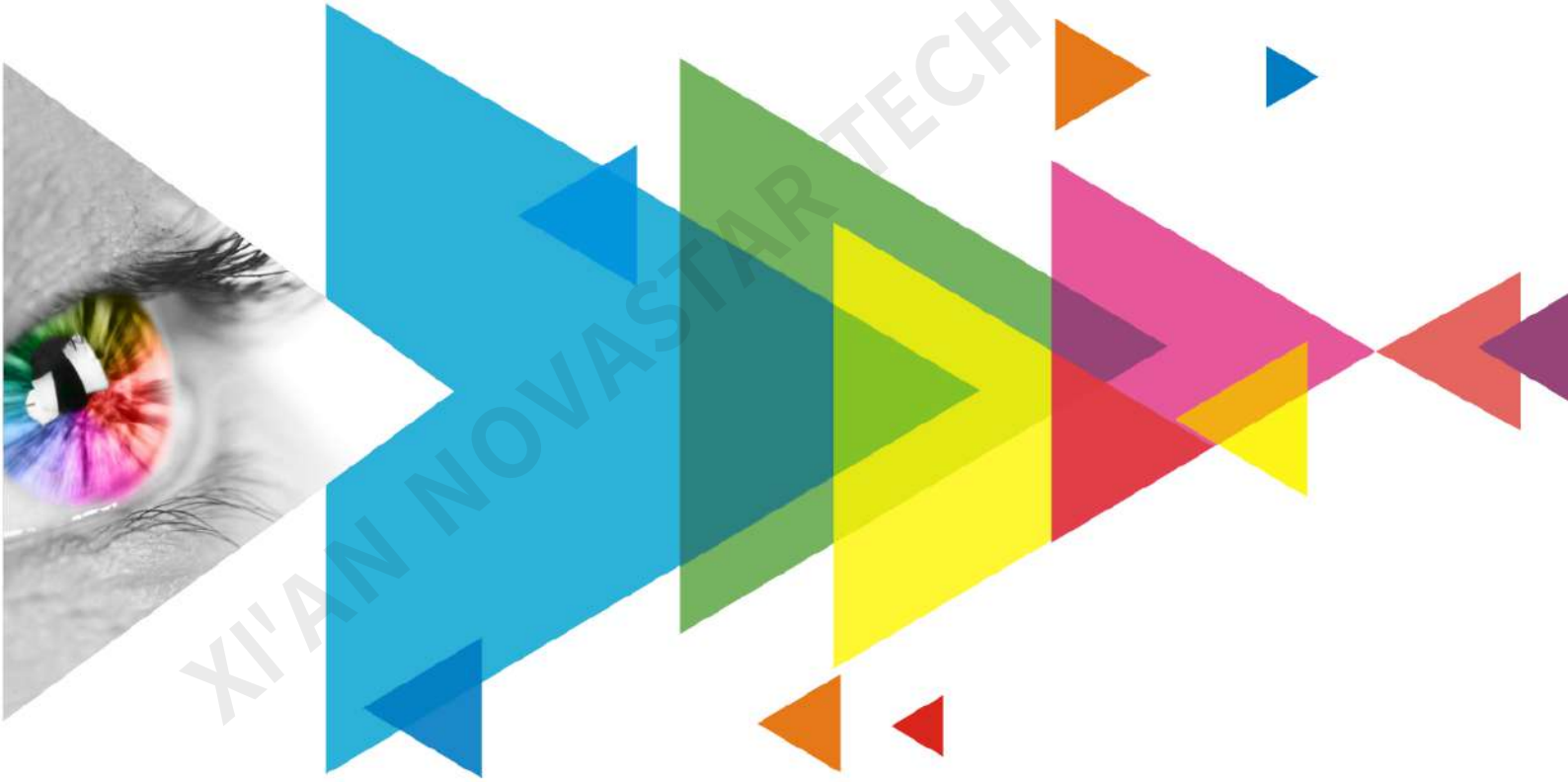
COEX Series
Interface API



User Manual

Contents

Contents i
1 General Introduction 1
1.1 About This Document 1
1.2 Applications 1
1.3 Function Overview 1
1.4 Supported Protocols 2
1.5 Must-Reads Before You Begin 2
2 Display Control 4
2.1 Set Display Mode 4
3 Input Source 5
3.1 Select Input Source 5
3.2 Obtain Input Source Information 6
3.3 EDID Settings 9
3.4 InfoFrame Override 10
3.5 Internal Source 13
4 Output 15
4.1 Brightness Adjustment 15
4.2 Gamma Adjustment 16
4.3 Color Gamut Switching 17
4.4 Color Temperature Adjustment 18
4.5 Output Bit Depth Adjustment 19
5 Others 20
5.1 Project File 20
5.2 Switch Presets 21
5.3 Set Working Mode 22

1 General Introduction

1.1 About This Document

The API interface provides the basic functions of COEX series controllers.

COEX series products include the CX80 Pro, CX40 Pro, MX40 Pro and KU20.

1.2 Applications

This API reference document of COEX series controllers is provided for users to realize secondary development.

1.3 Function Overview

Function Module	Function Description	Remarks
Display control	Normal	Full screen display mode
	Blackout	
	Freeze	
Input source	Select input source	Select an input source as the current display content of the LED screen
	Show the information	Show the basic information on the selected input source, including input source type, group ID, input source ID, EDID and other information
	EDID	Set the resolution and frame rate for the selected input source
	InfoFrame override	Then information to override includes color space, color gamut, and quantization range
	Internal source	Select a test pattern as the active input source of the LED screen
