



ANALOG WAY CORE API

API and SDK Documentation

Version: 1.1.2



Analog Way Core API

Core API is a collection of easy to use interfaces for controlling CorePlay devices. This document describes the main REST-like HTTP API.

API can be accessed from `/api/core/v1/` path on the CorePlay device. The path segment `v1` indicates that major version 1 of the API is accessed. Currently, only version 1 exists.

For example, to get playback information, make a request to `http://192.168.2.140/api/core/v1/players/1`.

Semantic versioning is used:

Major:

- Increased when backwards compatibility is broken

Minor:

- Increased when endpoints, fields or values are added in a backwards compatible manner

Patch:

- Increased when change does not affect functionality, such as correcting a human-readable message.
- Increased when bug fix changes the implementation so that it matches the documentation.

Tag:

- Additional version tag, such as `-rc1`
- Present only in preview versions of builds and documents

Backwards compatibility is achieved by a promise:

For a given major version of the API, no documented endpoint or field shall be removed or changed in a way that could cause 3rd party integration to malfunction.

Endpoints, fields and enum values can be added later. For the integrator this means that care should be taken to accept any additional fields or values.

If backwards compatibility must be broken, a new major version of the API is published. Old version may be supported simultaneously.

To ensure proper operation, the client should limit their API requests: Client should not send more than 10 requests per second to the API.

In cases where an error occurs, the Core API returns a relevant HTTP status code along with a JSON error message in the response body. The error message consists of a single object containing a human-readable error field:

```
{
  "error": "Error description"
}
```

OpenAPI specification can be used to automatically generate client libraries for multiple languages such as Java, JavaScript, PHP, Python and Rust. Tools like `openapi-generator-cli` process this specification to create client-side code that includes pre-configured methods for API requests, authentication, and response handling.

How to Generate the API Client Using OpenAPI Generator:

```
java -jar openapi-generator-cli.jar generate -i core-api.yaml -g python -o python
```

How to use the openapi client:

```
import openapi_client
from openapi_client.rest import ApiException
from pprint import pprint
```

```
#Defining the host is optional and defaults to http://192.168.2.140/api/core/v1
#See configuration.py for a list of all supported configuration parameters.
```

```

configuration = openapi_client.Configuration(
host = "http://192.168.2.140/api/core/v1"
)
#Enter a context with an instance of the API client
with openapi_client.ApiClient(configuration) as api_client:
#Create an instance of the API class
api_instance = openapi_client.CoreApi(api_client)

try:
    api_response = api_instance.get_system_information()
    print(&quot;The response of Coreapi-&gt;get_system_information:\n&quot;)
    pprint(api_response)
except ApiException as e:
    print(&quot;Exception when calling Coreapi-&gt;get_system_information: %s\n&quot; % e)

```

More information: <https://www.analogway.com/products/coreplay-solo>
 Contact Info: techsupport@analogway.com
 Version: 1.1.2
 BasePath:/api/core/v1
 Proprietary License
<https://www.analogway.com>

Access

Methods

[Jump to [Models](#)]

Table of Contents

[Core](#)

- [DELETE /show/current](#)
- [GET /show/current](#)
- [GET /version](#)
- [GET /collections](#)
- [GET /jobs/{jobId}](#)
- [GET /jobs](#)
- [GET /metadata](#)
- [GET /players/{playerIndex}/{playbackName}](#)
- [GET /players/{playerIndex}](#)
- [GET /players](#)
- [GET /playlists](#)
- [GET /sse](#)
- [GET /storages/{uuid}](#)
- [GET /storages](#)
- [GET /system](#)
- [GET /players/{playerIndex}/takeConfiguration](#)
- [POST /show/current](#)
- [POST /jobs/{jobId}/cancel](#)
- [POST /players/{playerIndex}/{playbackName}/mute](#)
- [POST /players/{playerIndex}/{playbackName}/next](#)
- [POST /players/{playerIndex}/{playbackName}/pause](#)
- [POST /players/{playerIndex}/{playbackName}/play](#)
- [POST /players/{playerIndex}/takeConfiguration](#)
- [POST /players/{playerIndex}/{playbackName}/previous](#)
- [POST /system/reboot](#)
- [POST /players/{playerIndex}/{playbackName}/seek](#)
- [POST /players/{playerIndex}/{playbackName}/media](#)
- [POST /players/{playerIndex}/{playbackName}/setVolume](#)
- [POST /system/shutdown](#)
- [POST /players/{playerIndex}/{playbackName}/stop](#)
- [POST /storages/{uuid}/eject](#)
- [POST /storages/{uuid}/ingest](#)
- [POST /players/{playerIndex}/take](#)

- [POST /players/{playerIndex}/{playbackName}/unmute](#)

Core

DELETE /show/current

Delete the current show (**deleteShow**)

Output configuration and player state is reset to default values. Media library is cleared. Playback is immediately stopped. Does not delete media files on the server.

Responses

204

Successful operation

GET /show/current

Export a showfile (**exportShow**)

The exported showfile contains snapshot of current output configuration, player state and media library.

Calling this endpoint with browser will result in browser downloading the showfile. This is a convenient way of backing up the show without opening the RCS.

