Modator 2U

Overview

Chassis

General Settings

Network Settings

Firmware Update

User Management

Log Management

Universal Interfaces

Certificate

Introduction

For Modator 2U, we have rich APIs for developers to interact with products such as obtaining basic information about the device (device name, firmware version and etc.), modifying device configuration and upgrading firmware. These APIs are based on the HTTP protocol and are lightweight, connectionless interfaces that respond to data in JSON format. This document gives you a detailed understanding of each API's functions and request method.

APIs in this document apply to these products:

- Modator 2U

API Agreement

Introduction

This chapter introduces the elements of the request data and returned data of the API to help you understand how to initiate requests and how to interpret the response data.

- **Request Protocol**: HTTP
- **Request Method**: By default, data requests, submissions, and file uploads are done using the POST method. Some APIs also support the GET method, such as Check factory reset permission, determined by each functional interface.
- **Request Domain**: The device network card IP (IPv4) address, such as `http://192.168.66.1`
- **Request Headers**: Provides additional information about the request via HTTP Header. The following is an example of what is used in the API requests.
  - `Content-Type` : The content type, used to define the type and encoding of network files and web pages.

    The API mainly includes the following two syntax formats:

    1. `Content-Type: application/json; charset=utf-8` , used for data requests or submissions, encoded in UTF-8 and encapsulated in JSON format.
    2. `Content-Type: multipart/form-data; boundary=${boundary}` , used for file uploads, supporting image, video, music, and firmware.
  - `Cookie` : Data stored on the user's local terminal, used in the API to identify user identity, such as `Cookie: sid-A506220808450=6062n8wqjz9mm7m7op91a05j58bup424`
- **Request Body**: Allows additional data to be passed to the API in the form of HTTP Body. These parameters can be required or optional, determined by each functional interface.
- **Response Body**: Response data in the form of HTTP Body. When the HTTP status is 200, it returns data in JSON format; otherwise, it is the corresponding HTTP error code.
- **Login Authentication Method**: Carry sid-xx=xxxxxxxxx in the Cookie. Some APIs do not require authentication, such as: Ping test.
- **[xxx]**: Substitute identifier for a type of data.
  - [IP], the IP address of the device's network card, which should be replaced with actual content when used, such as: 192.168.66.1
  - [port], port number, which should be replaced with actual content when used, such as: 8080.
  - [serial number], the device's serial number, which should be replaced with actual content when used, such as: A506220808450.

Submitting Requests

Data Requests and Submissions

When making data requests and submissions, the attached data should be encoded in UTF-8 and encapsulated in JSON format.

Example 1: Adding a user (Login authentication required)

Request Headers

```
POST / HTTP/1.1
Host: 192.168.66.1/api/user/add
Content-Type: application/json; charset=utf-8
Cookie: sid-A506220808450=6062n8wqjz9mm7m7op91a05j58bup424
```

Request Body

```
{
    "username":"test",
    "password":"c4ca4238a0b923820dcc509a6f75849b"
}
```

Example 2: Testing device network connection (No login authentication required)

Request Headers

```
POST / HTTP/1.1
Host: 192.168.66.1/api/ping
Content-Type: application/json; charset=utf-8
```

Request Body

```
None
```

File Upload

When uploading files, the attachment data should be encapsulated in binary form, for example:

Request Headers

```
POST / HTTP/1.1
Host: 192.168.66.1/mwapi/upload-source-file
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryuCSyGCncblrUp3ed
Cookie: sid-A506220808450=6062n8wqjz9mm7m7op91a05j58bup424
```

Request Body

```
------WebKitFormBoundaryuCSyGCncblrUp3ed
Content-Disposition: form-data; name="file"; filename="204.jpeg"
Content-Type: application/octet-stream
XXXX
XXXX
------WebKitFormBoundaryuCSyGCncblrUp3ed--
```

Response

All response data is in JSON format.

The `status` / `result` attribute in the JSON object is the Common Error Codes. A value of `0` indicates that the data retrieval or operation was successful, otherwise, it represents the corresponding error code for failure.

Example 1: Single-level data encapsulation

```
{
  "status": 0,
  "code": "Success",
  "enable": true,
  "enable-web-control": true
}
```

Example 2: Data encapsulation with secondary objects

```
{
  "result": 0,
  "info": {}
}
```

Example 3: Data encapsulation with secondary objects as arrays

```
{
  "status": 0,
  "items": []
}
```

Example 4: Data encapsulation with business errors

```
{
  "status": 16,
  "message": "Item is not exist."
}
```

Login Authentication and API Access Verification

To ensure system security, most of API calls in this doc require authentication, and some APIs are accessible only to administrators, such as: Add a user and Delete a user.

Login Authentication

Login using the `username` and `password` .

Upon successful login, the Session ID will be stored in the Cookie:

`Cookie: sid-[serial number]=6440wa6u5wfw8wv43f91v55cqkctnpv6`

The Session ID remains valid until the device is turned off or restarted.

Request Headers

```
POST / HTTP/1.1
Host: 192.168.66.1//api/user/login
Content-Type: application/json; charset=utf-8
```

Request Body

```
{
    "username":"test",
    "password":"c4ca4238a0b923820dcc509a6f75849b"
}
```

Response Headers

```
Content-Type: application/json; charset=utf-8
Expires: 0
Set-Cookie: sid-A506220808450=6062n8wqjz9mm7m7op91a05j58bup424
```

Response Body

```
{
    "status":"test",
    "sid":"6440wa6u5wfw8wv43f91v55cqkctnpv6"
}
```

API Access Authentication

When accessing an API that requires authentication, the Session ID obtained from the login authentication interface must be transmitted in the Cookie.

Request Headers

```
POST / HTTP/1.1
Host: 192.168.66.1//api/user/add
Content-Type: application/json; charset=utf-8
Cookie: sid-A506220808450=6062n8wqjz9mm7m7op91a05j58bup424
```

Testing Tools

In different operating systems, you can install the tools wget and curl, and use wget or curl commands in the command line to call the API.

The locations where the cookie files are stored vary in different operating systems, and please modify according to the actual situation. (The following examples are based on Linux.)

wget

1. Login and save cookies

```
wget --save-cookies=sid.txt --keep-session-cookies --header="Content-Type: application/json" --post-data='{"usernam
e":"Admin", "password": "c1c224b03cd9bc7b6a86d77f5dace40191766c485cd55dc48caf9ac873335d6f"}' http://192.168.66.1/api/
user/login -d -q -O -
```

1. Get the user list

```
wget --load-cookies=sid.txt --keep-session-cookies --header="Content-Type: application/json" --post-data='' http://19
2.168.66.1/api/user/get-all -d -q -O -
```

1. Add a user

```
wget --load-cookies=sid.txt --keep-session-cookies --header="Content-Type: application/json" --post-data='{"usernam
e":"test","password":"9f86d081884c7d659a2feaa0c55ad015a3bf4f1b2b0b822cd15d6c15b0f00a08"}' http://192.168.66.1/api/use
r/add -d -q -O -
```

curl

1. Login and save cookies

```
curl --cookie-jar sid.txt http://192.168.66.1/api/user/login -X POST -H 'Content-Type: application/json' -d'{"usernam
e":"Admin", "password": "c1c224b03cd9bc7b6a86d77f5dace40191766c485cd55dc48caf9ac873335d6f"}'
```

1. Get the user list

```
curl --cookie sid.txt http://192.168.66.1/api/user/get-all -X POST -H 'Content-Type: application/json' -d ''
```

1. Add a user

```
curl --cookie sid.txt http://192.168.66.1/api/user/add -X POST -H 'Content-Type: application/json' -d '{"username":"t
est","password":"9f86d081884c7d659a2feaa0c55ad015a3bf4f1b2b0b822cd15d6c15b0f00a08"}'
```

Common Error Codes

Status	Definition	Code	Description
0	MW_STATUS_SUCCESS	Success	Successful operation
1	MW_STATUS_PENDING	Pending	Operation is pending
2	MW_STATUS_TIMEOUT	Timeout	Operation timed out
3	MW_STATUS_INTERRUPTED	Interrupted	Operation was interrupted
4	MW_STATUS_TRY_AGAIN	TryAgain	Operation should be retried
5	MW_STATUS_NOT_IMPLEMENTED	NotImplemented	The operation is not implemented
6	MW_STATUS_UNKNOWN_ERROR	UnknownError	An unknown error occurred
7	MW_STATUS_INVALID_ARG	InvalidArgument	Invalid argument provided
8	MW_STATUS_NO_MEMORY	OutOfmemory	Insufficient memory
9	MW_STATUS_UNSUPPORTED	Unsupported	The operation is not supported
10	MW_STATUS_FILE_BUSY	FileBusy	File system is busy
11	MW_STATUS_DEVICE_BUSY	DeviceBusy	Device is busy
12	MW_STATUS_DEVICE_LOST	DeviceLost	Device is lost
13	MW_STATUS_IO_FAILED	IOError	An IO error occurred
14	MW_STATUS_READ_FAILED	ReadError	IO read failed
15	MW_STATUS_WRITE_FAILED	WriteError	IO write failed
16	MW_STATUS_NOT_EXIST	NotExist	The content does not exist
17	MW_STATUS_TOO_MANY	TooMany	Exceeded the limit of count
18	MW_STATUS_TOO_LARGE	TooLarge	Exceeded the size limit
19	MW_STATUS_OVERFLOW	Overflow	Overflow (up)
20	MW_STATUS_UNDERFLOW	Underflow	Overflow (down)
21	MW_STATUS_FORMAT_ERROR	FormatError	Format error occurred
22	MW_STATUS_FILE_EXISTS	FileExists	File already exists
23	MW_STATUS_FILE_TYPE_ERROR	FileTypeError	Incorrect file type
24	MW_STATUS_DEVICE_TYPE_ERROR	DeviceTypeError	Incorrect device type
25	MW_STATUS_IS_DIRECTORY	IsDirectory	The content is a directory
26	MW_STATUS_READ_ONLY	ReadOnly	Read-only restriction
27	MW_STATUS_RANGE_ERROR	OutOfRange	Range error occurred
28	MW_STATUS_BROKEN_PIPE	BrokenPipe	Pipe connection was interrupted
29	MW_STATUS_NO_SPACE	NoSpace	Insufficient space
30	MW_STATUS_NOT_DIRECTORY	NotDirectory	Not a directory
31	MW_STATUS_NOT_PERMITTED	NotPermitted	Forbidden operation
32	MW_STATUS_BAD_ADDRESS	BadAddress	Invalid address
33	MW_STATUS_SEEK_ERROR	SeekError	Seek error occurred
34	MW_STATUS_CROSS_DEVICE_LINK	CrossDeviceLink	Cross-device link error
35	MW_STATUS_NOT_INITIALIED	NotInitialized	Not initialized