Output	Brightness	Adjust the brightness of a single or multiple cabinets
	Gamma	Adjust the gamma of a single or multiple cabinets
	Color gamut	Switch the color gamut of a single or multiple cabinets
	Color temperature	Adjust the color temperature of a single or multiple cabinets
	Bit depth	Adjust the output bit depth
Others	Project file	Import and export the project file of controller
	Preset	Switch the currently applied preset
	Working mode	Switch the working mode between Send-Only Controller and All-In-One Controller modes

1.4 Supported Protocols

Only the HTTP protocol is supported. The protocol port number used is 8001. The IP is displayed on the LCD home screen of the controller.

1.5 Must-Reads Before You Begin

1.5.1 Global Response Codes

Error Code	Field	Description
0	Success	Successful
1	InvalidParam	Invalid parameter entered
2	SendFailed	Failed to send
3	InternalErr	Internal error
4	AnalysisFailed	Data parsing failed
5	Busying	Device busy
6	NotSupport	Not supported function
7	LengthError	Parameter length error
8	SerializeError	Serialization error
9	NoSupportIrModule	The irregular module is not supported
10	OpenFileFailed	Failed to open file
11	CloseFileFailed	Failed to close file
12	ReadError	Failed to read back
13	CreateDirFailed	Failed to create directory
14	ReadFileFailed	Failed to read file
15	DecodeFailed	Failed to decode
16	EncodeFailed	Failed to encode
17	WriteFileFailed	Failed to write to file
18	RequestTimeout	Request timed out
19	ResponseErr	Response error
20	DBOperErr	Database operation error
21	RvCardSoftSpaceHeaderInvalid	Invalid header data of the receiving card software space
22	TagNotMatch	The tags do not match
23	UnknownSubBoardType	Unknown card type

