
mpv Documentation

Release 0.3.0

Cory Parsons

Aug 07, 2017

Contents

1	The Mpv Object	3
2	Templates	7
2.1	Base	7
2.2	Pure Python Template	8
2.3	PyQt5 Template	9
3	Events	11
3.1	Event	11
3.2	Property	11
3.3	LogMessage	12
3.4	ClientMessage	12
3.5	EndFile	12
4	Enums	13
4.1	EventID	13
4.2	LogLevel	14
4.3	Formats	14
4.4	Errors	15
4.5	End File	15
4.6	Sub Api	16
5	Exceptions	17
6	Indices and tables	19
	Python Module Index	21

Contents:

The Mpv Object

class `mpv.Mpv` (*name=None, options=None, **kwargs*)

Create an MPV instance. Any kwargs given will be passed to mpv as options. The instance must be initialized with `initialize()`.

Parameters

- **name** (*str, optional*) – the *name* argument for `ctypes.CDLL`.
- **options** (*dict, optional*) – dictionary of options to set with `mpv_set_option()`.
- ****kwargs** (*optional*) – options to send to mpv via `mpv_set_option()` before the handle is initialized. Use underscores in place of hyphens.

Raises

- `mpv.LibraryNotLoadedError` – if libmpv can't be loaded.
- `mpv.ApiVersionError` – if the loaded libmpv doesn't meet the minimum requirement.

handle

the mpv handle.

api_version()

Return the api version. see: [Client API Changes](#).

Returns libmpv version (major, minor)

Return type `tuple`

available_properties()

Returns names of properties that can be accessed.

Return type `list`

command(*args)

Send a command to the player. Commands are the same as those used in `input.conf`. see: [Input Commands](#).

Example:

```
mpv.command('loadfile', 'test.mp4', 'replace', 'start=+100,vid=no')
```

Parameters **args* – strings.

command_node (**args*)

Send a command to the player. Commands are the same as those used in `input.conf`. see: [Input Commands](#).

Example:

```
mpv.command_node('loadfile', 'test.mp4', 'replace', {
    'start': '+100',
    'vid': 'no'
})
```

Parameters **args* – arguments in any basic type.

detach_destroy ()

initialize ()

Initialize the mpv instance. This function needs to be called to make full use of the client API

observe_property (*name*, *mpv_format=None*, *reply_userdata=0*)

Get a notification whenever the given property changes.

Parameters

- **name** (*str*) – the name of the property.
- **mpv_format** (*mpv.Format*, optional) – The format of the data.
- **reply_userdata** (*int*, optional) – This will be used for the `mpv_event.reply_userdata` field for the received `MPV_EVENT_PROPERTY_CHANGE` events.

Raises

- `AttributeError` – if the property isn't available.
- `mpv.MpvError`

play (*filename*)

Shortcut for a `loadfile command()`

quit (*code=None*)

Shortcut for a `quit command()`

request_log_messages (*level*)

Enable or disable receiving of log messages.

Parameters **level** (*mpv.LogLevel*) – The log level mpv will use.

seek (*seconds*, *method='relative+exact'*)

Shortcut for a `seek command()`

set_option (*option*, *value*)

Parameters

- **option** (*str*) – Option name. This is the same as on the mpv command line, but without the leading “-”.

- **value** – Option value.

Raises `mpv.MpvError`

terminate_destroy()

unavailable_properties()

Returns names of properties that cannot be accessed.

Return type list

unobserve_property(*reply_userdata*)

Undo `observe_property()`. This will remove all observed properties for which the given number was passed as `reply_userdata` to `observe_property`.

Parameters **reply_userdata** (*int*) – `reply_userdata` that was passed to `observe_property`.

wait_event(*timeout=-1*)

Wait for the next event, or until the timeout expires, or if another thread makes a call to `mpv_wakeup()`. Passing 0 as timeout will never wait, and is suitable for polling.

Parameters **timeout** (*float, optional*) – Timeout in seconds, after which the function returns even if no event was received. A `MPV_EVENT_NONE` is returned on timeout. A value of 0 will disable waiting. Negative values will wait with an infinite timeout.

