMX output test signal is generated by DMX Ports B & D (on the CS-900) and DMX 1/2 Outputs (on the CS-940). A loopback to a corresponding input port is used so the Self-Test function can verify that the data sent out of the output port is what is received by the input port. The following table shows which output port should be looped with which input port.

CS-900	From (Output)	To (Input)
Loopback 1	Port B	Port A
Loopback 2	Port D	Port C
CS-940	**From (Output)**	**To (Input)**
Loopback 1	DMX 1 Out	DMX 1 In
Loopback 2	DMX 2 Out	DMX 2 In

The section of the display marked **DMX** shows the result of testing the DMX Input/Output circuitry. The following table shows the possible values that are shown for this portion of the test:

DMX	Meaning
OK	All tests have passed.
1	DMX Loopback 1 have errors.
2	DMX Loopback 2 have errors.
1+2	Both DMX Loopbacks have errors.

Use the **RIGHT** and **LEFT** navigation buttons to display either the DMX 1 or DMX 2 pages on the LCD to view additional details about the DMX loopback tests.

The extra LCD pages that show detailed DMX loopback information appear similar to the following:

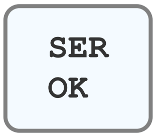
```
DMX | C1,2,3 | MIS | CHN | ST | DAT | OV
 1  | 000000 |   0 | 512 |  1 |   0 |  0
```

The number directly under **DMX** shows which loopback (1 or 2) that you are viewing. The hexadecimal six digits under C1,2,3 show the current DMX values for channels 1, 2 and 3 (these channels are copied throughout the entire 512 channels of the output). The value under **MIS** shows how many “missed” packets have been not read on the input that were sent on the output (which should be 0). The value under CHN shows how many DMX channels are being received (which should be 512). The value under ST shows how many unique start codes are being received (which should be 1). The value under DAT shows how many individual data errors have been detected (which should be 0). The value under OV shows how many packets received had more than 512 channels in it (which should be 0).

If any of the fields on this display counts up to a number higher than can be displayed, the field will show a *** to signify that more than 999 events were counted.

To reset all of the error counters back to zeros, press the **Enter** button on the navigation switch.

Serial Loopback Test



```
SER
OK
```

While the Self-Test is running, a test signal is generated at the RS-232 Tx pin. A loopback to the RS-232 Rx pin is used so the Self-Test function can verify that the data sent out of the port is what is received by the input. The following table shows which output port should be looped with which input port.

CS-900	From (Output)	To (Input)
Loopback	RS-232 Tx (Pin 2)	RS-232 Rx (Pin 3)
CS-940	**From (Output)**	**To (Input)**
Loopback	RS-232 Tx (Pin 11)	RS-232 Rx (Pin 12)

The section of the display marked **SER** shows the result of testing the Serial Port circuitry. The following table shows the possible values that are shown for this portion of the test:

SER	Meaning
OK	All tests have passed.
ERR	A serial port loopback test failed.

CueScript Language

CueServer uses a command language called ***CueScript*** as the basis of nearly all of CueServer's control and automation scripting capabilities. You will use CueScript to make CueServer perform actions. If you want 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, timers, 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 natural 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 – although you can type `Channel 1 At 100`, you can also type `C1A100`.

The following sections describe the language in more detail.

CueScript Overview

The following topics describe the details of the language:

- Executing Commands – how to submit CueScript commands for execution
- Command Syntax – all about the syntax for CueScript statements
- Expressions – snippets of CueScript used for logic commands
- Command Context – description of Command Contexts
- Levels – values given to channels
- Strings – using user-defined text
- Comments – adding comments to CueScript

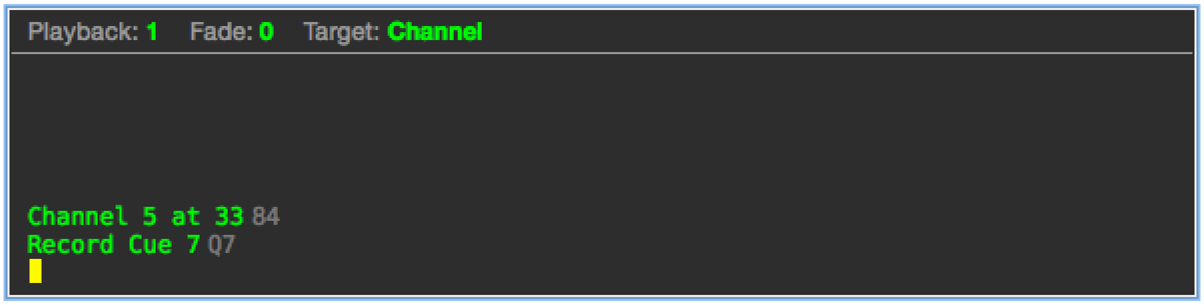
The specific commands available are detailed in the following sections:

- Selection Commands – for selecting various objects such as channels, buttons, fixtures, and universes
- Action Commands – for performing actions such as setting levels, executing cues, and activating presets
- Logic Commands – for adding logic to scripts such as if/then statements
- Effect Properties – a listing of properties related to effects
- System Variables – a listing of built-in system variables

Executing Commands

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

Command Line



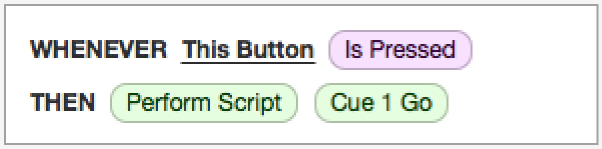
```
Playback: 1    Fade: 0    Target: Channel

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



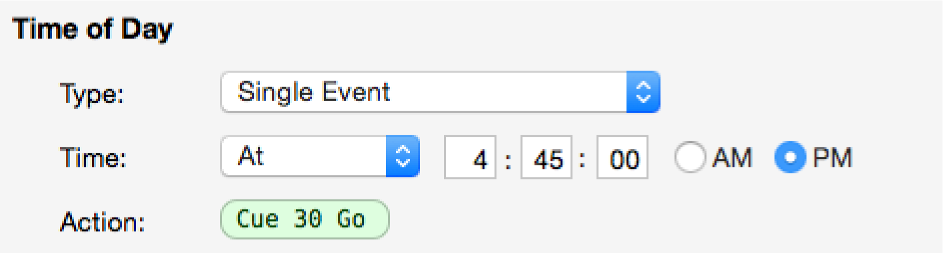
WHENEVER This Button Is Pressed
THEN Perform Script Cue 1 Go

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:

- CueScript via UDP
- CueScript via HTTP
- CueScript via Serial

CueScript via UDP

CueServer allows external CueScript commands to be sent to it via UDP packets.

There are two methods that can be used to send UDP packets to CueServer:

- **Unicast Method:** A UDP packet containing one or more CueScript commands can be *unicast* directly to the IP Address of the CueServer on port 52737. Using this method, only the specific CueServer sent the packet will execute the CueScript.

- **Multicast Method:** A UDP packet containing one or more CueScript commands can be *multicast* to the CueScript Group Address (239.255.204.2) on port 52737. Using this method, ***all*** CueServers on the local network will receive and execute the CueScript.

The contents of the UDP packet are simply the CueScript string that is to be executed by the CueServer.

Examples:

- `Cue 1 Go`
- `Button 1.5 On`
- `Macro 7`
- `P3CL; Q5G; B1ON`

CueScript via HTTP

CueServer allows external CueScript commands to be sent to it via HTTP protocol (a simple URL request).

Built into CueServer is a web server that allows CueScript commands to be executed by receiving them in a special URL. The typical format of this URL is:

```
http://<ip-of-CueServer>/exe.cgi?cmd=<command>&<optional-parameters>
```

For example, the following URL will execute the command `Cue 1 Go`:

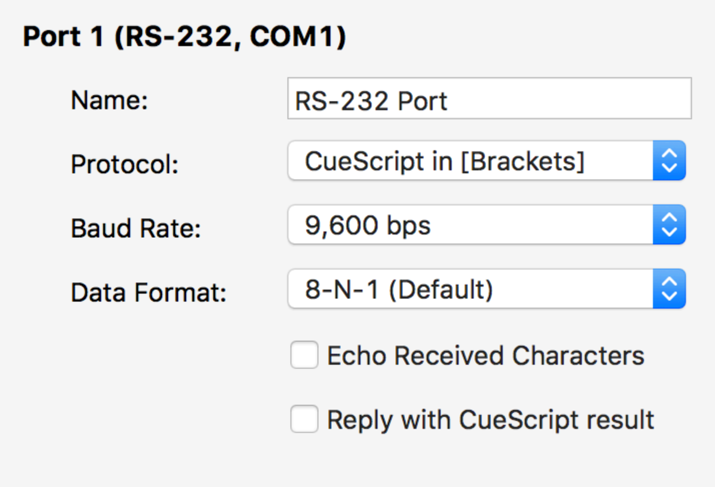
```
/exe.cgi?cmd=Cue+1+Go
```

Additional details about the `exe.cgi` URL is available in CGI API section of this manual.

CueScript via Serial

CueServer allows external CueScript commands to be sent to it via RS-232 and/or RS-485 serial strings.

A serial port on CueServer is configured by going into the *Stations* section and then editing the Built-In Station (Station 0). The serial ports available to CueServer will appear in the list of *Ports*.



Port 1 (RS-232, COM1)

Name: RS-232 Port

Protocol: CueScript in [Brackets]

Baud Rate: 9,600 bps

Data Format: 8-N-1 (Default)

□ Echo Received Characters

□ Reply with CueScript result

Other external station types that include serial ports would also be configured by choosing them in the *Stations* list and then clicking on one of the *Ports* that appears.

From the *Port Configuration* panel that appears there are two protocols that can be chosen that allow CueScript commands to be sent to the CueServer via a serial port:

- **CueScript in [Brackets]:** When this protocol is chosen, a CueScript string can be sent to this serial port as long as it is enclosed by “square brackets”. For example, `[Cue 1 Go]`. Any characters received outside of the brackets will be ignored. After the opening bracket, the command is received and buffered until a closing bracket is received. As soon as the closing bracket is received, the command string is executed. If a command is being received and then another open-bracket is received, then any accumulated characters in the receive buffer are cleared and a new string will begin to be received.

- **CueScript with CR/LF:** When this protocol is chosen, a CueScript string can be sent to this serial port with a terminating carriage-return (0×0D) and/or line-feed (0×0A). For example, `Cue 1 Go`, followed by either a 0×0D or 0×0A (or both) characters. All characters received on the serial port will be accumulated until a carriage-return or line feed is received. As soon as one of those terminating characters are received, the command string is executed.

The Baud Rate and Data Format selected for the serial port must match that of the transmitting device for the CueScript commands to be received properly.

If the “Echo Received Characters” box is checked, then every character received by the port will be retransmitted (“echoed”) back to the sender.

If the “Reply with CueScript result” box is checked, then the result of the CueScript string will be transmitted back to the sender.

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 specify a fade time and set a DMX channel to 50%, the command `Time 5 Channel 3 at 50` is used.

White spaces in a command (spaces, tabs, new lines, etc.) are ignored by CueScript 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 abbreviation 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 demonstrate 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

Concatenation Operator

The following operator performs its function on strings. Shared with the Addition Operator, if either side of the “+” is a string, the result will be a string:

Operator	Function	Example	Result
+	Concatenation	“Cue” + “Server” “CS” + 2 3 + “ is a crowd”	“CueServer” “CS2” “3 is a crowd”

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
==	Equal	5 == 5 3 == 5	1 0
!=	Not Equal	5 != 5 3 != 5	0 1

>	Greater Than	5 > 3 3 > 5	1 0
>=	Greater Than or Equal	5 >= 3 4 >= 4 2 >= 7	1 1 0
<	Less Than	3 < 5 5 < 3	1 0
<=	Less Than or Equal	3 <= 5 4 <= 4 7 <= 2	1 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`	Sets variable *x* to the number *3*
`"MyVariable" = 42`	Sets variable *MyVariable* to the number *42*
`"Message" = "Hello World"`	Sets variable *Message* to the string *“Hello World”*
`"y" = ('x' + 3)`	Sets variable *y* to the result of the expression *‘x’* + *3*
`"caption" = ("Press " + 'y' + " to Start")`	Sets variable *caption* to the string *Press 6 to Start*
`"caption" = "Press ${y} to Start"`	Sets variable *caption* to the string *Press 6 to Start*

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

`Set x 3`	Sets variable *x* to the number *3*
`Set MyVariable 42`	Sets variable *MyVariable* to the number *42*
`Set Message "Hello World"`	Sets variable *Message* to the string *“Hello World”*
`Set y ('x' + 3)`	Sets variable *y* to the result of the expression *‘x’* + *3*
`Set caption ("Press " + 'y' + " to Start")`	Sets variable *caption* to the string *Press 6 to Start*
`Set caption "Press ${y} to Start"`	Sets variable *caption* to the string *Press 6 to Start*

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
`Log 'y'`	Writes “6” to the System Log
`Write COM1 'caption'`	Sends “Press 6 to Start” to the RS-232 port

Using Variable Values as Commands

To use variables values in CueScript as commands, enclose the variable name in accent quotes (`` `myCommand` ``).

The following example script shows how to assign a string that contains valid CueScript text to the variable *myCommand*. On the second line of script, if the variable *x* is greater than 3, then the commands in *myCommand* will be executed.

```
"myCommand" = "Cue 1 Go"
If ('x' > 3) Then `myCommand`
```

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`.

The following example changes the brightness of the front-panel LCD display to 50%:

```
Set lcd.backlight 50
```

See the section on System Variables for a detailed listing of available system variables and how they are used.

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, the current cue stack, the zone, the active station 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 commands 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”).

If the value of a 16-bit channel is being set, the percentage values from 0 to 100 are still used, and CueScript will set the value of the channel appropriately.

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, unless you are specifically setting the value of a 16-bit channel where the range of values is from 0 to 65535.

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, unless you are specifically setting the value of a 16-bit channel where the range of values is from 0000 to FFFF.

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 `At 0` and `At 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.

Strings

CueScript commands frequently contain *strings*. A string is a series of zero or more alpha-numeric characters enclosed in “double-quotes”.

Examples of strings include: “Hello World”, “Press Stop to Cancel Show Playback”, “button.onColor”, and “My First Show”.

Examples of commands that use strings are AUDIO, LOAD, LOG, SET, STACK, and WRITE.

Variable Substitution

Sometimes it is necessary to include the value of a variable within a string. In CueScript, a variable’s value can be *substituted* into a string by using a dollar sign (`$`) followed by the variable name enclosed in curly-brackets. For example:

```
"Cue ${myCue} Is Running"
```

In this example, the value of the variable named *myCue* will be inserted into the string. If *myCue* is equal to “3”, then the string will read “Cue 3 Is Running”.

This syntax can be used anywhere a string is used in CueScript. For example, with the **Write** command:

```
Write COM1 "Timer ${whichTimer} is enabled"
```

Any user-defined or system-defined variable may be used in strings using this technique.

Special Characters

Sometimes it is necessary to enter special characters that are non-printable or difficult to enter into a string from the keyboard. Examples include carriage returns, linefeeds, tabs, quotation marks, or special hexadecimal characters such as NULL (0×00).

CueServer allows special characters to be entered into strings using *escape sequences* that start with the backslash character (\) followed by a single letter that designates the specific escape character desired. For example, the escape sequence “\n” becomes a new-line character.

Because the backslash is used to mean “escape”, a single backslash can’t be used to put a backslash into a string. If a backslash is needed, use the escape sequence for backslash which is a double-backslash (\\).

The following table shows the supported escape sequences:

Escape Sequence	Hex Value	Character Represented
`\a`	`0x07`	Alarm/Bell
`\b`	`0x08`	Backspace
`\f`	`0x0C`	Formfeed
`\n`	`0x0A`	Newline
`\r`	`0x0D`	Carriage Return
`\t`	`0x09`	Horizontal Tab
`\v`	`0x0B`	Vertical Tab
`\\`	`0x5C`	Backslash
`\'`	`0x27`	Single quotation mark
`\"`	`0x22`	Double quotation mark
`\xhh`	`0xhh`	Any hexadecimal byte
`\nnn`	`0xoo`	Any octal byte

Note that there are two special escape sequences for hexadecimal and octal bytes. The hexadecimal escape sequence is a “backslash-x” followed by exactly two hexadecimal characters (each from 0 thru F) that represents the desired byte. The octal escape sequence is a backslash followed by three digits from 0 thru 7. These three digits represent the desired byte value in octal.

The escape sequences used by CueServer are the same (or very similar) as those used by several popular programming languages, including C, C++, Java, JavaScript, JSON, Objective-C, PHP, Python, and SQL.

Value Substitutions

In addition to the *escape sequences* above that allow special characters to be inserted into a string, CueServer also supports a variety of escape sequences that are used to substitute special *values* into a string.

These values change depending on the context in which the string is being used. CueServer will substitute the escape sequence with the actual value at the time that the string is being used. Each substitution is only valid within the context(s) supported.

The following table shows the supported escape sequences for Value Substitutions:

Escape Sequence	Value Substituted	Valid Contexts
`\c`	Channel Number (8-bit)	DMX Triggers
`\C`	Channel Number (16-bit)	DMX Triggers
`\l`	Channel Value (8-bit)	DMX Triggers
`\L`	Channel Value (16-bit)	DMX Triggers
`\i`	Inverted Channel Value (8-bit)	DMX Triggers
`\I`	Inverted Channel Value (16-bit)	DMX Triggers
`\s`	Checksum; sum of preceding bytes (8-bit)	Any
`\S`	Checksum; sum of preceding bytes (16-bit)	Any
`\*`	Reset checksum to zero	Any

By default, each of the values above will be substituted as a binary value. Additional modifiers may be placed between the backslash and the character to change which character format is used to output the value. Supported modifiers are listed below:

Modifier	Result
`$`	The value will be output in hexadecimal ASCII characters (0-F). 8-bit values will output exactly 2 characters and 16-values will output exactly 4 characters.
`#`	The value will be output in decimal ASCII characters (0-9). 8-bit values can range from 0-255 and 16-bit values can range from 0-65535.
`%`	The value will be scaled to a percentage (0-100) output in decimal ASCII characters (0-9)

Examples

```
Write COM1 "Hello World\n"
```

Writes the string *Hello World* followed by a newline to the COM1 serial port.

```
Write COM1 "One\tTwo\Three\tFour"
```

Writes the strings *One*, *Two*, *Three*, and *Four* with tabs in-between each string to the COM1 serial port.

```
Write COM1 "Press \"Start\" to Begin\x00"
```

Writes the string *Press “Start” to Begin* followed by a NULL byte to the COM1 serial port.

```
Log "Channel \#C is set to \#l"
```

Adds a system log message with a string such as “Channel 701 is set to 255”.

> `Write "10.0.1.5" "5AA5\$C80\$l\$S"`
> Sends a UDP packet to 10.0.1.5 with a string such as “5AA502BD80FF033D”, assuming that the channel is 701 and the level is 255.

Comments

Comments can be included in CueScript. A comment is a human-readable explanation or annotation in the source code of a block of CueScript. The following CueScript code includes two different types of comments:

```
/* Start the Evening Show in Playback 1
While also clearing any overrides in Playback 2 */
Playback 2; Clear
Playback 1; Cue 1 Go // Evening Show

// Set our show variable to keep track
Set currentShow "Evening"
```

The following sections describe the two comment types supported by CueScript in detail.

Block Comments

CueScript can use *block comments* similarly to JavaScript and the C programming languages. When a *Block Comment Start* is encountered, which is a slash followed by an asterisk `/*`, CueScript begins ignoring any text it finds until it encounters a *Block Comment End*, which is an asterisk followed by a slash `*/`.

The block comment can begin anywhere on a line and end on the same line or many lines later.

Here is an example of a multi-line block comment:

```
/*
 This is a comment that is ignored by CueScript.

These are handy to be able to add lots of text to the body of a CueScript code block.
*/
```

Line Comments

CueScript can also use *line comments* similarly to JavaScript and the C programming language. A line comment starts with a double-slash `//` and ends at the end of the current line. Any time CueScript encounters a line comment, it ignores any text it finds until it encounters the next line of code.

The line comment can begin anywhere on a line and it will end as soon as the end of the line is reached.

Here’s an example of several line comments:

```
// Setup the main playback
Playback 1; Clear
Set playback.mode Merge // Standard mode

// Setup the playback used for dimming
Playback 2; Clear
Group 50 At FL // Start with all dimmers at Full
Set playback.mode Scale // For scaling dimmers
```

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 Clear` or `Button ;`	Deselect all buttons	`0`
`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.

Deselecting All Buttons

Using either **Button Clear** or **Button ;** will deselect all buttons while leaving “Button” as the current command target.

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 <universe>.<local> [<range...>]`	Select one or more DMX channels	The selected channels' value
`Channel Clear` or `Channel ;`	Deselect all channels	`0`
`Channel ?`	Return the currently selected DMX channels	A selection string

- `<number>`
    - A global channel number from `1` to `16384`.
    - This number addresses channels as a continuous block through *all* configured universes.
- `<universe>.<local>`
    - A channel number in "dotted" notation specifies both a universe number from `1` to `128` and a channel from `1` to `512` within that universe.
    - This dotted number addresses channels locally within *each* universe.

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.

Using Global vs. Local Channel Numbers

The **Channel** command can use either global or local channel numbers. Global channel numbers sequentially number every channel used by all universes in sequential order (typically from `1` up to `16384`). Local channel numbers restart at 1 for each universe and are denoted in a . format.

For example, `Channel 3.1` refers to the first channel of universe 3. In global channel numbering (assuming that universes 1 & 2 both have 512 channels each) the same channel would be referred to as `Channel 1025`.

Deselecting All Channels

Using either **Channel Clear** or **Channel ;** will deselect all channels while leaving "Channel" as the current command target.

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 7.1 at FL`
>
> Sets channel 1 of universe 7 to 100%.

> `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 Clear` or `Contact ;`	Deselect all contacts	`0`
`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.

Deselecting All Contacts

Using either **Contact Clear** or **Comtact ;** will deselect all contacts while leaving “Contact” as the current command target.

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**

Control

Syntax

Command	Description	Return Value
`Control <number> [<range...>]`	Select one or more Shared Controls	The pressed state of the selected control(s)
`Control Clear` or `Control ;`	Deselect all Shared Controls	`0`
`Control ?`	Return the current selection	A selection string

Abbreviation

`K`

Description

Selecting Shared Controls

The **Control** command selects one or more Shared Controls in the system. Shared Controls are generic triggers that operate like buttons or contacts and can be linked to physical buttons or contacts. Shared Controls are useful for creating behavior and/or logic in a single place and then having multiple physical buttons or contacts “point” to the Shared Control.

Use the **Control** command in conjunction with an action command like **At**, **On**, **Off**, **Set**, **Enable** or **Disable** to change the properties of one or more Shared Controls. When used alone or in logic expressions, the **Control** command returns the current state of the specified control(s).

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

The wildcard character `*` can be used as the control number to mean *all* controls.

Deselecting All Controls

Using either **Control Clear** or **Control ;** will deselect all shared control while leaving “Control” as the current command target.

Determining Which Shared Controls 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 controls are selected, `0` will be returned.

Examples

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

> ```
> Control 1>5 On
> ```
> Turns the LED indicators of Shared Controls 1 thru 5 on.

> ```
> Control 1>3+5>8 Off
> ```
> Turns the LED indicators of Shared Controls 1 through 3 and 5 through 8 off.

> ```
> Control 3>5 Enable
> ```
> Enables Shared Controls 3 through 5.

> ```
> Control 1
> Set Button.OnColor {100,50,0}
> Set Button.Flash 4
> On
> Disable
> ```
> Selects Shared Control 1, 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.

> ```
> Control ?
> ```
> Returns the current Shared Control 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**

Effect

Syntax

Command	Description	Return Value
`Effect <number> [<range...>]`	Select one or more effects in the active playback fader	`0`
`Effect Clear` or `Effect ;`	Deselect all effects	`0`
`Effect ?`	Return the current selection	A selection string

- `<number>`
  - An effect slot from `1` to `4`.

Abbreviation

`EF`

Description

Selecting Effects

The **Effect** command selects one or more effects in the active playback fader. Effects are dynamic overlays applied to each playback fader that can modify channel values in real-time. Each Playback Fader has four independent effect slots. Effects can modify the color, intensity, position, and other parameters of channels and/or fixtures.

Setting Effect Properties

Once an effect is selected using the **Effect** command, properties of effects can be changed using the **Set** or assignment commands. The following example demonstrates how a playback's effects can be manipulated using CueScript.

```
Playback 1
Effect 2
"effect.type" = 1 // Set type to Hue Rotate
"effect.rate" = 0.5 // Set rate to 0.5 seconds
```

For a detailed listing of the various effect properties that can be manipulated using CueScript, refer to the Effect Properties topic.

Enabling/Disabling Effects

Effects can be enabled or disabled individually using the **Enable** or **Disable** commands. A disabled effect does not contribute to the output of a playback fader.

Deselecting All Effects

Using either **Effect Clear** or **Effect ;** will deselect all effects while leaving “Effect” as the current command target.

Determining Which Effects 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 `1>4`), or if no effects are selected, `0` will be returned.

Examples

> `Effect 1`
> Selects effect 1 in the active playback.

> `Effect 1>4 Disable`
> Disables the contribution of effects 1 through 4.

See Also

- Effect Properties
- **Disable**, **Enable**

Fixture

Syntax

Command	Description	Return Value
`Fixture <number> [<range...>]`	Select one or more DMX fixtures	`0`
`Fixture Clear` or `Fixture ;`	Deselect all fixtures	`0`
`Fixture ?`	Return the currently selected DMX channels	A selection string

- `<number>`
  - A fixture number from `1` to `16384`.

Abbreviation

`F`

Description

Selecting Fixtures

The **Fixture** command selects one or more DMX fixtures in the currently active playback fader. A DMX fixture is a group of one or more control channels *patched* to represent the operation of a single controllable lighting fixture. Use the **Fixture** command to select fixtures. Once one or more fixtures are selected, use action commands such as **At**, **On**, **Off**, **Enable**, **Disable**, **Park**, **Unpark** or **Release** to set the fixtures' intensity level, change the enable or parked state of the fixtures' channels, or release them.

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

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

Deselecting All Fixtures

Using either **Fixture Clear** or **Fixture ;** will deselect all fixtures while leaving “Fixture” as the current command target.

Determining Which Channels Are Selected

The question mark `?` can be used to ask what the current *channel* 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

> `Fixture 1`
> Selects fixture 1. Future action commands will be directed towards fixture 1.

> `Fixture 1>5 At 33`
> Sets the intensity channel(s) of fixtures 1 through 5 to 33%.

> `Fixture 1>3+5>8 On`
> Sets the intensity channel(a) of fixtures 1 through 3 and 5 through 8 to 100%.

> @Fixture 2+5 Park
> Parks all of the channels of fixtures 2 and 5.

> ```
> Fixture 100
> Time 0
> At 75
> Time 5
> At 0
> ```
> Selects fixture 100, then sets the fade time to 0 (immediate), then sets the intensity channel of fixture 100 to 75%, then sets the fade time to 5 (seconds), then sets the intensity of fixture 100 to 0%.

> `Fixture ?`
> Returns the current selection of channels for the selection of fixtures 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**

Group

Syntax

Command	Description	Return Value
`Group <number> [<range...>]`	Select one or more channel groups	The selected channels' value
`Group Clear` or `Group ;`	Deselect all groups	`0`

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 ~.

Deselecting All Groups

Using either **Group Clear** or **Group ;** will deselect all groups while leaving "Group" as the current command target.

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 Clear` or `Indicator ;`	Deselect all indicators	`0`
`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.

Deselecting All Indicators

Using either **Indicator Clear** or **Indicator ;** will deselect all indicators while leaving “Indicator” as the current command target.

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 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**

Live

Syntax

Command	Description	Return Value
`Live`	Change the active playback fader to “Live”	`0`

Abbreviation

`L`

Description

Changing the Active Playback to “Live”

The **Live** command changes the currently active playback fader to the special “Live” playback. The Live playback is a special playback layer that is inserted at the end of the regular playbacks that is used to make live edits to the current CueServer output. The Live playback always overrides the output of all other playbacks and can be quite useful when building new cue/preset content, or previewing cues/presets while using the Cue/Preset Editor.

The Live playback layer is always in override mode and does not have a submaster. Any channel values put into the live playback always override all other playback levels and the DMX input. The only channel priority that comes above Live are any channels that are parked.

Examples

> `Live`
> Changes the active playback to Live.

> `Live Clear`
> Changes the active playback to Live and clears the Live playback’s contents.

> `Live Cue 1 Go`
> Changes the active playback to Live and executes Cue 1 in this layer.

See Also

- **Playback**

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 Clear` or `Output ;`	Deselect all outputs	`0`
`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.

Deselecting All Outputs

Using either **Output Clear** or **Output ;** will deselect all outputs while leaving “Output” as the current command target.

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**

Page

Syntax

Command	Description	Return Value
`Page <number>`	Selects a page on the current station	The page number selected
`Page Clear` or `Page ;`	Deselect the current page	`0`
`Page ?`	Return the current page	The current page number

Abbreviation

`PA`

Description

The **Page** command selects one or more pages of the current station. Use the **Page** command in conjunction with action commands like **Lock** or **Unlock** to adjust the PIN authorization of a page.

✻ Looking for how to change the active page of a station? See the **At Page** command instead.

Locking/Unlocking a Page

If a page of a station requires a PIN number, that page cannot be accessed unless the user enters the correct PIN first. Once the PIN has been entered, that page (and all other pages on the same station that use the same PIN) are unlocked and can be viewed/operated freely. To *expire* that PIN and effectively re-lock the page, use the **Lock** command. For example: `Page 3 Lock`. This command will *expire* the PIN number and require the user to re-enter the PIN to regain access to the page.

On the other hand, if you would like to unlock a page from within CueScript (without the user needing to know the PIN number), use the **Unlock** command. For example: `Page 3 Unlock`. This command will *authenticate* the page with the PIN number allowing the user to view and operate the page.

Clearing the Current Page

In some instances it may be useful to not specify any particular page on the current station. To *deselect* the current page, you can use **Page Clear**.

Examples

> `Page 1 Lock`
> Expires the PIN on Page 1 of the current station. If the page requires a PIN number, the user will have to enter the correct PIN before viewing or operating the page.

> `Station 4 Page 3 Unlock`
> Authenticates the PIN on Page 3 of Station 4. If the page requires a PIN number, it is automatically entered, allowing the user to use the page without needing to manually enter the correct PIN.

See Also

- Selection Operators
- **Lock**, **Unlock**

Playback

Syntax

Command	Description	Return Value
`Playback <number> [<range...>]`	Change the active playback fader and/or select a range of playbacks	The new playback number
`Playback Set <number>`	Change the active playback fader without changing the current selection	The new playback number
`Playback ?`	Return the currently active playback fader	The current playback number

- `<number>`
  - A playback fader number from `1` to `32`.
  - Optionally use `0` to refer to the special *Live* playback fader (or see the **Live** command).
  - Optionally use `Next` or `Previous` to increment/decrement to the next or previous active playback.

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 **Playback** 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.

Selecting a Range of Playbacks

The **Playback** command works differently from other selection commands. When selecting a range of playbacks (for example using the command `Playback 3>5`), the first playback of the range becomes the "active" playback while all of the playbacks in the range become selected. This is for two reasons, (1) only one playback can be active at a time, but (2) the selected range can be used for working with playbacks with the **At**, **Clear**, **Enable**, and **Disable** commands. For example, the command `Playback 1>7 Clear` will clear all seven playbacks, but only playback 1 will become "active".

Incrementing or Decrementing to the Next/Previous Playback

Use the commands `Playback Next` or `Playback Previous` to increment/decrement the currently active playback to the next/previous playback.

Incrementing past the last regular playback fader moves to the special *Live* playback. Also, the increment/decrement commands do not move past the Live playback or before the first playback.

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.

Changing The Active Playback Without Selected Object(s)

A special form of the **Playback** command is available that changes the active playback fader without modifying the current selection. For example, if the command `Playback 1; Channel 1 At FL` is executed, Channel 1 will be set to Full in Playback 1. After the command is finished, the current selection will be *Channel 1*. You can change the active playback without loosing the Channel 1 selection by the command `Playback Set 2; At FL`. The active playback is changed to Playback 2 without loosing the Channel 1 selection, so the At FL command will operate on Channel 1 in Playback 2.

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”.

> ```
> Playback 2>4 At 50
> ```
> Makes playback 2 active and sets playback 2, 3, and 4’s submaster to 50%.

> ```
> Playback 4>8 Disable
> ```
> Makes playback 4 active and disables playbacks 4 through 8.

> ```
> Playback 1; Channel 1>10 At 33
> Playback Set 2; At 66
> ```
> Sets channels 1 through 10 to 33% in Playback 1, and channels 1 through 10 to 66% in Playback 2.

> ```
> Playback 1 At FL
> Playback Next At 66
> Playback Next At 33
> ```
> Sets the submasters of Playback 1 to 100%, Playback 1 to 66%, and Playback 2 to 33%.

See Also

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

Property

Syntax

Command	Description	Return Value
`Property <number>`	Select individual properties of one or more DMX fixtures by number	`0`

- `<number>`
  - A property number from `1` to `16384`.

Abbreviation

`PRO`

Description

Selecting Properties

The **Property** command selects the *nth* channel of each currently selected fixture. If more than one fixtures are currently selected, the *nth* channel of each selected fixture is selected.

Once one or more fixture channels are selected, use action commands such as **At**, **On**, **Off**, **Enable**, **Disable**, **Park**, **Unpark** or **Release** to set channel levels, change the enable or parked state of the channels, or release them.

Examples

> `Fixture 1 Property 5`
> Selects the 5th channel of fixture 1.

> `Fixture 1>5 Property 2`
> Selects the 2nd channels of fixtures 1 through 5.

> `Fixture * Property 3 Park`
> Selects the 3rd channels of every fixture and then parks those channels.

See Also

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

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 At Page 3`
> Sets Station 1 as the default station, and then changes the default station to display Page 3.

> `Station 3+5 Lock`
> Sets Station 3 and 5 as the default station, and then locks them (Stations 3 and 5).

> `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
- **At**, **Button**, **Contact**, **Disable**, **Enable**, **Indicator**, **Lock**, **Output**, **Page**, **Unlock**

Universe

Syntax

Command	Description	Return Value
`Universe <number> [<range...>]`	Select one or more universes	The current enable state of the universe
`Universe Clear` or `Universe ;`	Deselect all universes	`0`
`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.

Deselecting All Universes

Using either **Universe Clear** or **Universe ;** will deselect all universes while leaving “Universe” as the current command target.

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.

Assert

Syntax

Command	Description	Return Value
`Assert`	Raises the priority of selected LTP channels in the active playback fader	`0`

Abbreviation

`AS`

Description

The **Assert** command raises the priority of selected LTP (Latest Takes Precedence) channels in the active playback fader.

When the current fixture patch contains LTP channels, and that channel has values in more than one playback fader, only a single playback fader that has priority for that channel will output that channel value. Specially, the most recent (latest) playback fader that received a value for that channel will have priority. Sometimes it is useful to raise the priority of of channels in a playback so they regain control of the output.

LTP channels are typically used for moving light fixture parameters such as Pan/Tilt, indexed color wheels, or control channels. These fixture properties do not work well in scenarios where values coming from multiple playbacks would normally be combined together in a Highest Takes Precedence (HTP) manner. For these properties, only the most recently adjusted playback will take precedence.

Example

Assume there are two cues that have the position (Pan/Tilt) recorded for a moving light fixture. Cue 1 points the fixture towards the Left-Rear of the room, and Cue 2 points the fixture towards the Front-Right of the room.

First, Cue 1 is played back in Playback 1:

```
Playback 1; Cue 1 Go
```

The fixture points to the Left-Rear of the room. There is no other playback contributing values to the Pan/Tilt channels of the fixture.

Next, Cue 2 is played back in Playback 2:

```
Playback 2; Cue 2 Go
```

The fixture points to the Right-Front of the room. This occurs because now the Pan/Tilt channels in Playback 2 have the highest LTP priority.

Next, we want the values for Pan/Tilt in Playback 1 to become prioritized, so we assert those channels:

```
Playback 1; Fixture 1; Assert
```

The fixture points to the Left-Rear of the room. This occurs because now the Pan/Tilt channels in Playback 1 have the highest LTP priority.

In this last command, we selected all channels in Fixture 1 to become asserted, but we could have narrowed down the selection to only target the Pan/Tilt channels using a command like `Channel 1>4; Assert`, or possibly use a group like `Group 3; Assert`, or any other channel selection command.

See Also

Fixture, * **Playback**

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**

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
`At Input`	Sets the selected channels to the values currently coming from the DMX input	The number of channels set
`At Output`	Sets the selected channels to the values currently being sent to the DMX output	The number of channels set
`At [+/-]<value>`	Set the value of the selected object(s) to an offset (or delta) from the object(s) current value	The value the object(s) were set to
`At Page <page>`	Set the page of the selected station(s) to the given page number	The page number
`At ?`	Get the value of the selected object(s)	The current value of the selected object(s)

- `<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%`
    - A sign `+` or `-` may appear before the value to specify an offset (or delta) from the current value.
    - An exclamation point `!` may appear before the value to indicate that the current fade time should be ignored (the value is set immediately).
- `<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`
- `<page>`
    - Any page number from `1` to `99`

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**, **Fixtures**, **Groups**, **Outputs**, **Playbacks**, and **Presets**.

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

Object	Result Of At Command
Buttons	Turns the button(s) indicator on or off or sets it to a special user value
Channels	The channel(s) are set to the specified value
Fixtures	The fixture(s) *intensity* channel(s) are to the specified value
Groups	The channels in the group are set to the specified value
Outputs	Turns the output(s) on or off
Playbacks	The playback(s) submasters are set to the specified value
Presets	The preset is activated with an intensity of the the specified value
Stations	Changes the active page number of the station

Setting Values With Timing

A playback fader’s **Time** parameter will cause the value of Channels, Fixtures, 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.

Setting 16-Bit Values

When working with channels that are patched as “16-Bit” values, the **At** command will intelligently calculate and place the MSB (most significant byte) into the upper channel and the LSB (least significant byte) into the lower channel. For example, if channels 1 and 2 are paired together to make a 16-Bit value and the command `Channel 1+2 At 75` is executed, the actual value placed into channel 1 will be 117 (decimal) and channel 2 will be 255 (decimal). These two individual 8-Bit values result in a 16-Bit value of 49151 (decimal), which represents 75% of the full scale of a 16-Bit value. Luckily one does not need to manually do this math themselves. CueScript does this for you!

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.

Pulling Values From the DMX Input

Using the **At Input** command allows the channels from the DMX Input to be used to set the currently selected channel values (instead of specifying a single fixed value). This is useful to copy (or snapshot) the DMX Input into a playback fader.

Pulling Values From the DMX Output

Using the **At Output** command allows the channels from the DMX Output to be used to set the currently selected channel values (instead of specifying a single fixed value). This is useful to copy (or snapshot) the DMX Output into a playback fader.

Using Delta Values

Using the **At +/-** command allows channels to be set to an offset (or delta) from their current value(s). This is useful if the current channel value is not known and an offset value need to be added or subtracted from the current value. For example, an offset can be used to bump a channel value up or down using commands such as `Channel 1 At +5` or `Group 7 At -25`.

Setting Indicators

Using the **At** command with buttons will turn a button’s LED indicator on or off. Use `At 0` or `OFF` to turn an indicator off. Use `At FL`, `At 100` or `ON` to turn an indicator on. Values of `1`, `2`, `3`, and `4` will set an indicator to one of the "user colors". Other special values set an indicator's state to one of the other predefined categories as shown in the following table:

Value	Indicator State
`0`, `OFF`	Off
`1`	User 1
`2`	User 2
`3`	User 3
`4`	User 4
`5` .. `95`	*Reserved*
`96`	Recorded
`97`	Locked
`98`	Mixed
`99`	On (flashing disabled)
`100`, `ON`, `FL`	On

When setting the indicator state of a button, a "temporary override" function can be invoked by including an exclamation point (`!`) before the indicator value. This temporary override sets the indicator only briefly to the new value and then automatically reverts back to the original value. The duration of the temporary indicator state is given by the current fade time given by the **Time** command. If no time is set, then the default override time of 1 second.

For example, the command `Time 2.5; Button 1 At ! 3` sets the time to 2.5 seconds, then temporarily changes the indicator state of Button 1 to "User 3". After 2.5 seconds, the indicator will return back to its normal state.

Setting Outputs

Using the **At** command with outputs will turn an output on or off. Use `At 0` or `OFF` to turn an output off. Use `At FL`, `At 100` or `ON` to turn an output on.

Ignoring Timing

Normally, when setting the value of an object such as a Channel, Fixture, Group, or Playback, the currently active **Time** will be applied to the value change. If the value is preceded by an exclamation point (`!`), the fade time will be ignored, resulting in the value being set immediately without fading.

Changing Station Pages

Using the **At Page** command with stations will change the station’s active page.

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.

> ```
> Channel 1>10 At +15
> ```
> Increases the value of Channels 1 thru 10 by 15%.

> ```
> Preset 1 At 33
> ```
> Activates preset 1 with intensity level 33%.

> ```
> Button 1 At FL
> ```
> Turns the indicator for Button 1 on.

> ```
> Button 2 At 3
> ```
> Turns the indicator for Button 2 to the color specified as “User 3”.

> ```
> Time 5 Button 3 At ! 4
> ```

> Temporarily turns the indicator for Button 3 to the “User 4” state for 5 seconds.

> `Output 3 At 0`
> Turns output 3 off.

> `Station 1 At Page 7`
> Changes Station 1 to display Page 7.

> `Channel 1` ! FL@
> Sets channel 1 to Full immediately, ignoring the current fade time.

See Also

- **Button**, **Contact**, **Group**, **Off**, **On**, **Output**, **Playback**, **Preset**, **Station**

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.

Clear

Syntax

Command	Description	Return Value
`Clear`	Clears the selected playback fader(s) or effect(s)	`0`

Abbreviation

`CL`

Description

The **Clear** command clears the selected playback fader(s) or effect(s), depending on which object is currently selected. If a different object other than a playback or effect is selected, the **Clear** command clears the currently active playback fader.

Clearing a Playback Fader

Clearing a playback fader has the following effect:

- Releases all DMX channels (including locked 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%

Clearing an Effect

Clearing an effect has the following effect:

- The effect type is returned to *None*
- All effect properties are reset

Examples

> `Playback 1 Clear`
> Clears playback 1.

> `Playback 4>8 Clear`
> Clears playbacks 4 through 8.

> `Playback 3 Effect 1 Clear`
> Clears Effect 1 in Playback 3.

See Also

- **Effect**, * **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**

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**, **Effects**, **Groups**, **Playbacks**, **Stations**, 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	Channel(s) contribute to playback	Channel(s) do not contribute to playback
Contacts	Responds to closures normally	Does not trigger any actions
Effects	Modifies channels in the playback	Does not modify channels in the playback
Groups	Channel(s) contribute to playback	Channel(s) do not contribute to playback
Playbacks	Playback(s) contribute to output	Playback(s) do not contribute to output
Stations	Station(s) operate normally	Station(s) power down
Universes	Normal broadcast from universe	No broadcast from universe

✻ See also the **Lock** command. *Locking* a button, contact, or station offers similar but different behaviors.

Examples

> `Button 1 Disable`
> Disables button 1.

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

> `Playback 3 Disable`
> Disables playback 3.

> `Station 7 Disable`
> Disables station 7. The station will power-off.

> `Universe 1+7 Disable`
> Disables universes 1 and 7.

See Also

Button, **Channel**, **Contact**, **Disable**, **Enable**, **Group**, **Lock**, **Playback**, **Station**, **Unlock**, **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**, **Effects**, **Groups**, **Playbacks**, **Stations**, 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	Channel(s) contribute to playback	Channel(s) do not contribute to playback
Contacts	Responds to closures normally	Does not trigger any actions
Effects	Modifies channels in the playback	Does not modify channels in the playback
Groups	Channel(s) contribute to playback	Channel(s) do not contribute to playback
Playbacks	Playback(s) contribute to output	Playback(s) do not contribute to output
Stations	Station(s) operate normally	Station(s) power down
Universes	Normal broadcast from universe	No broadcast from universe

✻ See also the **Lock** command. *Locking* a button, contact, or station offers similar but different behaviors.

Examples

> `Button 1 Enable`
> Enables button 1.

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

> `Playback 3 Enable`
> Enables playback 3.

> `Station 7 Enable`
> Enables station 7.

> `Universe 1+7 Enable`
> Enables universes 1 and 7.

See Also

Button, **Channel**, **Contact**, **Disable**, **Enable**, **Group**, **Lock**, **Playback**, **Station**, **Unlock**, **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.

Join

Syntax

Command	Description	Return Value
`Join <join group>`	Sets the join group of the current zone	The join group number
`Join Clear`	Sets the join group of the current zone to 0	`0`
`Join ?`	Returns the current zone's join group	The join group number

- `<join group>`
  - A numeric ID from `0` through `32`.

Abbreviation

None

Description

Use the **Join** command to set the *Join Group Number* for the selected zone.

Each Zone can be assigned a Join Group. By default each Zone starts in Group 0 (zero). When a Zone is a member of a non-zero Join Group, that zone becomes logically “joined” with all other Zones that are members of the same group.

When Zones are joined together by having the same Join Group Number, the channels of all Zones in the group are merged together. Also, any Preset number that is activated a Zone will be also activated in all joined zones.

For example, a project has three zones named “Ballroom A”, “Ballroom B”, and “Ballroom C”. Each of these zones has eight presets numbered 1 through 8. The channels in each zone are 1>10, 11>20, and 21>30, respectively.

If a user activates any preset in any of the three zones, that preset only operates in that zone. It is assumed that removable walls are separating each of these zones.

Next, the wall that separates Ballroom B and C are removed. The desired effect is to “join” these two rooms. To perform this connection between the two zones, the *Join Group Number* of each of these two zones must match. The following commands will set both zone’s group number to 7:

```
Zone "Ballroom B" Join 7
Zone "Ballroom C" Join 7
```

Once both of these zones have the same *Join Group Number*, any action taken in one zone will be mirrored in the other zones. For instance, if the following command is executed:

```
Zone "Ballroom B" Preset 3 On
```

Preset 3 will become activated in the Ballroom B Zone. Additionally, Ballroom C’s Preset 3 will also be activated.

Join Group “7” was arbitrarily chosen for this example. Any Join Group from 1 through 32 could have been used. It is only important that the _same* Join Group number be used for each zone that should be joined together.

The concept of the Join Group is flexible enough to create scenarios where very complex and not-necessarily physically connected spaces in the project can be joined together. And because there are 32 possible Join Groups to use, a large number of different logical blocks of zones can be joined together.

When a zone should no longer be joined with any other zones, use the command **Join 0**, or **Join Clear** to set the zone’s Join Group to zero.

> Please note that Join Group numbers are ***not*** related to channel Groups.

Examples

```
Join 3
```

Sets the currently selected zone’s Join Group to 3.

```
Zone "Foyer" Join 5
```

Sets the Join Group for the Foyer zone to 5.

```
Zone "Happy" Join Clear
```

Clears (sets to zero) the Join Group for the Happy zone.

```
Join ?
```

Returns the currently selected zone’s Join Group.

See Also

- **Preset**, **Zone**

Length

Syntax

Command	Description	Return Value
`Length <playback time>`	Sets the current streaming cue’s playback length	The playback length set
`Length ?`	Returns the current streaming cue’s playback length	The current playback length

- `<playback time>`
  - A decimal number of seconds (optionally using decimal digits for fractions of seconds)
  - `0` means not to override the cue’s playback length (this is the default)
  - A whole number proceeded by a `#` specifies a number of frames instead of seconds

Abbreviation

`LEN`

Description

Setting a Streaming Cue’s Playback Length

Use the **Length** command to set the *playback length* for playing back a streaming cue. This time is used to adjust the duration of a streaming cue before it begins playing back. For instance, if a stream is recorded too long, a shorter playback length can be given to have the stream stop playing at the proper place in the cue. The **Length** command must be issued after the cue is selected but before the corresponding **Go** command is issued. For example, **Cue 1 Length 5.5 Go**.

If a playback length is specified that is shorter than the recorded length of the cue, then the cue will stop (or loop, or follow, etc.) at the specified time, which will be before the recorded data in the cue is finished playing back. If a playback length is specified that is longer than the recorded length of the cue, then there will be a delay after the end of the recorded data before the cue will stop (or loop, or follow, etc.).

Playback times are normally expressed in seconds. By using a pound-sign (`#`) before the value allows the playback time to be expressed in *fames*. A frame is 1/40th of a second and is the smallest unit of time that streaming cues are recorded in.

> ✻ The **Length** command is frequently used in conjunction with the **offset** command, which is used to adjust the starting position of streaming cue playback independently from the actual recorded starting point of the cue.

Determining The Current Playback Time

Use the **Length** command with the question mark (?) to return the current playback time. A playback time such as `5` or `1.025` will be returned.

Examples

> `Length 1`
> Sets the current streaming cue's playback time to 1 second.

> `Cue 4 Length 8.075 Go`
> Loads cue 4, then sets its playback time to 8.075 seconds, and then begins playing the cue.

> `Cue 17 Offset #1 Length 3.5 Go`
> Loads cue 17, then sets its offset time to 1 frame (0.025 seconds) seconds, then sets the playback length to 2.5 seconds, and then begins playing the cue.

See Also

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

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 its linked cue to cue 1 before executing it.

> `Link Clear`
> Clears any currently linked cue from the active playback fader.

See Also

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

Lock

Syntax

Command	Description	Return Value
`Lock`	Locks the selected object(s)	The number of objects locked

Abbreviation

None

Description

The **Lock** command *locks* the currently selected object(s). The **Lock** command can be used with **Buttons**, **Channels**, **Contacts**, **Pages**, and **Stations**. Locked buttons or contacts do not trigger events when operated. If the button has an indicator, it appears in the “locked” state. If a channel is locked in a playback fader, its value becomes unchangeable. If a page that requires a PIN number is locked, then the user will need to enter the correct PIN number before using the page. If a station is locked, all of its buttons will appear in the “locked” state and/or a lock icon will appear on the screen. **Unlock** has the opposite effect as **Lock**.

The following table shows the various effect of locking or unlocking an object:

Object	When Unlocked	When Locked
Buttons	Responds to presses normally	Does not trigger any actions, displays *locked* indicator state
Channels	Value in playback fader is changeable by cues, presets, and manual commands	Value in playback fader is not able to be changed
Contacts	Responds to closures normally	Does not trigger any actions
Pages	The user may use the page	The user must enter the correct PIN number to use the page
Stations	Station operates normally	Does not trigger any actions, all controls display *locked* indicator state and/or screen displays lock icon

✻ *Disabling* a button, channel, contact, or station offers similar but different behaviors. See the **Disable** command for more information.

Parking a channel makes a channel unchangeable at a global level instead of just in a single playback. See the **Park** command for more information.

Examples

> `Button 1 Lock`
> Locks button 1.

> `Button 3>5 Lock`
> Locks buttons 3 through 5.

> `Channel 1>10 Lock`
> Locks channels 1 through 10 in the current playback fader.

> `Page 4 Lock`
> If page 4 of the current station requires a PIN number, the user will have to enter that PIN number to be able to use the page.

> `Station 7 Lock`
> Locks station 7. It will display its *locked* state.

See Also

Button, **Channel**, **Contact**, **Disable**, **Enable**, **Page**, **Station**, **Unlock**

Log

Syntax

Command	Description	Return Value
`Log [options] <string>`	Writes a message to the system log	The message is returned
`Log Clear`	Clears the important message indicator	The number of messages cleared
`Log Reset`	Removes all messages from the system log	The number of messages removed
`Log ?`	Returns the current important message count	The number of important messages pending

- `<string>`
  - A user-defined string “in quotes”.

- `[options]`
  - `!` causes the message to be logged with “Error” level of priority.
  - `~` causes the message to be logged with “Warning” level of priority.

Abbreviation

None

Description

The **Log** command writes a message to the system log.

> If the system writes an important message to the system log, the device’s power/status LED will blink with a magenta color indicating that an important message has been logged.

Adding either a `!` or `~` before the string will elevate the priority of the log message to either “Error” or “Warning”, respectively.

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

Using **Log Reset** will remove all messages from the system log.

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.

> ```
> Log "Current cue is ${myCue}"
> ```
> Writes the string “Current cue is “, followed by the value of the *myCue* variable to the system log. (See the section on Strings for more information about substituting variable values into strings.)

> ```
> Log ! "Show not started on time"
> ```
> Writes the string “Show not started on time” to the system log with “Error” level of priority.

> ```
> Log Clear
> ```
> Clears the “important message” indicator.

> ```
> Log ?
> ```
> Returns the number of important 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.

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%.

> `Output 6+8 Off`
> Turns off outputs 6 and 8 on the current station.

> `Playback 2 Off`
> Sets the submaster of playback 2 to 0%.

> `Preset 4 Off`
> Turns off preset 4 in the current zone.

See Also

Button, **Channel**, **Group**, **Indicator**, **Output**, **Playback**, **Preset**

Offset

Syntax

Command	Description	Return Value
`Offset <offset time>`	Sets the current streaming cue’s starting offset	The offset time set
`Offset ?`	Returns the current streaming cue’s starting offset	The current offset time

- `<offset time>`
  - A decimal number of seconds (optionally using decimal digits for fractions of seconds)
  - `0` means no offset time (this is the default)
  - A whole number proceeded by a `#` specifies a number of frames instead of seconds

Abbreviation

None

Description

Setting a Streaming Cue’s Offset Time

Use the **Offset** command to set the *offset time* for playing back a streaming cue. This time is used to adjust the starting point of a streaming cue before it begins playing back. For instance, if a stream is recorded one second too early, an offset time of one second can be given to have the stream start playing at the proper place in the cue. The **Offset** command must be issued after the cue is selected but before the corresponding **Go** command is issued. For example, **Cue 1 Offset 1.3 Go**.

Offset times may be either positive or negative. Positive offset times move the playback pointer into the streaming cue data, starting the stream already somewhat into the recorded data. Negative offset times move the playback pointer to a time *before* the beginning of the streaming cue data, causing a delay before the recorded data plays back.

Offset times are normally expressed in seconds. By using a pound-sign (`#`) before the value allows the offset time to be expressed in *fames*. A frame is 1/40th of a second and is the smallest unit of time that streaming cues are recorded in.

> The **Offset** command is frequently used in conjunction with the **Length** command, which is used to adjust the playback length of a streaming cue independently from the actual recorded length of the cue.

Determining The Current Offset Time

Use the **Offset** command with the question mark (?) to return the current offset time. An offset time such as `5` or `1.025` will be returned.

Examples

> `Offset 1`
> Sets the current streaming cue’s offset time to 1 second.

> `Cue 4 Offset 0.075 Go`
> Loads cue 4, then sets its offset time to 0.075 seconds, and then begins playing the cue.

> `Cue 17 Offset #1 Length 3.5 Go`
> Loads cue 17, then sets its offset time to 1 frame (0.025 seconds) seconds, then sets the playback length to 2.5 seconds, and then begins playing the cue.

See Also

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

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` or `At FL`.

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%.

> `Output 6+8 On`
> Turns on outputs 6 and 8 on the current station.

> `Playback 2 On`
> Sets the submaster of playback 2 to 100%.

> `Preset 4 On`
> Turns on (activates) preset 4 in the current zone.

See Also

Button, **Channel**, **Group**, **Indicator**, **Output**, **Playback**, **Preset**

Park

Syntax

Command	Description	Return Value
`Park`	Parks the selected channel(s)	The number of channels parked

Abbreviation

None

Description

The **Park** command causes the selected channels' output values to become unchangeable. Whatever values the selected channels were outputting from CueServer at the moment they were parked is now frozen and cannot be changed by anything occurring in the playback faders, submasters, cues, presets or from the DMX Input. Channels are parked at the output stage of the CueServer, after all other input, playback and live stages.

Use channel parking as a way to guarantee that certain channels will maintain their value regardless of other activities occurring on the CueServer.

The **Unpark** command has the opposite effect as the **Park** command. Additionally, if it is desired to freeze a channel value in an individual playback fader, see the **Lock** and **Unlock** commands.

! Please Note: Prior to software version 5.0, the **Park** command behaved differently. Instead of parking channel values globally in the output layer, it used to lock channels in a specific playback fader. Additionally, the **Clear** and **Reset** commands were able to reset any locked channels in a playback. Starting with version 5.0, the **Park** command parks channels in the output layer and are unaffected by the **Clear** or **Reset** commands. The previous behavior has been moved to the **Lock** command. The new **Park** behavior mirrors that of traditional lighting consoles.

Examples

```
Park
```

Parks the currently selected channels, making their output unchangeable.

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

Parks channels 1 through 3 and 5 through 8.