Status	Definition	Code	Description
36	MW_STATUS_AUTH_FAILED	AuthFailed	Authentication failed
37	MW_STATUS_NOT_LOGGED_IN	NotLoggedIn	Not logged in
38	MW_STATUS_WRONG_STATE	WrongState	Incorrect state
39	MW_STATUS_MISMATCH	Mismatch	Mismatch
40	MW_STATUS_VERIFY_FAILED	VerifyFailed	Verification failed
41	MW_STATUS_CONSTRAINT_VIOLATION	ConstraintViolatin	Constraint violation
42	MW_STATUS_CANCELED	Canceled	Operation was canceled
43	MW_STATUS_IN_PROGRESS	InProgress	Operation is in progress
44	MW_STATUS_CONN_REFUSED	ConnectionRefused	Connection refused
45	MW_STATUS_CONN_RESET	ConnectionReset	Connection reset
46	MW_STATUS_ADDR_IN_USE	AddressInUse	Address is in use
47	MW_STATUS_NO_RESPONSE	NoResponse	No response
48	MW_STATUS_INFO_CHANGED	InfoChanged	Information has changed
49	MW_STATUS_INVALID_DATA	InvalidData	Invalid data
50	MW_STATUS_NEED_MORE_DATA	NeedMoreData	More data is needed
51	MW_STATUS_NO_BUFFER	NoBuffer	Buffer is exhausted
52	MW_STATUS_BUFFER_TOO_SMALL	BufferTooSmall	Buffer is too small
53	MW_STATUS_BUFFER_IS_EMPTY	BufferIsEmpty	Buffer is empty
54	MW_STATUS_BUFFER_IS_FULL	BufferIsFull	Buffer is full

Glossary

HTTP: Hypertext Transfer Protocol, a protocol used for distributed, collaborative, and hypertext information systems at the application layer. It is the foundation of data communication on the World Wide Web (WWW).

GET: An HTTP request method, commonly used to retrieve data from the server. It is characterized by small amounts of data and parameters passed through the URL.

POST: An HTTP request method, commonly used to submit data to the server. POST requests place data in the request body, allowing for larger amounts of data and offering higher security.

HTTP Header: A part of the metadata transmitted in HTTP requests and responses. They are key-value pairs that contain metadata about the client, server, request, and response. Common HTTP headers include:

- Content-Type: Specifies the media type of the message body.
- Content-Length: Specifies the byte length of the message body.
- Cookie: Transmits cookies to the server.
- Set-Cookie: Sets cookies on the client.

HTTP Body : The part of the message body in an HTTP request or response. The body type can be any type of data, such as text, JSON, XML, HTML, images and videos. This system usually uses JSON.

JSON (JavaScript Object Notation): A lightweight data exchange format based on text. JSON is language-independent, stored and represented in a human-readable and machine-parsable text format, which also facilitates machine generation. It effectively enhances network transmission efficiency. JSON data format uses a "name/value pair" approach and supports the representation of arrays.

sha256: sha256 is a widely used hashing algorithm for generating digital fingerprints, commonly used in cryptography and digital integrity verification fields.

Version History

Version

Chassis

Updated API

- get-card
  - New output parameters: stream
  - Modified output parameters: device, ndi

Get the fan control board info

1. API Description

This API is used to get the information of fan control board.

Request mode: POST [ip]/api/chassis/get-fan

Administrator Rights	Logged-in
No	Yes

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
device-name	String	The device name
fan-num	Int	The number of fans on the fan control board
sensor-num	Int	The number of temperature sensors on the fan control board
state	Int	The status of the fan control board 0: Not powered on 1: Communication failure 2: Communication success 3: Firmware upgrade in progress
fan	Array of FanData	The fan list
sensor	Array of SensorData	The sensor list

FanData

Name	Type	Description
id	Int	The fan ID
speed	Int	The fan speed, in rpm
status	Int	Status code

SensorData

Name	Type	Description
id	Int	The sensor ID
temperature	Double	Temperature read by the temperature sensor, in degrees Celsius
status	Int	Status code

4. Example

Getting the information of the fan control board.

Input Example

```
None
```

Output Example


```
{
    "device-name": "Modator 2U FAN CONTROL",
    "fan-num": 5,
    "sensor-num": 3,
    "state": 2,
    "fan": [
        {
            "id": 1,
            "speed": 2250,
            "status": 0
        },
        {
            "id": 2,
            "speed": 0,
            "status": 6
        },
        {
            "id": 3,
            "speed": 0,
            "status": 6
        },
        {
            "id": 4,
            "speed": 2430,
            "status": 0
        },
        {
            "id": 5,
            "speed": 2100,
            "status": 0
        }
    ],
    "sensor": [
        {
            "id": 1,
            "temperature": 28.125,
            "status": 0
        },
        {
            "id": 2,
            "temperature": 27.0,
            "status": 0
        },
        {
            "id": 3,
            "temperature": 27.25,
            "status": 0
        }
    ],
    "status": 0,
    "code": "Success"
}

{
    "device-name": "Modator 2U FAN CONTROL",
    "fan-num": 5,
    "sensor-num": 3,
    "state": 0,
    "status": 0,
```

```
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
4	MW_STATUS_TRY_AGAIN	Array of FanData, try to start the fan again

Get the power board info

1. API Description

This API is used to get the information of the power board connected with the chassis.

Request mode: POST [ip]/api/chassis/get-backplane

Administrator Rights	Logged-in
No	Yes

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
device-name	String	The device name
power-num	Int	The number of power modules
state	Int	The current status of the power board 0: Communication failure 1: Communication successful 2: Firmware upgrade in progress
power	Array of PowerData	The list of power modules

PowerData

Name	Type	Description
id	Int	The ID of power module
state	String	The status of power module, including ON, OFF, ERROR, No AC power and Power module disconnected

4. Example

Getting the information of power board.

Input Example

```
None
```

Output Example


```
{
    "device-name": "Modator 2U POWER BOARD",
    "power-num": 2,
    "state": 2,
    "power": [
        {
            "id": 1,
            "state": "OFF"
        },
        {
            "id": 2,
            "state": "OFF"
        }
    ],
    "status": 0,
    "code": "Success"
}
```

```
{
    "device-name": "Modator 2U POWER BOARD",
    "power-num": 2,
    "state": 0,
    "status": 0,
    "code": "Success"
}
```


5. Error Code

No error code related to the API business logic. For other error codes, see Common Error Codes.

Get alert info

1. API Description

This API is used to get the alert information of the chassis.

Request mode: POST [ip]/api/chassis/get-alert

Administrator Rights	Logged-in
No	Yes

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
alert-num	Int	The total number of alerts
alert	Array of AlertData	Alert list

AlertData

Name	Type	Description
index	Int	The index of alerts
information	String	Alert information

4. Example

Getting the alert information of the device.

Input Example

```
None
```

Output Example


```
{
    "alert-num": 3,
    "alert": [
        {
            "index": 1,
            "information": "2023/05/29 19:38:18\npower1 DC output below minimum detection level"
        },
        {
            "index": 2,
            "information": "2023/05/29 19:38:18\nfan 2 failed to start and stopped rotating"
        },
        {
            "index": 3,
            "information": "2023/05/29 19:38:18\nfan 3 failed to start and stopped rotating"
        }
    ],
    "status": 0,
    "code": "Success"
}
```


5. Error Code

No error code related to the API business logic. For other error codes, see Common Error Codes.

Get the module info

1. API Description

This API is used to get the information of modules.

Request mode: POST [ip]/api/chassis/get-card

Administrator Rights	Logged-in
No	Yes

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
card-num	Int	The number of slots
card	Array of CardData	The module list

CardData

Name	Type	Description
position	Int	The slot number where the module is inserted.
connect	Boolean	Whether the module is connected true: Yes; false: No
device	Array of device	Refer to Basic Information of Decoder Basic Information of Encoder Basic Information of Ultra Encode Plus
ndi	Array of ndi	Refer to NDI Status of Decoder NDI Status of Encoder No such info. available for Ultra Encode Plus
stream	Array of stream	Refer to Stream Status of Ultra Encode Plus Only available for Ultra Encode Plus

4. Example

Getting the information of modules.

Input Example