Returns *Event*

Base

`class mpv.templates.AbstractTemplate`

`before_initialize()`

`on_audio_reconfig()`

`on_chapter_change()`

Deprecated.

`on_client_message(event)`

Parameters `event` (`mpv.events.ClientMessage`) – the event data.

`on_command_reply()`

`on_end_file(event)`

Parameters `event` (`mpv.events.EndFile`) – the event data.

`on_file_loaded()`

`on_get_property_reply(event)`

Parameters `event` (`mpv.events.Property`) – the event data.

`on_idle()`

`on_log_message(event)`

Parameters `event` (`mpv.events.LogMessage`) – the event data.

`on_metadata_update()`

Deprecated.

`on_none()`

`on_pause()`
Deprecated.

`on_playback_restart()`

`on_property_change(event)`

This method is called when the value of an observed property is changed. Reimplement this function in a subclass to handle the different properties.

Parameters `event` (`mpv.events.Property`) – the event data.

`on_queue_overflow()`

`on_script_input_dispatch()`
Deprecated.

`on_seek()`

`on_set_property_reply()`

`on_shutdown()`

`on_start_file()`

`on_tick()`

`on_track_switched()`
Deprecated.

`on_tracks_changed()`
Deprecated.

`on_unpause()`
Deprecated.

`on_video_reconfig()`

Pure Python Template

```
class mpv.templates.MpvTemplate(options=None, observe=None, log_level='info',
                                log_handler=None, **kwargs)
    Bases: AbstractTemplate, Mpv.
```

A Template that can be subclassed. It uses a `threading.Thread` for the event loop.

Parameters

- **options** (`dict`, optional) – dictionary of options to set with `mpv_set_option()`.
- **observe** (`list` of `str`) – a list of properties to be observed.
- **log_level** (`mpv.LogLevel`) – the log level for mpv to use.
- **log_handler** (`callable`) – a function that will be called with the log message as its only argument.
- ****kwargs** (`optional`) – options to set with `mpv_set_option()`.

Raises `mpv.ApiVersionError` – if the loaded libmpv doesn't meet the minimum requirement.

`exit()`

PyQt5 Template

Event

class `mpv.events.Event`

These events are returned by `Mpv.wait_event()`.

event_id

mpv.EventID – the event id.

error

mpv.ErrorCode – This is mainly used for events that are replies to (asynchronous) requests.

reply_userdata

If the event is in reply to a request (made with this API and this API handle), this is set to the `reply_userdata` parameter of the request call. Otherwise, this field is 0.

data

The meaning and contents of the data member depend on the `event_id`. *Property LogMessage ClientMessage EndFile* or None

Property

class `mpv.events.Property`

The event data of a *PROPERTY_CHANGE* event.

name

str – The name of the property.

data

Value of the property. The type is dependent on the property.

LogMessage

class `mpv.events.LogMessage`

The event data of a `LOG_MESSAGE` event.

prefix

str – The module prefix, identifies the sender of the message.

level

str – The log level as string.

text

str – The log message.

ClientMessage

class `mpv.events.ClientMessage`

The event data of a `CLIENT_MESSAGE` event.

args

list – Arbitrary arguments chosen by the sender of the message. What these arguments mean is up to the sender and receiver.

EndFile

class `mpv.events.EndFile`

The event data of a `END_FILE` event.

reason

mpv.EndFileReason

error

mpv.ErrorCode

EventID

```
class mpv.EventID
```

```
    NONE = 0
```

```
    SHUTDOWN = 1
```

```
    LOG_MESSAGE = 2
```

```
    GET_PROPERTY_REPLY = 3
```

```
    SET_PROPERTY_REPLY = 4
```

```
    COMMAND_REPLY = 5
```

```
    START_FILE = 6
```

```
    END_FILE = 7
```

```
    FILE_LOADED = 8
```

```
    TRACKS_CHANGED = 9
```

deprecated – equivalent to using `mpv_observe_property()` on the “track-list” property.

```
    TRACK_SWITCHED = 10
```

deprecated – equivalent to using `mpv_observe_property()` on the “vid”, “aid”, “sid” property.

```
    IDLE = 11
```

```
    PAUSE = 12
```

deprecated – equivalent to using `mpv_observe_property()` on the “pause” property.

```
    UNPAUSE = 13
```

deprecated – equivalent to using `mpv_observe_property()` on the “pause” property.

```
    TICK = 14
```

SCRIPT_INPUT_DISPATCH = 15
deprecated – This event never happens anymore.

