Leading Solution Provider for IP-based Video Transmission

API

N5/N6 Bi-Directional NDI Converter

NDI High Bandwidth+NDI|HX

(2023-Aug)

Through this manual, you will learn:

- Interface descriptions of the API
- Methods for using the API

As the product undergoes continuous upgrades and updates, the content of this manual may vary from the product you have purchased. Please refer to the actual product within the package for accurate information.

Legal Statement

By receiving this document from Changsha Kiloview Electronic Technology Co., Ltd. (referred to as "Kiloview" hereafter), you agree to the following terms. If you do not agree with these terms, please discontinue the use of this document.

This document is copyrighted by Changsha Kiloview Electronic Technology Co., Ltd. All rights not expressly granted in this document are reserved by Kiloview. This document contains proprietary information of Kiloview. Without prior written consent from Kiloview, no individual or entity shall copy, transmit, distribute, use, or disclose this document, including any images, tables, data, and other information contained herein.

is a registered trademark of Kiloview. Names and logos of Kiloview products are trademarks or registered trademarks of Kiloview. Other product or company names mentioned in this document may be trademarks or registered trademarks of their respective owners. Reading this document does not imply, by any means, an assignment of rights to the reader with respect to any mark appearing in this document, unless prior written consent is obtained from Kiloview or the respective third-party rights holder.

This product complies with design requirements related to environmental protection and personal safety. Storage, use, and disposal of the product should adhere to the requirements of the product manual, relevant contracts, and the laws and regulations of the respective country.

This document is provided on an "as is" and "as available" basis. Information in this document is subject to change as Kiloview products and technologies evolve, and Kiloview is not obligated to inform about such updates.

For matters not covered in this document, please visit Kiloview's website at www.kiloview.com for relevant information and technical support.

CONTENT

1. Security Rules 3
1.1 Authorization 3
1.2 Passing the Authorization Token in HTTP API Requests 5
1.3 Cancel User Authentication (Not Recommended) 6
2. Status and Switch between Encoding and Decoding 7
2.1 Retrieve the Current Working Mode 7
2.2 Switch between Encoder/Decoder 8
2.3 Check if Encoder/Decoder Mode Ready 9
3. NDI Encoder Status and Settings 11
3.1 Obtain NDI Encoding Settings 11
3.2 Configure NDI Encoding Settings 14
3.3 Retrieve NDI Encoding Status and Video/ Audio Format Information ... 16
3.4 Get Input Video Source List in Encoding 19
3.5 Set the Encoding Video Source 20
3.6 Retrieve Audio Signal and Volume 21
3.7 Configure the Input Signal and Volume 22
3.8 Retrieve NDI Status: Enable/Disable (Only Valid for FULL NDI) 23
3.9 Set Enable/Disable of NDI (Only Valid for FULL NDI) 24
4. NDI Sources Discovery 25
4.1 Retrieve NDI Sources Discovered in the Network 25
4.2 Manual Discovery: Specified IP or NDI Groups 28
4.3 Retrieve Specified Manual Discovery IP Addresses and NDI Groups 31
5. NDI Decoding: Preset 32
5.1 Retrieve the Current List and Status of Presets 33
5.2 Add an NDI Source to a Specified Preset 36
5.3 Remove NDI Source Definition from Preset 37
5.4 Define the Color of Preset 0 (Blank) 38
6. Decoding Output 39
6.1 Retrieve the Current Decoding Status. 39
6.2 Switch the Preset or Specified NDI Source to Current Decoding Output 42
7. Advanced Decoding Output Options 43
7.1 Retrieve the Configurable Resolution List 43
7.2 Advanced Settings of Video/Audio Output 44
7.3 Retrieve Advanced Video/Audio Output Setting Options 46
8. Tally Status and Control 47
8.1 Retrieve Current Tally Status 47
8.2 Configure the Current Tally Status 48
9. System Time 50
9.1 Obtain the current system time and configuration 50
9.2 Set the Current System Time/Configuration 51

10. System Management and Control 52
10.1 Retrieve the Operational Status of the Device 52
10.2 Restart the Device 54
10.3 Restore to Factory Settings 55
10.4 Retrieve the Host Name 57
10.5 Set the Host Name 58
10.6 Configure "Discovery Server" 58
10.7 Obtain the Status of the Discovery Server 59
10.8 Get the Current System Status 60
10.9 Obtain the System Version 62
11. User Management 63
11.1 Retrieve the Current User List 63
11.2 Adding a User 64
11.3 Modify User Information 65
11.4 Delete a User 66
12. Network Configuration 67
12.1 Get Current Network Configuration 67
12.2 Modify Network Configuration 69

1. Security Rules

In order to ensure network security, proper authorization is required for HTTP API access. Failure to adhere to the security rules outlined in this section will result in your HTTP requests being denied by the device.

> Unless under specific circumstances, we strongly recommend utilizing HTTPS (rather than HTTP) for executing HTTP API operations. Using HTTP might expose sensitive information (such as usernames and passwords) to potential security risks.

1.1 Authorization

Before utilizing any HTTP API described in this document, you must obtain authorization for using the HTTP API.

The authorization mechanism for NDI Devices is outlined as follows:

1. First, you need to provide a valid username and password. The NDI Device will validate the legitimacy of your username and password.
2. Upon successful validation of your username and password, the NDI Device will generate a pair of random Authorization Tokens for you and return them in the response.
3. It is imperative that you record the Authorization Token. For every subsequent HTTP API request, you must pass the Authorization

Token in the HTTP request's Cookie.

This process ensures the security and authentication of your interactions with the NDI Device through the HTTP API.

Authorization itself is also an HTTP API, but it differs from other APIs in that it does not verify the Authorization Token. Instead, it generates an Authorization Token for you by examining the username and password you provide.

API URL

/api/user/authorize.json

Request

Method：**GET/POST**

Field	Value	Description
user	**[STRING],Required**	Valid username to get authorization
password	**[STRING],Required**	Password of the username

Response

Format（Example）：

```
{
   "result": "ok",
   "data": {
```

```
        "token": "2e33a9b0b06aa0a01ede70995674ee23",
         "alias": "Admin"
    }
}
```


Data field description：

Field	Value	Description
token	**[STRING]**	Random Authorization Token
alias	**[STRING]**	Username Alias

Suggestions：

1) Please be reminded! It is highly recommended not to use the built-in system administrator user "admin" for API authorization, as this poses a significant security risk! You can create other users in the NDI Devices' Web UI and use those users to perform HTTP API requests.
2) Use HTTPS instead of HTTP for requesting authorization to avoid the risk of exposing usernames and passwords.