```
None
```

Output Example


```
{
    "card-num": 10,
    "card": [
        {
            "position": 1,
            "connect": true,
            "module-type": 0,
            "device": {
                "model": "NDI to HDMI",
                "product-id": 1040,
                "serial-no": "Z410221102009",
                "hw-revision": "Z",
                "fw-version": "1.1.1029",
                "board-id": 1,
                "use-https": false,
                "device-name": "NJ-play05",
                "output-state": "connected",
                "cpu-usage": 18.920000000000002,
                "memory-usage": 22.528959274291992,
                "core-temp": 36.360000610351562,
                "up-time": 330195,
                "ip-addr": "10.10.8.110"
            },
            "ndi": {
                "name": "ULTRA ENCODE (Z316241118001)",
                "url": "",
                "connected": true,
                "tally-preview": false,
                "tally-program": false,
                "audio-drop-frames": 0,
                "video-drop-frames": 0,
                "video-bit-rate": 10261,
                "audio-bit-rate": 3139,
                "audio-jitter": 17,
                "video-jitter": 28,
                "video-width": 1920,
                "video-height": 1080,
                "video-scan": "progressive",
                "video-field-rate": 60.0,
                "audio-num-channels": 2,
                "audio-sample-rate": 48000,
                "audio-bit-count": 16
            }
        },
        {
            "position": 2,
            "connect": true,
            "module-type": 0,
            "device": {
                "model": "NDI to HDMI 4K",
                "product-id": 1041,
                "serial-no": "Z411240924002",
                "hw-revision": "Z",
                "fw-version": "1.2.140",
                "board-id": 2,
                "use-https": false,
                "device-name": "Pro Convert",
                "output-state": "unconnected",
                "cpu-usage": 13.16,
                "memory-usage": 20.660823822021484,
```

```
            "core-temp": 31.110000610351562,
            "up-time": 68980,
            "ip-addr": "10.10.14.32"
        },
        "ndi": {
            "name": "ULTRA ENCODE (Z316241118001)",
            "url": "",
            "connected": true,
            "tally-preview": false,
            "tally-program": false,
            "audio-drop-frames": 0,
            "video-drop-frames": 0,
            "video-bit-rate": 10959,
            "audio-bit-rate": 3070,
            "audio-jitter": 10,
            "video-jitter": 28,
            "video-width": 1920,
            "video-height": 1080,
            "video-scan": "progressive",
            "video-field-rate": 60.0,
            "audio-num-channels": 2,
            "audio-sample-rate": 48000,
            "audio-bit-count": 16
        }
    },
    {
        "position": 3,
        "connect": true,
        "module-type": 1,
        "device": {
            "model": "HDMI 4K Plus",
            "product-id": 1025,
            "serial-no": "Z401240701005",
            "hw-revision": "Z",
            "fw-version": "1.1.342",
            "board-id": 3,
            "use-https": false,
            "device-name": "Pro Convert",
            "output-state": "unconnected",
            "cpu-usage": 26.32,
            "memory-usage": 60.42,
            "core-temp": 42.45,
            "up-time": 68998,
            "ip-addr": "10.10.15.38"
        },
        "ndi": {
            "name": "#03 (Z401240701005)",
            "enabled": true,
            "num-clients": 2,
            "tally-preview": false,
            "tally-program": false,
            "audio-drop-frames": 0,
            "video-drop-frames": 0,
            "video-bit-rate": 140513,
            "audio-bit-rate": 1413,
            "video-width": 1920,
            "video-height": 1080,
            "video-scan": "progressive",
            "video-field-rate": 60.00,
            "audio-num-channels": 2,
            "audio-sample-rate": 44100,
            "audio-bit-count": 16
```

```
            }
        },
        {
            "position": 4,
            "connect": true,
            "module-type": 2,
            "device": {
                "model": "Ultra Encode SDI Plus",
                "product-id": 790,
                "serial-no": "Z316241118001",
                "hw-revision": "Z",
                "fw-version": "2.3.254",
                "board-id": 4,
                "use-https": false,
                "device-name": "Ultra Encode Z316241118001",
                "output-state": "loop-through",
                "cpu-usage": 0.17,
                "memory-usage": 47.73,
                "core-temp": 44.90,
                "up-time": 56129,
                "ip-addr": "10.10.11.150"
            },
            "stream": {
                "codec": {
                    "main-stream": {
                        "result": 0,
                        "cx": 1920,
                        "cy": 1080,
                        "duration": 166667,
                        "kbps": 10240,
                        "cur-bps": 9046858,
                        "i-frame-count": 3366681,
                        "o-frame-count": 3366680,
                        "drop-count": 0,
                        "add-count": 0,
                        "check-surface": 1,
                        "rotation": 0
                    },
                    "sub-stream": {
                        "enable": 1,
                        "result": 0,
                        "cx": 1280,
                        "cy": 720,
                        "duration": 333333,
                        "kbps": 2048,
                        "cur-bps": 2161800,
                        "i-frame-count": 1683348,
                        "o-frame-count": 1683347,
                        "drop-count": 0,
                        "add-count": 0,
                        "check-surface": 1,
                        "rotation": 0
                    },
                    "audio": {
                        "channels": 2,
                        "kbps": 128,
                        "cur-bps": 131384,
                        "i-frame-count": 2630229,
                        "o-frame-count": 2630229
                    }
                },
                "live": [
```

```
                {
                    "id": 0,
                    "type": 130,
                    "is-use": 1,
                    "is-skd-runnung": 0,
                    "name": "NDI HX2",
                    "run-ms": 56026839,
                    "result": 22,
                    "on-preview": false,
                    "on-program": false,
                    "main-inst-bps": 1219008,
                    "sub-inst-bps": 257578,
                    "video-frame-count": 3361555,
                    "audio-frame-count": 2626221,
                    "last-video-pts": 561278560518,
                    "last-audio-pts": 561278696666
                },
                {
                    "id": 1,
                    "type": 100,
                    "is-use": 1,
                    "is-skd-runnung": 0,
                    "name": "RTSP Server",
                    "run-ms": 56026804,
                    "result": 22,
                    "max-connections": 0,
                    "enable-main": 1,
                    "main-num-client": 0,
                    "main-clients": [
                    ],
                    "enable-sub": 0,
                    "sub-num-client": 0,
                    "sub-clients": [
                    ],
                    "video-frame-count": 3361555,
                    "audio-frame-count": 2626220,
                    "last-video-pts": 56026872,
                    "last-audio-pts": 56026869
                }
            ]
        }
    },
    {
        "position": 5,
        "connect": true,
        "module-type": 2,
        "device": {
            "model": "Ultra Encode SDI Plus",
            "product-id": 790,
            "serial-no": "Z316241118002",
            "hw-revision": "Z",
            "fw-version": "2.3.254",
            "board-id": 5,
            "use-https": false,
            "device-name": "Ultra Encode Z316241118002",
            "output-state": "loop-through",
            "cpu-usage": 0.09,
            "memory-usage": 44.08,
            "core-temp": 42.70,
            "up-time": 55768,
            "ip-addr": "10.10.4.183"
        },
```

```
            "stream": {
                "codec": {
                    "main-stream": {
                        "result": 0,
                        "cx": 1920,
                        "cy": 1080,
                        "duration": 333333,
                        "kbps": 4096,
                        "cur-bps": 67450,
                        "i-frame-count": 1672572,
                        "o-frame-count": 1672572,
                        "drop-count": 0,
                        "add-count": 0,
                        "check-surface": 1,
                        "rotation": 0
                    },
                    "sub-stream": {
                        "enable": 1,
                        "result": 0,
                        "cx": 1280,
                        "cy": 720,
                        "duration": 333333,
                        "kbps": 2048,
                        "cur-bps": 79154,
                        "i-frame-count": 1672574,
                        "o-frame-count": 1672574,
                        "drop-count": 0,
                        "add-count": 0,
                        "check-surface": 1,
                        "rotation": 0
                    },
                    "audio": {
                        "channels": 2,
                        "kbps": 128,
                        "cur-bps": 131208,
                        "i-frame-count": 2613396,
                        "o-frame-count": 2613396
                    }
                },
                "live": [
                    {
                        "id": 0,
                        "type": 100,
                        "is-use": 1,
                        "is-skd-runnung": 0,
                        "name": "RTSP Server",
                        "run-ms": 55714924,
                        "result": 22,
                        "max-connections": 0,
                        "enable-main": 1,
                        "main-num-client": 0,
                        "main-clients": [
                        ],
                        "enable-sub": 0,
                        "sub-num-client": 0,
                        "sub-clients": [
                        ],
                        "video-frame-count": 1671423,
                        "audio-frame-count": 2611595,
                        "last-video-pts": 55714097,
                        "last-audio-pts": 55714097
                    }
```

```
                ]
            }
        },
        {
            "position": 6,
            "connect": true,
            "module-type": 2,
            "device": {
                "model": "Ultra Encode SDI Plus",
                "product-id": 790,
                "serial-no": "Z316241118003",
                "hw-revision": "Z",
                "fw-version": "2.3.254",
                "board-id": 6,
                "use-https": false,
                "device-name": "Ultra Encode Z316241118003",
                "output-state": "loop-through",
                "cpu-usage": 0.08,
                "memory-usage": 41.98,
                "core-temp": 41.20,
                "up-time": 55722,
                "ip-addr": "10.10.5.242"
            },
            "stream": {
                "codec": {
                    "main-stream": {
                        "result": 0,
                        "cx": 1920,
                        "cy": 1080,
                        "duration": 333333,
                        "kbps": 4096,
                        "cur-bps": 117376,
                        "i-frame-count": 1671162,
                        "o-frame-count": 1671161,
                        "drop-count": 0,
                        "add-count": 0,
                        "check-surface": 1,
                        "rotation": 0
                    },
                    "sub-stream": {
                        "enable": 1,
                        "result": 0,
                        "cx": 1280,
                        "cy": 720,
                        "duration": 333333,
                        "kbps": 2048,
                        "cur-bps": 78928,
                        "i-frame-count": 1671163,
                        "o-frame-count": 1671163,
                        "drop-count": 0,
                        "add-count": 0,
                        "check-surface": 1,
                        "rotation": 0
                    },
                    "audio": {
                        "channels": 2,
                        "kbps": 128,
                        "cur-bps": 131832,
                        "i-frame-count": 2611192,
                        "o-frame-count": 2611192
                    }
                },
```

