1 Introduction

Note: This document applies to VL-BYOD200 firmware version V1.1.0 or later.

1.1 Preparation

This section takes a third-party control device windows 10 as an example. You may also use other control devices.

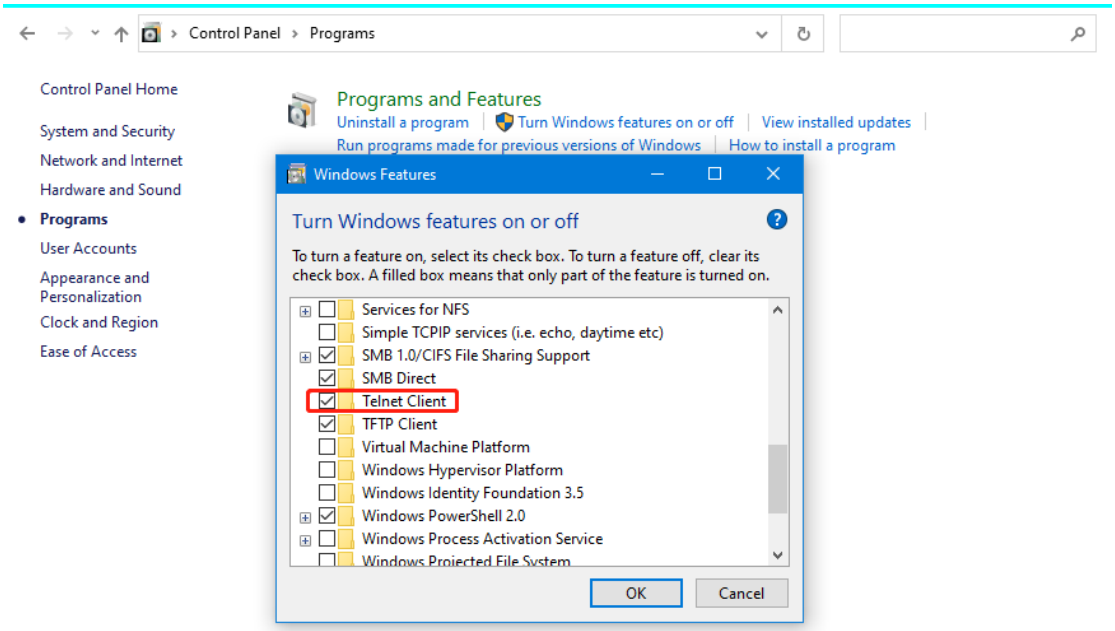
1.1.1 Setting IP Address in Your Computer

The detailed operation steps are omitted here, please consult your IT manager or search online for relevant guidance.

1.1.2 Enabling Telnet Client

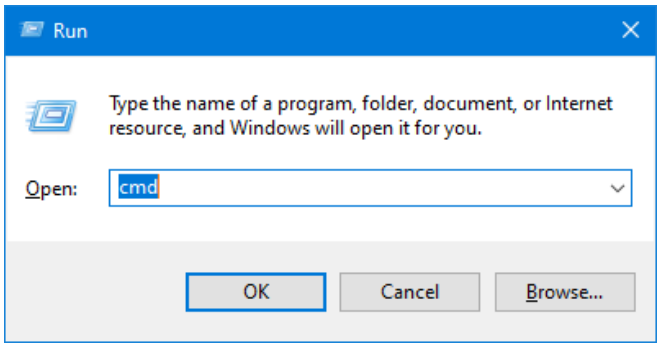
Before logging in to the device via command-line interface, make sure that **Telnet Client** is enabled. By default, **Telnet Client** is disabled in Windows OS. To turn on **Telnet Client**, do as follows.

1. Choose **Start** > **Windows System** > **Control Panel**.
2. In the popped dialog, click **Programs**.
3. In the **Programs and Features** area, click **Turn Windows features on or off**.
4. In **Windows Features** dialog box, select **Telnet Client** check box.