```
Channel * Park
```

Parks all channels.

See Also

Channel, **Lock**, , **Unlock**, **Unpark**

Preset

Syntax

Command	Description	Return Value
`Preset <preset number> On`	Activates the specified preset in the current zone	The preset number activated
`Preset <preset number> Off`	Deactivates the specified preset in the current zone	The preset number deactivated
`Preset <preset number> At <value>`	Activates the specified preset in the current zone at the given intensity level	The preset number activated
`Preset <preset number> Toggle <value>`	Activates or deactivates the specified preset in the current zone at the given intensity level	The preset number toggled

- `<preset number>`
  - Any whole number from `1` to `999`
- `<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

`PR`

Description

Use the **Preset** command to operate on presets within the current zone. Presets can be activated, deactivated, toggled and/or set to a desired intensity level.

Normally, Presets only operate within their *Zone*. See the **Zone** command to see how to manage Zones. In certain applications, it is useful to be able to join multiple zones together (such as in a Ballroom with removable “air walls”). See the **Join** command to learn how to join Zones together.

Activating Presets

To activate a preset, use the **Preset *n* On** command. For example, to activate Preset 3, use this command:

```
Preset 3 On
```

The stored look in Preset 3 will appear in the current zone.

Any time a preset is activated in a particular zone, any other preset in that zone is automatically deactivated. The only exception to this rule is when more than one preset contains the same channel values. If more than one preset share the same channel values, they will *all* become activated.

Deactivating a Preset

If a preset needs to be deactivated without activating another, the **Off** action can be used:

```
Preset 3 Off
```

If a zone contains one or more presets that are recorded with all zero channel levels, those presets will become active when other presets are deactivated.

Setting a Preset’s Intensity Level

If a Preset needs to be recalled, but at a lower intensity level than recorded, the **Preset *n* At *value*** command can be used:

```
Preset 3 At 33
```

In this example Preset 3 is activated, but the channel levels are scaled down to 33% of the original levels recorded in the Preset.

The value used to scale the preset can be given in percentage, decimal, or hexadecimal, similarly to the **At** command.

Toggling Presets

To cause a preset to toggle on and off, use the **Preset *n* Toggle *value*** command. For example:

```
Preset 3 Toggle On
```

Each time the **Toggle** action is used, the preset will turn on or off. The new state will be the opposite of the previous state.

The **Toggle** action can also be used with values other than “On”. For example, to toggle a preset between “Off” and “75%”, use this command:

```
Preset 3 Toggle 75
```

Examples

> `Preset 5 On`
> Activates Preset 5 in the current zone.

> `Zone "Foyer" Preset 3 On`
> Activates Preset 3 in the Zone “Foyer”.

> `Preset 1 Toggle On`
> Activates or deactivates Preset 1 depending on Preset 1’s previous state.

> `Preset 7 Off`
> Deactivates Preset 7 in the current zone.

See Also

- **At**, **Join**, **Zone**

Press

Syntax

Command	Description	Return Value
`Press`	Presses the selected object(s)	`0`

Abbreviation

None

Description

The **Press** command is used to perform the same event that would occur if the user physically *presses* a button (or *closes* a contact). **Release** has the opposite effect as **Press**.

When the **Press** command is executed, the selected Button (or Contact, or Shared Control) will receive a “press event”, which causes all of the *press events* to be executed for that object.

It is not necessary to always issue a **Release** command for each **Press** command. If a Button receives multiple “press events”, it will execute its “Whenever this Button is Pressed” actions each time. However, if the Button has a “Whenever this Button is Held” rule, that rule will be executed some number of seconds in the future if a **Release** is not issued for that Button.

Examples

> `Button 1 Press`
> Performs the same actions as if the user has physically pressed Button 1.

> `Button 1+3+5 Press`
> Presses Buttons 1, 3, and 5.

> `Button 2.3 Press; Release`
> Presses and then immediately releases Button 2.3.

See Also

Button, **Contact**, **Control**, **Release**

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, Presets, Streams and Groups.

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

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

Record Cue

Syntax

Command	Description	Return Value
`Record [options] 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`

- `[options]`
  - Selection:
    - `All` causes all DMX channels to be recorded; **this is the default**
    - `Empty` causes no DMX channels to be recorded into the cue
    - `Selected` causes only the currently selected DMX channels to be recorded into the cue
    - `Active` causes only active DMX channels to be recorded into the cue
  - Source:
    - `Input` causes the DMX Input to be recorded
    - `Playback n` causes only the channels from Playback *n* to be recorded
    - `Live` causes only the channels from the Live playback to be recorded
  - Other:
    - `Stack s` record into Stack *s*
    - `Merge` record channels without modifying other existing channels in the cue

Abbreviation

`R` [`ALL`, `EMPTY`, `SEL`, `ACTIVE`, `IN`, `P n`, `STACK s`] `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. Use the command *options* to change which channels and what source the channels are recorded from.

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. More than one option may be used, and they can be listed in any order.

All
If this option is used, then the new cue will be created containing all DMX channels. When this cue is played back, it will affect every DMX channel.

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 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 specific channels when being played back.

Active
If this option is used, then the new cue will be created containing only the currently active DMX channels. Using this option, cues can be created that only affect specific channels when being played back. When recording the DMX output or input, active channels are any channels that have a non-zero value. When recording from one of the Playbacks, active channels are any channels active in the playback (including zero value channels).

Input
If this option is used, then the new cue will be recorded by using only values from the DMX Input. The playback faders and DMX output will not be recorded.

Playback *n*
If this option is used, then the new cue will be recorded by using only values from the specified playback fader. The DMX input and output will not be recorded.

Live
If this option is used, then the new cue will be recorded by using only values from the special *Live* playback fader. The DMX input and output will not be recorded.

Stack *s*
If this option is used, then the new cue will be recorded into Stack *s*. This option overrides the stack chosen in the current playback fader.

Merge
If this option is used, the recorded channels are stored in the cue without disturbing other existing channels in the cue that are not part of the current selection. For instance, if a cue previously had

values for channels 1>10 and only channel 1 is recorded with the merge option, then channel 1 is updated to the new value without changing the values for channels 2>10.

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.

> `Record Active Input Cue 5.1`
> Records only the active channels from the DMX Input as Cue 5.1.

> `Record Selected Playback 7 Cue 1.23`
> Records the currently selected channels from Playback 7 as Cue 1.23.

> `Record Active Playback 1 Stack "Test" Cue 101.5`
> Records only the active channels from Playback 1 into Cue 101.5 in the cue stack named “Test”.

> ```
> 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 [options] Group <group number>`	Records group *group number*	The group number recorded

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

- `[options]`
    - Selection:
        - `All` causes all DMX channels to be recorded
        - `Empty` causes no DMX channels to be recorded into the group
        - `Selected` causes only the currently selected DMX channels to be recorded into the group; **this is the default**
        - `Active` causes only active DMX channels to be recorded into the group
    - Source:
        - `Input` causes active channels in the DMX Input to be used as the template for the group
        - `Playback n` causes active channels in Playback *n* to be used as the template for the group
        - `Live` causes active channels in the special *Live* playback to be used as the template for the group
    - Other:
        - `Merge` add channels without modifying other existing channels in the group

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.

Record Group Options

Several options are available to change how a group is recorded. More than one option may be used, and they can be listed in any order.

All
If this option is used, then the new group will be created containing every DMX channel. When this group is recalled, it will select all DMX channels.

Empty
If this option is used, then the new group will be created containing no DMX channels. When this group is recalled, it will deselect all DMX channels.

Selected
If this option is used, then the new group will be created containing the currently selected DMX channels. This is the default option, so specifying it is redundant.

Active
If this option is used, then the new group will be created containing only the currently active DMX channels. When recording the DMX output or input, active channels are any channels that have a non-zero value. When recording from one of the Playbacks, active channels are any channels active in the playback (including zero value channels).

Input
If this option is used, then the new group will be recorded by using only values from the DMX Input.

Playback *n*
If this option is used, then the new group will be recorded by using only values from the specified playback fader.

Live
If this option is used, then the new group will be recorded by using only values from the special *Live* playback fader.

Merge
If this option is used, the recorded channels are stored in the group without disturbing other existing channels in the group that are not part of the current selection. For instance, if a group previously contained channels 1>10 and only channel 20 is recorded with the merge option, then channel 20 is added to the group without removing channels 1>10.

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 Preset

Syntax

Command	Description	Return Value
`Record [options] Preset <preset number>`	Records preset *preset number*	The preset number recorded

- `<preset number>`
  - Any whole number from `0` to `999`

- `[options]`
  - Source:
    - `Input` causes the DMX Input to be recorded
    - `Playback n` causes only the channels from Playback *n* to be recorded
    - `Live` causes only the channels from the Live playback to be recorded
  - Other:
    - `Zone "z"` record into Zone *z*

Abbreviation

`R` [`IN`, `P n`, `L`, `ZO "z"`] `PR`

Description

Recording Presets

The **Record Preset** command records (or re-records) a preset.

By default, all channels being output from the CueServer are captured into the preset, and the preset is recorded in the currently active zone. Use the command *options* to change what source the channels are recorded from or to choose a different zone to record into.

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

Record Preset Options

Several options are available to change how a preset is recorded. More than one option may be used, and they can be listed in any order.

Input
If this option is used, then the new preset will be recorded by using only values from the DMX Input. The playback faders and DMX output will not be recorded.

Playback *n*
If this option is used, then the new preset will be recorded by using only values from the specified playback fader. The DMX input and output will not be recorded.

Live
If this option is used, then the new preset will be recorded by using only values from the special *Live* playback fader. The DMX input and output will not be recorded.

Zone "*z*"
If this option is used, then the new preset will be recorded into Zone *z*. This option overrides the currently active zone.

Examples

> `Record Preset 1`
> Records the current output from the CueServer into Preset 1 in the current zone.

> `Record Input Preset 5`
> Records the current DMX Input to the CueServer into Preset 5 in the current zone.

> `Record Zone "B" Preset 99`
> Records the current output from the CueServer into Preset 99 in the zone named "B".

See Also

- **Update Preset**

Record Stream

Syntax

Command	Description	Return Value
`Record [options] 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`

- `[options]`
    - `Channel n` causes the stream recording to wait for Channel *n* to become non-zero before recording and return to zero to stop recording
    - `Time t` causes the stream recording to automatically stop after *t* seconds
    - `Stack s` record into Stack *s*

Abbreviation

`R` [`C n`, `T t`, `STA s`] `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 streaming cue with the cue number exists, a new streaming cue will be created. If a cue with the cue number already exists, it will be deleted first and replaced with an entirely new streaming cue.

Record Streaming Cue Options

Several options are available to change how a streaming cue is recorded. More than one option may be used, and they can be listed in any order.

Channel *n*
If this option is used, then channel *n* is used as a stream recording "trigger channel". Recording will wait until channel *n* becomes non-zero. Then, recording will continue until channel *n* returns to a zero value. This option is useful in situations where a particular channel coming from a DMX source has been

programmed to signal the precise beginning and end of a DMX clip to record.

Time *t*
If this option is used, then the stream will automatically stop recording after *t* seconds have been captured. If a trigger channel is being used, the timer does not start until the actual recording stream has begun.

Stack *s*
If this option is used, then the new cue will be recorded into Stack *s*. This option overrides the stack chosen in the current playback fader.

Examples

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

> `Record Channel 512 Stream 2`
> Begins recording the current output from the CueServer into Streaming Cue 2, while using Channel 512 as the channel that will trigger the automatic start and stop of the recording.

> `Record Time 3.5 Stream 101.5`
> Begins recording the current output from the CueServer into Streaming Cue 101.5, and the stream will automatically stop recording after 3.5 seconds.

> `Record Time 15 Channel 1024 Stream 42`
> Begins recording the current output from the CueServer into Streaming Cue 42, while using Channel 1024 as the channel that will trigger the automatic start and stop of the recording. If the stream is still recording after 15 seconds, it will automatically stop.

> `Record Stop`
> Stops recording the Streaming Cue.

See Also

- **Record Stop**
- **Update Stream**
- **Update 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.

The **Update Stop** command is an alias for the same 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**
- **Update Stream**
- **Update Stop**

Release

Syntax

Command	Description	Return Value
`Release`	Releases channels from the active playback fader *or* Releases the selected object(s)	*None* 0

Abbreviation

`REL`

Description

The **Release** command can be used with Channels, Buttons, Contacts, or Shared Controls.

Releasing Channels

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 locked channels. To release all channels, including locked channels, consider unlocking channels first with the **Unlock** command or use the **Clear** command to unlock, release and reset the entire playback fader.

Releasing Buttons, Contacts or Shared Controls

The **Release** command is used to perform the same event that would occur if the user physically *releases* a button (or *opens* a contact). **Press** has the opposite effect as **Release**.

When the **Release** command is executed, the selected Button (or Contact or Shared Control) will receive a “release event”, which causes all of the *release events* to be executed for that object.

It is not necessary to always issue a **Press** command before each **Release** command. If a Button receives multiple “release events”, it will execute its “Whenever this Button is Released” actions each time. However, if the Button has a “Whenever this Button is Held” rule, that rule will be cancelled if the **Release** is received before the timer expires.

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.

> ```
> Button 1 Release
> ```
> Performs the same actions as if the user has physically released Button 1.

> ```
> Button 1+3+5 Release
> ```
> Releases Buttons 1, 3, and 5.

> ```
> Button 2.3 Press; Release
> ```
> Presses and then immediately releases Button 2.3.

See Also

- **Button**, * **Clear**, * **Contact**, * **Control**, **Playback**, **Press**

Reset

Syntax

Command	Description	Return Value
`Reset`	Resets all playback faders and command context	`0`

Abbreviation

None

Description

The **Reset** command clears all playback faders and resets the command line context.

Reset performs the following actions:

- Clears all playback faders (including locked channels)
- Stops any pending Wait commands
- Resets the command context including active playback, fade time, cue stack, zone, station and page.

! The **Reset** command does not affect *Parked* channels. If any channels are parked, those channels will continue to output their parked values without change. If it is desired to also remove all parked channels, use the command `Channel * Unpark`.

Examples

> `Reset`
> Entirely resets all playback faders and the command line context.

Return

Syntax

Command	Description	Return Value
`Return <value>`	Returns from CueScript function while returning a value to the caller.	`<value>`

Abbreviation

`RET`

Description

The **Return** command stops execution of the current CueScript code block and returns the given value to the caller. This command is useful inside of functions written in CueScript.

Examples

> ```
> Return 1
> ```
> Stops executing this CueScript and returns the value `1` to the calling function.

> ```
> If ('x' > 5) Then
>   Return "A"
> Else
>   Return "B"
> EndIf
> ```
> If variable `x` is greater than 5 then the CueScript function will return the string `A`, otherwise it will return the string `B`.

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**

SMPTE

Syntax

Command	Description	Return Value
`SMPTE Start`	Begins generating internal timecode	`1`
`SMPTE Stop`	Stops generating internal timecode	`0`
`SMPTE Clear`	Sets the current timecode to `00:00:00:00`	`0`
`SMPTE "<timecode>"`	Sets the current timecode to the specified time	`0`
`SMPTE <frame-number>`	Sets the current timecode to the specified frame number	`0`
`SMPTE [+/-]<frames>`	Increments or decrements the current frame	`0`
`SMPTE Input <source>`	Changes the SMPTE input source	`0`

- `<timecode>`
  - A string value representing a timecode, such as `"23:59:59:29"`.
  - Fewer places can be specified, for example `"1:23"`, which will be right-justified into a timecode of `"00:00:01:23"`.

- `<frame-number>`
  - A number of frames since “time zero”.

- `<source>`
  - The source of the SMPTE time. May be one of the following:
    - `0` or `"Internal"` to set to Internal Generation.
    - `1` or `"Audio"` to set to Audio Input.

Abbreviation

`SMPTE STA`, `SMPTE STO`, `SMPTE CL`, or `SMPTE IN`

Description

The **SMPTE** command can be used to manage timecode, including the internal timecode generator, and external timecode input.

Generating Timecode Internally

The **SMPTE** command can be used to start, stop, reset and set the current timecode using the CueServer’s internal timer. When generating timecode internally, events in CueServer’s timecode event list will be triggered at the specified times.

Use the **SMPTE Clear** command to reset the current timecode to zero (`00:00:00:00`).

Use the **SMPTE Start** command to start internal generation of timecode. The current timecode will begin incrementing at 30fps. Any timecode events in the system will trigger when the timecode reaches their marks.

Use the **SMPTE Stop** command to stop internal generation of timecode. The current timecode will freeze at the current time.

Use the **SMPTE** command to set the current timecode to a specific time. A timecode string such as `"01:00:04:29"` can be given. Additionally, a shorter string can be used, which will be padded with zeros on the left-side. For example, specifying a timecode of `"22:11"` will set a timecode of `"00:00:22:11"`. Furthermore, a frame number can be given, such as `50`, which will be interpreted as the 20th frame of the first second (`"00:00:01:20"`).

Use the **SMPTE ±** command to increment or decrement the current frame by the specified number of frames. For instance, the command `SMPTE +5` will increment the current timecode by 5 frames.

> ✱ Using any of the `SMPTE` commands that modify the timecode will automatically switch the input source to “Internal”.

Using External Timecode

Use the `SMPTE Input "Audio"` command to switch reception of timecode to the audio input port. Timecode will begin tracking the audio input if a valid signal is present. While timecode is being received at the audio input, any timecode events in the system will trigger when the timecode reaches their marks.

Examples

> `SMPTE Start`
> Starts generating timecode internally beginning with the current time. The input source will be changed to “Internal”.

> `SMPTE Stop`
> Timecode will freeze. The input source will be changed to “Internal”.

> `SMPTE Clear`
> Sets the current timecode to `00:00:00:00`. The input source will be changed to “Internal”.

> `SMPTE "5:43:21"`
> Sets the current timecode to `00:05:43:21`. The input source will be changed to “Internal”.

> `SMPTE 32`
> Sets the current timecode to `00:00:01:02`. The input source will be changed to “Internal”.

> `SMPTE -1`
> Decrements the current timecode by 1 frame. The input source will be changed to “Internal”.

> `SMPTE Input 0` or `SMPTE Input "Internal"`
> Switches the SMPTE input source to “Internal”.

> `SMPTE Input 1` or `SMPTE Input "Audio"`
> Switches the SMPTE input source to “Audio Input”.

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)	The number of channels unparked

Abbreviation

None

Description

The **Unpark** command causes previously *parked* channels to become changeable again. When a channel transitions from a parked state to an unparked state, the output value of the channel immediately “snaps” to whatever value is being driven by the current state of the input, playbacks, and live layers.

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

Examples

> `Unpark`
> Unparks the currently selected channels, making them respond to the input, playback, and live layers again.

> `Channel 1>3+5>8 Unpark`
> Unparks channels 1 through 3 and 5 through 8.

> `Channel * Unpark`
> Unparks all channels.

See Also

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

Update

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

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

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

Update Cue

Syntax

Command	Description	Return Value
`Update [options] Cue <cue number>`	Updates the DMX 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`

- `[options]`
  - Selection:
    - `All` causes all DMX channels to be updated
    - `Empty` causes all DMX channels to be removed from the cue
    - `Selected` causes the currently selected DMX channels to become the DMX channels in the cue
    - `Active` causes only active DMX channels to become the DMX channels in the cue
  - Source:
    - `Input` causes the DMX Input to be recorded
    - `Playback n` causes only the channels from Playback *n* to be recorded
    - `Live` causes only the channels from the Live playback to be recorded
  - Other:
    - `Merge` causes the selected or active channels to be merged with the cue's current content (not applicable with `All` or `Empty` modes)
    - `Stack s` record into Stack *s*

Abbreviation

`UP` [`ALL`, `EMPTY`, `SEL`, `ACTIVE`, `IN`, `P n`, `ME`, `STACK s`] `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 changing any of the other properties recorded in the cue (such as the fade and follow time, link, and rules).

By default, any channels currently recorded in the cue are updated.

If no cue with the cue number exists, a new cue will be created. If a cue with the cue number already exists, only the cue's DMX channels will be updated by using this command. All other cue parameters will not be affected.

Update Cue Options

Several options are available to change how a cue is updated. More than one option may be used, and they can be listed in any order.

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 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 specific channels when being played back.

Active
If this option is used, then the updated cue will contain only the currently active DMX channels. Using this option, cues can be created that only affect specific channels when being played back. When recording the DMX output or input, active channels are any channels that have a non-zero value. When recording from one of the Playbacks, active channels are any channels active in the playback (including zero value channels).

Input
If this option is used, then the updated cue will be recorded by using only values from the DMX Input. The playback faders and DMX output will not be recorded.

Playback *n*
If this option is used, then the updated cue will be recorded by using only values from the specified playback fader. The DMX input and output will not be recorded.

Live
If this option is used, then the updated cue will be recorded by using only values from the special *Live* playback fader. The DMX input and output will not be recorded.

Merge
If this option is used, then the channels being updated will be merged into the cue without disturbing other channels already existing in the cue.

Stack *s*
If this option is used, then the updated cue will be recorded into Stack *s*. This option overrides the stack chosen in the current playback fader.

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.

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

> ```
> Update Active Input Cue 5.1
> ```
> Stores (updates) only the active channels from the DMX Input as Cue 5.1.

> ```
> Update Selected Playback 7 Cue 1.23
> ```
> Stores (updates) the currently selected channels from Playback 7 as Cue 1.23.

> ```
> Update Active Playback 1 Stack "Test" Cue 101.5
> ```
> Stores (updates) only the active channels from Playback 1 into Cue 101.5 in the cue stack named “Test”.

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

> ```
> Channel 11>20
> Update Selected Merge Cue 5
> ```
> Stores (updates) Channels 11 through 20 into Cue 5 without disturbing any other previously recorded channels in the cue.

See Also

- **Record Cue**

Update Group

Syntax

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

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

- `[options]`
  - Selection:
    - `All` adds all channels to the group
    - `Empty` removes all channels from the group
    - `Selected` changes the group to contain only the currently selected channels; **this is the default**
    - `Active` changes the group to contain only the currently active channels
  - Source:
    - `Input` causes active channels in the DMX Input to be used as the template for the group
    - `Playback n` causes active channels in Playback *n* to be used as the template for the group
    - `Live` causes active channels in the special *Live* playback fader to be used as the template for the group
  - Other:
    - `Merge` add channels without modifying other existing channels in the group

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, only the group's channels will be updated by using this command. All other group parameters will not be affected.

Record Group Options

Several options are available to change how a group is updated. More than one option may be used, and they can be listed in any order.

All
If this option is used, then the updated group will be created containing every DMX channel. When this group is recalled, it will select all DMX channels.

Empty
If this option is used, then the updated group will be created containing no DMX channels. When this group is recalled, it will deselect all DMX channels.

Selected
If this option is used, then the updated group will be created containing the currently selected DMX channels. This is the default option, so specifying it is redundant.

Active
If this option is used, then the updated group will be created containing only the currently active DMX channels. When updating from the DMX output or input, active channels are any channels that have a non-zero value. When updating from one of the Playbacks, active channels are any channels active in the playback (including zero value channels).

Input
If this option is used, then the updated group will be recorded by using only non-zero values from the DMX Input.

Playback *n*
If this option is used, then the updated group will be recorded by using only active values from the specified playback fader.

Live
If this option is used, then the updated group will be recorded by using only active values from the special *Live* playback fader.

Merge
If this option is used, the updated channels are added to the group without disturbing other existing channels in the group that are not part of the current selection. For instance, if a group previously contained channels 1>10 and only channel 20 is updated with the merge option, then channel 20 is added to the group without removing channels 1>10.

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**

Update Preset

Syntax

Command	Description	Return Value
`Update [options] Preset <preset number>`	Updates the DMX channels in preset *preset number*	The preset number updated

- `<preset number>`
  - Any whole number from `0` to `999`

- `[options]`
  - Source:
    - `Input` causes the DMX Input to be recorded
    - `Playback n` causes only the channels from Playback *n* to be recorded
    - `Live` causes only the channels from the Live playback to be recorded
  - Other:
    - `Zone "z"` record into Zone *z*

Abbreviation

`UP` [`IN`, `P n`, `L`, `ZO "z"`] `PR`

Description

Updating Presets

The **Update Preset** command updates the DMX channels of a preset in the currently active zone. Use this command to change the DMX channels stored in a preset without changing any of the other properties recorded in the preset (such as the fade time and rules).

If no preset with the preset number exists in the active zone, a new preset will be created. If a preset with the preset number already exists, only the preset’s DMX channels will be updated by using this command. All other preset parameters will not be affected.

Update Preset Options

Several options are available to change how a preset is updated. More than one option may be used, and they can be listed in any order.

Input

If this option is used, then the updated preset will be recorded by using only values from the DMX Input. The playback faders and DMX output will not be recorded.

Playback *n*
If this option is used, then the updated preset will be recorded by using only values from the specified playback fader. The DMX input and output will not be recorded.

Live
If this option is used, then the updated preset will be recorded by using only values from the special *Live* playback fader. The DMX input and output will not be recorded.

Zone "*z*"
If this option is used, then the updated preset will be recorded into the Zone named *z*. This option overrides the currently active zone.

Examples

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

> `Update Input Preset 5`
> Stores (updates) the current DMX Input into the CueServer into Preset 5.

> `Update Zone "B" Preset 99`
> Stores (updates) the current output from the CueServer into Preset 99 in the zone named "B".

See Also

- **Record Preset**

Update Stream

Syntax

Command	Description	Return Value
`Update [options] Stream <cue number>`	Updates streaming 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`

- `[options]`
  - `Channel n` causes the stream updating to wait for Channel *n* to become non-zero before recording and return to zero to stop recording
  - `Time t` causes the stream recording to automatically stop after *t* seconds
  - `Stack s` record into Stack *s*

Abbreviation

`UP` [`C n`, `T t`, `STA s`] `STR`

Description

Updating Streaming Cues

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

If no streaming cue with the cue number exists, a new streaming cue will be created. If a streaming cue with the cue number already exists, only the cue's DMX channels will be re-recorded by using this command. All other cue parameters will not be affected.

Update Streaming Cue Options

Several options are available to change how a streaming cue is updated. More than one option may be used, and they can be listed in any order.

Channel *n*

If this option is used, then channel *n* is used as a stream recording “trigger channel”. Recording will wait until channel *n* becomes non-zero. Then, recording will continue until channel *n* returns to a zero value. This option is useful in situations where a particular channel coming from a DMX source has been programmed to signal the precise beginning and end of a DMX clip to record.

Time *t*
If this option is used, then the stream will automatically stop recording after *t* seconds have been captured. If a trigger channel is being used, the timer does not start until the actual recording stream has begun.

Stack *s*
If this option is used, then the new cue will be recorded into Stack *s*. This option overrides the stack chosen in the current playback fader.

Examples

> `Update Stream 1`
> Begins recording the current output from the CueServer as an update to Streaming Cue 1.

> `Update Channel 512 Stream 2`
> Begins recording the current output from the CueServer as an update to Streaming Cue 2, while using Channel 512 as the channel that will trigger the automatic start and stop of the recording.

> `Update Time 3.5 Stream 101.5`
> Begins recording the current output from the CueServer as an update to Streaming Cue 101.5, and the stream will automatically stop recording after 3.5 seconds.

> `Update Time 15 Channel 1024 Stream 42`
> Begins recording the current output from the CueServer as an update to Streaming Cue 42, while using Channel 1024 as the channel that will trigger the automatic start and stop of the recording. If the stream is still recording after 15 seconds, it will automatically stop.

> `Update Stop`
> Stops updating the Streaming Cue.

See Also

- **Record Stream**
- **Record Stop**
- **Update Stop**

Update Stop

Syntax

Command	Description	Return Value
`Update Stop`	Stops updating a streaming cue	The cue number updated

Abbreviation

`UP STO`

Description

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

The **Record Stop** command is an alias for the same command.

Examples

> `Update Stream 1`
> Begins updating the current output from the CueServer into Streaming Cue 1.

> `Update Stop`
> Stops updating the Streaming Cue.

See Also

- **Record Stream**
- **Record Stop**
- **Update Stream**

Unlock

Syntax

Command	Description	Return Value
`Unlock`	Unlocks the selected object(s)	The number of objects unlocked

Abbreviation

None

Description

The **Unlock** command *unlocks* the currently selected object(s). The **Unlock** command can be used with **Buttons**, **Channels**, **Contacts**, **Pages**, and **Stations**. Unlocked buttons or contacts triggers its events normally. If a button has an indicator, it appears in the “normal” state. If a channel is unlocked, it regains the ability to be changed. If a page that requires a PIN number is unlocked, then the user will be able to use the page without manually entering the PIN number. If a station is unlocked, all of its buttons will appear in the “normal” state and/or the lock icon will be removed from the screen. **Unlock** has the opposite effect as **Lock**.

The following table shows the various effect of locking or unlocking an object:

Object	When Unlocked	When Locked
Buttons	Responds to presses normally	Does not trigger any actions, displays *locked* indicator state
Channels	Value in playback fader is changeable by cues, presets, and manual commands	Value in playback fader is not able to be changed
Contacts	Responds to closures normally	Does not trigger any actions
Pages	The user may use the page	The user must enter the correct PIN number to use the page
Stations	Station operates normally	Does not trigger any actions, all controls display *locked* indicator state and/or screen displays lock icon

✻ *Disabling* a button, channel, contact, or station offers similar but different behaviors. See the **Disable** command for more information.

Parking a channel makes a channel unchangeable at a global level instead of just in a single playback. See the **Park** command for more information.

Examples

> `Button 1 Unlock`
> Unlocks button 1.

> `Button 3>5 Unlock`
> Unlocks buttons 3 through 5.

> `Channel 1>10 Unlock`
> Locks channels 1 through 10 in the current playback fader.

> `Page 1 Unlock`
> If page 1 of the current station requires a PIN number, it will become unlocked without the user manually entering the PIN number.

> `Station 7 Unlock`
> Unlocks station 7.

See Also

Button, **Contact**, **Disable**, **Enable**, **Lock**, **Station**

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 pending commands
`Wait Clear`	Causes **all** commands that are currently waiting to be cancelled	The number of cancelled waits
`Wait Stop <id>`	Causes the pending commands with the given *id* to be cancelled	The number of cancelled waits
`Wait ?`	Returns the number of currently waiting commands	A number

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

- `<id>`
  - A decimal number internally assigned to the pending commands. Use this number with the **Wait Stop** command.

Abbreviations

`W`, `WCL`, `WSTO`, `W?`

Description

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

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

The **Wait Stop** command cancels only the commands with the given *id*.

Using the Wait Stop command

To use the **Wait Stop** command, the *id* of the pending commands must be stored in a variable.

For instance, if the following command is executed on the command line:

```
Button 1 On; Wait 5; Button 2 On
```

Button 1 will turn on immediately, then 5 seconds later Button 2 will turn on. A more detailed look at what is happening reveals that the first **Button** command executes immediately, then the **Wait** command executes. When CueScript encounters a **Wait** command, the remainder of the command line is placed in a queue and assigned an *id*. Then, CueScript stops processing commands, returning the *id* of the pending commands.

To cancel the queued commands, the **Wait Stop** command can be used. The **Wait Stop** command requires the *id* of the queued commands to stop. In order to save this special *id* for use with the **Wait Stop** command, the *id* can be placed in a variable. See the following example:

```
"myID" = (Button 1 On; Wait 5; Button 2 On)
```

The commands from the first example are placed in parenthesis. The parenthesis cause the enclosed commands to execute as a single expression. The result of that expression is stored in the variable **myID**.

If after a short delay (for example, 3 seconds later) it is desired to cancel the execution of the “Button 2 On” command that is in the queue to be executed, the following command can be executed:

```
Wait Stop 'myID'
```

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 all currently waiting commands.

> ```
> "stopCue" = (Playback 1 Clear; Wait 10; Cue 1 Go)
> ```
> Clears Playback 1, then waits 10 seconds, then executes Cue 1. Also, the *id* of the pending “Cue 1 Go” is placed in the variable “stopCue”.

> ```
> Wait Stop 'stopCue'
> ```
> Stops the pending “Cue 1 Go” from the previous example. Will return “1” if the command was stopped. Will return “0” if the command was not found in the queue.

Write

Syntax

Command	Description	Return Value
`Write <port> <string>`	Writes the given string to the specified port	The number of characters written
`Write <ip-address> <string>`	Sends the given string via UDP to the specified *ip-address* using the default port of 52737	The number of characters written
`Write <ip-address>:<port> <string>`	Sends the given string via UDP to the specified *ip-address* and *port*	The number of characters written
`Write URL <curl-string>`	Sends the given string via built-in CURL tool	The number of characters written
`Write LCD <string>`	Places the string into the *User String* variable displayed on the LCD	The number of characters written

- `<port>`
  - `COM1` refers to the built-in RS-232 port (on devices with a fixed RS-232 port).
  - `COM2` refers to the built-in RS-485 port (on devices with a fixed RS-232 port).
- `<ip-address>`
  - Any valid ip address (such as `192.168.1.234`)
- `<curl-string>`
  - Any valid CURL request. See Appendix A for additional details about CURL.

Abbreviation

`WR`

Writes (or sends) the given string to the specified serial port, UDP destination, or CURL destination.

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

> ✻ Special non-printing characters can be added to strings by using *escape sequences*, such as “\n” for newline, “\r” for carriage return, etc. See the strings section for more information about escape sequences.

Examples

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

> `Write COM2 "This is a test\r"`
> Sends the string “This is a test” followed by a carriage return character to the RS-485 port.

> `Write "10.0.1.5" "Cue 1 Go"`
> Sends the string “Cue 1 Go” via UDP to the ip address “10.0.1.5” and the default port of 52737.

> `Write "10.0.1.5:1234" "Mission Accepted\x00"`
> Sends the string “Mission Accepted” followed by a NULL byte via UDP to the ip address “10.0.1.5” and the port of 1234.

> `Write URL "http://10.0.1.5/request"`
> Sends an HTTP GET request to 10.0.1.5 to fetch the “/request” URL.

> `Write URL "--user-agent 'MyAgent' -d 'Hello World' http://10.0.1.5/cgi-bin/post"`
> Sends an HTTP POST request to 10.0.1.5 to fetch the “/cgi-bin/post” URL. The data “Hello World” will be posted.

Zone

Syntax

Command	Description	Return Value
`Zone "<zone name>"`	Changes the current zone	The zone name
`Zone Clear` or `Zone ""`	Cancels the current zone	*none*
`Zone ?`	Returns the current zone	The zone name

- `<zone name>`
  - The alphanumeric name of a zone.
  - Zone names must be enclosed in quotes.
  - May be empty (“”) to clear the current zone.

Abbreviation

`ZO`

Description

Use the **Zone** command to select a zone. Zones are used to separate different areas of a particular project. Zones are also “parent containers” for presets, meaning that any preset must be a member of a zone.

When a zone is selected, only channels within that zone can be changed. For example, a zone named “Meeting Room” only contains channels 10 through 19. If that zone is selected, then any subsequent commands that might change channel values (such as **On**, **Off**, **At**, **Go**, **Release**, **Toggle**, etc.) will ***only*** operate on channels 10 through 19, regardless of which channels are being targeted by the Cue, Group, Preset, etc.

To remove the restrictions of having a zone selected, use the **Zone Clear** command. Once the zone is cleared *all* channels will again respond to the various channel setting commands.

In certain applications, it is useful to be able to join multiple zones together (such as in a Ballroom with removable “air walls”). See the **Join** command to learn how to join Zones together. When zones are joined, the channel setting commands can affect the combined channels of all joined zones.

Examples

> `Zone "Ballroom A"`
> Selects the zone named “Ballroom A”.

> `Zone "Foyer" Preset 3 On`
> Activates Preset 3 in the Zone “Foyer”.

> `Zone "Theater" Channel 1>100` FL@
> Selects the “Theater” zone and then attempts to set channels 1 through 100 to Full. Only the channels defined in the zone will actually be set to Full.

> `Zone Clear`
> Remove the restrictions of the current zone.

> `Zone ?`
> Returns the currently selected zone.

See Also

- **At**, **Join**, **Preset**

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
> EndIf
> Button 1 On
> ```
> 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.

> ```
> Cue 1 Go; `continue`; Button 1 On
> ```
> Executes Cue 1, then the contents of the variable *continue* are executed in place. If *continue* contains the value “Break” then execution stops here. If *continue* is empty, then execution continues on to “Button 1 On”. Note that the \`accent-quotes\` are used around the variable name to *execute* the contents of the variable.

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. Since no commands are present after the "Cue 1 Go", an **EndIf** is not necessary.

See the section on Expressions for more information about the kinds of expressions that can follow an **If** statement.

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. Since no commands are present after the “Cue 2 Go”, an **EndIf** is not necessary.

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

<hr>
```

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
```

Effect Properties

The properties of Playback Effects can be manipulated using CueScript. This allows a show's automation to dynamically change the type, rate, direction, and other effect properties to be changed based on input from the user, or other logic and triggers.

Selecting a Playback and Effect Slot

First, target a Playback Fader and a specific Effect slot using the **Playback** and **Effect** commands. For example:

```
Playback 3
Effect 1
```

Changing Effect Properties

Once an Effect in a Playback Fader is selected, use the **Set** command or the assignment operator to change effect parameter values. The following examples show multiple ways to do this:

```
// Set the effect type to 1 (Hue Rotate) using Set command
Set effect.type 1

// Set the effect type to 1 (Hue Rotate) using assignment
"effect.type" = 1
```

Effect Property Listing

The following properties are defined to allow CueScript to manipulate effects.

Property	Description
`angle`	The current angle of the effect cycle in degrees, from `0.0` to `359.999`. As an effect is animating it's angle rotates continuously from 0 degrees around to 360 degrees during each cycle. When the effect reaches 360 degrees, it wraps back around to 0 degrees. Some effects use this angle as an offset from the starting point, other effects use this as a virtual clock to trigger an event, such as a spark or twinkle. When an if running in reverse, it's angle decreases to 0 and wraps around to 360.
`attack`	The amount of time it takes for an aspect of the effect to reach it's maximum intensity. May be any duration from `0.0` to `300.0` seconds. Only applicable to certain effects such as Sparkle or Twinkle.

`bpm`	The rate of the effect in beats per minute (BPM). May be a decimal number from `0.0` to `2399.999`. Changing the BPM of an effect also changes its *rate* and *period* properties.
`decay`	The amount of time it takes for an aspect of the effect to reach it's minimum intensity. May be any duration from `0.0` to `300.0` seconds. Only applicable to certain effects such as Sparkle or Twinkle.
`group`	An optional *group number* that should be used to filter which channels the effect is operating on. May be any group number from `1` through `99999`. Use `0` to indicate that the effect should not be filtered by any group.
`intensity`	The relative intensity of the effect. A decimal number from `0.0` through `1.0` is used to indicate minimum (off) intensity to full (on) intensity.
`period`	The rate of the effect in seconds. May be a decimal number from `0.0` to `3600.000`. Changing the Period of an effect also changes its *bpm* and *rate* properties.
`rate`	The rate of the effect in cycles per second (Hz). May be a decimal number from `0.0` to `39.999`. Changing the Rate of an effect also changes its *bpm* and *period* properties.
`reverse`	Determines the direction of an effect's cycle. May be `0` for forward direction, or `1` for reverse direction.
`type`	The effect type. Choose from one of the following: `0` = None `1` = Hue Rotate `2` = Sparkle `3` = Twinkle

Examples

The following example places a Sparkle effect on Playback 1.

```
/* Start a Sparkle Effect in Playback 1 */
Playback 1; Effect 1
"effect.type" = 2 // Sparkle
"effect.bpm" = 120 // 120 beats per minute
"effect.attack" = 0.25 // 1/4 second rise time
"effect.decay" = 2 // 2 second fall time
"effect.group" = 101 // Only apply to channels in group 101
```

The following example takes an existing effect in Playback 1 and dynamically changes its rate and direction.

```
/* Speed Up Sparkle and Increase Sparkyness over 3 seconds*/
Playback 1; Effect 1
```

```
"effect.bpm" = 150
"effect.attack" = 0.15
"effect.decay" = 2.5
Wait 1.5
"effect.bpm" = 180
"effect.attack" = 0.10
"effect.decay" = 3
Wait 1.5
"effect.bpm" = 240
"effect.attack" = 0
"effect.decay" = 4
```

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:

Astro

Accesses the astronomical time parameters of CueServer.

`astro.light`	Gets wether it is currently “light” out. A returned value of `1` means that it is light, `0` means that it is dark. Read-Only
`astro.phase`	Gets the current sun phase. One of the following values will be returned: `0` = The sun phase is unknown `1` = Night `2` = Twilight `3` = Nautical Twilight `4` = Civil Twilight `5` = Daylight Read-Only
`astro.latitude`	Gets the current system latitude in degrees. A number from `-90` to `90` will be returned. Read-Only
`astro.longitude`	Gets the current system longitude in degrees. A number from `-180` to `180` will be returned. Read-Only
`sunrise.hour`	Gets today’s current sunrise hour. A number from `0` to `23` will be returned. Read-Only
`sunrise.minute`	Gets today’s current sunrise minute. A number from `0` to `59` will be returned. Read-Only
`sunrise.second`	Gets today’s current sunrise second. A number from `0` to `59` will be returned. Read-Only
`sunrise.time`	Gets today’s current sunrise time. A formatted as `"HH:MM:SS"` will be returned. Read-Only
`sunset.hour`	Gets today’s current sunset hour. A number from `0` to `23` will be returned. Read-Only
`sunset.minute`	Gets today’s current sunset minute. A number from `0` to `59` will be returned. Read-Only
`sunset.second`	Gets today’s current sunset second. A number from `0` to `59` will be returned. Read-Only
`sunset.time`	Gets today’s current sunset time. A formatted as `"HH:MM:SS"` will be returned. Read-Only

Examples:

```
Log 'astro.phase'
```

The example above logs the current sun phase to the system log.

```
If ('astro.light') Then Cue 1 Go
```

The example above executes Cue 1 if it is currently light outside.

```
Set lcd.bottomLeft "Today's Sunset: ${sunset.time}"
```

The example above displays today's subnet time on the LCD.

Audio

Sets the output volume for the Audio Output jack.

`audio.volume`	Sets the line level output of the Audio Output jack. Available levels range from `0` to `100`. A value of `0` produces no output. The default value is `90`.

Example:

```
Set audio.volume 50
```

The example above sets the audio volume of the Audio Output jack to 50%.

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”.

Clock

Provides access to the real-time clock.

`clock.date`	Gets or sets the current date and time. When querying the date variable, a string such as `Mon Jan 01 18:30:59 EST 2018` will be returned. When setting the date variable, a variety of formats are supported including `MM/DD/YY`, `HH:MM:SS`, `YYYY-MM-DD HH:MM:SS`, `YYYYMMDD HH:MM`, `next year`, `last friday`, and many others. See the Linux documentation on the *date* command for more information.
`clock.second`	Gets or sets the current Second. Valid values range from `0` to `59`.
`clock.minute`	Gets or sets the current Minute. Valid values range from `0` to `59`.
`clock.hour`	Gets or sets the current Hour. Valid values range from `0` to `23`.
`clock.day`	Gets or sets the current Day. Valid values range from `1` to `31`.
`clock.month`	Gets or sets the current Month. Valid values range from `1` to `12`.
`clock.year`	Gets or sets the current Year. Valid values range from `1900` to `2999`.
`clock.weekday`	Gets the number of days since Sunday. Valid values range from `0` to `6`. *This is a read-only property.*
`clock.yearday`	Gets the number of days since January 1. Valid values range from `0` to `365`. *This is a read-only property.*
`clock.dst`	Returns `1` if Daylight Saving Time is currently in effect, and `0` if not. Can return `-1` in the case where the DST status cannot be determined. Read-Only

clock.zone	Gets or sets the time zone. When querying this variable, a string such as `America/New York` will be returned. When setting this variable, be sure to provide a time zone in the format of `<region>/<location>` from CueServer's available listing of [time zones](). Zones that have spaces in their names can use either space characters (such as `America/New York`, or underscore characters (such as `America/New_York`).

Examples:

```
Set clock.date "1/1/18 12:00:00"
```

The example above sets the time and date to January 1, 2018 at noon.

```
"clock.zone" = "America/Los_Angeles"
```

The example above sets the time zone to "America/Los Angeles"

```
if ('clock.hour' > 12) then
    Cue 1 Go
endif
```

The example above executes Cue 1 only if the hour is greater than 12.

Please note that when setting the time or date, if the CueServer had previously been configured to receive its time via an NTP server, the system will switch to "manual" time mode.

Debug

Enables various debugging functions. Each of the following variables can be set to `1` to enable the function and `0` to disable the function.

debug.all	Enables/disables *all* of the individual diagnostic functions (see below).
debug.buttons	Enables/disables system logging of button and contact related events (both built-in buttons/contacts and CueStation buttons/contacts).
debug.cue	Enables/disables system logging of all Cue related events processed by the system.
debug.cuescript	Enables/disables system logging of all CueScript commands processed by the system.
debug.javascript	Enables/disables system logging of the JavaScript `print()` function.

`debug.rtc`	Enables/disables system logging of real-time clock related events.
`debug.show`	Enables/disables system logging of show related events.
`debug.triggers`	Enables/disables system logging of trigger functions.
`debug.udp`	Enables/disables system logging of all UDP packets received on the CueScript port.
`debug.variables`	Enables/disables system logging of all changes to variable values.

Example:

```
Set debug.udp 1
Set debug.cuescript 0
```

The example above enables UDP event logging and disables CueScript event logging.

> Please note that all debugging functions are reset to “off” when a CueServer is power-cycled. Each desired function must be re-enabled each time a CueServer is rebooted. Also, it is not recommended to leave a debug log function enabled indefinitely, as some of these functions can result in the system log overflowing.

Device

Returns various hardware specific properties.

`device.ip`	Gets the device’s primary IP address. Read-Only
`device.ipA`	Gets the device’s IP address for LAN A. Read-Only
`device.ipB`	Gets the device’s IP address for LAN B. Read-Only
`device.gateway`	Gets the device’s gateway address. Read-Only
`device.model`	Gets the device’s model string. Read-Only
`device.name`	Gets the device’s name string. Read-Only
`device.serial`	Gets the device’s serial number. Read-Only
`device.subnet`	Gets the device’s primary subnet address. Read-Only
`device.subnetA`	Gets the device’s subnet address for LAN A. Read-Only
`device.subnetB`	Gets the device’s subnet address for LAN B. Read-Only

Examples:

```
Log 'device.model'
```

The example above logs the device's model string to the system log.

```
If ('device.serial' == "640123") Then Log "Hello World"
```

The example above adds the entry "Hello World" to the system log if the device's serial number is 640123.

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.

Panel

Changes properties of the front-panel of the device.

`panel.brightness`	Sets the overall brightness of the front-panel function button indicators and the navigation switch backlight. Brightness values range from `0` to `100`.

Example:

```
Set panel.brightness 33
```

The example above sets the overall front-panel brightness to 33% of its maximum brightness.

Playbacks

Changes properties of a Playback fader.

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

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.

Show

Gets various show related properties.

`show.channels`	Gets the number of configured channels in the active show. Read-Only
`show.name`	Gets the name of the active show. Read-Only
`show.path`	Gets the pathname of the active show on the SD Card. Read-Only
`show.playbacks`	Gets the number of configured playbacks of the active show. Read-Only
`show.ports`	Gets the number of configured ports of the active show. Read-Only
`show.universes`	Gets the number of configured universes of the active show. Read-Only

Example:

```
Log "The active show is ${show.name}"
```

The example above logs the message “The active show is *MyShowName*”, where *MyShowName* will be the actual active show name.

Universes

Sets properties of the DMX Universes.

`universe.priority`	This is an *alias* for `universe.txpriority`.
`universe.rxpriority`	Sets the both the higher and lower limit of sACN priorities that will be received by this universe to the same value. Any received data with a priority different from this number will be rejected. Available values range from `0` to `200`.
`universe.rxpriorityhigh`	Sets the higher limit of sACN priorities that will be received by this universe. Any received data with a priority higher than this number will be rejected. Available values range from `0` to `200`.
`universe.rxprioritylow`	Sets the lower limit of sACN priorities that will be received by this universe. Any received data with a priority lower than this number will be rejected. Available values range from `0` to `200`.
`universe.txpriority`	Sets the priority level of sACN transmissions from this universe. Available values range from `0` to `200`.

Example:

```
Universe 7
Set universe.txpriority 150
```

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

```
Universe 5+7
Set universe.rxpriority 42
```

The example above sets the sACN receive priority of Universes 5 and 7 to 42. These universes will only receive data with a priority of 42.