```
                "live": [
                    {
                        "id": 0,
                        "type": 121,
                        "is-use": 1,
                        "is-skd-runnung": 0,
                        "name": "SRT Listener",
                        "run-ms": 55598200,
                        "result": 22,
                        "stream-index": 0,
                        "max-connections": 1,
                        "num-client": 0,
                        "clients": [
                        ],
                        "video-frame-count": 1667932,
                        "audio-frame-count": 2606141,
                        "last-video-pts": 55597998,
                        "last-audio-pts": 55598026
                    }
                ]
            }
        },
        {
            "position": 7,
            "connect": true,
            "module-type": 0,
            "device": {
                "model": "NDI to AIO",
                "product-id": 1057,
                "serial-no": "Z421230224003",
                "hw-revision": "Z",
                "fw-version": "1.1.1029",
                "board-id": 7,
                "use-https": false,
                "device-name": "NJ-play002  ddddwwwww",
                "output-state": "connected",
                "cpu-usage": 19.050000000000001,
                "memory-usage": 11.753218650817871,
                "core-temp": 48.639999389648438,
                "up-time": 330161,
                "ip-addr": "10.10.14.53"
            },
            "ndi": {
                "name": "ULTRA ENCODE (Z316241118001)",
                "url": "",
                "connected": true,
                "tally-preview": false,
                "tally-program": false,
                "audio-drop-frames": 0,
                "video-drop-frames": 0,
                "video-bit-rate": 10959,
                "audio-bit-rate": 3070,
                "audio-jitter": 17,
                "video-jitter": 27,
                "video-width": 1920,
                "video-height": 1080,
                "video-scan": "progressive",
                "video-field-rate": 60.0,
                "audio-num-channels": 2,
                "audio-sample-rate": 48000,
                "audio-bit-count": 16
            }
```

```
        },
        {
            "position": 8,
            "connect": true,
            "module-type": 2,
            "device": {
                "model": "Ultra Encode SDI Plus",
                "product-id": 790,
                "serial-no": "Z316241118004",
                "hw-revision": "Z",
                "fw-version": "2.3.254",
                "board-id": 8,
                "use-https": false,
                "device-name": "Ultra Encode Z316241118004",
                "output-state": "loop-through",
                "cpu-usage": 0.12,
                "memory-usage": 40.35,
                "core-temp": 43.00,
                "up-time": 55523,
                "ip-addr": "10.10.8.199"
            },
            "stream": {
                "codec": {
                    "main-stream": {
                        "result": 0,
                        "cx": 1920,
                        "cy": 1080,
                        "duration": 166667,
                        "kbps": 63488,
                        "cur-bps": 484464,
                        "i-frame-count": 3322009,
                        "o-frame-count": 3322009,
                        "drop-count": 0,
                        "add-count": 0,
                        "check-surface": 1,
                        "rotation": 0
                    },
                    "sub-stream": {
                        "enable": 1,
                        "result": 0,
                        "cx": 640,
                        "cy": 360,
                        "duration": 333333,
                        "kbps": 3072,
                        "cur-bps": 78477,
                        "i-frame-count": 1661009,
                        "o-frame-count": 1661008,
                        "drop-count": 0,
                        "add-count": 0,
                        "check-surface": 1,
                        "rotation": 0
                    },
                    "audio": {
                        "channels": 2,
                        "kbps": 128,
                        "cur-bps": 131040,
                        "i-frame-count": 2601925,
                        "o-frame-count": 2601925
                    }
                },
                "live": [
                    {
```

```
                    "id": 0,
                    "type": 141,
                    "is-use": 1,
                    "is-skd-runnung": 0,
                    "name": "NDI HX3",
                    "run-ms": 53800343,
                    "result": 22,
                    "on-preview": false,
                    "on-program": false,
                    "main-inst-bps": 61163,
                    "sub-inst-bps": 12841,
                    "video-frame-count": 3227969,
                    "audio-frame-count": 2521855,
                    "last-video-pts": 555234632656,
                    "last-audio-pts": 555234340000
                }
            ]
        }
    },
    {
        "position": 9,
        "connect": true,
        "module-type": 0,
        "device": {
            "model": "NDI to HDMI 4K",
            "product-id": 1041,
            "serial-no": "Z411240924003",
            "hw-revision": "Z",
            "fw-version": "1.2.138",
            "board-id": 9,
            "use-https": true,
            "device-name": "Pro Convert",
            "output-state": "unconnected",
            "cpu-usage": 21.050000000000001,
            "memory-usage": 69.50286865234375,
            "core-temp": 28.75,
            "up-time": 76976,
            "ip-addr": "10.10.6.213"
        },
        "ndi": {
            "name": "111",
            "url": "http:\/\/10.10.12.232:5599\/Sony.mp4?mw-bitrate=4096&mw-buffer-duration=60",
            "connected": true,
            "tally-preview": false,
            "tally-program": false,
            "audio-drop-frames": 0,
            "video-drop-frames": 17,
            "video-bit-rate": 80143,
            "audio-bit-rate": 194,
            "audio-jitter": 9,
            "video-jitter": 21,
            "video-width": 3840,
            "video-height": 2160,
            "video-scan": "progressive",
            "video-field-rate": 50.0,
            "audio-num-channels": 2,
            "audio-sample-rate": 48000,
            "audio-bit-count": 16
        }
    },
    {
        "position": 10,
```

```
            "connect": true,
            "module-type": 0,
            "device": {
                "model": "NDI to AIO",
                "product-id": 1057,
                "serial-no": "Z421230224001",
                "hw-revision": "Z",
                "fw-version": "1.1.1029",
                "board-id": 10,
                "use-https": false,
                "device-name": "Pro Convert",
                "output-state": "connected",
                "cpu-usage": 74.359999999999999,
                "memory-usage": 10.188580513000488,
                "core-temp": 57.729999542236328,
                "up-time": 76085,
                "ip-addr": "10.10.3.60"
            },
            "ndi": {
                "name": "PRO CONVERT (#03 (Z401240701005))",
                "url": "",
                "connected": true,
                "tally-preview": false,
                "tally-program": false,
                "audio-drop-frames": 0,
                "video-drop-frames": 0,
                "video-bit-rate": 131754,
                "audio-bit-rate": 2823,
                "audio-jitter": 6,
                "video-jitter": 3,
                "video-width": 1920,
                "video-height": 1080,
                "video-scan": "progressive",
                "video-field-rate": 60.0,
                "audio-num-channels": 2,
                "audio-sample-rate": 44100,
                "audio-bit-count": 16
            }
        }
    ],
    "status": 0,
    "code": "Success"
}
```


5. Error Code

No error code related to the API business logic. For other error codes, see Common Error Codes.

Get the beeper status

1. API Description

This API is used to get the status of beeper.

Request mode: POST [ip]/api/chassis/get-beeper

Administrator Rights	Logged-in
No	Yes

2. Input Parameters

No

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
beeper-enable	Boolean	Whether the beeper is enabled true: Yes; false: No

4. Example

Getting the status of the beeper.

Input Example

```
None
```

Output Example

```
{
    "beeper-enable": false,
    "status": 0,
    "code": "Success"
}
```

5. Error Code

No error code related to the API business logic. For other error codes, see Common Error Codes.

Set the beeper

1. API Description

This API is used to set the beeper. When it is true, if there is an alert, the beeper beeps; when it is false, if there is an alert, the beeper does not beep.

Request mode: POST [ip]/api/chassis/set-beeper

Administrator Rights	Logged-in
No	Yes

2. Input Parameters

Name	Required	Type	Description
beeper-enable	Yes	Boolean	Whether to enable the beeper true: Yes; false: No

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Setting the beeper.

Input Example

```
{
    "beeper-enable": false
}
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	Missing required parameters

Get the status of power module

1. API Description

This API is used to get the status of the power module.

Request mode: POST [ip]/api/chassis/get-power

Administrator Rights	Logged-in
No	Yes

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
power-on	Int	Power module status 0: Standby 1: Powering on 2: Power on 3: Entering standby status

4. Example

Getting the status of the power module.

Input Example

```
None
```

Output Example

```
{
    "power-on": 2,
    "status": 0,
    "code": "Success"
}
```

5. Error Code

No error code related to the API business logic. For other error codes, see Common Error Codes.

Set the power module

1. API Description

This API is used to set the status of the power module

Request mode: POST [ip]/api/chassis/set-power

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Name	Required	Type	Description
power-on	Yes	Boolean	Whether to power on true: Yes; false: No

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Powering on the power module.