CLIENT_MESSAGE = 16

VIDEO_RECONFIG = 17

AUDIO_RECONFIG = 18

METADATA_UPDATE = 19
deprecated – equivalent to using `mpv_observe_property()` on the “metadata” property.

SEEK = 20

PLAYBACK_RESTART = 21

PROPERTY_CHANGE = 22

CHAPTER_CHANGE = 23
deprecated – equivalent to using `mpv_observe_property()` on the “chapter” property.

QUEUE_OVERFLOW = 24

LogLevel

`class mpv.LogLevel`

NONE = ‘no’
disable absolutely all messages.

FATAL = ‘fatal’
critical/aborting errors.

ERROR = ‘error’
simple errors.

WARN = ‘warn’
possible problems.

INFO = ‘info’
informational message.

V = ‘v’
noisy informational message.

DEBUG = ‘debug’
very noisy technical information.

TRACE = ‘trace’
extremely noisy.

Formats

`class mpv.Format`

NONE = 0

STRING = 1

```
OSD_STRING = 2
FLAG = 3
INT64 = 4
DOUBLE = 5
NODE = 6
```

Errors

`class mpv.ErrorCode`

For documentation on these, see `libmpv/client.h`

```
SUCCESS = 0
EVENT_QUEUE_FULL = -1
NOMEM = -2
UNINITIALIZED = -3
INVALID_PARAMETER = -4
OPTION_NOT_FOUND = -5
OPTION_FORMAT = -6
OPTION_ERROR = -7
PROPERTY_NOT_FOUND = -8
PROPERTY_FORMAT = -9
PROPERTY_UNAVAILABLE = -10
PROPERTY_ERROR = -11
COMMAND = -12
LOADING_FAILED = -13
AO_INIT_FAILED = -14
VO_INIT_FAILED = -15
NOTHING_TO_PLAY = -16
UNKNOWN_FORMAT = -17
UNSUPPORTED = -18
NOT_IMPLEMENTED = -19
```

End File

`class mpv.EndFileReason`

```
EOF = 0
STOP = 2
```

```
QUIT = 3
ERROR = 4
REDIRECT = 5
```

Sub Api

`class mpv.SubApi`

This is used for additional APIs that are not strictly part of the core API.

```
MPV_SUB_API_OPENGL_CB = 1
```

exception `mpv.MpvError`

func
callable – The function which caused the exception.

args
list – The arguments to the function.

error_code
mpv.ErrorCode – The error code.

reason
str – A string describing the error.

exception `mpv.LibraryNotLoadedError`

exception `mpv.ApiVersionError`

version
tuple – The version of the library that was loaded.

target
tuple – The required minimum version.

CHAPTER 6

Indices and tables

- `genindex`
- `modindex`
- `search`

m

mpv, [1](#)

A

AbstractTemplate (class in mpv.templates), 7
AO_INIT_FAILED (mpv.ErrorCode attribute), 15
api_version() (mpv.Mpv method), 3
ApiVersionError, 17
args (mpv.ClientMessage attribute), 12
args (mpv.MpvError attribute), 17
AUDIO_RECONFIG (mpv.EventID attribute), 14
available_properties() (mpv.Mpv method), 3

B

before_initialize() (mpv.templates.AbstractTemplate method), 7

C

CHAPTER_CHANGE (mpv.EventID attribute), 14
CLIENT_MESSAGE (mpv.EventID attribute), 14
ClientMessage (class in mpv.events), 12
COMMAND (mpv.ErrorCode attribute), 15
command() (mpv.Mpv method), 3
command_node() (mpv.Mpv method), 4
COMMAND_REPLY (mpv.EventID attribute), 13