1.2 Passing the Authorization Token in HTTP API Requests

If you successfully request authorization following the method described in section 1.1, you will receive an Authorization Token generated by the NDI Device. Subsequently, when you request any other HTTP API, the NDI

Device will require validation of your Authorization Token's legitimacy. If you fail to provide an Authorization Token or provide an invalid value, your HTTP API request will return an error message similar to the following (example):：

```
{
    "result": "auth-failed",
    "msg": "Invalid token"
}
```

Pass through Cookie

Pass the Authorization Token through HTTP Cookie field name “token”。

eg:

```
POST /api/your/api.json HTTP/1.1
...
Cookie: token=9b6bdc43a8884cbd9503a1cda00a2d3b
...

<BODY>
```

If test with curl, the command is as below:

```
$ curl -b 'token=9b6bdc43a8884cbd9503a1cda00a2d3b'
https://192.168.100.168/api/your/api.json
```

1.3 Cancel User Authentication (Not Recommended)

You can disable HTTP API Authorization through the user management interface on the web page. By doing so, API calls will no longer undergo user authentication.

2. Status and Switch between Encoding and Decoding

Module name: mode

Basic URL: /api/mode/

N6 can be used either as a encoder or decoder, you can switch between them.

2.1 Retrieve the Current Working Mode

API URL

/api/mode/get.json

Request

Method: **GET/POST**

Parameters: NONE

Response

Example:

```
{
    "result": "ok",
    "data": {
        "mode": "encoder" /*Or "decoder"*/
  }
}
```

Data Field Description：

Field	Value	Description<
mode	**[STRING]**	"encoder" or "decoder" returns "encoder" to indicate that the device is operating in Encoder mode; "decoder" indicates that the device is operating in Decoder mode.

2.2 Switch between Encoder/Decoder

API URL

/api/mode/switch.json

Request

Method: **GET/POST**

Field	Value	Description
mode	**[STRING]** , **Required**	"encoder" or "decoder", specifies encoder or decoder mode

Response

Example:

```
{
   "result": "ok"
}
```

Note：

Switching the Encoder/Decoder mode requires a relatively long waiting time. Therefore, if you need to handle HTTP requests/responses with specific timeouts, please be aware that the execution time for this request might take up to 5 seconds at most. You should configure the API request timeout accordingly to avoid unexpected API failures due to a timeout that is too short.

If you need to fully confirm whether the Encoder/Decoder mode is functioning correctly, even after a successful return from calling this API to switch modes, you should further attempt to call /api/mode/status.json to verify the operational status of the NDI Device. The reason for this seemingly unusual approach is that switching the Encoder/Decoder mode requires an internal quick reboot action within the device.

2.3 Check if Encoder/Decoder Mode Ready

After you perform the Encoder/Decoder mode switch operation and the device undergoes a quick reboot action internally, if you need to confirm

that the switched mode has properly entered the operational state, you might need to periodically and continuously call this API until the returned result indicates that it has entered the working mode. Typically, the transition from switching to entering the working state might take around 10 to 15 seconds.

Please be assured that your current Time stamp and Authorization Token remain valid even after switching the encoding/decoding mode and the quick internal reboot within the device.

API URL

/api/mode/status.json

Request

Method: **GET/POST**

Parameters: NONE

Response

If the encoder/decoder mode is ready, it will return as:

```
{
    "result": "ok",
    "data": {
        "mode": "encoder", /*Or "decoder"*/
       "status": "ready"
  }
}
```

If it's not ready, it will return with error as:

```
{
    "result": "error",
    "msg": "waitChangeModeError"
}
```

3. NDI Encoder Status and Settings

Notes: APIs of this part applies only when the device is in Encoder mode.

3.1 Obtain NDI Encoding Settings

API URL

/api/encoder/ndi/get_config.json

Request

Method: **GET/POST**

Field	Value	Description
types	**[STRING] , Required**	"ndihx" or "ndifull" specifies to get information for NDI or NDI|HX.

Response

Example:

```
{
    "result": "ok",
```

```
    "data": {
        "device_group": "",
        "netmask": "255.255.0.0",
        "resolution": "none",
        "resolution_list": [
            {
                "name": "1920x1080P 60Hz",
                "value": "1080P60"
            },
            {
                "name": "1920x1080P 50Hz",
                "value": "1080P50"
            }
        ],
        "netprefix": "239.255.0.0",
        "channel_name": "Channel-HX",
        "ttl": 127,
        "quality": 2000,
        "ndi_connection": "tcp",
        "video_source":"HDMI",
        "codec":"H265",
        "hx_version":3
    }
}
```


Data Field Description：

Field	Value	Description
device_group	**[STRING]**	NDI Group Name. If this value is an empty string "", it indicates the use of the default NDI group, which is "public".
channel_name	**[STRING]**	NDI Encoding channel name.
ndi_connection	**[STRING]**	"tcp" or "multicast" or "no_multi_tcp". These are the recommended connection methods for NDI Receivers as an NDI Sender, according to NewTek's advice. Currently, TCP (default) or UDP multicast methods are permitted.
netprefix	**[STRING]**	***(Valid only when ndi_connection is "multicast")*** A multicast IP address (224.0.0.0 ~ 239.255.255.255) netprefix and netmask have a paired relationship. Similar to the conventional relationship between IP and netmask, the netmask determines the subnet for NDI multicast addresses. The NDI SDK will randomly select multicast addresses within this subnet for actual NDI transmission.
netmask	**[STRING]**	***(Valid only when NDI_connection is "multicast")*** The netmask for multicast IP address
quality	**[INT]**	Valid value range 2000~40000.Only NDI HX has this setting. (Full NDI no)
ttl	**[INT]**	Life time
resolution	**[STRING]**	Current scaling option, no scaling when “none”
resolution_list	**[OBJECT]**	Supported resolution list. Request with the value listed.
video_source	**[STRING]**	Current video source "HDMI"/"SDI" or "USB"
codec	**[STRING]**	Video encoding format, "H265" or "H264", only applicable when the type is "ndihx".
hx_version	**[INT]**	NDIHX version, 2 or 3, valid only when types shown as "ndihx"

Please note: The current version of the HTTP API may include certain additional fields in the returned results that are not documented here. These fields exist for software compatibility purposes and should be

disregarded. In subsequent software updates, fields not listed in the documentation may undergo changes or be removed.