Return type

File

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/vnd.analogway.coreplay.show

Responses

200

Successful operation [File](#)

GET /version

API Version (**getAPIVersion**)

Returns the server API version

Return type

[Version](#)

Example data

Content-Type: application/json

```
{
  "patch" : 0,
  "major" : 1,
  "minor" : 0,
  "tag" : "-rc1",
  "isPublicRelease" : false
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

Successful operation [Version](#)

404

Requested resource not found. [Error](#)

GET /collections

All collections in the Media Library (**getCollections**)

Return type

map[[MediaCollection](#)]

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

Successful operation

GET /jobs/{jobId}

Returns a single job (**getJob**)

Returns a single job specified by jobId

Path parameters

jobId (required)

Path Parameter — Id of the job default: null

Return type

[Job](#)

Example data

Content-Type: application/json

```
{
  "result" : "InProgress",
  "timeElapsed" : 123,
  "timeTotal" : 600,
  "progress" : 0.56,
  "type" : "ingestion",
  "error" : "ioError"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

Successful operation [Job](#)

404

Requested resource not found. [Error](#)

500

An unknown error occurred. [Error](#)

GET /jobs

List of Jobs (**getJobs**)

Returns an object containing all the running or finished jobs

Return type

map[[Job](#)]

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

Successful operation

500

An unknown error occurred. [Error](#)

GET /metadata

Request metadata (**getMetadata**)

Metadata for a specific content (media file) can be queried.

Same metadata is included in MediaSlot.

Query parameters

contentType (required)

Query Parameter — The URL of the file for which metadata is needed default: null

Return type

[Metadata](#)

Example data

Content-Type: application/json

```
{
  "duration" : 10.0,
  "container" : {
    "size" : 6,
    "created" : "2000-01-23T04:56:07.000+00:00",
    "modified" : "2000-01-23T04:56:07.000+00:00",
    "supported" : true
  }
}
```

```

},
"image" : {
  "codec" : "jpeg",
  "aspectRatioText" : "16/9",
  "width" : 1920,
  "aspectRatio" : 1.7777777777777777,
  "caveats" : [ null, null ],
  "supported" : true,
  "height" : 1080
},
"imageSequence" : {
  "firstFrame" : 1,
  "lastFrame" : 5900
},
"video" : [ {
  "profileDescription" : "Main10",
  "colorRange" : "limited",
  "framerate" : 60.0,
  "aspectRatioText" : "16/9",
  "pixelFormatDescription" : "yuv420",
  "aspectRatio" : 1.7777777777777777,
  "bitrate" : 5625000,
  "caveats" : [ "unsupportedVideo", "unsupportedVideo" ],
  "colorSpace" : "mpeg",
  "duration" : 10.0,
  "codec" : "h264",
  "width" : 1920,
  "averageFramerate" : 60.0,
  "supported" : true,
  "height" : 1080
}, {
  "profileDescription" : "Main10",
  "colorRange" : "limited",
  "framerate" : 60.0,
  "aspectRatioText" : "16/9",
  "pixelFormatDescription" : "yuv420",
  "aspectRatio" : 1.7777777777777777,
  "bitrate" : 5625000,
  "caveats" : [ "unsupportedVideo", "unsupportedVideo" ],
  "colorSpace" : "mpeg",
  "duration" : 10.0,
  "codec" : "h264",
  "width" : 1920,
  "averageFramerate" : 60.0,
  "supported" : true,
  "height" : 1080
} ],
"audio" : [ {
  "duration" : 10.0,
  "codec" : "pcm",
  "sampleFormatDescription" : "flt",
  "channels" : 2,
  "bitrate" : 192000,
  "language" : "language",
  "sampleRate" : 0,
  "caveats" : [ "unsupportedAudio", "unsupportedAudio" ],
  "supported" : true
}, {
  "duration" : 10.0,
  "codec" : "pcm",
  "sampleFormatDescription" : "flt",
  "channels" : 2,
  "bitrate" : 192000,
  "language" : "language",
  "sampleRate" : 0,
  "caveats" : [ "unsupportedAudio", "unsupportedAudio" ],
  "supported" : true
} ],
"type" : "videoFile",

```

```

"thumbnails" : {
  "small" : {
    "width" : 256,
    "url" : "/thumbnails/medium/ZmIsZTovLy9tZWRpYS92aWRlbzEubXA0.jpg?1729889470",
    "height" : 144
  },
  "large" : {
    "width" : 256,
    "url" : "/thumbnails/medium/ZmIsZTovLy9tZWRpYS92aWRlbzEubXA0.jpg?1729889470",
    "height" : 144
  },
  "medium" : {
    "width" : 256,
    "url" : "/thumbnails/medium/ZmIsZTovLy9tZWRpYS92aWRlbzEubXA0.jpg?1729889470",
    "height" : 144
  }
},
"caveats" : [ "unsupportedVideo", "unsupportedVideo" ],
"supported" : true
}

```

Example data

Content-Type: application/json

```

{
  "error" : "error"
}

```

Example data

Content-Type: application/json

```

{
  "error" : "error"
}

```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

Successful operation [Metadata](#)

400

Badly formatted request. Content url value is invalid. [Error](#)

404

Requested resource not found. [Error](#)

GET /players/{playerIndex}/{playbackName}

State of a given playback (**getPlayback**)

Single playback contains a media and the usual playback controls (stop, pause, play...).