24	FindComponentObjErr	An error occurred while searching for component object
25	InterfaceConvertErr	Interface conversion error
26	SerializeDataFailed	Failed to serialize data
27	FunctionalRestrictions	The constraints for enabling the function do not meet the requirements
28	InvalidPointer	Null pointer
29	LowDelayFunctionalRestrictions	The constraints for enabling low latency function do not meet the requirements
30	ThreadDFunctionalRestrictions	The constraints for enabling 3D function do not meet the requirements
31	GenLockFunctionalRestrictions	The constraints for enabling Genlock function do not meet the requirements
32	AdditionFrameDelayFunctionalRestrictions	The constraints for enabling additional latency function do not meet the requirements
33	MultiplierFunctionalRestrictions	The constraints for enabling frame multiplication function do not meet the requirements
34	ScreenYPosIsNotEqualForOpenLowDelay	The Y coordinates of the circumscribed rectangles for the cabinets loaded by each Ethernet port are inconsistent when low latency is turned on
35	NoScreenLayoutInfo	No screen layout
36	NotImplement	Interface not implemented
37	PresetNameRepeatError	The preset name already exits
38	MemoryNotEnough	Insufficient memory
39	CfgFileNotExist	The cabinet file does not exist
41	NonStandardFileName	The file name does not meet requirements because it contains special characters

1.5.2 Response Example

Unless otherwise specified by the interface, the data format is in JSON format. The PUT request body data is requested or returned in standard JSON object format. If a PUT request returns an error, it may be that the request body JSON format is wrong or missing. A response example of successful PUT request is as follows:

```
{
    "code": 0,
    "data": null,
    "message": "Success"
}
```

2 Display Control

2.1 Set Display Mode

Interface Descriptions

The display mode is the current display status of the LED screen. Display modes include normal, blackout, and freeze.

Request Method

PUT

Request URL

/api/v1/device/screen/displaymode

Request Parameters

Parameter	Required	Type	Description
value	Yes	uint8	Display mode: 【 0: Normal 1: Blackout 2: Freeze 】

- Request example: Set the display status to normal status.

```
PUT http://{ip}:{port}/api/v1/device/screen/displaymode
{
    "value":0
}
```

Response Data Description

If the setting is successful, the returned error code is 0, and the corresponding information is “Success”. See 1.5.2 Response **Example** for response example.

Response Status Code

For details, see 1.5.1 Global Response Codes.

3 Input Source

3.1 Select Input Source

Interface Descriptions

In the Send-Only Controller working mode, select an input source as the current display content of the LED screen.

Request Method

PUT

Request URL

/api/v1/device/screen/input

Request Parameters

Parameter	Required	Type	Description
groupId	Yes	uint8	This parameter can be obtained from the input source information. It corresponds to the groupId field in the input source information. When the groupId of the corresponding input source is entered, the corresponding source will be selected to display on the screen.

- Request example: Switch the input source.

PUT http://{ip}:{port}/api/v1/device/screen/input

```
{
    "groupId":0
}
```

Response Data Description

If the setting is successful, the returned error code is 0, and the corresponding information is “Success”. See 1.5.2 Response **Example** for response example.

Response Status Code

For details, see 1.5.1 Global Response Codes.

3.2 Obtain Input Source Information

Interface Descriptions

Obtain the information on the input source currently connected to the controller, including input source type, group ID, input source ID, EDID and other information.

Request Method

GET

Request URL

/api/v1/device/input/sources

Request Parameters

None

- Request example: Obtain all the input source information.

GET http://{ip}:{port}/api/v1/device/input/sources

Response Parameters

Parameter	Type	Description
id	int	Input connector No.
type	uint8	Input connector type: HDMI, SDI, etc.: 【 0x00: DVI 0x01: DualDVI 0x02: HDMI1.4 0x03: HDMI2.0 0x04: DP1.1 0x05: DP1.2 0x06: DP1.4 0x07: SDI_3G 0x08: SDI_6G 0x09: SDI_12G 0x0a: PIPVideo 0x10: HDMI1.3