D

data (mpv.Event attribute), 11
data (mpv.Property attribute), 11
DEBUG (mpv.LogLevel attribute), 14
detach_destroy() (mpv.Mpv method), 4
DOUBLE (mpv.Format attribute), 15

E

END_FILE (mpv.EventID attribute), 13
EndFile (class in mpv.events), 12
EndFileReason (class in mpv), 15
EOF (mpv.EndFileReason attribute), 15
error (mpv.EndFile attribute), 12
ERROR (mpv.EndFileReason attribute), 16
error (mpv.Event attribute), 11
ERROR (mpv.LogLevel attribute), 14

error_code (mpv.MpvError attribute), 17
ErrorCode (class in mpv), 15
Event (class in mpv.events), 11
event_id (mpv.Event attribute), 11
EVENT_QUEUE_FULL (mpv.ErrorCode attribute), 15
EventID (class in mpv), 13
exit() (mpv.templates.MpvTemplate method), 8

F

FATAL (mpv.LogLevel attribute), 14
FILE_LOADED (mpv.EventID attribute), 13
FLAG (mpv.Format attribute), 15
Format (class in mpv), 14
func (mpv.MpvError attribute), 17

G

GET_PROPERTY_REPLY (mpv.EventID attribute), 13

H

handle (mpv.Mpv attribute), 3

I

IDLE (mpv.EventID attribute), 13
INFO (mpv.LogLevel attribute), 14
initialize() (mpv.Mpv method), 4
INT64 (mpv.Format attribute), 15
INVALID_PARAMETER (mpv.ErrorCode attribute), 15

L

level (mpv.LogMessage attribute), 12
LibraryNotLoadedError, 17
LOADING_FAILED (mpv.ErrorCode attribute), 15
LOG_MESSAGE (mpv.EventID attribute), 13
LogLevel (class in mpv), 14
LogMessage (class in mpv.events), 12

M

METADATA_UPDATE (mpv.EventID attribute), 14
Mpv (class in mpv), 3

mpv (module), 1
 MPV_SUB_API_OPENGL_CB (mpv.SubApi attribute), 16
 MpvError, 17
 MpvTemplate (class in mpv.templates), 8

N

name (mpv.Property attribute), 11
 NODE (mpv.Format attribute), 15
 NOMEM (mpv.ErrorCode attribute), 15
 NONE (mpv.EventID attribute), 13
 NONE (mpv.Format attribute), 14
 NONE (mpv.LogLevel attribute), 14
 NOT_IMPLEMENTED (mpv.ErrorCode attribute), 15
 NOTHING_TO_PLAY (mpv.ErrorCode attribute), 15

O

observe_property() (mpv.Mpv method), 4
 on_audio_reconfig() (mpv.templates.AbstractTemplate method), 7
 on_chapter_change() (mpv.templates.AbstractTemplate method), 7
 on_client_message() (mpv.templates.AbstractTemplate method), 7
 on_command_reply() (mpv.templates.AbstractTemplate method), 7
 on_end_file() (mpv.templates.AbstractTemplate method), 7
 on_file_loaded() (mpv.templates.AbstractTemplate method), 7
 on_get_property_reply() (mpv.templates.AbstractTemplate method), 7
 on_idle() (mpv.templates.AbstractTemplate method), 7
 on_log_message() (mpv.templates.AbstractTemplate method), 7
 on_metadata_update() (mpv.templates.AbstractTemplate method), 7
 on_none() (mpv.templates.AbstractTemplate method), 7
 on_pause() (mpv.templates.AbstractTemplate method), 7
 on_playback_restart() (mpv.templates.AbstractTemplate method), 8
 on_property_change() (mpv.templates.AbstractTemplate method), 8
 on_queue_overflow() (mpv.templates.AbstractTemplate method), 8
 on_script_input_dispatch() (mpv.templates.AbstractTemplate method), 8
 on_seek() (mpv.templates.AbstractTemplate method), 8
 on_set_property_reply() (mpv.templates.AbstractTemplate method), 8
 on_shutdown() (mpv.templates.AbstractTemplate method), 8