Path parameters

playerIndex (required)

Path Parameter — The player index starting from one default: null

playbackName (required)

Path Parameter — The playback name program or preview default: null

Return type

[Playback](#)

Example data

Content-Type: application/json

```
{
  "volume" : 0.8008281904610115,
  "normalizedPosition" : 0.5,
  "mediaUrl" : "mediaUrl",
  "state" : "playing",
  "position" : 1.2,
  "mediaPosition" : 2.9,
  "isMuted" : true,
  "isResourceConstrained" : true
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

Successful operation [Playback](#)

404

Player or Playback not Found. [Error](#)

500

An unknown error occurred. [Error](#)

GET /players/{playerIndex}

State of the player engine (**getPlayer**)

Player engine refers to a pair of Preview and Program playbacks and their Take functionality.

Path parameters

playerIndex (required)

Path Parameter — The player index starting from one default: null

Return type

[Player](#)

Example data

Content-Type: application/json

```
{
  "preview" : {
    "volume" : 0.8008281904610115,
    "normalizedPosition" : 0.5,
    "mediaUrl" : "mediaUrl",
    "state" : "playing",
    "position" : 1.2,
    "mediaPosition" : 2.9,
    "isMuted" : true,
    "isResourceConstrained" : true
  },

```

```

"takeProgress" : 0.5,
"takeConfiguration" : {
  "mode" : "swap",
  "startPoint" : "inpoint",
  "transitionDuration" : 1.5,
  "transitionType" : "crossfade"
},
"program" : {
  "volume" : 0.8008281904610115,
  "normalizedPosition" : 0.5,
  "mediaUrl" : "mediaUrl",
  "state" : "playing",
  "position" : 1.2,
  "mediaPosition" : 2.9,
  "isMuted" : true,
  "isResourceConstrained" : true
}
}

```

Example data

Content-Type: application/json

```

{
  "error" : "error"
}

```

Example data

Content-Type: application/json

```

{
  "error" : "error"
}

```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

Successful operation [Player](#)

404

Player Index not found. [Error](#)

500

An unknown error occurred. [Error](#)

GET /players

State of all player engines (**getPlayers**)

Player engine refers to a pair of Preview and Program playbacks and their Take functionality.

Return type

map[[Player](#)]

Example data

Content-Type: application/json

```

{
  "error" : "error"
}

```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

Successful operation

500

An unknown error occurred. [Error](#)

GET /playlists

All playlists in the Media Library (**getPlaylists**)

Return type

map[[Playlist](#)]

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

Successful operation

GET /sse

Create an SSE stream (**getSSE**)

See <https://datatracker.ietf.org/doc/html/rfc6902> for details about the message format.

Opening a connection to this endpoint creates an SSE stream which pushes changes to API items in real-time, using JSON Patch format.

Example SSE patch message:

```
data: [ { "op": "replace", "path": "/players/1/program/mediaUrl", "value": "core://playlists/1/entries/1.0" } ]
```

Frequency of the SSE patch messages depend on multiple factors, such as playback state and refresh rate of the output(s).

Rate may be limited and operations may be bundled together to save bandwidth and processing power.

There may be multiple patch operations on any given resource in a single SSE patch message.

Query parameters

topic (required)

Query Parameter —

Array of topics to include in the stream. If no topics are specified, the endpoint returns everything.

- players: Includes playback status information
- system: Includes system uptime that can be used as a connection heartbeat.
- collections: Includes media collections and media slots
- playlists: Includes playlists and playlist entries
- storages: Storage devices, such as connected USB storage devices
- jobs: Jobs

default: null

Return type

array[[JSONPatch](#)]

Example data

Content-Type:

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- text/event-stream
- application/json

Responses

200

Successful operation

404

Requested resource not found. [Error](#)

GET /storages/{uuid}

Get information about specific storage (**getStorage**)

List details about storage. This path in Patch message can also be used to detect apperance or disappearance of a specific storage over SSE.

Path parameters

uuid (required)

Path Parameter — UUID of the storage. default: null

Return type

[Storage](#)

Example data

Content-Type: application/json

```
{
  "deviceType" : "internal",
  "totalSize" : 20000000000000,
  "name" : "name",
  "deviceName" : "deviceName",
  "ingestionJob" : "ingestionJob",
  "filesystem" : [ "exfat", "xfs", "vfat", "ntfs" ],
  "availableSize" : 10000000000000,
  "supported" : true
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

Successful operation [Storage](#)

404

Requested resource not found. [Error](#)

500

An unknown error occurred. [Error](#)

GET /storages

List of storages (**getStorages**)

Returns an associative array of available storages. Storage is local or external storage device that can contain media content. Local storage supports playback of media files. USB storage devices support ingestion of media files, but playback of media directly from USB storage device is not supported. USB storage devices support exFAT, FAT32 and NTFS partitions. Partition size must be at least one gigabyte.

For GPT based storage partitions, the GTP GUID is used as the key. For MBR based storage partitions, the key is deterministically based on the device label and partition number.

Return type

map[[Storage](#)]

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

Successful operation

500

An unknown error occurred. [Error](#)

GET /system

System Information (**getSystemInformation**)

Returns information about the device and its firmware.

Return type

[System](#)

Example data

Content-Type: application/json

```
{
  "hostname" : "coreplay-12345",
  "serialNumber" : "DV0001",
  "firmwareVersion" : {
    "patch" : 0,
    "major" : 1,
    "minor" : 0,
    "tag" : "-rc1",
    "isPublicRelease" : false
  },
  "productType" : "CPS01-R1",
  "uptime" : 0
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

Successful operation [System](#)

404

Requested resource not found. [Error](#)

GET /players/{playerIndex}/takeConfiguration

Gets the default take settings (**getTakeConfiguration**)

The methods returns the current default Take configuration

Path parameters

playerIndex (required)

Path Parameter — The player index starting from one default: null

Return type

[Take](#)

Example data

Content-Type: application/json

```
{
  "mode" : "swap",
  "startPoint" : "inpoint",
  "transitionDuration" : 1.5,
  "transitionType" : "crossfade"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

Successful operation [Take](#)

404

Player or Playback not Found. [Error](#)

500

An unknown error occurred. [Error](#)

POST /show/current

Import a showfile (**importShow**)

Imports properties from uploaded showfile to the server.

If this endpoint is called without include parameter, all properties from the showfile are imported.