Input Example

```
{
    "power-on": true
}
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
1	MW_STATUS_PENDING	Pending
7	MW_STATUS_INVALID_ARG	Missing required parameters
12	MW_STATUS_DEVICE_LOST	Communication failure between host and power board

Check touchscreen's sleep mode

1. API Description

This API is used to check the sleep mode of the touchscreen. When it is true, if it is not touched within 1 minute, the touchscreen goes to sleep; when it is false, the touchscreen is always on.

Request mode: POST [ip]/api/chassis/get-screen

Administrator Rights	Logged-in
No	Yes

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
screen-sleep	Boolean	Whether the sleep mode of touchscreen is enabled true: Yes; false: No

4. Example

Checking whether the sleep mode of the touchscreen is enabled.

Input Example

```
None
```

Output Example

```
{
    "screen-sleep": false,
    "status": 0,
    "code": "Success"
}
```

5. Error Code

No error code related to the API business logic. For other error codes, see Common Error Codes.

Set touchscreen's sleep mode

1. API Description

This API is used to enable or disable the sleep mode of the touchscreen. When it is true, if it is not touched within 1 minute, the touchscreen goes to sleep; when it is false, the touchscreen is always on.

Request mode: POST [ip]/api/chassis/set-screen

Administrator Rights	Logged-in
No	Yes

2. Input Parameters

Name	Required	Type	Description
screen-sleep	Yes	Boolean	Whether to enable the sleep mode of the touchscreen true: Yes; false: No

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Enabling the sleep mode of the touchscreen.

Input Example

```
{
    "screen-sleep": true
}
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	Missing required parameters

Schedule automatic reboots

1. API Description

This API is used to schedule automatic reboots for your device.

Request mode: POST [ip]/api/system/auto-reboot

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Name	Required	Type	Description
enable	Yes	Boolean	Whether to enable auto-reboot true: Yes; false: No
hour	Yes	Int	Time to auto-reboot, hour
min	Yes	Int	Time to auto-reboot, minute
week	Yes	[Array]	Weekly repeat 1: Monday; 2: Tuesday; 3: Wednesday; 4: Thursday; 5: Friday; 6: Saturday; 7: Sunday

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Scheduling automatic reboots for your device.

Input Example

```
{
    "enable":true,
    "hour":23,
    "min":59,
    "week":[1,2]
  }
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	The parameters are missing or incorrect

Get device info

1. API Description

This API is used to get the device information. Please check whether each sub-item of capability is true, and only when it is true, the corresponding API can be accessed.

Request mode: POST [ip]/api/system/device-info

Administrator Rights	Logged-in
No	Yes

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
device-name	String	The device's name
product-id	String	The device's ID
product-name	String	The product family name
hardware-rev	String	The hardware version
serial-number	String	The device's serial number
firmware-ver	String	The device's firmware version
firmware-name	String	The device's firmware name
build-time	String	The device's firmware build time
capability	CapabilityInfo	The device's capability list.

CapabilityInfo

Name	Type	Description
support-timezone	Boolean	Whether the device supports time zone true: Yes; false: No
support-ntp	Boolean	Whether the device supports NTP true: Yes; false: No
support-station	Boolean	Whether the device supports Wi-Fi mode true: Yes; false: No
support-ap	Boolean	Whether the device supports AP mode true: Yes; false: No
support-online-upgrade	Boolean	Whether the device supports online upgrade. true: Yes; false: No
support-sc-control	Boolean	Whether the device supports Control Hub true: Yes; false: No
support-if-prio	Boolean	Whether the device supports network card priority setting true: Yes; false: No
support-usb-ncm	Boolean	Whether the device supports enabling or disabling USB NET, which taking effect after rebooting device. true: Enabled; false: Disabled
support-wifi-mutex	Boolean	The WiFi supports working in AP and STA modes simultaneously (only applicable to USB Fusion) true: Yes; false: No
support-ipv6	Boolean	Whether the device supports IPv6. true: Yes; false: No

4. Example

Getting the device information.

Input Example

```
None
```

Output Example


```
{
    "device-name": "Pro Router ONE",
    "product-id": "0x601",
    "product-name": "Pro Router ONE",
    "hardware-rev": "B",
    "serial-number": "0123456789",
    "firmware-ver": "0.9.210",
    "firmware-name": "Development",
    "build-time": "2023-04-14 06:48:13",
    "capability": {
        "support-usbc-name": false,
        "support-timezone": true,
        "support-ntp": true,
        "support-station": true,
        "support-ap": true,
        "support-online-upgrade": false,
        "support-sc-control": false,
        "support-if-prio": false,
        "support-usb-ncm": false,
        "support-wifi-mutex": false,
        "support-ipv6": true
    },
    "status": 0,
    "code": "Success"
}
```


5. Error Code

No error code related to the API business logic. For other error codes, see Common Error Codes.

Get basic info

1. API Description

This API is used to get the device's basic information, including CPU, memory, up time, etc.

Request mode: POST [ip]/api/system/info

Administrator Rights	Logged-in
No	Yes

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
device-name	String	The device's name
mdns-url	String	mDNS URL
uptime	Int	The uptime, in seconds
cpu	CPUInfo	CPU information
mem	MemInfo	Memory information
datetime	DateTimeInfo	The system time
auto-reboot	AutoRebootInfo	The auto-reboot information

CPUInfo

Name	Type	Description
total	Int	The total time of CPU
idle	Int	The idle time of CPU
usage	Int	The CPU usage x 100

MemInfo

Name	Type	Description
total	Int	The system's total memory, in KB
avail	Int	The system's available memory, in KB

DateTimeInfo

Name	Type	Description
cur-time	String	The system time Time format: yyyy-MM-dd HH:mm:ss
zonename	String	The time zone name
ntp-enable	Boolean	Whether NTP is enabled true: Yes; false: No
ntp-server1	String	The NTP server 1
ntp-server2	String	The NTP server 2

AutoRebootInfo

Name	Type	Description
enable	Boolean	Enable auto-reboot true: Enabled; false: Disabled
hour	Int	Time to auto-reboot, hour
min	Int	Time to auto-reboot, minute
week	[Array]	Weekly repeat 1: Monday; 2: Tuesday; 3: Wednesday; 4: Thursday; 5: Friday; 6: Saturday; 7: Sunday

4. Example

Getting the device's basic information.

Input Example

```
None
```

Output Example


```
{
    "device-name": "USB Fusion",
    "mdns-url":"xxxxx.local",
    "uptime": 8410,
    "cpu": {
        "total": 1624896,
        "idle": 1281701,
        "usage": 2110
    },
    "mem": {
        "total": 8069612,
        "avail": 7171768
    },
    "datetime": {
        "cur-time": "2021-12-20 13:25:57",
        "zonename": "Asia/Shanghai",
        "ntp-enable": true,
        "ntp-server1": "0.pool.ntp.org",
        "ntp-server2": "1.pool.ntp.org"
    },
  "auto-reboot":{
    "enable":true,
    "hour":23,
    "min":59,
    "week":[
      1,
      2
    ]
  },
    "status": 0,
    "code": "Success"
}
```


5. Error Code

No error code related to the API business logic. For other error codes, see Common Error Codes.

Set date and time

1. API Description

This API is used to set the date and time.

Request mode: POST [ip]/api/system/set-date-time

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Name	Required	Type	Description
ntp-enable	Yes	Boolean	Whether to enable NTP true: Yes; false: No
ntp-server1	Yes	String	The NTP server 1
ntp-server2	Yes	String	The NTP server 2
time	No	String	The system time, required when ntp-enable = false. Time format: yyyy-MM-dd HH:mm:ss,

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Setting the date and time manually.

Input Example

```
{
  "ntp-enable": false,
  "ntp-server1": "0.pool.ntp.org",
  "ntp-server2": "1.pool.ntp.org",
  "time": "2024-04-11 09:50:18"
}
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	The parameters are missing or incorrect

Set device name

1. API Description

This API is used to set the device name.

Request mode: POST [ip]/api/system/set-device-name

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Name	Required	Type	Description
name	Yes	String	The device name

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
ext-need-reboot	Boolean	Whether a reboot is required. true: Yes; false: No

4. Example

Setting the device name to Magewell-1.

Input Example

```
{
  "name": "Magewell-1"
}
```

Output Example

```
{
    "status": 0,
    "code": "SUCCESS",
    "ext-need-reboot": true
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	The parameters are missing or incorrect

Set time zone

1. API Description

This API is used to set time zone.

Request mode: POST [ip]/api/system/timezone-set

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Name	Required	Type	Description
zonename	Yes	String	The time zone name

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Setting time zone.

Input Example

```
{
  "zonename": "Asia/Shanghai"
}
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	Missing required parameters
16	MW_STATUS_NOT_EXIST	Time zone does not exist

Get DNS

> Deprecated since September 2024.

1. API Description

This API is used to get DNS.

Request mode: POST [ip]/api/network/get-dns

Administrator Rights	Logged-in
No	Yes

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
is-manual	Boolean	Whether the DNS is set manually true: Yes; false: No
dns1	String	Primary DNS. Empty characters mean not set.
dns2	String	Secondary DNS. Empty characters mean not set.

4. Example

Getting DNS.

Input Example

```
None
```

Output Example

```
{
  "is-manual": false,
  "dns1": "10.0.1.3",
  "dns2": "",
  "status": 0,
  "code": "Success"
}
```

5. Error Code

No error code related to the API business logic. For other error codes, see Common Error Codes.

Get network card info

1. API Description

This API is used to get network card information.