on_start_file() (mpv.templates.AbstractTemplate method), 8
 on_tick() (mpv.templates.AbstractTemplate method), 8
 on_track_switched() (mpv.templates.AbstractTemplate method), 8
 on_tracks_changed() (mpv.templates.AbstractTemplate method), 8
 on_unpause() (mpv.templates.AbstractTemplate method), 8
 on_video_reconfig() (mpv.templates.AbstractTemplate method), 8
 OPTION_ERROR (mpv.ErrorCode attribute), 15
 OPTION_FORMAT (mpv.ErrorCode attribute), 15
 OPTION_NOT_FOUND (mpv.ErrorCode attribute), 15
 OSD_STRING (mpv.Format attribute), 14

P

PAUSE (mpv.EventID attribute), 13
 play() (mpv.Mpv method), 4
 PLAYBACK_RESTART (mpv.EventID attribute), 14
 prefix (mpv.LogMessage attribute), 12
 Property (class in mpv.events), 11
 PROPERTY_CHANGE (mpv.EventID attribute), 14
 PROPERTY_ERROR (mpv.ErrorCode attribute), 15
 PROPERTY_FORMAT (mpv.ErrorCode attribute), 15
 PROPERTY_NOT_FOUND (mpv.ErrorCode attribute), 15
 PROPERTY_UNAVAILABLE (mpv.ErrorCode attribute), 15

Q

QUEUE_OVERFLOW (mpv.EventID attribute), 14
 QUIT (mpv.EndFileReason attribute), 15
 quit() (mpv.Mpv method), 4

R

reason (mpv.EndFile attribute), 12
 reason (mpv.MpvError attribute), 17
 REDIRECT (mpv.EndFileReason attribute), 16
 reply_userdata (mpv.Event attribute), 11
 request_log_messages() (mpv.Mpv method), 4

S

SCRIPT_INPUT_DISPATCH (mpv.EventID attribute), 13
 SEEK (mpv.EventID attribute), 14
 seek() (mpv.Mpv method), 4
 set_option() (mpv.Mpv method), 4
 SET_PROPERTY_REPLY (mpv.EventID attribute), 13
 SHUTDOWN (mpv.EventID attribute), 13
 START_FILE (mpv.EventID attribute), 13
 STOP (mpv.EndFileReason attribute), 15
 STRING (mpv.Format attribute), 14

SubApi (class in mpv), [16](#)
SUCCESS (mpv.ErrorCode attribute), [15](#)

T

target (mpv.ApiVersionError attribute), [17](#)
terminate_destroy() (mpv.Mpv method), [5](#)
text (mpv.LogMessage attribute), [12](#)
TICK (mpv.EventID attribute), [13](#)
TRACE (mpv.LogLevel attribute), [14](#)
TRACK_SWITCHED (mpv.EventID attribute), [13](#)
TRACKS_CHANGED (mpv.EventID attribute), [13](#)

U

unavailable_properties() (mpv.Mpv method), [5](#)
UNINITIALIZED (mpv.ErrorCode attribute), [15](#)
UNKNOWN_FORMAT (mpv.ErrorCode attribute), [15](#)
unobserve_property() (mpv.Mpv method), [5](#)
UNPAUSE (mpv.EventID attribute), [13](#)
UNSUPPORTED (mpv.ErrorCode attribute), [15](#)

V

V (mpv.LogLevel attribute), [14](#)
version (mpv.ApiVersionError attribute), [17](#)
VIDEO_RECONFIG (mpv.EventID attribute), [14](#)
VO_INIT_FAILED (mpv.ErrorCode attribute), [15](#)

W

wait_event() (mpv.Mpv method), [5](#)
WARN (mpv.LogLevel attribute), [14](#)