			0x11: HDMI2.1 0x12: PCIE 0x13: Serdes 0x14: LVDS 0xE0: InternalSource 0xFF: NoInputType]
name			string	Input source name
step			int	The horizontal stepping when customizing EDID resolution
supportFrameRate			string	All the supported frame rates
supportResolution			string	All the supported resolutions
maxwidth			int	The maximum resolution width supported
maxheight			int	The maximum resolution height supported
minwidth			int	The minimum resolution width supported
minheight			int	The minimum resolution height supported
actualResolution			Object	The actual resolution
	height		int	Width
	width		int	Height
actualRefreshRate			float	The actual frame rate
bitDepth			int	The input source bit depth
colorSpace			string	The video format
dynamicRange			string	The dynamic range
gamut			string	The color gamut
range			int	0x00: limit (Limited) 0x01: full (Full)
scanMode			int	0x00: progressive signal scanning 0x01: Interlaced signal scanning
defaultEDID			Object	The default resolution
	resolution		Object	Resolution
		width	int32	Width
		height	int32	Height
	refreshRate		float	Frame rate
usable			bool	Connector usage status

groupId		uint8	Group ID
isSupportHDR		bool	Whether HDR is supported
isSupportMetaData		bool	Whether metaData is supported
isSupportEDID		bool	Whether EDID settings are supported
isSupportInputOverride		bool	Whether InfoFrame Override is supported
isSupportColorAdjust		bool	Whether color adjustment is supported
sourceChannel		uint8	The input source channel
metaData		Object	metaData
	minMasterDisplayLight	float	Min Display Mastering Luminance
	maxMasterDisplayLight	float	Max Display Mastering Luminance
	maxContentLight	float	Max Content Light Level (MaxCLL)
	maxFrameAvgLight	float	Max Frame-Average Light Level (MaxFLL)
	whitePointX	float	Whitepoint coordinate X
	whitePointY	float	Whitepoint coordinate Y
hDRParams			
	overrideHdrType	int	The current override mode: 【 0x00: HDR10 0x01: HLG 0x02: SDR 0x03: DCI 0xFF: AUTO 】
	pqMode	int	PQ mode
	pqMaxCllChecked	bool	Whether MaxCLL override is enabled
	pqMaxCll	int	MaxCLL value
	realHdrType	int	The real mode. If the HDR mode is overridden to automatic, this value is the dynamic range type of the input source. If the HDR mode is overridden to other modes, this value is the same as the mode after overridden.
isSupportHDRParams		bool	Whether HDR parameters are supported
isSupportPQMaxCllChecked		bool	Whether “override” for Max Content Light Level (MaxCLL) in HDR PQ mode is enabled

hdrList	array	List of HDR types supported by input source: 【 0x00: HDR10 0x01: HLG 0x02: SDR 0x03: DCI 0xFF: AUTO 】

3.3 EDID Settings

Interface Descriptions

The input source standard resolution and frame rate can be set through EDID. Custom resolution setting is included.

Request Method

PUT

Request URL

/api/v1/device/input/{id}/edid

Request Parameters

Parameter		Required	Type	Description
resolution		Yes	Object	Resolution
	width	Yes	int32	Width
	height	Yes	int32	Height
refreshRate		Yes	float32	Frame rate

- Request example: Set EDID to 3840*2160@60Hz.

PUT http://{ip}:{port}/api/v1/device/input/1/edid

```
{
    "resolution":{
        "width":3840,
        "height":2160
    }
```

```
        "refreshRate":60.00
    }
```

The parameter "1" is the input source ID, which can be obtained from the input source information. The input source ID corresponds to the id field in the input source information.

Response Data Description

If the setting is successful, the returned error code is 0, and the corresponding information is “Success”. See 1.5.2 Response **Example** for response example.

Response Status Code

For details, see 1.5.1 Global Response Codes.

3.4 InfoFrame Override

3.4.1 Color Space Override

Interface Descriptions

The color space of the input source can be modified through color space override.