Request mode: POST [ip]/api/network/if-info

Administrator Rights	Logged-in
No	Yes

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
device-name	String	The device name
net	Array of NetData	The network card list
ext-mobile-first	Booean	Reserved

NetData

Name	Type	Description
support-enable	Boolean	Whether to support Enable/Disable Wi-Fi/AP true: Yes; false: No
enable	Boolean	Whether the network card service is enabled true: Yes; false: No
prio	Int	Network card priority 0: No priority, generally refering to USB, ETH CONSOLE; 1 ~ 99 is normal
iface	String	The network card name
type	Int	The network card type 0: Ethernet 1: Wi-Fi/AP 2: USB Sharing 3: USB NET 4: Built-in 4G/5G 5: Bridge
use-dhcp	Boolean	Whether to enable DHCP to get IP address true: Yes; false: No
ipaddr	String	IP address
ipv6addr	String	IPv6 address
netmask	String	The subnet mask
gateway	String	The gateway address
dns1	String	Primary DNS. Empty characters mean not set.
dns2	String	Secondary DNS. Empty characters mean not set.
mac	String	The MAC address
link-speed	Int	The link speed 10: 10Mbps 100: 100Mbps 1000: 1Gbps 2500: 2.5Gbps 10000: 10Gbps 12: full-speed 480: high-speed 5000: super-speed-5g 10000: super-speed-10g
link-state	Int	The link state 0: Network port exception 1: Not connected 2: Connected
tx-speed-kbps	Int	The sending speed (Kbps)
rx-speed-kbps	Int	The receiving speed (Kbps)
mode	Int	The working mode of wireless network card 0: STA mode; 1: AP mode
mode-lock	Boolean	Whether the wireless card working mode is locked or not true: Yes; false: No
ssid	String	The AP name
reboot-require	Boolean	Whether it requires reboot when the wireless network card switches working mode true: Yes; false: No

Name	Type	Description
enable-ncm	Boolean	Whether USB NET is enabled. true: Enabled; false: Disabled
sim-state	Int	The SIM card state 0: not ready 1: ready 2: PIN 3: PUK
pin-remaind	Int	The remaining PIN times
standard	Int	Cellular network signal types 0: Unknown; 2: 2G; 3: 3G; 4: 4G; 5: 5G Signal quality for 3G and below refers to RSSI Signal reference for 4G/5G is RSRP
rssi	Int	Cellular Network Signal Strength Very Good: RSSI > -65dBm Good: RSSI = -65 ~ -75dBm Average: RSSI = -75 ~ -85dBm Poor: RSSI = -85 ~ -95dBm Very Poor: RSSI < -95dBm
rsrp	Int	4G/5G RSRP Very Good: RSRP>-115dBm Good: RSRP=-120 to -115dBm Average: RSRP=-125 to -120dBm Poor: RSRP=-130 to -125dBm Very Poor: RSRP<-130dBm
rsrq	Int	4G/5G RSRQ value, in dB
sinr	Int	4G/5G SINR value, in dB
imei-no	String	IMEI information
operator	String	The operator, including CHN-MOBLIE, CHN-UNICOM, CHN-CT, Orange, O2, Vodafone, AT&T, T-Mobile, Verizon, Google Fi
phone-number	String	SIM number
vendor	String	The vendor of the 4G/5G module
product	String	The product information of the 4G module

4. Example

Obtaining network card information.

Input Example

```
None
```

Output Example


```
{
    "device-name": "00A601230913015",
    "ext-mobile-first": false,
    "net": [
        {
            "support-enable": false,
            "enable": true,
            "prio": 34,
            "iface": "eth0",
            "type": 0,
            "use-dhcp": true,
            "ipaddr": "10.10.6.222",
            "netmask": "255.255.240.0",
            "gateway": "10.10.0.1",
            "dns1": "10.10.1.3",
            "dns2": "",
            "ipv6addr": [
                "fe80::d2c8:57ff:fe81:c75e"
            ],
            "mac": "d0:c8:57:81:c7:5e",
            "link-speed": 1000,
            "link-state": 2,
            "tx-speed-kbps": 0,
            "rx-speed-kbps": 35
        }
        {
            "support-enable": true,
            "enable": true,
            "prio": 41,
            "iface": "wlan0",
            "type": 1,
            "mode": 1,
            "mode-lock": false,
            "ssid": "Magewell_ASR_3015_5G",
            "reboot-require": false,
            "use-dhcp": true,
            "ipaddr": "",
            "netmask": "",
            "gateway": "",
            "dns1": "",
            "dns2": "",
            "ipv6addr": [
                "fe80::d2c8:57ff:fe81:b7f1"
            ],
            "mac": "d0:c8:57:81:b7:f1",
            "link-speed": -1,
            "link-state": 2,
            "tx-speed-kbps": 0,
            "rx-speed-kbps": 0
        }
        {
            "support-enable": false,
            "enable": true,
            "prio": 36,
            "iface": "wwan0",
            "type": 4,
            "use-dhcp": true,
            "ipaddr": "",
            "netmask": "",
            "gateway": "",
```

```
            "dns1": "",
            "dns2": "",
            "ipv6addr": [
            ],
            "mac": "1a:97:9f:84:3a:e2",
            "link-speed": -1,
            "link-state": 0,
            "tx-speed-kbps": 0,
            "rx-speed-kbps": 0,
            "enable-ipv6": false,
            "sim-state": 2,
            "pin-remaind": 3,
            "standard": 0,
            "rssi": 0,
            "rsrp": 0,
            "rsrq": 0,
            "sinr": 0,
            "band": "",
            "imei-no": "",
            "phone-number": "",
            "operator": "",
            "version": "",
            "vendor": "Quectel Wireless Solutions Co., Ltd.",
            "product": "EC20\/EC25\/EM05-CE LTE modem"
        }
    ],
    "status": 0,
    "code": "Success"
}
```


5. Error Code

No error code related to the API business logic. For other error codes, see Common Error Codes.

Configure Ethernet

1. API Description

This API is used to configure Ethernet.

Request mode: POST [ip]/api/network/if-set

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Name	Required	Type	Description
iface	Yes	String	The network card name, eth0
use-dhcp	Yes	Boolean	Whether to enable DHCP to get IP address true: Yes; false: No
ipaddr	No	String	The IP address, which must be filled in when use-dhcp is false
netmask	No	String	The subnet mask, which must be filled in when use-dhcp is false
gateway	No	String	The gateway address, which must be filled in when use-dhcp is false
dns1	No	String	Primary DNS
dns2	No	String	Secondary DNS

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Configuring Ethernet.

Input Example

```
{
  "iface": "eth0",
  "use-dhcp": false,
  "ipaddr": "10.10.10.88",
  "netmask": "255.255.240.0",
  "gateway": "10.10.0.1",
  "dns1": "10.10.0.3",
  "dns2": "",
}
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	Missing required parameters
31	MW_STATUS_NOT_PERMITTED	The Ethernet card is missing

Set DNS

> Deprecated since September 2024.

1. API Description

This API is used to set DNS.

Request mode: POST [ip]/api/network/set-dns

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Name	Required	Type	Description
is-manual	No	Boolean	Whether to configure DNS manually true: Yes; false: No
dns1	No	String	Primary DNS. Empty characters mean not set.
dns2	No	String	Secondary DNS. Empty characters mean not set.

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Set DNS.

Input Example

```
{
  "is-manual": true,
  "dns1": "10.0.1.3",
  "dns2": ""
}
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

No error code related to the API business logic. For other error codes, see Common Error Codes.

Configure USB NET

1. API Description

This API is used to configure USB NET.

Request mode: POST [ip]/api/network/usb-config

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Name	Required	Type	Description
iface	Yes	String	The network card name, usb0
ipaddr	Yes	String	IP address

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Configuring USB NET.

Input Example

```
{
  "iface": "usb0",
  "ipaddr": "192.168.66.1"
}
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	Missing required parameters
12	MW_STATUS_DEVICE_LOST	The USB network card is missing

Clear upgrade status

1. API Description

This API is used to clear the upgrade status.

Request mode: POST [ip]/api/upgrade/clear

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Clearing the upgrade status.

Input Example

```
None
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

No error code related to the API business logic. For other error codes, see Common Error Codes.

Get firmware version and upgrade status

1. API Description

This API is used to get the current firmware version and upgrade status.

Request mode: POST [ip]/api/upgrade/state

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
state	Int	The task execution status 0: Idle 1: Initialize and upgrade 2: Upgrading 3: Upgraded 4: Online firmware downloading
cur-ver	String	The current firmware version
update-version	String	The latest firmware version
num-steps	Int	The total number of steps for upgrade, only available when state is 2
step	Int	The current step number, only available when state is 2
step-name	String	The name of the current step, only available when state is 2
step-progress	Int	The progress of the current step, only available when state is 2 Value range: 0 - 100 Unit: %
download-percent	Float	The percentage of online download

4. Example

Getting the current firmware version and upgrade status.

Input Example

```
None
```

Output Example

```
{
  "status": 0,
  "code": "Success",
  "state": "updating",
  "cur-ver": "1.1.72",
  "update-version": "1.1.72",
  "num-steps": 4,
  "step": 2,
  "step-name": "Erasing image",
  "step-progress": 28
}
```

5. Error Code

No error code related to the API business logic. For other error codes, see Common Error Codes.

Upgrade firmware

1. API Description

This API is used to upgrade firmware. During the upgrade process, you can get firmware version and upgrade status.

Request mode: POST [ip]/api/upgrade/update

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Name	Required	Type	Description
is-online	Yes	Boolean	Whether it is online upgrade. true: Yes; false: No, it is manual upgrade.
mode	Yes	Int	The upgrade mode 0: Auto
timeout	Yes	Int	Upgrade fails with timeout (upgrade progress keeps unchanged), in seconds

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Upgrading firmware.