3.2 Configure NDI Encoding Settings

API URL

/api/encoder/ndi/set_config.json

Request

Method: **GET/POST**

Field	Value	Description
types	**[STRING]**	Valid value **"ndihx"**or**"ndifull"**， represents NDIHX or NDI.
device_group	**[STRING]**, **Optional**	When you specified any field of device group, device name, channel name, you are changing the NDI Group, Device name or Channel Name. If no specified value, it will remain unchanged.
channel_name	**[STRING]**, **Optional**	Chanel name. Notes: NDI and NDI|HX can't share the same channel name.
ndi_connection	**[STRING]**, **Optional**	Valid values are "tcp" or "multicast" or "no_multi_tcp," specifying the recommended connection method for NDI Receivers. When specified as "multicast," you should also provide both the netprefix and netmask parameters.
netprefix	**[STRING]**, **Optional**	netprefix and netmask are used in pairs. Similar to the conventional relationship between IP and netmask, the netmask determines the subnet for NDI multicast addresses. The NDI SDK will

Field	Value	Description
		randomly select multicast addresses within this subnet for actual NDI transmission.
netmask	[STRING], Optional	If you set net_connection="multicast" but do not specify netprefix/netmask, the previously set netprefix/netmask values will be retained, if they were previously set. Otherwise, the NDI Device will generate a random multicast address and select 255.255.0.0 as the netmask.
quality	[INT], Optional	Encoding quality, valid only when types=ndihx, valid range 2000~40000
ttl	**[INT]**, **Optional**	Life time
resolution	**[STRING]**, **Optional**	Scaling, no scaling if value is **"none"**. Please refer to the resolution list of 3.1 for the value index.
codec	**[STRING]**, **Optional**	Reference value as "H265" or "H264", valid only when types as**"ndihx"**.
hx_version	**[INT]**, **Optional**	NDIHX version, reference value 2 or 3, valid only when types as**"ndihx"**.

Response

Example:z

```
{
    "result": "ok"
}
```

Please Note:

1. If the values of NDI Encoding parameters are changed, the NDI Device will disconnect the current NDI connections, reconfigure the

NDI Sender, and restart. This process is typically swift, and most NDI Receivers can quickly reconnect. However, it's important to note that this action will reset all NDI connections currently established with the NDI Device.

2. All the parameters listed above are optional. If you do not specify any parameters, it means that no action will be taken.

3.3 Retrieve NDI Encoding Status and Video/ Audio Format Information

---

API URL

/api/encoder/ndi/status.json

Request

Method: **GET/POST**

Field	Value	Description
types	**[STRING]** , **Required**	"**ndihx**" or "ndifull" specified obtaining the information of NDI or NDI|HX.

Response

Example:

```
{
```

```
    "result": "ok",
    "data": {
        "audio_sampling": 48000,
        "audio_format": "AAC",
        "audio_channels": 2,
        "audio_signal": "embedded",
        "samp_chan": "48KHz/2ch",
        "video_signal": false,
        "yRes": 1080,
        "xRes": 1920,
        "frame_rate": 30,
        "bitrate": 383222,
        "video_source": "HDMI",
        "interlaced": false,
        "serial_number": "1233211234567",
        "resolution": "1920x1080@30",
        "mode_version": "H.264 NDI|HX2",
        "src_resolution": "1920x1080@60",
        "src_frame_rate": 60,
        "src_xRes": 1920,
        "src_yRes": 1080
    }
}
```


Data Field Description：

Field	Value	Description
video signal	**[BOOLEAN]**	The video signal is used to indicate the current signal status of the NDI encoding video input. When video signal is set to false, it indicates that there is currently no video input obtained from the input interface (or the video format is not recognized/supported). When set to true, it signifies that a signal is present.
resolution	[STRING]	Video resolution name. Please note: This is a user-friendly resolution name, and it might not be suitable for obtaining detailed video format information in your program. You can use the xRes / yRes / frame_rate / interlaced fields to obtain program-friendly resolution information.
xRes	[INT]	xRes/yRes represents the current width/height of the video in pixels. If xRes/yRes = 0, it indicates that the current resolution is invalid.
yRes	[INT]	
frame_rate	[NUMBER]	Frame rate, unit: fps. If frame rate = 0, it means the current video frame rate is invalid.
interlaced	[BOOLEAN]	If the current video is in interlaced format, the value is true; otherwise, it's false
bitrate	[INT]	Current real time bitrate of NDI encoding, Kbps
audio_format	[STRING]	Audio format. Please note: This is a user-friendly audio format name (string), and it might not be suitable for obtaining detailed audio format information in your program. You can use audio sampling / audio channels to obtain program-friendly audio information.
audio_signal	[STRING]	none: No audio input embedded: Embedded digital audio from HDMI/SDI analog: Analog audio input Please note: The audio sources represented by "embedded" may vary depending on the device. For HDMI Encoders, this indicates audio from embedded digital audio in HDMI; for SDI Encoders, this represents audio from embedded digital audio in SDI.

Field	Value	Description
		Similarly, the meaning of "analog" can vary across different devices. Some devices may have Line In as their analog audio input, while others might have Microphone In. Please distinguish between them accordingly.
audio_sampling	[INT]	The current audio input sampling rate.
audio_channels	[INT]	The current number of audio input channels.
serial_number	[STRING]	Serial Number, unique ID of each device.
video_source	[STRING]	Input video signal, currently as HDMI
mode_version	[STRING]	Video format
src_resolution	[STRING]	Input video source resolution
src_frame_rate	[INT]	Input video source frame rate
src_xRes	[INT]	The video resolution width of the input video Source.
src_yRes	[INT]	The video resolution height of the input video source

3.4 Get Input Video Source List in Encoding

API URL

/api/encoder/ndi/get_video_source.json

Request

Method: **GET/POST**

Response

Example:

```
{
    "result":"ok",
        "data":["HDMI","USB"]
}
```

Data Field Description：

Field	Value	Description
DATA	[ARRAY]	Optional video source list: "HDMI", "SDI", "USB". Note: "USB" will only appear in the list when a USB video source is connected.

3.5 Set the Encoding Video Source

API URL

/api/encoder/ndi/select_video_source.json

Request

Method: **GET/POST**

Field	Value	Description
video_source	[STRING]	Selected audio source, for parameter refers to the data returned by the interface in section 3.4.

Response

Example:

```
{
    "result": "ok",
}
```

3.6 Retrieve Audio Signal and Volume

API URL

/api/audio/get_audio.json

Request

Method: **GET/POST**

Parameters: NONE

Response

Example:

```
{
    "result": "ok",
    "data": {
        "signal": "HDMI",
        "volume": 100,
        "channels": 2
```

```
    }
}
```

Data Field Description：

Field	Value	Description
signal	**[STRING]**	HDMI/SDI: Embedded digital audio from HDMI/SDI Line In: Analog audio input USB: Audio from USB video input
volume	**[INT]**	Current audio volume, valid range 0 ~ 200. 100 represents the original audio gain; <100 indicates a volume decrease; >100 indicates a volume increase.
channels	**[INT]**	Audio channels (Currently, the product only offers a two-channel option).