```
Universe 1>4
Set universe.rxprioritylow 175
```

The example above sets the sACN receive priority of Universes 1 thru 4 to 175. These universes will no longer receive data with a priority 174 or lower.

Internals

- Web Server
- CGI API
- Show File Format

Web Server

Environment Variables

The following *environment variables* are defined in the built-in Apache 2 web server. These variables are available for use within custom HTML pages and/or CGI type scripts being served from CueServer.

Variable	Description	Example
`SHOW_NAME`	The name of the current show file	`My First Show`
`SHOW_UNIVERSES`	The number of currently configured universes	`8`
`SHOW_CHANNELS`	The number of currently configured channels	`4096`
`SHOW_PLAYBACKS`	The number of currently configured playback faders	`16`
`DEVICE_MODEL`	The model number of the device	`CS-940 (Rev. A)`
`DEVICE_NAME`	The assigned name of the device	`CueServer 2`
`DEVICE_SERIAL`	The serial number of the device	`601234`

Using Environment Variables with SSI

Apache environment variables can be used in HTML by utilizing *Server Side Includes* (SSI).

In the HTML code of the page, a special SSI tag can be included that Apache will automatically substitute into the HTML when the page is served from the server. The SSI tag looks like this:

```
<!--#echo var="DEVICE_NAME" -->
```

By default, SSI is not enabled on HTML pages. SSI can be enabled in one of several ways.

- To enable SSI for a single page, the extension of the HTML document can be changed to **.shtml**. The **.shtml** extension causes Apache to process the SSI tags inside the HTML content of the document.
- To enable SSI for a directory, a **.htaccess** file can be created in the directory that includes the `Options +Include` directive.
- To enable SSI site-wide, the Apache configuration can be modified.

CGI API

CueServer includes an embedded web server that responds to HTTP requests. In addition to the standard URLs that a user of the Web Interface would see, a special set of URLs are available in CueServer that can be used to run CueScript commands, fetch real-time information from CueServer, set operating parameters and more.

This section describes the various URLs that are available and their function.

Command Throttling (IMPORTANT)

It is very imporant to employ a throttling mechanism when sending HTTP requests. **New requests should only be sent after a response to the previous request has been received from CueServer.** This allows CueServer to manage requests and responses gracefully and it prevents your code from sending a large number of commands blindly in a very short period of time (for example, moving a slider in a rapid fashion). Not implementing a throttling mechanism could result in many dropped requests and commands and/or performance issues affecting other functions of CueServer due to high CPU usage.

CueServer CGI URLs:

- exe.cgi
- get.cgi
- pcmd.cgi
- set.cgi

exe.cgi

The `exe.cgi` URL is used to execute CueScript commands on the CueServer.

The typical format of this URL is:

```
http://<ip-of-CueServer>/exe.cgi?cmd=<command>&<optional-parameters>
```

For example, the following URL will execute the command `Cue 1 Go`:

```
/exe.cgi?cmd=Cue+1+Go
```

Note that when a command is URL-encoded, spaces must be changed to plus (`+`) characters, and other “special” characters must use standard URL escaping methods, for example a `$` character should be encoded as `%24`. See Percent Encoding on Wikipedia for more details.

Parameters

- `cmd=<string>`
  - A CueScript command.
  - Special characters must be Percent Encoded

- `def=<defaultPlayback>` *(optional)*
  - `1` to `32` specifies the playback that the command will default to.
  - If this parameter is not specified, then no change will be made to the default playback for this context (i.e., the playback will remain the same as it was after a previous command was executed within this context).

- `usr=<contextID>` *(optional)*
  - `-1` specifies that a temporary context should be used.
  - `0` specifies the default context (same as used by CueServer Studio).
  - `1` to `4` specifies User 1 thru User 4 contexts (for multi-user input).
  - `5` specifies the Ethernet context.
  - `6` specifies the Serial context.
  - `7` specifies the Rule Actions context.
  - If this parameter is not specified, then a temporary context will be used.

Examples

CueScript Command	URL

`M1`	`/exe.cgi?cmd=M1`
`Cue 73 Go`	`/exe.cgi?cmd=Cue+73+Go`
`WRITE "Hello World!"`	`/exe.cgi?cmd=WRITE+%22Hello+World%21%22`
`Clear`, on Playback 3	`/exe.cgi?cmd=Clear&def=3`
`Button 1.5 Off`, in Context 0	`/exe.cgi?cmd=Button+1.5+Off&usr=0`

get.cgi

The `get.cgi` URL is used to fetch information from the CueServer.

The typical format of this URL is:

```
http://<ip-of-cueserver>/get.cgi?req=<request>&<optional-parameters>
```

Depending on the value of the `<request>` parameter, this URL can fetch many different pieces of information from CueServer.

The following variations of the `get.cgi` URL are available:

- Button Values [bv]
- Command Context [cc]
- CPU SysInfo [cpu]
- Cue Stack Info [csi]
- DMX Input [in]
- DMX Output [out]
- Extended Command Context [ecc]
- Extended Playback Info [epi]
- Fade Engine Data [fed]
- Group Level [grp]
- Hardwired DMX Input [hdi]
- Network Info [net]
- Ping [ping]
- Playback Info [pi]
- Playback Values [p*]
- Preset Zone Info [pzi]
- Record Stream Info [rs]
- System Log [log]
- System Status [ss]
- Time Info [ti]
- Time Status [ts]
- Variables [var]
- Zone Data [zones]

Button Values [bv]

This request returns the current state of of the CueServer's front-panel buttons.

! This request is available in CueServer 2 only for compatibility with the original CueServer 1 API. Use of this request is **depreciated** and is not encouraged.

This request only returns the 8-bit indicator state of the first eight buttons of each of the first 64 stations (as this is what CueServer 1 was limited to).

URL:

```
/get.cgi?req=bv
```

Response:

This request will return 520 bytes. Each byte is an 8-bit color value for each of the 8 buttons on the first 64 button stations (stations 1 thru 64) as defined in CueServer. The final eight bytes returned correspond to the built-in station (station 0).

If there is no "Station 1" defined in the current show file, then the built-in buttons (station 0) will also appear in the first 8 bytes of the returned data.

This structure is provided for compatibility applications that are expecting this format of data as it was previously provided by CueServer 1. Note that CueServer 1 only returned 512 bytes, corresponding to its stations 1 thru 64 (there was no "Station 0") in CueServer 1.

Errors:

If an error occurs during the processing of the request, only a single byte will be returned by this URL. The value of the returned byte is explained in the following table:

Error Value	Description
`0xFF`	An internal memory error occurred.

Command Context [cc]

This request returns a *Command Context* data structure for the specified command context. This structure contains detailed information about a command context.

! This request is available in CueServer 2 only for compatibility with the original CueServer 1 API. Use of this request is **depreciated** and is not encouraged. Use Extended Command Context [ecc] instead.

The CC selector was originally designed for CS1 and therefore is restricted to fixed point fade times and only up to 512 bits of the select buffer. This selector is provided for compatibility only and is not recommended for use with CS2.

URL:

`/get.cgi?req=cc`

Response:

The following data structure will be returned by this request.

```
typedef struct CCResult {       // (80 bytes)
    uint8_t             curPlayback;            // Current Playback (1..32)
    uint8_t             reserved1;              // -
    uint8_t             curTarget;              // Current Target
    uint8_t             isDMXTarget;            // Is the target a Channel-based object?
    uint16_t            fadeDownTime;           // Fade down time in tenth-second increments
    uint16_t            fadeUpTime;             // Fade up time in tenth-second increments
    uint16_t            delayDownTime;          // Delay down time in tenth-second increments
    uint16_t            delayUpTime;            // Delay up time in tenth-second increments
    uint8_t             reserved2[4];           // -
    uint8_t             selectBuf[64];          // Selection buffer
} CCResult;
```

Errors:

If an error occurs during the processing of the request, only a single byte will be returned by this URL. The value of the returned byte is explained in the following table:

Error Value	Description
`0xFF`	An internal shared memory error occurred.
`0xFE`	An internal memory error occurred.

CPU SysInfo [cpu]

This request returns the CueServer's internal operating system *SysInfo* structure. This structure contains detailed information about the device's uptime, CPU load, RAM usage, processes and more.

URL:

`/get.cgi?req=cpu`

Response:

The following data structure will be returned by this request.

```
typedef struct SysInfo {        // (64 bytes)
    uint32_t            uptime;                 // Seconds since boot
    uint32_t            loads[3];               // 1, 5, and 15 minutes load aver
ages
    uint32_t            totalRAM;               // Total usable main memory size
    uint32_t            freeRAM;                // Available memory size
    uint32_t            sharedRAM;              // Amount of shared memory
    uint32_t            bufferRAM;              // Memory used by buffers
    uint32_t            totalSwap;              // Total swap space size
    uint32_t            freeSwap;               // Swap space still available
    uint16_t            procs;                  // Number of current processes
    uint16_t            reserved1;              // -
    uint32_t            totalHigh;              // Total high memory size
    uint32_t            freeHigh;               // Available high memory size
    uint32_t            memUnit;                // Memory unit size in bytes
    uint8_t             reserved2[8];           // -
} SysInfo;
```

Errors:

If an error occurs during the processing of the request, only a single byte will be returned by this URL. The value of the returned byte is explained in the following table:

Error Value	Description
`0xFF`	An unspecified error occurred.

Cue Stack Info [csi]

This request returns a *Cue Stack Info* data structure for the specified cue stack. Use this response to determine which cues are active in a given cue stack.

URL:

`/get.cgi?req=csi&name=<stackName>`

Parameters:

- `name=<stackName>` *(optional)*
  - The name of the desired cue stack.
  - If this parameter is not supplied, the default cue stack is assumed.

Response:

The following variable-length structure is returned by this request:

```
#define STACK_NAME_BUF_SIZE     16
#define CSI_TYPE_CUES           0
#define CSI_TYPE_PRESETS        1

typedef struct CueStackInfo {
    uint16_t            signature;              // Signature = 'CS'
    int16_t             version;                // Version = 0x0001 (or negative
error code)
    char                stackName[STACK_NAME_BUF_SIZE]; // Name of stack
    uint8_t             type;                   // 0 = Cues, 1 = Presets
    uint8_t             playback;               // The playback number (for prese
ts only)
    uint16_t            count;                  // Number of CueID/Status pairs
    uint32_t            data[64];               // Array of CueID/Status pairs (3
2 pairs max)
} CueStackInfo;
```

Please note that the actual number of bytes returned by this request only includes the actual number of pairs of `uint32_t` elements in the `data[]` array.

The CueID/Status pairs are included for any cue in the cue stack that is active in a playback fader. The CueID denotes the cue number and the Status value indicates in which playback fader the cue is active in. The CueID may have the value `0x40000000` added to it to indicate that the cue is active but modified.

Errors:

If an error occurs during the processing of the request, only the first four bytes of the above structure will be returned by this URL. The first two bytes will have the “CS” signature and the next two bytes will contain an error value as described by the following table:

Error Value	Description
`-1`	An internal shared memory error occurred.

DMX Input [in]

This request returns an array of DMX Input channels. The result contains channel values from *both* Ethernet-based DMX input and the hardwired DMX input ports.

URL:

`/get.cgi?req=in&index=<0..16383>&count=<0..16384>&pad=<0,1>`

Parameters:

- `index=<0..16383>` *(optional)*
  - Specifies the starting channel index of the returned array of values.
  - If this parameter is not supplied, the default value is `0`.

- `count=<0..16384>` *(optional)*
  - Specifies the number of channels to return values for.
  - If this parameter is not supplied, all channels up to and including the highest configured channel will be returned.

- `pad=<0,1>` *(optional)*
  - If given as `0`, the returned data will *not* be padded, only the requested channels will be returned.
  - If given as `1`, the returned data will be padded with extra zero bytes (`0x00`) to satisfy the `count` parameter, even if those channels do not exist in the current configuration.

Response:

The data returned is an array of bytes, each one corresponding to the requested channels. Each byte ranges from zero (`0x00`) to 100% (`0xFF`).

Errors:

If an error occurs during the processing of the request, only a single byte will be returned by this URL. The value of the returned byte is explained in the following table:

Error Value	Description
`0xFF`	An internal shared memory error occurred.
`0xFE`	A fade engine error occurred.
`0xFD`	An internal memory error occurred.

DMX Output [out]

This request returns an array of DMX Output channels. The result contains channel values from the final output stage of the playback stack.

URL:

`/get.cgi?req=out&index=<0..16383>&count=<0..16384>&pad=<0,1>`

Parameters:

- `index=<0..16383>` *(optional)*
  - Specifies the starting channel index of the returned array of values.
  - If this parameter is not supplied, the default value is `0`.

- `count=<0..16384>` *(optional)*
  - Specifies the number of channels to return values for.
  - If this parameter is not supplied, all channels up to and including the highest configured channel will be returned.

- `pad=<0,1>` *(optional)*
  - If given as `0`, the returned data will *not* be padded, only the requested channels will be returned.
  - If given as `1`, the returned data will be padded with extra zero bytes (`0x00`) to satisfy the `count` parameter, even if those channels do not exist in the current configuration.

Response:

The data returned is an array of bytes, each one corresponding to the requested channels. Each byte ranges from zero (`0x00`) to 100% (`0xFF`).

Errors:

If an error occurs during the processing of the request, only a single byte will be returned by this URL. The value of the returned byte is explained in the following table:

Error Value	Description
`0xFF`	An internal shared memory error occurred.
`0xFE`	An improper channel range was specified.

Extended Command Context [ecc]

This request returns an *Extended Command Context* data structure for the specified command context. This structure contains detailed information about a command context.

URL:

`/get.cgi?req=ecc&id=<contextID>`

Parameters:

- `id=<contextID>` *(optional)*
    - `0` specifies the default context (same as used by CueServer Studio) [Default].
    - `1` to `4` specifies User 1 thru User 4 contexts (for multi-user input).
    - `5` specifies the Ethernet context.
    - `6` specifies the Serial context.
    - `7` specifies the Rule Actions context.

Response:

The following variable-length data structure will be returned by this request.

```
typedef struct ECCDataV3 {
    uint16_t            signature;              // Signature = 'CS'
    int16_t             version;                // Version = 0x0003 (or negative
error code)
    uint8_t             curPlayback;            // Current Playback (1..32)
    uint8_t             curTarget;              // Current Target
    uint8_t             isDMXTarget;            // Is curTarget a DMX channel sel
ection?
    uint8_t             reserved[13];           // -
    FadeTimes           fadeTimes;              // Current Fade Times
    char                stackName[STACK_NAME_BUF_SIZE];     // Name of current pl
ayback's stack
    char                zoneName[STACK_NAME_BUF_SIZE];      // Name of current zo
ne
    uint8_t             variableData[];         // Selection buffers (see below)
//  uint16_t            sizeofSelectData;       // Number of bytes in selectData
//  uint8_t             selectData[];           // RLE compressed selection bitma
sk (max 2064 bytes)
//  uint16_t            sizeofMaskData;         // Number of bytes in maskData
//  uint8_t             maskData[];             // RLE compressed mask bitmask (m
ax 2064 bytes)
//  uint16_t            sizeofStationData;      // Number of bytes in stationData
//  uint8_t             stationData[];          // RLE compressed station bitmas
k (max 129 bytes)
} ECCDataV3;
```

Errors:

If an error occurs during the processing of the request, only the first four bytes of the above structure will be returned by this URL. The first two bytes will have the “CS” signature and the next two bytes will contain an error value as described by the following table:

Error Value	Description
`-1`	An internal shared memory error occurred.
`-2`	An invalid contextID was given.

Extended Playback Info [epi]

This request returns one or more *Extended Playback Info* data structures for specified playback faders. This structure contains detailed information about the current status of each playback fader.

URL:

`/get.cgi?req=epi&id=<playbackID>`

Parameters:

- `id=<playbackID>`
  - `1` to `32` specifies an individual playback fader to fetch information for.
  - `0` returns a string of multiple data structures for each of the active playback faders.

Response:

The following data structure will be returned by this request.

```
typedef struct EPIData {        // (160 bytes total)
    uint8_t             version;                // Result Version = 0x02
    uint8_t             playback;               // Playback number (1..32)
    uint8_t             flags;                  // Flags (0 = Normal, 1 = Stopped, -1 = Not Installed)
    uint8_t             mode;                   // Mode (0 = Merge, 1 = Override, 2 = Scale, 3 = Pin)
    uint8_t             reserved1[4];           // -

    int32_t             curCueID;               // Current Cue ID (0..MAX_CUE_NUMBER, -1=CUE_NONE, -2=CUE_ACTIVE_CHANNELS)
    int32_t             nextCueID;              // Next Cue ID to "Go" to
    int32_t             linkCueID;              // Link Cue ID to link to

    FadeTimes           fadeTimes;              // Current fade/split/delay times
    float               followTime;             // Follow time for next cue go (0 = Do Not Auto-Follow)
    uint8_t             submaster;              // Submaster
    uint8_t             reserved2[3];           // -

    uint32_t            fadeCurTime;            // Fade progress
    uint32_t            fadeTotalTime;          // Fade total time

    float               followTimeRemain;       // Follow progress
    float                   followTotalTime;        // Follow total time

    uint32_t                streamCurTime;          // Stream playback position (tick
s)
    uint32_t                streamTotalTime;        // Stream total time (ticks)

    uint8_t                 reserved3[12];          // -

    char                    stackName[STACK_NAME_BUF_SIZE]; // Name of current stack
    char                    curCueName[32];                 // Name of current cue
    char                    nextCueName[32];                // Name of next cue
} EPIData;
```

Errors:

If an error occurs during the processing of the request, only a single byte will be returned by this URL. The value of the returned byte is explained in the following table:

Error Value	Description
0xFF	An internal memory error occurred.
0xFE	Invalid playback number was specified.

Fade Engine Data [fed]

This request returns a *Fade Engine Data* data structure. This structure contains detailed information about all playbacks, all universes and all ports simultaneously.

URL:

`/get.cgi?req=fed`

Response:

The `FadeEngineData` structure is variable length. The header of 16 bytes is followed by a variable number of `EPIData`, `UniverseData`, and `PortData` records. The total length of a maximum of 32 playbacks, 128 universes, and 4 ports is currently 8,240 bytes. This may grow in future versions.

```
typedef struct FadeEngineData { // (16 bytes)
    uint16_t            signature;              // Signature = 'CS'
    int16_t             version;                // Version = 0x0002 (or negative error code)
    uint8_t             playbacks;              // Number of EPIData records (0..32)
    uint8_t             playbackSize;           // Size of EPIData record (currently 160)
    uint8_t             universes;              // Number of UniverseData records (0..128)
    uint8_t             universeSize;           // Size of UniverseData record (currently 24 bytes)
    uint8_t             ports;                  // Number of PortData records (0..4)
    uint8_t             portSize;               // Size of PortData record (currently 8 bytes)
    uint8_t             reserved[6];            // -
    uint8_t             variableData[];         // EPIData, UniverseData, and PortData records start here
} FadeEngineData;

typedef struct EPIData {        // (160 bytes total)
    uint8_t             version;                // Result Version = 0x02
    uint8_t             playback;               // Playback number (1..32)
    uint8_t             flags;                  // Flags (0 = Normal, 1 = Stopped, -1 = Not Installed)
    uint8_t             mode;                   // Mode (0 = Merge, 1 = Override, 2 = Scale, 3 = Pin)
    uint8_t             reserved1[4];           // -

    int32_t             curCueID;               // Current Cue ID (0..MAX_CUE_NUM
BER, -1=CUE_NONE, -2=CUE_ACTIVE_CHANNELS)
    int32_t             nextCueID;              // Next Cue ID to "Go" to
    int32_t             linkCueID;              // Link Cue ID to link to

    FadeTimes           fadeTimes;              // Current fade/split/delay times
    float               followTime;             // Follow time for next cue go
(0 = Do Not Auto-Follow)
    uint8_t             submaster;              // Submaster
    uint8_t             reserved2[3];           // -

    uint32_t            fadeCurTime;            // Fade progress
    uint32_t            fadeTotalTime;          // Fade total time

    float               followTimeRemain;       // Follow progress
    float               followTotalTime;        // Follow total time

    uint32_t            streamCurTime;          // Stream playback position (tick
s)
    uint32_t            streamTotalTime;        // Stream total time (ticks)

    uint8_t             reserved3[12];          // -

    char                stackName[STACK_NAME_BUF_SIZE]; // Name of current stack
    char                curCueName[32];                 // Name of current cue
    char                nextCueName[32];                // Name of next cue
} EPIData;

typedef struct UniverseData {   // (24 bytes)
    uint8_t             version;                // Result Version = 0x02
    uint8_t             universe;               // Universe number (1..128)
    uint16_t            channelIndex;           // Starting channel index
    uint16_t            channelCount;           // Width of universe
    uint8_t             rxProtocol;             // Rx Protocol
    uint8_t             reserved;               // -
    int16_t             rxChannels;             // Rx Channels (0..512, -1)
    uint16_t            rxUniverse;             // Rx Universe
    uint32_t            rxExtra;                // Rx Extra Data (port number fo
r KiNET v2)
    uint8_t             txProtocol;             // Tx Protocol
    uint8_t             txEnabled;              // Tx Enabled
    uint16_t            txUniverse;             // Tx Universe
    uint32_t            txExtra;                // Tx Extra Data (port number fo
r KiNET v2)
} UniverseData;

typedef struct PortData {        // (8 bytes)
    uint8_t             version;                // Result Version = 0x01
    uint8_t             port;                   // Port number (1..4)
    uint8_t             universe;               // Universe number (1..32)
    uint8_t             direction;              // Data direction
    uint8_t             led;                    // LED Indicator value
    uint8_t             reserved1;              // -
    int16_t             channels;               // Tx/Rx Channels (0..512, -1)
} PortData;
```

Errors:

If an error occurs during the processing of the request, only the first four bytes of the above structure will be returned by this URL. The first two bytes will have the “CS” signature and the next two bytes will contain an error value as described by the following table:

Error Value	Description
-1	An internal shared memory error occurred.
-2	An internal memory error occurred.

Group Level [grp]

This request returns the current level of a specified channel group.

URL:

`/get.cgi?req=grp&id=<groupID>&p=<playback>`

Parameters:

- `id=<groupID>`
    - `0` to `99999` specifies the group ID (number).

- `p=<playback>` *(optional)*
    - `0` specifies that the CueServer's output should be used to query the group's channels.
    - `1` to `32` specifies that a specific playback should be used to query the group's channels.
    - If this parameter is omitted, the CueServer's output will be used to query the group's channels.

Response:

A 16-bit signed value (in network-byte order) is always returned from this request.

If every channel in the group is set to the same level, that level is returned from zero (`0x0000`) to 100% (`0x00FF`).

If the group's channels have mixed values, then -1 (`0xFFFF`) is returned.

Errors:

If an error occurs during the processing of the request, the meaning of the returned value is explained in the following table:

Error Value	Description
`-2`	An internal shared memory error occurred.
`-3`	An invalid parameter was specified.

Hardwired DMX Input [hdi]

This request returns the DMX Input data that is present on *only* the Hardwired DMX input ports.

URL:

`/get.cgi?req=hdi`

Response:

The following variable-length data structure will be returned by this request.

```
typedef struct DMXInputUniverse {
    uint8_t             universeIndex;          // Index of universe
    uint16_t            channels;               // Number of channels in this universe
    uint8_t             values[];               // Variable-length array of channel values
} DMXInputUniverse;


typedef struct DMXInput {
    uint8_t             universeCount;          // Number of universes received
    DMXInputUniverse    universe[];             // Structure repeated for each universe
} DMXInput;
```

Errors:

If an error occurs during the processing of the request, only a single byte will be returned by this URL. The value of the returned byte is explained in the following table:

Error Value	Description
`0xFF`	An internal memory error occurred.

Network Info [net]

This request returns a *Net Info* data structure for the CueServer. This structure contains detailed information about the current operating parameters of CueServer’s network interfaces.

URL:

```
/get.cgi?req=net
```

Response:

The following data structure will be returned by this request.

```
typedef struct NetInfo {        // (116 bytes)
    uint16_t            signature;              // Signature = 'CS'
    int16_t             version;                // Version = 0x0002 (or negative
error code)

    char                deviceName[32];         // Device Name (hostname)
    uint16_t            switchModel;            // √ Ethernet switch model number
    uint8_t             switchMode;             // √ Ethernet Switch Mode (0 = Sw
itch, 1 = VLAN)
    uint8_t             physicalPorts;          // √ Number of Ethernet ports
    uint8_t             reserved3[12];          // -

    uint32_t            primaryIP;              // √ Primary interface IP Address
    uint32_t            primarySubnet;          // √ Primary interface Subnet Mas
k
    uint32_t            primaryGateway;         // Primary interface Default Gate
way
    uint8_t             primaryMAC[6];          // Primary interface MAC Address
    uint8_t             primaryDHCP;            // Primary interface DHCP Mode
    uint8_t             primaryLinkStatus;      // √ Primary interface Link Statu
s
    uint8_t             reserved1[12];          // -

    uint32_t            secondaryIP;            // √ Secondary interface IP Addre
ss
    uint32_t            secondarySubnet;        // √ Secondary interface Subnet M
ask
    uint32_t            secondaryGateway;       // Secondary interface Default Ga
teway
    uint8_t             secondaryMAC[6];        // Secondary interface MAC Addres
s
    uint8_t                 secondaryDHCP;          // Secondary interface DHCP Mode
    uint8_t                 secondaryLinkStatus;    // √ Secondary interface Link Sta
tus
    uint8_t                 reserved2[12];          // -
} NetInfo;
```

Errors:

If an error occurs during the processing of the request, only the first four bytes of the above structure will be returned by this URL. The first two bytes will have the “CS” signature and the next two bytes will contain an error value as described by the following table:

Error Value	Description
-1	An internal shared memory error occurred.
-2	Could not communicate with Ethernet hardware.
-3	Unknown Ethernet switch model.
-4	TCP/IP stack error.

Ping [ping]

This request returns the device's *Auto-Discovery* data string. Use this request to fetch pertinent device information using a TCP (HTTP) connection.

URL:

```
/get.cgi?req=ping
```

Response:

The returned data is an ASCII string. It begins with `#PING` and contains several fields, each separated by a bar character (`|`).

The following is an example ping string from a CueServer:

```
#PING|600001|10.0.1.5|1.5.5|CueServer 2|255.0.0.0|10.0.1.1|0|239|
000000000000|6/30/2017 12:59:59 PM|N|shows/My Show|1|1|0.0.0.0|0.0.0.0|0
```

The bar-separated fields are explained in the following table:

Field	Example	Description
Fields below are present on both CueServer 1 and CueServer 2 models		
1	`600001`	Serial number (6 characters, may include *only* numbers and letters, no special characters)
2	`10.0.1.5`	Primary interface IP address (standard IPv4 notation)
3	`1.5.5`	Current firmware version (format is `<number>.<number>.<number>[<dev-stage><number>[<letter>]]`, an extreme example would be `2.34.567b99z`)
4	`CueServer 2`	Device name (maximum 15 characters)
5	`255.0.0.0`	Primary interface subset mask (standard IPv4 notation)
6	`10.0.1.1`	Gateway address (standard IPv4 notation)
7	`0`	Primary interface DHCP mode (may be `0` or `1`)
8	`239`	Hardware model (see Hardware Model Identifiers)
9	`000000000000`	Reserved for future use (only used on CueServer 1)
10	`6/30/2017 12:59:59 PM`	The current device time (MM/DD/YY HH:MM:SS AP)

11	`N`	Reserved for future use (only used on CueServer 1)
Fields below are only present on CueServer 2 models		
12	`shows/My Show`	Current show path
13	`1`	Number of physical Ethernet ports (may be `1` or `2`)
14	`1`	Number of logical Ethernet interfaces (may be (`1` or `2`)
15	`0.0.0.0`	Secondary interface IP address (standard IPv4 notation)
16	`0.0.0.0`	Secondary interface subnet mask (standard IPv4 notation)
17	`0`	Secondary interface DHCP mode (may be `0` or `1`)

Errors:

If an error occurs during the processing of the request, only a single byte will be returned by this URL. The value of the returned byte is explained in the following table:

Error Value	Description
`0xFF`	An internal memory error occurred.
`0xFE`	Invalid playback number was specified.

Playback Info [pi]

This request returns a *Playback Info* data structure for the specified playback fader.

This request is available in CueServer 2 only for compatibility with the original CueServer 1 API. Use of this request is **depreciated** and is not encouraged. Use Extended Playback Info [epi] instead.

The PI selector was originally designed for CS1 and therefore is restricted to fixed point fade times, cue IDs with only one digit of precision after the decimal point and it is missing several new playback features available in CS2. This selector is provided for compatibility only and is not recommended for use with CS2.

URL:

`/get.cgi?req=pi&id=<playbackID>`

Parameters:

- `id=<playbackID>`
    - Specifies the a playback fader number from `1` to `32`.

Response:

The following data structure will be returned by this request.

```
typedef struct PlaybackInfo {                   // (96 bytes total)
    uint8_t             playback;               // Playback number (1..32)
    uint8_t             runMode;                // Run Mode (0 = Normal, 1 = Stop
ped)
    uint8_t             outputLevel;            // Output level (0..255)
    uint8_t             combineMode;            // Combine Mode (0 = Merge, 1 = O
verride, 2 = Scale)
    uint16_t            fadeTimer;              // Remaining fade time in progres
s
    uint16_t            followTimer;            // Remaining follow time in progr
ess
    uint32_t            streamTimer;            // Stream playback position

    uint16_t            currentCue;             // Cue currently playing (0 = Non
e, 1 = Cue 0.1, -1 = Active Channels)
    uint16_t            nextCue;                // Next cue (0 = None, 1 = Cue
0.1)
    uint16_t            fadeUpTime;             // Fade Up Time for next cue
    uint16_t            fadeDownTime;           // Fade Down Time for next cue
    uint16_t            followTime;             // Follow Time for next cue
    uint16_t            linkCue;                // Linked cue for next cue

    uint8_t             reserved[8];            // Reserved

    char                currentName[32];        // Name of current cue
    char                nextName[32];           // Name of next cue
} PlaybackInfo;
```

Errors:

If an error occurs during the processing of the request, only a single byte will be returned by this URL. The value of the returned byte is explained in the following table:

Error Value	Description
`0xFF`	An internal shared memory error occurred.
`0xFE`	An invalid playback number was given.

Playback Values [p*]

This request returns the current channel values of a given playback.

URL:

`/get.cgi?req=p1` – Return Playback 1 Values
…
`/get.cgi?req=p32` – Return Playback 32 Values
`/get.cgi?req=po` – Return Playback Output Values

Response:

This request will return twice the number of configured channels in bytes. For example, if 1,024 channels are configured, this request will return 2,048 bytes.

The first half of the returned bytes (equal to the number of configured channels) will be the individual channel values from zero (`0x00`) to 100% (`0xFF`).

The second half of the returned bytes (also equal to the number of configured channels) will be *channel flags* for each channel. Each channel flag byte is interpreted as a set of eight flag bits. The following table defines the meaning of each bit:

Bit	Description
`0x01`	The channel is active (i.e. not released)
`0x02`	–
`0x04`	–
`0x08`	The channel is disabled (its value does not propagate beyond playback)
`0x10`	–
`0x20`	The channel is parked (its value cannot be changed)
`0x40`	–
`0x80`	The channel defaults to a full value when (i.e. in Scale Mode)

Errors:

If an error occurs during the processing of the request, only a single byte will be returned by this URL. The value of the returned byte is explained in the following table:

Error Value	Description
`0xFF`	An internal shared memory error occurred.
`0xFE`	An internal memory error occurred.
`0xFD`	No show is loaded.

Preset Zone Info [pzi]

This request returns a *Cue Stack Info* data structure for the specified preset zone. Use this response to determine which presets are active in a given zone.

URL:

`/get.cgi?req=pzi&name=<zoneName>`

Parameters:

- `name=<zoneName>`
    - The name of the desired zone.

Response:

The following variable-length structure is returned by this request:

```
#define STACK_NAME_BUF_SIZE     16
#define CSI_TYPE_CUES           0
#define CSI_TYPE_PRESETS        1

typedef struct CueStackInfo {
    uint16_t            signature;              // Signature = 'CS'
    int16_t             version;                // Version = 0x0001 (or negative
error code)
    char                stackName[STACK_NAME_BUF_SIZE]; // Name of stack
    uint8_t             type;                   // 0 = Cues, 1 = Presets
    uint8_t             playback;               // The playback number (for prese
ts only)
    uint16_t            count;                  // Number of CueID/Status pairs
    uint32_t            data[64];               // Array of CueID/Status pairs (3
2 pairs max)
} CueStackInfo;
```

Please note that the actual number of bytes returned by this request only includes the actual number of pairs of `uint32_t` elements in the `data[]` array.

The CueID/Status pairs are included for any preset in the zone that is active. The CueID denotes the preset number and the Status value indicates the preset’s active state. The following table shows preset states:

Preset Status	Description
1	This preset is active.
2	This preset is modified.

Errors:

If an error occurs during the processing of the request, only the first four bytes of the above structure will be returned by this URL. The first two bytes will have the “CS” signature and the next two bytes will contain an error value as described by the following table:

Error Value	Description
-1	An internal shared memory error occurred.

Record Stream Info [rs]

This request returns a *Record Stream Info* data structure. Use this request to determine the real-time status of a stream being recorded.

URL:

```
/get.cgi?req=rs
```

Response:

The following data structure will be returned by this request.

```
typedef struct RecordStreamInfo {   // (16 bytes)
    uint16_t            signature;              // Signature = 'CS'
    int16_t             version;                // Version = 0x0001 (or negative error code)

    uint32_t            recordTime;             // Record time (in clicks [1/40th second])
    uint32_t            recordID;               // ID of cue being recorded
    uint8_t             recordState;            // Stream recording state
    uint8_t             reserved[3];            // -
} RecordStreamInfo;
```

Errors:

This request does not return any error codes.

System Log [log]

This request returns the current system log.

URL:

```
/get.cgi?req=log
```

Response:

The *System Log* is returned as plain ASCII text. The length of this text is variable and might be quite large.

Errors:

If an error occurs during the processing of the request, a single byte is returned. The meaning of this byte is explained in the following table:

Error Value	Description
`0xFF`	The system log file could not be opened.

System Status [ss]

This request returns a *System Status* data structure for the CueServer. This structure contains detailed information about the current status of the device.

URL:

```
/get.cgi?req=ss
```

Response:

The following data structure will be returned by this request.

```
typedef struct SystemStatus { // (256 bytes)
    uint16_t            signature;              // Signature = 'CS'
    int16_t             version;                // Version = 0x0002 (or negative
error code)

    uint8_t             pcbIndicators[8];       // PCB Indicator Data
    uint8_t             functionButtons[24];    // Function Button Indicator Dat
a (8xRGB)
    char                lcdData[80];            // LCD Display Buffer
    uint32_t            licenseData;            // License Data

    uint8_t             universeActive[16];     // Universe active bits

    uint16_t            logMessages;            // Number of new CueServer log me
ssages (rolling count)
    uint16_t            importantLogMessages;   // Number of unread "important" C
ueServer log messages (cleared by "log clear")

    uint32_t            processRunning;         // Bitmask of running processes
    uint8_t             debugFlags;             // Bitmask of debug logging flags
    uint8_t             reserved1[11];          // -

    uint8_t             resChangeSeed[32];      // Array of resSeed values (index
ed by RES_SEED_XXX constants)

    uint16_t            boardID;                // The board ID of the device [+v
1.4.1]

    uint8_t             dmxInputDisabled;       // Is the DMX Input disabled? [+v
1.5.0]
```

```
    uint8_t                 lcdBacklight;           // LCD backlight level [+v2.0.0]

    uint8_t                 reserved2[64];          // -
} SystemStatus;
```

Errors:

If an error occurs during the processing of the request, only the first four bytes of the above structure will be returned by this URL. The first two bytes will have the “CS” signature and the next two bytes will contain an error value as described by the following table:

Error Value	Description
-1	An internal shared memory error occurred.

Time Info [ti]

This request returns a *Time Info* string from the CueServer. This string contains information about the NTP servers and time zone currently in use.

URL:

```
/get.cgi?req=ti
```

Response:

The *Time Info* string contains two fields separated by a bar character (`|`).

The first field contains a comma-separated list of NTP Servers (only if Automatic time adjustments are enabled). The second field contains the current POSIX time zone.

The following is an example of a *Time Info* string:

```
0.ntp.pool.org,1.ntp.pool.org,2.ntp.pool.org|America/New_York
```

Errors:

This request does not return any error codes.

Time Status [ts]

This request returns a *Time Status* data structure for the CueServer. This structure contains detailed information about the current time, date, astronomical features, time zone and more.

URL:

`/get.cgi?req=ts`

Response:

The following data structure will be returned by this request.

```
typedef struct TimeStatus {
    uint16_t            signature;              // Signature = 'CS'
    int16_t             version;                // Version = 0x0001 (or negative
error code)

    uint32_t            seconds;                // Current time seconds
    uint8_t             day;                    // Day (0..30)
    uint8_t             month;                  // Month (0..11)
    uint8_t             year;                   // Year (0 = 1900, 255 = 2155)
    uint8_t             weekday;                // Weekday (0 = Sunday, 1 = Monda
y, etc.)
    uint8_t             dst;                    // DST (0 = No, 1 = Yes)
    uint8_t             light;                  // Current light (0 = Dark, 1 = L
ight)
    uint8_t             ntpSync;                // NTP Synchronization Status (0
= None, 1 = Synched)
    uint8_t             reserved;               // -

    float               offset;                 // Offset from GMT/UTC (Hours)
    float               latitude;               // Latitude
    float               longitude;              // Longitude
    uint32_t            sunriseSeconds;         // Sunrise seconds
    uint32_t            sunsetSeconds;          // Sunset seconds

    char                tzAbbreviation[8];      // Time zone abbreviation ("ES
T") (null-terminated)
    char                tzName[32];             // Time zone full name ("America
s/New York") (null-terminated)
} TimeStatus;
```

Errors:

This response does not return any error codes.

Variables [var]

This request returns the value of one or more *variables*.

URL:

`/get.cgi?req=var&id=<variableName>`

Parameters:

- `id=<variableName>`
  - Given a name of a variable (such as `x` or `MyCue`), the value of that single variable will be returned.
  - If `id=*` (an asterisk), then the entire database of user variables will be returned.

Response:

In the case where a single variable value is being requested, the actual value will be returned.

In the case where the variable database is being requested, an array of null-terminated (C-style) strings is returned. The strings are in pairs of then . If the database contains non-volatile variables, they will be at the end of the database, separated by a `*,*` pair.

Errors:

If an error occurs during the processing of the request, only a single byte will be returned by this URL. The value of the returned byte is explained in the following table:

Error Value	Description
`0xFF`	An internal memory error occurred.

Zone Data [zones]

This request returns a *Zones Data* data structure. Use this response to determine what zones are defined and their playbacks, join groups, and active presets.

URL:

`/get.cgi?req=zones`

Response:

The following variable-length structure is returned by this request:

```
typedef struct ZonesData {
    uint16_t            signature;              // Signature = 'CS'
    int16_t             version;                // Version = 0x0001 (or negative
error code)
    uint8_t             reserved[2];            // -
    uint16_t            zoneCount;              // Number of zones
    ZoneRecord          zones[];                // Variable array of zone records
} ZonesData;

#define STACK_NAME_BUF_SIZE     16

typedef struct ZoneRecord {
    char                name[STACK_NAME_BUF_SIZE];  // Name of zone
    uint8_t             playbackIndex;          // Playback index
    uint8_t             joinGroup;              // Join group
    uint16_t            count;                  // Number of PresetID/Status pair
s
    uint32_t            data[];                 // Array of PresetID/Status pair
s (32 pairs max)
} ZoneRecord;
```

Please note that the actual number of bytes returned by this request only includes the actual number of ZoneRecords, each of which only includes the actual number of pairs of `uint32_t` elements in the `data[]` array.

The PresetID/Status pairs are included for any preset in the zone that is active. The PresetID denotes the preset number and the Status value indicates the preset’s active state. The following table shows preset states:

Preset Status	Description
`1`	This preset is active.
`2`	This preset is modified.

Errors:

If an error occurs during the processing of the request, only the first four bytes of the above structure will be returned by this URL. The first two bytes will have the “CS” signature and the next two bytes will contain an error value as described by the following table:

Error Value	Description
`-1`	An internal shared memory error occurred.

pcmd.cgi

The `pcmd.cgi` URL is used to translate (or "parse") a CueScript string into an English language string.

The typical format of this URL is:

```
http://<ip-of-CueServer>/pcmd.cgi?cmd=<command>
```

For example, the following URL will translate the CueScript `Q1G`.:

```
/pcmd.cgi?cmd=Q1G
```

This URL will return an English language string that is the expanded version of the given CueScript.

In the above example, the returned string will be `Cue 1 Go`.

Note that when a command is URL-encoded, spaces must be changed to plus (`+`) characters, and other "special" characters must use standard URL escaping methods, for example a `$` character should be encoded as `%24`. See Percent Encoding on Wikipedia for more details.

Parameters

- `cmd=<string>`
  - A CueScript command.
  - Special characters must be Percent Encoded

Examples

CueScript Command	URL	Returned Value
`M1`	`/pcmd.cgi?cmd=M1`	`Macro 1`
`Q73G`	`/pcmd.cgi?cmd=Q73G`	`Cue 73 Go`
`B1.5OFF`	`/pcmd.cgi?cmd=B1.5OFF`	`Button 1.5 Off`

set.cgi

The `set.cgi` URL is used to store information into the CueServer.

The typical format of this URL is:

```
http://<ip-of-cueserver>/set.cgi?dst=<destination>&<optional-parameters>
```

Depending on the value of the `<destination>` parameter, this URL can store many different pieces of information to the CueServer.

The following variations of the `set.cgi` URL are available:

- Audio Properties [audio]
- LCD Properties [lcd]
- Network Properties [net]
- Time Properties [time]
- Station Color Properties [stcol]

Audio Properties [audio]

This request sets various audio system properties.

URL:

`/set.cgi?dst=audio&<optional-parameters>`

Optional Parameters:

- `volume=<0..100>` *(optional)*
  - This parameter (if present) sets the master audio output volume.
  - Valid range is from `0` to `100`.

Response:

A single byte is returned. The following table explains the possible return values.

Result	Description
`0x00`	The operation was successful.
`0xFF`	The volume failed to be set properly.

Examples:

Function	URL
Set volume to 75%	`/set.cgi?dst=audio&volume=75`

LCD Properties [lcd]

This request sets various LCD Display properties.

URL:

`/set.cgi?dst=lcd&<optional-parameters>`

Optional Parameters:

- `backlight=<0..255>` *(optional)*
  - This parameter (if present) sets the LCD Display’s backlight brightness.
  - Valid range is from `0` to `255`.

- `field=<1..4>&id=<0..16>` *(optional)*
  - This parameter group (if present) sets one of the four quadrants of the LCD Display to one of the built-in display functions.
  - Valid fields are from `1` to `4` (see below).
  - Valid display function ID is from `0` to `16` (see below).

LCD Field	Description
`1`	Top-Left
`2`	Top-Right
`3`	Bottom-Left
`4`	Bottom-Right

LCD Function	Description
`0`	Blank
`1`	Device Name
`2`	Long Date + 12-Hour Time
`3`	Short Date + 12-Hour Time
`4`	Long Date + 24-Hour Time
`5`	Short Date + 24-Hour Time
`6`	Long Date
`7`	Short Date

`8`	12-Hour Time
`9`	24-Hour Time
`10`	User String
`11`	IP Address
`12`	Timecode
`13`	I/O Status
`14`	CPU Load
`15`	Show Path
`16`	Show Name

Response:

A single byte is returned. The following table explains the possible return values.

Result	Description
`0x00`	The operation was successful.
`0xFE`	An internal shared memory error occurred.

Examples:

Function	URL
Set the backlight to 75%	`/set.cgi?dst=lcd&backlight=191`
Set the Top-Left quadrant of the LCD to I/O Status	`/set.cgi?dst=lcd&field=1&id=13`

Network Properties [net]

This request sets various Network properties.

URL:

`/set.cgi?dst=net&<optional-parameters>`

Optional Parameters:

- `name=<string>` *(optional)*
  - This parameter (if present) sets the device’s name.
  - Maximum length of this string is 15 characters.

- `ipA=<ipAddress>` *(optional)*
  - This parameter (if present) sets the device’s primary IP Address.

- `subA=<ipAddress>` *(optional)*
  - This parameter (if present) sets the device’s primary subnet mask.

- `dhcpA=<0,1>` *(optional)*
  - This parameter (if present) sets the device’s primary DHCP Mode.
  - This parameter may be set to `0` or `1`.

- `ipB=<ipAddress>` *(optional)*
  - This parameter (if present) sets the device’s secondary IP Address.
  - This parameter is of no use on a device with a single Ethernet port.

- `subB=<ipAddress>` *(optional)*
  - This parameter (if present) sets the device’s secondary subnet mask.
  - This parameter is of no use on a device with a single Ethernet port.

- `dhcpB=<0,1>` *(optional)*
  - This parameter (if present) sets the device’s secondary DHCP Mode.
  - This parameter may be set to `0` or `1`.
  - This parameter is of no use on a device with a single Ethernet port.

- `gateway=<ipAddress>` *(optional)*
  - This parameter (if present) sets the device’s gateway address.

- `interfaces=<1,2>` *(optional)*
  - This parameter (if present) sets the device’s number of interfaces.
  - A setting of `1` puts the device into a mode where each physical port is connected to a built-in

switch connected to a single interface.
    - A setting of `2` puts the device into a mode where each of the two physical ports become their own separate interfaces, each with their own IP addresses.
    - This parameter has no effect on a device with a single Ethernet port.

Response:

A single byte is returned. The following table explains the possible return values.

Result	Description
`0x00`	The operation was successful.
`0xFF`	The network interfaces file could not be read.

Examples:

Function	URL
Set the device name to “My CueServer”	`/set.cgi?dst=net&name=My+CueServer`
Set the primary IP parameters	`/set.cgi?dst=net&ipA=10.0.1.5&subA=255.0.0.0&dhcpA=0`

Time Properties [time]

This request sets various Network properties.

URL:

`/set.cgi?dst=time&<optional-parameters>`

Optional Parameters:

- `ntpList=` *(optional)*
    - This parameter (if present) sets the device's list of NTP servers.
    - This parameter should not be used if the time is being set manually.
- `year=<1900..2199>` *(optional)*
    - This parameter (if present) sets the device's clock's year.
- `month=<1..12>` *(optional)*
    - This parameter (if present) sets the device's clock's month.
- `day=<1..31>` *(optional)*
    - This parameter (if present) sets the device's clock's day.
- `hour=<0..23>` *(optional)*
    - This parameter (if present) sets the device's clock's hour.
- `minute=<0..59>` *(optional)*
    - This parameter (if present) sets the device's clock's minute.
- `second=<0..59>` *(optional)*
    - This parameter (if present) sets the device's clock's second.
- `timezone=` *(optional)*
    - This parameter (if present) sets the device's time zone.
    - The time zone string must be from the tzdatabase (i.e.: "America/New_York").

Response:

A single byte is returned. The following table explains the possible return values.

Result	Description

`0x00`	The operation was successful.
`0xFF`	A required parameter was missing (i.e.: month was given but day was not)

Examples:

Function	URL
Set the time manually to 6/30/2017 1:00:42 PM	`/set.cgi?dst=time&year=2017&month=6&day=30&hour=13&minute=0&second=42`
Set the time via list of NTP servers	`/set.cgi?dst=time&ntpList=pool0.ntp.org%0Dpool1.ntp.org%0Dpool2.ntp.org`
Set the time zone	`/set.cgi?dst=time&timezone=America%2FNew_York`

Station Color Properties [stcol]

This request sets various Station Indicator Color properties.

URL:

`/set.cgi?dst=stcol&<optional-parameters>`

Optional Parameters:

- `station=<-1,0..1000>` *(optional)*
  - This parameter (if present) chooses which station to operate on.
  - If this parameter is not preset or `-1` is specified, then this function will operate on the “global” station settings.

- `button=<-1,0..1000>` *(optional)*
  - This parameter (if present) chooses which button to operate on.
  - If this parameter is not preset or `-1` is specified, then this function will operate on all buttons of the specified station.

- `=` *(optional)*
  - The colorName parameter may be any of `user1`, `user2`, `user3`, `user4`, `on`, `off`, `mixed`, `locked`.
  - The rgbColor parameter may be a 6-digit or 8-digit hexadecimal color. For example, Red would be expressed as `FF0000`, and a Dark Blue would be `000033`.
  - One or more color parameters may be specified in the same URL.

Response:

A single byte is returned. The following table explains the possible return values.

Result	Description
`0x00`	The operation was successful.
`0xFF`	An internal shared memory error occurred.

Examples:

Function	URL
Set the “On” color of Button 2 of Station 3 to Green	`/set.cgi?dst=stcol&station=3&button=2&on=00FF00`

Set the “Off” color of all buttons on Station 4 to Dark Yellow	`/set.cgi?dst=stcol&station=4&on=222200`
Set the “User 1” and “User 2” colors of all buttons	`/set.cgi?dst=stcol&user1=FF8800&user2=00FF44`

Show File Format

Directory Structure

The show file for CueServer is arranged in a directory structure.

The following table illustrates the directory structure of a typical show file:

File	Description
audio chime.wav MySound.ogg My Music.mp3	The **audio** directory contains audio files. Audio files can be any type supported by the system.
cues 1.00.cue 101.50.cue 123456.78.cue **MyStack** 1.00.cue 2.10.cue	The **cues** directory contains both cue resources and/or cue stack directories. Cue file names use the same number as the cue with two decimal places of precision. Cue files end with the `.cue` extension. Valid cue numbers range from `0.00` to `999999.99`. Each cue stack is its own directory in the **cues** directory. Cue stack names are limited to 15 characters any may not contain spaces. Cues in cue stack directories follow the same rules as cues in the root **cues** directory.
dmxtriggers 1.dmxtrigger 2.dmxtrigger 3.dmxtrigger	The **dmxtriggers** directory contains DMX Input Trigger resources. DMX Trigger file names are based on the resource ID followed by the `.dmxtrigger` extension. See DMXTrigger Resource for file format details.
groups 1.group 2.group 3.group	The **groups** directory contains Group resources. Group file names are based on the resource ID followed by the `.group` extension. See Group Resource for file format details.
macros 1.macro 2.macro 3.macro	The **macros** directory contains Macro resources. Macro file names are based on the resource ID followed by the `.macro` extension. See Macro Resource for file format details.

presets **Conference** 0.zone 1.preset 2.preset 3.preset **Lobby** 0.zone 1.preset 2.preset	The **presets** directory contains zone directories. Each zone includes a **0.zone** zone configuration file. Within each zone, Preset file names use the same number as the preset. Preset files end with the `.preset` extension. Valid preset numbers range from `1` to `999`. Each additional zone is its own directory in the root directory. Zone names are limited to 15 characters any may not contain spaces. Each zone directory also contains a **0.zone** configuration file and its own set of Preset resources.
rules 1.rule 2.rule 3.rule	The **rules** directory contains Rule resources. Rule file names are based on the resource ID followed by the `.rule` extension.
show.cfg	This is the main show configuration file. It is always included in the root level off the show directory. The **show.cfg** file contains settings for DMX patching, physical location, LCD display, playbacks, global show preferences and more. See show.cfg for details on this file format.
stations 0 1.button 2.button 3.button 1.contact 2.contact 3.contact 1.output 2.output 3.output 1.port	The **stations** directory contains sub-directories for each station. Station directories are named with the ID number of the station. Station IDs always start at `0`. Individual resources for Buttons, Contacts, Outputs, and Serial Ports are included in each station directory. The ID numbers for each resource and number of resources for each type is dictated by the type and configuration of station. These files end with the `.button`, `.contact`, `.output`, and `.port` extensions. Each station directory includes a **station.cfg** file that defines the station properties. A station with ID `0` is always present and represents the “built-in” station that corresponds to the front-panel buttons and included contact closures, outputs and serial ports.

📄 2.port 📄 station.cfg 📁 1 📄 1.button 📄 2.button 📄 3.button 📄 station.cfg 📁 2 📄 1.button 📄 1.contact 📄 2.contact 📄 station.cfg	
📁 **timers** 📄 1.timer 📄 2.timer 📄 3.timer	The **timers** directory contains Timer resources. Timer file names are based on the resource ID followed by the `.timer` extension.
📁 **web** 📄 index.html 📄 picture.png 📄 My Script.js 📁 **MorePages** 📄 Document.pdf 📄 index.shtml	The **web** directory is served by the embedded Apache web server. This directory may contain any combination of files and folders as needed.

Configuration Files

- show.cfg

show.cfg

The `show.cfg` file is a simple text file in LibConfig format. This file is located in the root of the show's file system directory structure.

The following elements appear in the `show.cfg` file:

`astro`	Dictionary (astro)	Contains a dictionary of Astronomical Time configuration elements.
`audio`	Dictionary (audio)	Contains a dictionary of Audio configuration elements.
`description`	String	A textual description for the show file that appears in the Settings/Description panel of CueServer Studio.
`dmx`	Dictionary (dmx)	Contains a dictionary of DMX configuration elements.
`lcd`	Dictionary (lcd)	Contains a dictionary of LCD Display configuration elements.
`name`	String	*DEPRECIATED: This element used to contain the name of the show. Now, the actual name of the show file directory is used as the show's name.*
`preferredModel`	Integer	Contains the Model ID number of the "preferred" CueServer model for this show file. CueServer Studio uses this ID to display the correct UI for specific configuration elements that are different for each model. • 0 = Any • 160 = CS-900 • 176 = CS-920 • 224 = CS-940
`station`	Dictionary (station)	Contains a dictionary of Station related configuration elements.

'astro' Dictionary

The following elements appear in the `astro` dictionary:

`latitude`	Float	The device's latitude coordinate (in degrees, for example 34.24963).
`longitude`	Float	The device's longitude coordinate (in degrees, for example -84.05723).
`offset`	Float	*DEPRECIATED: This element had previously been used to specify the device's offset from*

		UTC/GMT. Now the time zone offset is retrieved from the system's global clock settings.

'audio' Dictionary

The following elements appear in the `audio` dictionary:

`volume`	Integer	The audio output volume (from 0 to 100).

'dmx' Dictionary

The following elements appear in the `dmx` dictionary:

`channelCount`	Integer	The number of channels being output (must be equal to `universeCount` * 512).
`playbackCount`	Integer	The number of playbacks configured.
`playbacks`	Dictionary Array (playbacks)	An array containing the configuration dictionaries for each playback.
`ports`	Dictionary Array (ports)	An array containing the configuration dictionaries for each port.
`universeCount`	Integer	The number of CueServer universes configured (must be equal to `channelCount` / 512).
`universes`	Dictionary Array (universes)	An array containing the configuration dictionaries for each universe.

'dmx/playbacks' Dictionary

The following elements appear in the `dmx/playbacks` dictionary:

`name`	String	The descriptive name for the playback (may be blank).
`mode`	Integer	An ID for the default mode of the playback. • 0 = Merge • 1 = Override • 2 = Scale • 3 = Pin

‘dmx/ports’ Dictionary

The following elements appear in the `dmx/ports` dictionary:

`direction`	Integer	The input/output “direction” for the port. [OPTIONAL] • 0 = Off • 1 = Input • 2 = Output
`universe`	Integer	The CueServer universe number (1..32) corresponding to this port. *Previous versions of CueServer used universe 0 to mean that the port was disabled, which is depreciated. Instead use the direction element to indicate that a port is disabled.*

‘dmx/universes’ Dictionary

The following elements appear in the `dmx/universes` dictionary:

`channels`	Integer	The number of channels in this universe (1-512). If this element is missing, the default is 512.
`name`	String	Descriptive name for the universe (may be blank).
`rx_port`	Integer	The port number for KiNET v2 protocol. *Only used for the KiNET v2 protocol.*
`rx_priority_high`	Integer	High limit of the priority range of received packets (0-200). *Only used for the sACN protocol.*
`rx_priority_low`	Integer	Low limit of the priority range of received packets (0-200). *Only used for the sACN protocol.*
`rx_protocol`	Integer	Number of protocol to use for receiving DMX over Ethernet for this universe. • 0 = None • 1 = sACN • 2 = KiNet v1 • 3 = KiNet v2 • 4 = Art-Net
`rx_universe`	Integer	External universe number to receive channels from. • sACN = Universe 1..63999 • KiNET = Universe 0..2147483647, -1 = All • Art-Net = Port-Address 0..32767
`tx_flags`	Integer	The transmit flags for KiNET v2. • 0×01 = Chromatic

		• 0×02 = Sync Packets
`tx_ip`	String	IP Address to transmit packets to. *Only used for KiNet and Art-Net protocols.*
`tx_mode`	Integer	*DEPRECIATED: This element had previously been used to specify the Art-Net broadcast mode. Now the broadcast mode is implied by special values in the tp_ip element.*
`tx_port`	Integer	The port number for KiNET v2 protocol. *Only used for the KiNET v2 protocol.*
`tx_portout`	String	*DEPRECIATED: This element had previously been used to list KiNET v2 "portout" parameters.*
`tx_priority`	Integer	Priority level for transmitted packets (0-200). *Only used for the sACN protocol.*
`tx_protocol`	Integer	Number of protocol to use for transmitting DMX over Ethernet for this universe. • 0 = None • 1 = sACN • 2 = KiNet v1 • 3 = KiNet v2 • 4 = Art-Net
`tx_universe`	Integer	External universe number to transmit channels to. • sACN = Universe 1..63999 • KiNET = Universe 0..2147483647, -1 = All • Art-Net = Port-Address 0..32767

'lcd' Dictionary

The following elements appear in the `lcd` dictionary:

`top-left`	Integer	The ID of the display element that will appear in the top-left of the LCD Display. Possible values include: • 0 = None • 1 = Device Name • 2 = Long Time/Date 12 Hour • 3 = Short Time/Date 12 Hour • 4 = Long Time/Date 24 Hour • 5 = Short Time/Date 24 Hour • 6 = Long Date • 7 = Short Date • 8 = 12 Hour Time • 9 = 24 Hour Time • 10 = User String • 11 = IP Address

		• 12 = Timecode • 13 = I/O Status • 14 = CPU Load • 15 = Show Path • 16 = Show Name
`top-right`	Integer	The ID of the display element that will appear in the top-right of the LCD Display (uses same constants as above).
`bottom-left`	Integer	The ID of the display element that will appear in the bottom-left of the LCD Display (uses same constants as above).
`bottom-right`	Integer	The ID of the display element that will appear in the bottom-right of the LCD Display (uses same constants as above).
`backlight`	Integer	The brightness level of the LCD backlight (from 0 to 255).

‘station’ Dictionary

The following elements appear in the `station` dictionary:

`onColor`	String	The default onColor for stations, represented as a hexadecimal string. For example, an Orange color would be represented as `FF8800`. Colors may optionally have an Alpha component, which is used to denote the flash pattern. For example, Green with a flash pattern of 3 would be represented as `00FF0003`.
`offColor`	String	The default offColor for stations, represented as a hexadecimal string.
`mixedColor`	String	The default mixedColor for stations, represented as a hexadecimal string.
`lockedColor`	String	The default lockedColor for stations, represented as a hexadecimal string.
`user1Color`	String	The default user1Color for stations, represented as a hexadecimal string.
`user2Color`	String	The default user2Color for stations, represented as a hexadecimal string.
`user3Color`	String	The default user3Color for stations, represented as a hexadecimal string.
`user4Color`	String	The default user4Color for stations, represented as a hexadecimal string.

Resource Structures

Cue Resource

A Cue (or Preset) Resource is a binary file with a format described by the following C structures and constants:

```
//
---------------------------------------------------------------------------------
------------------------------------
//      Cue Resource (Public)
//
---------------------------------------------------------------------------------
------------------------------------

// Constants
#define CUEID_MIN                   0           // Minimum cueID corresponds to C
ue 0.00
#define CUEID_MAX                   99999999    // Maximum cueID corresponds to C
ue 999,999.99
#define CUEID_MULTIPLIER            100         // CueID is 100x the natural cue
number
#define PRESETID_MAX                999         // presetIDs do not use decimal n
umbers
#define CUE_OFFSET_STREAM_DATA      0x1000      // This is a fixed file position
for the start of streaming data blocks

// Resource Identifiers
#define CUE_RESTYPE                 'C'         // Resource type identifier
#define CUE_RESVERS                 '1'         // Version 1 identifier

typedef struct Cue {
    //  ----------------------------------------------------------
    uint8_t         resType;            // (0x00) Resource type (Cue = 'C')
    uint8_t         resVers;            // (0x01) Resource version (Cue = '1')
    uint8_t         cueType;            // (0x02) Cue Type
    uint8_t         cueFlags;           // (0x03) Cue flags
    FadeTimes       fadeTimes;          // (0x04) Fade times (up/down/delay)
    float           followTime;         // (0x14) Follow time (0 = none)
    int32_t         linkCueID;          // (0x18) Link Cue ID (-1 = none)
    uint32_t        reserved1;          // (0x1C)
    //  ----------------------------------------------------------
    uint32_t        streamDuration;     // (0x20) Total time of stream (clicks [4
0Hz])
    uint32_t        streamTrimStart;    // (0x24) Number of clicks into stream t
o start playback
    uint32_t        streamTrimEnd;      // (0x28) Number of clicks from end of st
ream to end playback
    uint8_t         streamMode;         // (0x2C) Auto-Follow/Loop/Hold/Release
    uint8_t         reserved2[3];       // (0x2D) -
    //  ----------------------------------------------------------
    uint8_t         reserved3[13];      // (0x30) -
    uint8_t         ruleCount;          // (0x3D) Number of rules in rules[]
    uint16_t        channels;           // (0x3E) Channel count (must be multipl
e of 8)
    uint8_t         mask[];             // (0x40) Bitmask (size is channels/8)
//  uint8_t         levels[*];          // (????) Channel values (size is channe
l count) [This field has a size of zero for streaming cues!]
//  char            name[*];            // (????) Cue Name (c-string)
//  char            action[*];          // (????) CueScript action (c-string) [de
preciated; must include termination byte]
//  char            rules[*];           // (????) Rules (c-string)
    //  -------- NULL PADDING FOR STREAMING CUES ONLY --------
//  char            streamData[*];      // (0x1000) Streaming Data Starts at 0x10
00 (CUE_OFFSET_STREAM_DATA)
    //  ----------------------------------------------------------
} Cue;

// cueType
#define CUE_TYPE_NORMAL             0x00        // This cue is a normal cue
#define CUE_TYPE_STREAMING          0x01        // This cue is a streaming cue
#define CUE_TYPE_PRESET             0x02        // This cue is a preset

// streamMode
#define STREAM_MODE_FOLLOW          0x00        // At the end of this stream, fol
low to the next cue
#define STREAM_MODE_LOOP            0x01        // At the end of this stream, loo
p back to the beginning of the stream
#define STREAM_MODE_HOLD            0x02        // At the end of this stream, hol
d the final channel values
#define STREAM_MODE_RELEASE         0x03        // At the end of this stream, rel
ease all channel values


//
---------------------------------------------------------------------------------
------------------------------------
//      Fade Times (Public)
//
```

```
---------------------------------------------------------------------------------
------------------------------------

typedef struct FadeTimes {      // 16 bytes
    float               upTime;                 // (0x00) Fade Up Time (in second
s)
    float               upDelay;                // (0x04) Fade Up Delay (in secon
ds)
    float               downTime;               // (0x08) Fade Down Time (in seco
nds)
    float               downDelay;              // (0x0C) Fade Down Delay (in sec
onds)
} FadeTimes;
```

Additionally, if the Cue is a *streaming cue*, then a series of “stream blocks” will be written to the file starting at file offset `0x1000`. Each stream block has the format as described by the following C structures and constants:

```
//
---------------------------------------------------------------------------------
------------------------------------
//      Streaming Cues (Public)
//
---------------------------------------------------------------------------------
------------------------------------

typedef struct StreamBlockHeader0 {     // (4 bytes)
    uint16_t                universeIndex;          // Index of universe (0..127)
    uint16_t                reserved;               // -
} StreamBlockHeader0;

typedef struct StreamBlockHeader1 {     // (4 bytes)
    uint32_t                endToken;               // 'END!'
} StreamBlockHeader1;

typedef struct StreamBlockHeader2 {     // (4 bytes)
    uint16_t                channelIndex;           // Index of first channel of
changes (0..511)
    uint16_t                channelCount;           // Channels in update (1..51
2)
} StreamBlockHeader2;
```

```
typedef struct StreamBlockHeader {      // (16 bytes)
    uint16_t                identifier;             // Constant = "SB"
    uint8_t                 blockType;              // 0 = One universe of data
    uint8_t                 reserved1;              // -
    uint16_t                reserved2;              // -
    uint16_t                blockSize;              // Size of data after header
    uint32_t                time;                   // Timestamp for block (expressed in 1/100 second units)

    union {
        StreamBlockHeader0  type0;                  // StreamBlockHeader0
        StreamBlockHeader1  type1;                  // StreamBlockHeader1
        StreamBlockHeader2  type2;                  // StreamBlockHeader2
    } info;

} StreamBlockHeader;

//  Constants
#define STREAM_BLOCK_ID             0x4253          // 'SB'
#define STREAM_END_TOKEN            0x21444E45      // 'END!'

//  blockType
#define STREAM_BLOCK_UNIVERSE       0               // Single universe
#define STREAM_BLOCK_END            1               // End Block
#define STREAM_BLOCK_RANGE          2               // Range of channels
```

DMXTrigger Resource

A DMXTrigger Resource is a binary file with a format described by the following C structure and constants:

```
#define DMXTRIG_RESTYPE     'D'
#define DMXTRIG_RESVERS     '1'

#define MAX_DMXTRIG_COUNT   100     // Maximum number of DMX Triggers loaded at o
nce

typedef struct DMXTrigRange {       // 6 bytes
    uint16_t        rangeLow;               // Low end of range
    uint16_t        rangeHigh;              // High end of range
    uint8_t         reserved[2];            // -
} DMXTrigRange;

typedef struct DMXTrigSubmaster {   // 6 bytes
    uint16_t        playback;               // Playback index
    uint16_t        reserved1;              // -
    bool            invert;                 // Invert input
    uint8_t         reserved2;              // -
} DMXTrigSubmaster;

typedef struct DMXTriggerResource {
    //  ----------------------------------------------------------
    uint8_t         resType;                // (0x00) Resource type (DMXTrig =
'T')
    uint8_t         resVers;                // (0x01) Resource version (DMXTrig
= '1')
    uint8_t         mode;                   // (0x02) Mode (0=Range, 1=Submaste
r, 2=Continuous, etc.)
    uint8_t         reserved1;              // (0x03) -
    uint16_t        channel;                // (0x04) DMX Channel (0..16383)

    union {
        struct DMXTrigRange     range;          // (0x06) Data for Range
        struct DMXTrigSubmaster submaster;      // (0x06) Data for Submaster
    } info;

    uint8_t         reserved2[3];           // (0x0C) -
    uint8_t         ruleCount;              // (0x0F) Number of rules in variable
Params
    //  ----------------------------------------------------------
    char            variableParams[];       // (0x10) Beginning of variable "C-String" parameters
    //  char            name[];             // (0) Name (c-string)
    //  char            rules[][];          // (1+) Rules list (c-string list)
    //  ----------------------------------------------------------
} DMXTriggerResource;

//  Modes
#define DMXTRIG_MODE_RANGE              0   // Trigger occurs within a range of channel values
#define DMXTRIG_MODE_SUBMASTER          1   // Trigger directly controls a submaster value

//  Variable Strings
#define DMXTRIG_STR_NAME                0
#define DMXTRIG_STR_RULES               1
```

Group Resource

A Group Resource is a binary file with a format described by the following C structure and constants:

```
#define GROUP_RESTYPE       'G'
#define GROUP_RESVERS       '1'

typedef struct GroupResource {
    //  ----------------------------------------------------------
    uint8_t         resType;            // (0x00) Resource type (Group = 'G')
    uint8_t         resVers;            // (0x01) Resource version (Group = '1')
    uint16_t        maskBytes;          // (0x02) Number of bytes in mask
    uint8_t         reserved1[12];      // (0x04) -
    //  ----------------------------------------------------------
    uint8_t         variableParams[];   // (0x10) Beginning of variable parameter
s
//  uint8_t         mask[*];            // (????) Bitmask (size is maskBytes)
//  char            name[*];            // (????) Group Name (c-string)
    //  ----------------------------------------------------------
} GroupResource;
```

Marco Resource

A Macro Resource is a binary file with a format described by the following C structure and constants:

```
typedef struct MacroResource {
    uint8_t         resType;
    uint8_t         resVers;
    uint8_t         showInMenu;
    uint8_t         reserved[13];
    //  ----------------------------------------------------------
    char            variableParams[];       // (0x10) Beginning of variable "C-String" parameters
    //  char            name[];             // (0) Name (c-string)
    //  char            script[];           // (1) Macro script (c-string)
    //  ----------------------------------------------------------
} MacroResource;

// Variable Strings
#define MACRO_STR_NAME              0
#define MACRO_STR_SCRIPT            1
```

Hardware Model Identifiers

One of the fields in the Ping response string from a CueServer is the *Hardware Model Identifier*.

This number is a 16-bit value divided into several fields. When looking at this value in hexadecimal, its digits are broken into the following meanings:

Value	Description
`0xWXYZ`	`W` = Reserved for future use, should be `0` `X` = Hardware Revision, see below (`0` to `F`) `Y` = Hardware Platform, see below (`0` to `F`) `Z` = Hardware Variant, see below (`0` to `F`)

Hardware Revision

A single hexadecimal digit from `0` to `F` corresponds to Hardware Revision "A" through "P".

Note that CueServer 1 products do not report their hardware revision, and therefore they always return `0` in this field.

Hardware Platform

Hex Digit	Description
`0`	CS-8xx Series (model indicated by Hardware Variant field)
`A`	CS-900 CueServer 2 Pro
`B`	CS-920 CueServer 2 Mini
`E`	CS-940 CueServer 2 DIN
All others	Reserved for future use

Hardware Variant

For the CueServer 1 series, this field indicates the specific model of CueServer:

Hex Digit	Description
`0`	Unknown Model

`1`	CS-800 CueServer Pro
`2`	CS-810 CueServer Mini
`3`	CS-820 CueServer Mini DIN
`4`	CS-830 CueServer Mini DIN with Buttons
`5`	CS-PCB CueServer PCB
`6`	CS-815 CueServer BTO
`7`	CS-816 CueServer Express
`8`	CS-840 CueServer DIN
`9`	CS-811 CueServer Mini II
All others	Reserved for future use

For the CueServer 2 series, this field is reserved for future use and typically returns `F`.

Examples

Identifier (Hex)	Value (Decimal)	Decoded Meaning
`0x0001`	`1`	CS-800 CueServer Pro
`0x0007`	`7`	CS-816 CueServer Express
`0x00EF`	`239`	CS-940 CueServer 2 DIN, Rev. A
`0x01AF`	`431`	CS-900 CueServer 2 Pro, Rev. B
`0x03BF`	`959`	CS-920 CueServer 2 Mini, Rev. D
`0x07EB`	`2027`	CS-940 CueServer 2 DIN, Rev. H, Special Variant

> Please note that since all CueServer 1 series report `0` as the hardware revision and `0` as the hardware platform, it is safe to assume that if the Hardware Identifier is 15 (`0x000F`) or lower, then the device is a CueServer 1. If the Hardware Identifier is 16 (`0x0010`) or higher, then the device is a CueServer 2.

Autodiscovery

CueServers on the network can be discovered by using an *auto-discovery* technique.

All CueServers (both the original CueServer and the CueServer 2 series) are listening on the CueServer Multicast Group Address (239.255.204.2) on port 52737. This socket is typically used to send CueScript commands to the CueServer, but it also used for auto-discovery.

To ask CueServers to report themselves, simply send the 6 character string `#PING#` via UDP to this group address and port number.

Every CueServer that receives this message will reply to the sender with a UDP packet. The returned data is an ASCII string. It begins with `#PING` and contains several fields, each separated by a bar character (`|`).

The following is an example ping reply packet from a CueServer:

```
#PING|600001|10.0.1.5|1.5.5|CueServer 2|255.0.0.0|10.0.1.1|0|239|
000000000000|6/30/2017 12:59:59 PM|N|shows/My Show|1|1|0.0.0.0|0.0.0.0|0
```

The bar-separated fields are explained in the following table:

Field	Example	Description
Fields below are present on both CueServer 1 and CueServer 2 models		
1	`600001`	Serial number (6 characters, may include *only* numbers and letters, no special characters)
2	`10.0.1.5`	Primary interface IP address (standard IPv4 notation)
3	`1.5.5`	Current firmware version (format is `<number>.<number>.<number>[<dev-stage><number>[<letter>]]`, an extreme example would be `2.34.567b99z`)
4	`CueServer 2`	Device name (maximum 15 characters)
5	`255.0.0.0`	Primary interface subset mask (standard IPv4 notation)
6	`10.0.1.1`	Gateway address (standard IPv4 notation)
7	`0`	Primary interface DHCP mode (may be `0` or `1`)
8	`239`	Hardware model (see Hardware Model Identifiers)
9	`000000000000`	Reserved for future use (only used on CueServer 1)
10	`6/30/2017 12:59:59 PM`	The current device time (MM/DD/YY HH:MM:SS AP)

11	`N`	Reserved for future use (only used on CueServer 1)
Fields below are only present on CueServer 2 models		
12	`shows/My Show`	Current show path
13	`1`	Number of physical Ethernet ports (may be `1` or `2`)
14	`1`	Number of logical Ethernet interfaces (may be (`1` or `2`)
15	`0.0.0.0`	Secondary interface IP address (standard IPv4 notation)
16	`0.0.0.0`	Secondary interface subnet mask (standard IPv4 notation)
17	`0`	Secondary interface DHCP mode (may be `0` or `1`)

Appendix A: CURL Documentation

CueServer uses the standard Linux CURL tool as part if its implementation of the Write command.

Use **Write** with the `URL` option to cause CueServer to use CURL to send an HTTP request to another device on the network. For example:

```
Write URL "http://10.0.1.5/cgi-bin/request"
```

The above example sends an HTTP request to 10.0.1.5 to GET the /cgi-bin/request URL.

The CURL tool contains a large amount of options that can be used to customize the request in many ways. It is beyond the scope of this manual to teach all of the various options available through CURL, but the CURL documentation is included below for reference. You can use these options to modify the HTTP request or to switch the request to various other protocols.

```
curl(1)                                    Curl Manua
l                                     curl(1)

NAME
       curl - transfer a URL

SYNOPSIS
       curl [options] [URL...]

DESCRIPTION
       curl  is a tool to transfer data from or to a server, using one of the sup
ported protocols
       (DICT, FILE, FTP, FTPS, GOPHER, HTTP, HTTPS, IMAP, IMAPS, LDAP, LDAPS, POP
3, POP3S,  RTMP,
       RTSP,  SCP,  SFTP, SMTP, SMTPS, TELNET and TFTP).  The command is designe
d to work without
       user interaction.

       curl offers a busload of useful  tricks  like  proxy  support,  user  auth
entication,  FTP
       upload,  HTTP  post,  SSL connections, cookies, file transfer resume and m
ore. As you will
       see below, the number of features will make your head spin!

       curl is powered by libcurl for all transfer-related features. See libcur
l(3) for details.
```

```
URL
       The URL syntax is protocol-dependent. You'll find a detailed description i
n RFC 3986.

       You can specify multiple URLs or parts of URLs by writing part sets withi
n braces as in:

        http://site.{one,two,three}.com

       or you can get sequences of alphanumeric series by using [] as in:

        ftp://ftp.numericals.com/file[1-100].txt
        ftp://ftp.numericals.com/file[001-100].txt    (with leading zeros)
        ftp://ftp.letters.com/file[a-z].txt

       Nested sequences are not supported, but you can use several ones next to e
ach other:

        http://any.org/archive[1996-1999]/vol[1-4]/part{a,b,c}.html

       You can specify any amount of URLs on the command line. They will be fetch
ed in a  sequen-
       tial manner in the specified order.

       You can specify a step counter for the ranges to get every Nth number or l
etter:

        http://www.numericals.com/file[1-100:10].txt
        http://www.letters.com/file[a-z:2].txt

       If  you  specify  URL without protocol:// prefix, curl will attempt to gue
ss what protocol
       you might want. It will then default to HTTP but try other protocols base
d  on  often-used
       host  name prefixes. For example, for host names starting with "ftp." cur
l will assume you
       want to speak FTP.

       curl will do its best to use what you pass to it as a URL. It is not tryin
g to validate it
       as  a  syntactically  correct  URL  by  any means but is instead very libe
ral with what it
       accepts.
       Curl will attempt to re-use connections for multiple file transfers, so th
at getting  many
       files  from  the  same  server  will  not do multiple connects / handshake
s. This improves
       speed. Of course this is only done on files specified on a single command
line and  cannot
       be used between separate curl invokes.

PROGRESS METER
       curl normally displays a progress meter during operations, indicating the
amount of trans-
       ferred data, transfer speeds and estimated time left, etc.

       curl displays this data to the terminal by default, so if you invoke curl
to do an  opera-
       tion and it is about to write data to the terminal, it disables the progre
ss meter as oth-
       erwise it would mess up the output mixing progress meter and response dat
a.

       If you want a progress meter for HTTP POST or PUT  requests,  you  need  t
o  redirect  the
       response output to a file, using shell redirect (>), -o [file] or similar.

       It  is  not  the same case for FTP upload as that operation does not spit
out any response
       data to the terminal.

       If you prefer a progress "bar" instead of the regular meter, -# is your fr
iend.

OPTIONS
       In general, all boolean options are enabled with --option  and  yet  agai
n  disabled  with
       --no-option.  That  is,  you use the exact same option name but prefix it
with "no-". How-
       ever, in this list we mostly only list and show the --option version of th
em.  (This  con-
       cept with --no options was added in 7.19.0. Previously most options were t
oggled on/off on
       repeated use of the same command line option.)

       -#, --progress-bar
              Make curl display progress as a simple progress bar instead of th
```

```
e  standard,  more
              informational, meter.

       -0, --http1.0
              (HTTP) Forces curl to issue its requests using HTTP 1.0 instead of
using its inter-
              nally preferred: HTTP 1.1.

       -1, --tlsv1
              (SSL) Forces curl to use TLS version 1 when negotiating with a remo
te TLS server.

       -2, --sslv2
              (SSL) Forces curl to use SSL version 2 when negotiating with a remo
te SSL server.

       -3, --sslv3
              (SSL) Forces curl to use SSL version 3 when negotiating with a remo
te SSL server.

       -4, --ipv4
              If libcurl is capable of resolving an address to multiple IP versio
ns (which it  is
              if  it  is  IPv6-capable),  this  option  tells  libcurl  to  resol
ve names to IPv4
              addresses only.

       -6, --ipv6
              If libcurl is capable of resolving an address to multiple IP versio
ns (which it  is
              if  it  is  IPv6-capable),  this  option  tells  libcurl  to  resol
ve names to IPv6
              addresses only.  default statistics.

       -a, --append
              (FTP/SFTP) When used in an upload, this will tell curl to append t
o the target file
              instead  of  overwriting  it.  If the file doesn't exist, it will b
e created.  Note
              that this flag is ignored by some SSH servers (including OpenSSH).

       -A, --user-agent <agent string>
              (HTTP) Specify the User-Agent string to send to the HTTP server.  S
ome  badly  done
```

```
              CGIs fail if this field isn't set to "Mozilla/4.0". To encode blank
s in the string,
              surround the string with single quote marks. This can also  be  se
t  with  the  -H,
              --header option of course.

              If this option is set more than once, the last one will be the one
that's used.

       --anyauth
              (HTTP)  Tells  curl to figure out authentication method by itself,
and use the most
              secure one the remote site claims to support. This is done by firs
t doing a request
              and  checking  the response-headers, thus possibly inducing an extr
a network round-
              trip. This is used instead of setting a specific authentication met
hod,  which  you
              can do with --basic, --digest, --ntlm, and --negotiate.

              Note that using --anyauth is not recommended if you do uploads fro
m stdin, since it
              may require data to be sent twice and then the client must be abl
e  to  rewind.  If
              the need should arise when uploading from stdin, the upload operati
on will fail.

       -b, --cookie <name=data>
              (HTTP) Pass the data to the HTTP server as a cookie. It is supposed
ly the data pre-
              viously received from the server in a "Set-Cookie:" line.  The dat
a  should  be  in
              the format "NAME1=VALUE1; NAME2=VALUE2".

              If  no  '='  symbol is used in the line, it is treated as a filenam
e to use to read
              previously stored cookie lines from, which should be used in this s
ession  if  they
              match.  Using  this  method also activates the "cookie parser" whic
h will make curl
              record incoming cookies too, which may be handy if you're using thi
s in combination
              with  the  -L,  --location option. The file format of the file to r
ead cookies from
```

```
              should be plain HTTP headers or the Netscape/Mozilla cookie file fo
rmat.

              NOTE that the file specified with -b, --cookie is only used as  inp
ut.  No  cookies
              will  be  stored  in the file. To store cookies, use the -c, --cook
ie-jar option or
              you could even save the HTTP headers to a file using -D, --dump-hea
der!

              If this option is set more than once, the last one will be the one
that's used.

       -B, --use-ascii
              Enable ASCII transfer when using FTP or LDAP. For FTP, this can als
o be enforced by
              using an URL that ends with ";type=A". This option causes data sen
t to stdout to be
              in text mode for win32 systems.

       --basic
              (HTTP) Tells curl to use HTTP Basic authentication. This is the  de
fault  and  this
              option  is usually pointless, unless you use it to override a previ
ously set option
              that sets a different authentication method (such as --ntlm, --dige
st, or --negoti-
              ate).

       -c, --cookie-jar <file name>
              Specify  to  which file you want curl to write all cookies after a
completed opera-
              tion. Curl writes all cookies previously read from a specified fil
e as well as  all
              cookies  received  from  remote server(s). If no cookies are know
n, no file will be
              written. The file will be written using the Netscape cookie file fo
rmat. If you set
              the file name to a single dash, "-", the cookies will be written t
o stdout.

              This command line option will activate the cookie engine that make
s curl record and
              use cookies. Another way to activate it is to use the -b, --cookie
```

```
option.

              If the cookie jar can't be created or written to, the whole  curl
operation  won't
              fail  or  even  report an error clearly. Using -v will get a warnin
g displayed, but
              that is the only visible feedback you get about this possibly letha
l situation.

              If this option is used several times, the last specified file name
will be used.

       -C, --continue-at <offset>
              Continue/Resume a previous file transfer at the given offset. The g
iven  offset  is
              the  exact number of bytes that will be skipped, counting from the
beginning of the
              source file before it is transferred to the destination.  If used w
ith uploads, the
              FTP server command SIZE will not be used by curl.

              Use "-C -" to tell curl to automatically find out where/how to resu
me the transfer.
              It then uses the given output/input files to figure that out.

              If this option is used several times, the last one will be used.

       --ciphers <list of ciphers>
              (SSL) Specifies which ciphers to use in the connection. The list  o
f  ciphers  must
              specify  valid  ciphers.  Read  up  on  SSL  cipher  list  detail
s  on  this  URL:
              http://www.openssl.org/docs/apps/ciphers.html

              NSS ciphers are done differently than OpenSSL and GnuTLS.  The  ful
l  list  of  NSS
              ciphers  is  in  the NSSCipherSuite entry at this URL: http://direc
tory.fedora.red-
              hat.com/docs/mod_nss.html#Directives

              If this option is used several times, the last one will override th
e others.

       --compressed
```

```
              (HTTP) Request a compressed response using one of the algorithms li
bcurl  supports,
              and save the uncompressed document.  If this option is used and th
e server sends an
              unsupported encoding, curl will report an error.

       --connect-timeout <seconds>
              Maximum time in seconds that you allow the connection to the serve
r to take.   This
              only limits the connection phase, once curl has connected this opti
on is of no more
              use. See also the -m, --max-time option.

              If this option is used several times, the last one will be used.

       --create-dirs
              When used in conjunction with the -o option, curl will create the
necessary  local
              directory  hierarchy  as needed. This option creates the dirs menti
oned with the -o
              option, nothing else. If the -o file name uses no dir or if the  di
rs  it  mentions
              already exist, no dir will be created.

              To create remote directories when using FTP or SFTP, try --ftp-crea
te-dirs.

       --crlf (FTP) Convert LF to CRLF in upload. Useful for MVS (OS/390).

       --crlfile <file>
              (HTTPS/FTPS)  Provide  a  file  using PEM format with a Certificat
e Revocation List
              that may specify peer certificates that are to be considered revoke
d.

              If this option is used several times, the last one will be used.

              (Added in 7.19.7)

       -d, --data <data>
              (HTTP) Sends the specified data in a POST request to the HTTP serve
r, in  the  same
              way that a browser does when a user has filled in an HTML form and
presses the sub-
```

```
              mit button. This will cause curl to pass the data to the server usi
ng the  content-
              type application/x-www-form-urlencoded.  Compare to -F, --form.

              -d,  --data  is  the  same  as --data-ascii. To post data purely bi
nary, you should
              instead use the --data-binary option. To URL-encode the value of a
form  field  you
              may use --data-urlencode.

              If  any  of these options is used more than once on the same comman
d line, the data
              pieces specified will be merged together with a separating  &-symbo
l.  Thus,  using
              '-d  name=daniel  -d  skill=lousy'  would  generate  a  post  chun
k that looks like
              'name=daniel&skill=lousy'.

              If you start the data with the letter @, the rest should be a file
name to read the
              data  from, or - if you want curl to read the data from stdin.  Th
e contents of the
              file must already be URL-encoded. Multiple files can  also  be  spe
cified.  Posting
              data from a file named 'foobar' would thus be done with --data @foo
bar.

       -D, --dump-header <file>
              Write the protocol headers to the specified file.

              This  option  is  handy  to use when you want to store the headers
that a HTTP site
              sends to you. Cookies from the headers could then be read in a seco
nd curl  invoca-
              tion  by  using  the  -b, --cookie option! The -c, --cookie-jar opt
ion is however a
              better way to store cookies.

              When used in FTP, the FTP server response lines are considered bein
g "headers"  and
              thus are saved there.

              If  this option is used several times, the last one will be used.
IP "--data-ascii
```

```
              <data>" See -d, --data.

       --data-binary <data>
              (HTTP) This posts data exactly as specified with no extra processin
g whatsoever.

              If you start the data with the letter @, the rest should be a  file
name.   Data  is
              posted in a similar manner as --data-ascii does, except that newlin
es are preserved
              and conversions are never done.

              If this option is used several times, the ones following the first
will append data
              as described in -d, --data.

       --data-urlencode <data>
              (HTTP) This posts data, similar to the other --data options with th
e exception that
              this performs URL-encoding. (Added in 7.18.0)

              To be CGI-compliant, the <data> part should begin with a name follo
wed by a separa-
              tor and a content specification. The <data> part can be passed to c
url using one of
              the following syntaxes:

              content
                     This will make curl URL-encode the content and pass that o
n. Just be careful
                     so  that  the  content doesn't contain any = or @ symbols, a
s that will then
                     make the syntax match one of the other cases below!

              =content
                     This will make curl URL-encode the content and pass that o
n. The preceding =
                     symbol is not included in the data.

              name=content
                     This  will make curl URL-encode the content part and pass th
at on. Note that
                     the name part is expected to be URL-encoded already.
              @filename
                     This will make curl load data from the given file (includin
g any  newlines),
                     URL-encode that data and pass it on in the POST.

              name@filename
                     This  will make curl load data from the given file (includin
g any newlines),
                     URL-encode that data and pass it on in the POST. The name pa
rt gets an equal
                     sign appended, resulting in name=urlencoded-file-content. No
te that the name
                     is expected to be URL-encoded already.

       --delegation LEVEL
              Set LEVEL to tell the server what it is allowed to delegate when i
t comes  to  user
              credentials. Used with GSS/kerberos.

              none   Don't allow any delegation.

              policy Delegates if and only if the OK-AS-DELEGATE flag is set in t
he Kerberos ser-
                     vice ticket, which is a matter of realm policy.

              always Unconditionally allow the server to delegate.

       --digest
              (HTTP) Enables HTTP Digest authentication. This is a authenticatio
n  that  prevents
              the  password  from being sent over the wire in clear text. Use thi
s in combination
              with the normal -u, --user option to set user name and password. Se
e  also  --ntlm,
              --negotiate and --anyauth for related options.

              If this option is used several times, the following occurrences mak
e no difference.

       --disable-eprt
              (FTP)  Tell curl to disable the use of the EPRT and LPRT commands w
hen doing active
              FTP transfers. Curl will normally always first  attempt  to  use  E
PRT,  then  LPRT
```

```
              before using PORT, but with this option, it will use PORT right awa
y. EPRT and LPRT
              are extensions to the original FTP protocol, and may not work on al
l  servers,  but
              they enable more functionality in a better way than the traditiona
l PORT command.

              --eprt  can  be  used to explicitly enable EPRT again and --no-epr
t is an alias for
              --disable-eprt.

              Disabling EPRT only changes the active behavior. If you want to swi
tch  to  passive
              mode you need to not use -P, --ftp-port or force it with --ftp-pas
v.

       --disable-epsv
              (FTP)  Tell  curl  to  disable  the  use of the EPSV command when d
oing passive FTP
              transfers. Curl will normally always first attempt to use  EPSV  be
fore  PASV,  but
              with this option, it will not try using EPSV.

              --epsv  can  be  used to explicitly enable EPSV again and --no-eps
v is an alias for
              --disable-epsv.

              Disabling EPSV only changes the passive behavior. If you want to sw
itch  to  active
              mode you need to use -P, --ftp-port.

       -e, --referer <URL>
              (HTTP)  Sends  the  "Referer Page" information to the HTTP server.
This can also be
              set with the -H, --header flag of course.  When used with -L,  --lo
cation  you  can
              append ";auto" to the --referer URL to make curl automatically set
the previous URL
              when it follows a Location: header. The ";auto" string can be used
alone,  even  if
              you don't set an initial --referer.

              If this option is used several times, the last one will be used.
       -E, --cert <certificate[:password]>
              (SSL)  Tells  curl to use the specified client certificate file whe
n getting a file
              with HTTPS, FTPS or another SSL-based protocol. The certificate mus
t be in PEM for-
              mat.   If the optional password isn't specified, it will be querie
d for on the ter-
              minal. Note that this option assumes a "certificate" file that is t
he  private  key
              and  the  private  certificate  concatenated!  See --cert and --ke
y to specify them
              independently.

              If curl is built against the NSS SSL library then this option  ca
n  tell  curl  the
              nickname  of the certificate to use within the NSS database define
d by the environ-
              ment variable SSL_DIR (or by default /etc/pki/nssdb). If the NSS PE
M PKCS#11 module
              (libnsspem.so) is available then PEM files may be loaded. If you wa
nt to use a file
              from the current directory, please precede it with "./" prefix, in
order  to  avoid
              confusion with a nickname.

              If this option is used several times, the last one will be used.

       --engine <name>
              Select the OpenSSL crypto engine to use for cipher operations. Use
--engine list to
              print a list of build-time supported engines. Note that not all (o
r  none)  of  the
              engines may be available at run-time.

       --environment
              (RISC OS ONLY) Sets a range of environment variables, using the nam
es the -w option
              supports, to allow easier extraction of useful information after ha
ving run curl.

       --egd-file <file>
              (SSL) Specify the path name to the Entropy Gathering Daemon socke
t. The  socket  is
              used  to  seed  the  random  engine for SSL connections. See also t
```

```
he --random-file
              option.

       --cert-type <type>
              (SSL) Tells curl what certificate type the provided certificate is
in. PEM, DER and
              ENG are recognized types.  If not specified, PEM is assumed.

              If this option is used several times, the last one will be used.

       --cacert <CA certificate>
              (SSL) Tells curl to use the specified certificate file to verify th
e peer. The file
              may contain multiple CA certificates. The certificate(s) must  be
in  PEM  format.
              Normally  curl is built to use a default file for this, so this opt
ion is typically
              used to alter that default file.

              curl recognizes the environment variable named 'CURL_CA_BUNDLE' if
it is  set,  and
              uses the given path as a path to a CA cert bundle. This option over
rides that vari-
              able.

              The windows version of curl will automatically look  for  a  CA  ce
rts  file  named
              'curl-ca-bundle.crt',  either  in the same directory as curl.exe, o
r in the Current
              Working Directory, or in any folder along your PATH.

              If curl is built against the NSS SSL library then this option tell
s curl the  nick-
              name  of  the CA certificate to use within the NSS database define
d by the environ-
              ment variable SSL_DIR (or by default /etc/pki/nssdb).  If the NSS P
EM PKCS#11  mod-
              ule (libnsspem.so) is available then PEM files may be loaded.

              If this option is used several times, the last one will be used.

       --capath <CA certificate directory>
              (SSL)  Tells  curl  to  use the specified certificate directory to
verify the peer.
```

```
              Multiple  paths   can   be   provided   by   separating   them   wi
th   ":"   (e.g.
              "path1:path2:path3").  The certificates must be in PEM format, and
if curl is built
              against OpenSSL, the directory must have been processed using the
c_rehash  utility
              supplied  with  OpenSSL. Using --capath can allow OpenSSL-powered c
url to make SSL-
              connections much more efficiently than using --cacert if the --cace
rt file contains
              many CA certificates.

              If  this option is set, the default capath value will be ignored, a
nd if it is used
              several times, the last one will be used.

       -f, --fail
              (HTTP) Fail silently (no output at all) on server errors. This is
mostly  done  to
              better enable scripts etc to better deal with failed attempts. In n
ormal cases when
              a HTTP server fails to deliver a document, it returns an HTML docum
ent  stating  so
              (which  often  also  describes why and more). This flag will preven
t curl from out-
              putting that and return error 22.

              This method is not fail-safe and there are occasions where non-succ
essful  response
              codes will slip through, especially when authentication is involve
d (response codes
              401 and 407).

       -F, --form <name=content>
              (HTTP) This lets curl emulate a filled-in form in which a user has
pressed the sub-
              mit  button.  This  causes curl to POST data using the Content-Typ
e multipart/form-
              data according to RFC 2388. This enables uploading of binary file
s  etc.  To  force
              the  'content'  part to be a file, prefix the file name with an @ s
ign. To just get
              the content part from a file, prefix the file name with the symbol
<.  The  differ-
```

```
              ence between @ and < is then that @ makes a file get attached in th
e post as a file
              upload, while the < makes a text field and just get  the  content
s  for  that  text
              field from a file.

              Example,  to send your password file to the server, where 'passwor
d' is the name of
              the form-field to which /etc/passwd will be the input:

              curl -F password=@/etc/passwd www.mypasswords.com

              To read content from stdin instead of a file, use - as the filenam
e. This goes  for
              both @ and < constructs.

              You can also tell curl what Content-Type to use by using 'type=', i
n a manner simi-
              lar to:

              curl -F "web=@index.html;type=text/html" url.com

              or

              curl -F "name=daniel;type=text/foo" url.com

              You can also explicitly change the name field of a  file  upload  p
art  by  setting
              filename=, like this:

              curl -F "file=@localfile;filename=nameinpost" url.com

              See further examples and details in the MANUAL.

              This option can be used multiple times.

       --ftp-account [data]
              (FTP)  When  an FTP server asks for "account data" after user name
and password has
              been provided, this data is sent off using the ACCT command. (Adde
d in 7.13.0)

              If this option is used twice, the second will override the previou
s use.
```

```
       --ftp-alternative-to-user <command>
              (FTP) If authenticating with the USER and PASS commands fails, sen
d  this  command.
              When  connecting  to  Tumbleweed's Secure Transport server over FTP
S using a client
              certificate, using "SITE AUTH" will tell the server to retrieve th
e  username  from
              the certificate. (Added in 7.15.5)

       --ftp-create-dirs
              (FTP/SFTP)  When  an  FTP  or SFTP URL/operation uses a path that d
oesn't currently
              exist on the server, the standard behavior of curl is to fail. Usin
g  this  option,
              curl will instead attempt to create missing directories.

       --ftp-method [method]
              (FTP)  Control  what method curl should use to reach a file on a FT
P(S) server. The
              method argument should be one of the following alternatives:

              multicwd
                     curl does a single CWD operation for each path part in the
given  URL.  For
                     deep hierarchies this means very many commands. This is how
RFC 1738 says it
                     should be done. This is the default but the slowest behavio
r.

              nocwd  curl does no CWD at all. curl will do SIZE, RETR, STOR etc a
nd give  a  full
                     path to the server for all these commands. This is the faste
st behavior.

              singlecwd
                     curl  does  one  CWD with the full target directory and the
n operates on the
                     file "normally" (like in the multicwd case). This is somewha
t more standards
                     compliant than 'nocwd' but without the full penalty of 'mult
icwd'.
       (Added in 7.15.1)
       --ftp-pasv
              (FTP)  Use  passive  mode  for the data connection. Passive is the
internal default
              behavior, but using this option can be used to  override  a  previo
us  -P/-ftp-port
              option. (Added in 7.11.0)

              If this option is used several times, the following occurrences mak
e no difference.
              Undoing an enforced passive really isn't doable but you must then
instead  enforce
              the correct -P, --ftp-port again.

              Passive  mode means that curl will try the EPSV command first and t
hen PASV, unless
              --disable-epsv is used.

       --ftp-skip-pasv-ip
              (FTP) Tell curl to not use the IP address the server suggests in  i
ts  response  to
              curl's  PASV  command when curl connects the data connection. Inste
ad curl will re-
              use the same IP address it already uses  for  the  control  connect
ion.  (Added  in
              7.14.2)

              This option has no effect if PORT, EPRT or EPSV is used instead of
PASV.

       --ftp-pret
              (FTP) Tell curl to send a PRET command before PASV (and EPSV). Cert
ain FTP servers,
              mainly drftpd, require this non-standard command for directory list
ings as well  as
              up and downloads in PASV mode.  (Added in 7.20.x)

       --ftp-ssl-ccc
              (FTP)  Use CCC (Clear Command Channel) Shuts down the SSL/TLS laye
r after authenti-
              cating. The rest of the control channel communication  will  be  un
encrypted.  This
              allows  NAT routers to follow the FTP transaction. The default mod
e is passive. See
              --ftp-ssl-ccc-mode for other modes.  (Added in 7.16.1)
       --ftp-ssl-ccc-mode [active/passive]
              (FTP) Use CCC (Clear Command Channel) Sets the CCC mode. The passiv
e mode will  not
              initiate the shutdown, but instead wait for the server to do it, an
d will not reply
              to the shutdown from the server. The active mode initiates the shut
down  and  waits
              for a reply from the server.  (Added in 7.16.2)

       --ftp-ssl-control
              (FTP) Require SSL/TLS for the FTP login, clear for transfer.  Allow
s secure authen-
              tication, but non-encrypted data transfers for efficiency.  Fails t
he  transfer  if
              the  server  doesn't support SSL/TLS.  (Added in 7.16.0) that can s
till be used but
              will be removed in a future version.

       --form-string <name=string>
              (HTTP) Similar to --form except that the value string for the  name
d  parameter  is
              used  literally.  Leading  '@'  and  '<' characters, and the ';typ
e=' string in the
              value have no special meaning. Use this in preference to --form if
there's any pos-
              sibility  that the string value may accidentally trigger the '@' o
r '<' features of
              --form.

       -g, --globoff
              This option switches off the "URL globbing parser". When you set th
is  option,  you
              can  specify  URLs  that  contain the letters {}[] without having t
hem being inter-
              preted by curl itself. Note that these letters are not normal  lega
l  URL  contents
              but they should be encoded according to the URI standard.

       -G, --get
              When  used,  this  option  will  make all data specified with -d,
--data or --data-
              binary to be used in a HTTP GET request instead of the POST reques
t that  otherwise
```

```
              would be used. The data will be appended to the URL with a '?' sepa
rator.

              If  used  in combination with -I, the POST data will instead be app
ended to the URL
              with a HEAD request.

              If this option is used several times, the following occurrences mak
e no difference.
              This  is  because  undoing  a  GET  doesn't make sense, but you sho
uld then instead
              enforce the alternative method you prefer.

       -H, --header <header>
              (HTTP) Extra header to use when getting a web page. You may specif
y any  number  of
              extra  headers.  Note that if you should add a custom header that h
as the same name
              as one of the internal ones curl would use, your externally set hea
der will be used
              instead  of the internal one. This allows you to make even trickie
r stuff than curl
              would normally do. You should not replace internally set  headers
without  knowing
              perfectly well what you're doing. Remove an internal header by givi
ng a replacement
              without content on the right side of the colon, as in: -H "Host:".
If you send  the
              custom  header  with  no-value then its header must be terminated w
ith a semicolon,
              such as -H "X-Custom-Header;" to send "X-Custom-Header:".

              curl will make sure that each header you add/replace is sent with t
he  proper  end-
              of-line  marker,  you  should thus not add that as a part of the he
ader content: do
              not add newlines or carriage returns, they will only mess things u
p for you.

              See also the -A, --user-agent and -e, --referer options.

              This option can be used multiple times to add/replace/remove multip
le headers.
```

```
       --hostpubmd5 <md5>
              Pass a string containing 32 hexadecimal digits. The string should b
e  the  128  bit
              MD5  checksum of the remote host's public key, curl will refuse th
e connection with
              the host unless the md5sums match. This option is only for SCP and
SFTP  transfers.
              (Added in 7.17.1)

       --ignore-content-length
              (HTTP)  Ignore  the  Content-Length header. This is particularly us
eful for servers
              running Apache 1.x, which will report incorrect  Content-Length  fo
r  files  larger
              than 2 gigabytes.

       -i, --include
              (HTTP)  Include the HTTP-header in the output. The HTTP-header incl
udes things like
              server-name, date of the document, HTTP-version and more...

       -I, --head
              (HTTP/FTP/FILE) Fetch the HTTP-header only! HTTP-servers feature th
e  command  HEAD
              which  this uses to get nothing but the header of a document. When
used on a FTP or
              FILE file, curl displays the file size and last modification time o
nly.

       --interface <name>
              Perform an operation using a specified interface. You can enter int
erface name,  IP
              address or host name. An example could look like:

               curl --interface eth0:1 http://www.netscape.com/

              If this option is used several times, the last one will be used.

       -j, --junk-session-cookies
              (HTTP)  When  curl is told to read cookies from a given file, this
option will make
              it discard all "session cookies". This will basically have the sam
e effect as if  a
              new  session  is  started.  Typical  browsers  always  discard sess
```

```
ion cookies when
              they're closed down.

       -J, --remote-header-name
              (HTTP) This option tells the -O, --remote-name option to use  the
server-specified
              Content-Disposition filename instead of extracting a filename from
the URL.

       -k, --insecure
              (SSL)  This option explicitly allows curl to perform "insecure" SS
L connections and
              transfers. All SSL connections are attempted to be made secure by u
sing the CA cer-
              tificate  bundle installed by default. This makes all connections c
onsidered "inse-
              cure" fail unless -k, --insecure is used.

              See       this       online       resource       for        furthe
r        details:
              http://curl.haxx.se/docs/sslcerts.html

       -K, --config <config file>
              Specify  which  config  file to read curl arguments from. The confi
g file is a text
              file in which command line arguments can be written which then wil
l be used  as  if
              they  were written on the actual command line. Options and their pa
rameters must be
              specified on the same config file line, separated by whitespace, co
lon, the  equals
              sign  or  any  combination  thereof (however, the preferred separat
or is the equals
              sign). If the parameter is to contain whitespace, the parameter  mu
st  be  enclosed
              within  quotes. Within double quotes, the following escape sequence
s are available:
              \\, \", \t, \n, \r and \v. A backslash preceding any other letter
is  ignored.  If
              the  first column of a config line is a '#' character, the rest of
the line will be
              treated as a comment. Only write one option per physical line in th
e config file.
```

```
              Specify the filename to -K, --config as '-' to make curl read the f
ile from stdin.

              Note that to be able to specify a URL in the config file, you nee
d  to  specify  it
              using  the  --url option, and not by simply writing the URL on its
own line. So, it
              could look similar to this:

              url = "http://curl.haxx.se/docs/"

              Long option names can optionally be given in the config file  witho
ut  the  initial
              double dashes.

              When  curl  is  invoked,  it always (unless -q is used) checks for
a default config
              file and uses it if found. The default config file is checked for i
n the  following
              places in this order:

              1)  curl  tries  to find the "home dir": It first checks for the CU
RL_HOME and then
              the HOME environment variables. Failing that, it uses getpwuid() o
n UNIX-like  sys-
              tems  (which  returns  the home dir given the current user in your
system). On Win-
              dows, it then checks for the APPDATA variable, or as a last resor
t  the  '%USERPRO-
              FILE%\Application Data'.

              2)  On  windows,  if there is no _curlrc file in the home dir, it c
hecks for one in
              the same dir the curl executable is placed. On UNIX-like systems,
it  will  simply
              try to load .curlrc from the determined home dir.

              # --- Example file ---
              # this is a comment
              url = "curl.haxx.se"
              output = "curlhere.html"
              user-agent = "superagent/1.0"

              # and fetch another URL too
              url = "curl.haxx.se/docs/manpage.html"
              -O
              referer = "http://nowhereatall.com/"
              # --- End of example file ---

              This option can be used multiple times to load multiple config file
s.

       --keepalive-time <seconds>
              This  option  sets  the  time  a  connection  needs  to  remain idl
e before sending
              keepalive probes and the time between individual keepalive probes.
It is  currently
              effective  on  operating systems offering the TCP_KEEPIDLE and TC
P_KEEPINTVL socket
              options (meaning Linux, recent AIX, HP-UX and more). This option ha
s no  effect  if
              --no-keepalive is used. (Added in 7.18.0)

              If  this  option  is  used  multiple times, the last occurrence set
s the amount. If
              unspecified, the option defaults to 60 seconds.

       --key <key>
              (SSL/SSH) Private key file name. Allows you to provide your  privat
e  key  in  this
              separate file.

              If this option is used several times, the last one will be used.

       --key-type <type>
              (SSL) Private key file type. Specify which type your --key provide
d private key is.
              DER, PEM, and ENG are supported. If not specified, PEM is assumed.