Request Method

PUT

Request URL

/api/v1/device/input/{id}/colorspace

Request Parameters

Parameter	Required	Type	Description
colorSpace	Yes	uint8	Color space types: 【 0x00: RGB 0x01: YUV444 0x02: YUV422 0x03: YUV420 0x04: XYZ 0xff:AutoColorSpace 】

- Request example: Override the color space of the input source to RGB.

PUT http://{ip}:{port}/api/v1/device/input/1/colorspace

```
{
    "colorSpace":0
}
```

The parameter "1" is the input source ID, which can be obtained from the input source information. The input source ID corresponds to the id field in the input source information.

Response Data Description

If the setting is successful, the returned error code is 0, and the corresponding information is “Success”. See 1.5.2 Response **Example** for response example.

Response Status Code

For details, see 1.5.1 Global Response Codes.

3.4.2 Color Gamut Override

Interface Descriptions

The color gamut of the input source can be modified through color gamut override.

Request Method

PUT

Request URL

/api/v1/device/input/{id}/colourgamut

Request Parameters

Parameter	Required	Type	Description
colourGamut	Yes	uint8	Standard color gamut types 【 0x02: Rec.709 0x03: Rec.2020 0x04: DCI-P3 0xff:AutoColorGamut 】

- Request example: Override the color gamut of the input source to Rec.709.

PUT http://{ip}:{port}/api/v1/device/input/1/colourgamut

```
{
    "colourGamut":2
}
```

The parameter "1" is the input source ID, which can be obtained from the input source information. The input source ID corresponds to the id field in the input source information.

Response Data Description

If the setting is successful, the returned error code is 0, and the corresponding information is “Success”. See 1.5.2 Response **Example** for response example.

Response Status Code

For details, see 1.5.1 Global Response Codes.

3.4.3 Quantization Range Override

Interface Descriptions

The quantization range of the input source can be modified through quantization range override.

Request Method

PUT

Request URL

/api/v1/device/input/{id}/range

Request Parameters

Parameter	Required	Type	Description
range	Yes	uint8	Quantization range types: 【 0x00: Limit 0x01: Full 0xff: AutoRange 】

- Request example: Override the quantization range of the input source to Limited.

PUT http://{ip}:{port}/api/v1/device/input/1/range

```
{
```

```
        "range":0
    }
```

The parameter "1" is the input source ID, which can be obtained from the input source information. The input source ID corresponds to the id field in the input source information.

Response Data Description

If the setting is successful, the returned error code is 0, and the corresponding information is “Success”. See 1.5.2 Response **Example** for response example.

Response Status Code

For details, see 1.5.1 Global Response Codes.

3.5 Internal Source

3.5.1 Select a Test Pattern

Interface Descriptions

Select a test pattern as the current display content of the LED screen.

Request Method

PUT

Request URL

/api/v1/device/screen/controller/pattern/test

Request Parameters

Parameter	Required	Type	Description
mode	Yes	uint8	Mode: 【 0: Pure color (the color is controlled by the red, green and blue component values below) 16: Horizontal stripes to the bottom 17: Horizontal stripes to the right 18: Slashes 19: Backslashes 20: Grid to the bottom right 21: Grid to the right 32: Left-to-right red gradient

				33: Left-to-right green gradient 34: Left-to-right blue gradient 35: Left-to-right gray gradient 36: Top-to-bottom red gradient 37: Top-to-bottom green gradient 38: Top-to-bottom blue gradient 39: Top-to-bottom gray gradient 48: Lightning **]**
parameters		Yes	Object	Parameters
	red	Yes	uint16	Red component
	green	Yes	uint16	Green component
	blue	Yes	uint16	Blue component
	gray	Yes	uint16	Grayscale
	gridWidth	Yes	uint8	Grid
	moveSpeed	Yes	uint8	Moving speed
	gradientStretch	Yes	uint16	Grid size
	state	Yes	uint8	Status

- Request example: Select a pure white test pattern.

```
PUT  http://{ip}:{port}/api/v1/device/screen/controller/pattern/test
    {
        "mode": 0,
        "parameters": {
            red": 4095,
            "green": 4095,
            "blue": 4095,
            "gray": 4095,
            "gridWidth": 1,
            "moveSpeed": 50,
            "gradientStretch": 8,
            "state": 0
        }
    }
```

Response Data Description

If the setting is successful, the returned error code is 0, and the corresponding information is “Success”. See 1.5.2 Response **Example** for response example.

