

# Avolites web API

## Introduction

The Avolites web API allows you to interact with connected consoles, pragmatically. Allowing you to define your own interface. The API provides access to functionality and resources present on connected consoles. Utilising HTTP as a mean for communication and supporting JSON formatted messages. running on port 4430.

## Requests

The Avolites web API can perform various request such as setting and getting property values; running scripts and retrieving handles. This section will detail the types of requests and give examples on how to use these requests.

### Get Property Value – “get/\*”

Gets the value of the specified property. Use the HTTP GET method. For example:

```
http://169.254.237.136:4430/titan/get/Fixtures/ShowDmxAddress
```

Here the value of the “ShowDmxAddress” property would be returned.

### Set Property Value – “set/\*”

Sets a value for the specified property. Use the HTTP POST method, specifying the new value in the body of the message. For example:

```
POST /titan/set/Fixtures/ShowDmxAddress HTTP/1.1
Host: 169.254.237.136:4430
Cache-Control: no-cache
Postman-Token: 9b09529f-d694-7cbc-0167-ef60194359d2
False
```

This request would set the “ShowDmxAddress” to false.

### Run scripts – “script/\*”

Runs a specified script and returns a response in JSON format, if there is any response. Utilise the HTTP GET method and include any required parameters in the URL. For example:

```
http://169.254.237.136:4430/titan/script/Playbacks/Select/Process?string=playback&int=1723
```

In this example the script “Process” requires a string and integer and these are specified at the end of the URL after the “?”. It is mandatory to put “?” before specifying parameters. When specifying a parameter its type and value should be included in the request.

## Parameter Types

There are number of variable types supported by the API. The basic type's Boolean, float, integer and string are supported. There are also some advanced types supported in the API these include level, level Delta, titanId and userNumber.

### *Level*

The level is used to replace an existing value. For example:

```
http://{IP Address}:4430/titan/script/Masters/SetMasterLevel?string=playback&int=1&level=0.5&float=1.0&bool=false
```

The following example above would result in setting a master level at 0.5.

### *Level Delta*

The level delta add the specified level to an existing value. See example:

```
http://{IP Address}:4430/titan/script/Masters/SetMasterLevel?string=playback&int=1&levelDelta=0.5&float=1.0&bool=false
```

This example would result in incrementing the master level by 0.5.

### *Titan Id*

The titan Id is a unique integer number in titan that is used to identify a property in the system. See example:

```
http://{IP Address}:4430/titan/script/Playbacks/FirePlaybackAtLevel?titanId=1723&level=0.5&bool=false
```

Here the titan Id is used to identify the playback to fire.

### *User Number*

A user number is a number that is assigned to a property in titan that is used for identification purposes. This number can be defined by users. For example:

```
http://{IP Address}:4430/titan/script/Group/StoreGroup?userNumber=2.0
```

In this example a new group is being stored with a user number of 2.0.

## Handle requests

The HTTP GET method should be used for all get handle requests as these request will return handle information.

Get page handles – “handles/ {group name}/ {page index}”

Gets all handles on a specified page (page index) in a specified group (group name). See example:

```
http://{IP Address}:4430/titan/handles/Groups/0
```

Get group handles – “handles/ {group name}”

Gets all handles within the specified group (group name). For example:

```
http://{IP Address}:4430/titan/handles/Fixtures
```

Get all handles – “handles”

Gets all handles. See example:

<http://{IP Address}:4430/titan/handles>

## Get Handle response

All get handle requests will return handle information in a JSON format. See example response below:

```
[
  {
    "handleLocation":{"group":"Fixtures","index":0,"page":1},
    "properties":[{"Key":"lockState","Value":"Unlocked"}],
    "titanId":1689,
    "type":"fixtureHandle",
    "Active":false,
    "Legend":""
  ]
]
```

## Titan scripts and properties

The web API has access to providers. Providers manage functions and properties each provider provides different functionality. In this section some provider have been detailed. Details include functions and properties available.

### Playbacks

The Playbacks provider is responsible for playback functionality. Containing function to record and modify playbacks. Here are some basic playback functionality.

Void FirePlaybackAtLevel(titanId Id, level Level, bool Refire)

Fires a playback with the specified level. This function forces the playback to a level and always loads it if it's not loaded. If refire is set then it kills it before firing.

#### Parameters:

Id – Id of the playback to fire.

Level – the level to set the playback.

Refire – determines if the playback should refire if already loaded

#### Example:

`http://{IP_Address}:4430/titan/script/Playbacks/FirePlaybackAtLevel?titanId=1684&level=1.0&bool=false`

Void KillPlayback(titanId Id)

Kills the specified playback.

#### Parameters:

Id – The Id of the playback to delete.

#### Example:

`http://{IP Address}:4430/titan/script/Playbacks/KillPlayback?titanId=1684`

Void ToggleLatchPlayback(titanId Id)

displays the contents of a cue. In the case of a cue list this would result in moving on to the next cue and displaying its contents.

#### Parameters:

Id – The playback Id of playback to toggle

#### Example:

`http://{IP Address}:4430/titan/script/Playbacks/ToggleLatchPlayback?titanId=1684`

Void KillAllPlaybacks()

Kills all playbacks.

**Example:**

http://{IP Address}:4430/titan/script/Playbacks/KillAllPlaybacks

## Titan

The Titan Provider manages device connection information.

### DeviceInfo

DeviceInfo is a property of the Titan provider. Reporting the current connection status.

**Example:**

http://{IP Address}:4430/titan/get/Titan/DeviceInfo

**Returns:**

Returns the connections status. In JSON format.