              If this option is used several times, the last one will be used.

       --krb <level>
              (FTP) Enable Kerberos authentication and use. The level must be ent
ered and  should
              be  one  of  'clear',  'safe', 'confidential', or 'private'. Shoul
d you use a level
              that is not one of these, 'private' will instead be used.
              This option requires a library built with kerberos4 or GSSAPI (GSS-
Negotiate)  sup-
              port. This is not very common. Use -V, --version to see if your cur
l supports it.

              If this option is used several times, the last one will be used.

       -l, --list-only
              (FTP)  When  listing  an FTP directory, this switch forces a name-o
nly view.  Espe-
              cially useful if you want to machine-parse the contents of an FTP
directory  since
              the normal directory view doesn't use a standard look or format.

              This  option  causes  an  FTP  NLST command to be sent.  Some FTP s
ervers list only
              files in their response to NLST; they do not include  subdirectorie
s  and  symbolic
              links.



       -L, --location
              (HTTP/HTTPS) If the server reports that the requested page has move
d to a different
              location (indicated with a Location: header and a 3XX response cod
e),  this  option
              will  make  curl  redo  the  request  on  the  new place. If used t
ogether with -i,
              --include or -I, --head, headers from all  requested  pages  will
be  shown.  When
              authentication  is  used, curl only sends its credentials to the in
itial host. If a
              redirect takes curl to a  different  host,  it  won't  be  able  t
o  intercept  the
              user+password. See also --location-trusted on how to change this. Y
ou can limit the
              amount of redirects to follow by using the --max-redirs option.

              When curl follows a redirect and the request is not a plain GET (fo
r  example  POST
              or  PUT), it will do the following request with a GET if the HTTP r
esponse was 301,
              302, or 303. If the response code was any other 3xx code,  curl  wi
ll  re-send  the
```

```
              following request using the same unmodified method.

       --libcurl <file>
              Append  this  option to any ordinary curl command line, and you wil
l get a libcurl-
              using C source code written to the file that does the equivalent o
f what your  com-
              mand-line operation does!

              If this option is used several times, the last given file name wil
l be used. (Added
              in 7.16.1)

       --limit-rate <speed>
              Specify the maximum transfer rate you want curl to use. This featur
e is  useful  if
              you  have  a limited pipe and you'd like your transfer not to use y
our entire band-
              width.

              The given speed is measured in bytes/second, unless a suffix is app
ended.   Append-
              ing  'k'  or  'K' will count the number as kilobytes, 'm' or M' mak
es it megabytes,
              while 'g' or 'G' makes it gigabytes. Examples: 200K, 3m and 1G.

              The given rate is the average speed counted during the entire  tran
sfer.  It  means
              that  curl  might use higher transfer speeds in short bursts, but o
ver time it uses
              no more than the given rate.

              If you also use the -Y, --speed-limit option, that option will tak
e precedence  and
              might  cripple  the  rate-limiting  slightly, to help keeping the s
peed-limit logic
              working.

              If this option is used several times, the last one will be used.

       --local-port <num>[-num]
              Set a preferred number or range of local port numbers to use for th
e connection(s).
              Note  that  port numbers by nature are a scarce resource that will
```

```
be busy at times
              so setting this range to something too narrow might  cause  unneces
sary  connection
              setup failures. (Added in 7.15.2)

       --location-trusted
              (HTTP/HTTPS) Like -L, --location, but will allow sending the name
+ password to all
              hosts that the site may redirect to. This may  or  may  not  introd
uce  a  security
              breach if the site redirects you to a site to which you'll send you
r authentication
              info (which is plaintext in the case of HTTP Basic authentication).

       -m, --max-time <seconds>
              Maximum time in seconds that you allow the whole operation to tak
e.  This is useful
              for preventing your batch jobs from hanging for hours due to slow n
etworks or links
              going down.  See also the --connect-timeout option.

              If this option is used several times, the last one will be used.

       --mail-auth <address>
              (SMTP) Specify a single address. This will be used to  specify  th
e  authentication
              address (identity) of a submitted message that is being relayed to
another server.

              (Added in 7.25.0)

       --mail-from <address>
              (SMTP) Specify a single address that the given mail should get sen
t from.

              (Added in 7.20.0)

       --max-filesize <bytes>
              Specify the maximum size (in bytes) of a file to download. If the f
ile requested is
              larger than this value, the transfer will not start and curl will r
eturn with  exit
              code 63.
              NOTE:  The file size is not always known prior to download, and fo
r such files this
              option has no effect even if the file transfer ends up being large
r than this given
              limit. This concerns both FTP and HTTP transfers.

       --mail-rcpt <address>
              (SMTP) Specify a single address that the given mail should get sen
t to. This option
              can be used multiple times to specify many recipients.

              (Added in 7.20.0)

       --max-redirs <num>
              Set maximum number of redirection-followings allowed. If -L,  --loc
ation  is  used,
              this  option can be used to prevent curl from following redirection
s "in absurdum".
              By default, the limit is set to 50 redirections. Set this option t
o -1 to  make  it
              limitless.

              If this option is used several times, the last one will be used.

       -n, --netrc
              Makes  curl  scan  the .netrc (_netrc on Windows) file in the use
r's home directory
              for login name and password. This is typically used for FTP on UNI
X. If  used  with
              HTTP,  curl  will enable user authentication. See netrc(4) or ft
p(1) for details on
              the file format. Curl will not complain if that file doesn't have t
he right permis-
              sions  (it should not be either world- or group-readable). The envi
ronment variable
              "HOME" is used to find the home directory.

              A quick and very simple example of how to setup a .netrc to allow c
url  to  FTP  to
              the  machine  host.domain.com  with user name 'myself' and passwor
d 'secret' should
              look similar to:

              machine host.domain.com login myself password secret
       -N, --no-buffer
              Disables the buffering of the output stream. In normal work situati
ons,  curl  will
              use a standard buffered output stream that will have the effect tha
t it will output
              the data in chunks, not necessarily exactly when  the  data  arrive
s.   Using  this
              option will disable that buffering.

              Note  that this is the negated option name documented. You can thu
s use --buffer to
              enforce the buffering.

       --netrc-file
              This option is similar to --netrc, except that you provide the  pat
h  (absolute  or
              relative)  to  the netrc file that Curl should use.  You can only s
pecify one netrc
              file per invocation. If several --netrc-file options are provide
d,  only  the  last
              one will be used.  (Added in 7.21.5)

              This  option  overrides any use of --netrc as they are mutually exc
lusive.  It will
              also abide by --netrc-optional if specified.


       --netrc-optional
              Very similar to --netrc, but this option makes the .netrc usage  op
tional  and  not
              mandatory as the --netrc option does.


       --negotiate
              (HTTP)  Enables GSS-Negotiate authentication. The GSS-Negotiate met
hod was designed
              by Microsoft and is used in their web applications. It is primaril
y meant as a sup-
              port for Kerberos5 authentication but may be also used along with a
nother authenti-
              cation method. For more information see IETF draft draft-brezak-spn
ego-http-04.txt.
```

```
              If you want to enable Negotiate for your proxy authentication,  the
n  use  --proxy-
              negotiate.

              This  option requires a library built with GSSAPI support. This is
not very common.
              Use -V, --version to see if your version supports GSS-Negotiate.

              When using this option, you must also provide a fake -u, --user opt
ion to  activate
              the  authentication  code properly. Sending a '-u :' is enough as t
he user name and
              password from the -u option aren't actually used.

              If this option is used several times, the following occurrences mak
e no difference.

       --no-keepalive
              Disables the use of keepalive messages on the TCP connection, as  b
y  default  curl
              enables them.

              Note  that this is the negated option name documented. You can thu
s use --keepalive
              to enforce keepalive.

       --no-sessionid
              (SSL) Disable curl's use of SSL session-ID caching.  By default al
l  transfers  are
              done using the cache. Note that while nothing should ever get hurt
by attempting to
              reuse SSL session-IDs, there seem to be broken SSL implementations
in the wild that
              may require you to disable this in order for you to succeed. (Adde
d in 7.16.0)

              Note  that this is the negated option name documented. You can thu
s use --sessionid
              to enforce session-ID caching.

       --noproxy <no-proxy-list>
              Comma-separated list of hosts which do not use a proxy, if one is
specified.   The
              only  wildcard  is  a  single * character, which matches all host
```

```
s, and effectively
              disables the proxy. Each name in this list is matched as either a d
omain which con-
              tains  the  hostname,  or  the  hostname itself. For example, loca
l.com would match
              local.com, local.com:80, and www.local.com, but not  www.notlocal.c
om.   (Added  in
              7.19.4).

       --ntlm (HTTP)  Enables NTLM authentication. The NTLM authentication metho
d was designed by
              Microsoft and is used by IIS web servers. It is a  proprietary  pro
tocol,  reverse-
              engineered  by  clever  people and implemented in curl based on the
ir efforts. This
              kind of behavior should not be endorsed, you should  encourage  eve
ryone  who  uses
              NTLM  to  switch  to a public and documented authentication method
instead, such as
              Digest.

              If you want to enable NTLM for your proxy authentication, then use
--proxy-ntlm.

              This option requires a library built with SSL support. Use -V, --ve
rsion to see  if
              your curl supports NTLM.

              If this option is used several times, the following occurrences mak
e no difference.

       -o, --output <file>
              Write output to <file> instead of stdout. If you are using {} or
[] to fetch multi-
              ple documents, you can use '#' followed by a number in the <file>
specifier.  That
              variable  will  be replaced with the current string for the URL bei
ng fetched. Like
              in:

                curl http://{one,two}.site.com -o "file_#1.txt"

              or use several variables like:
                curl http://{site,host}.host[1-5].com -o "#1_#2"

              You may use this option as many times as the number of URLs you hav
e.

              See also the --create-dirs option to  create  the  local  directori
es  dynamically.
              Specifying  the  output  as '-' (a single dash) will force the outp
ut to be done to
              stdout.

       -O, --remote-name
              Write output to a local file named like the remote file we get. (On
ly the file part
              of the remote file is used, the path is cut off.)

              The  remote  file  name  to use for saving is extracted from the gi
ven URL, nothing
              else.

              Consequentially, the file will be saved in the current working  dir
ectory.  If  you
              want  the file saved in a different directory, make sure you chang
e current working
              directory before you invoke curl with the -O, --remote-name flag!

              You may use this option as many times as the number of URLs you hav
e.

       -p, --proxytunnel
              When an HTTP proxy is used (-x, --proxy), this option will cause no
n-HTTP protocols
              to  attempt  to tunnel through the proxy instead of merely using i
t to do HTTP-like
              operations. The tunnel approach is made with the HTTP  proxy  CONNE
CT  request  and
              requires  that the proxy allows direct connect to the remote port n
umber curl wants
              to tunnel through to.

       -P, --ftp-port <address>
              (FTP) Reverses the default initiator/listener roles when connectin
g with FTP.  This
              switch  makes curl use active mode. In practice, curl then tells th
```

```
e server to con-
              nect back to the client's specified address and port, while passiv
e mode  asks  the
              server  to  setup  an IP address and port for it to connect to. <ad
dress> should be
              one of:

              interface
                     i.e "eth0" to specify which interface's IP address you  wan
t  to  use  (Unix
                     only)

              IP address
                     i.e "192.168.10.1" to specify the exact IP address

              host name
                     i.e "my.host.domain" to specify the machine

              -      make curl pick the same IP address that is already used for
the control con-
                     nection

       If this option is used several times, the last one will be used. Disable t
he use  of  PORT
       with  --ftp-pasv.  Disable  the  attempt  to use the EPRT command instead
of PORT by using
       --disable-eprt. EPRT is really PORT++.

       Starting in 7.19.5, you can append ":[start]-[end]" to the right of the ad
dress,  to  tell
       curl  what  TCP  port range to use. That means you specify a port range, f
rom a lower to a
       higher number. A single number works as well, but do note that it increase
s  the  risk  of
       failure since the port may not be available.

       --pass <phrase>
              (SSL/SSH) Passphrase for the private key

              If this option is used several times, the last one will be used.

       --post301
              Tells  curl  to  respect  RFC  2616/10.3.2  and  not convert POST r
equests into GET
```

```
              requests when following a 301 redirection. The non-RFC behaviour i
s  ubiquitous  in
              web  browsers, so curl does the conversion by default to maintain c
onsistency. How-
              ever, a server may require a POST to remain a POST after such a  re
direction.  This
              option is meaningful only when using -L, --location (Added in 7.1
7.1)

       --post302
              Tells  curl  to  respect  RFC  2616/10.3.2  and  not convert POST r
equests into GET
              requests when following a 302 redirection. The non-RFC behaviour i
s  ubiquitous  in
              web  browsers, so curl does the conversion by default to maintain c
onsistency. How-
              ever, a server may require a POST to remain a POST after such a  re
direction.  This
              option is meaningful only when using -L, --location (Added in 7.1
9.1)

       --proto <protocols>
              Tells  curl  to  use  the listed protocols for its initial retrieva
l. Protocols are
              evaluated left to right, are comma separated, and  are  each  a  pr
otocol  name  or
              'all', optionally prefixed by zero or more modifiers. Available mod
ifiers are:

              +  Permit  this  protocol  in  addition to protocols already permit
ted (this is the
                 default if no modifier is used).

              -  Deny this protocol, removing it from the list of protocols alrea
dy permitted.

              =  Permit only this protocol (ignoring the list already permitte
d), though  subject
                 to later modification by subsequent entries in the comma separat
ed list.

              For example:

              --proto -ftps  uses the default protocols, but disables ftps
              --proto -all,https,+http
                             only enables http and https

              --proto =http,https
                             also only enables http and https

              Unknown  protocols  produce  a warning. This allows scripts to safe
ly rely on being
              able to disable potentially dangerous protocols, without relying up
on  support  for
              that protocol being built into curl to avoid an error.

              This  option  can  be  used multiple times, in which case the effec
t is the same as
              concatenating the protocols into one instance of the option.

              (Added in 7.20.2)

       --proto-redir <protocols>
              Tells curl to use the listed protocols after a redirect. See --prot
o for how proto-
              cols are represented.

              (Added in 7.20.2)

       --proxy-anyauth
              Tells  curl  to  pick  a suitable authentication method when commun
icating with the
              given proxy. This might cause  an  extra  request/response  round-t
rip.  (Added  in
              7.13.2)

       --proxy-basic
              Tells  curl  to  use  HTTP  Basic  authentication when communicatin
g with the given
              proxy. Use --basic for enabling HTTP Basic with a remote host. Basi
c is the default
              authentication method curl uses with proxies.

       --proxy-digest
              Tells  curl  to  use  HTTP  Digest authentication when communicatin
g with the given
              proxy. Use --digest for enabling HTTP Digest with a remote host.
       --proxy-negotiate
              Tells curl to use HTTP Negotiate authentication when communicating
with  the  given
              proxy.  Use  --negotiate  for enabling HTTP Negotiate with a remot
e host. (Added in
              7.17.1)

       --proxy-ntlm
              Tells curl to use HTTP NTLM authentication when communicating with
the given proxy.
              Use --ntlm for enabling NTLM with a remote host.

       --proxy1.0 <proxyhost[:port]>
              Use  the  specified  HTTP  1.0  proxy.  If  the port number is not
specified, it is
              assumed at port 1080.

              The only difference between this and the HTTP proxy option (-x, --p
roxy),  is  that
              attempts to use CONNECT through the proxy will specify an HTTP 1.0
protocol instead
              of the default HTTP 1.1.

       --pubkey <key>
              (SSH) Public key file name. Allows you to provide your public key i
n this  separate
              file.

              If this option is used several times, the last one will be used.

       -q     If used as the first parameter on the command line, the curlrc conf
ig file will not
              be read and used. See the -K, --config for  details  on  the  defau
lt  config  file
              search path.

       -Q, --quote <command>
              (FTP/SFTP)  Send  an arbitrary command to the remote FTP or SFTP se
rver. Quote com-
              mands are sent BEFORE the transfer takes place (just after the init
ial PWD  command
              in  an  FTP  transfer, to be exact). To make commands take place af
ter a successful
```

```
              transfer, prefix them with a dash '-'.  To make commands be sent af
ter libcurl  has
              changed the working directory, just before the transfer comman
d(s), prefix the com-
              mand with a '+' (this is only supported for FTP). You may  specif
y  any  number  of
              commands.  If the server returns failure for one of the commands, t
he entire opera-
              tion will be aborted. You must send syntactically correct FTP comma
nds as  RFC  959
              defines  to FTP servers, or one of the commands listed below to SFT
P servers.  This
              option can be used multiple times. When speaking to a FTP server, p
refix  the  com-
              mand  with an asterisk (*) to make libcurl continue even if the com
mand fails as by
              default curl will stop at first failure.

              SFTP is a binary protocol. Unlike for FTP, libcurl interprets SFT
P  quote  commands
              itself  before sending them to the server.  File names may be quote
d shell-style to
              embed spaces or special characters.  Following is the list of  al
l  supported  SFTP
              quote commands:

              chgrp group file
                     The chgrp command sets the group ID of the file named by th
e file operand to
                     the group ID specified by the group operand. The group opera
nd is a  decimal
                     integer group ID.

              chmod mode file
                     The  chmod  command  modifies  the file mode bits of the spe
cified file. The
                     mode operand is an octal integer mode number.

              chown user file
                     The chown command sets the owner of the file named by the  f
ile  operand  to
                     the  user  ID  specified  by the user operand. The user oper
and is a decimal
                     integer user ID.
              ln source_file target_file
                     The ln and symlink commands create a symbolic link at the ta
rget_file  loca-
                     tion pointing to the source_file location.

              mkdir directory_name
                     The mkdir command creates the directory named by the directo
ry_name operand.

              pwd    The  pwd command returns the absolute pathname of the curren
t working direc-
                     tory.

              rename source target
                     The rename command renames the file or directory named by th
e source operand
                     to the destination path named by the target operand.

              rm file
                     The rm command removes the file specified by the file operan
d.

              rmdir directory
                     The rmdir command removes the directory entry specified by t
he directory op-
                     erand, provided it is empty.

              symlink source_file target_file
                     See ln.

       -r, --range <range>
              (HTTP/FTP/SFTP/FILE) Retrieve  a  byte  range  (i.e  a  partial  do
cument)  from  a
              HTTP/1.1,  FTP  or SFTP server or a local FILE. Ranges can be speci
fied in a number
              of ways.

              0-499     specifies the first 500 bytes

              500-999   specifies the second 500 bytes

              -500      specifies the last 500 bytes
              9500-     specifies the bytes from offset 9500 and forward

              0-0,-1    specifies the first and last byte only(*)(H)

              500-700,600-799
                        specifies 300 bytes from offset 500(H)

              100-199,500-599
                        specifies two separate 100-byte ranges(*)(H)

       (*) = NOTE that this will cause the server to reply with a multipart respo
nse!

       Only digit characters (0-9) are valid in the 'start' and 'stop' fields of
the 'start-stop'
       range  syntax.  If a non-digit character is given in the range, the serve
r's response will
       be unspecified, depending on the server's configuration.

       You should also be aware that many HTTP/1.1 servers do not have this featu
re  enabled,  so
       that when you attempt to get a range, you'll instead get the whole documen
t.

       FTP  and SFTP range downloads only support the simple 'start-stop' syntax
(optionally with
       one of the numbers omitted). FTP use depends on the extended FTP command S
IZE.

       If this option is used several times, the last one will be used.

       -R, --remote-time
              When used, this will make libcurl attempt to figure out the timesta
mp of the remote
              file, and if that is available make the local file get that same ti
mestamp.

       --random-file <file>
              (SSL)  Specify  the  path name to file containing what will be cons
idered as random
              data. The data is used to seed the random engine for SSL connection
s.  See also the
              --egd-file option.
       --raw  When  used, it disables all internal HTTP decoding of content or tr
ansfer encodings
              and instead makes them passed on unaltered, raw. (Added in 7.16.2)

       --remote-name-all
              This option changes the default action for all given URLs to be dea
lt  with  as  if
              -O, --remote-name were used for each one. So if you want to disabl
e that for a spe-
              cific URL after --remote-name-all has been used, you  must  use  "-
o  -"  or  --no-
              remote-name. (Added in 7.19.0)

       --resolve <host:port:address>
              Provide  a  custom  address  for a specific host and port pair. Usi
ng this, you can
              make the curl requests(s) use a specified address and prevent  th
e  otherwise  nor-
              mally  resolved  address  to  be used. Consider it a sort of /etc/h
osts alternative
              provided on the command line. The port number should be the  numbe
r  used  for  the
              specific  protocol  the host will be used for. It means you need se
veral entries if
              you want to provide address for the same host but different ports.

              This option can be used many times to add many host names to resolv
e.

              (Added in 7.21.3)

       --retry <num>
              If a transient error is returned when curl tries to perform  a  tra
nsfer,  it  will
              retry  this number of times before giving up. Setting the number t
o 0 makes curl do
              no retries (which is the default). Transient error means either: a
timeout, an  FTP
              4xx response code or an HTTP 5xx response code.

              When  curl is about to retry a transfer, it will first wait one sec
ond and then for
              all forthcoming retries it will double the waiting time until it re
aches 10 minutes
```

```
              which  then  will  be the delay between the rest of the retries.  B
y using --retry-
              delay you disable this exponential backoff algorithm. See also --re
try-max-time  to
              limit the total time allowed for retries. (Added in 7.12.3)

              If this option is used multiple times, the last occurrence decide t
he amount.

       --retry-delay <seconds>
              Make  curl  sleep  this amount of time before each retry when a tra
nsfer has failed
              with a transient error (it changes  the  default  backoff  time  al
gorithm  between
              retries).  This  option  is  only interesting if --retry is also us
ed. Setting this
              delay to zero will make curl use the default backoff time.  (Added
in 7.12.3)

              If this option is used multiple times, the last occurrence determin
es the amount.

       --retry-max-time <seconds>
              The retry timer is reset before the first transfer attempt. Retrie
s will be done as
              usual  (see  --retry)  as long as the timer hasn't reached this giv
en limit. Notice
              that if the timer hasn't reached the limit, the request will be mad
e and while per-
              forming,  it  may  take  longer  than  this  given  time  period. T
o limit a single
              request's maximum time, use -m, --max-time.  Set this option to zer
o to not timeout
              retries. (Added in 7.12.3)

              If this option is used multiple times, the last occurrence determin
es the amount.

       -s, --silent
              Silent  or  quiet  mode.  Don't  show progress meter or error messa
ges.  Makes Curl
              mute.

       -S, --show-error
```

```
              When used with -s it makes curl show an error message if it fails.

       --ssl  (FTP, POP3, IMAP, SMTP) Try to use SSL/TLS for the connection.  Rev
erts to  a  non-
              secure  connection  if the server doesn't support SSL/TLS.  See als
o --ftp-ssl-con-
              trol and --ssl-reqd for different levels of encryption required. (A
dded in 7.20.0)

              This option was formerly known as --ftp-ssl (Added in 7.11.0). Tha
t option name can
              still be used but will be removed in a future version.

       --ssl-reqd
              (FTP, POP3, IMAP, SMTP) Require SSL/TLS for the connection.  Termin
ates the connec-
              tion if the server doesn't support SSL/TLS. (Added in 7.20.0)

              This option was formerly known as --ftp-ssl-reqd (added  in  7.1
5.5).  That  option
              name can still be used but will be removed in a future version.

       --ssl-allow-beast
              (SSL)  This  option  tells  curl to not work around a security fla
w in the SSL3 and
              TLS1.0 protocols known as BEAST.  If this option isn't used, the SS
L layer may  use
              work-arounds known to cause interoperability problems with some old
er SSL implemen-
              tations. WARNING: this option loosens the SSL security, and by usin
g this flag  you
              ask for exactly that.  (Added in 7.25.0)

       --socks4 <host[:port]>
              Use  the specified SOCKS4 proxy. If the port number is not specifie
d, it is assumed
              at port 1080. (Added in 7.15.2)

              This option overrides any previous use of -x, --proxy, as they are
mutually  exclu-
              sive.

              Since  7.21.7, this option is superfluous since you can specify a s
ocks4 proxy with
```

```
              -x, --proxy using a socks4:// protocol prefix.

              If this option is used several times, the last one will be used.

       --socks4a <host[:port]>
              Use the specified SOCKS4a proxy. If the port number is not specifie
d, it is assumed
              at port 1080. (Added in 7.18.0)

              This  option overrides any previous use of -x, --proxy, as they ar
e mutually exclu-
              sive.

              Since 7.21.7, this option is superfluous since you can specify a so
cks4a proxy with
              -x, --proxy using a socks4a:// protocol prefix.

              If this option is used several times, the last one will be used.

       --socks5-hostname <host[:port]>
              Use  the  specified  SOCKS5 proxy (and let the proxy resolve the ho
st name). If the
              port number is not specified, it is assumed at port 1080. (Added i
n 7.18.0)

              This option overrides any previous use of -x, --proxy, as they are
mutually  exclu-
              sive.

              Since  7.21.7,  this  option is superfluous since you can specify
a socks5 hostname
              proxy with -x, --proxy using a socks5h:// protocol prefix.

              If this option is used several times, the last one will be used. (T
his  option  was
              previously wrongly documented and used as --socks without the numbe
r appended.)

       --socks5 <host[:port]>
              Use  the  specified  SOCKS5  proxy - but resolve the host name loca
lly. If the port
              number is not specified, it is assumed at port 1080.

              This option overrides any previous use of -x, --proxy, as they are
```

```
mutually  exclu-
              sive.

              Since  7.21.7, this option is superfluous since you can specify a s
ocks5 proxy with
              -x, --proxy using a socks5:// protocol prefix.

              If this option is used several times, the last one will be used. (T
his  option  was
              previously wrongly documented and used as --socks without the numbe
r appended.)

              This option (as well as --socks4) does not work with IPV6, FTPS or
LDAP.

       --socks5-gssapi-service <servicename>
              The default service name for a socks server is rcmd/server-fqdn. Th
is option allows
              you to change it.

              Examples: --socks5 proxy-name --socks5-gssapi-service sockd would u
se  sockd/proxy-
              name   --socks5   proxy-name   --socks5-gssapi-service  sockd/real-
name  would  use
              sockd/real-name for cases where the proxy-name does not match the
principal  name.
              (Added in 7.19.4).

       --socks5-gssapi-nec
              As part of the gssapi negotiation a protection mode is negotiated.
RFC 1961 says in
              section 4.3/4.4 it should be protected, but the NEC reference  impl
ementation  does
              not.  The option --socks5-gssapi-nec allows the unprotected exchang
e of the protec-
              tion mode negotiation. (Added in 7.19.4).

       --stderr <file>
              Redirect all writes to stderr to the specified file instead. If th
e file name is  a
              plain '-', it is instead written to stdout.

              If this option is used several times, the last one will be used.
       -t, --telnet-option <OPT=val>
              Pass options to the telnet protocol. Supported options are:

              TTYPE=<term> Sets the terminal type.

              XDISPLOC=<X display> Sets the X display location.

              NEW_ENV=<var,val> Sets an environment variable.

       -T, --upload-file <file>
              This transfers the specified local file to the remote URL. If ther
e is no file part
              in the specified URL, Curl will append the local file name. NOTE th
at you must  use
              a  trailing  /  on the last directory to really prove to Curl that
there is no file
              name or curl will think that your last directory name is the remot
e  file  name  to
              use. That will most likely cause the upload operation to fail. If t
his is used on a
              HTTP(S) server, the PUT command will be used.

              Use the file name "-" (a single dash) to use stdin instead of a giv
en file.  Alter-
              nately,  the file name "." (a single period) may be specified inste
ad of "-" to use
              stdin in non-blocking mode to allow reading server  output  while
stdin  is  being
              uploaded.

              You  can specify one -T for each URL on the command line. Each -T
+ URL pair speci-
              fies what to upload and to where. curl also supports "globbing" of
the -T argument,
              meaning  that  you  can upload multiple files to a single URL by us
ing the same URL
              globbing style supported in the URL, like this:

              curl -T "{file1,file2}" http://www.uploadtothissite.com

              or even

              curl -T "img[1-1000].png" ftp://ftp.picturemania.com/upload/
       --tcp-nodelay
              Turn on the TCP_NODELAY option. See the curl_easy_setopt(3) man  pa
ge  for  details
              about this option. (Added in 7.11.2)

       --tftp-blksize <value>
              (TFTP)  Set  TFTP  BLKSIZE  option (must be >512). This is the bloc
k size that curl
              will try to use when transferring data to or from a TFTP  server.
By  default  512
              bytes will be used.

              If this option is used several times, the last one will be used.

              (Added in 7.20.0)

       --tlsauthtype <authtype>
              Set  TLS  authentication  type.  Currently, the only supported opti
on is "SRP", for
              TLS-SRP (RFC 5054). If --tlsuser and --tlspassword are specified bu
t  --tlsauthtype
              is not, then this option defaults to "SRP".  (Added in 7.21.4)

       --tlsuser <user>
              Set   username   for   use  with  the  TLS  authentication  metho
d  specified  with
              --tlsauthtype. Requires that --tlspassword also be set.  (Added in
7.21.4)

       --tlspassword <password>
              Set  password  for  use  with  the  TLS  authentication   method
specified   with
              --tlsauthtype. Requires that --tlsuser also be set.  (Added in 7.2
1.4)

       --tr-encoding
              (HTTP)  Request a compressed Transfer-Encoding response using one o
f the algorithms
              libcurl supports, and uncompress the data while receiving it.

              (Added in 7.21.6)

       --trace <file>
              Enables a full trace dump of all incoming and outgoing data, includ
```

```
ing  descriptive
              information,  to the given output file. Use "-" as filename to hav
e the output sent
              to stdout.

              This option overrides previous uses of -v, --verbose or --trace-asc
ii.

              If this option is used several times, the last one will be used.

       --trace-ascii <file>
              Enables a full trace dump of all incoming and outgoing data, includ
ing  descriptive
              information,  to the given output file. Use "-" as filename to hav
e the output sent
              to stdout.

              This is very similar to --trace, but leaves out the hex part  and
only  shows  the
              ASCII  part  of  the dump. It makes smaller output that might be ea
sier to read for
              untrained humans.

              This option overrides previous uses of -v, --verbose or --trace.

              If this option is used several times, the last one will be used.

       --trace-time
              Prepends a time stamp to each trace or verbose line that curl displ
ays.  (Added  in
              7.14.0)

       -u, --user <user:password>
              Specify  the user name and password to use for server authenticatio
n. Overrides -n,
              --netrc and --netrc-optional.

              If you just give the user name (without entering a colon) curl wil
l  prompt  for  a
              password.

              If  you  use  an SSPI-enabled curl binary and do NTLM authenticatio
n, you can force
              curl to pick up the user name and password from your environment b
```

```
y simply specify-
              ing a single colon with this option: "-u :".

              If this option is used several times, the last one will be used.

       -U, --proxy-user <user:password>
              Specify the user name and password to use for proxy authentication.

              If  you  use  an SSPI-enabled curl binary and do NTLM authenticatio
n, you can force
              curl to pick up the user name and password from your environment b
y simply specify-
              ing a single colon with this option: "-U :".

              If this option is used several times, the last one will be used.

       --url <URL>
              Specify a URL to fetch. This option is mostly handy when you want t
o specify URL(s)
              in a config file.

              This option may be used any number of times. To control where this
URL is  written,
              use the -o, --output or the -O, --remote-name options.

       -v, --verbose
              Makes  the  fetching  more  verbose/talkative.  Mostly useful for d
ebugging. A line
              starting with '>' means "header  data"  sent  by  curl,  '<'  mean
s  "header  data"
              received by curl that is hidden in normal cases, and a line startin
g with '*' means
              additional info provided by curl.

              Note that if you only want HTTP headers in the output, -i, --includ
e might  be  the
              option you're looking for.

              If  you  think  this  option  still doesn't give you enough detail
s, consider using
              --trace or --trace-ascii instead.

              This option overrides previous uses of --trace-ascii or --trace.
              Use -s, --silent to make curl quiet.

       -w, --write-out <format>
              Defines what to display on stdout after a completed and successfu
l  operation.  The
              format  is a string that may contain plain text mixed with any numb
er of variables.
              The string can be specified as "string", to get read from  a  parti
cular  file  you
              specify  it  "@filename"  and  to tell curl to read the format fro
m stdin you write
              "@-".

              The variables present in the output format will be substituted by t
he value or text
              that  curl  thinks  fit, as described below. All variables are spec
ified as %{vari-
              able_name} and to output a normal % you just write them as %%.  Yo
u  can  output  a
              newline by using \n, a carriage return with \r and a tab space wit
h \t.

              NOTE:  The  %-symbol is a special symbol in the win32-environment,
where all occur-
              rences of % must be doubled when using this option.

              The variables available at this point are:

              url_effective  The URL that was fetched last. This is  most  meanin
gful  if  you've
                             told curl to follow location: headers.

              filename_effective
                             The ultimate filename that curl writes out to. This
is only meaning-
                             ful if curl is told to write to a file  with  the
--remote-name  or
                             --output  option. It's most useful in combination wi
th the --remote-
                             header-name option. (Added in 7.25.1)

              http_code      The numerical response code that was found  in  th
e  last  retrieved
                             HTTP(S)  or  FTP(s)  transfer. In 7.18.2 the alias r
```

```
esponse_code was
                             added to show the same info.

              http_connect   The numerical code that was found  in  the  last  re
sponse  (from  a
                             proxy) to a curl CONNECT request. (Added in 7.12.4)

              time_total     The total time, in seconds, that the full operation
lasted. The time
                             will be displayed with millisecond resolution.

              time_namelookup
                             The time, in seconds, it took from the start until t
he name  resolv-
                             ing was completed.

              time_connect   The  time,  in seconds, it took from the start unti
l the TCP connect
                             to the remote host (or proxy) was completed.

              time_appconnect
                             The time, in seconds, it took from the start until
the  SSL/SSH/etc
                             connect/handshake  to  the  remote  host  was  compl
eted.  (Added in
                             7.19.0)

              time_pretransfer
                             The time, in seconds, it took from the start until t
he file transfer
                             was just about to begin. This includes all pre-trans
fer commands and
                             negotiations  that  are  specific  to  the  particul
ar   protocol(s)
                             involved.

              time_redirect  The time, in seconds, it took for all redirection st
eps include name
                             lookup, connect, pretransfer and transfer before th
e final  transac-
                             tion  was  started.  time_redirect shows the complet
e execution time
                             for multiple redirections. (Added in 7.12.3)
              time_starttransfer
                             The time, in seconds, it took from the start until
the  first  byte
                             was just about to be transferred. This includes tim
e_pretransfer and
                             also the time the server needed to calculate the res
ult.

              size_download  The total amount of bytes that were downloaded.

              size_upload    The total amount of bytes that were uploaded.

              size_header    The total amount of bytes of the downloaded headers.

              size_request   The total amount of bytes that were sent in the HTT
P request.

              speed_download The average download speed that curl measured for th
e complete down-
                             load. Bytes per second.

              speed_upload   The average upload speed that curl measured for the
complete upload.
                             Bytes per second.

              content_type   The Content-Type of the requested document, if ther
e was any.

              num_connects   Number of new connects  made  in  the  recent  trans
fer.  (Added  in
                             7.12.3)

              num_redirects  Number  of  redirects  that  were followed in the re
quest. (Added in
                             7.12.3)

              redirect_url   When a HTTP request was made without -L to  follow
redirects,  this
                             variable  will  show  the  actual  URL a redirect wo
uld take you to.
                             (Added in 7.18.2)

              ftp_entry_path The initial path libcurl ended up in when logging o
n to  the  remote
```

```
                             FTP server. (Added in 7.15.4)

              ssl_verify_result
                             The  result  of  the  SSL  peer  certificate  verifi
cation  that was
                             requested. 0  means  the  verification  was  success
ful.  (Added  in
                             7.19.0)

       If this option is used several times, the last one will be used.

       -x, --proxy <[protocol://][user:password@]proxyhost[:port]>
              Use the specified HTTP proxy. If the port number is not specified,
it is assumed at
              port 1080.

              This option overrides existing environment variables that set the p
roxy to use.  If
              there's  an  environment variable setting a proxy, you can set prox
y to "" to over-
              ride it.

              All operations that are performed over a HTTP proxy will transparen
tly be converted
              to HTTP. It means that certain protocol specific operations might n
ot be available.
              This is not the case if you can tunnel through the  proxy,  as  on
e  with  the  -p,
              --proxytunnel option.

              User  and  password  that  might be provided in the proxy string ar
e URL decoded by
              libcurl. This allows you to pass in special characters such as @ b
y  using  %40  or
              pass in a colon with %3a.

              The  proxy  host can be specified the exact same way as the proxy e
nvironment vari-
              ables, including the protocol prefix (http://) and the embedded use
r + password.

              From 7.21.7, the proxy string may be specified with a protocol:// p
refix to specify
              alternative  proxy protocols. Use socks4://, socks4a://, socks5://
```

```
or socks5h:// to
              request the specific SOCKS version to be used. No protocol specifie
d,  http://  and
              all others will be treated as HTTP proxies.

              If this option is used several times, the last one will be used.

       -X, --request <command>
              (HTTP)  Specifies  a  custom request method to use when communicati
ng with the HTTP
              server.  The specified request will be used instead of the  metho
d  otherwise  used
              (which  defaults  to GET). Read the HTTP 1.1 specification for deta
ils and explana-
              tions. Common additional HTTP requests include PUT and DELETE,  bu
t  related  tech-
              nologies like WebDAV offers PROPFIND, COPY, MOVE and more.

              (FTP)  Specifies  a custom FTP command to use instead of LIST when
doing file lists
              with FTP.

              If this option is used several times, the last one will be used.


       --xattr
              When saving output to a file, this option tells curl to store certa
in file metadata
              in  extened  file  attributes.  Currently,  the URL is stored in th
e xdg.origin.url
              attribute and, for HTTP, the content type is stored in the mime_typ
e attribute.  If
              the file system does not support extended attributes, a warning is
issued.


       -y, --speed-time <time>
              If  a  download  is  slower  than  speed-limit bytes per second dur
ing a speed-time
              period, the download gets aborted. If speed-time is used, the  defa
ult  speed-limit
              will be 1 unless set with -Y.

              This  option controls transfers and thus will not affect slow conne
```

```
cts etc. If this
              is a concern for you, try the --connect-timeout option.

              If this option is used several times, the last one will be used.

       -Y, --speed-limit <speed>
              If a download is slower than this given speed (in bytes per secon
d)  for  speed-time
              seconds it gets aborted. speed-time is set with -y and is 30 if no
t set.

              If this option is used several times, the last one will be used.

       -z/--time-cond <date expression>|<file>
              (HTTP/FTP)  Request  a  file  that  has been modified later than th
e given time and
              date, or one that has been modified before that time. The <date exp
ression> can  be
              all  sorts of date strings or if it doesn't match any internal one
s, it is taken as
              a filename and tries to get the modification date (mtime) from <fil
e> instead.  See
              the curl_getdate(3) man pages for date expression details.

              Start the date expression with a dash (-) to make it request for a
document that is
              older than the given date/time, default is a document that is newe
r than the speci-
              fied date/time.

              If this option is used several times, the last one will be used.

       -h, --help
              Usage help.

       -M, --manual
              Manual. Display the huge help text.

       -V, --version
              Displays information about curl and the libcurl version it uses.

              The  first  line  includes  the  full  version of curl, libcurl an
d other 3rd party
              libraries linked with the executable.
              The second line (starts with "Protocols:") shows all protocols tha
t libcurl reports
              to support.

              The third line (starts with "Features:") shows specific features li
bcurl reports to
              offer. Available features include:

              IPv6   You can use IPv6 with this.

              krb4   Krb4 for FTP is supported.

              SSL    HTTPS and FTPS are supported.

              libz   Automatic decompression of compressed files over HTTP is sup
ported.

              NTLM   NTLM authentication is supported.

              GSS-Negotiate
                     Negotiate authentication and krb5 for FTP is supported.

              Debug  This curl uses a libcurl built with Debug. This enables mor
e  error-tracking
                     and memory debugging etc. For curl-developers only!

              AsynchDNS
                     This curl uses asynchronous name resolves.

              SPNEGO SPNEGO Negotiate authentication is supported.

              Largefile
                     This curl supports transfers of large files, files larger th
an 2GB.

              IDN    This curl supports IDN - international domain names.

              SSPI   SSPI  is  supported.  If  you  use NTLM and set a blank use
r name, curl will
                     authenticate with your current user and password.

              TLS-SRP
                     SRP (Secure Remote Password) authentication is supported fo
```

```
r TLS.

FILES
       ~/.curlrc
              Default config file, see -K, --config for details.

ENVIRONMENT
       The environment variables can be specified in lower case or upper  case.
The  lower  case
       version has precedence. http_proxy is an exception as it is only availabl
e in lower case.

       Using  an  environment  variable to set the proxy has the same effect as u
sing the --proxy
       option.


       http_proxy [protocol://]<host>[:port]
              Sets the proxy server to use for HTTP.

       HTTPS_PROXY [protocol://]<host>[:port]
              Sets the proxy server to use for HTTPS.

       [url-protocol]_PROXY [protocol://]<host>[:port]
              Sets the proxy server to use for [url-protocol], where the protoco
l is  a  protocol
              that  curl  supports  and  as specified in a URL. FTP, FTPS, POP3,
IMAP, SMTP, LDAP
              etc.

       ALL_PROXY [protocol://]<host>[:port]
              Sets the proxy server to use if no protocol-specific proxy is set.

       NO_PROXY <comma-separated list of hosts>
              list of host names that shouldn't go through any proxy. If set to
a  asterisk  '*'
              only, it matches all hosts.

PROXY PROTOCOL PREFIXES
       Since  curl version 7.21.7, the proxy string may be specified with a proto
col:// prefix to
       specify alternative proxy protocols.

       If no protocol is specified in the proxy string or if the string doesn't m
```

```
atch a supported
       one, the proxy will be treated as a HTTP proxy.

       The supported proxy protocol prefixes are as follows:

       socks4://
              Makes it the equivalent of --socks4

       socks4a://
              Makes it the equivalent of --socks4a

       socks5://
              Makes it the equivalent of --socks5

       socks5h://
              Makes it the equivalent of --socks5-hostname

EXIT CODES
       There are a bunch of different error codes and their corresponding error m
essages that may
       appear during bad conditions. At the time of this writing, the exit codes
are:

       1      Unsupported protocol. This build of curl has no support for this pr
otocol.

       2      Failed to initialize.

       3      URL malformed. The syntax was not correct.

       4      A feature or option that was needed to perform the desired request
was not  enabled
              or  was explicitly disabled at build-time. To make curl able to do
this, you proba-
              bly need another build of libcurl!

       5      Couldn't resolve proxy. The given proxy host could not be resolved.

       6      Couldn't resolve host. The given remote host was not resolved.

       7      Failed to connect to host.

       8      FTP weird server reply. The server sent data curl couldn't parse.
       9      FTP access denied. The server denied login  or  denied  access  to  the  particular
              resource  or  directory  you  wanted  to reach. Most often you tried to change to a
              directory that doesn't exist on the server.

       11     FTP weird PASS reply. Curl couldn't parse the reply sent to the PASS request.

       13     FTP weird PASV reply, Curl couldn't parse the reply sent to the PASV request.

       14     FTP weird 227 format. Curl couldn't parse the 227-line the server sent.

       15     FTP can't get host. Couldn't resolve the host IP we got in the 227-line.

       17     FTP couldn't set binary. Couldn't change transfer method to binary.

       18     Partial file. Only a part of the file was transferred.

       19     FTP couldn't download/access the given file, the RETR (or similar) command failed.

       21     FTP quote error. A quote command returned error from the server.

       22     HTTP page not retrieved. The requested url was not found or returned another  error
              with  the  HTTP error code being 400 or above. This return code only appears if -f,
              --fail is used.

       23     Write error. Curl couldn't write data to a local filesystem or similar.

       25     FTP couldn't STOR file. The server denied the STOR operation, used for FTP  upload-
              ing.

       26     Read error. Various reading problems.

       27     Out of memory. A memory allocation request failed.
       28     Operation  timeout. The specified time-out period was reached accor
ding to the con-
              ditions.

       30     FTP PORT failed. The PORT command failed. Not all FTP servers suppo
rt the PORT com-
              mand, try doing a transfer using PASV instead!

       31     FTP  couldn't  use  REST. The REST command failed. This command is
used for resumed
              FTP transfers.

       33     HTTP range error. The range "command" didn't work.

       34     HTTP post error. Internal post-request generation error.

       35     SSL connect error. The SSL handshaking failed.

       36     FTP bad download resume. Couldn't continue an earlier aborted downl
oad.

       37     FILE couldn't read file. Failed to open the file. Permissions?

       38     LDAP cannot bind. LDAP bind operation failed.

       39     LDAP search failed.

       41     Function not found. A required LDAP function was not found.

       42     Aborted by callback. An application told curl to abort the operatio
n.

       43     Internal error. A function was called with a bad parameter.

       45     Interface error. A specified outgoing interface could not be used.

       47     Too many redirects. When following redirects, curl hit the maximum
amount.

       48     Unknown option specified to libcurl. This indicates that you passe
d a weird  option
              to curl that was passed on to libcurl and rejected. Read up in the
manual!
```

```
       49     Malformed telnet option.

       51     The peer's SSL certificate or SSH MD5 fingerprint was not OK.

       52     The server didn't reply anything, which here is considered an erro
r.

       53     SSL crypto engine not found.

       54     Cannot set SSL crypto engine as default.

       55     Failed sending network data.

       56     Failure in receiving network data.

       58     Problem with the local certificate.

       59     Couldn't use specified SSL cipher.

       60     Peer certificate cannot be authenticated with known CA certificate
s.

       61     Unrecognized transfer encoding.

       62     Invalid LDAP URL.

       63     Maximum file size exceeded.

       64     Requested FTP SSL level failed.

       65     Sending the data requires a rewind that failed.

       66     Failed to initialise SSL Engine.

       67     The user name, password, or similar was not accepted and curl faile
d to log in.

       68     File not found on TFTP server.

       69     Permission problem on TFTP server.

       70     Out of disk space on TFTP server.

       71     Illegal TFTP operation.
       72     Unknown TFTP transfer ID.

       73     File already exists (TFTP).

       74     No such user (TFTP).

       75     Character conversion failed.

       76     Character conversion functions required.

       77     Problem with reading the SSL CA cert (path? access rights?).

       78     The resource referenced in the URL does not exist.

       79     An unspecified error occurred during the SSH session.

       80     Failed to shut down the SSL connection.

       82     Could not load CRL file, missing or wrong format (added in 7.19.0).

       83     Issuer check failed (added in 7.19.0).

       84     The FTP PRET command failed

       85     RTSP: mismatch of CSeq numbers

       86     RTSP: mismatch of Session Identifiers

       87     unable to parse FTP file list

       88     FTP chunk callback reported error

       XX     More  error  codes will appear here in future releases. The existin
g ones are meant
              to never change.

AUTHORS / CONTRIBUTORS
       Daniel Stenberg is the main author, but the whole list of contributors  i
s  found  in  the
       separate THANKS file.