3.7 Configure the Input Signal and Volume

API URL

/api/audio/set_audio.json

Request

Method: **GET/POST**

Field	Value	Description
signal	**[STRING]**, **Optional**	HDMI/SDI: Embedded digital audio from HDMI/SDI Line In: Analog audio input USB: Audio from USB video input

Field	Value	Description
		Select the current audio input. Please refer to section 3.4 for details about audio signal. If not specified, the current audio input will not be modified.
volume	**[INT], Optional**	Adjust the audio input volume. Valid range is 0 ~ 200. 100 represents the original audio gain; <100 indicates a volume decrease; >100 indicates a volume increase. If not specified, the current audio volume will not be modified.

Response

Example:

```
{
    "result": "ok",
}
```

3.8 Retrieve NDI Status: Enable/Disable (Only Valid for FULL NDI)

API URL

/api/encoder/ndi/get_NDI_enable.json

Request

Method: **GET/POST**

Parameters: NONE

Response

Example:

```
{
    "result": "ok",
    "data": {
        "enable": true
    }
}
```

Data Field Description：

Field	Value	Description
enable	**[BOOLEAN]**	When the value is true, it indicates that FULL NDI is enabled. When the value is false, it indicates that FULL NDI is not enabled.

3.9 Set Enable/Disable of NDI (Only Valid for FULL NDI)

API URL

/api/encoder/ndi/set_NDI_enable.json

Request

Method: **GET/POST**

Field	Value	Description
enable	**[BOOLEAN], Optional**	When the value is true, it indicates that FULL NDI is enabled. When the value is false, it indicates that FULL NDI is disabled.

Response

Example:

```
{
    "result": "ok",
}
```

4. NDI Sources Discovery

Module name: decoder/discovery

Basic URL: /api/decoder/discovery/

Note: The APIs in this module are only valid when the device is operating in Decoder mode.

4.1 Retrieve NDI Sources Discovered in the Network

API URL

/api/decoder/discovery/get.json

Request

Method: **GET/POST**

Field	Value	Description
force	**[ANY]**, **Optional**	If you specify the force parameter, regardless of its value, it signifies a request for the NDI Device to perform a forced network rescan.

Response

Example:


```
{
    "result": "ok",
    "data": [
        {
            "enable": 1,
            "id": "1",
            "device_name": "E1_NDI-1212-111",
            "url": "192.168.2.170:5963",
            "series": "NDI",
            "group_name": "public",
            "ip": "192.168.2.170",
            "group": "public",
            "channel_name": "Chan 1",
            "original_url": "192.168.2.170:5963",
            "name": "E1_NDI-1212-111 (Chan 1)",
            "port": 5963
```

```
            },
            /* ... Each discovered items in the array ... */
        ],
        "data_size": 10
}
```

Data Field Description：

"data" is a JSON array, and "data_size" is a field that assists in describing the size of the "data" array (which you might not need).

Each item in the array represents an NDI Source discovered in the network.

Field	Value	Description
group	**[STRING]**	The NDI Group in which the NDI Source is located. If the Group you receive is an empty string "", according to NDI rules, it signifies the default public group.
name	**[STRING]**	NDI Source Name. This is the original name of the NDI Source. If you wish to obtain the device name and channel name, please refer to device_name and channel_name.
device_name	**[STRING]**	NDI Device Name parsed from the name, which typically also represents the host name of the NDI Source.
channel_name	**[STRING]**	NDI Channel Name parsed from the name.
url	**[STRING]**	NDI Source URL. Please note: The current URL format is <IP>:<port>, but it's advised not to assume this as a fact. The NDI SDK does not guarantee this. When using it, please retain the original value of the URL and refrain from attempting to alter it.
enable	**[INT]**	1: This Preset is defined. 0: This Preset has no defined NDI Source; in this case, all subsequent fields are irrelevant.

Field	Value	Description
id	**[INT]**	Serial Number
series	**[STRING]**	NDI 源
group_name	**[STRING]**	The NDI Group in which the NDI Source is located. If the Group you receive is an empty string "", according to NDI rules, it signifies the default "public" group.
ip	**[STRING]**	Ip address
original_url	**[STRING]**	NDI Source URL. Please note: The current format of the URL is <IP>:<port>, but it is advised not to assume this as a definitive format. The NDI SDK does not guarantee this. When using the URL, please retain its original value and refrain from attempting to modify it.
port	**[STRING]**	Port Number

Tips:

A. NDI Discovery is based on the mDNS mechanism and, by default, it can only discover NDI Sources within the same subnet as the NDI Device.

B. Considering that your network might have a substantial number of NDI Sources, due to the fact that NDI Discovery is not as fast as you might anticipate, the NDI Device will cache previously discovered sources and strive to quickly provide you with the results. Meanwhile, the NDI Device will continue to scan and update the cache in the background. As a result, the results you obtain may not be entirely up-to-date (although they are in most cases). You can periodically call this API to obtain updates.

4.2 Manual Discovery: Specified IP or NDI Groups

In most cases, we can automatically discover NDI Sources over the network. However, there are some special situations:

NDI automatic discovery cannot find NDI Sources that are not within the same subnet.

NDI has logical grouping, and by default, NDI automatic discovery only finds devices in the "public" group.

Other cases where network policies restrict network multicast.

In these situations, we may need to manually specify the target IP addresses and/or NDI Groups for NDI discovery.

API URL

/api/decoder/discovery/set_manual_targets.json

Request

Method: **POST** (Please note that this API can only be used with the POST method, and the submitted parameters need to be in JSON format.)

Parameters (example):

```
{
```

```
    "ip": ["192.168.100.3", "172.16.10.5" /*, ...*/],
    "group_name": ["public", "private_group", "my-group" /*,...*/]
}
```

Field	Value	Description
ip	**[ARRAY(STRING)]** , **Optional**	Specify one or more target IP addresses for manually discovering NDI Sources using an array. The specified IP addresses can be outside the current NDI Device's subnet. This parameter is optional. If not specified, previously designated Manual IPs will be cleared.
group_name	**[ARRAY(STRING)]** , **Optional**	Group Names. Specify one or more NDI Group Names for manually discovering NDI Sources using an array. This parameter is optional. If not specified, previously designated Group Names will be cleared.

Please note:

A) The IP addresses and NDI Groups you manually specify will be recorded and saved by the device. Therefore, even if you restart the device, these manually specified IP addresses and NDI Groups will remain effective.

B) Manually specifying IP addresses and NDI Groups will not affect the automatic discovery feature. As a result, in the list of discovery results, you will see all automatically discovered NDI Sources as well as manually specified sources (if the targets exist).

C) As indicated in the parameter description, if you do not specify "ip" or "group_name," it will clear the previous settings. If you need to retain the previous settings, be sure to pass the original values as parameters. You

can retrieve the previous settings by using the API

/api/decoder/discovery/get_manual_targets.json.