If this endpoint is called with the force parameter, even incompatible showfile versions will be imported.

The include parameter can be used to specify what is imported from the showfile. Possible import keys:

- video: Video output configuration
- audio: Audio output configuration
- playback: Playing media, volume and autostart setting.
- medialibrary: MediaCollections, MediaSlots, Playlists, Playlist Entries and Cues

Consumes

This API call consumes the following media types via the Content-Type request header:

- application/octet-stream

Request body

body [file](#) (optional)

Body Parameter —

Query parameters

include (optional)

Query Parameter — Properties to include default: null

force (optional)

Query Parameter — Force import showfile regardless of showfile version default: null

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

204

Successful operation

400

Badly formatted request. One or more parameters were invalid. [Error](#)

422

Showfile version is not supported. Use force parameter to import it anyways.

POST /jobs/{jobId}/cancel

Cancels a job (**jobCancel**)

Path parameters

jobId (required)

Path Parameter — Id of the job to cancel default: null

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

204

Cancellation of the job was successfully requested

400

Badly formatted request. One or more parameters were invalid. [Error](#)

404

Job was not found [Error](#)

500

An unknown error occurred. [Error](#)

POST /players/{playerIndex}/{playbackName}/mute

Mute playback (**mute**)

Path parameters

playerIndex (required)

Path Parameter — The player index starting from one default: null

playbackName (required)

Path Parameter — The playback name program or preview default: null

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

204

Successful operation

404

Player or playback not found [Error](#)

POST /players/{playerIndex}/{playbackName}/next

Advance to next item (**next**)

Transition is applied

Advances to the next item in the playlist

Path parameters

playerIndex (required)

Path Parameter — The player index starting from one default: null

playbackName (required)

Path Parameter — The playback name program or preview default: null

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

204

Successful operation

404

Player or playback not found [Error](#)

409

Not playing from a playlist [Error](#)

POST /players/{playerIndex}/{playbackName}/pause

Pause Playback (**pause**)

Sets the playback state to pause.

Path parameters

playerIndex (required)

Path Parameter — The player index starting from one default: null

playbackName (required)

Path Parameter — The playback name program or preview default: null

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

204

Successful operation

404

Player or playback not found [Error](#)

409

The playback has no media to pause [Error](#)

POST /players/{playerIndex}/{playbackName}/play

Play playback (**play**)

Sets the playback state to play.

Path parameters

playerIndex (required)

Path Parameter — The player index starting from one default: null

playbackName (required)

Path Parameter — The playback name program or preview default: null

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

204

Successful operation

404

Player or playback not found [Error](#)

409

Playback has no media for playing [Error](#)

POST /players/{playerIndex}/takeConfiguration

Configures the default take settings (**postTakeConfiguration**)

The method configures the default take settings that are used when take is triggered without parameters

Path parameters

playerIndex (required)

Path Parameter — The player index starting from one default: null

Consumes

This API call consumes the following media types via the Content-Type request header:

- application/json

Request body

Take [Take](#) (optional)

Body Parameter —

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

204

Successful operation

400

Badly formatted request. One or many of the parameters were invalid. [Error](#)

404

Player Index not found. [Error](#)

500

An unknown error occurred. [Error](#)

POST /players/{playerIndex}/{playbackName}/previous

Move to previous item (**previous**)

Transition is not applied

Moves to the previous item in the playlist

Path parameters

playerIndex (required)

Path Parameter — The player index starting from one default: null

playbackName (required)

Path Parameter — The playback name program or preview default: null

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

204

Successful operation

404

Player or playback not found [Error](#)

409

Not playing from a playlist [Error](#)

POST /system/reboot

Reboot (**reboot**)

Reboots the device

Responses

204

Successful operation

404

Requested resource not found.

POST /players/{playerIndex}/{playbackName}/seek

Seek inside the current media (**seek**)

Positions the playhead.

For media containing keyframes, the playhead is positioned at the nearest previous keyframe.

This means that for media using inter-frame compression (where decoding requires information from previous or forward frames), the granularity of frames that can be accurately seeked to depends largely on the codec used and its specific settings.

If the playback is stopped, the state is changed to paused.

Only one of position, mediaPosition or normalizedPosition parameters is allowed, but not both at the same time.

Path parameters

playerIndex (required)

Path Parameter — The player index starting from one default: null

playbackName (required)

Path Parameter — The playback name program or preview default: null

Query parameters

position (optional)

Query Parameter — The seek position in seconds from inpoint to endpoint. default: null

mediaPosition (optional)

Query Parameter — The seek position in seconds from start of media. default: null

normalizedPosition (optional)

Query Parameter — The normalized seek position between inpoint (0.0) and outpoint (1.0)
default: null

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

204

Successful operation

400

Badly formatted request. One or more parameters were invalid. [Error](#)

404

Player or playback not found [Error](#)

409

The playback has no media to seek [Error](#)

POST /players/{playerIndex}/{playbackName}/media

Set Playback Media (**setPlaybackMedia**)

Sets the given media on the playback, without changing the playback state.

If the URL refers to an playlist entry index, the value will be resolved to the index format url for the given entry.

Setting an empty string will clear the currently playing media from the playback.