WWW
       http://curl.haxx.se
```

```
FTP
       ftp://ftp.sunet.se/pub/www/utilities/curl/

SEE ALSO
       ftp(1), wget(1)



Curl 7.25.0                                 16 February 201
2                                  curl(1)
```

Release Notes

The following list shows a revision history of the software releases for CueServer Studio.

The current version may be downloaded from the main CueServer Downloads page.

Pre-release and archived versions may be downloaded from the CueServer Software Site.

- Release v5.0.7 [July 25, 2023]
- Release v5.0.6 [June 13, 2023]
- Release v5.0.5 [April 25, 2023]
- Release v5.0.4 [March 29, 2023]
- Release v5.0.3 [March 28, 2023]
- Release v5.0.2 [March 17, 2023]
- Release v5.0.1 [February 6, 2023]
- Release v5.0.0 [January 16, 2023]
- Release v4.0.8 [March 17, 2021]
- Release v4.0.7 [November 20, 2020]
- Release v4.0.6 [October 8, 2020]
- Release v4.0.5 [September 28, 2020]
- Release v4.0.4 [August 19, 2020]
- Release v4.0.3 [July 22, 2020]
- Release v4.0.2 [June 22, 2020]
- Release v4.0.1 [April 21, 2020]
- Release v4.0.0 [March 10, 2020]
- Release v3.1.5 [November 8, 2019]
- Release v3.1.4 [November 6, 2019]
- Release v3.1.3 [September 11, 2019]
- Release v3.1.2 [September 4, 2019]
- Release v3.1.1 [August 1, 2019]
- Release v3.1.0 [June 12, 2019]
- Release v3.0.1 [May 2, 2019]
- Release v3.0.0 [April 4, 2019]
- Release v2.1.2 [January 18, 2019]
- Release v2.1.1 [May 18, 2018]
- Release v2.1.0 [May 16, 2018]
- Release v2.0.4 [March 14, 2018]
- Release v2.0.3 [February 14, 2018]
- Release v2.0.2 [January 22, 2018]
- Release v2.0.1 [November 3, 2017]
- Release v2.0.0 [October 24, 2017]
- Release v1.5.5 [October 28, 2016]
- Release v1.5.4 [September 8, 2016]
- Release v1.5.3 [August 9, 2016]

Release v5.0.7 [July 25, 2023]

Version 5.0.7

Version 5.0.7 is a maintenance release for all CueServer models with 60+ bug fixes and improvements. It is a recommended update for all v5.0.0 thru v5.0.6 users.

- **Stage View**
  - Bug The Effect Filter drop-down menu now properly displays the names of the groups in the list.
  - Bug Fixed the size of playback icons shown in the drop-down menu on Windows only.
  - Bug An appropriate error message is now displayed if no show is active.
  - Bug Addressed an issue that caused the Fader Wheel to adjust all fixture channels instead of only selected channels when clicked.
  - Bug Improved Fader Wheel scaling factor for 16-bit channels.

- **Layout Engine**
  - New Added new *Page Navigation* control.
  - New Added an option to Buttons that provides a way for users to change the Button's Label directly from touchscreens/web pages.
  - New Added an "OR" operator (using double-bars, like `||`) to Button Labels that automatically changes the label of the button based on the indicator state. For example a Label with the contents of "Turn Off||Turn On" will automatically toggle between the two labels based on the indicator state.
  - New Links to objects that are specified by variables are now automatically promoted to `${variable}` syntax and colorized.
  - New Slider objects can now optionally target a Fixture's attributes directly using the link field.
  - New When specifying links to buttons, the field can now prompt the user if the target station has pages or not and visually warn if the target exists or not.
  - New Added *Opacity* property to controls, which has an additional use of being able to *hide* controls during runtime when combined with variables.
  - New If the color field of an object contains a variable, and the color picker is used to change the color, the variable is updated with the new color.
  - New An object targeting a fixture property can now specify a specific instance of that property if the fixture has multiple instances of that property (such as a multi-pixel fixture).
  - New Buttons can now individually have a PIN number assigned to prevent unauthorized access to the Button's function.
  - New Using drag-and-drop to place an image on a layout now automatically creates an Image object.
  - New Variables in fields can now specify default values using `${{{variable|default}}}` syntax.
  - Bug Fixed a problem with flashing indicators that would not show the proper "off color" while flashing.

- Bug Addressed an issue that was preventing SVG images to be used with Image objects.
- Bug Repaired the DIO-588 hardware simulator layout.
- Bug Addressed an issue with resizing Image Buttons to very small sizes would break the button's aspect ratio.
- Bug Fixed a problem that could cause the alignment tools to stop functioning.
- Bug Addressed an issue that could cause a crash of a Button's label consists of only numerical digits.
- Bug Fixed an issue with variables being used as an Image object's width/height properties.
- Bug Repaired the ability to use variables with the targets of Links.
- Bug Improved the behavior of switching shows while the Layout Editor is active.
- Bug Addressed an issue that caused a page in the Layout Editor to change to Edited state even though no changes were made.
- Bug Improve show switching behavior while web stations are visible.
- Bug Fixed an issue that would occur while typing a new value into an image object's aspect ratio field.
- Bug The Image object no longer "wiggles" when manually drag-resizing.

- **CueScript**
  - Bug Fixed a parsing problem that prevented 3+ sequential string concatenations from being handled properly, such as `"A" + "B" + "C"`.
  - Bug Fixed a problem with the parser that wouldn't recognize a variable substitution immediately after a numeric base specifier, such as `CHANNEL 1 AT #`x``.
  - Bug Addressed an issue that caused loops involving macros calling other macros after `WAIT` commands to add a phantom *space* character after each iteration of the command.
  - Bug Repaired a problem with commands not responding to inherited targeted station context.
  - Bug Fixed a problem with the `PROPERTY` command not always addressing the nth property of a fixture when the fixture contains 16-bit channels.
  - Bug Fixed `LOCK`, `UNLOCK`, `ENABLE`, and `DISABLE` to work properly on Shared Control objects.
  - Bug Using the Up/Down arrows in the CueScript console now properly skips blank lines.

- **Rules**
  - New Added a new *Mimic* function for triggers that provides automatic indicator state, but doesn't have any of its own interactive behavior. This provides an opportunity for the programmer to benefit from automatic indicator updates while optionally providing interactive logic using a Trigger's *Additional Rules*.
  - Bug Addressed a problem where rules attached to a particular object weren't inheriting the same default playback fader as the parent object.
  - Bug Fix an issue that would cause conditionals to fail on Shared Controls if the "And This Control" clause was used.
  - Bug If a Rule has an AND clause that is blank, this clause is now ignored instead of always causing the Rule's conditions to fail.

- **Effects Engine**

  - Bug The *Hue Rotate* effect now properly handles fixtures with 16-bit color channels.
  - Bug Fixed an issue with *Twinkle* and *Sparkle* effects that could cause non-color channels to be affected by the running effect.

- **Fade Engine**
  - Bug Improved the behavior of non-Merge mode playbacks to behave logically with LTP channels.
  - Bug Fixed hardwired DMX Output streams that are set to output at less than 40Hz update rate on CueServer 3 hardware to adhere to the requested update rate.

- **DMX Settings**
  - Bug Addressed a crash that could occur if changes are made to the Universe Patch and then the user switches to the Fixture Patch window without first saving changes.

- **Fixture Patch Editor**
  - Bug Fixed a potential crash if the user clicks in the white space below the list of available fixtures.

- **Plugins**
  - New Updated the OSC Plugin to version 1.0.4.
  - Bug Fixed a regression where instances of a plugin that are modified by the user may disappear from view.

- **Show Database**
  - Bug Improved behavior of starting a CueServer 3 without an SD Card inserted.
  - Bug Improved SD Card hot-swapping capability on CueServer 3.
  - Bug When no show is loaded, the embedded web server now displays an appropriate web page.

- **Variables**
  - Bug Astronomical clock system variables are now resolved earlier in the system startup sequence on CueServer 3 to allow them to be used as part of System Power-On and Show Loaded global rules during startup.

- **System Log**
  - New Added the option to turn on logging for Trigger Functions.
  - Bug Improved debug messages for Cue Execution in the Live Playback.

- **Serial Ports**
  - Bug Fixed a problem that could cause the built-in RS-232 port to be unreliable specifically on the CS-3150 model.

Release v5.0.6 [June 13, 2023]

Version 5.0.6

Version 5.0.6 is a maintenance release for all CueServer models with 50+ bug fixes and improvements. It is a recommended update for all v5.0.0 thru v5.0.5 users.

- **Stage View**
  - New Added new CTC fixture control.
  - New Improved touch-usability multiple aspects of the Stage View on touchscreens.
  - New Improved the search function to include channels/fixtures *inside* of groups.
  - Bug Fixed a problem with generic control ranges that would limit the maximum decimal value to 254 in some cases.
  - Bug Addressed an issue that could cause the fader wheel to not properly scroll individual fixture properties.
  - Bug Repaired the Backspace key functionality on Windows in the Record panel.

- **Playbacks View**
  - New A universe's *sACN Transmit Priority* can now be directly changed using a popup menu.
  - Bug Fixed a problem that could cause the display of sACN receive priorities to be out of sync with reality.
  - Bug Removed the grayed-out submaster and associated controls for the Live playback to avoid user confusion.
  - Bug Fixed the ability for the Fader subview to be able to select cues for the Live playback.
  - Bug Removed the improper ability for Presets to select a Next/Previous playback.

- **Cue/Preset Editor**
  - Bug Repaired a Windows-only issue that sometimes prevented the Cue Capture panel from not displaying properly.
  - Bug Addressed a problem on Windows that could cause the Cue Editor to not be drawn properly after switching from a Live view to the editor panel.
  - Bug Fixed an issue that could sometimes display channels that are not included in the Preset's zone.

- **Layout Engine**
  - New Added new CTC mode for the Color Picker.
  - New External layouts now adhere to system-wide authentication requirements.
  - New Improved object alignment tools to either work in single-selection or multi-selection modes.
  - Bug Improved the performance of the SV color pickers on Chrome browsers.
  - Bug If the host CueServer's firmware is updated while a station is being displayed, now properly reload the page to benefit from the newly updated host firmware.
  - Bug Addressed an issue that could prevent Text objects from being resized using the inspector.
  - Bug Fixed an issue that caused some shows upgraded from v4.x to not properly display

previously existing Line objects.
  - Bug Improved the behavior of Image Buttons to not resize when swapping out the image.
  - Bug Fixed the behavior of the align tools to not be able to move locked objects.
  - Bug Pointer events are now rejected on objects without an assigned function.
  - Bug Addressed a problem with the Revert point in the Layout Editor not being updated after each Apply.
  - Bug Fixed a problem with push-hold-record visual button feedback while the button was still being held down.
  - Bug Improve the hit-testing on Icon Buttons.
  - Bug Fix a problem that could occur when unsetting the action for a Touch Area object.

- **Stations**
  - Bug Station pages with PIN numbers assigned now don’t trigger *Is Viewed* events or start *Idle Timers* until the user successfully moved past the PIN entry overlay.
  - Bug Fixed a problem that caused poor performance of station PIN entry screens specifically on iPhone clients.
  - Bug Addressed a problem that causes automatic preset indicators to not always appear correctly if the preset includes fixtures patched with 16-bit channels.
  - Bug Repaired a potential crash if a Button/Contact/Control targets a Group or contains a direct reference to CueScript or a Macro and the Button/Contact/Control is not directly/indirectly a member of any Zone.
  - Bug Addressed an issue that could cause button resources to not load properly when switching shows in certain rare situations.
  - Bug Fix a potential crash if a show has a very large (8,000+) button/contact/control resources.
  - Bug Optimized the performance of loading large numbers of buttons/contacts/controls.
  - Bug Addressed a potential crash if the user disabled *Web Access* to a station while it was being viewed by a client.
  - Bug Address an issue that could cause a crash when switching shows that contain Virtual Stations.
  - Bug Prevent page dimensions from being reset when adding new pages to a station.

- **CueScript**
  - New Added the ability for *CueScript Functions* to accept function parameters.
  - Bug Fixed a regression introduced in v5.0 that could cause a channel that is set to a value and then immediately parked in the same command string (such as `Channel 1 At FL; Park`) to not properly park at the requested level.
  - Bug Repaired the `BREAK` keyword to work properly from within the scope of an IF … THEN statement.
  - Bug Fixed an issue that could cause unintentional strings to be received by the CueScript parser if the *Received CueScript UDP* is enabled on a CueServer 3.
  - Bug Addressed an issue where the current command context’s zone mask would not be updated live when a zone’s configuration is changed while it also active.
  - Bug The `AT` command no longer attempts to adjust the nonexistent submaster level for the Live

playback.

- **Fade Engine**
    - Bug Improved the handling of Streaming Cue playback with fixtures patched with LTP channels.
    - Bug Channels in playback faders in Override mode can no longer be overridden by LTP channels in lower-numbered playback faders.

- **Variables**
    - Bug Fixed a regression that could cause variables with default values to not be properly restored upon show loading.

- **Plugins**
    - Bug Fixed a problem that could cause a boot-time crash if a plugin is used on CueServer 3 that uses sockets.

Release v5.0.5 [April 25, 2023]

Version 5.0.5

Version 5.0.5 is a maintenance release for all CueServer models with 40+ bug fixes and improvements. It is a recommended update for all v5.0.0 thru v5.0.4 users.

- **Stage View**
  - Bug Addressed an issue with effects stored in cues not restoring the proper units of the rate parameter.
  - Bug Restored the usage of the `FL` moniker on channels at 100% value.
  - Bug Added Retina versions of layer icons.

- **Playbacks View**
  - New Added improved *Input Disabled* indication to the *Flowchart View*.
  - New Added display of the current sACN priority reception range for each sACN universe.
  - New Added the precise submaster level value to the *Faders View* placard if not at 100%.
  - Bug Improved the behavior of the *Unlock All Channels* menu option.
  - Bug Changed the color of the notification about channels locked in a playback to match the Stage view.
  - Bug Restored the ability for the *Faders View* to scroll horizontally.
  - Bug Properly display additional output protocols attached to a universe.
  - Bug Repair ability for auto-fill items to be clicked.

- **Layout Engine**
  - Bug The Slider control now properly honors the locked state of the element it is linked to.
  - Bug Fix a problem that could occur when quickly changing views immediately after the initial view loads.
  - Bug Properly resolve variable values used in links.
  - Bug Resolve several issues with the “triangle” and “diamond” style color pickers.
  - Bug Repair a regression that prevented the color wheel from reaching 100% saturation.
  - Bug Prevent the intensity ring around the color wheel from behaving improperly after resizing the window.
  - Bug Properly prevent object actions while editing.

- **Stations**
  - Bug Fixed a regression introduced in v5.0.4 that caused indicator updates on physical stations to lag.

- **Preset Editor**
  - Bug Address an issue where improper channels may be visible in the channel grid after the zone’s channel range is changed to include fewer channels.

Release v5.0.4 [March 29, 2023]

Version 5.0.4

Version 5.0.4 is a maintenance release for all CueServer models with 40+ bug fixes and improvements. It is a recommended update for all v5.0.0 thru v5.0.3 users.

- **Navigator Window**
  - New If a device is discovered that does not have a valid IP address, a prompt is displayed to open the Network Settings dialog to assign an address.
  - Bug Addressed a problem that prevented device autodiscovery on secondary network interfaces if the host computer has multiple active interfaces.
  - Bug Fix a problem that caused the *Check Updates* window on Windows 10/11 to display a message that an unknown application needed to be installed.

- **Stage View**
  - New When using “lasso” selections for channels or fixtures, the entire range of channels/ fixtures is now selected.
  - New No longer hide zero values coming from the Live layer.
  - Bug Fixed a problem introduced in v5.0.2 that prevented channel values from displaying 100%.
  - Bug Vastly improved the loading speed of large fixture patches when using the Safari web browser.
  - Bug Addressed a problem with the RGB color wheel where the wrong color might be displayed in the intensity ring.
  - Bug Fixed an issue with the fader wheel not scaling it’s target values to 16/24/32-bit properties.
  - Bug Repaired the Direct Emitter color control to not disturb other emitter’s values when one emitter is adjusted.
  - Bug Fixed a regression that made it difficult to choose fully saturated colors from the RGB color wheel.
  - Bug Addressed a situation where scrolling the Stage View might not update the channel values scrolled into view.
  - Bug Repaired a problem where the currently displayed channel values might not update when changing active playbacks via CueScript.
  - Bug Improved the behavior of the RGB color wheel when picking a color for a currently released 16-bit color fixture.
  - Bug Corrected an issue with the RGB color wheel not translating colors to CMY fixtures properly.

- **Playbacks View**
  - Bug Fix an issue that could show “undefined” for a Universe’s status.

- **Layout Engine**
  - Bug Improved the ability for touchscreens displaying on remote clients to reconnect to

CueServer after it reboots.

  - Bug Fixed a problem introduced in v5.0.2 that caused the virtual layouts of Mystique and Ultra stations to appear with the wrong colors.
  - Bug Addressed a problem that caused touchscreen pages to not appear properly after dynamically changing shows in certain circumstances.
  - Bug Prevent the “Loading Patch” overlay from appearing on non-Stage views.
  - Bug Improve the reliability of remote virtual stations reloading when the active show is changed.
  - Bug Improved the ability of touchscreen controls linked to properties to properly scale to 16/24/32-bit values.
  - Bug Repair a regression where Playbacks could no longer be linked to a control.
  - Bug Fixed the resolution of Slider controls to use full 8-bit values.
  - Bug Addressed an issue where a touchscreen button that changes pages could change the page before the user’s finger is lifted off the screen, which could trigger the button on the new page.
  - Bug Fixed a problem that was not interpreting a control link in the form of “.“.
  - Bug Fixed a regression introduced in 5.0.2 that broke web station background images.

- **Cue & Preset Editor**
  - Bug Changing the patch while the Cue Editor is open did not update properly.
  - Bug Fix the Live editing mode.

- **Trigger Functions**
  - Bug Fixed a problem with button/contact/control triggers set to execute CueScript or a Macro not adhering to the parent station/page/button’s zone.
  - Bug Addressed an issue with button/contact/control triggers that operate on Groups not adhering to the parent station/page/button’s zone.

- **Rules**
  - Bug Fixed a problem that caused the “Was/Was Not Held” conditionals from being available for Shared Controls.

- **CueScript**
  - Bug Restored the ability to perform a factory reset from the command line.

- **DMX Engine**
  - Bug Fixed a problem that could occur when setting a 16-bit LTP fixture channel to zero, depending on the previous output value of that channel it may have not been set completely to zero.

- **GDTF Parser**
  - Bug Fix interpretation of XYZ and CIE fixture properties.
  - Bug Properly handle fixtures with only one or two of the classic HSV components.

- **System**
    - New Improved the loading speed of shows that have many touchscreen elements during boot time.
    - Bug Addressed a problem that could cause the system to not boot properly if it is loading a very large show file with 1000’s of individual touchscreen elements.
    - Bug Fixed a problem that prevented autodiscovery on a CueServer 3 from functioning if the device did not have any valid IP address.

Release v5.0.3 [March 28, 2023]

Version 5.0.3

Special Note This build has been removed from circulation because a serious bug was discovered related to Shared Controls that use CueScript.

Please update to version 5.0.4 at your earliest convenience.

Release v5.0.2 [March 17, 2023]

Version 5.0.2

Version 5.0.2 is a maintenance release for all CueServer models with more than 40 bug fixes and improvements. It is a recommended update for all v5.0.0 and v5.0.1 users.

- **Live Status**
  - Bug Addressed an issue that could rarely cause the display of the System Log to be empty on CueServer 3 hardware.

- **Stage View**
  - New Added Color Temperature control for fixtures that use CW/WW and/or RGB emitters.
  - Bug Fixed a problem that would occur when a fixture includes emitters other than RGB and all of the fixture's color channels are at zero, where the color preview would display the wrong color.
  - Bug Repaired the *Select* > *From Cue* menu command.
  - Bug Fixed a bug affecting the stream recording time clock display.

- **Cue/Preset Editor**
  - Bug Addressed an issue where the units for effect rates were not being stored properly.
  - Bug Fixed a problem with automatic button indicators that could occur after manually editing preset values.

- **Layout Engine**
  - New Stations can now be nested within other layouts using the Station control.
  - New Added support for GDTF fixtures that use HSB color spaces.
  - New Added support for GDTF fixtures that use xyY and xyz color spaces.
  - New CueScript variables can now be inserted and used in all "inspector" fields.
  - New Implemented several optimizations that improve loading time, especially when the show file contains many groups and/or cues.
  - Bug Addressed an issue that could cause tooltips to be clipped when rendering.
  - Bug Fixed an issue that caused Image Buttons to display the wrong image when the indicator was in a mixed state.
  - Bug Addressed an issue that would cause variables in layouts to not appear correctly if they contained underscore or bracket characters.
  - Bug Fixed a problem that caused touchscreen buttons to not properly inherit the current station and page.
  - Bug Vastly improved the behavior of the HSV Diamond and HSV Circle controls.
  - Bug Attempt to mitigate a scenario where parts of the layout controls might become shifted in the corresponding viewport.
  - Bug Fixed the ability for layout-based popup menus to scroll.
  - Bug Improved performance of backend file loading operations.

  - Bug Addressed an issue with the rendered dimensions of virtual Mystique and Ultra stations.
  - Bug Improve synchronization of indicator flash patterns across multiple touchscreen instances.

- **Stations**
  - New Added automatic indicator feedback when a preset/cue/group/channel/variable has been recorded as a result of the user pressing and holding a button.
  - New Added a new indicator state that represents the color and flash pattern of an indicator while providing record feedback.
  - New Changed the CueStation Address and Hub ID fields to drop-down menus instead of text fields to help make it easier for users to choose proper values.
  - New Added the ability to adjust the number of Button/Contact/Output/Port resources linked to the built-in station.
  - Bug Fix a bug that would sometimes cause fields for Buttons/Contacts/Controls to appear with a seemingly random number (2147483647) when viewed on Apple Silicon Macs.
  - Bug Eliminate an error message that could occur when button events are originated from a station's Page 0.
  - Bug Fixed a crash that could occur when pressing a button linked to a shard control, if that shared control had a "Held" event associated with it.
  - Bug Addressed a problem with Held events for shared controls always being fired even if the button was released before the hold-timeout.
  - Bug Fixed a bug that would allow Adjust/Raise/Lower functions to operate on a control even if that control was locked or disabled.
  - Bug Fixed a problem that caused presets that included all zero values to not properly illuminate button indicators linked to that preset.
  - Bug Addressed an issue with the "Off" state of a preset not always illuminating the proper button indicators.

- **CueScript**
  - New Added new `AT ! n` syntax for indicators that temporarily overrides the color of the target indicator for `TIME` seconds.
  - Bug Repaired an issue that prevented the `AT CUE n` command from working properly with 16-bit channels.
  - Bug The `UNIVERSE` command now properly returns the current enable state of the given universe.

- **DMX Engine**
  - Bug Fixed an issue that could cause sACN input to fail to receive data after a reboot on CueServer 3 hardware.

- **Navigator Window**
  - Bug Improved device auto-discovery in unusual network scenarios like misconfigured switches or improper IGMP settings.
  - Bug Fixed a rare crash that could occur when deleting a CueServer object from the Navigator window.

- **Hardware Settings**
  - New Added the ability for SM-DMX modules to also support Station Bus and RS_485 protocols.
  - New Added support for SM-RS232 modules.
  - Bug Addressed a problem where DMX modules were not fully powering down when set of “Off”.

- **Hardware Support**
  - New Added virtual layout view of CS-3900.
  - Bug Addressed an issue where in rare cases the front LCD display of the CS-3900 would not turn on after reboot.

- **Software Update**
  - Bug Fixed a server-side issue that could cause errors when CueServer Studio starts up.

Release v5.0.1 [February 6, 2023]

Version 5.0.1

Version 5.0.1 is a maintenance release that adds support for the new CS-3900 CueServer 3 Pro hardware, and it also includes several important general bug fixes.

- **Hardware**
  - New Added support for the CS-3900 CueServer 3 Pro.
- **Navigator Window**
  - Bug Fixed an issue that prevented CueServer 3 devices to not be able to be added as remote CueServers.
- **Stage View**
  - New Added new optimizations that increase the performance of the Stage View when many items are visible.
  - Bug Adjusted appearance of gaps and spacers to be more consistent.
  - Bug Updated RGB tools to better reflect a fixtures capabilities.
  - Bug Improved the fixture color approximation to include additional emitter data.
- **Layout Engine**
  - New Various improvements to layout rendering on Insite hardware.
  - New Added the ability for Image Button labels to contain line breaks.
  - Bug Fixed a potential hang on touchscreens when the active show is changed.
  - Bug Updated layout-object function/link field to use interface similar to button resources.
- **Cue Editor**
  - Bug Fixed potential crash while editing cues and presets offline.
- **Presets**
  - Bug Addressed a problem introduced in v5.0.0 that caused the active states of Presets to not properly be reflected in the UI.
  - Bug Fixed automatic preset state resolution when a preset contains patched 16-bit channels.
  - Bug Fixed issue where channels/fixtures could erroneously be masked.
  - Bug The Preset editor now properly adheres to the corresponding zone’s mask.
  - Bug Channel editing outside of an active zone’s channels is no longer possible.
- **Controls**
  - New Buttons/Contacts/Controls can now target a specific zone.
- **CueScript**
  - New Added `device.ip` and related network system variables.

  - Bug The `AT` command can now be used with multiple Controls selected.

- **Variables**
  - Bug Fixed a memory-corruption issue that could occur when adding many variables at once.
  - Bug Repaired non-volatile variable power-on recovery on CueServer 3 hardware.

- **LCD DIsplay**
  - Bug The displayed time on the CueServer 3 Touchscreen now properly updates if the timezone is changed.

- **Web Server**
  - Bug Restored the ability to use active-show related CGI variables in custom web pages on CueServer 3 hardware.

- **JavaScript**
  - Bug Addressed an issue that prevented non-integer results from being returned.

Release v5.0.0 [January 16, 2023]

Version 5.0.0

Version 5.0.0 is a major new release for CueServer that includes many new features. Major features are focused on content creation, fixture personalities, built-in effects, a new fade-engine, improved performance, an upgraded touchscreen engine, and a considerable number of other improvements and bug fixes. CueServer v5.0.0 is available to install on all CueServer 2 and CueServer 3 models.

- **Significant Feature Additions**
    - Feature Create Cues and Presets directly in CueServer.
    - Feature Design custom Layouts to organize fixtures.
    - Feature Add dynamic animation to Cues using the new Effects Engine.
    - Feature Control Channels, Groups, and Fixtures, directly in a newly interactive Stage View.
    - Feature Manage Playbacks, Cue Stacks, and Submasters in a new point-and-click Playbacks View.
    - Feature Enjoy a new Web Station experience rewritten to make touchscreens faster and more responsive.
    - Feature Access Stage, Playback, and Zone screens remotely as fully-functional web pages.
    - Feature Patch moving lights and other multi-parameter fixtures into DMX universes.
    - Feature Assign channels to new HTP, LTP, Snap, Non-Dim modes.
    - Feature Crossfade between scenes in full 16-bit resolution.
    - Feature Use a new Live layer for performing live edits on top of cues.
    - Feature Edit Cue and Preset content in blind or live modes.
    - Feature Modify global settings in real-time.
    - Feature Benefit from new network protocols that significantly increases the performance of CueServer Studio.
    - Feature Park channels globally.
    - Feature Use new Global Controls to consolidate button programming.
    - Feature Point buttons to other buttons to mirror their behavior.
    - Feature Import GDTF based fixture profiles.
    - Feature Support CueServer 3 hardware models.

- **Navigator Window**
    - New The status icon of a CueServer now shows a progress spinner when actively working to connect to the device.
    - New Pressing *Return* or *Enter* now opens the currently selected CueServer, Insite, or show file.
    - New Improvements have been made to the network protocols between Studio and CueServer to make the connection more reliable.
    - New Support for future hardware models has been added.
    - Bug An issue in the autodiscovery network protocol was resolved related to reporting the correct local time of a remote device.

- Bug Addressed a problem that caused remote CueServers that are on local but adjacent subnets to not be reachable when the host computer does not have Internet access.

- **Editor Window**
  - New The lower-left of the window now includes a status bar that indicates if the show file is Active, Not Active or is being Edited Offline. Clicking on this area while Not Active provides a pop-up menu that can activate the show file.
  - New A *Console Toggle* button has been added to the status bar area that allows the CueScript/JavaScript console to be hidden or shown. When hidden, more screen space becomes available for the main editor panel.
  - New The list-based editor panels now have user-selectable layout orientations, Tall and Wide, to accommodate different usage styles and screen sizes.
  - New List-based editor panels now save and restore the user's chosen column widths.
  - New Under the Resources section, added Layouts.
  - New Reorganized the General, Hardware, and DMX settings navigation items.

- **Stage View**
  - New The *Stage View* has been completely rewritten to provide significant new features, and to improve performance.
  - New A new *Layout* mode of the *Stage New* switches to an alternate view that displays user-designed Layouts in place of the regular grid-view.
  - New Channels are grouped together by fixture using a header that displays the fixture number and name.
  - New Patched channels are now displayed with fixture parameters underneath, such as Pan, Tilt, Dimmer, Color, Zoom, Iris, etc.
  - New If a group of one or more channels is used by the fixture to render a color, an approximation of the color is shown above the channel numbers.
  - New The adjacent channels of 16-bit fixture parameters are now grouped together to show high-resolution values.
  - New A new fixture control panel appears whenever the current selection contains one or more fixtures and/or one or more channels patched to fixtures.
  - New The fixture control panel provides a rich user-interface for controlling fixture parameters using easy-to-use sliders and buttons.
  - New Pan/Tilt fixtures invoke a Position panel that intuitively controls a fixture's position in either Cartesian or Polar coordinate planes.
  - New Color fixtures (or pixels) invoke a Color panel that includes mechanisms for choosing colors via RGB, CMY, HSV, Palette, Switch and Direct color modes.
  - New Indexed color/gobo channels allow selection using a virtual color wheel.
  - New Fixtures that use Gobos display the actual Gobo images (if available).
  - New Color macro channels utilize a unique radial color-swatch selector.
  - New Additional custom fixture control panels appear for special channel property types.
  - New Each property in the fixture control panel includes separate Active, Lock, and Disable icon buttons to manipulate the live state of each channel.

- New A fixture control category header appears at the top of the fixture controls that quickly navigates to groups of channels for Position, Color, Beam, Gobo, etc.
- New A new Virtual Wheel control has been added that directly manipulates the selected channels, or the intensity of the selected fixtures by either clicking or dragging within the Wheel.
- New A panel at the top of the Wheel control displays the target of the current selection.
- New Direct buttons at the top and bottom of the Wheel control immediately set the selected channels or fixture intensities to either Full or Off.
- New A button at the bottom of the Wheel control area allows the currently selected channels or fixtures to become Released.
- New The Wheel control will switch to fine adjustment mode by holding the Option/Alt key when scrolling it.
- New Double-clicking on a channel value allows the user to directly edit the value using text input.
- New A View selector appears at the top of the Stage View that switches between Input, Playback, Effects, and Output display modes.
- New The View Settings menu allows selection of a user preference for displaying channel values in Percentage, Decimal, Hexadecimal, and a new Bar Graph display.
- New The View Settings menu has options for showing/hiding the Fixture Controls or the Wheel Control, displaying Local/Global channel numbers, drawing the fixture’s color approximation, hiding unpatched channels, and toggling fixture spacers and/or one-fixture-per-line features.
- New The Active Layer popup menu switches the active layer between the currently available playbacks or the Live layer.
- New The Select popup menu in the Modify Fixtures section provides a host of options for selecting fixtures or channels, including by Group, by built-in patterns, by random selection, current selection modification, and more.
- New The Edit popup menu in the Modify Fixtures section provides menu items for Cut/Copy/Paste functions for manipulating channels, plus Release, Clear, Park, and Assert functions.
- New A button is available in the Modify Fixtures section that directly activates the Effects Engine control panel.
- New A new Effects panel has been added that allows multiple effects to be added to a Playback along with choosing the type and parameters of each effect.
- New A popup menu in the Record & Playback section allows the user to choose the current cue stack ard/or preset zone for the active layer.
- New The Record/Update combination control in the Record & Playback section provides a direct method for recording new cues/presets, or updating existing cues/presets.
- New The Go combination control in the Record & Playback section provides a direct “Go” button, and/or a popup menu for directly selecting a cue to execute.
- New A Search function is available in the *Stage View* that uses CueScript-like language to select channels or fixtures.
- New A contextual menu will appear when right-clicking on a channel or fixture that provides quick shortcuts to common channel/fixture operations.
- New Clicking on the header of a universe header in the grid view will collapse/expand the universe.

  - New A dock control allows the fixture controls to be moved to the left, right, top, or bottom of the view.

- **Playbacks View**
  - New The *Playbacks View* has been completely rewritten to provide significant new features, and to improve performance.
  - New Two Playbacks views are now available, including the classic *Flowchart* view that displays the flow of channel values from Input, through the Playbacks, and to the Output, and a new *Faders* view that organizes each Playback as a large slider with status and direct-action buttons.
  - New The Flowchart Input block now includes a popup menu that provides a method for Enabling/Disabling the input layer.
  - New Clicking in any of the Playback Blocks' *Current Cue* or *Next Cue* columns displays a context field that allows the user to directly execute a cue/preset or select the next cue/preset.
  - New Hovering over any of the Playback Blocks' *Options* column displays the Playback's submaster slider.
  - New The *Current Cue* panel now additionally displays presets and zones that are active in the playback fader.
  - New The *Current Cue* panel also displays any effects that are running in that playback fader.
  - New Each Playback Block now includes direct access buttons for Go, Stop/Start, and Clear.
  - New Each Playback Block now includes a popup menu control that displays menu options for a wide variety of common Playback commands, such as executing cues, switching active playbacks, locking channels, enabling/disabling the Playback, and changing Playback combine modes.
  - New The Output Block has context menus for each output universe that allows that universe to be enabled or disabled.
  - New If a Playback is Disabled, the corresponding Playback Block will appear in a Dimmed state.
  - New If a Playback is Stopped, the corresponding Playback Block will appear with a Red background.
  - New The Fader view provides an array of submaster controllers that include individual sliders, with direct access buttons for Go, Stop/Start, Full/Off, and Clear, and dynamic fields for the current cue, next cue, and fader options.
  - New Clicking a fader number in the Fader view changes the current playback fader.
  - New Each fader in the Fader view includes miniature versions of the *Current Cue*, *Next Cue*, and *Fader Options* panels as visible in the Flowchart view.
  - New A View Settings drop down menu allows changing of the *active* playback fader.
  - New The Select popup menu in the Modify Playback section provides a several options for selecting playbacks.
  - New The Edit popup menu in the Modify Playback section provides a host of options for modifying the selected playback(s) including performing a Go, setting the Next Cue, changing Cue Stacks, performing a Release, Clear, or Assert, locking channels, disabling/enabling, stopping/starting, and changing playback combine modes.

- **Zones View**
  - New The *Zones View* has been completely rewritten to provide new features, and to improve performance.
  - New The Join grid is now interactive, allowing point-and-click joining of zones.

- **Front-Panel Status**
  - New The Port indicators are now dynamically named and the proper quantity appear depending on the physical CueServer model.
  - Bug The Power LED indicator now properly shows the Blue/Magenta blinking state of the LED if an important log message is pending.

- **Timecode Status**
  - Bug Fixed an issue that caused the Timecode Status panel to repeatedly send CueScript commands to the device.

- **CPU Info Panel**
  - New The device’s CPU Temperature is shown if the device has the ability to measure this parameter.
  - New A new Additional Information section has been added that displays pertinent version numbers, model IDs, revisions, and operational parameters of the device.
  - New Support for new processes statuses have been added for hardware that uses different processes.

- **Cue/Preset Editor**
  - New The *Contents* panel of the Cue/Preset Editor now features the ability to edit cues directly using the same functionality as the *Stage View*, complete with fixture controls, virtual wheel, effects, and point-and-click editing.
  - New A new “Live” toggle switch allows the currently edited cue/preset to be switched to Live mode, sending the channels directly to the *Live* layer so the cue’s contents can be seen on the physical lighting instruments connected to CueServer’s output.
  - New The orientation of the Cue/Preset List can be changed from Portrait to Landscape modes to better facilitate user’s screen sizes and using the new *Contents* editor.
  - New Added the ability to Copy/Paste cue contents.
  - Bug The *Timing* column of the Cue Editor now shows a streaming cue’s follow time, if present.
  - Bug Columns in the Cue List now properly display ellipses when their contents would be truncated.

- **Group Editor**
  - New Groups can be switched between Channel or Fixture mode.
  - New A Group Wizard has been added that automates building groups by allowing the user to specify ranges, patterns, and boolean operations to perform on the group.
  - New The contents column now displays wether a group is specifying channel or fixture numbers.

- **Layouts Editor**
    - New Leveraging CueServer's unique web-based Touchscreen Engine, *Layouts* have been added that allows the user to design their own custom fixture layouts for use in the Stage View and Cue Editor. Similar to Magic Sheets in other controllers, *Layouts* takes this concept to a whole new level by also allowing the entire suite of touchscreen controls (such as Buttons, Sliders, Color Pickers, etc.) to be added allowing the designer to create their own screens that visualize their project in real time while also accelerating their productivity and providing unique control solutions for their clients.

- **Variables Editor**
    - Bug Addressed a problem when creating empty non-volatile variables that could cause the empty value to not be stored in the non-volatile database.
    - Bug Fixed a possible crash that could occur when exiting the Variables Editor panel.
    - Bug Addresses a problem that could cause unexpected characters to appear in the values of non-volatile variables in the *Variables* panel.
    - Bug Improved the reliability of changing a variable's scope between volatile and non-volatile.

- **Stations**
    - New Station layouts use the new shared Touchscreen Engine, bringing many new features, bug fixes, and performance improvements to web-based stations.
    - New Built-in fully interactive Layouts are now available for each physical CueServer type with working displays, buttons, and LED indicators.
    - New The new *Fixture* object type can be added to touchscreen layouts.
    - New Changing a station's *Web Access* setting or password now immediately logs-out or provides access to connected users.
    - New Added "Page Is Activated", "Page is Deactivated", "Page is Viewed" and "Page is Hidden" event options to pages.
    - New Added option to extend the background color of a station's page to fill the space created by certain scale modes.
    - Bug Addressed a problem when editing the properties of a station configured as a DIO-588 not properly showing the correct Address or Hub ID.
    - Bug Fixed multiple bugs that could cause a crash with the LED Indictor color chooser, including a crash if the main window is closed while the custom color picker is visible, or if the chooser window is opened/closed multiple times.
    - Bug Repaired an issue that could cause a station's *Web Access* setting to not function properly after a show file is unpacked and repacked.
    - Bug Escape key no longer dismisses current changes in layout editor.

- **Controls**
    - New Added a new global trigger object called a *Control*. This trigger type acts like a station's button, but is not attached to any particular station. These global control objects are useful for adding automation to Stage Layouts or for use with Button Pointers.

- **Timers**

  - Bug Addressed a problem where Hourly Timers would not execute on the specified schedule in certain circumstances.
  - (BUG) Deleting a Timer from the list no longer possibly executes by being left in the Timer cache in the device.

- **Timecode Triggers**
  - Bug Fixed a problem on Windows that caused the text insertion cursor to jump to unexpected positions while editing timecode.

- **General Settings**
  - New Improved settings organization by moving *LCD Display* and *Location* into *General Settings*.
  - Bug Addressed a problem that caused changes to General Settings to be unexpectedly committed when clicking on the current setting category in the navigation list.
  - Bug Fixed an issue that caused the current General Setting to become deselected when clicking the Revert button.
  - Bug Opening the Timecode settings panel no longer resets the Skip Seconds parameter.

- **Hardware Settings**
  - {NEW) Entirely new *Hardware Settings* panels that makes it easier to select the preferred hardware type, and to change specific hardware-related settings such as audio volume, LCD display preferences, and the configuration of DMX Ports and/or Modules.
  - New A new panel for editing the DMX Port/Module configuration allows ports to be named, the type of port to be selected, and that port's parameters to be configured.
  - New Added support for upcoming hardware models with new features.

- **DMX Settings**
  - New Added a new *Fixture Patch* panel that allows the user to patch fixtures.
  - New Adding a new fixture presents a fixture chooser dialog that lists fixtures by manufacturer, and then by model, and then by fixture variant.
  - New The fixture chooser includes user-defined parameters for starting fixture number, starting channel, number of fixtures to add, and an optional fitting/spacing parameter.
  - New The fixture chooser provides fixture thumbnail images and general description.
  - New The fixture chooser includes a simple portal for opening the GDTF Share fixture library web site, and facilitating installation of new GDTF files into the user's local *Fixture Definitions* folder.
  - New The *Fixture Patch* panel displays the patch by Channel, Fixture, or by Fixture Type.
  - New If a fixture definition already imported into a show file becomes outdated, the *Fixture Patch* panel allows the user to update the fixture definition.
  - New If a fixture definition cannot be parsed properly by the GDTF Distiller, the *Fixture Patch* panel displays a log of import errors.
  - New Additional properties have been added to the Playback Settings panel including the enable and stopped state, the default Cue Stack, and an option to disable the LTP mode of the playback.

  - New Any changes made to the properties of a Playback Fader take effect immediately. Conversely, if properties are changed via other external methods (such as CueScript or by using the Stage or Playbacks views) an option appears to save the live setting as the new default setting.
  - Bug Addressed a problem that caused changes to the combine mode of Playback 2 or greater to not be saved.

- **Plugins**
  - Bug Fixed an issue where the application could crash if the user tries to change plugin properties after deselecting a plugin.

- **Touchscreen Engine**
  - New The *Touchscreen Engine* has been completely rewritten to take advantage of the latest web technologies, improving performance, optimizing network traffic, and decreasing CPU demands.
  - New The *Touchscreen Engine* is now shared between web-based touchscreens (including Insite) and the Stage View and Cue Editor Layouts.
  - New Most layout objects can now be arbitrarily *linked* to virtually any resource in the system, including Buttons, Channels, Contacts, Controls, Fixtures, Groups, Outputs, Submasters, or Variables.
  - New Added new *Alignment Snapping* function when dragging objects.
  - New Added *Distribute Horizontally* and *Distribute Vertically* functions.
  - New Added full support for Undo and Redo while designing layouts.
  - New Objects in layouts can now be rotated at arbitrary angles.
  - New Added *Fixture* as a new primitive object with options for displaying as either a moving light or pixel fixture.
  - New Line objects can now have custom endcaps such as arrows, bars, and circles.
  - New Line objects with endcaps now have the option to offset the endcap to extend beyond the end of the line.
  - New Line objects can be converted into Curves with options for Quadratic or Cubic Bézier types.
  - New Image objects can now specify custom images for User 1..4 indicator states.
  - New Text fields in the inspector panel now have a click-and-drag control on their left-side that can be used to quickly adjust the value in the field.
  - New Sections of the inspector panel can be collapsed using the corresponding disclosure triangle.
  - New The Touch Area control now supports Swipe Up/Down/Right/Left gestures.
  - New Arrays of Fixtures can be added to layouts by dragging a new fixture object onto a primitive such as a Rectangle, Ellipse, Line or Curve.
  - New When adding a Fixture Array to a Rectangle, an option appears allowing the rectangle to be filled as a Matrix or the perimeter being traced as an Outline.
  - New A Layout preference was added to display Tooltips when hovering over objects in the layout, displaying the object’s ID, Link, or Type.

  - New The *Layout Inspector* can be reassigned to either the right or left side of the edit canvas and it's width can be adjusted.
  - New The grid's opacity and color can now be changed as a user preference.
  - New The grid's color can be set to Auto, allowing it to automatically shift to a light color on dark backgrounds and a dark color on light backgrounds.
  - New Editor Options have been added to individually enable/disable the Lasso, Dragging, Resizing, and Rotating functions of the editor.
  - New Added `${value|default}` syntax to touchscreen variable substitutions.

- **Button/Control Editor**
  - New Buttons, Contacts and Controls may now simultaneously define both an Action as well as one or more additional Rules.
  - New Added "Point to Button", "Point to Contact" and "Point to Control" functions to allow one Button/Contact/Control to be defined by the behavior of another.
  - New Choosing the CueScript action in a Button now uses the rich script editor popup window.
  - Bug Fixed a bug that caused the "THEN Next Cue" rule to fail in certain circumstances.

- **Global Rules**
  - Bug Addressed a problem that prevented Global Rules from executing properly for Buttons and Contacts.
  - Bug Fixed an issue that could cause global rules related to the sun's position from not firing properly if attached to the System Startup condition.

- **CueScript**
  - New The `FIXTURE` (or `F`) command has been added to allow fixture selections by fixture number or fixture.property.
  - New A `PROPERTY` (or `PROP`) command was added to allow individual properties of fixtures by channel offset or property name.
  - New The `LIVE` (or `L`) command was added to set the active playback layer to "Live". Alternatively, the Live layer can be accessed as `PLAYBACK 0`.
  - New Added the `EFFECT` (or `EF`) keyword to select one of the active playback fader's effect slots.
  - New The `CONTROL` (or `K`) was added to select Shared Controls.
  - New Added the `ASSERT` (or `AS`) command that "asserts" LTP channels in the current playback fader.
  - New A new syntax `PLAYBACK SET n` has been added that changes the *active* playback without changing the current *selection*.
  - New Selections commands such as Button, Channel, Contact, Fixture, Group, Indicator, Output, and Universe can now be given the keyword `CLEAR` as their parameter to remove the associated selection.
  - New Added optional `LIVE` source keyword to the Record/Update Cue/Group/Preset commands.
  - New Added the `!` option to the `AT` command that causes the setting of channel values to temporarily ignore fade times.

  - New The `AT` command now seamlessly handles 8-bit and 16-bit channel values.
  - New The `LOG` command can now take either a `!` or `~` to elevate the severity of the logged message to “error” or “warning”, respectively.
  - New The `PARK` / `UNPARK` commands now globally park/unpark channels in the Output stage of the Fade Engine. This is a change from previous behavior where parked channels would occur in individual playback layers.
  - New The `LOCK` / `UNLOCK` commands are now used to freeze/unfreeze a channel value in an individual playback fader. This is the previous behavior that the park command used to exhibit.
  - New Added `PLAYBACK NEXT` and `PLAYBACK PREV` to increment/decrement the active playback to the next/previous playback. These command can use the optional `SET` keyword to indicate that the current selection should not be changed.
  - New Selection keywords such as `CHANNEL`, `FIXTURE`, `GROUP`, `BUTTON`, `UNIVERSE`, etc., can now use the semicolon `;` token instead of a specific object number to change the current selection to the specified target but with no objects selected.
  - New Added Effects to the object types that `ENABLE` / `DISABLE` / `CLEAR` operates on.
  - New The `FADE`, `FOLLOW`, `LINK` and `NAME` keywords can be used in the `RECORD` and `UPDATE` commands to optionally provide or modify those parameters when recording or updating a cue/preset.
  - New The `ZONE` command can now accept an empty string (“”) to switch to *no zone*.
  - Bug The `PARK` command no longer attempts to operate on non-DMX targets or non-existent channels.
  - Bug Improved the language validator to properly recognize negative numbers as parameters to the `SET` and `=` commands.
  - Bug Fixed a parsing issue that could cause CueScript to fail if a comment is encountered inside of an IF/THEN block.
  - Bug Addressed an issue that would allow invalid playback numbers to become selected.
  - Bug Parsing of literal numbers no longer clear the current command target.
  - Bug Fix a problem with using the `TOGGLE` command with Presets.
  - Bug Fixed CueScript issue with targeting multiple playbacks with the Release command.
  - Bug Fixed an issue with the `PLAYBACK NEXT` command.
  - Bug Restore the ability to use `BUTTON x.y.z` style syntax to simultaneously refer to station/page/button.

- **Fade Engine**
  - New Fixture patching is now supported, giving the Fade Engine the ability to “understand” the function of each channel in the context of what fixture attributes it is controlling.
  - New An *Effects Engine* has been added that can be applied to the output of each playback fader. The static, fading or streaming channels in each fader pass through a dynamic effects processing layer before those channel values pass down to the next fader or CueServer’s output, creating dynamic animations that operate on top of base channel values from any cue or preset.
  - New Added an special playback fader called “Live” that is always in the playback stack after the last regular fader. This new fader is used to create content without interfering with any of the

regular faders, and is used by the Cue Editor when switching into Live mode. Channel values in Live always overrides the other playbacks and this fader does not have a submaster.

  - New LTP (Latest Takes Precedence) channel priority mapping now occurs on non-Dim channels between playback faders.
  - New Crossfades now occur in either 8-bit or 16-bit domains depending on what fixture channels are patched.
  - New Channels may be declared as “Snap” channels by the fixture patch, causing those channels to ignore crossfade times between cues.
  - New Non-Dim channels are no longer scaled by a playback fader’s submaster.
  - New A new *Crossfade* fader mode was added. This mode provides the functionality of a “dipless crossfade” between the channel values in the fader and the channel values from the previous fader. The fader’s submaster is used to fade between the two scenes.
  - New Parked channels now reside in the Output layer, they override any other source of values for those channels, and cannot be changed by any operation other than specifically “unparking” those channels.
  - New If a playback fader is stopped while a fade and/or follow is running, the fade/follow progress is paused until the fader is restarted.
  - New Channels may be locked in any playback fader, causing their values that playback fader from changing when executing cues or setting channel values. Until those channels are unlocked, or the playback fader is fully reset, those channels will not change.
  - New Channels locked in individual playback faders are no longer cleared with the `CLEAR` command.
  - New Channels locked in individual playback faders now unaffected by the playback’s submaster.
  - {NEW) If the *LTP Disable* option is chosen for a playback fader, then any LTP channels in that playback will instead be handled in the classic HTP Combine manner.
  - Bug Fixed an issue where a loss of DMX input could inappropriately trigger a DMX Input Trigger in certain circumstances.
  - Bug Addressed an issue where looping a zero-length streaming cue would hog all available CPU resources.
  - Bug Fix a problem that could cause Streaming Cue playback to be frozen on the wrong channels in certain circumstances.

- **Effects Engine**
  - New The *Effects Engine* provides the ability to add animations, filters, and generators to the output of individual playback faders. This allows previously static cues or presets to employ dynamic content.
  - {NEW) Each effect can be optionally filtered to only include certain channels by choosing a Group.
  - New Up to 4 effects can be added to each Playback Fader.
  - New Each effect can be enabled/disabled.
  - New Effect configuration is stored into Cues/Presets and is recalled when the Cue/Preset is executed or activated.
  - New A *Hue Rotate* effect was added that animates the “Hue Angle” of any color channels in the

active fixtures. Parameters to *Hue Rotate* include Rate and Direction.
    - New A *Sparkle* effect was added that animates color channels in active fixtures in a “sparkling” effect. As each pixel sparkles, it fades from its current color to White, and then back to its original color. Parameters to *Sparkle* include Rate, Intensity, Attack, and Decay.
    - New A *Twinkle* effect was added that animated color channels in active fixtures in a “twinkling” effect. As each pixel twinkles, it fades from its current color to Black, and then back to its original color. Parameters to *Twinkle* include Rate, Intensity, Attack, and Decay.
    - Special Note More effects are planned and will be included in future releases.

- **Show Database**
    - New Changed the default number of Playbacks in a new show file to 4.
    - Bug Addressed an issue where the system log may report “Could not open stream file” when switching shows.

- **GDTF Distiller**
    - New Core to the new fixture patching mechanism in CueServer is a native understanding of GDTF (General Device Type Format) fixture definition libraries. The GDTF file format is an open standards technology designed to facilitate interoperability and data exchange of fixture specifications between manufacturers and their designer base. For more information, please visit https://gdtf-share.com.
    - New When fixtures are patched, an optimized version of the necessary GDTF data is imported into the show file, including information and media related to the fixtures’ DMX channels, functions, properties, attributes, colors, emitters, gobos, geometries, wheels, and more.

- **System Variables**
    - New Added `Playback.Name` and `Playback.LTP`.
    - New Added `Effect` class of variables, including `Angle`, `Attack`, `BPM`, `Decay`, `Group`, `Intensity`, `Period`, `Rate`, `Reverse`, and `Type`.
    - New Added `Show` class of variables, including `Channels`, `Name`, `Path`, `Playbacks`, `Ports`, `Universes`.

- **Network Settings**
    - Bug Addressed a problem where the fields for LAN B would not be auto-populated correctly when changing from Single-LAN to Dual-LAN mode.

- **Time Settings**
    - Bug After changing the device’s timezone, a problem was addressed that could leave certain parts of the system reporting time from the previous timezone until a reboot.

- **MacOS Build**
    - New *CueServer Studio* is now a Universal Binary, providing native M1 support for Apple Silicon.
    - New *CueServer Studio* is now signed by our developer certificate.
    - New *CueServer Studio* will now launch and/or activate when the `studio://` URI is requested.
    - Bug Addressed issues for compatibility with macOS *Big Sur* and *Monterrey*.

- **Windows Build**
  - New Updated to the latest framework libraries for improved Windows compatibility.

- **App Updater**
  - New App update checker now uses Beta track for users currently using Beta software.

Release v4.0.8 [March 17, 2021]

Version 4.0.8

Version 4.0.8 is a maintenance release with 5 bug fixes and one minor new feature. These changes improve the overall reliability and performance of the product. The fixes for the `.cs2` file format and the web station framework are important updates for all CueServer users.

- **Navigator Window**
  - **Bug** Addressed a problem that could cause `.cs2` show file uploads to the device to fail if the show contains a large number of individual resources.
- **CueScript**
  - **Bug** Fixed a problem that caused the `RELEASE` command to improperly release all channels if the selection contained only channels above 4096.
  - **Bug** Addressed a potential crash that could occur if a show file is opened that has the CueScript Console server enabled.
- **Stations**
  - **Bug** Addressed a bug in the web station framework that caused performance problems with switching pages and that would ultimately cause the host browser to run out of memory.
- **JavaScript**
  - **Bug** Fixed problem with `getChannelLevel()` alias.
- **General**
  - **Feature** Added an internal function that can display important bulletins as available from the software update server.

Release v4.0.7 [November 20, 2020]

Version 4.0.7

Version 4.0.7 is a maintenance release with 3 bug fixes and one minor new feature. These changes improve the reliability of the product and introduce the ability to enable/disable the recently added IGMPv2 Querier feature.

- **Navigator Window**
  - **Bug** Fix a Windows-only crash that could occur in the Password Entry window when attempting to reveal the list of shows stored on a password protected CueServer.

- **Network Settings**
  - **Feature** A new *Advanced…* option is available in the *Network Settings* window that reveals a new option to enable/disable CueServer's built-in IGMPv2 Querier.

- **Timers**
  - **Bug** Addressed a problem that could cause timers set to Hourly Schedule to prematurely execute earlier than scheduled.

- **Timer Editor**
  - **Bug** Fixed a potential crash that may occur when closing the Hourly Schedule popup window.

Release v4.0.6 [October 8, 2020]

Version 4.0.6

Version 4.0.6 is a maintenance release with 6 bug fixes. Most importantly, an issue has been addressed that improves network reliability on the CS-900 model when it is configured for Dual-LAN mode.

- **CueServer Studio**
    - Bug After the application is open and running, if a `.cs2` file is opened from the Finder, it now is properly added to the Offline Shows list and the Editor Window opens.
    - Bug Repaired an issue that could cause CueServer Studio application version checking to report erroneous results when launched in 32-bit Windows.

- **Network Settings**
    - Bug Addressed an issue that could sometimes cause a yellow caution icon to appear in the lower-left corner of the *Network Settings* window, even if no warning condition is currently active.

- **Editor Window**
    - Bug Fixed an issue that caused the window splitter between the main editor panel and the command line to not remember its position properly after the window is resized.

- **Timers**
    - Bug Fixed a problem that caused On/Off timers to not fire during system startup if the timer was within range, but a global rule was also executing during startup.

- **Ethernet Connectivity**
