

MPV plugin that triggers cues with OSC

This is MPV player plugin that can **send triggers via OSC**, and **receive OSC messages to remote control** the player.

Introduction

This plugin sends Open Sound Control (OSC) triggers based on a list of timecodes for a video or audio file. These OSC triggers can be used to trigger cues, scenes, chasers, etc. on a lighting console or other show control software like QLab.

The timecode list contains time markers and a OSC path to call when the marker is reached.

Additionally the plugin sends OSC messages on playout events like start, pause, etc. These are the OSC paths for the supported events:

- /mpv/pause
- /mpv/unpause
- /mpv/start
- /mpv/end

You can also remote control the player by sending OSC messages to the plugin.

Understanding trial mode

When you first download and install the plugin, the plugin will run in trial mode. In this unlicensed mode, there is a maximum of 5 trigger points and maximum of 10 OSC that will be send messages in total.

To unlock the full feature set purchase a license key. After you added to license key and the name on your license to the configuration file, the plugin will run in licensed mode.

A word of caution

Don't rely on this plugin in mission critical situations.

There is no guarantee that triggers will be delivered on time or delivered at all.

Reliability depends on multiple factors:

- CPU load. When your computer or MPV player is busy doing other stuff, triggers might be missed or delayed.
- Network latency. A slow network connection can mean triggers are being delivered late.
- Network reliability. OSC triggers are being sent via UDP, which is a connectionless protocol (unlike TCP). UDP doesn't guarantee packet delivery simply because it doesn't confirm if data packets arrive at the destination or not.

- Although every effort has been made to deliver a high quality plugin, there can be unforeseen issues or bugs that prevent triggers to be delivered in certain situations.

Installing the plugin

First, install MPV player if you have not done that already.

`mpv-osc-cues-plugin` is a MPV C plugin. C plugins are put into the `mpv scripts` directory in its config directory. They must have a `.so` or `.dll` file extension. They can also be explicitly loaded with the `--script` option.

Usually, the location of your MPV script directory is as follows:

- macOS/Linux: `~/.config/mpv/scripts`
- Windows: `%APPDATA%/mpv/scripts/`

To install this plugin, copy the plugin binary to you mpv script directory. Depending on your platform this is the `.so` file (macOS) or `.dll` (Windows) file.

Security policy error (macOS)

Due to macOS security policy, running just any `.so` file is not allowed. You probably encounter an error something along the lines of this when you start MPV with the plugin installed:

```
(..) not valid for use in process: library load disallowed by system policy
```

Not ideal, but a solution is to run this command in your Terminal:

```
$ sudo spctl --master-disable
```

Then open System Settings, and Privacy & Security. On this page scroll down to the tab Security and you will see your `.dylib` file with button open anyway.

This should enable loading binary plugins from the unidentified developers. Please be aware that this relaxes the security & privacy settings of your Mac.

Configuration

This plugin reads settings from a TOML configuration file. The name of the configuration file is `mpv-osc-cues-plugin.toml`. Its default location is the `script-opts` directory inside your mpv config directory.

You can override the default location, and use a custom location for the configuration file, by setting the `osc_cues_config_file` script option:

```
$ mpv --script-opts="osc_cues_config_file=<path to your config file>"
```

The configuration file is not created automatically. You need to create the configuration file yourself. If there is no configuration file, default settings will be used.

Example path on macOS:

```
~/ .config/mpv/script-opts/mpv-osc-cues-plugin.toml
```

Example configuration file contents:

```
# OSC output settings
[osc_out]
host = "127.0.0.1"
port = 8010 # default is 9001

# OSC input settings
[osc_in]
port = 8011 # default is 9002

# License information
[license]
name = "John Doe"
key = "Enter your license key here"
```

Creating and loading a list of timecodes

This plugin takes a list of timecodes stored in a CSV file as input.

- The timecode list defines times at which to send a trigger.
- During playback, the plugin monitors the video's current time.
- When a timecode time is reached, the plugin sends the OSC message to the host and port specified in the configuration file.

Timecodes are specific for a video or audio file. When an audio or video file is loaded in MPV, the plugin tries to find a timecode file with the same name as the loaded file, but with the added extension `.csv`.