Path parameters

playerIndex (required)

Path Parameter — The player index starting from one default: null

playbackName (required)

Path Parameter — The playback name program or preview default: null

Consumes

This API call consumes the following media types via the Content-Type request header:

- application/json

Request body

MediaIdentifier [MediaIdentifier](#) (optional)

Body Parameter —

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

204

Successful operation

400

Badly formatted request. One or many of Media parameters were invalid. [Error](#)

404

Player or playback not found [Error](#)

500

An unknown error occurred. [Error](#)

409

The player is not in a state where media can be set [Error](#)

POST /players/{playerIndex}/{playbackName}/setVolume

Set playback volume (**setVolume**)

Path parameters

playerIndex (required)

Path Parameter — The player index starting from one default: null

playbackName (required)

Path Parameter — The playback name program or preview default: null

Query parameters

volume (required)

Query Parameter — New volume for the playback. Value is in decibels. default: null format: double

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

204

Successful operation

400

Badly formatted request. Query parameter was invalid. [Error](#)

404

Player or playback not found [Error](#)

POST /system/shutdown

Shutdown (**shutdown**)

Turns off the device

Responses

204

Successful operation

404

Requested resource not found.

POST /players/{playerIndex}/{playbackName}/stop

Stop playback (**stop**)

Sets the playback state to stop.

Path parameters

playerIndex (required)

Path Parameter — The player index starting from one default: null

playbackName (required)

Path Parameter — The playback name program or preview default: null

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

204

Successful operation

404

Player or playback not found [Error](#)

POST /storages/{uuid}/eject

Ejects an USB storage device (**storageEject**)

The endpoint creates a job for ejecting an USB storage devices. All partitions of the linked device disappear as result and all linked jobs are canceled.

Path parameters

uuid (required)

Path Parameter — UUID of the storage to eject default: null

Return type

[JobIdResponse](#)

Example data

Content-Type: application/json

```
{
  "jobId" : "jobId"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

Successful operation, ejection is started and the job id is returned. [JobIdResponse](#)

404

Requested resource not found. [Error](#)

500

An unknown error occurred. [Error](#)

POST /storages/{uuid}/ingest

Ingests a USB storage device partition (**storageIngest**)

Imports all media files from the specified storage device, such as connected USB storage device partition. Ingest copies supported media files from the storage. Ingestion completes fully or not at all. Only one ingestion can run for a storage. If there already

running ingestion, the request will fail.

Path parameters

uuid (required)

Path Parameter — UUID of the storage to ingest. default: null

Consumes

This API call consumes the following media types via the Content-Type request header:

- application/json

Request body

storageIngest_request [storageIngest_request](#) (optional)

Body Parameter —

Query parameters

destinationDirectory (optional)

Query Parameter — Destination directory on server default storage. Created if doesn't exists. If parameter is omitted, source storage name is used as destination directory name. default: null

overwrite (optional)

Query Parameter — default: overwrite

preview (optional)

Query Parameter — If set to true, ingestion will only preview what would occur if the ingestion would run. This parameter can be used to test include filter finds expected files or check what would be overwritten if ingestion would run. Defaults to false. default: null

Return type

[JobIdResponse](#)

Example data

Content-Type: application/json

```
{
  "jobId" : "jobId"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

Successful operation, ingestion is started and the job id is returned. [JobIdResponse](#)

400

Badly formatted request. One or more parameters were invalid. [Error](#)

409

Ingestion cannot be started. Possibly there is already an ingestion running for the source storage. [Error](#)

POST /players/{playerIndex}/take

Take from Preview to Program (**take**)

Take operation is used to take media from Preview playback to Program playback. If empty object is specified as the request body, the default take configuration is used

Path parameters

playerIndex (required)

Path Parameter — The player index starting from one default: null

Consumes

This API call consumes the following media types via the Content-Type request header:

- application/json

Request body

take_request [take_request](#) (optional)

Body Parameter —

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

204

Successful operation

400

Badly formatted request. One or many of Take parameters were invalid. [Error](#)

404

Player Index not found. [Error](#)

500

An unknown error occurred. [Error](#)

POST /players/{playerIndex}/{playbackName}/unmute

Unmute playback (**unmute**)

Path parameters

playerIndex (required)

Path Parameter — The player index starting from one default: null

playbackName (required)