    - Bug Addressed a problem that was causing the CS-900 model when in Dual-LAN mode to drop Ethernet packets if the packets had their DSCP “QoS” bits set. This would disrupt communications with services such as DHCP, SSH, or NTP if those services were specifying non-zero QoS priority.

Release v4.0.5 [September 28, 2020]

Version 4.0.5

Version 4.0.5 is a maintenance release with 7 bug fixes and 1 new feature. The improvements in this release focus on improved network behavior when IGMP Snooping is enabled, and a variety of minor bug fixes.

- **Rules**
  - Bug Fixed a problem that would cause a contact’s rules to stop executing properly if a contact closure input had a “was held” rule and that contact was closed/opened 16 times in succession.

- **Web Stations**
  - Bug Addressed a problem that could cause the Layout Editor library to not show its contents if the same CueServer appeared twice in the Navigator window, once as a local device, and again as a remote device.
  - Bug Fixed a bug that caused buttons with Button IDs above 64 to not work with the `AT`, `ENABLE`, `DISABLE`, `LOCK`, or `UNLOCK` commands.

- **Settings**
  - Bug Adjusted the UI of the *CueScript Settings* panel to make it clear that Incoming CueScript UDP messages are always received using UDP Port 52737.

- **Web API**
  - Bug Repaired problem in the `set.cgi` API for setting time remotely on a CueServer that could cause the API call to fail.

- **JavaScript**
  - Bug Improved the CURL command to be able to receive larger responses from clients, allowing the Philips Hue plugin to address many more lights simultaneously.

- **Networking**
  - Feature CueServer now sends *IGMP General Query* messages periodically to the network to help ensure that any switches configured to perform *IGMP Snooping* will receive the IGMP Membership Reports necessary to maintain CueServer auto-discovery and sACN multicast traffic as appropriate.

- **Operating System**
  - Bug Fixed a problem that would occur when setting the current date/time of a CueServer if the new date has a different Daylight Saving Time status than the old date. The resulting time would be offset by the difference in the DST change instead of setting the time as expected.

Release v4.0.4 [August 19, 2020]

Version 4.0.4

Version 4.0.4 is a maintenance release with 22 bug fixes and 6 new features. The changes in this release focus on a variety of issues including offline show editing performance, Windows compatibility, auto-discovery fixes, Art-Net protocol improvements, and addressing of several potential crashes.

- **Navigator Window**
    - Feature If it can be determined that a remote CueServer has become permanently unavailable, it is automatically removed from the device list.
    - Bug Addressed a problem introduced recently that caused auto-discovery to only query the primary network interface.
    - Bug Fixed a potential crash that could occur if the host computer does not have a connection to the Internet.
    - Bug Corrected a problem that caused unnecessary network traffic to Insite touchscreen panels.
    - Bug The *Open…* file menu now correctly allows a .cs2 file to be opened.
- **Stations**
    - Bug Fixed a problem that would prevent custom viewport size or scale mode from being saved when first creating a new custom touchscreen station.
    - Bug Corrected a previous fix that caused deleting touchscreen objects to sometimes remove the wrong object.
    - Bug Addressed a problem that could sometimes display stale or blank information in the Station Editor’s navigation panel on Mac.
- **Variables**
    - Bug Fix an issue that might cause the Apply button to appear when first opening the Variable Editor (before making any changes).
    - Bug Improved the reliability of large changes to non-volatile variables being stored correctly in the hardware backing store.
    - Bug Editing but not changing a variable’s name would remove that variable’s value. This has been fixed.
    - Bug Addressed an issue that could make a variable’s value as displayed in the Variable Editor “bounce” briefly to its old value before showing its newly changed value.
- **Settings**
    - Feature Added an Input Source menu to the Timecode Settings window.
    - Bug Fixed a problem with the Art-Net protocol editor that would not properly display the current broadcast mode in the popup menu.
    - Bug The Art-Net protocol now uses Network numbers from 0 to 7F (hexadecimal) instead of 1 through 128 as specified by the most recent documentation from Artistic License.

- **Directory Editor**
  - Bug Fixed a recently introduced problem that prevented files to be added to either the Sounds or Web directories when editing offline shows.

- **JavaScript**
  - Bug The `newResource()` API now properly returns a new resourceID when it was unspecified or `-1`.
  - Feature A new mechanism has been added to make it easier to determine the resourceID of a newly created Timer resource.

- **Offline Shows**
  - Feature Double-clicking on a .cs2 file in the Finder now launches *Studio* and opens the selected file(s) for editing.
  - Bug Optimized the file format of .cs2 files, allowing *very* large shows to be uploaded/downloaded as much as 20x faster than before.
  - Bug Repaired the ability to create offline shows on the Windows version of Studio.
  - Bug The Layout Editor now works again while editing offline shows.
  - Bug Reopening an offline project's window now correctly reveals the existing window instead of creating a duplicate window.

- **Network Settings**
  - Bug Restored the ability for the *Network Settings* window to change settings on remote CueServers.
  - Bug Addressed a problem that would sometimes cause a warning icon to appear in the lower-left of the window even when there were no warnings to report.

- **Diagnostic Tools**
  - Feature Added new menu items that allow a .cs2 file to be unpacked/bundled for troubleshooting purposes.
  - Feature Added redesigned Network Info window that displays more information about the network interfaces on the host computer.

- **General**
  - Bug Addressed an issue that could crash the app when quitting in very rare circumstances.

Release v4.0.3 [July 22, 2020]

Version 4.0.3

Version 4.0.3 is a maintenance release with 9 important bug fixes and two new features. New features include improved non-volatile variable identification and a special recovery mode accessible by physical access to the device. Bug fixes focus on show file loading, reliability of non-volatile variables, and minor JavaScript improvements.

- **Navigator Window**
  - Bug Fixed an issue introduced in v4.0.1 that caused show files to be unable to be loaded into CueServers that did not have a password set.

- **Variables**
  - Feature Variables declared as *non-volatile* in either the Variable Editor or live Variables status monitor now appear with a light green background to denote that their values will be retained through a power loss and/or show switching.
  - Bug Addressed a problem where *non-volatile* variables would internally be reclassified as *volatile* if their value was set to NULL.
  - Bug Switching shows now removes the previous show’s *non-volatile* variables.
  - Bug The live Variable status window would get stuck if the current show contained no user-defined variables. This has been fixed.
  - Bug Fixed a problem where the Apply button could sometimes appear enabled even though the user had not made any changes in the window.
  - Bug Addressed a problem that could cause the unloading of *non-volatile* variables during a show switch to fail.
  - Bug Addressed a problem that rarely caused the contents of the FRAM chip to not be loaded properly during boot time, which would result in a loss of *non-volatile* variable restoration.

- **JavaScript**
  - Bug Addressed a problem where Timer resources were not being modified in a consistent way.

- **Operating System**
  - Feature A new method has been added that allows for activation of the self test mode, password reset, network reset, and/or factory reset by users with direct physical access to the pinhole Reset button on the device.

- **General**
  - Bug Fixed a problem introduced in v4.0.1 where a firmware update would not display its progress in real-time.

Release v4.0.2 [June 23, 2020]

Version 4.0.2

Version 4.0.2 is a maintenance release with 10 bug fixes. The major focus areas of this release include improving Windows compatibility, and addressing other user-reported issues.

- **Editor Window**
  - **Bug** Editing an offline show on Windows no longer crashes if the show file is using the `.cs2` format.
  - **Bug** The console view no longer has cosmetic blemishes on the edges of buttons on Windows.
- **Station Editor**
  - **Bug** Fix a problem that caused a crash if the Layout Editors from two different CueServers were opened at the same time.
  - **Bug** Fixed misleading labels for Increment/Decrement button actions when the button has a custom adjustment amount.
- **Functions Editor**
  - **Bug** Fixed a problem that caused the *Insert Function Call* button to be inoperable on Windows.
- **Navigator Window**
  - **Bug** Fixed an issue that could sometimes make the empty listbox banners display inappropriately.
- **Networking**
  - **Bug** Addressed a problem that caused a manual `Connect()` on a TCP Client that is already connected to crash CueServer.
  - **Bug** Addressed a problem that could cause a crash in the Telnet server.
  - **Bug** Improved HTTP connection reliability on Windows.
- **General**
  - **Bug** The *Software Update* window no longer appears blank on Windows.

Release v4.0.1 [April 21, 2020]

Version 4.0.1

Version 4.0.1 is a maintenance release with 3 new features and 11 bug fixes. The major focus areas of this release include resolving several bugs related to working with password-protected CueServers, improving auto-discovery, and fixing some issues with the layout editor.

- **Navigator Window**
  - Feature When a password-protected CueServer has been accessed using the correct password, the padlock icon now displays as being *unlocked*.
  - Bug Improved the auto-discovery feature to work across non-local subnets in more cases.
  - Bug Addressed a problem that may cause a crash when renaming a show on a password-protected CueServer.
  - Bug Fixed a problem that prevented the new .cs2 show files type to not be choosable via the Upload Show dialog window.
  - Bug Studio now properly asks the user to enter the password when trying to upload a show file to a password-protected CueServer.
  - Bug Displaying show files on a password-protected CueServer now prompts for the password before failing.
  - Bug Removed the padlock icon from show files in the CueServer directory listing.

- **CueScript**
  - Feature A new `PAGE CLEAR` syntax has been added allowing the current command context's page to be cleared.
  - Bug Addressed a problem that make it impossible to set button parameters of a non-paged station when accessing it from within the context of a different station that has pages.

- **Layout Editor**
  - Feature Added the ability to "export" images from Icon Buttons and page backgrounds back to files on the user's computer.
  - Bug Fixed the viewport dimensions for the HDTV/4K entry in the Viewport Chooser dialog.
  - Bug Deleting a layout widget no longer removes a linked resource unless there are no other widgets on the page that link to the same resource.
  - Bug Addressed a problem that could cause IDs of newly added widgets to conflict with existing resources after resources were manually added to the page.

- **Web Stations**
  - Bug Addressed a problem that could cause virtual Ultra stations to not appear properly.

Release v4.0.0 [March 12, 2020]

Version 4.0.0

Version 4.0.0 is a major new firmware release for CueServer that includes many new features. Major features include the ability to create interactive Web Stations with CueServer Studio’s built-in touchscreen layout editor and publish these screens on virtually any device with a web browser. A new Plugin system has been added that allows installers to easily add new functionality to CueServer by simply adding the plugins of their voice to their CueServer project. Plugins are available that respond to weather conditions, add new protocols, interface with various 3rd party devices, and even control Hue lightbulbs. Also, the JavaScript scripting language has been added to CueServer to allow advanced programmers to further customize their projects. In addition, many dozens of little feature enhancements and bug fixes makes v4.0.0 the biggest CueServer software upgrade we’ve released to-date. CueServer v4.0.0 is available to install on all existing CueServer 2 models.

- **Major Features**
  - New Web Stations: Create and deploy web-based touchscreen content.
  - New Plugins: Add new skills to CueServer to interface with the world.
  - New JavaScript: Develop custom scripts that take CueServer to the next level.

- **CueServer Studio**
  - New CueServer Studio now defaults to using compressed CueServer Project Bundle files (.cs2) to upload and download shows. These files are smaller and faster to work with than the previous folder-based show project files.
  - New Added a new JavaScript console to the Editor Window for entering live JavaScript commands.
  - New The Editor Window’s console area now includes a settings pushbutton in its top-right corner to access options for enabling the JavaScript console, toggling between the two console modes, and changing other console-related settings.
  - New Added new *Functions* panel to the Resources section to allow the user to create a mixture of CueScript and/or JavaScript functions.
  - New Added a *Function Browser* window that provides access to a library of built-in functions, documentation, and examples.
  - New Added a new *Event Handlers* panel to the Triggers section for creating *handlers* for events that are exported from installed plugins.
  - New Added the *Plugins* panel of the Settings section to provide management of plugins and their instances.
  - Feature Added the ability to take a remote CueServer offline with a new “Take Remote Offline” menu item.
  - Feature When attempting to open a remote CueServer that is offline, Studio will now prompt the user if they want to return the device to online status.
  - Feature Added the “Set Password” option to the CueServer device contextual menu.
  - Bug Typing a carriage return or enter key in the Cue Fade Times field now properly activates

the panel’s Apply button.

- Bug Fixed a bug that could cause CueServer Studio to become unresponsive if a very complicated group of thousands of channels is selected.
- Bug Studio now disallows certain special characters (`` +:;` ``) from being used in show file names to avoid various file system problems.
- Bug The warning icon next to *Settings* → *Universe Patch* now no longer appears for offline shows.
- Bug Fixed a problem that could cause a crash when editing offline show files that were missing some of its required components.

- **CueScript**
    - Feature Allow comments in CueScript using either `/* */` or `//` syntax.
    - Feature The `Record Group` and `Update Group` commands now implement the same command modifiers used by Cues and Presets. This includes the `Active`, `All`, `Empty`, `Selected`, `Input`, `Playback n`, and `Merge` modifiers.
    - Feature The `Write` command now accepts `LCD` as a destination, allowing user messages to appear on the LCD.
    - Feature Added the `Return` command so Functions written in CueScript can return values to the caller.
    - Feature The `Reboot` command now reboots the device much more quickly.
    - Feature Added new system variables for accessing device properties including `device.model`, `device.name`, and `device.serial`.
    - Bug The `Thru` command no longer fails if given `1` as its parameter.
    - Bug The `Reset` command now properly updates station indicator states.
    - Bug Fixed the `Record Group` command to be able to record groups sourced from other groups.
    - Bug Addressed a problem with the `Wait` command that would allow pending waits to execute after a `Reset` or after switching active shows.

- **JavaScript Framework**
    - New Added built-in *CueServer* library including 30 API functions for DMX, Network, Playback, Resources, System, and Variables sections of CueServer.
    - New Added built-in *JavaScript* library that includes 42 popular API functions common to JavaScript programming.

- **Fade Engine**
    - Bug The `Stack Clear` command previously reported that “Cue 0” was missing in certain situations. This has been fixed.
    - Bug Addressed a problem that could occur when using multiple cue stacks that would prevent the “Next Cue” from being properly calculated.
    - Bug Fixed an issue that was causing a capture of selected or active channels to fail when capturing into a Cue or Preset when the capture source was DMX Input, and the project was using more than one universe, and the selection was not an even multiple.

- **Touchscreen Controls**
    - New Added “Icon Button” control.
    - New Added “Touch Area” control.
    - Bug The Line control no longer behaves poorly when dragging endpoints or nudging its location with the keyboard arrows.

- **Stations**
    - Feature Pages and/or individual buttons on stations can now override the default zone for the station.
    - Bug Fixed a problem that caused touchscreen content to not appear on Insite if the show name contained an ampersand (&) character.
    - Bug Addressed a problem that could cause errors when loading station data from a show created by v1.x software.
    - Bug Optimized the performance of Web Station loading to prevent unnecessary support files from loading multiple times.
    - Bug Fixed a bug that may crash the event processor when an on-screen button is set to change pages.
    - Bug Resolved an issue that causes undefined behavior when a button has an assigned press-hold action and that same button also changes pages on its Press event.
    - Bug Addressed an issue where a touchscreen button could erroneously send its Release event to the wrong page when the station’s page changes while the user’s finger is still on the screen.
    - Bug Fixed a problem that could cause a crash if a Page event triggers a page change.

- **Layout Editor**
    - Feature Added a new icon chooser window used for the Icon Button control.
    - Bug Resolved an issue that would sometimes cause the Layout Editor to not load correctly.
    - Bug Addressed a problem that sometimes caused some of the Touchscreen Controls to not appear in the Library panel when editing a blank touchscreen layout.

- **Rules**
    - Feature Added new “Deactivated” and “Modified” events for Presets.
    - Bug Rule conditions for button indicators now work properly if the rule is being executed outside of the context of a button.

- **Timers**
    - Feature Automatically update the list of timers when timers are modified via JavaScript.
    - Bug Fixed a problem that prevented the user from entering offsets from Sunrise/Sunset in the second phase of an On/Off style timer.

- **Settings**
    - New Added a JavaScript panel in General Settings to adjust the JavaScript TCP server.
    - New Added CueScript TCP server settings to the CueScript panel of General Settings.
    - Feature Added a new *User String* option to the list of fields available to show in a quadrant of the LCD Display.

- **Real-Time Clock**
  - Bug The System Time panel would sometimes indicate that the system clock is synchronized with the NTP server, when it was still negotiating with the server.
  - Bug Changing between NTP and Manual time mode in the *Time Settings* dialog no longer changes the NTP Interval back to 0 (zero).
  - Bug Addressed a problem that made CueServer's Real-Time Clock drift inaccurately in certain situations when not synchronized via a Network Time Server.

- **Networking**
  - New Added new CueScript Console TCP server that can accept connections from clients via raw TCP or via TELNET protocol.
  - New Added new JavaScript Console TCP server that can accept connections from clients via raw TCP or via TELNET protocol.

- **Operating System**
  - Bug Fixed a problem that might have caused Web Stations to not operate after a reboot in very rare circumstances.
  - Bug Addressed a problem that could cause hardwired DMX I/O to fail to start after a reboot in extremely rare circumstances.

- **Plugins**
  - New Added *AccuWeather* plugin.
  - New Added *Color Math* plugin.
  - New Added *Lutron QS* plugin.
  - New Added *National Weather Service* plugin.
  - New Added *OSC Protocol* plugin.
  - New Added *Philips Hue* plugin.
  - New Added *TCP Client* plugin.

- **Examples**
  - New Added new *Restaurant Example*.
  - New Added new *Insite Demo* example.

- **Installer**
  - New CueServer Studio is now installed in a *CueServer Studio* folder containing the application, plugins, and examples.

Release v3.1.5 [November 8, 2019]

Version 3.1.5

Version 3.1.5 is a maintenance release with 2 new features and 5 bug fixes. This release is a “quick patch” for issues discovered in v3.1.4.

- **Station Editor**
  - Feature Added new direct button functions for *CueScript*, *Macro*, and *Switch to Page*.
- **Macro Editor**
  - Feature Renamed the macro testing button for clarity.
  - Bug Clicking on the *Test* button to test a Macro now properly executes the Macro in a temporary command context.
- **Timer Editor**
  - Bug Clicking on the *Test* button to test a Timer now properly executes the Timer in a temporary command context.
- **Rules**
  - Bug Addressed a problem that could cause the zone set in one rule to negatively affect other unrelated rules.
- **Web Stations**
  - Bug Addressed a problem introduced in v3.1.4 that caused the virtual keyboard to appear when viewing a web station on a mobile device.
- **Web API**
  - Bug Addressed a problem introduced in v3.0.0 that could cause the `get.cgi?req=bv` selector to crash.

Release v3.1.4 [November 6, 2019]

Version 3.1.4

Version 3.1.4 is a maintenance release with 6 new features and 20 bug fixes. The major focus areas of this release include the Layout Editor, Copy/Paste, Grid Settings, Control fixes, and general stability improvements.

- **Cues Editor**
  - Bug Addressed a problem that could cause a cue or preset to appear saved, when it had not been saved.

- **Zones Editor**
  - Feature Double-clicking on a zone now switches to the Zone Settings tab of the corresponding preset list.

- **Variables Editor**
  - Bug Fixed a problem that could cause a crash if the Revert button was pressed before saving a newly created variable.

- **Station Editor**
  - Bug Renumbering a station’s page no longer deletes the page’s layout.
  - Bug Stations without an assigned zone now display “None” instead of “Default”.

- **Layout Editor**
  - Feature Selecting and Moving layout objects has been improved by requiring a minimum drag threshold before interpreting a selection as a drag.
  - Feature Grid settings are now saved between sessions.
  - Bug Changing the Snap-To-Grid setting no longer requires selected objects to be re-selected to take effect.
  - Bug Prevent the layout grid from being visible after switching to Run mode.
  - Bug Moving an object in the layout now always makes the document modified.
  - Bug Copying and Pasting layout objects no longer sometimes copies outdated object properties.
  - Feature Pasting a copied object back into the same page from which it was copied now offsets the pasted object by the grid distance.
  - Bug Copy and pasting layout objects now works between pages and stations as intended.
  - Bug The Copy menu item is now properly enabled when first clicking on one or more objects.

- **Layout Controls**
  - Feature Mystique and Ultra stations now have improved legend display options that more properly match their physical engraving options.
  - Bug Slider track size and border width properties now work as expected.

- Bug Dragging a transparent slider thumb now properly immediately becomes opaque without waiting for the destination value to settle first.
- Bug Fixed a problem that caused long text labels in controls creating unexpected object selection handles.

- **CueScript**
  - Bug Fixed a problem with the `astro.light` system variable that caused it to always report that it was light out.

- **Stations**
  - Bug Addressed a problem that caused Press-Hold-Record to not work properly for Presets in certain circumstances.
  - Bug Fixed an issue that caused preset states to not update when using certain button actions.

- **Rules**
  - Bug Addressed a problem that sometimes caused the first action of a System Startup global rule to not execute.
  - Bug Indicator conditionals now properly default to the correct station number.

- **Web Stations**
  - Bug Improved the responsiveness of the PIN Entry overlay.

- **System Status**
  - Bug Improved reporting of running processes to avoid misleading information.

- **Windows Installer**
  - Feature The Windows installer now provides the option to install a desktop shortcut.

Release v3.1.3 [September 11, 2019]

Version 3.1.3

Version 3.1.3 is a quick patch that resolves a few minor issues found in v3.1.2. Most notably, this version restores the ability to set the timezone and set passwords.

- **Navigator Window**
  - Bug Restore the ability to set or change a device’s password. This was broken in v3.1.2.
- **CueScript**
  - Bug Fix a problem that caused the minus (-) selection command to fail if subtracting 1. For example, the command `BUTTON 1>8-1` would fail.
- **Time Settings**
  - Bug Restore the ability to set the device’s timezone. This was broken in v3.1.2.
- **General**
  - Bug Performance improvements to web API and front-panel UI.
- **Hardware**
  - Feature The Power LED will now flash Red/Yellow when it is unsafe to remove power from the unit. This occurs during firmware updates, and when setting the network or time preferences.

Release v3.1.2 [September 4, 2019]

Version 3.1.2

Version 3.1.2 contains 8 minor feature improvements and 13 bug fixes. Some of the more significant improvements include improved *Push-Hold-Record* options for buttons, new button actions, and optimized performance. The included bug fixes address virtual button stations, cue recording, minor CueScript issues, user-interface issues, and increased system reliability.

- **Stations**
  - Feature Added the ability to adjust *Variables* when defining a button’s actions.
  - Feature Added *Push-Hold-Record* and *Push-Hold-Adjust* options to button actions.
  - Feature Add a new *Raise/Lower* function to buttons.
  - Bug Fix a problem that prevented incremental button actions (increment/decrement/raise/lower) to not work properly on Presets.
  - Bug Analog input now properly works on buttons that act on Presets.
  - Bug Properly disable certain combinations of button function and target that aren’t applicable.
  - Bug Fixed a problem that caused virtual Mystique stations to display their indicators when being viewed from within CueServer Studio.
  - Bug Indicators on virtual Mystique stations now properly flash when instructed to do so.

- **Groups**
  - Bug The *Group* editor now properly restricts the group number field to the proper range of values (0 to 99999).

- **Timers**
  - Feature Renamed several of the timer modes to be more clear about their corresponding behavior.

- **CueScript**
  - Bug The CueScript validator now properly recognizes *Group 0* as being valid.
  - Bug The BUTTON x AT y command was improperly restricting group numbers if the button pointed to a group.

- **DMX Engine**
  - Bug Recording or Updating a cue now properly marks the playback fader as no longer being “edited”.

- **Navigator Window**
  - Bug Fixed spelling mistake in dialog that appears when a show cannot be opened.

- **Operating System**
  - Feature Optimized the device startup process, improving startup by 2.3 seconds.

  - **Feature** Improved overall reliability by fixing a problem that could cause the device to fail to boot in rare circumstances.
  - **Feature** Added fallback OpenDNS nameserver addresses when the device is not given addresses via DHCP to improve the chance that services such as NTP will be able to reach their services.
  - **Bug** Addressed a problem that could cause certain device services to not start properly when a CueServer is turned on.
  - **Bug** Improved the NTP client's ability to track time drift.
  - **Bug** Fixed an extremely rare problem that could cause a CueServer to not automatically reboot after a firmware update.

- **Insite Touchscreen**
  - **Feature** CueServer Studio now includes Insite firmware v1.0.4.

Release v3.1.1 [August 1, 2019]

Version 3.1.1

Version 3.1.1 contains 4 minor feature improvements and 10 bug fixes. Some of the more significant improvements include the ability to write HTTP requests to 3rd party devices, better scripting access to astronomical time parameters, and expanded variable substitution syntax. The included bug fixes improve performance and stability of both CueServer Studio and CueServer's firmware.

- **CueScript**
  - Feature Added the ability to send HTTP requests with the WRITE command.
  - Feature Added new system variables for accessing various astronomical time parameters including current daylight phase, and sunrise/sunset times.
  - Feature Added variable substitution syntax to CueScript strings (using ${variable-name}).
  - Bug String escape characters are now recognized properly by the CueScript validator.
  - Bug Fixed a problem that would crash the CueScript parser if it is given a script with a large nested command.
  - Bug Addressed a problem that could cause the ZONE command to become unresponsive after many repeated calls.
  - Bug Variable substitution no longer fails when the variable value contains a single-quote character.

- **Live Zones Panel**
  - Bug Fixed a crash when clicking on a zone when more zones are in the list than fit in the pane at once.

- **Zones Editor**
  - Bug Zones are now properly synched with the device when a Zone is renamed.

- **Macros Editor**
  - Bug The macro script text area now properly ignores styled text attributes.

- **Rules**
  - Bug The "This Button Was Held" condition now works properly in more scenarios.

- **CueServer OS**
  - Bug Removed spurious "lws_ring_insert_failed" log messages when using variable data on Insite pages.
  - Bug The incremental firmware updater now properly fully disables Telnet access.

- **Windows Installer**
  - Feature Added step to allow user to create desktop shortcut.

Release v3.1.0 [June 12, 2019]

Version 3.1.0

Version 3.1.0 is focused on Security. Many new features have been added that make CueServer devices installed on networks more secure, including the ability to add passwords, prevent remote login attempts, and lock down various security risks. Other major features include the ability to add dynamic text to layouts, labeling of Mystique and Ultra buttons, and improved Copy/Paste features. Over 30 features and bug fixes are included in this release.

- **Major Features**
  - New New Security Features
  - New Dynamic Text in Layouts
  - New Labeling of Mystique/Ultra Buttons
  - New Improved Copy/Paste of Station Resources

- **Security**
  - Feature Allow CueServer devices to be password protected.
  - Feature Added UDP CueScript reception filters based on IP address and/or subnet.
  - Feature Disable reception of UDP CueScript commands by default.
  - Feature Added “Cross Origin Resource Sharing” (CORS) settings for CueServer’s Web-API.
  - Feature No longer allow Telnet access to CueServer’s back-end.
  - Feature Enable SSH for console access.
  - Feature Disable root login by default.

- **Macro Editor**
  - Feature The *Macros* editor panel now uses better formatting for the Script column to show more of each macro’s CueScript.

- **DMX Engine**
  - Bug Addressed a problem where sACN output could stop if the Ethernet switch is disconnected and reconnected.

- **Layout Editor**
  - Bug When adjusting the z-order or alignment of controls, the Apply button now properly becomes enabled.
  - Bug Added missing vertical-align-center button.
  - Bug When “nudging” controls with the arrow keys, the inspector now properly updates.
  - Bug Fixed a problem that prevented multiple objects to be dragged simultaneously.
  - Bug Navigating between pages now auto-saves the layout.

- **Layout Controls**
  - Feature Added new syntax for inserting live (dynamic) variable values into layout controls (using

${variable-name}).
    - **Feature** Added the ability to add legends to virtual Mystique and Ultra stations.
    - **Feature** The layout controls have been optimized for increased performance and reliability.
    - **Bug** Improved the Color Picker's calculation of SV values.
    - **Bug** The Color Picker no longer clips the ends of its sliders when resizing the control bounding box.

- **Rules**
    - **Bug** Addressed a problem that would cause SET rules to fail on paged stations in certain circumstances.

- **Show Database**
    - **Bug** Fixed a problem that would report an error when renaming a show that contained non-volatile variables.
    - **Bug** Attempting to upload a show with the same name as the active show now properly provides an warning dialog.

- **Stations**
    - **Feature** Resources within stations (Buttons, Contacts, Outputs, Ports) can now be Copied/Pasted between stations, pages, and/or shows.
    - **Bug** Addressed a problem that could cause the indicators on first-generation 5-wire CueStations to not operate properly.

- **Timers**
    - **Bug** The AM/PM sectors are now visible in Repeating and Hourly Timers.
    - **Bug** Fixed the action text to indicate that repeating timers use seconds.
    - **Bug** Fixed a problem that would create UI artifacts when switching timer types in certain circumstances.

- **Time Settings**
    - **Feature** Added the ability to set the maximum interval in which NTP Servers are polled.

- **CueServer OS**
    - **Feature** Changed the factory default NTP polling interval from 36 hours to 1 hour.
    - **Bug** Improved the reliability of the NTP process.

- **Windows**
    - **Feature** Windows installer now properly adds *CueServer Studio* to the Add/Remove Programs interface.

Release v3.0.1 [May 2, 2019]

Version 3.0.1

Version 3.0.1 contains 17 minor feature improvements and 22 bug fixes. Some of the more significant improvements include eight new color picker variants, two new timer triggering modes, editable Button IDs, enhanced NTP server integration, and improved SD Card error handling. The included bug fixes improve performance and stability of both CueServer Studio and CueServer's firmware.

- **Navigator Window**
  - Feature A warning icon (with help text) will appear next to any CueServer that is either missing its SD Card, or if an error was detected when writing to the card.
  - Feature If a CueServer is opened that has an SD Card with read/write problems, a warning dialog will appear allowing the user to try to resolve the issue.
  - Bug Addressed a problem that would make a show not disappear from the show list when deleting a show from a CueServer device.

- **Editor Window**
  - Feature If a write error occurs on the device's SD Card while editing, a warning dialog will appear and the window will close.

- **Live Status**
  - Feature The *System Clock* panel now includes an advanced option to show the current NTP Server Status.
  - Feature The *System Clock* panel now resizes to fill available window space.
  - Feature A new debug option to show "Real Time Clock Events" has been added to the *System Log* panel.
  - Bug System Log messages regarding enabling and disabling of the NTP client have been fixed.

- **Stage View**
  - Bug Fixed a bug that would crash when trying to view the Stage when no channels were patched.

- **Cue/Preset Editor**
  - Feature The *Capture* panel now includes options for merging selected or active channels in the Source drop-down menu.
  - Feature The *Capture* panel now remembers the last used input source.
  - Bug When pasting a Cue as a new resource, the new cue number now uses a whole number instead of a fractional number.
  - Bug Addressed a problem that created drawing artifacts when scrolling recorded channels in the *Cue Contents* panel.

- **DMX Engine**

    - Feature Playbacks set to *Scale* or *Pin* mode now begin fading previously released channels from FL to produce a more gradual introduction of the channel value.
    - Bug Fixed a bug that could cause a crash when updating a Cue from DMX Input when the existing cue had more channels recorded in it than the new cue will have.
    - Bug Addressed a problem that could corrupt a cue when capturing from DMX Input when the Universe Patch includes universes with less than 512 channels.

- **Stations**
    - Feature Button IDs may now be changed on stations with variable numbers of buttons.
    - Bug Web-based *Ultra* stations now work properly again.
    - Bug Stations without a layout (for instance the built-in station) now display a warning icon when that station's web view is opened.

- **Layout Editor**
    - Bug Addressed a problem that would cause mouse clicks inside the Layout Editor to sometimes not properly "focus" the editor for keyboard input.

- **Layout Controls**
    - Feature Eight new color picker variants including new dials and sliders have been added to the Color Picker control.
    - Feature The inspector panel for the Line object has been improved.
    - Bug The Color Picker now displays in the center of the bounding rectangle instead of the top-left corner.
    - Bug The Color Picker no longer draws a partial thumb when displaying full red.
    - Bug The Color Picker can now track mouse or finger movements that temporarily move outside of its bounding rectangle.
    - Bug Fixed a memory leak in the Slider object that would cause a touchscreen or web browser to become slow or unresponsive after a large number of touches.
    - Bug A newly created Line object now appears with similar orientation to its thumbnail icon.
    - Bug The *Tail (Hollow)* decoration of the Line object now appears in the correct location relative to the end of the line.

- **Timers**
    - Feature Added new *Hourly Schedule* mode to timers, allowing a timer to execute at one or more specified times within each hour.
    - Feature Added new *Repeating Events* mode to timers, allowing a timer to execute multiple times within a time window.
    - Bug The *Timers* panel now properly indicates when a timer does not have an action assigned to it by displaying *none*.

- **CueScript**
    - Bug Addressed an issue that prevented the `MERGE` option in the `UPDATE CUE` command to work properly.
    - Bug The `TIME` and `FADE` commands can once again use variables as their parameters.

- **CueServer OS**
    - Feature Improve the SD Card write performance to lessen the likelihood that the card could become corrupted.
    - Bug Addressed an issue that could cause one or more of the internal processes to crash in certain circumstances after operating under moderate load for an extended period of time. This issue may have affected serial port reception, DMX Input Triggers, timers, and/or station button presses.

- **Windows Version**
    - Feature Various performance enhancements to make the application more responsive.
    - Bug Listbox sort indicators now appear properly.
    - Bug Listbox hierarchical widgets and checkboxes no longer look blurry.

- **Application**
    - Feature The application’s Splash Screen (and About window) have been redesigned to not cover up as much of the project photo, and to also look better on Windows.

Release v3.0.0 [April 4, 2019]

Version 3.0.0

Version 3.0.0 adds many exciting new functions to CueServer including fully interactive web-based CueStations, a built-in graphical layout editor, entirely new station station pages, support for the new Insite 7” Touchscreen, plus the ability to create generic web-based touchscreens.

- **Major Features**
  - New Fully Interactive Web-Based CueStations.
  - New Built-in Graphical Layout Editor.
  - New Entirely new Station Management Tools.
  - New Advanced Button Actions.
  - New Station Pages.
  - New Support for new 7” Insite touchscreen.
  - New Support for the new CS-950 model.
  - New Copy & Paste for Resources is Now Available

- **Special Notes for v3.0.0**
  - Special Note CueServers updated to firmware v3.x can be downgraded to v2.x by simply loading firmware v2.1.2. It is NOT recommended that a v3.x CueServer is downgraded to v2.1.1 or earlier in a single step.

- **Editor Window**
  - Feature The command line’s status bar now shows the current station’s active page and the target page.
  - Bug An issue that could cause a crash when opening an offline show has been fixed.
  - Bug The Editor Window no longer crashes if it is closed while the stack/zone lists are loading.

- **Stage View**
  - Bug Fixed a spelling mistake.

- **Playbacks View**
  - Bug Fixed a problem that could cause Playbacks 26+ to appear improperly in the Playbacks view.

- **Stations**
  - Feature The *Stations* panel has entirely been redesigned.
  - Feature Stations are now viewed and selected by changing the popup menu in the top-left of the panel.
  - Feature The Station navigation panel shows choices for a station’s settings, optional page settings, layout, and the button/contact/output/port resources applicable for that station.
  - Feature Support for the Insite 7” Touchscreen has been added.

- Feature Support for generic web-based touchscreens has been added.
- Feature Layouts can be built using a variety of built-in controls, including Button, Clock, Color Picker, Ellipse, Image, Image Button, Line, Rectangle, Slider, Text, and Toggle.
- Feature *Web Access* for any station may be enabled, providing a CueServer-hosted URL that displays the station's web representation.
- Feature Web-based stations may be protected by a user name and password.
- Feature Stations now have *pages* of buttons.
- Feature Legacy stations (Mystique/Ultra) can be optionally upgraded to use pages.
- Feature Pages have various automation rules, including Page Open, Page Close, Page Idle, etc.
- Feature Pages may be protected by PIN number.
- Feature The *layout* of each station's page(s) may be edited with a new graphical layout editor.
- Feature Buttons and contacts can now use predefined *Actions, Targets, and Options* that make it easy to define sophisticated functions and indicator behavior for a variety of controls.
- Feature Predefined button actions may use variable values to change button behavior on the fly.
- Feature Mystique and Ultra stations now turn off when disabled, mimicking the same behavior as an Insite touchscreen.

- **CueScript**
  - Feature The STATION command can now be used with the AT command to change the active page of a Station. For example, the command `STATION 1 AT PAGE 3` changes Station 1's active page to 3.
  - Feature A new PAGE command can now be used to change the station page for which subsequent commands operate. For example, the command `PAGE 5 BUTTON 1 ON` turns on Button 1 on Page 5 of the current station.
  - Feature The LOCK and UNLOCK commands have been extended to work on pages that include PIN numbers. Locking a Page using a command such as `PAGE 3 LOCK` de-authenticates users from viewing the page. Unlocking a Page using a command such as `PAGE 3 UNLOCK` authenticates users to view the page.
  - Feature The LOCK and UNLOCK commands have been extended to work on stations with page layouts. Locking a Station using a command such as `STATION 1 LOCK` causes the page contents to become grayed out with a superimposed lock icon on top. No user interaction can occur on locked pages. The UNLOCK command unlocks a previously locked station.
  - Feature The ENABLE and DISABLE commands have been extended to work on stations with page layouts. Disabling a hardware station (such as the Insite touchscreen) causes the station to turn its screen off. Disabling a software station (such as a station viewed through a web browser) displays a gray screen with a "null" (Ø) icon. Enabling a station returns its operation to normal.
  - Bug Non-volatile variables are now preserved properly after power loss or reboot.
  - Bug The inline CueScript parser would improperly show that a SET command with a numeric literal, such as `Set "x" = 3` was an error.
  - Bug The CUE command no longer fails to report an error if no parameter is given.

- **DMX**
  - Feature Added Mask mode to Playback faders.
  - Bug Fixed an issue that prevented more than 32 sACN input universes to be able to be received. The new limit is 128 sACN input universes.
  - Bug Fixed an issue that caused DMX Input Triggers to not function properly when set within universes that were patched with less than 512 channels.

- **Serial**
  - Bug Fixed an issue that could cause a crash if incoming serial is being processed as CueScript commands while replies are being sent back to the serial output, and the hardwired DMX output ports are in use.

- **Windows**
  - Feature A new Windows installer is smaller and faster.

Release v2.1.2 [January 18, 2019]

Version 2.1.2

Version 2.1.2 contains several bug fixes. This version also adds support for the new CS-950, a new playback fader mode, and it includes a smaller, faster Windows installer.

- **Major Features**
  - New Added support for the new CS-950 model.
- **Editor Window**
  - Bug An issue that could cause a crash when opening an offline show has been fixed.
  - Bug The Editor Window no longer crashes if it is closed while the stack/zone lists are loading.
- **Playbacks View**
  - Bug Fixed a problem that could cause Playbacks 26+ to appear improperly in the Playbacks view.
- **CueScript**
  - Bug Non-volatile variables are now preserved properly after power loss or reboot.
  - Bug The inline CueScript parser work improperly show that a command like `Set "x" = 3` was an error.
- **DMX**
  - Feature Added Mask mode to Playback faders.
  - Bug Fixed an issue that prevented more than 32 sACN input universes to be able to be received. The new limit is 128 sACN input universes.
- **Serial**
  - Bug Fixed an issue that could cause a crash if incoming serial is being processed as CueScript commands while replies are being sent back to the serial output, and the hardwired DMX output ports are in use.
- **Windows**
  - Feature A new Windows installer is smaller and faster.

Release v2.1.1 [May 18, 2018]

Version 2.1.1

Version 2.1.1 is a quick patch that resolves a few minor issues found in v2.1.0.

- **CueScript**
  - Bug Minor CueScript Helper syntax improvements for the Enable, Record, and Wait commands.

- **Settings**
  - Bug Removed an unnecessary checkbox from the *Settings* > *General* > *SMPTE* panel.

Release v2.1.0 [May 16, 2018]

Version 2.1.0

Version 2.1.0 adds a suite of SMPTE Timecode features, a Live CueScript Helper, high-resolution graphics, plus dozens of other important enhancements to CueServer 2.

- **Major Features**
  - New SMPTE Timecode
  - New Live CueScript Helper
  - New Retina/Hi-DPI Graphics
  - New New Windows framework for improved flicker-free drawing
  - New CueServer Studio is now 64-bit for macOS

- **Editor Window**
  - Feature The command line and other CueScript entry windows now include a live “CueScript Helper”.
  - Feature A new *Timecode Event Editor* has been added to the Triggers section.
  - Feature The CueScript editor popup window is now resizable.
  - Bug Fixed a crash that could occur if the Stage View palette window is opened before the embedded Stage View panel.
  - Bug The *Station Status* field of the command line could temporarily display incorrect information.
  - Bug Typing the *Forward-Delete* key on the command line did not delete the proper character.

- **CueScript**
  - Feature A new SMPTE command has been added that allows the current timecode to be set, cleared, started, and stopped. Additionally the SMPTE command can also be used to enable or disable the external SMPTE Timecode Audio input and control internal generation of timecode.
  - Feature CueScript buttons are now formatted with better multi-line support.
  - Bug Updating a Preset now properly updates the same preset in joined zones.
  - Bug Fixed a possible crash that could occur if a WAIT CLEAR is executed in the same command string after a WAIT.
  - Bug The BREAK command could crash if it was in an IF/THEN/ELSE clause.
  - Bug The CUE command was improperly limiting the highest Cue Number below the actual maximum of 999,999.99.

- **Groups**
  - Feature Hovering over the Channels column in the Group List now reveals a popup window that lists all of the channels in the group.
  - Bug Deleting all of the text from the Channels field is now possible.

- **Zones**

- Bug Fixed a crash that could occur if the Zones panel is closed while it is still fetching the zone list from the show file.

- **Rules**
    - Bug Fixed a firmware crash that would occur if a SET rule is created and then saved without specifying an action.
    - Bug Some *Preset Actions* were failing to execute if assigned to the highest numbered button on a station.
    - Bug Popup buttons in rules are now visible on lines with long CueScript.
    - Bug Rule actions within Cues and Presets did not have the proper default Playback Fader.

- **Settings**
    - Feature A new *Timecode* panel in General settings allows external SMPTE Timecode audio input processing to be enabled or disabled, and also to set a threshold for jumping over or fast-forwarding through discontinuous time periods.
    - Bug Fixed a problem that could cause show file corruption after increasing the number of Playbacks in the *Settings* > *DMX* > *Resources* panel.

- **Status**
    - Feature A new *Timecode* panel in the System Status area displays the live timecode along with internal generation and external input indicators. The current dBFS scale of the audio input is displayed as well.

- **Audio**
    - Feature A new Audio Input processor is included that can receive and decode incoming Linear Timecode (LTC) signals.

- **CueServer Studio**
    - Bug Restore the ability to update the firmware of a remote CueServer that was broken in v2.0.4.

Release v2.0.4 [March 14, 2018]

Version 2.0.4

Version 2.0.4 is an incremental update to CueServer 2 including 10 general feature enhancements, and 11 bug fixes. Focus areas of this release include working with show files, CueScript additions, system log improvements and general usability.

- **Navigator Window**
  - Feature Downloading a show now warns the user if the operation will overwrite an existing file.
  - Feature Creating a show with an existing name now warns the user.
  - Feature Firmware updates are now disallowed on CueServers that are not on the same network subnet as CueServer Studio.
  - Feature The contextual menu for CueServer devices now includes the *Apply License Code* and *Update Firmware* items.
  - Bug The *Upload Show* file chooser no longer allows non-show directories to be uploaded.
  - Bug Addressed an issue that could cause a crash if a show upload/download is cancelled.
  - Bug If a show upload is cancelled while in progress, the partially uploaded show is now properly removed from the device.
  - Bug Fixed a crash that would occur if an offline show is deleted or moved before editing.
  - Bug Fixed a crash that would occur if an offline show is deleted or moved while its editor window is open.

- **Editor Window**
  - Feature Switching away from the *Status* panel and then back now remembers the last used sub-panel.

- **CueScript**
  - Feature A new LOG RESET command has been added that removes all entires from the System Log.
  - Bug Unterminated strings now fail properly.
  - Bug The PLAYBACK command now provides the proper error message if an incorrect parameter is given.
  - Bug Addressed a problem that prevented decimal numbers without a leading zero in expressions (such as “5 * .3”) to be evaluated properly.

- **Rules**
  - Feature A confirmation dialog is presented before a rule is deleted.
  - Feature The “port letters” for DMX Ports can now be used in DMX Port event rules.
  - Bug Global button/contact rules would fail when specified without a station number has been fixed.

- **System Log**

  - **Feature** A new *Clear Log* button has been added to the System Log panel.

- **DMX Triggers**
  - **Bug** The 16-bit checksum opcode (“\S”) had previously been outputting the value in the wrong byte order.

- **LCD Display**
  - **Bug** Fixed a problem that could cause Macros to be listed out of numerical order.

- **Installer**
  - **Feature** The macOS .dmg image now includes Retina artwork.

Release v2.0.3 [February 14, 2018]

Version 2.0.3

Version 2.0.3 is an incremental update that concentrates on bug fixes. This update includes the following 13 improvements:

- **Editor Window**
  - Feature Renamed “Rules” to “Global Rules” for clarity.
  - Bug Editing an offline show now properly hides the live controls.
  - Bug Chevron buttons no longer disappear after a warning icon displays on that row.

- **CueScript**
  - Feature Variables can now be used in arrays.
  - Bug The PLAYBACK command no longer reports an error if used with the wildcard (*) to select all Playbacks.
  - Bug Nested CueScript can now return string values to the calling context.

- **Presets**
  - Bug Rules in Presets no longer improperly show “Whenever This Cue…”.
  - Bug The “Whenever This Preset” event is now triggered properly.
  - Bug The Preset Toggle rule action now operates as expected.

- **Timers**
  - Bug Addressed a problem that could prevent astronomical time events with negative offsets to be switched to positive offsets.
  - Bug Attempting to set the offset of an astronomical event beyond 360 minutes no longer created a UI inconsistency.

- **LCD Display**
  - Bug Fixed a problem introduced in v2.0.2 that prevented the LCD Menu to be able to change shows and/or switch DHCP mode.

- **Serial Ports**
  - Bug Setting the serial output format to “8-O-1” or “8-E-1” now properly outputs the parity bit.

Release v2.0.2 [January 22, 2018]

Version 2.0.2

Version 2.0.2 is an important update to CueServer 2 including 11 general feature enhancements, and 17 bug fixes. Focus areas of this release include the WAIT command, Macro behavior, and CueScript bug fixes.

- **Navigator Window**
    - Feature The *Network Settings* window now displays the device's MAC Address(es).
    - Feature A show can now be deleted by pressing the Delete or Backspace keys.
    - Bug Addressed a problem that sometimes caused shows to not be removed from the show listing after being deleted.
    - Bug Renaming a show no longer produces an error message when the show contains non-volatile variables.
    - Bug No longer allow slash characters in show file names.
    - Bug Addressed a problem that could allow the window's toolbar items to be improperly enabled after deleting a show file.

- **CueScript**
    - Feature Several new escape codes have been added to the handling of strings to enable the automatic substitution of channels, levels and checksums into strings.
    - Feature Added several new system variables to get and set the date, time, and timezone in several ways.
    - Feature CueScript editors now allow tab characters.
    - Feature Added the WAIT STOP command that allows a single pending Wait to be stopped before it fires.
    - Feature Added new \`accent quotes\` syntax for allowing the value of a variable to be executed on the command line.
    - Bug The remainder of a command that executes after a WAIT now properly executes in the same command context as the beginning of the command.
    - Bug Commands that operate on triggers (such as PRESS and RELEASE) now properly work after a WAIT command.
    - Bug Calling a macro that contains WAIT commands no longer causes an out-of-order command execution sequence.
    - Bug Macros now properly execute in the same command context as the calling script.
    - Bug Multiplication and Division operators now work as expected.
    - Bug Concatenation now works properly when combining numeric values to strings.
    - Bug Inline variable substitutions of string values now work as expected.
    - Bug Addressed a problem with nested CueScript returning floating point numerical results.
    - Bug Pressing Enter or Return on an empty command line no longer sends an empty command to the CueServer.
    - Bug Addressed a problem that could occasionally cause the CueScript parser to crash when an

inline variable substitution occurs within an IF/THEN/ELSE/ENDIF block.

- **Editor Window**
    - Bug Adding or removing a rule from a Cue, Station, or DMX Trigger now no longer scrolls the rule list back to the top.

- **DMX Triggers**
    - Feature A new “Act on Changes” function has been added to DMX Triggers.

- **Status**
    - Feature Status panels can now be opened into their own separate palette windows.
    - Bug The *RAM Used* bar graph in the *CPU Info* panel was incorrect and misleading. More accurate RAM Usage numbers are now shown.

- **LCD Display**
    - Feature Time, Date, and Time Zone can now be changed from the LCD Display.

- **Show Database**
    - Feature Creating a new show file now pre-configures the show based on the hardware device it is created on.

- **API**
    - Bug Setting the Timezone via set.cgi or UDP is now much faster.

Release v2.0.1 [November 3, 2017]

Version 2.0.1

- **Live Stage**
    - Bug The *Input* layer in the *Stage View* no longer shows incorrect channel values when the DMX patch contains universes with less than 512 channels.
    - Bug The *Stage View* no longer crashes when viewing more channels that are licensed on the device.
    - Bug The *Stage View* no longer crashes in certain circumstances when displayed universes have less than 512 channels.

- **Stations**
    - Bug Stations that are “locked” no longer allow button presses to execute.

- **Settings**
    - Bug The *Settings* > *LCD Display* panel now works properly when editing an offline show file.
    - Bug The *Settings* > *LCD Display* panel no longer “beeps” when opening.

- **LCD and Front-Panel Display**
    - Feature The Power LED indicator now flashes Red/Blue when the device is in Identify Mode.
    - Bug Fixed a bug introduced in v2.0.0 that prevented the *Self Test* to start properly in certain circumstances.

Release v2.0.0 [October 24, 2017]

Version 2.0.0

Version 2.0.0 is a significant update to CueServer 2 including 14 major new features, 67 general feature enhancements, and 54 bug fixes.

- **Major Features**
    - New Zones and Presets have been added.
    - New Rules have 18 new built-in action templates.
    - New Object lists now have a live column for easy monitoring and control.
    - New New Stage View options show channels grouped by universe.
    - New Automatically updating indicators.
    - New Button Indicators are now fully configurable via popup palettes.
    - New Improved universe patching system that’s more flexible and powerful.
    - New The licensing model now enables channels instead of universes.
    - New Variables can now be predefined and/or designated as non-volatile.
    - New Added additional rule conditions.
    - New New Settings panels, including settings for Hardware, Stations and Audio.
    - New Redesigned command line status provides more contextual details.
    - New Expanded KiNET v2 support.
    - New Application Preferences have been added.

- **Live Stage**
    - Feature The Stage View now shows channels that are not accessible because of the currently selected zone.
    - Feature The Stage View now shows selected channels in the current playback color.
    - Bug Addressed a problem that caused the input view of the *Stage* window to not show channel values for incoming Ethernet protocols.

- **Live Playbacks**
    - Bug Addressed a problem where the active playback indicator was not being drawn properly in the *Playbacks* panel [Windows only].

- **Live Zones**
    - Feature Added new Zones panel to the Live section that shows the current presets and join status of each zone.
    - Feature When in a zone, commands that set channel values are now prevented from modifying channels outside of the zone.

- **Live Status**
    - Feature The Front Panel Status panel now reflects the LCD backlight brightness.
    - Feature The *Status* > *Variables* panel now sorts the variables alphabetically.

- **Cues**
    - Feature Added additional cue details to the Extras column.
    - Feature When adding Cue Stacks, the main *Cues* navigation item auto-expands if it had been closed.
    - Bug Addressed a problem that caused the capture channels popup to default to *No Channels* when a Cue was first created.
    - Bug If a Cue or Preset is new or edited, the capture panel now requires the user to save or apply changes before new channel levels can be captured.
    - Bug Addressed a problem that caused streaming cues to not be duplicated properly with the Duplicate Cue command.
    - Bug Addressed a problem that caused a crash if a Cue was duplicated on an international system that uses the comma character as numerical decimal separator.
    - Bug Addressed a problem that could cause a cue to loose its name when re-recording its streaming content.
    - Bug Addressed a problem that could cause the Capture and Record buttons in the Capture panel to be enabled when no cue is selected.
    - Bug Addressed a problem that could cause capturing channels in a cue to fail if the cue is in the default stack but the current playback fader has a different stack selected.

- **Zones**
    - Feature Added new Zones list view that shows an overview of each zone defined in the system.
    - Feature Added Zones sub-views that allow Presets to be added to each zone.
    - Feature Added a zone configuration panel that configures each zone.

- **Variables**
    - Feature Added new Variables panel that allows a project to predefine variables and/or designate them as “non-volatile”.
    - Feature Non-volatile variables retain their values when the power is lost or shows are switched.

- **Timers**
    - Feature Added additional information to the details column of the Timers list.
    - Bug Addressed a problem with the Timer List panel that would draw the list controls improperly.
    - Bug Addressed a problem where the checkbox labels may be truncated in the year-picker window.

- **Rules**
    - Feature The THEN clause in rules now include 18 new canned actions, including operation on cues, playbacks, channels, groups, presets, indicators, outputs, etc.
    - Feature Added new Date, Month, Day, Year, Hour, Minute and Second conditionals to rules.
    - Feature Added new Was Held/Was Not Held conditions for Button and Contact rules.
    - Feature When choosing a time-based date conditional, the current time and/or date appear as defaults.

- **Stations**

- Feature A new cascading button indicator architecture has been added. Each indicator in the system now has four levels of scope: live, button, station and global. The global scope has the lowest priority and the live scope (as set by the SET command) has the highest priority.
- Feature Button indicators now have eight customizable states named: On, Off, Mixed, Locked, User 1, User 2, User 3, and User 4.
- Feature A new button indicator color picker has been added to general preferences, station and button editor panels.
- Feature Stations now have a Zone popup menu that allows each station to be assigned to a zone.
- Feature The *Station* and *Button* editor panels now allow their indicator colors to be set.
- Feature Added a Test button to the button/contact closure panel to allow for live testing of press/release events.
- Feature Added the ability to communicate with CueStation Hubs via RS-232 and/or RS-485.
- Feature Added the ability to set serial ports to use CueStation Hub protocol.
- Bug Switching shows now automatically refreshes all button indicators in the system.
- Bug The serial port panel no longer allows the user to choose to reply with CueScript result if one of the CueScript protocols are not selected.
- Bug Improved station handling performance.

- **Timers**
  - Bug Addressed a problem that could cause the Months or Years popup controls to show “Unknown” if the corresponding popup window was dismissed while no months or years were selected by clicking outside of the window.
  - Bug Addressed a problem that caused Sunrise and Sunset offset times to be limited to a maximum of +/- 60 minutes.

- **Settings**
  - Feature Added a new *Hardware* settings panel to choose which hardware type is being used.
  - Feature Moved the old *General* settings panel to a new *Notes* settings panel.
  - Feature Added a new *General* settings panel that contains several subcategories of simple preferences.
  - Feature Added a new *General* > *Indicators* settings panel to set the global button indicator colors.
  - Feature Added a new *General* > *Audio* settings panel to set the audio output volume.
  - Feature The LCD Display settings panel now updates the device “live” when adjusting settings.
  - Feature Added backlight brightness control to the LCD Display settings panel.
  - Feature The “Show on Maps” button in the Location panel now defaults to using the Apple Maps service instead of Google Maps (although pressing the Option key switches back to Google Maps).
  - Feature Added a “Search timeanddate.com” button.
  - Bug Addressed a problem that caused Latitude/Longitude changes to require a reboot or show reload to take effect in certain circumstances.

- **Editor Window**
  - Feature Added an expanded target description and stack, zone and station context to the command line status bar.
  - Feature The command line status bar is now drawn in the currently selected playback color.