Response

Example:

```
{
    "result": "ok"
}
```

4.3 Retrieve Specified Manual Discovery IP Addresses and NDI Groups

---

API URL

/api/decoder/discovery/get_manual_targets.json

Request

Method: **GET/POST**

Parameter: NONE

Response

Example:

```
{
```

```
    "result": "ok",
    "data": {
        "ip": [
            "[\"192.168.100.3\", \"172.16.10.5\" /*, ...*/]"
        ],
        "group_name": [
            "[\"public\", \"private_group\", \"my-group\" /*,...*/]"
        ]
    }
}
```


Data Field Description：

Field	Value	Description
ip	[ARRAY(STRING)]	Return an array containing the list of all manually specified IP addresses.
group_name	[ARRAY(STRING)]	Return an array containing the list of all manually specified NDI Groups.

5. NDI Decoding: Preset

Our NDI decoding device has a very useful feature called "Preset." The significance of Preset is that you can add discovered NDI Sources to the 9 Presets represented by numbers 1 to 9, creating your "Favorites" list. When you need to quickly select and decode an NDI Source added to a Preset, you only need to specify the Preset's ID (1~9).

Preset can provide additional functional advantages for our NDI decoding device. For instance, in the future, we will support an external physical keyboard that enables rapid switching by pressing the numeric keys 1 to 9.

There is also a special Preset on the NDI decoding device, denoted as Preset 0. This Preset represents "Blank" and is used to fill the screen with a customizable color

Module name: decoder/preset

Basic URL: /api/decoder/preset/

5.1 Retrieve the Current List and Status of Presets

API URL

/api/decoder/preset/status.json

Request

Method: **GET/POST**

Parameters: NONE

Response

Example:


```
{
    "result": "ok",
    "data": [
        {
            "id": "1",
            "enable": 1, /*or 0*/
            "group": "public",
            "name": "Spark_IO-1916001002 (Channel 1)",
            "device_name": "Spark_IO-1916001002",
            "group_name":"public",
            "channel_name": "Channel 1",
            "url": "192.168.100.168:5961",
            "ip": "192.168.100.168",
            "online": "on", /*or "off"*/
        },
        /*...*/
        {
            "id": "0",
            "current": true, /*or false*/
            "color": "#000000"
        }
    ],

    "data_size": 10
```

```
}
```

Data Field Description：

"data" is a JSON array, and "data_size" is a field that assists in describing the size of the "data" array (you might not need this).

Each item in the array represents a Preset:

Field	Value	Description
id	**[STRING]**	Presets with IDs 1 to 9 represent regular NDI Source Presets; ID 0 represents "Blank."
enable	**[INT]**	"1": This Preset is defined; "0": This Preset is not defined (no NDI Source). In this case, all subsequent fields are not meaningful.
group	**[STRING]**	Group name of the NDI Source defined for this Preset.
name	**[STRING]**	Name of the NDI Source defined for this Preset.
device_name	**[STRING]**	Device Name of the NDI Source extracted from the "name" field.
channel_name	**[STRING]**	Channel Name of the NDI Source extracted from the "name" field.
url	**[STRING]**	Original URL of the NDI Source.
ip	**[STRING]**	IP address of the NDI Source extracted from the "url" field.
online	**[STRING]**	Online status of the NDI Source defined by this Preset based on network NDI discovery results: "on" - Current NDI Source is online and discoverable; "off" - Current NDI Source is offline and not discoverable. Note: The online status is determined by NDI Discovery results and does not indicate whether the NDI Source can be connected to and retrieve video data. It provides reference information and cannot be used as a definitive online determination for the device.

Field	Value	Description
current	**[BOOLEAN]**	"true": This Preset is currently being decoded; "false": This Preset is not currently being decoded.
color	**[STRING]**	Only valid for id = "0." For Preset 0 (Blank), it does not include fields like enable, group, name, device_name, channel_name, url, ip, online. Instead, it returns the "color" field representing the current Blank fill color. Refer to section 6.4 /api/decoder/preset/set_blank for the specific format description.

5.2 Add an NDI Source to a Specified Preset

API URL

/api/decoder/preset/add.json

Request

Method: **GET/POST**

Field	Value	Description
position	**[STRING]**, **Required**	Specify the Preset ID, valid range 1 to 9.
name	**[STRING]**, **Required**	NDI Source Name, you can retrieve the list of discovered NDI Sources using the API /api/decoder/discovery/get.json and extract the "name" field from it.
url	**[STRING]**, **Required**	NDI Source's original URL, you can retrieve the list of discovered NDI Sources using the API /api/decoder/discovery/get.json and extract the "url" field from it.
group	**[STRING]**, **Optional**	NDI Source's NDI group name. This parameter is optional, but if you can retrieve and extract the "group" field from the API /api/decoder/discovery/get.json, it would be beneficial to pass it as a parameter.

Please note: If you specify a Preset ID that already had an NDI Source definition, the newly added NDI Source will overwrite the old definition.

Response

Example:

```
{
    "result": "ok"
}
```

5.3 Remove NDI Source Definition from Preset

API URL

/api/decoder/preset/remove.json

Request

Method: **GET/POST**

Field	Value	Description
id	**[STRING]**, **Required**	Specify the Preset ID to be removed, with a valid range of 1 to 9.

Response

Example:

```
{
```

```
    "result": "ok"
}
```

5.4 Define the Color of Preset 0 (Blank)

API URL

/api/decoder/preset/set_blank.json

Request

Method: **GET/POST**

Field	Value	Description
color	**[STRING]**, **Required**	Specify the color of Preset 0 (Blank). The color description format follows the same conventions as HTML, which is as follows: Use RGBA color representation. Start with a "#" symbol, followed by hexadecimal values for R (red), G (green), B (blue), and A (alpha). Common color description formats include: #RGB (e.g., #f00 for red) #RRGGBB (e.g., #ff00ff for purple) #RRGGBBAA (e.g., #00ff0080 for green with 50% opacity) Please note: The current version does not support the Alpha functionality.

Response

Example:

```
{
```

```
    "result": "ok"
}
```

6. Decoding Output

Module name: decoder/current

Basic URL: /api/decoder/current/

6.1 Retrieve the Current Decoding Status.

API URL

/api/decoder/current/status.json

Request

Method: **GET/POST**

Parameters: NONE

Response

Example:

```
{
    "result": "ok",
    "data": {
        "isFull": false,
```

```
            "name": "Spark_IO-1916001002 (Channel 1)",
            "url": "192.168.100.168:5961",
            "ip": "192.168.100.168",
            "online": true, /*or false*/
            "resolution": "1920x1080p@59.94Hz",
            "codec": "NDI-FULL",
            "xRes": 1920,
            "yRes": 1080,
            "frame_rate": 59.94,
            "inst_frame_rate": 59.93,
            "bitrate": 125000,
            "audio_format": "48KHz / 2CH",
            "audio_sampling": 48000,
            "audio_channels": 2,
            "warning": "",
            "ptz":1,
            "video_mode":"hdmi"
        }
}
```


Data Field Description：

Field	Value	Description
isFull	**[BOOLEAN]**	Whether it is full NDI.
name	**[STRING]**	The name of the currently decoding NDI Source. If currently outputting Preset 0 (Blank), then name = "(Blank)".