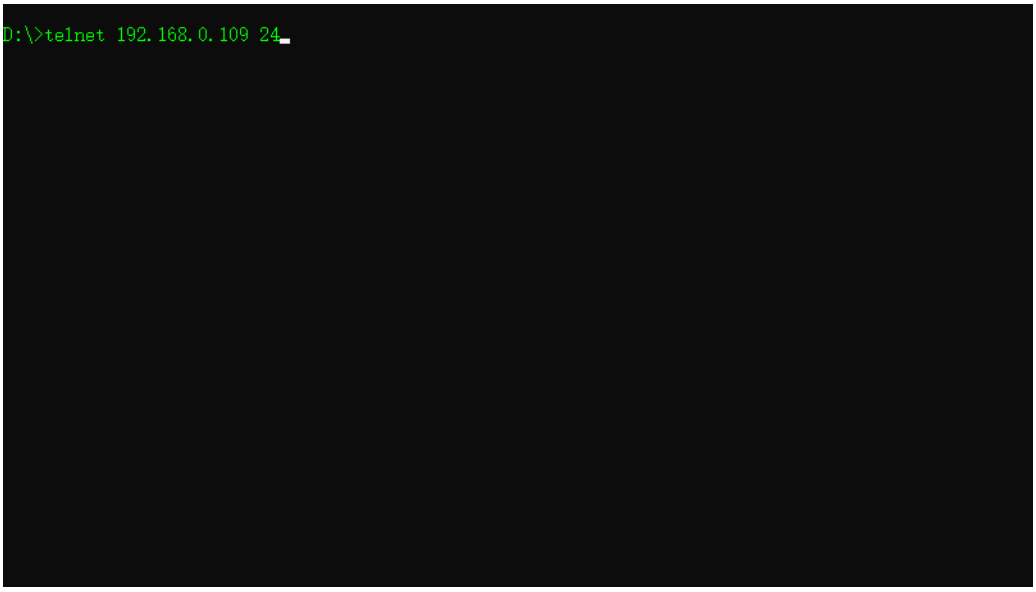


1.2 Logging In via Command Line Interface

1. Choose **Start** > **Run**.
2. In the **Run** dialog box, enter **cmd** then click **OK**.



3. Enter **telnet 192.168.0.109 24** if the device's IP address is 192.168.0.109, and then press **Enter**.



4. The device will show the command prompt as below:



Now, the device is ready to execute the CLI API command.

1.3 Access API via RS232 Interface

In addition to Telnet protocol, the API can be accessed via RS232 interface. This chapter provides the necessary information.

1.3.1 Enable API via RS232

As the default, the RS232 interface is used to control the peripherals, namely, the RS232 interface does not accept any API message. To change its purpose to provide API services, you should invoke the API gbconfig --rs232-role at first, the corresponding API request message is as follows

```
gbconfig --rs232-role api
```

After completing the above operation, please reboot the device, then you can access the API via the RS232 interface.

1.3.2 Configure RS232 Interface

Please configure the RS232 interface with the same settings as the device, the default is as below:

- ✧ Baud rate: 115200
- ✧ Data bits: 8
- ✧ Stop bits: 1
- ✧ Parity: None

1.3.3 End Text Line Correctly

As a complete text line, the two special characters: Carriage Return and Line Breaks must be added at the end of each API request message in order. Their ASCII codes are 0x0D and 0x0A respectively. The specific means to achieve this vary depending on the serial communication software you use.

1. Most terminal emulation software (such as SecureCRT) adds these two characters automatically; no extra action is needed.
2. Some serial communication software (such as UART Assistant) provides visual configuration to add extras characters at the end of each text line automatically, please adjust the configuration accordingly.
3. Some software neither automatically adds the two special characters nor provides the aforementioned visual configurations, but they may provide shortcut key to add the two characters. For an example, in DockLight, after input a text line, press Ctrl-Enter will add the two characters. Please review the manual of the serial communication software to find whether there is a shortcut key to add the two characters.

1.4 Introduction to Terminology

The terminology used in API command description is listed as follows.