Input Example

```
{
  "is-online": false,
  "mode": 0,
  "timeout": 120
}
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	Missing required parameters
11	MW_STATUS_DEVICE_BUSY	Upgrading
16	MW_STATUS_NOT_EXIST	The file does not exist

Upload firmware

1. API Description

This API is used to upload firmware. The upload file format should be .mwf.

Request mode: POST [ip]/api/system/upload-fw

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Binary encapsulated attachment data.

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
need-clean-data	Boolean	Whether user data is cleared when upgrading true: Yes; false: No
up-to-date	Boolean	Whether the firmware is the latest version true: Yes; false: No
version	String	The firmware version to upload
size	Int	The firmware size to upload

4. Example

Uploading firmware.

Input Example

```
------WebKitFormBoundaryIQYfOLWb1KdjY6f3
Content-Disposition: form-data; name="file"; filename="usb_fusion_rev_a_2_5_0.mwf"
Content-Type: application/octet-stream


------WebKitFormBoundaryIQYfOLWb1KdjY6f3--
```

Output Example

```
{
    "status": 0,
    "code": "Success",
    "need-clean-data": false,
    "up-to-date": false,
    "version": "2.5.0",
    "size": 258818308
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
23	MW_STATUS_FILE_TYPE_ERROR	The file is error

Add a user

1. API Description

This API is used to add a user.

Request mode: POST [ip]/api/user/add

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Name	Required	Type	Description
username	Yes	String	The user name
password	Yes	String	The password which is encrypted with sha256

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Adding a user.

Input Example

```
{
    "username":"test",
    "password":"c4ca4238a0b923820dcc509a6f75849b"
}
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	The parameters are missing
31	NOT_PERMITTED	Only administrators have permission
41	CONSTRAINT_VIOLATION	The existed user

Change login password

1. API Description

This API is used to change the user's login password. The current password must be input when changing the password.

Request mode: POST [ip]/api/user/ch-password

Administrator Rights	Logged-in
No	Yes

2. Input Parameters

Name	Required	Type	Description
password	Yes	String	The current password which is encrypted with sha256
new-password	Yes	String	The new password which is encrypted with sha256

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Changing the current password.

Input Example

```
{
    "password":"c4ca4238a0b923820dcc509a6f75849b",
    "new-password":"c1c224b03cd9bc7b6a86d77f5dace40191766c485cd55dc48caf9ac873335d6f"
}
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	Missing required parameters
39	MISMATCH	Wrong current password

Delete a user

1. API Description

This API is used to delete a user.

Request mode: POST [ip]/api/user/del

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Name	Required	Type	Description
username	Yes	String	The user name, obtained using Get system user list

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Deleting a user.

Input Example

```
{
    "username":"test"
}
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	The parameters are missing
31	NOT_PERMITTED	Only administrators have permission

Get system user list

1. API Description

This API is used to obtain the user list of the system.

Request mode: POST [ip]/api/user/get-all

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
users	Array of user	User list

USER

Name	Type	Description
username	String	User name
group	String	User group, Admin or User

4. Example

Obtaining the user list.

Input Example

```
None
```

Output Example

```
{
    "users": [
        {
            "username": "Admin",
            "group": "Admin"
        },
        {
            "username": "test",
            "group": "User"
        }
    ],
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
31	NOT_PERMITTED	Only administrators have permission

Log in

1. API Description

This API is used to log in. After the user logs in successfully, the Session ID is stored in the Cookie (Cookie: sid-[serial number]=t2i704wbvoy51y408p588bpji010ibp0).

Request mode: POST [ip]/api/user/login

Administrator Rights	Logged-in
No	No

2. Input Parameters

Name	Required	Type	Description
username	Yes	String	The user name, obtained using Get system user list
password	Yes	String	The password which is encrypted with sha256

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
sid	String	User ID

4. Example

Logging in.

Input Example

```
{
    "username":"test",
    "password":"c4ca4238a0b923820dcc509a6f75849b"
}
```

Output Example

Response Headers

```
    Content-Type: application/json; charset=utf-8
    Expires: 0
    Set-Cookie: sid-A506220808450=6440wa6u5wfw8wv43f91v55cqkctnpv6
```

Response Body

```
{
    "status": 0,
    "code": "Success",
    "sid": "6440wa6u5wfw8wv43f91v55cqkctnpv6"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	The parameters are missing
16	MW_STATUS_NOT_EXIST	The user does not exist
36	MW_STATUS_AUTH_FAILED	Wrong password

Log out

1. API Description

This API is used to log out and return to the login screen.

Request mode: POST [ip]/api/user/logout

Administrator Rights	Logged-in
No	Yes

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Logging out.

Input Example

```
None
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
37	MW_STATUS_NOT_LOGGED_IN	Not logged in, Invalid Log-in

Reset password

1. API Description

This API is used to reset the password, and it does not need to input the current password.

Request mode: POST [ip]/api/user/set-password

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Name	Required	Type	Description
username	Yes	String	The user name, obtained using Get system user list
password	Yes	String	The password which is encrypted with sha256

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Resetting the "test" user's password.

Input Example

```
{
    "username":"test",
    "password":"c4ca4238a0b923820dcc509a6f75849b"

}
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	Missing required parameters
16	NOT_EXIST	The user does not exist
31	NOT_PERMITTED	Only administrators have permission

Get the Token

1. API Description

This API is used to get the user's Token.

Request mode: POST [ip]/api/user/token-all

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Null

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
tokens	Array of TokenInfo	Token list

TokenInfo

Name	Type	Description
username	string	The user name
token	string	Token
expire	int	Token expiration time, starting from 1970-01-01 00:00:00 +0000 (UTC), in seconds. -1 means unlimited.

4. Example

Getting the user's Token.

Input Example

```
None
```

Output Example

```
{

    "tokens": [
        {
            "username": "Admin",
            "token": "HTrkbOUWh3KCl57aYmSLHT3qDyswncF4ynAGJatr8i5laiZZbUT83MnhRJOlghFB",
            "expire": -1
        }
    ],
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
31	NOT_PERMITTED	Only administrators have permission

Add a Token

1. API Description

This API is used to add a token for a user.

Request mode: POST [ip]/api/user/token-add

Administrator Rights	Logged-in
No	Yes

2. Input Parameters

Name	Required	Type	Description
username	Yes	String	The user name
token	Yes	String	Token
expire	Yes	Int	Token expiration time, starting from 1970-01-01 00:00:00 +0000 (UTC), in seconds. -1 means unlimited.

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Adding a Token.

Input Example

```
{
    "username":"Admin",
    "token":"HTrkbOUWh3KCl57aYmSLHT3qDyswncF4ynAGJatr8i5laiZZbUT83MnhRJOlghFB",
    "expire": -1
}
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	Missing required parameters
31	NOT_PERMITTED	Only administrators have permission
41	CONSTRAINT_VIOLATION	The existed user

Delete a Token

1. API Description

This API is used to delete the user's Token.

Request mode: POST [ip]/api/user/token-del

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Name	Required	Type	Description
token	Yes	string	Token

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Deleting the user's Token.

Input Example

```
{
    "token":"HTrkbOUWh3KCl57aYmSLHT3qDyswncF4ynAGJatr8i5laiZZbUT83MnhRJOlghFB"
}
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	Missing required parameters

Clear logs

1. API Description

This API is used to clear all the system logs.

Request mode: POST [ip]/api/log/clear

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Clearing all the system logs.

Input Example

```
None
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

No error code related to the API business logic. For other error codes, see Common Error Codes.

Export logs

1. API Description

This API is used to export the current system log of the device as a .html file.

Request mode: POST [ip]/api/log/export

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Name	Required	Type	Description
filename	Yes	String	The exported filename

3. Output Parameters

The log is downloaded as a .html file and saved to a local folder.

4. Example

Exporting the current system log of the device as Log_2024_04_11_14_01_08.html.

Input Example

```
{
  "filename": "Log_2024_04_11_14_01_08.html"
}
```

Output Example

Download the html file and save to a local folder.

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	Missing required parameters

Filter logs

1. API Description

This API is used to filter logs.

Request mode: POST [ip]/api/log/filter

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Name	Required	Type	Description
types	Yes	Int	Log types, including all, info, warn and error, which can be separated by commas if multiple types are requested.
key	Yes	String	The key word for filtering, which can be an empty string.

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
logs	Array of LogList	Log list

LogList

Name	Type	Description
no	Int	The log number
time	String	The log time
type	String	The log type, including info, warn, and error
message	String	The log content

4. Example

Filtering all logs.

Input Example

```
{
  "types": "all",
  "key": ""
}
```

Output Example


```
{
    "status": 0,
    "code": "Success",
    "logs": [
        {
            "no": 0,
            "time": "2022/09/09 16:11:07.920",
            "type": "info",
            "message": "xxxxxx"
        },
        {
            "no": 1,
            "time": "2022/09/09 16:11:04.721",
            "type": "info",
            "message": "xxxxxx"
        }
    ]
}
```


5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	Missing required parameters

Check factory reset permission

1. API Description

This API is used to check whether the device is allowed to be reset.

Request mode: GET/POST [ip]/api/factory-reset-permission

Administrator Rights	Logged-in
No	No

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
reset-enable	Boolean	Whether it is allowed to reset your device. true: Yes; false: No

4. Example

Checking whether the device is allowed to be reset.

Input Example

```
None
```

Output Example

```
{
  "status": 0,
  "code": "Success",
  "reset-enable": true
}
```

5. Error Code

No error code related to the API business logic. For other error codes, see Common Error Codes.

Ping test

1. API Description

This API is used to detect whether the device is accessible.

This function is used to ensure that the device has restarted completely after `firmware update`, `reset all settings` or `change IP address`.

Request mode: GET/POST [ip]/api/ping

Administrator Rights	Logged-in
No	No

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Detecting your device.

Input Example

```
None
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

No error code related to the API business logic. For other error codes, see Common Error Codes.