Response Status Code

For details, see 1.5.1 Global Response Codes.

4 Output

4.1 Brightness Adjustment

Interface Descriptions

Adjust the LED screen brightness by cabinet (a single or multiple cabinets).

Request Method

PUT

Request URL

/api/v1//device/cabinet/brightness

Request Parameters

Parameter	Required	Type	Description
idList	Yes	[]uint64	ID list of the cabinets that are operated
ratio	Yes	float	The brightness percentage of the cabinet
nit	No	uint16	The nit value corresponding to the current cabinet brightness

- Request example: Set the LED screen brightness to 100%.

PUT http://{ip}:{port}/api/v1//device/cabinet/brightness

```
{
    "idList": [
        93138183199495
    ],
    "ratio": 1.0,
    "nit": 1000
}
```

Response Data Description

If the setting is successful, the returned error code is 0, and the corresponding information is “Success”. See 1.5.2 Response **Example** for response example.

Response Status Code

For details, see 1.5.1 Global Response Codes.

4.2 Gamma Adjustment

Interface Descriptions

Adjust the LED screen gamma value by cabinet (a single or multiple cabinets).

Request Method

PUT

Request URL

/api/v1/device/cabinet/gamma

Request Parameters

Parameter	Required	Type	Description
idList	Yes	[]uint64	ID list of the cabinets that are operated
type	No	int	RGB type: 【 0x00: Red gamma 0x01: Green gamma 0x02: Blue gamma 0x03: All 】
value	Yes	float	Gamma value (1.0 to 4.0)

- Request example: Set the gamma value to 2.8.

PUT http://{ip}:{port}/api/v1/device/cabinet/gamma

```
{
    "idList": [
        93138183199495
```

```
        ],
        "type": 3,
        "value": 2.8
    }
```

Response Data Description

If the setting is successful, the returned error code is 0, and the corresponding information is “Success”. See 1.5.2 Response **Example** for response example.

Response Status Code

For details, see 1.5.1 Global Response Codes.

4.3 Color Gamut Switching

Interface Descriptions

After the LED screen color gamut is corrected, the original color gamut of the screen can be managed to the selected target color gamut as much as possible by switching the color gamut (taking the intersection of the horseshoe-like diagram of the maximum color gamut supported by the screen and the horseshoe-like diagram of the target color gamut).

Request Method

PUT

Request URL

/api/v1/device/correctionop/cabinets/gamut

Request Parameters

Parameter	Required	Type	Description
name	Yes	string	The name of the color gamut after switching 【 "From input" "DCI-P3" "Rec.709" "Rec.2020" "Custom" (the default custom color gamut name) "" (the original color gamut) 】

- Request example: Switch the color gamut to DCI-P3.

```
PUT  http://{ip}:{port}/api/v1/device/correctionop/cabinets/gamut
    {
        "name": "DCI-P3"
    }
```

Response Data Description

If the setting is successful, the returned error code is 0, and the corresponding information is “Success”. See 1.5.2 Response **Example** for response example.

Response Status Code

For details, see 1.5.1 Global Response Codes.

4.4 Color Temperature Adjustment

Interface Descriptions

Color temperature adjustment is to control the color temperature of the LED screen with a single or multiple cabinets by adjusting the RGB ratio of the screen. The adjustment options include single-cabinet and multiple-cabinet adjustments.