  - Bug Addressed an issue in the Macros, Timers, Rules, and DMX Triggers editor panels that could display the "Cancel"/"Save" buttons when making changes to an existing resource instead of the "Revert"/"Apply" buttons.
  - Bug Addressed a problem that could cause a crash if the stand-alone Playbacks window is opened during switching of active shows.
  - Bug Addressed a problem that could cause a crash if a show file's resources are not saved properly (due to card removal or power outage).
  - Bug Addressed a problem that could cause a rare crash during show switching.
  - Bug Addressed a problem that could sometimes cause CueScript statements to draw outside the bounds of a CueScript button.
  - Bug Addressed a problem that sometimes caused command status changes to appear to lag after a new command was submitted on the command line.
  - Bug Addressed a problem that could case a rare crash while entering text into a new rule.

- **Navigator Window**
  - Feature Time Zone changes now take effect more quickly.
  - Bug Addressed a problem that could cause a navigation list to have no visibly selected item even though its editor panel is shown if the item is clicked quickly while the mouse is moving off the side of the list.
  - Bug The *New Show* window now prevents show names from containing double-quote characters.
  - Bug Addressed a problem where CueServer Studio could become unresponsive after a CueServer device goes offline and then later returns.
  - Bug Addressed a problem that caused shows from being deleted if they included 'single-quote' characters.

- **CueScript**
  - Feature The CHANNEL command and associated selection commands have been extended to support channel numbers in the *universe.channel* format.
  - Feature A new DO command was added to allow manual triggering of a button or contact's events based on the current physical state of the button or contact.
  - Feature The LOG command can now log scalar values and arrays in addition to strings.
  - Feature Escape characters (i.e.: \n, \r, \x00, etc.) are now processed in all strings.
  - Feature Added the AT INPUT syntax to the AT command.
  - Feature Added the AT OUTPUT syntax to the AT command.
  - Feature The AT command can be used to set button indicators to User colors by using the constants 1, 2, 3, and 4.
  - Feature The AT command can now set the submasters of multiple playbacks simultaneously using a command such as [Playback 1>5 At 50].

  - Feature The CLEAR command can now clear multiple playbacks simultaneously using a command such as [Playback 6>10 Clear].
  - Feature Playback faders can now be enabled and/or disabled using [Playback *n* Enable/Disable].
  - Feature Added the JOIN command. Zones can become members of a join pool with a command such as [Zone “Lobby” Join 3].
  - Bug Nested CueScript statements now properly inherit the context of the parent.
  - Bug Addressed a problem that caused the WRITE command to not properly process escape characters in the output string if the destination was a UDP message.
  - Bug Addressed a problem with the MINUS command that would sometimes deselect the wrong object.
  - Bug Addressed a problem that caused SET and WRITE commands nested within an IF/THEN/ELSE statement to cause the entire script to fail to execute.
  - Bug Addressed a race-condition that could allow an auto-follow to occur while the RESET command is executing which could cause Cue 0 (zero) to attempt to execute.
  - Bug Addresses a problem that could cause the RECORD or UPDATE commands to record a cue into the wrong cue stack in certain circumstances.
  - Bug Addressed a problem where the assignment operator would not accept negative numbers.
  - Bug Addressed a problem where the RELEASE command would not entirely release a streaming cue.

- **DMX**
  - Feature CueServer Universes can now be scaled down to any number of channels for more efficient usage of resources.
  - Feature Each CueServer Universe can now have an arbitrary number of “extra outputs” added to allow the same data to be retransmitted with a different protocol or to a different IP address, different KiNET port, etc.
  - Feature A range of incoming priorities can now be specified in the sACN input configuration of a universe.
  - Feature New system variables ‘universe.rxpriority’, ‘universe.rxprioritylow’ and ‘universe.rxpriorityhigh’ have been added to dynamically control the range of incoming sACN priorities that will be received for a particular universe.
  - Feature The ‘universe.priority’ system variable has been renamed to ‘universe.txpriority’ to avoid confusion with the new receive priority variables.
  - Feature DMX Output ports can now independently transmit DMX in one of 5 speeds (40Hz, 38Hz, 35Hz, 30Hz, or 20Hz), each of which has increasingly exaggerated DMX timing to allow receivers with poorly implemented DMX protocols to (hopefully) work properly.
  - Feature The built-in DMX output ports now only transmit as many channels as their corresponding universe are configured for.
  - Feature KiNET v1 and v2 protocols can now be received, captured, output and/or converted to other protocols.
  - Feature KiNET v2 protocol now supports Chromasic fixtures and synchronization of multiple universes.

  - Bug Addressed a problem that could cause incoming sACN priorities to not properly block lower priority input.
  - Bug Addressed a problem that allowed broadcast Art-Net output to be received by the same CueServer, possibly creating a channel value loop.
  - Bug Addressed a problem that would cause CueServer to drop out of real-time while recording a high-bandwidth streaming cue.
  - Bug Addressed a problem with the *Settings > DMX > DMX Ports* panel that could show a warning about the wrong universe chosen for ports that are turned off.
  - Bug Addressed a problem that caused the RELEASE command to not properly terminate streaming cues in certain circumstances.
  - Bug Addressed a problem that caused DMX signal loss events to occur just after the DMX input buffer was cleared instead of just before.

- **LCD and Front-Panel Display**
  - Feature Added optional CPU Load, IO Status, and Timecode displays to the LCD idle screen.
  - Feature The brightness curves for the built-in function button indicators LEDs have been adjusted to produce a more linear visual response.
  - Bug Improved performance.

- **Show Database**
  - Bug Addressed a problem that could occur if a copy of the show directory was manipulated by the Finder on Mac OS and then re-uploaded into the CueServer.

- **Web API**
  - Feature Added additional variables to the ‘in’ and ‘out’ get.cgi API functions to limit the scope of the returned result.
  - Bug The correct indicator colors are now properly sent to the CuePad app and CueTouch panels.
  - Bug Addressed a problem with the ‘in’ and ‘out’ selectors of the get.cgi API not returning the proper data.
  - Bug Improved performance of the API for multiple clients.

- **Auto-Discovery**
  - Bug Improved the Internet reachability detection algorithm.
  - Bug Addressed a problem that could cause device discovery to not function after the host computer sleeps and then re-awakes.

- **Diagnostic Tools**
  - Feature The *CueStation Network Monitor* now automatically scrolls the event window as events are added to the bottom of the list.

- **Windows Build**
  - Bug Addressed several problem areas that were creating excessive flickering on Microsoft Windows.

  - **Bug** Addressed the problem of pushbutton control height being too small on Windows builds.
  - **Bug** Several popup windows are no longer resizable nor do they have close/minimize buttons.
  - **Bug** Addressed a problem that caused the Helvetica Neue font to be used in several places instead of the Windows theme font.

- **CueServer Studio**
  - **Feature** The first time the application is run, the user is asked to register their copy of the software.
  - **Feature** The splash screen now shows a banner in the top-right corner if the build is a "pre-release" version.
  - **Feature** The user can choose to receive application updates only for public releases, or receive notices about pre-release versions.
  - **Feature** Added new time zones for Chile and generic "Etc" zones (including GMT, UTC, Zulu, etc.).
  - **Bug** Addressed a problem that could cause a crash when network interfaces are changed while the app is open.

- **Firmware**
  - **Special Note** Devices upgraded from firmware version 1.5.5 or earlier that have show configurations set to output KiNET v2 protocol will need to manually update the KiNET v2 settings in *Settings* > *DMX* > *Universe Patch*.
  - **Feature** Firmware updates now show their progress on the LCD Display in addition to within CueServer Studio.

Release v1.5.5 [October 28, 2016]

Version 1.5.5 (10/28/16)

- **CueServer Studio 2**
    - Bug Auto-discovery now works properly if network interfaces are enabled and/or disabled while the app is open.
    - Bug The CueServer device name now appears in the stand-alone Stage and Playbacks windows.
    - Bug Addressed a problem that could cause a crash if the active show is switched while a Playbacks view is visible.
    - Bug Addressed a problem introduced in v1.5.4 that could cause a crash if a newly created cue’s number is changed from the default assigned number before saving the cue for the first time.
    - Bug Addressed a problem that could cause corruption of a station’s configuration when editing a show offline and the station’s name is reduced in size.
    - Bug Addressed a problem that could cause a UI inconsistency and/or a crash when expanding a station’s contents when the station was not previously selected.
    - Bug Addressed a problem introduced in v1.5.3 that could cause undefined variables to substitute unexpected values into a CueScript statement.
    - Feature Application crashes are now handled by a built-in error reporting mechanism.
- **Firmware**
    - Feature Added the AT ? syntax to the AT command to query the value of various selectable objects.

Release v1.5.4 [September 8, 2016]

Version 1.5.4 (9/8/16)

- **CueServer Studio 2**
  - Bug Addressed a problem that could cause a streaming cue to become corrupted if its cue number is changed.
  - Bug Addressed a problem that caused offline shows to not show cue stacks in the hierarchical menu below the *Cues* item.
  - Bug Addressed a problem with the *sACN Network Monitor* that could cause it to show an incorrect RGB color in the last channel position of the preview area.
- **Firmware**
  - Bug Streaming cue playback now freezes properly when a playback fader is stopped.
  - Bug Front-panel brightness is now properly reset when switching shows.
  - Bug Addressed a problem that caused the show file to not receive a System Power-On event.
  - Bug Addressed an issue that made it not possible for a show to set the LCD brightness when a show was loaded.
  - Bug Addressed a problem that could cause CueServer to become unresponsive when a CS-940 is improperly configured as a CS-900 and one of the DMX Input ports is set to be a DMX Output.

Release v1.5.3 [August 9, 2016]

Version 1.5.3 (8/9/16)

- **New Features**
  - New Added a new *Debug Mode* feature to the System Log. Now various system functions such as button presses, CueScript commands, UDP messages, variable assignments, etc., can be logged to the System Log for general project troubleshooting.
- **Firmware**
  - Feature Variable substitution is now handled by direct textual replacement inline as they are encountered in CueScript statements. This allows variables to be expanded anywhere in a script, and variables may be numbers, strings, or even additional CueScript commands.
  - Feature Added new “Stream Recording Monitor” to the DMX Utilities menu on the LCD Display.
  - Feature Added new “panel.brightness” system variable that adjusts the overall brightness of the function buttons and navigation switch backlighting.
  - Feature Added new “debug.buttons”, “debug.cue”, “debug.cuescript”, “debug.show”, “debug.udp”, and “debug.variables” system variables to enable system logging of various internal events.
  - Bug Fixed a bug that could cause a crash in the *Network Settings* LCD Menu if a displayed IP Address had 14 or more characters.
  - Bug Addressed a problem that prevented some selection commands from accepting nested script statements. For example, “CHANNEL (RANDOM{1,10}) @ FL” previously did not work.
  - Bug Addressed a problem that could cause timers to not fire properly if other timers in the list were disabled.
  - Bug Addressed a problem with the assignment operator that would mistakenly attempt to set a variable value instead of perform an equality operation if the left-hand value was an undeclared variable.
- **Windows Installer**
  - Feature The windows installer now includes both 32-bit and 64-bit Microsoft Visual C++ packages.

Release v1.5.2 [July 25, 2016]

Version 1.5.2 (7/25/16)

- **CueServer Studio 2**
  - Bug Addressed a problem that could cause KiNET v2 parameters to not be saved properly to the show file.
- **Firmware**
  - Feature Now supports Revision D of the CS-940 hardware.

Release v1.5.1 [July 19, 2016]

Version 1.5.1 (7/19/16)

- **New Features**
    - New Added a *System Clock* panel to allow the device’s time, date and various astronomical parameters to be viewed live.
- **CueServer Studio 2**
    - Feature When changing protocols in *Settings* > *DMX* > *Universes*, the appropriate next field is automatically focused for convenience.
    - Bug Fixed a bug in the *Network Settings* window that displayed the wrong mode when changing settings for a device with only a single LAN port.
    - Bug Fixed a bug in the *Sounds* panel that would improperly place an imported file in the web directory if the “+” button was used to add a file.
    - Bug Addressed a problem in the various resource editors (Cues, Groups, Macros, Rules, etc.) that could prevent a new resource from being created if the current resource has unsaved changes.
    - Bug Fixed several bugs in the file browser panel related to the selected item caption, delete button, and directory refreshing.
    - Feature Added additional legal notices as required.
- **Firmware**
    - Feature Added OFFSET and LENGTH commands that are used to manually set the starting point and playback length of a streaming cue.
    - Bug Fixed a bug that caused KiNET v2 IP addresses to have their bytes reversed.
    - Bug Fixed a bug that caused the *Apply License Code* window to improperly indicate that the code had not been accepted.
    - Bug No longer display an error message for having multiple CueServer universes set to receive the same sACN universe.
    - Bug The factory reset function now properly resets the NTP Server parameters.
    - Feature Improved build optimization to reduce resource usage and increase performance.

Release v1.5.0 [June 3, 2016]

Version 1.5.0 (6/3/2016)

- **New Features**
  - New Added DMX Input Triggers.
  - New Added Art-Net Protocol.
  - New New *Playbacks* view now shows DMX Input/Output.
  - New New *Cues* editor panel now has separate tabs for Properties, Contents and Capture.
- **CueServer Studio 2**
  - Feature Improved the *Playbacks* view to show blocks for DMX Input and DMX Output that feature what sources and destinations are flowing into and out of the CueServer via various Ethernet protocols and/or the hardwired DMX ports.
  - Feature Added flowchart-style arrows between the blocks in the *Playbacks* view to visually indicate the direction of data movement through the fade engine.
  - Feature Layer blocks in the *Playbacks* view now dynamically resize as needed.
  - Feature Added an explicit “Add Rule” button to resources that have rules (Buttons, Contacts, Cues, DMX Input Triggers, etc.) instead of using the small, circular “+” button.
  - Feature Improved the Details column of the Triggers listings, including Timers, Rules and DMX Input Triggers.
  - Feature Added flowchart style arrows to the Playbacks view showing the direction of data movement.
  - Feature Added a new *Contents* tab to the Cue Editor panel that shows all of the recorded channels in a cue.
  - Feature Added a new *Capture* tab to the Cue Editor panel that is used to take snapshots and/or record streaming cue data.
  - Feature Added an indicator to the Playbacks view that shows when DMX Input is disabled.
  - Feature Added a visible “Captured” acknowledgement to the *Cues* > *Capture* panel that appears after a snapshot is captured.
  - Feature Streaming cues are now only partially loaded when editing the cue’s properties, vastly improving load and save speeds.
  - Bug Addressed a problem that caused the *CueStation Network Monitor* to fail to capture packets in certain circumstances.
  - Bug Missing show directories are now automatically created if needed.
  - Bug Switched the order of the Input and Output sections of the *Settings* > *DMX* > *Universes* editor panel to follow the conventions used elsewhere in the application.
  - Bug Scroll bars in several windows (such as *Stage*, *Playbacks*, *Contents*, etc.) now have more reasonable paging and step sizes.
  - Bug Fixed a bug that caused the sACN Universe field to loose focus when entering a new value.
  - Bug Fixed a bug that caused the *Settings* > *DMX* scrolling list to return to the top each time a universe configuration was saved.
  - Bug Fixed a bug that caused the default values that would appear in *Settings* > *DMX* >

Universes to be set to unexpected values.
    - Bug Fixed a bug that caused the hour field for on/off timers to not be able to be set to zero after it was set to a non-zero value.
    - Bug Fixed a bug with the *Cue > Properties* panel that was not calculating the scroll bar height correctly when a cue had rules.
    - Bug Fixed a bug that caused the hierarchical control to not show as expended in the Navigator Window when a new show is created on a CueServer device when it’s show listing is collapsed.
    - Bug Fixed a bug that caused a crash if the *Network Info* window was closed shortly after opening it, while it was still checking for a connection to the Internet.
    - Bug Addressed a problem with the Playback view that sometimes caused the bottom part of the display to disappear when the window was resized smaller than its original size.
    - Bug Addressed a problem that could cause the *Groups* editor panel to be very slow when complex groups are shown in the list.
    - Bug Addressed a problem with the mini text field used in rule conditions that allowed non-printing characters to be entered into the field (such as page up/down, arrow keys, forward delete, etc.).
    - Bug Addressed a problem introduced in v1.4.2 that caused device auto-discovery to fail from computers that have multiple active network interfaces in certain circumstances.
    - Bug Addressed a problem introduced in v1.4.3 that prevented remote CueServers from displaying their live status.
    - Bug Addressed a problem that was causing excessive UI flickering in the *Settings > DMX > Universes* editor panels [Windows only].
    - Bug Renamed the Resources sub-directory of the Windows version of the application to include the application’s name [Windows only].
    - Bug Fixed a bug that would cause the installed application to work fine for the user that installed the application but when another user was logged into the same computer the application icon, version information, and other important application data would be missing [OS X only].
- **Firmware**
    - Feature Restructured the show database server for faster show switching.
    - Feature Added better LCD display messages for when the device is rebooting or shutting down.
    - Feature Implemented a fail-safe mechanism during streaming cue recording that prevents runaway stream recording if the CueServer Studio client disappeared because of a network error or program crash.
    - Feature The built-in Hardware Self Test function can now be accessed via the LCD Display menu.
    - Feature Optimized the DMX fade engine code for increased performance in several key areas, especially during stream recording.
    - Feature Optimized the CGI code for increased performance.
    - Feature Switching shows now clears all cached user variables and command contexts.
    - Feature The built-in front-panel function buttons’ default colors are now reset to defaults each time a show is loaded for consistency.
    - Bug Addressed a problem that was preventing Groups from being joined using the THRU

command.

- Bug Fixed a bug that could cause a crash of the fade engine if a streaming cue is updated (instead of recorded) before the streaming cue existed yet.
- Bug Fixed a bug that caused universes to not be able to be disabled if they had no output protocol selected.
- Bug Fixed a bug that caused DMX Input Restore/Fail events for the “DMX 2 Input” port on the CS-940 to be reported as Port 3.
- Bug Addressed a problem that could cause show switching to fire a WAIT command or perform a playback auto-follow in the middle of the switch when the old show is partially unloaded.
- Bug Addressed a problem that was causing a show to not be able to properly set the built-in function button colors in its Show Loaded event during system boot time.
- Bug Addressed a problem that caused a playback fader to show that Cue 0 was active after the RELEASE command was used.
- Bug Addressed several problems with show switching that would cause the show to be restarted whenever DMX settings were changed.
- Bug Addressed a problem that caused newly created shows to have improper default values for some universe settings.
- Bug Addressed a problem that caused disabled buttons, contacts or DMX input to become re-enabled whenever any show configuration settings were changed.
- Bug The Apache web server’s CGI interface is shut down during firmware updates to prevent a race condition that occasionally caused firmware updates to fail.

Release v1.4.3 [April 18, 2016]

[April 18, 2016]h3. Version 1.4.3 (4/18/2016)

- **CueServer Studio 2**
  - Bug Addressed a problem that could cause the discovery of local CueServers to be intermittent if the found devices are not on the same subnet as the host computer.
  - Bug Addressed a problem that could cause a crash when editing certain objects within offline shows.
  - Bug Addressed a problem that could cause CueServer Studio to experience a long delay on startup if the computer is on a network, but the internet is not reachable [OS X only].
  - Bug Addressed a problem that caused the sACN Network Monitor window to flicker while updating and redraw improperly when the window is resized [Windows only].
  - Feature The Windows version of CueServer Studio now uses the Microsoft Universal C Runtime libraries for better compatibility and stability [Windows only].
- **Firmware**
  - Feature In the Network Settings LCD menu, a “null icon” will be displayed with the gateway address if the chosen gateway is not accessible on the local network.
  - Bug Addressed a problem that would cause sACN data to not be transmitted if the chosen gateway address is not accessible on the local network.
  - Bug Addressed a problem that sometimes caused the Network Settings LCD menu to display a gateway address of 0.0.0.0 even though a non-zero gateway address was actually in use.
  - Feature The gateway address can now be explicitly set to 0.0.0.0 to configure the network to have no default gateway.
  - Bug Addressed a problem that could cause the “Contacting DHCP Server” message to get stuck on the LCD Display.
  - Feature The ‘ssh’ and ‘rpcbind’ services are no longer running by default in CueServer for increased security and performance.
  - Bug Addressed a problem that caused network time updates to fail in certain circumstances.
  - Bug Addressed a problem that caused show file modification dates to be written in UTC time instead of local time.
  - Bug Addressed a problem that caused the AT+ or AT- commands from working properly when changing the submaster level of a playback fader.

Release v1.4.2 [March 17, 2016]

Version 1.4.2 (3/17/2016)

- **CueServer Studio 2**
    - Feature Added a new Diagnostic Tools section to the Help menu. Two network analysis tools have been added:
        - **sACN Network Monitor** – Scans the network for active universes of sACN data. Shows the sources of sACN data on a particular network. Visualizes a chosen sACN source by showing the channel values as bar graphs or RGB pixels.
        - **CueStation Network Monitor** – Monitors and captures any CueStation button and/or indicator commands found on the network. Displays (or filters) each event based on event type, source, station or button.
    - Feature When switching to the Stage or Playback views, the command line is now always active.
    - Feature KiNET v2 now allows each port to be assigned it’s own separate IP Address.
    - Bug Addressed a problem that caused a crash when changing the number of buttons on a station while editing an offline show file.
    - Bug Addressed a problem that occurred when renaming the active show file, or a show with an open editor window.
    - Feature Changed the behavior of the Network Settings window to automatically adjust the gateway field when changing the IP Address and/or Subnet Mask to guarantee that the chosen gateway would always be reachable on the network.
    - Bug Addressed a problem with the System Log view on Retina displays that caused the text to appear very small.
    - Bug Addressed a problem that caused the Navigator Window to not become frontmost when right-clicking on the Device or Project lists [OS X Only].
- **Firmware**
    - Feature Added an update routine that periodically refreshes indictor values in case a CueStation Hub misses update messages or is power cycled.
    - Bug Addressed a problem with CueStation Hub communications that occurred when setting many indicator values at once that could cause some indicators to not properly change value.
    - Feature The WRITE command now processes embedded C-style escape characters in the string.
    - Feature Changed the KiNET v2 driver to support separate IP Addresses for each port.
    - Feature Changed the behavior of the LCD Display’s Network Settings to automatically adjust the gateway field when changing the IP Address and/or Subnet Mask to guarantee that the chosen gateway would always be reachable on the network.
    - Bug Addressed a problem on the CS-900 that caused ports set to DMX Outputs to not be configured properly in certain circumstances.
    - Bug Addressed a problem that caused hardwired DMX and sACN inputs on the same universe to not be merged properly.
    - Bug Addressed a small memory leak that could occur when transmitting strings out one of the

serial ports.

- Bug Addressed a problem that caused the network time daemon to not start properly in certain circumstances.
- Bug Adjusted the network time daemon parameters to generate less network traffic.
- Feature Increased compiler optimization to improve overall system performance.

Release v1.4.1 [February 24, 2016]

Version 1.4.1b (2/24/2016)

- **Windows Installer**
    - Bug Addressed a problem with the Windows installer that corrupted the application in a way that would cause it to not be able to properly create new offline show files [Windows Only].

Version 1.4.1 (2/9/2016)

- **CueServer Studio 2**
    - Feature Editor Windows now darken and display a “CueServer Offline” message when the device becomes unreachable.
    - Bug Addressed a problem that would cause cue display problems and a crash in the stream capture window on host computers that are localized to use a comma as the decimal value separator.
    - Bug Addressed a problem that could cause some show files to not appear in the Navigator window when multiple CueServers’ show listings were being displayed at the same time.
    - Bug Addressed a problem that could cause the “Input Status” indicator in the *Settings* > *DMX* > *Universes* panel to show an improper status.
    - Bug Addressed a problem introduced in v1.4.0 that caused a warning icon to appear for the *Settings* > *DMX* navigation item for offline shows.
    - Bug Addressed a problem that caused Blue LED Indicators on the CueServer device to appear Magenta in CueServer Studio.
    - Bug Addressed a problem that caused the Indicators in the *Status* > *Front Panel* to not be labeled properly.
- **Firmware**
    - Feature Temporary internal files are no longer stored in non-volitale memory to reduce wear on flash memory.
    - Feature Changed the default subnet mask after factory reset to 255.0.0.0.
    - Bug Fixed a problem that caused the IP Address to show “No Address” when the Ethernet cable was unplugged, instead of properly showing the IP Address.
    - Special Note Devices running firmware version 1.4.1 (or higher) can not be downgraded to 1.4.0 (or lower) without assistance from Technical Support. Downgrading firmware is not typically necessary. Should a situation arise that requires a firmware downgrade, please contact Technical Support first.

Release v1.4.0 [January 21, 2016]

Version 1.4.0 (1/21/2016)

- **CueServer Studio 2**
    - Feature Added ability to set the Input/Output direction of DMX Ports in the DMX Port Settings panel.
    - Feature Added a new Network Settings window that allows the network mode to be switched between Single and Dual LAN configurations (on hardware that supports this).
    - Feature Added preferred CueServer model selection to DMX Port Settings panel (used to show the appropriate DMX port options).
    - Feature Added a warning to the DMX Resources Settings panel when the allocated number of universes exceeds the licensed number of universes.
    - Feature Added warnings to the DMX Port Settings panel when the port direction or universe number is questionable.
    - Feature Added ability to duplicate Cues, Groups, Macros, Timers, and Rules using the list's settings menu or contextual menu.
    - Feature Added a warning icon that shows when a local CueServer is discovered but is not accessible on the network because it is on a different subnet.
    - Feature Added the ability to specify what ranges of DMX channels are broadcast to each port of a KiNET v2 device.
    - Feature CueServer devices now show “A” or “B” after their IP Address if the device has Dual-LANs enabled to indicate which LAN the device is connected via.
    - Feature The Macro editor now fills the available window space with the script editor.
    - Feature The “create new show” dialog window now opens with the text field selected.
    - Feature Added buttons to the System Log panel that will add marks or text to the log file.
    - Feature Show files in the Navigator Window are now listed in alphabetical order.
    - Feature Devices that go offline in the Navigator Window now collapse their show listing, loose their hierarchical control and cannot be edited.
    - Feature CueServer 1 models now appear in the device list in gray text.
    - Feature Opening a CueServer 1 model now provides a warning that CueServer Studio 2 cannot edit older CueServer models.
    - Feature The Device and Project lists are now sorted by name by default.
    - Feature Improved the Show Network Info window to display if the Internet is reachable from the local machine.
    - Feature New shows files now default to not transmitting or receiving DMX over Ethernet.
    - Feature The DMX Settings sub-panels are now scrollable when the content is larger than the parent window.
    - Feature The Universe Settings sub-panel now shows only the properties that apply to the selected input/output protocol.
    - Bug Addressed a problem that could cause the renumbering of resources to fail.
    - Bug Addressed a problem that caused a crash when offline show files contained more than about 375 of a single resource (such as Groups, Macros, Cues, etc.).

  - Bug Fixed a bug that caused active show changes to remote CueServers to not be updated properly in the Navigator Window.
  - Bug Fixed a bug that could cause the DMX Resource Settings bar graphs to draw improperly after user changes were reverted.
  - Bug Addressed a problem that could cause unwanted settings to appear in the universe settings after increasing the number of universes in a project.
  - Bug Fixed a bug that would cause the Done button in the License Code Window to redraw incorrectly in certain circumstances.
  - Bug Addressed a problem that caused remote CueServers to not show their online status properly.
  - Bug Addressed a problem that could cause the time to not be able to be updated on a CueServer if the host computer had more than one active network interface.
  - Bug Fixed a bug that caused the application to experience a long (30 second) delay when launching on a machine that was not connected to the Internet.
- **Firmware**
  - Feature Added support for the CS-900 bi-directional DMX port hardware. Now, each DMX port can be switched to either an input or an output (or disabled).
  - Feature Added support for the CS-900 Dual-LAN hardware. Now, CueServer networking can run in one of two modes, (1) a single LAN that carries both lighting and management data combined with a built-in Ethernet switch between the two physical ports, or (2) dual separate LANs with management data on LAN A and lighting data on LAN B.
  - Feature Improved DMX port LED indicators to show additional port status information.
  - Feature Improved the scroll speed of adjusting LCD menu items.
  - Feature Whenever the system contacts a DHCP server, it is shown on the LCD display.
  - Feature Added the ON and OFF keywords to the TOGGLE command.
  - Feature Added ability for CueServer’s Ethernet ports to be “hot plugged” while the device is running.
  - Feature The LCD display now shows a small “null” icon next to the device’s IP address if there is no active link on that port.
  - Feature Added additional power-on hardware test for FPGA bitstream.
  - Feature A new DHCP fallback address of 10.0.1.234 has been set for the primary interface, and the secondary interface (if present) will fallback to 192.168.1.234.
  - Feature Changed the DHCP function to timeout more quickly if a DHCP server could not be reached.
  - Bug Fixed a bug that caused streaming cues inside of cue stacks to not record properly.
  - Bug Fixed a bug that caused KiNET v2 to not be received properly by some CK power supplies.
  - Bug Addressed a problem that could cause the timer and trigger daemons to crash if no show file is loaded.
  - Bug Fixed a problem that caused the csportd and csfpuid daemons to report that they were improperly killed during a firmware update.
  - Bug Addressed a problem that could cause cue numbers with decimal points to fail to record properly in certain circumstances.
  - Bug Fixed the warning during the update of apache configuration during a firmware update.

  - Feature Improved the factory initialization routine.
  - Bug Addressed a problem that caused a failed assertion when a web client was requesting channel values and no show was loaded.
  - Bug Addressed a problem that could cause a memory leak during an NTP lookup.
  - Bug Addressed a problem that could cause networking to fail if the device is connected directly to a laptop without a router.
  - Feature The LCD display now shows additional information about the boot process.
- **Kernel**
  - Feature Updated to CueServer Kernel v1.3.
  - Feature Supports booting of CS-900 Hardware.
  - Feature Supports auto-update of FPGA bitstream.

Release v1.3.0 [November 11, 2015]

Version 1.3.0 (11/3/2015)

- **CueServer Studio 2**
    - **Feature** Macros editor now has an inline script editor, instead of requiring the user to click into a separate script editor window.
    - **Feature** The “New…” menu item in the File menu now creates a new resource or trigger of the currently selected type.
    - **Feature** Enabled Cut/Copy/Paste/Clear menu commands for the command line.
    - **Feature** Added “Command Line” (Command-L) menu option to move keyboard focus to command line.
    - **Feature** The command line now has focus when opening a new Editor Window.
    - **Feature** Changing navigation pages or clicking on editor panels no longer removes keyboard focus from the command line.
    - **Feature** Added user-selectable serial port protocols.
    - **Feature** Added local echo option to serial ports.
    - **Bug** Fixed a bug that prevented the Capture window of a cue to not operate properly when a Cue Stack is selected.
    - **Bug** Fixed a bug that prevented decimal cue number from being entered, introduced in v1.2.0.
    - **Bug** Fixed a bug that caused the Output and Port panels to not clear properly when an output or port component of a station was deselected.
    - **Bug** Fixed a bug that caused the Open Web button to do nothing if the active show is selected in the Navigator Window.
    - **Feature** Windows: Add ability for Delete key to perform same actions as Backspace key.
- **Firmware**
    - **Feature** Added initial hardware support for new CS-900 model.
    - **Feature** Added “Macros” LCD menu item to manually trigger macros.
    - **Feature** Added “Shows” LCD menu item to manually switch between shows.
    - **Feature** Added parsing module for incoming serial data as CueScript commands in either CueServer 1 compatible format or new CR/LF format.
    - **Feature** Added syntax to AT command to handle +/- delta values.
    - **Feature** Added comparison operators !=, >= and <=.
    - **Bug** Playing audio clips are now halted when switching shows.
    - **Bug** LCD Display overrides are now cleared when switching shows.
    - **Bug** Addressed a problem with certain sACN alternate start codes being improperly recognized as dimmer levels.
    - **Bug** Addressed a problem with AT command that prevented it from properly handling array values.
    - **Bug** Addressed a problem that caused the live fade countdown indicator from appearing properly during some fades.
    - **Bug** Addressed a problem with serial port baud rate and character format not changing when switching shows.

- Bug Addressed a problem that could cause the front-panel display to become unresponsive.
- Bug Addressed a problem that caused improperly formed IF/THEN statements to cause the system to become unresponsive.

Release v1.2.0 [July 24, 2015]

Version 1.2.0 (7/24/2015)

- **CueServer Studio 2**
    - Feature Added entirely new way to create cues and to capture scenes and/or streams.
    - Feature Added stream recording trigger channel and recording duration parameters.
    - Feature Added new record modes for capturing scenes.
    - Feature Added ability to create and edit cues offline.
    - Feature Added new “Button Held for n Seconds” rule.
    - Feature Added the ability to test scripts from within rule definitions.
    - Feature Improved the end-of-stream action choices (None, Loop, Follow, Release).
    - Feature Improved fields and controls available for various editors when the show is online/active/offline.
    - Feature The delete key now removes the selected Cue, Macro, Group, Station, Timer or Rule; holding down the Option/Alt key avoids the warning dialog.
    - Bug Fixed a bug that caused remote devices appear offline if their connection was interrupted and restored.
    - Bug Fixed a bug that could cause the drop target of a show upload to remain highlighted even if the drop did not occur.
    - Bug Fixed a bug that could cause the selected button/contact/output to change selection when the apply button was pressed.
    - Bug Fixed a bug that displayed a cue’s rule list in the editor panel after a cue was deselected.
    - Bug Fixed a bug that caused a crash if viewing a streaming cue with a length of zero.
    - Bug Fixed a bug with trimers set to operate between two dates that would cause the timer to not fire in certain circumstances.
    - Bug Windows: Addressed a problem that caused some displays to flicker and/or display incorrect information in certain circumstances.
    - Bug Windows: Addressed a problem that could cause show file uploads/downloads to fail.
- **Firmware**
    - Feature Added ACTIVE, ALL, CHANNEL n, INPUT, PLAYBACK n, and TIME n options to the RECORD and UPDATE commands.
    - Feature Added UPDATE STREAM syntax.
    - Feature Added ability to convert normal cues to streaming cues and vice versa.
    - Feature Added ability to send UDP messages using the WRITE command.
    - Feature Adjusted the value returned by the Indicator command for external button stations.
    - Bug Fixed a bug that caused the DIO-588 interface to not trigger it’s contact rules.
    - Bug Fixed a bug that caused the DIO-588 to attempt to use default RGB indicator colors to turn on/off its outputs.
    - Bug Fixed a bug that caused external button station events to trigger their events twice.
    - Bug Fixed a bug with the TOGGLE command when operating on indictors of external button stations.
    - Bug Fixed a bug that prevented streams from being recorded into cue stacks.

- **Bug** Fixed a bug that would cause a single frame of stale DMX Input data to pass through the fade engine after DMX Input was disabled and then re-enabled.
- **Bug** Addressed a problem that caused jitter in the DMX Input stream coming from the built-in DMX ports.

Release v1.1.0 [May 22, 2015]

Version 1.1.0 (5/22/2015)

- **CueServer Studio 2**
  - Feature Added KiNET v1 and v2 protocols.
  - Feature Added the ability to rename shows.
  - Feature Removed ambiguous “Name” field from General Settings.
  - Bug Fixed a problem with the “Set Time Now” button in the Clock dialog.
  - Bug Fixed a bug introduced in 1.0.8 that prevented editing of remote devices that have custom port numbers.
  - Bug Fixed a bug that would cause the stand-alone Stage and Playbacks windows to not be able to be reopened after closing them.
  - Bug Adjusted the rendering of LED Indicators on OS X to eliminate color bleeding.
  - Feature Improved contextual menus in the Navigator window.
- **Firmware**
  - Feature Added support for newly updated CuePad app (v2.2).
  - Feature Added support for interactive web content.
  - Feature Added KiNET v1 and v2 protocol output for Philips/Color Kinetics fixtures.
  - Feature Added audio.volume system variable for adjusting the audio output level.
  - Bug Adjusted the default stereo audio line-out level.
  - Feature Improved the switching behavior between manual and automatic time adjustment.
  - Bug Fixed a bug that could cause the real-time clock from being properly updated.
  - Bug Improved the DMX fade engine’s timebase to be immune from accumulated drift.
  - Bug Addressed a problem that could cause the front-panel user-interface to crash when the NTP daemon receives a time update.
  - Bug Addressed a problem that could cause Show Loaded/Unloaded events to not be triggered.
  - Feature Implemented group query in the get.cgi API.
  - Bug Fixed a bug that could cause the CueScript parser to crash when the length of the result string was certain multiples of 8 bytes long.
  - Bug Addressed a problem that could cause overlapping audio clips to not properly terminate.
  - Feature Rolled the LCD display driver into the UI server for better performance.

Release v1.0.8 [April 27, 2015]

Version 1.0.8 (4/27/2015)

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

of timer scheduling.

  - Bug Windows: Fixed a problem with offline show paths appearing with slashes instead of backslashes.
  - Bug Windows: Fixed a problem that prevented Drag & Drop to the Web and Sound file browsers from the Desktop.
  - Bug Windows: Fixed a problem with field validation that sometimes caused the insertion point to move unexpectedly.
  - Bug Windows: Fixed a problem in the Specific Month and Year dialogs that would cause the checkboxes to not draw properly in certain circumstances.
  - Bug Windows: Fixed a problem where the System Log may not display the entire log file.
- **Firmware**
  - Feature Added the INPUT ENABLE/DISABLE syntax to enable or disable the DMX Input layer of the playback stack.
  - Feature Added automatic updating of playback fader user preferences for combine modes when loading or switching shows.
  - Feature 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.).
  - Feature Added the ability to query variable values to the get.cgi API.
  - Bug Addressed a problem with CueScript parsing timeouts being raised too quickly.
  - Bug Fixed an issue that would cause the CueScript parser to erroneously report itself as shut down after executing a WAIT command.
  - Bug Fixed a problem introduced in 1.0.7 that caused the WAIT CLEAR command to raise an exception.
  - Bug Fixed a problem where incomplete CueScript strings would silently fail without reporting an error.
  - Bug Fixed a problem that could cause CueStations to not respond after switching active shows.
  - Bug Addressed a problem that could cause timers to fail to trigger that are set for “nth weekday of the month” in certain circumstances.
  - Feature Improved error descriptions for unrecognized commands.

Release v1.0.7 [April 7, 2015]

Version 1.0.7 (4/7/2015)

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

- **Bug** Addressed a problem that could prevent setting of static IP address parameters via the LCD Menu.
- **Bug** Addressed a problem that would cause the device to not be discoverable when booted on a network without a router.
- **Bug** Addressed several issues with the get.cgi API for compatibility with the CuePad iOS app. CueServer 2 requires CuePad v2.2 or greater.
- **Bug** Addressed a problem that prevented CueScript commands to be able to be unicast to the CueServer’s IP Address.
- **Feature** Added additional error checking and reporting to the various daemon processes.

Release v1.0.6 [March 13, 2015]

Version 1.0.6 (3/13/2015)

- **CueServer Studio 2**
    - Bug 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**
    - Bug Addressed a problem introduced in 1.0.5 that caused buttons and contacts on external stations to not trigger properly.

Release v1.0.5 [March 11, 2015]

Version 1.0.5 (3/11/2015)

- **CueServer Studio 2**
  - Feature Show project files can now be downloaded/uploaded to/from your computer.
  - Feature Offline project files can now be edited without the CueServer hardware.
  - Feature Added a second list view to the main Navigator window to make is easier to work with offline project files.
  - Feature Added the ability to create new projects from within CueServer Studio.
  - Feature Drag and Drop has been added to move show project files between a CueServer and your computer.
  - Feature Added Network Settings window to remotely change a CueServer’s network settings.
  - Feature Added Time Settings window to set manual or automatic time settings, including definitions for over 400 time zones.
  - Feature A warning dialog now appears when you try to edit a CueServer that has outdated firmware.
  - Feature Changed CueScript buttons to show newlines as semicolons (;) to be more consistent with syntax rules.
  - Bug Addressed a problem that caused device discovery to only work on the host’s default Ethernet interface.
  - Feature Windows: Enabled the main window’s close box.
  - Bug Windows: Reduced the flickering of the Playback and Status panels.
  - Bug Windows: Addressed a problem that would cause CueScript buttons to not display multiline text properly.
  - Bug Windows: Addressed a problem that caused the Control-C shortcut for “Copy” to not work properly in the CueScript popup editor window.
  - Bug Windows: Addressed a problem that caused the popup menu controls in the Stage view to display incorrect labels.
  - Bug Windows: Addressed a problem that would cause the main window to not open properly if reopened after the app was closed while minimized.
- **Firmware**
  - Feature Added the ability for Buttons and Contacts to be enabled/disabled.
  - Feature Implemented the Update Cue and Update Group commands.
  - Feature Pressing and holding the Up/Down navigation buttons while editing values on the LCD Display now continuously adjusts the value.
  - Feature Changed the behavior of the “=” command to dynamically act as either Assign or Equals, depending on the context of the parameters.
  - Bug Addressed a problem that caused sACN to not receive data properly from certain consoles.
  - Feature Improved sACN receive logic to deal with transmitters that do not properly terminate
  - Bug Addressed problems with setting the Network Settings using the LCD Display that would cause unexpected results.

  - **Bug** Addressed a problem with the Fade and Time commands that caused them to not be able to receive their values from variables.
  - **Bug** Addressed a problem with the LCD Display that could cause it to freeze if the system time was adjusted in certain circumstances.
  - **Bug** Addressed a problem with the sACN protocol not properly supplying a valid CID field for transmit packets.
  - **Feature** Added additional network diagnostics support to csctl.

Release v1.0.4 [February 9, 2015]

Version 1.0.4 (2/9/2015)

- **CueServer Studio 2**
  - Feature Added the ability to open local offline show files.
  - Feature Added an Cue Fade Times popup window that provides direct access to extended fade time attributes.
  - Feature Added cursor and history control to the command line using the up/down/right/left arrow keys.
  - Feature Added contextual menu to Web Pages that includes option to open files in the user’s web browser.
  - Feature Added new rule conditions for testing indicators and outputs.
  - Feature Added a watermark that appears when no editor panel is selected.
  - Feature Added support for Rev. B hardware.
  - Feature The navigator window now remembers it’s preferred size and column widths.
  - Bug Improved the description and input validation of the Add Remote CueServer window.
  - Bug 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.
  - Bug Addressed a cosmetic problem with the Month/Day/Year popup menus in the Active Days section of the Timers panel.
  - Bug Addressed a problem with Timers set to trigger between two dates that would cause the Weekdays field to have an invalid default value.
  - Bug Addressed a problem that caused the default value of the Sun Brightness Rule Condition to be undefined.
  - Bug Addressed a problem that could cause a crash if a CueScript popup editor button was double-clicked.
  - Bug Windows: Fixed the titles of several dialog box windows.
  - Bug Windows: Select entire text field when entering the Rename Cue Stack window.
  - Bug Windows: Change the Sounds and Web Pages panels to display file paths with the correct path delimiters.
- **Firmware**
  - Feature 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.
  - Feature Added a new default index.shtml file to the Web Resources of new shows.
  - Feature Exported several show and device related environment variables in the Apache virtualhost for use with web pages’ CGI and SSI scripting.
  - Bug Addressed a problem with switching shows not causing Apache to properly switch the served web pages.
  - Bug Addressed a problem that caused the Fade command to fail to modify the next cue’s execution time in certain circumstances.
  - Bug Addressed a problem that caused the Toggle command to not work with button indicators or digital outputs.

- Bug Addressed a problem that caused shows with spaces or other special characters in their name to not be able to be deleted.
- Bug Addressed a problem that caused Telnet sessions to hang in certain situations.

Release v1.0.3 [January 22, 2015]

Version 1.0.3 (1/22/2015)

- **CueServer Studio 2**
  - **Feature** First version available as both OS X and Windows builds.
  - **Feature** Added the ability to add and remove remote devices to the CueServer Navigator window.
  - **Feature** Added firmware version column to the Navigator window.
  - **Feature** Added a built-in firmware image that matches the release version of Studio.
  - **Bug** Addressed a problem that caused the command line text to appear very small on Retina displays.
  - **Bug** Addressed a problem that could allow automatic text substitutions to occur in the script editor popup window.
  - **Bug** Addressed a problem that would cause the app to not launch properly if the splash screen was clicked.
- **Firmware**
  - **Bug** Addressed a problem that caused remote devices to not return the correct ping data via HTTP.

Release v1.0.2 [January 9, 2015]

Version 1.0.2 (1/9/2015)

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

Release v1.0.1 [December 23, 2014]

Version 1.0.1 (12/23/2014)

- **CueServer Studio 2**
  - Feature Added “User’s Manual…”, “Support Website…”, and “Release Notes…” to the Help Menu.
  - Feature Changed the font used for displaying CueScript commands.
  - Bug 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.
  - Bug Addressed a problem with the *Open Web* command that could cause the web page to not open properly.
- **Firmware**
  - Feature Changed the LCD Menu display for System Information.
  - Bug Addressed a problem that could cause show data to not be synchronized with the memory card.
  - Feature Improved error reporting for I2C Bus Daemon.

Release v1.0.0 [December 18, 2014]

Version 1.0.0 (12/18/2014)

- First public release version.
- All versions prior to v1.0.0 were private.

Legal Notices

Acknowledgements:

Portions of CueServer Studio and/or CueServer 2 Firmware may utilize the following copyrighted material, the use of which is hereby acknowledged.

- **alsa-lib**
  Parts of this software include the alsa-lib libraries (alsa-project.org). Under terms of the GNU Lesser General Public License (LGPLv2.1), this package's corresponding source code is included alongside the compiled binary within the filesystem of this product. Complete instructions for accessing and re-compiling this software will be provided upon written request to Interactive. This software is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details; a copy of the LGPLv2.1 is included below.

  GNU LESSER GENERAL PUBLIC LICENSE
  Version 2.1, February 1999
  Copyright © 1991, 1999 Free Software Foundation, Inc.
  51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
  Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.
  [This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]
  Preamble
  The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users.
  This license, the Lesser General Public License, applies to some specially designated software packages—typically libraries—of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.
  When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.
  To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or

to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author’s reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the “Lesser” General Public License because it does Less to protect the user’s freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users’ freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a “work based on the library” and a “work that uses the library”. The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called “this License”). Each licensee is addressed as “you”.

A “library” means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables. The “Library”, below, refers to any such software library or work which has been distributed under these terms. A “work based on the Library” means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term “modification”.)

“Source code” for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library’s complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) The modified work must itself be a software library.

b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a “work that uses the Library”. Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a “work that uses the Library” with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a “work that uses the library”. The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a “work that uses the Library” uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a “work that uses the Library” with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer’s own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable “work that uses the Library”, as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user’s computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the “work that uses the Library” must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients’ exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may

add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and “any later version”, you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

one line to give the library’s name and an idea of what it does.

Copyright © year name of author
This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.
This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.
You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Also add information on how to contact you by electronic and paper mail.
You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:
Yoyodyne, Inc., hereby disclaims all copyright interest in the library `Frob' (a library for tweaking knobs) written by James Random Hacker.
signature of Ty Coon, 1 April 1990
Ty Coon, President of Vice
That's all there is to it!

- **Apache**
Apache License Version 2.0, January 2004
TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION
1. Definitions.
"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.
"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.
"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.
"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.
"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.
"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.
"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).
"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other

modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

“Contribution” shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, “submitted” means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as “Not a Contribution.”

“Contributor” shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

You must give any other recipients of the Work or Derivative Works a copy of this License; and

You must cause any modified files to carry prominent notices stating that You changed the files; and

You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

If the Work includes a “NOTICE” text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: HOW TO APPLY THE APACHE LICENSE TO YOUR WORK

To apply the Apache License to your work, attach the following boilerplate notice, with the fields enclosed by brackets “[]” replaced with your own identifying information. (Don’t include the brackets!) The text should be enclosed in the appropriate comment syntax for the file format. We also recommend that a file or class name and description of purpose be included on the same “printed page” as the copyright notice for easier identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the “License”);

- **Art-Net™**
  Art-Net™ Designed by and Copyright Artistic Licence Holdings Ltd. Information about Art-Net can be found at art-net.org.uk.

- **CGIC**
  CGIC, copyright 1996-2011 by Thomas Boutell and Boutell.Com, Inc (boutell.com). Permission is granted to use CGIC in any application, commercial or noncommercial, at no cost. HOWEVER, this copyright paragraph must appear on a “credits” page accessible in the public online and offline documentation of the program. Modified versions of the CGIC library should not be distributed without the attachment of a clear statement regarding the author of the modifications, and this notice may in no case be removed. Modifications may also be submitted to the author for inclusion in the main CGIC distribution.

- **CZMQ**
  Parts of this software include the CZMQ libraries (czmq.zeromq.org). This software used under license of the GNU Lesser General Public License (LGPLv3) with Static Link Exception. A copy of the LGPLv3 is included below.
  STATIC LINK EXCEPTION
  As a special exception, the Authors give you permission to link this library with independent modules to produce an executable, regardless of the license terms of these independent modules, and to copy and distribute the resulting executable under terms of your choice, provided that you also meet, for each linked independent module, the terms and conditions of the license of that module. An independent module is a module which is not derived from or based on this library. If you modify this library, you must extend this exception to your version of the library.
  Note: this exception relieves you of any obligations under sections 4 and 5 of this license, and section 6 of the GNU General Public License.

  GNU LESSER GENERAL PUBLIC LICENSE
  Version 3, 29 June 2007
  Copyright © 2007 Free Software Foundation, Inc.
  Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.
  This version of the GNU Lesser General Public License incorporates the terms and conditions of version 3 of the GNU General Public License, supplemented by the additional permissions listed below.
  0. Additional Definitions.

As used herein, “this License” refers to version 3 of the GNU Lesser General Public License, and the “GNU GPL” refers to version 3 of the GNU General Public License.

“The Library” refers to a covered work governed by this License, other than an Application or a Combined Work as defined below.

An “Application” is any work that makes use of an interface provided by the Library, but which is not otherwise based on the Library. Defining a subclass of a class defined by the Library is deemed a mode of using an interface provided by the Library.

A “Combined Work” is a work produced by combining or linking an Application with the Library. The particular version of the Library with which the Combined Work was made is also called the “Linked Version”.

The “Minimal Corresponding Source” for a Combined Work means the Corresponding Source for the Combined Work, excluding any source code for portions of the Combined Work that, considered in isolation, are based on the Application, and not on the Linked Version.

The “Corresponding Application Code” for a Combined Work means the object code and/or source code for the Application, including any data and utility programs needed for reproducing the Combined Work from the Application, but excluding the System Libraries of the Combined Work.

1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License without being bound by section 3 of the GNU GPL.

2. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or

b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

3. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License.

b) Accompany the object code with a copy of the GNU GPL and this license document.

4. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.

b) Accompany the Combined Work with a copy of the GNU GPL and this license document.

c) For a Combined Work that displays copyright notices during execution, include the copyright notice

for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document.
d) Do one of the following:
0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.
1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time a copy of the Library already present on the user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version.
e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)
5. Combined Libraries.
You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:
a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.
b) Give prominent notice with the combined library that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.
6. Revised Versions of the GNU Lesser General Public License.
The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.
Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.
If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.

- **Fat Icons Pack**
  Fat Icons icon pack made by Designerz Base from flaticon.com is licensed by CC 3.0 BY.

- **Freepik Icons**

Some icons included in this software are made by Freepik from flaticon.com is licensed by CC 3.0 BY.

- **Justicons Icon Pack**
  Justicons icon pack made by Rami McMin from flaticon.com is licensed by CC 3.0 BY.

- **LibConfig**
  Parts of this software include the LibConfig libraries owned by Hyperrealm. Under terms of the GNU Lesser General Public License (LGPLv2.1), this package's corresponding source code is included alongside the compiled binary within the filesystem of this product. Complete instructions for accessing and re-compiling this software will be provided upon written request to Interactive. This software is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.
  [Interactive Note: For a copy of the GNU Lesser General Public License Version 2.1, please refer to the alas-lib attribution, where a copy of the license is included.]

- **ZeroMQ**
  Parts of this software include the ZeroMQ libraries (zeromq.org). This software used under license of the GNU Lesser General Public License (LGPLv3) with Static Link Exception. [Interactive Note: For a copy of the GNU Lesser General Public License Version 3, please refer to the czmq attribution, where a copy of the license is included.]
  SPECIAL EXCEPTION GRANTED BY COPYRIGHT HOLDERS
  As a special exception, copyright holders give you permission to link this library with independent modules to produce an executable, regardless of the license terms of these independent modules, and to copy and distribute the resulting executable under terms of your choice, provided that you also meet, for each linked independent module, the terms and conditions of the license of that module. An independent module is a module which is not derived from or based on this library. If you modify this library, you must extend this exception to your version of the library.