Path Parameter — The playback name program or preview default: null

Example data

Content-Type: application/json

```
{
  "error" : "error"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses**204**

Successful operation

404

Player or playback not found [Error](#)

Models

[[Jump to Methods](#)]

Table of Contents

1. [AlignmentModifier -](#)
2. [AudioCaveat -](#)
3. [Caveat -](#)
4. [ColorRange -](#)
5. [ColorSpace -](#)
6. [Error -](#)
7. [FitModifier -](#)
8. [FlipModifier -](#)
9. [GammaModifier -](#)
10. [HueModifier -](#)
11. [JSONPatch -](#)
12. [Job -](#)
13. [JobIdResponse -](#)
14. [MediaCollection -](#)
15. [MediaIdentifier -](#)
16. [MediaSlot -](#)
17. [MediaSlot_modifiers_inner -](#)
18. [MediaSlot_thumbnails -](#)
19. [Metadata -](#)
20. [Metadata_audio_inner -](#)
21. [Metadata_container -](#)
22. [Metadata_image -](#)
23. [Metadata_imageSequence -](#)
24. [Metadata_thumbnails -](#)
25. [Metadata_video_inner -](#)
26. [Overwrite -](#)
27. [Playback -](#)
28. [Player -](#)
29. [Playlist -](#)
30. [PlaylistEntry -](#)
31. [RGBModifier -](#)
32. [SlotImageSequence -](#)
33. [Storage -](#)
34. [System -](#)
35. [Take -](#)
36. [Thumbnail -](#)

- 37. [ToneModifier -](#)
- 38. [TransformModifier -](#)
- 39. [TransitionType -](#)
- 40. [Version -](#)
- 41. [VideoCaveat -](#)
- 42. [VolumeModifier -](#)
- 43. [storageIngest_request -](#)
- 44. [take_request -](#)

AlignmentModifier -

type

[oas_any_type_not_mapped](#)

alignment

[String](#).

Enum:

center
centerLeft
centerRight
centerTop
centerBottom
topLeft
topRight
bottomLeft
bottomRight

AudioCaveat -

Limitations in audio playback:

- unsupportedAudio: The audio codec or format is unsupported.

Caveat -

Limitations in content playback:

- unsupportedVideo: The video codec or format is unsupported.
- unsupportedAudio: The audio codec or format is unsupported.
- performance: Playback of the media might require more performance than can be provided.

ColorRange -

ColorSpace -

Error -

error (optional)

[String](#). Error description

FitModifier -

type

[oas_any_type_not_mapped](#)

fitMode

[String](#).

Enum:

fit
cover
stretch

native

FlipModifier -

type

[*oas_any_type_not_mapped*](#)

flipX

[*Boolean*](#)

flipY

[*Boolean*](#)

GammaModifier -

type

[*oas_any_type_not_mapped*](#)

gamma

[*Double*](#) format: double

HueModifier -

type

[*oas_any_type_not_mapped*](#)

hue

[*Double*](#) format: double

JSONPatch -

JSON Patch record

op

[*String*](#).

Enum:

add

replace

test

path

[*String*](#).

value

[*oas_any_type_not_mapped*](#)

Job -

type

[*String*](#).

Enum:

ingestion

ejection

result

[*String*](#).

Enum:

inProgress

succeeded

cancelling

cancelled

failed

error (optional)

[*String*](#).

Enum:

ioError

fileSystemError
unknownError

progress

[*Double*](#) Current ingestion progress as a value from 0.0 to 1.0 format: double

timeElapsed

[*Long*](#) Elapsed time on current job in seconds format: int64

timeTotal

[*Long*](#) Total time on current job in seconds format: int64

JobIdResponse -

jobId

[*String*](#) The job identifier

MediaCollection -

name (optional)

[*String*](#) Name of the media collection

slots (optional)

[*map\[String, MediaSlot\]*](#) A map from integer to MediaSlot

MediaIdentifier -

mediaUrl

[*String*](#) Url of the media library item to assign to the playback

MediaSlot -

name (optional)

[*String*](#) Customizable name of the media slot

contentUrl (optional)

[*String*](#) A URL string that starts with file:// or other supported prefix and contains user absolute path to a file or image sequence. The user absolute paths are based on the path accessible using FTP client or RCS filebrowser. On server filesystem level, these paths resolve to /opt/coreplay/user/, where media subfolder is user writable.

inpoint (optional)

[*BigDecimal*](#) The starting position of the media in seconds

outpoint (optional)

[*BigDecimal*](#) The ending position of the media in seconds. Defaults to null, indicating the end of media

thumbnails (optional)

[*MediaSlot_thumbnails*](#)

metadata (optional)

[*Metadata*](#)

imageSequence (optional)

[*SlotImageSequence*](#)

duration (optional)

[*Double*](#) Effective playback duration of the media in seconds. Takes account content duration, inpoint, outpoint, image sequence frame rate. Null means infinite. format: double

mediaDuration (optional)

[*BigDecimal*](#) Absolute media duration when the media has an inherent duration. Null otherwise.

customDuration (optional)

[*Double*](#) Present and valid only when playing a slot that doesn't have inherent duration, such as still image. The value is null if it has not been set. format: double

infiniteDuration (optional)

[*Boolean*](#) Flag indicating that the media has infinite duration. If it is set, the customDuration field is ignored. Infinite duration is used for still images and live streams that do not have inherent duration and can be played forever.

mediaExists (optional)

[*Boolean*](#) A flag indicating that the media attached to the slot exists.

isPlayable (optional)

[*Boolean*](#) A flag indicating the playability of the slot

modifiers (optional)

[*array\[MediaSlot_modifiers_inner\]*](#).

MediaSlot_modifiers_inner -

type

[*oas_any_type_not_mapped*](#)

flipX

[*Boolean*](#)

flipY

[*Boolean*](#)

scale

[*array\[Double\]*](#) format: double

rotate

[*Double*](#) format: double

translate

[*array\[Double\]*](#) format: double

fitMode

[*String*](#).

Enum:

fit
cover
stretch
native

alignment

[*String*](#).

Enum:

center
centerLeft
centerRight
centerTop
centerBottom
topLeft
topRight
bottomLeft
bottomRight

hue

[*Double*](#) format: double

r (optional)

[*Double*](#) format: double

g (optional)

[*Double*](#) format: double

b (optional)

[*Double*](#) format: double

saturation

[*Double*](#) format: double

contrast

[*Double*](#) format: double

brightness

[*Double*](#) format: double

gamma

[*Double*](#) format: double

volume (optional)

[*Double*](#) format: double

muted

[*Boolean*](#)

MediaSlot_thumbnails -

Slot specific thumbnails

small (optional)

[*Thumbnail*](#)

medium (optional)

[*Thumbnail*](#)

large (optional)

[*Thumbnail*](#)

Metadata -**type (optional)**

[*String*](#).

Content type:

- videoFile: A single video file on server local storage. This can be for example .mp4 file with H.264 video and PCM audio.