Request Method

PUT

Request URL

/api/v1/device/cabinet/colortemperature

Request Parameters

Parameter	Required	Type	Description
idList	Yes	[uint64	ID list of the cabinets that are operated
value	Yes	uint32	Cabinet color temperature value (1700K to 15000K)

- Request example: Set the color temperature to 6500K.

```
PUT  http://{ip}:{port}/api/v1/device/cabinet/colortemperature
    {
        "idList": [
            93138183199495
```

```
        ],
        "value": 6500
    }
```

Response Data Description

If the setting is successful, the returned error code is 0, and the corresponding information is “Success”. See 1.5.2 Response **Example** for response example.

Response Status Code

For details, see 1.5.1 Global Response Codes.

4.5 Output Bit Depth Adjustment

Interface Descriptions

Adjust the output bit depth of the controller.

Request Method

PUT

Request URL

/api/v1/device/screen/video/bitdepth

Request Parameters

Parameter	Required	Type	Description
bitDepth	Yes	int	Output bit depth: 【 0: 8bit 1: 10bit 2: 12bit 255: Automatic 】

- Request example: Set the output bit depth to 8bit.

```
PUT  http://{ip}:{port}/api/v1/device/screen/video/bitdepth
    {
        "bitDepth":0
    }
```

Response Data Description

If the setting is successful, the returned error code is 0, and the corresponding information is “Success”. See 1.5.2 Response **Example** for response example.

Response Status Code

For details, see 1.5.1 Global Response Codes.

5 Others

5.1 Project File

5.1.1 Export Project File

Interface Descriptions

Export the configuration file data of the controller.

Request Method

GET

Request URL

/api/v1/device/hw/deviceengineeringdocdata

Request Parameters

None

Response Parameters

Parameter	Required	Type	Description
backupFile	Yes	string	The base64 encoding of the controller configuration data

5.1.2 Import Project File

Interface Descriptions

Import the configuration file data to the controller.

Request Method

PUT

Request URL

/api/v1/device/hw/deviceengineeringdocdata

Request Parameters

Parameter	Required	Type	Description
backupFile	Yes	string	The base64 encoding of the controller configuration data

5.2 Switch Presets

Interface Descriptions

Switch the preset currently used by the controller.

Request Method

PUT

Request URL

/api/v1/device/currentpreset

Request Parameters

Parameter	Required	Type	Description
sequenceNumber	Yes	int8	Set the currently applied preset (1 to 50)

- Request example: Apply preset 1.

PUT http://{ip}:{port}/api/v1/device/currentpreset

```
{
    "sequenceNumber":1
}
```

Response Data Description

If the setting is successful, the returned error code is 0, and the corresponding information is “Success”. See 1.5.2 Response **Example** for response example.

Response Status Code

For details, see 1.5.1 Global Response Codes.

5.3 Set Working Mode

Interface Descriptions

Set the working mode of the controller. Currently there are two working modes: Send-Only Controller and All-In-One Controller.

Request Method

PUT

Request URL

/api/v1/device/hw/mode

Request Parameters

Parameter	Required	Type	Description
mode	Yes	uint8	Working mode: 【 2: Send-Only Controller mode 3: All-In-One Controller mode 】

- Request example: Set the working mode to Send-Only Controller.

PUT http://{ip}:{port}/api/v1/device/hw/mode

```
{
    "mode":2
}
```

Response Data Description

If the setting is successful, the returned error code is 0, and the corresponding information is “Success”. See 1.5.2 Response **Example** for response example.

Response Status Code

For details, see 1.5.1 Global Response Codes.

Trademark

NOVASTAR is a trademark of Xi'an NovaStar Tech Co., Ltd.

Statement

Thank you for choosing NovaStar's product. This document is intended to help you understand and use the product. For accuracy and reliability, NovaStar may make improvements and/or changes to this document at any time and without notice. If you experience any problems in use or have any suggestions, please contact us via the contact information given in this document. We will do our best to solve any issues, as well as evaluate and implement any suggestions.

Official website
www.novastar.tech

Technical support
support@novastar.tech