Field	Value	Description
url	**[STRING]**	The original URL of the currently decoding NDI Source. If no NDI Source is being decoded, then url = "" (empty string).
ip	**[STRING]**	The IP address of the currently decoding NDI Source. If not decoding, then ip = "0.0.0.0".
online	**[BOOLEAN]**	true: The current NDI Source is online; false: The current NDI Source is offline.
resolution	**[STRING]**	The video resolution of the current NDI Source (user-friendly name). If you need program-friendly information about the video, such as width/height, refer to fields like xRes, yRes, frame_rate, interlaced, etc.
codec	[STRING]	The video CODEC ID of the current NDI Source: H264: ndi hx NDI-FULL: full ndi Other CODEC IDs may exist.
xRes	[INT]	The video resolution width of the current NDI Source.
yRes	[INT]	The video resolution height of the current NDI Source.
frame_rate	[NUMBER]	The frame rate of the current NDI Source. Note: frame_rate refers to the original frame rate of the NDI Source. Another field, inst_frame_rate, represents the Instant Frame Rate, the actual frame rate as calculated by the program.
inst_frame_rate	[NUMBER]	The Instant Frame Rate, the actual frame rate as calculated by the program.
bitrate	[INT]	The currently real-time statistic bitrate, in Kbps.
audio_format	[STRING]	The current audio format (user-friendly name). If you need program-friendly information about the audio, refer to audio_sampling, audio_channels.
audio_sampling	[INT]	The current audio output sampling rate.
audio_channels	[INT]	The current number of audio output channels
warning	[STRING]	The following string IDs (in bold) are defined to describe warnings/issues that may occur during

Field	Value	Description
		decoding output: WARN: url-changed Meaning: NDI source URL changed WARN: offline Meaning: Offline ERROR: no-video-output Meaning: Video output config error ERROR: no-audio-output Meaning: Audio output config error WARN: invalid Meaning: Invalid output configuration WARN: match-error Meaning: Format/resolution is unsupported WARN: unsupported-codec Meaning: CODEC is unsupported WARN: unsupported-resolution Meaning: Resolution is unsupported
ptz	**[STRING]**	ptz icon indication, where 1 means displayed and 0 means hidden.
video_mode	**[STRING]**	Audio and video signal, typically HDMI.

6.2 Switch the Preset or Specified NDI Source to Current Decoding Output

---

API URL

/api/decoder/current/set.json

Request

Method: **GET/POST**

Field	Value	Description
id	**[STRING]**, **Optional**	Specify the Preset ID (0-9) to switch the NDI Source saved under that Preset ID to the current decoding output. This parameter is optional. If you specify the id parameter, the other parameters below will be ignored. If you don't specify id, then you must provide the name and url parameters.

Field	Value	Description
		If you specify id = "0", it means the current output will be Blank, and decoding will not take place. Please refer to section 5 for more information about Preset 0.
name	**[STRING], Optional**	If you haven't specified the **id** parameter, then the **name** and **url** parameters are required. **name** is the NDI Source's name, which you can usually obtain using the 4.1 /api/decoder/discovery/get.json API. According to the existing rules of the NDI SDK, the **url** is a key parameter to establish an NDI Source connection, while **name** is provided only as a reference for discovering the NDI Source. Therefore, to some extent, you can ignore **name** or provide any arbitrary string. However, we strongly recommend that you provide the correct and valid **name**.
url	**[STRING], Optional**	The original URL of the NDI Source. If you haven't specified the id parameter, then the url parameter is required.

Response

Example:

```
{
    "result": "ok"
}
```

7. Advanced Decoding Output Options

7.1 Retrieve the Configurable Resolution List

This interface provides configurable resolution options for Interface 7.2.

API URL

/api/decoder/output/resolutions.json

Request

Response

Example:

```
{
    "result": "ok",
    "data": [
        {
            "name": "1920x1080P 60Hz",
            "value": "1080P60"
        },
        {
            "name": "1920x1080P 50Hz",
            "value": "1080P50"
        }
        /*...*/
    ]
}
```

7.2 Advanced Settings of Video/Audio Output

The default rule for NDI Decoder output video/audio is to follow the format of the NDI Source, including output video resolution/frame rate, audio sampling rate/channel count, and NDI Source consistency.

However, you have the option to specify advanced settings for video output resolution, frame rate, and audio sampling rate.

API URL

/api/decoder/output/set.json

Request

Method: **GET/POST**

Field	Value	Description
resolution	**[STRING]**, **Optional**	Configuring the decoding output resolution: You can obtain the configurable resolution options by using the 7.1 /api/decoder/output/resolutions.json interface to get the list of available resolutions. Then, use the "value" field from this list as the parameter value for this setting.
channels	**[INT]**, **Optional**	Setting audio channels: Please note that due to limitations in the current hardware capabilities, the "channels" parameter is only allowed to be set to 2. Therefore, for now, you can ignore this parameter.
audio_format	**[INT]**, **Optional**	Setting audio sampling rate: Similarly, due to limitations in the current hardware capabilities, the "sample_rate" parameter can only be set to 48000. As a result, you can also disregard this parameter for now.

Response

Example:

```
{
    "result": "ok"
```

```
}
```

7.3 Retrieve Advanced Video/Audio Output Setting Options

API URL

/api/decoder/output/get.json

Request

Method: **GET/POST**

Parameters: **NONE**

Response

Example:

```
{
    "result": "ok",
    "data": {
        "enable":       1,
        "frameRate":    0,
        "audio_format": 48000,
        "hdmi_mode":"hdmi",
        "resolution": "1080P50",
        "frame_rate": 59.94,
```

```
            "sample_rate": 48000,
            "colorspace":"auto",
            "channels":2
        }
}
```


Field	Value	Description
resolution	**[STRING]**	Refer to 7.2 Descriptions /api/decoder/output/set.json
channels	**[INT]**	Refer to 7.2 Descriptions /api/decoder/output/set.json
audio_format	**[INT]**	Refer to 7.2 Descriptions /api/decoder/output/set.json

8. Tally Status and Control

Module name: tally

Basic URL: /api/tally/

Tally status and control are effective for the Decoder mode. In Decoder mode, the Tally status you set (Program On/Off, Preview On/Off) will be sent to the NDI Source that it is currently decoding.