- audioFile: A single audio file on server local storage. This can be for example .wav file with PCM audio
- imageFile: A single still image file on server local storage. This can be for example .png file
- imageSequence: A group of still images on server local storage which meet specific requirements: There are multiple image files on same path. At least 49 still images must be Present They have same file extension and extension is supported. Supported extensions include dpx They have ordinal number in filename, number of characters in ordinal numbers is identical and ordinal number value is continuous. To refer to image sequence: path + stem + %0 + width of ordinal numbers + d + extension where stem is the filename up to ordinal number. Image sequences require playback frame rate. This value is provided in MediaSlot. Length of image sequence can be limited by providing first and last frame numbers in MediaSlot.
- unknown: The media type could not be determined

Enum:

videoFile
audioFile
imageSequence
imageFile

duration (optional)

[*Double*](#) Total duration of the media in seconds, or null if unknown. If the media contains multiple streams, this corresponds to the longest stream duration. format: double

supported (optional)

[*Boolean*](#) boolean flag indicating if the server supports the content. This attribute also appears in audio and video or other objects to indicate that specific track of the content is supported or not. At content entry root level, the value indicates if the content is considered supported. For example, if a file contains both audio and video stream, file is considered supported even if the audio stream is unsupported. But not vice versa. The value can be considered the expected common case. For specific use cases, please also check attribute value on specific streams.

caveats (optional)

[*array\[Caveat\]*](#) All playback caveats for this specific media. The child objects' caveats can be observed to determine where the individual caveats come from.

thumbnails (optional)

[*Metadata_thumbnails*](#)

video (optional)

[array\[Metadata_video_inner\]](#). Properties for the different video streams of the content. Please note that currently, no Device supports playback of anything other than the first video track.

audio (optional)

[array\[Metadata_audio_inner\]](#). Properties for different audio streams of the content. Please note that currently, no Device supports playback of anything other than the first audio track.

container (optional)

[Metadata_container](#)

imageSequence (optional)

[Metadata_imageSequence](#)

image (optional)

[Metadata_image](#)

Metadata_audio_inner -**supported (optional)**

[Boolean](#) Is the audio stream supported?

caveats (optional)

[array\[AudioCaveat\]](#). Caveats for the audio stream

channels (optional)

[Integer](#) Number of audio channels format: int32

codec (optional)

[String](#). Audio codec, or null if unknown

bitrate (optional)

[Integer](#) Bitrate of the audio stream in bits per second

sampleRate (optional)

[Integer](#) Sample rate of the audio stream in Hz

sampleFormatDescription (optional)

[String](#). Sample format of audio stream

language (optional)

[String](#). 3-letter ISO 639 Language code, or null if unknown or not applicable

duration (optional)

[Double](#) Duration of the audio in seconds, or null if unknown format: double

Metadata_container -

Properties of the container of the content

supported (optional)

[Boolean](#) Is the container supported?

size (optional)

[Long](#). The size of the file in bytes. null if size cannot be determined. format: int64

created (optional)

[Date](#) Creation time of the file. null if time cannot be determined. format: date-time

modified (optional)

[Date](#) Last modification time of the file. null if time cannot be determined. format: date-time

Metadata_image -

Present only for still image type of media

supported (optional)

[Boolean](#) Is the image supported?

caveats (optional)

[array\[VideoCaveat\]](#). Caveats for the image

width (optional)

[*Integer*](#) Pixel width of the image

height (optional)

[*Integer*](#) Pixel height of the image

aspectRatio (optional)

[*Double*](#) Approximate aspect ratio of the image format: double

aspectRatioText (optional)

[*String*](#) Exact aspect ratio of the image as text in fractional format

codec (optional)

[*String*](#) Codec of the image, or null if unknown

colorSpace (optional)

[*ColorSpace*](#)

Metadata_imageSequence -

Present only for image sequence type of media

firstFrame (optional)

[*Integer*](#) First frame number of the image sequence. Minimum value for use in a media slot.

lastFrame (optional)

[*Integer*](#) Last frame number of the image sequence. Maximum value for use in a media slot.

Metadata_thumbnails -

Default thumbnails for content

small (optional)

[*Thumbnail*](#)

medium (optional)

[*Thumbnail*](#)

large (optional)

[*Thumbnail*](#)

Metadata_video_inner -

supported (optional)

[*Boolean*](#) Is the video stream supported?

caveats (optional)

[*array\[VideoCaveat\]*](#) Caveats for the video stream

width (optional)

[*Integer*](#) Pixel width of the video stream

height (optional)

[*Integer*](#) Pixel height of the video stream

aspectRatio (optional)

[*Double*](#) Approximate aspect ratio of the video stream format: double

aspectRatioText (optional)

[*String*](#) Exact aspect ratio of the video stream as text in fractional format

codec (optional)

[*String*](#) Codec of the video stream, or null if unknown

profileDescription (optional)

[*String*](#) Codec profile info if applicable

colorRange (optional)

[*ColorRange*](#)

colorSpace (optional)

[*ColorSpace*](#)

pixelFormatDescription (optional)

[*String*](#) Pixel format of media

bitrate (optional)

[Long](#) Average bitrate of the video stream in bits per second format: int64

framerate (optional)

[Double](#) The nominal frame rate of the media (frames per second / Hz) This framerate is usually from the video metadata format: double

averageFramerate (optional)

[Double](#) The average frame rate of the media (frames per second / Hz) This framerate is calculated from the duration and frame count format: double

duration (optional)

[Double](#) Duration of the video in seconds, or null if unknown format: double

Overwrite -

Parameter used for the ingestion. Overwrite behavior. Applied at file level. If same directory exists on both source and destination, ingesting iterates files on both, and applies overwrite per file basis.

