visual engineering
LIGHTWARE


# SUMMARY OF LIGHTWARE'S REST API

## Introduction

The Lightware REST API is designed to provide HTTP server services, this way, UCX series switchers can be controlled by HTTP requests. REST API is a software architectural style based on HTTP protocol, so it can be used via web browser, Node.js or with terminal programs.

The Lightware REST API is not a new protocol, the LW3 tree structure became available via HTTP(s). LW3 protocol consists of read-only, read-write properties and methods which operate the same way as REST API GET/POST methods.


## Advantages of the Lightware REST API


**Provides standard HTTP(S)-based control interface**

**Easily accessible from various 3rd-party clients**

Including JS in embedded webpage, Node.js, etc.

**LW3 tree structure became available**

Not a re-definition of LW3 semantics, rather a mapping between LW3 commands and REST API requests.


**Network Security over HTTPS**

Requests and responses can be transferred via HTTPS (443 port). HTTPS protocol encrypts clear text, so it becomes incomprehensible for a third-party.


**Network Security with Basic Authentication**

To limit user access for HTTP/HTTPS server services, username and password authentication can be enforced.


**Serial Message sending**

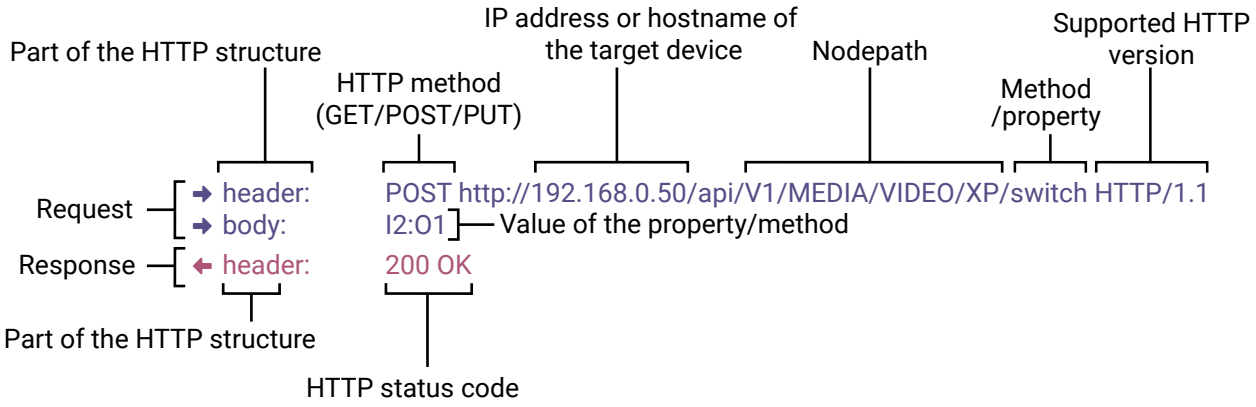
RS-232 message sending up to 100 kB is available via REST API.

## Protocol Rules

- All names and parameters are case-insensitive.
- The methods, nodes and properties are separated by a slash ('/') character.
- The node name consists of letters of the English alphabet and numbers.
- All properties and methods are available HTTP(S) below **/api** as an URL.
- The HTTP server is available on port 80, the HTTPS server is available on port 443.
- GET / PUT/ POST methods are supported.
- Header contains the IP address (or hostname) and the nodepath.
- Arguments and property values should be given in the HTTP request's body as a plain text.
- REST API - LW3 converter does escaping automatically.
- The node paths describe the exact location of the node, listing each parent node up to the root.
- The supported HTTP protocol: standard HTTP/1.1
- There is no maxiumum size or character length of the request.

## Command Structure

### HTTP Request




The examples below show, how to apply the REST API in different environments:

### Curl in Command Line Terminal

```
➜ curl -X POST -i 'http://192.168.0.55/api/V1/MEDIA/VIDEO/XP/switch' --data I2:01
```

### REST API Client in Mozilla

**Method:** POST
**URL:** http://192.168.0.55/api/V1/MEDIA/VIDEO/XP/switch
**Body content type:** text/plain
**Body:** I2:01

## Lightware REST API vs. LW3 Protocol

All the methods and properties of the LW3 tree structure appear below /api as a HTTP(S) URLs. The separator character is always slash ('/') character instead of point ('.') and colon (':'). The URL is case-insensitive.