Reboot device

1. API Description

This API is used to reboot the device. After rebooting, you need to log in again. You can use Ping Test to check whether the device is rebooted.

Request mode: GET/POST [ip]/api/reboot

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
delay	Int	Delay time before rebooting, in seconds.
estimate-sec	Int	Estimated time for a restart, in seconds.

4. Example

Rebooting your device.

Input Example

```
None
```

Output Example

```
{
  "status": 0,
  "code": "Success",
  "delay": 5,
  "estimate-sec": 15
}
```

5. Error Code

No error code related to the API business logic. For other error codes, see Common Error Codes.

Reset device

1. API Description

This API is used to reset your device to default settings.

API request: GET/POST [ip]/api/factory-reset

Administrator Rights	Logged-in
No	No

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Resetting device.

Input Example

```
None
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
31	MW_STATUS_NOT_PERMITTED	No permission

Get certificate info

1. API Description

This interface is used to get the information of your certificate.

Request mode: POST [ip]/api/certificate/info

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description
enable	Boolean	Enable HTTPS access true: Enabled; false: Disabled
certificate	CertificateInfo	The information of the certificate

CertificateInfo

Name	Type	Description
subject-name	String	Subject name, containing the identity information of the certificate holder
issuer-name	String	Issuer name, containing the identity information of the institution issuing the certificate
version	Int	Version
serial-num	String	Serial number
expires	String	Expiration date
valid-before	String	Valid before
public-key	String	Public key
algorithm	String	Signing algorithm
type	String	Algorithm type

4. Example

Getting certificate info.

Input Example

```
None
```

Output Example


```
{
    "status": 0,
    "code": "Success",
    "enable": true,
    "certificate": {
        "subject-name": "/C=CN/ST=GuangDong/L=ShenZhen/O=bolin-av.com/CN=bolin-av.com/emailAddress=av@bolin-av.com",
        "issuer-name": "/C=CN/ST=GuangDong/L=ShenZhen/O=Bolin Technology Root CA",
        "version": 1,
        "serial-num": "C11B50824C53B27C",
        "expires": "Mar 25 02:06:25 3022 GMT",
        "valid-before": "Nov 22 02:06:25 2022 GMT",
        "public-key": "30818902818100ba1e3aff73f880f3bc219f3d6714edb2cf4a4b8fb072cdf55c5058903af7691eeeb4cae6aacb7148
6b6cb2001e14cb5b9113d52f8db666c87b65465a4a1204976390d33ad42de91597bcab511d6ca9c0b7e3dad4f75844206721024066 05eb4a1dcbf
9871f85ec412947b27648ee48b03d2af9e8b9f915f534bec99d4d6ed3d70203010001",
        "algorithm": "1.2.840.113549.1.1.11",
        "type": "EVP_PKEY_RSA"
    }
}
```


5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
6	MW_STATUS_UNKNOWN_ERROR	An unknown error occurred

Enable HTTPS access

1. API Description

This API is used to enable HTTPS access, which takes effect after rebooting.

Request mode: POST [ip]/api/certificate/enable

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Name	Required	Type	Description
enable	Yes	Boolean	Enable HTTPS access true: Enabled; false: Disabled

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Enabling HTTPS access.

Input Example

```
{
    "enable": false
}
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
6	MW_STATUS_UNKNOWN_ERROR	An unknown error occurred
7	MW_STATUS_INVALID_ARG	Missing required parameters

Import SSL certificate

1. API Description

This API is used to import the SSL certificate.

Request mode: POST [ip]/api/certificate/import

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

Name	Required	Type	Description
certificate	Yes	String	Root certificate
private-key	Yes	String	Root certificate private key

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Importing the SSL certificate.

Input Example


```
{
    "certificate": "-----BEGIN CERTIFICATE-----\nMIIC1DCCAbwCCQDBG1CCTFOyfDANBgkqhkiG9w0BAQsFADBXMQswCQYDVQQGEwJD\nTj
ESMBAGA1UECAwJR3VhbmdEb25nMREwDwYDVQQHDAhTaGVuWmhlbjEhMB8GA1UE\nCgwYQm9saW4gVGVjaG5vbG9neSBSb290IENBMCAXDTIyMTEyMjAyM
DYyNVoYDzMw\nMjIwMzI1MDIwNjI1WjCBgjELMAkGA1UEBhMCQ04xEjAQBgNVBAgMCUd1YW5nRG9u\nZzERMA8GA1UEBwwIU2hlblpoZW4xFTATBgNVBA
oMDGJvbGluLWF2LmNvbTEVMBMG\nA1UEAwwMYm9saW4tYXYuY29tMR4wHAYJKoZIhvcNAQkBFg9hdkBib2xpbi1hdi5j\nb20wgZ8wDQYJKoZIhvcNAQE
BBQADgY0AMIGJAoGBALoeOv9z+IDzvCGfPWcU7bLP\nSkuPsHLN9VxQWJA692ke7rTK5qrLcUhrbLIAHhTLW5ET1S+NtmbIe2VGWkoSBJdj\nkNM61C3p
FZe8q1EdbKnAt+Pa1PdYRCBnIQJAZgXrSh3L+YcfhexBKUeydkjuSLA9\nKvnoufkV9TS+yZ1NbtPXAgMBAAEwDQYJKoZIhvcNAQELBQADggEBAEsWsAb
M8fFb\nvDtdkkE3pmjeZq1xUVBdNHgryVFzOl9UiC7eV48SGhLyLZ5AQF6htIJ0Z70pn+2G\nMoGZCu7CRmP/tvFHxUmFE4GRxHJYyxXlej3t2ju+5qiI
0o3iGAaeobk6+OR5l9D0\nL+UvLrhb0wbE53oZwMgGuH0cN08zAmh56ySGdJtmlrXVZugtEgOFvOORVFwUpERA\nXLG6HyyFYaZf4yBYFGk6N804QF8JB
sx68vavH41LODCY9kFUagVUSzTGvrLx2p0R\nroxsbYJfimRtRs3xLifZaC+pXke1jiKIfq8qyxzzybRFf4KsdmhaWebgeXmb2H9M\n7VzaytYC1/k=\n
-----END CERTIFICATE-----\n",
    "private-key": "-----BEGIN RSA PRIVATE KEY-----\nMIICXQIBAAKBgQC6Hjr/c/iA87whnz1nFO2yz0pLj7ByzfVcUFiQOvdpHu60yuaq
\ny3FIa2yyAB4Uy1uRE9UvjbZmyHtlRlpKEgSXY5DTOtQt6RWXvKtRHWypwLfj2tT3\nWEQgZyECQGYF60ody/mHH4XsQSlHsnZI7kiwPSr56Ln5FfU0v
smdTW7T1wIDAQAB\nAoGBAKzN/rdKRPPt2cJZ6Jf6DYzU/EVV/11Ir0Rr8R7RMQAwUTAZp7vyspar7zsB\n8uwZuEXmZS0G5dFe2Xs5rAFT41/aVFEb+T
YxpdDUx+k3GpveXUhBx04qMYtWw4za\njzuKZSS2C6MdtOgayxSis+inrfRwPhV8mceia8yT6XiAD7OJAkEA6Hpz1MFs2Ujx\nYj8X05eHT+jmxARLCXK
y/h6YmE5zPavPU5nCsvpglEj3N7oVy9mMxHyhR6iF+Cpb\nKCKAjCGeewJBAMzy+eZ9kpcdDOU62DrPQvDecWqJ+gL28taUndMbAFjgGOZjpDES\nq9LC
vZi75CBQvL/InMx42PMCGRrWKcc0D1UCQQDkLdRk/ZjdyDeMK8syFmdsaned\n62zPEDJZNCSoUU/RN0JeluvACfojcV6c5tuf4p7E+GLAV8tuofwo73x
mjlTjAkAZ\nA8MKpeMBfLM5O1BvBSMXgR74sPSJ7hg/po4I2S/WqSuA20KMFspQBCLeBHjYlOIk\nyliRAGMCbgwgYWlh/fGtAkAcJf8uHBMHsb+GzOSr
QHs0chOt/cu7kRUVno5bvsVC\nX52nDmaC/QTfb9mFodkv3VvBHNOvG3vS/15YKWD8b5Pw\n-----END RSA PRIVATE KEY-----\n"
}
```


Output Example


```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
7	MW_STATUS_INVALID_ARG	Missing required parameters
15	MW_STATUS_WRITE_FAILED	IO write failed
40	MW_STATUS_VERIFY_FAILED	Verification failed

Delete SSL certificate

1. API Description

This API is used to delete the SSL certificated.

Request mode: POST [ip]/api/certificate/delete

Administrator Rights	Logged-in
Yes	Yes

2. Input Parameters

None

3. Output Parameters

Name	Type	Description
status	Int	Status code
code	String	Status description

4. Example

Deleting the SSL certificate.

Input Example

```
None
```

Output Example

```
{
    "status": 0,
    "code": "Success"
}
```

5. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

Status	Definition	Description
6	MW_STATUS_UNKNOWN_ERROR	An unknown error occurred