8.1 Retrieve Current Tally Status

API URL

/api/tally/status.json

Request

Method: **GET/POST**

Parameters: NONE

Response

Example:

```
{
    "result": "ok",
    "data": {
        "pgm": 1,
        "pvw": 0
    }
}
```

Data Field Description：

Field	Value	Description
pgm	**[INT]**	1: PGM(Program) On 0: PGM(Program) Off
pvw	**[INT]**	1: PVW(Preview) On 0: PVW(Preview) Off

8.2 Configure the Current Tally Status

API URL

/api/tally/set.json

Request

Method: **GET/POST**

Field	Value	Description
pgm	**[INT], Optional**	1: Set Program On 0: Set Program Off Not specified: Maintain the previous state
pvw	**[INT], Optional**	1: Set Preview On 0: Set Preview Off Not specified: Maintain the previous state

Response

Example:

```
{
    "result": "ok",
     "data": {
         "pgm": 1
    }
}
```

Notes:

If the NDI Device is operating in Decoder mode, the Tally status you set will be sent to the NDI Source it is currently decoding. When you switch NDI Sources, it will clear the Tally status of the previous NDI Source and send the status to the new NDI Source.

9. System Time

Module name: sys-time

Basic URL: /api/sys-time/

9.1 Obtain the current system time and configuration

API URL

/api/sys-time/get.json

Request

Method: **GET/POST**

Parameters: NONE

Response

Example:

```
{
    "result":"ok",
    "data":{
        "ntp":"ntp1.aliyun.com ntp2.aliyun.com ntp3.aliyun.com",
        "Timezone":"Asia\/Shanghai",
```

```
            "time":"2023-02-28 16:50:50",
            "timetype":"pc"
            }
}
```


Data Field Description：

Field	Value	Description
ntp	**[STRING]**	NTP server addresses, separated by spaces.
Timezone	**[STRING]**	Time Zoom
time	**[STRING]**	Current system time
timetype	**[STRING]**	Current time synchronization method: PC/NTP.

9.2 Set the Current System Time/Configuration

API URL

/api/sys-time/set.json

Request

Method: **GET/POST**

Field	Value	Description
ntp	**[STRING]**	NTP server addresses, separated by spaces.
Timezone	**[STRING]**	Time Zoom
time	**[STRING]**	Current system time
timetype	**[STRING]**	Current time synchronization method: PC/NTP.

Response

Example:

```
{
    "result": "ok"
}
```

10. System Management and Control

Module name: sys

Basic URL: /api/sys/

10.1 Retrieve the Operational Status of the Device

API URL

/api/sys/server_info.json

Request

Method: **GET/POST**

Parameters: **NONE**

Response

Example:


```
{
    "result": "ok",
    "data": {
        "addr": "192.168.0.100",
        "port": 443,
        "name": "https://192.168.0.100",
        "persis": "4H 15M 32S",
        "start_time": "2020-05-03 12:10:09",
        "cpu_cores": 2,
        "cpu_payload": 33,
        "mem_used": 124.06,
        "mem_total": 620.00
    }
}
```


Field	Value	Description
addr	**[STRING]**	Current device's IP address
port	**[INT]**	The port of the HTTP API you are accessing (Default HTTP: 80, HTTPS: 443)
name	**[STRING]**	Note: This is a field that may be deprecated or modified. Currently, it represents the URL you use to access the HTTP API.
persis	**[STRING]**	Duration of device operation, formatted as: <hours>H <minutes>M <seconds>S
start_time	**[STRING]**	Device's start-up time, formatted as: Year-Month-Day Hour:Minute:Second
cpu_cores	**[INT]**	Current number of CPU cores in the device

Field	Value	Description
cpu_payload	**[INT]**	Current CPU load %
mem_used	**[NUM]**	Current amount of used memory, in MB
mem_total	**[NUM]**	Total available memory in the device, in MB

10.2 Restart the Device

This operation will initiate a restart of the NDI device.

When calling this API, the HTTP response will be immediate, but the actual restart process of the device will begin 3 seconds after a successful API execution. The entire restart process takes approximately 20 seconds. As the device's network typically uses DHCP to obtain an address, the time you can expect to regain access to the device will be >= 30 seconds, depending on your actual network conditions.

During the device restart process, you will be unable to access any HTTP APIs until the restart is complete.

Please note: After the device restarts, your existing security credentials (Session ID and Authorization Token) will become invalid. You must follow the method described in Section 2 to reauthorize the HTTP API access.

API URL

/api/sys/reboot.json

Request

Method: **GET/POST**

Parameters: **NONE**

Response

Example:

```
{
    "result": "ok"
}
```

10.3 Restore to Factory Settings

This operation will restore the NDI Device to its default factory settings.

The affected settings that will be reverted include:

- Network address acquisition method will be restored to DHCP.
- Network's Failsafe address will be restored to the factory default value (192.168.100.168 or 192.168.1.168, depending on the device. Refer to the product manual for details).
- NDI Group will be restored to the default value (public).

• NDI Device Name will be restored to the format "<PRODCT_TYPE>-<SERIAL-NUMBER>".

• NDI Channel Name will be restored to the factory default name (typically Channel-1, but refer to the actual product).

• NDI Connection method will be restored to the default (TCP) connection.

• Current decoding Source and Preset of NDI Decoder will be cleared, and Preset 0 (Blank) color will be restored to default (black).

• All users will be removed, and the admin user will have its password reset to the default.

When calling this API, the HTTP response will be immediate, but the device will begin the restart process 3 seconds after a successful API execution. The restart behavior and its impact will be consistent with the description in Section 10.3.

API URL: [Insert API URL here]

API URL

/api/sys/restore.json

Request

Method: **GET/POST**

Parameters: **NONE**

Response

Example:

```
{
    "result": "ok"
}
```

10.4 Retrieve the Host Name

API URL

/api/device/get_hostname.json

Request

Method: **GET/POST**

Response

Example:

```
{
    "result": "ok",
    "data": {
```

```
        "hostname": "N6-123456"
    }
}
```


10.5 Set the Host Name

API URL

/api/device/set_hostname.json

Request

Method: **GET/POST**

Field	Value	Description
hostname	**[STRING]**	Host Name

Response

Example:


```
{
    "result": "ok"
}
```


Notes: After modifying the hostname, NDI will restart to accommodate the new hostname.

10.6 Configure “Discovery Server”

API URL

api/device/set_discovery_server.json

Request

Method: **GET/POST**

Field	Value	Description
address	**[STRING]**	Discovery server addresses, multiple discovery servers can be separated by ",".
enable	**[BOOLEAN]**	Enable: true or false.

Response

Example:

```
{
    "result": "ok"
}
```

10.7 Obtain the Status of the Discovery Server

API URL

/api/device/get_discovery_server.json

Request

Method: **GET/POST**

Response

Example:

```
{
    "result": "ok",
    "data": {
        "address": "192.168.1.168",
        "enable": true
    }
}
```

10.8 Get the Current System Status

Get current system status including CPU Status, memory status, system uptime and other status information.