Some example requests are listed in the Lightware REST API Commands (Extraction) chapter. Other commands can be inferred by the LW3 tree structure, where the read-only (eg. pr /V1/MEDIA/VIDEO/I5.Connected) and read-write properties (pw /V1/MEDIA/VIDEO/I5.Name) can be listed. For more details about LW3 tree structure, see Lightware's Open API Environment white paper on Lightware website (https://lightware.com/our-open-api-environment).

## HTTP Methods

- **GET:** GET method can be used to get the value of a property. (It works the same way as LW3 GET command.)
- **POST / PUT:** In this case, POST and PUT are equivalent, they are for modifying read-write properties or invoke methods. (They replace LW3 SET and CALL commands.)

## Supported Commands

### Query property value (GET)

The requested value is in the body of the response.

REST API Example

➜ header: **GET**•<ip>**/api/**<NodePath>**/**<PropertyName>•**HTTP/1.1**
➜ header: GET http://192.168.0.1/api/V1/MEDIA/VIDEO/XP/I2/SignalPresent HTTP/1.1

LW3 Example

▸ **GET**•**/**<NodePath>**.**<PropertyName>
▸ GET /V1/MEDIA/VIDEO/XP/I2.SignalPresent

### Set property value (SET)

The desired property value should be given as a plain text in the body of the request. The new value is in the body of the response.

REST API Example

➜ header: **POST**•<ip>**/api/**<NodePath>**/**<PropertyName>•**HTTP/1.1**
➜ body: <new_value>

➜ header: POST http://192.168.0.1/api/V1/MEDIA/VIDEO/XP/I2/Mute HTTP/1.1
➜ body: false

LW3 Example

▸ **SET**•**/**<NodePath>**.**<PropertyName>**=**<new_value>
▸ SET /V1/MEDIA/VIDEO/XP/I2.Mute=false

### Invoke method (CALL)

The argument should be given in the body of the request.

REST API Example

➜ header: **POST**•<ip>**/api/**<NodePath>**/**<PropertyName>•**HTTP/1.1**
➜ body: <new_value>

➜ header: POST http://192.168.0.1/api/V1/MEDIA/VIDEO/XP/switch HTTP/1.1
➜ body: I5:01

LW3 Example

- **CALL•/**<NodePath>**:**<MethodName>**=**<value>
- CALL /V1/MEDIA/VIDEO/XP:switch(I5:01)

## Not Supported Commands

### Query node (GET)

REST API Example
Not supported (404 error code)

LW3 Example

- **GET•/**<NodePath>
- CALL /V1/MEDIA/VIDEO

### Subscribe to a node (OPEN)

REST API Example
Not interpreted (Not supported)

LW3 Example

- **OPEN•/**<NodePath>
- OPEN /V1/MEDIA/VIDEO

For subscription, you can use Websocket or WSS or the unencrypted port 6107 LW3.

### Unsubscribe from a Node (CLOSE)

REST API Example
Not interpreted (Not supported)

LW3 Example

- **CLOSE•/**<NodePath>
- CLOSE /V1/MEDIA/VIDEO

## Status Codes, Error Messages

The standard HTTP response codes are defined to supply information about the response and the executed command like:

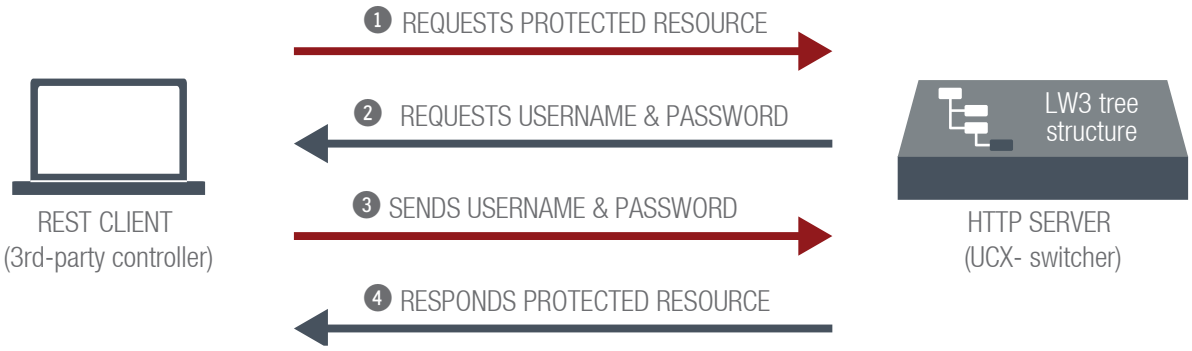
- **200 OK:** Standard response for successful HTTP request.
- **405 Method Not Allowed:** A request method is not supported for the requested resource. This is the error code when trying to modify a read-only property.
- **404 Not Found:** Invalid nodepath or property name.
- **406 Not Acceptable:** LW3 server returned an error for POST and PUT method (equals the following LW3 error codes: pE - an error for the property, mE - an error for a method).
- **500 Internal Server Error:** All other errors (Lw3ErrorCodes_InternalError).