For example, if you wish to have a timecode list for a file called `music.mp3`, create a file named `music.mp3.csv`. Put that file in the same directory as the file to be played.

A timecode list CSV has two columns. The first column is the timecode in the format `HH:MM:SS.SS`. Note that the number of seconds can be expressed as a decimal number. The second column is the OSC path to send a trigger to.

Example CSV:

```
00:00:02.2,/trigger/cue1
00:00:07,/trigger/cue2
00:00:12.8,/trigger/cue3
```

This will trigger OSC message `/trigger/cue2` when player position reaches 2.2 seconds.

REAPER CSV import

The plugin is also able to import CSVs with cue-points exported from audio software REAPER.

It autodetects input from REAPER: when first line starts with a # it assumes REAPER as format.

The supported file format looks like this:

```
#,Name,Start
M1,,00:00:02:26
M2,,00:00:06:12
M3,,00:00:11:15
```

- The supported time format is Hours:Minutes:Seconds:Frames (HH:MM:SS:FF). This is same format supported by MADRIX 5, ChamSys MagicQ, etc. REAPER uses its currently set time format during the export process. Make sure to set it to HH:MM:SS:FF before the export.
- If there are more columns, for example for End timecode, those columns will be ignored.
- The assumed framerate is 24 fps. The exported CSV does not include the exported frame rate.
- The first column is used to form an path for the OSC trigger as follows:
/mpv/<name>

For more information on how to export form REAPER, see Avolites support wiki

Using the plugin directly in MPV

When the plugin file is present in the mpv `scripts` directory, you can just start mpv player with an audio or video file and if there is a valid timecode file present, triggers will be sent automatically by the plugin.

```
# macOS
# Command below will send triggers if:
# - myvideo.mp4.csv exists
# - plugin is present in MPV script directory
$ mpv myvideo.mp4

# Windows
> mpv.com myvideo.mp4
```

Using the plugin with MPV frontends

While you can play audio/video directly with mpv, mpv is also used as the underlying player in many other multimedia players, such as IINA. Those players are often called *MPV frontends*.

As long as an MPV frontend supports MPV user scripts, this plugin should work there as well. Although sometimes additional configuration is needed. More information on the MPV frontends that this plugin has been tested with, below.

IINA player

Do the following to enable MPV user scripts in IINA:

- “Settings” → “Advanced” → select “Use config directory”. Make sure that this plugin is in the `script` directory of the config directory.

ImPlay

ImPlay will use its own config dir for mpv by default (`~/Library/Application Support/implay` on macOS).

You can either copy the plugin in ImPlay’s own config dir or use mpv’s config dir instead as follows:

- Settings -> “Use mpv’s config dir”

Remote control the player via OSC

With this plugin you can remotely control the MPV player instance via OSC messages. The plugin runs an OSC server, by default on port 9002. It listens to the following paths:

- `/pause` toggle play/pause
- `/playlist_next` if a playlist is active, go to next in playlist
- `/playlist_prev` if a playlist is active, go to previous in playlist

With the `oscsend` utility (part of `liblo` you can easliy test this functionality:

```
$ oscsend osc.udp://localhost:9002 /pause
```

Logging

The plugin logs messages to standard output. This is visible when starting `mpv` from the command line. Logs are also written to the system logger.

On Apple systems, realtime logs from this plugin can be viewed from the Console application, or from the command line:

```
$ log stream --predicate 'eventMessage contains "eu.underweg.mpv-osc-cues"' --info
```

On Windows, the plugin logs to the Windows Event log.

Known issues

- In IINA player, sometimes the plugin is not able to determine MPV’s current player path. Error: “Reason: can’t get path prop. (-10)”. Cause

unknown, might be an issue with IINA or MPV.

Disclaimer

This software is provided “as is,” and you use the software at your own risk. The author makes no warranties as to performance, merchantability, fitness for a particular purpose, or any other warranties whether expressed or implied. No oral or written communication from or information provided by the author shall create a warranty. Under no circumstances shall the author be liable for direct, indirect, special, incidental, or consequential damages resulting from the use, misuse, or inability to use this software, even if the author has been advised of the possibility of such damages.