API URL

/api/sys/server_info.json

Request

Method: **GET/POST**

Response

Example:


```
{
    "result": "ok",
    "data": {
        "cpu_cores": 8,
        "cpu_payload": 22,
        "mem_used": 300.15,
        "addr": "192.168.40.161",
        "persis": "1H 20M 23S",
        "port": "443",
        "name": "https://192.168.40.161",
        "start_time": "2023-02-28 15:47:21",
        "mem_total": 1889.72
    }
}
```


Data Field Description：

Field	Value	Description
cpu_cores	**[INT]**	CPU Cores
cpu_payload	**[NUMBER]**	CPU Usage Rate (%)
mem_total	**[STRING]**	Total Memory (MB)
mem_used	**[STRING]**	Used Memory (MB)
addr	**[STRING]**	Device IP
name	**[STRING]**	Device Web Address
port	**[STRING]**	web https Port

Field	Value	Description
persis	**[STRING]**	Device Uptime
start_time	**[STRING]**	Device Boot Time

10.9 Obtain the System Version

API URL

/api/firmware/get.json

Request

Method: **GET/POST**

Response

Example:

```
{
    "result": "ok",
    "data": {
        "softwareVersion": "1.01.0001",
        "hardwareVersion": "1.0",
        "ndiVersion": "5.1.3"
    }
}
```

Field	Value	Description
softwareVersion	**[STRING]**	Software Version
hardwareVersion	**[STRING]**	Hardware Version
ndiVersion	**[STRING]**	NDI Version

11. User Management

Module name: users

Basic URL: /api/users/

11.1 Retrieve the Current User List

API URL

/api/users/list.json

Request

Method: **GET/POST**

Parameters: **NONE**

Response

Example:

```
{
    "result": "ok",
```

```
    "data": [
        {
            "alias": "Admin",
            "api": true,
            "create_time": "1970-01-01 00:00:00",
            "id": "admin",
            "web": true
        }
    ]
}
```


Data Field Description：

Field	Value	Description
alias	**[STRING]**	Nickname
id	**[STRING]**	User Name
api	**[BOOLEAN]**	Enable API Permissions
web	**[BOOLEAN]**	Enable Web Login Permissions
create_time	**[STRING]**	User Creation Time

11.2 Adding a User

API URL

/api/users/add.json

Request

Method: **GET/POST**

Parameters:

Field	Value	Description
alias	**[STRING]**	Alias
id	**[STRING]**	User Name
api	**[BOOLEAN]**	Enable API Permissions
web	**[BOOLEAN]**	Enable Web Login Permissions
password	**[STRING]**	Password

Response

Example:

```
{
    "result": "ok"
}
```

11.3 Modify User Information

API URL

/api/users/modify.json

Request

Method: **GET/POST**

Parameters:

Field	Value	Description
id	**[STRING]**	Username (required, cannot modify this field)
alias	**[STRING]**	Alias
api	**[BOOLEAN]**	Enable API Permissions
web	**[BOOLEAN]**	Enable Web Login Permissions
password	**[STRING]**	Password

Response

Example:

```
{
    "result": "ok"
}
```

11.4 Delete a User

API URL

/api/users/remove.json

Request

Method: **GET/POST**

Parameters:

Field	Value	Description
id	**[STRING] /[ARRAY(STRING)]**	Username (string or array)

Response

Example:

```
{
    "result": "ok"
}
```

12. Network Configuration

Module name: network

Basic URL: /api/network/

12.1 Get Current Network Configuration

API URL

/api/network/get.json

Request

Method: **GET/POST**

Parameters: **NONE**

Response

Example:


```
{
    "result": "ok",
    "data": [
        {
         "device": "eth0",
         "state": "up",
         "ip": "192.168.2.160",
         "mask": "255.255.255.0",
         "mac": "68:3A:7F:8C:A7:96",
         "gw": "192.168.2.1",
         "dynamic": "y",
         "dns": "8.8.8.8; 4.4.4.4",
        }
    ],
    "data_size": 1
}
```


Data Field Description：

Field	Value	Description
device	**[STRING]**	Current network interface device name. Please note this device name, as it will be used as a reference when modifying network configuration. In other words, when you modify the network configuration, you must specify this device name.

Field	Value	Description
state	**[STRING]**	Current network operational status: "up": Network is operational and working normally. "down": Network is disconnected due to unplugged Ethernet cable. "disabled": Network is disabled for other reasons. "error": Network is experiencing operational issues.
ip	**[STRING]**	Current IP address configuration. Note: If your network is configured to use DHCP, the "ip" will display the actual IP address obtained through DHCP. If the network is configured with a static IP address, "ip" will represent the static IP address you have set.
mask	**[STRING]**	Current subnet mask configuration (netmask).
mac	**[STRING]**	MAC address of the current network interface.
gw	**[STRING]**	Current default gateway of the network. If not set, it may appear as "" (empty string) or "0.0.0.0".
dynamic	**[STRING]**	"y": DHCP configuration is enabled. "n": Static IP address configuration is used.
dns	**[STRING]**	Current DNS server addresses. Multiple addresses are separated by ';'

12.2 Modify Network Configuration

Important Note: Modifying network configuration is a sensitive operation, and incorrect configuration may result in the device becoming inaccessible. Therefore, we remind you to take note of the following:

1. For routine network address modifications, please use the NDI Device's Web UI whenever possible.

2. In case of any accessibility issues caused by modifying network parameters, you can use the device's Reset button to restore it to factory settings.

API URL

/api/network/set.json

Request

Method: **GET/POST**

Field	Value	Description
device	**[STRING], Required**	Specify the name of the network interface to be modified. Refer to Section 12.1 in /api/network/get.json for details on the "device" parameter; this parameter is mandatory.
dynamic	**[STRING], Optional**	y: Use DHCP for IP address assignment. If dynamic="y", parameters like ip, mask, gw, and dns will have no meaning. n: Configure with a static IP address.
ip	**[STRING], Optional**	Configure the IP Address
mask	**[STRING], Optional**	Subnet mask (netmask)
mac	**[STRING], Optional**	If specifying a MAC address (mac), you can modify the current network interface MAC address. Note: You must provide a valid and legal MAC address. Only change the device's MAC address if you have a clear intention to do so.
gw	**[STRING], Optional**	Default gateway (gateway). Specify "" (empty string) to indicate no gateway configuration.

Field	Value	Description
dns	**[STRING], Optional**	Specify zero or more DNS server addresses. Separate multiple addresses with ';' character. Use "" (empty string) to indicate no DNS configuration.

Response

Example:

```
{
    "result": "ok"
}
```

Note: After modifying the network address, if the IP address changes, you must use the new IP address for HTTP API access. The changes to the network address typically take effect in less than 1 second. However, if you specify an IP address for DHCP assignment, the time it takes to obtain a valid IP address depends on your actual network environment and may vary.