## REST API Security

### Authentication

To limit the user access for the HTTP/HTTPS server services, basic authentication can be turned on ports 80 and 443 separately.

The picture below illustrates the successful authentication process:




USER

- The current implementation can manage one user (with fixed username: admin) with full access.

PASSWORD

- No password is set for default, authentication can be enabled after setting a password.
- The following characters are allowed: Letters (A-Z) and (a-z) and numbers (0-9). Max length: 100 characters.
- The device does not store the password string, so it can not be queried.
- The password can be reset by calling factory defaults by a button press on the front panel.
- The password will not be encrypted by this standard HTTP authentication mode, it remains accessible when the communication happens on HTTP.


Peterdy 15, Budapest H-1071, Hungary +36 1 255 3800 sales@lightware.com www.lightware.com

Follow the instructions to set the authentication on the HTTP (80) port:

1. Set the password.
➔ header: POST http://<ip>/api/V1/MANAGEMENT/NETWORK/AUTHENTICATION/setPassword HTTP/1.1
➔ body: <password>

2. Enable the basic authentication on the HTTP port.
➔ header: POST http://<ip>/api/V1/MANAGEMENT/NETWORK/SERVICES/HTTP/AuthenticationEnabled HTTP/1.1
➔ body: true

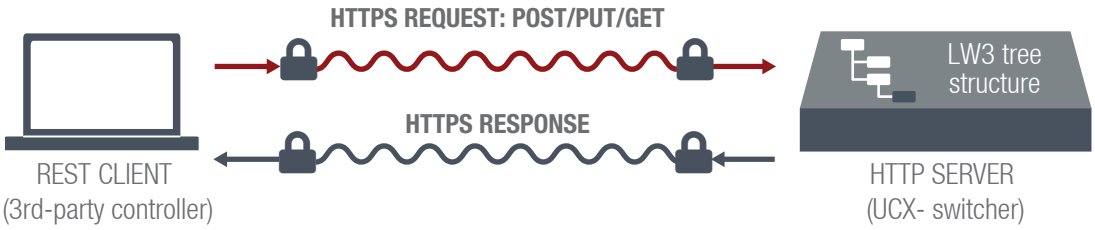
3. Restart HTTP services.
➔ header: POST /V1/MANAGEMENT/NETWORK/SERVICES/HTTP/restart HTTP/1.1

### Encryption (HTTPS)

HTTP protocol uses clear text format for data transport. This method allows a third-party to listen in and eavesdrop on the transferred information. To ensure secure data transmission, the HTTP port (80) can be disabled, and all the information can be transferred via HTTPS (443 port). HTTPS protocol encrypts clear text, so it becomes incomprehensible for a third-party and keeps the data secure.

- The UCX series switcher generates a self-signed certificate, so the user does not have to deal with its configuration.
- A new certificate is generated when the hostname changes or the factory default settings are restored.
- Please check the proper date/time setting of the UCX unit for the certificate acceptance.