- skip: Skip copying file if it already exists in the destination directory
- overwrite: Overwrite existing file
- overwriteOld: Overwrite existing file if it is older than file being ingested.
- cancel: Cancel ingestion if a file already exists on the destination directory.

Playback -**mediaUrl (optional)**

[String](#) Subtype of CoreUrl that identifies a playable media. It can be either a Entry URL or a Slot URL.

state (optional)

[String](#) Playback status

Enum:

playing
paused
stopped

position (optional)

[BigDecimal](#) Playback position in seconds

normalizedPosition (optional)

[BigDecimal](#) Normalized playback position

mediaPosition (optional)

[BigDecimal](#) Playback position in seconds from the start of media.

volume (optional)

[Double](#) Playback volume in dB format: double

isMuted (optional)

[Boolean](#) Flag that indicates that the playback is muted

isResourceConstrained (optional)

[Boolean](#) Flag that indicates that the playback is resource constrained, and the video quality is downgraded or the playback is stopped altogether. This happens when there are not enough system resources for all active playbacks.

Player -**takeProgress (optional)**

[Double](#) Normalized position of take transition. Null if not in progress. format: double

preview (optional)

[Playback](#)

program (optional)

[Playback](#)

takeConfiguration (optional)

[Take](#)

Playlist -

name (optional)

[String](#) Name of the playlist

entries (optional)

[map\[String, PlaylistEntry\]](#) A map of playlist entries

PlaylistEntry -

mediaUrl (optional)

[String](#).

Subtype of CoreUrl that allows addressing a specific Slot in a Collection.

Formats:

1. `core://collections/{collection_id}/slots/{slot_index}`: Refers to a specific Slot within a Collection.

endAction (optional)

[String](#) Playback action to be done when the entry finishes

Enum:

next
pause
stop
repeat

transitionType (optional)

[TransitionType](#)

transitionDuration (optional)

[BigDecimal](#) Duration of the transition in seconds Transition duration can be limited by the duration of the playing medias.

RGBModifier -

type

[oas_any_type_not_mapped](#)

r (optional)

[Double](#) format: double

g (optional)

[Double](#) format: double

b (optional)

[Double](#) format: double

SlotImageSequence -

framerate (optional)

[Double](#) Playback framerate for the image sequences. format: double

firstFrame (optional)

[Integer](#) First frame number of the image sequence for playback. If null, first frame from metadata will be used.

lastFrame (optional)

[Integer](#) Last frame number of the image sequence for playback. If null, last frame from metadata will be used.

Storage -

name

[*String*](#) Storage name. Name is based on device or partition name or label if it can be determined.

deviceName

[*String*](#) Storage device name. Name is based on device name if it can be determined.

totalSize

[*Long*](#) Total size of the storage in bytes. format: int64

filesystem

[*String*](#) Storage filesystem type. Null if cannot be determined.

supported

[*Boolean*](#) Indicates if the storage is supported. Unsupported storages cannot be used or ingested. Please note that some unsupported storages do not appear even as unsupported storages.

availableSize

[*Long*](#) Size of the storage space available in bytes. Null if size cannot be determined. format: int64

deviceType

[*String*](#) Storage device type.

Enum:

internal
usb

ingestionJob (optional)

[*String*](#) Identifier of the job doing ingestion for this storage or null if no ingestion created.

System -**hostname (optional)**

[*String*](#) The hostname of the device

productType (optional)

[*String*](#) The product code of the Device

serialNumber (optional)

[*String*](#) The serial number of the Device

firmwareVersion (optional)

[*Version*](#) The version of the current firmware

uptime (optional)

[*Integer*](#) System uptime in seconds format: int32

Take -**mode**

[*String*](#) Affects preview state after take transition. Either swap program and preview, or copy so preview remains unchanged

Enum:

swap
copy

startPoint

[*String*](#) After take, the content starts from inpoint or the position the playhead was in the preview prior to take

Enum:

inpoint
position

transitionType

[*TransitionType*](#)

transitionDuration

[*BigDecimal*](#) Duration of the transition in seconds Transition duration can be limited by the duration of the playing medias.

Thumbnail -

width (optional)

[*Integer*](#) Thumbnail width in pixels

height (optional)

[*Integer*](#) Thumbnail height in pixels

url (optional)

[*String*](#) Thumbnail URL. The URL should be considered a relative URL, so the base URL should be applied before use.

ToneModifier -

type

[*oas_any_type_not_mapped*](#)

saturation

[*Double*](#) format: double

contrast

[*Double*](#) format: double

brightness

[*Double*](#) format: double

TransformModifier -

type

[*oas_any_type_not_mapped*](#)

scale

[*array\[Double\]*](#) format: double

rotate

[*Double*](#) format: double

translate

[*array\[Double\]*](#) format: double

TransitionType -

Type of the transition applied between two consecutive media

Version -

major

[*Integer*](#) format: int32

minor

[*Integer*](#) format: int32

patch

[*Integer*](#) format: int32

tag (optional)

[*String*](#) Present only in alpha and beta builds and documents

isPublicRelease (optional)

[*Boolean*](#) Indicates that this is a supported public release.

VideoCaveat -

Limitations in video playback:

- unsupportedVideo: The video codec or format is unsupported.
- performance: Playback of the media might require more performance than can be provided.

VolumeModifier -

type

[*oas_any_type_not_mapped*](#)

volume (optional)

[*Double*](#) format: double

muted

[*Boolean*](#)

storageIngest_request -

filter (optional)

[*array\[String\]*](#)

take_request -

mode

[*String*](#) Affects preview state after take transition. Either swap program and preview, or copy so preview remains unchanged

Enum:

swap

copy

startPoint

[*String*](#) After take, the content starts from inpoint or the position the playhead was in the preview prior to take

Enum:

inpoint

position

transitionType

[*TransitionType*](#)

transitionDuration

[*BigDecimal*](#) Duration of the transition in seconds Transition duration can be limited by the duration of the playing medias.