HTTPS request-response




## Lightware REST API Commands (Extraction)

### Switch Video Input

REST API Example

➔ header: POST http://192.168.0.50/api/V1/MEDIA/VIDEO/XP/switch HTTP/1.1
➔ body: I5:01;I3:02

➔ header: 200 OK

Curl Example

➔ curl -X POST -i 'http://192.168.0.50/Apl/V1/MEDIA/VIDEO/XP/switch' --data 'I5:01;I3:02'

*Explanation: Input 5 is switched to Output 1 and Input 3 is switched to Output 2. The whole crosspoint can be changed by sending this command as more switching parameters is separated by semicolon.*

### Query the Video Signal Presence

REST API Example

➜ header: GET http://192.168.0.50/api/V1/MEDIA/VIDEO/I1/SignalPresent HTTP/1.1

➜ header: 200 OK
➜ body: true

Curl Example

➜ curl -X GET -i 'http://192.168.0.55/api/V1/MEDIA/VIDEO/I1/SignalPresent'

*Explanation: 'True' means active video signal, 'False' displays no signal presence.*

### Query the Connected Source

REST API Example

➜ header: GET http://192.168.0.50/api/V1/MEDIA/VIDEO/XP/O2/ConnectedSource HTTP/1.1

➜ header: 200 OK
➜ body: I1

Curl Example

➜ curl -X GET -i 'http://192.168.0.55/api/V1/MEDIA/VIDEO/XP/O2/ConnectedSource'

*Explanation: The response shows, that Input 1 is connected to Output 2.*

### Set Displayport Alternate Mode Policy

REST API Example

➜ header: POST http://192.168.0.50/api/V1/MEDIA/USB/U2/DpAltModePolicy HTTP/1.1
➜ body: Force C

➜ header: 200 OK
➜ body: Force C

Curl Example

➜ curl -X POST -i 'http://192.168.0.55/api/V1/MEDIA/USB/U2/DpAltModePolicy' --data 'Force C'

*Explanation: Force C means prefer video: all the four lanes reserved for video transmission, USB 3.1 data transmission does not operate.*

### Restart Link Training

REST API Example

➜ header: POST http://192.168.0.50/api/V1/MEDIA/VIDEO/I1/DP/restartLinkTraining HTTP/1.1

➜ header: 200 OK

Curl Example

➜ curl -X POST -i 'http://192.168.0.55/api/V1/MEDIA/VIDEO/I1/DP/restartLinkTraining'

*Explanation: This method is equal with pulling out and plug in again the USB-C connector.*



Peterdy 15, Budapest H-1071, Hungary +36 1 255 3800 sales@lightware.com www.lightware.com

### Sending a Message via RS-232

REST API Example

➜ header: POST http://192.168.0.50/api/V1/MEDIA/SERIAL/P1/send
➜ body: PWR0

➜ header: 200 OK

Curl Example

➜ curl -X POST -i 'http://192.168.0.55/api/V1/MEDIA/SERIAL/P1/send' --data PWR0

*Explanation: The 'PWR0' message is sent out via the P1 serial port.*

*In UCX series devices, message sending via RS-232 is implemented only in REST API, but not in LW3 protocol.*

### Restore the Factory Default Settings

REST API Example

➜ header: POST http://192.168.0.50/api/V1/SYS/factoryDefaults HTTP/1.1

➜ header: 200 OK

Curl Example

➜ curl -X POST -i 'http://192.168.0.55/api/V1/SYS/factoryDefaults'

*Explanation: The device is restarted, current connections are terminated, and the default settings are restored.*

### Change the IP Address (Static)

REST API Example

➜ header: POST http://192.168.0.50/api/V1/MANAGEMENT/NETWORK/StaticIpAddress HTTP/1.1
➜ body: 192.168.0.100

➜ header: 200 OK
➜ body: 192.168.0.100
➜ header: POST http://192.168.0.50/api/V1/MANAGEMENT/NETWORK/applySettings HTTP/1.1

➜ header: 200 OK
➜ body: OK

Curl Example

➜ curl -X POST -i 'http://192.168.0.55/api/V1/MANAGEMENT/NETWORK/StaticIpAddress ' --data '192.168.0.100'
➜ curl -X POST -i 'http://192.168.0.55/api/V1/MANAGEMENT/NETWORK/applySettings'

*Explanation: A static IP address is set to 192.168.0.100; it will be valid only if the 'DhcpEnabled' property is set to 'false'. The applySettings method will save and apply the new value, the current connections are terminated.*


Peterdy 15, Budapest H-1071, Hungary +36 1 255 3800 sales@lightware.com www.lightware.com