Terminology	Description
Device **Product**	The VL-BYOD200 unit being controlled. Unless otherwise specified, these two words appearing in this document refer to VL-BYOD200.
Airplay Mirroring	A screen mirroring approach developed by Apple, it is supported by many Apple devices, such as Macbook, iPad and iPhone. In this document, we use Airplay as its abbreviation.
Miracast	A screen mirroring approach developed by Wi-Fi alliance, it is supported by all Android devices and Windows PC.
Google Cast **ChromeCast**	A screen mirroring approach developed by Google. It is supported by Chrome browser and ChromeBook PC.
Dongle	A wireless casting transmitter with dongle form. When works, it is plugged into the USB-C interface of the laptop.
BYOD Source	Airplay Mirroring, Miracast, Google Cast and Dongle are BYOD solutions, they are named **BYOD (video) source**.
Hardware Source **Wired Source**	The device has some hardware video input interfaces, such as HDMI and USB-C, they are named **hardware (video) source** or **wired source.**

1.5 API Syntax Overview

1.5.1 Classification

All APIs can be divided into the following categories according to their functions:

1. Device management

Manage the device name, reboot or reset device, etc...

2. Input & output management

Manage input EDID, HDCP, output resolution, etc...

3. Video switching

Many different approaches to switch video source

4. Audio & auxiliary signal switching

Route audio signal and switch USB host

5. Standby related & peripheral control

Control the device to standby and wake-up, send CEC and RS232 message

6. Network & BYOD

Manage IP address and BYOD feature

1.5.2 Format or style

API commands can be mainly divided into the following categories:

- ✧ `gbconfig`: manage the configurations of the device
- ✧ `gbcontrol`: control the device to do something
- ✧ `gblayout`: adjust the features related to screen layout

1.5.3 Terms & conventions

1. The italic string in the message format indicates that the string should be replaced by the actual parameter in the actual message, and the italic string itself should not appear in the final message body. For example:

```
DeviceName
```

Indicates the device name. The actual device name should be given in the actual message body.

2. Square brackets indicate optional parameters, for example

```
[ WinNo ]
```

indicates an optional window number parameter. Obviously, optional parameters can only appear at the end of the message.

3. The combination of curly braces and vertical bars is used to specify the possible values for the parameter, for example

```
{ usbc | hdmi }
```

Represents a parameter whose value can be usbc or hdmi.

4. The combination of square brackets and vertical bars is used to specify the possible values for optional parameters, for example

```
[ start | stop ]
```

Indicates an optional parameter whose value can be start or stop

5. Definition of common error response messages:

```
ERROR
```

If any error occurs when executing the API, the device can print specific prompt information or return this same message to indicate an execution error. The subsequent chapters will not explain it separately.

6. Some commonly used parameters

✧ Video (source) name

```
VideoName
```

Valid values are as follows:

```
usbc, hdmi
airplay1, airplay2, airplay3, airplay4
miracast1, miracast2, miracast3, miracast4
dongle1, dongle2, dongle3, dongle4
chromecast1, chromecast2, chromecast3, chromecast4
```

The four values in the above row 1 represent two wired sources, they correspond to the USB-C IN and HDMI IN interface respectively. Most APIs use this parameter *VideoName* to refer to a video source.

✧ Guide screen

```
guide
```

A special video source which means to show guide screen.

2 Command Details

2.1 Device management

2.1.1 Configure device name

【Format】

Request	`gbconfig --name DeviceName`
Response	【No text response message】
Description	Configure the device's name. As a prompt, the new name will appear on the top-right corner of the screen if the operation is successful. As the factory default, the device name has the following form: `VL-BYOD200XXXXXX` Where *XXXXXX* is the hexadecimal format of the last three digits of the MAC address. **Note:** The device name must be 1~20 characters in length, furthermore, it must include only letters, numbers, space and two special character ('_' and '-').

【Example】

To change the name to MeetingRoom:

Request:

```
gbconfig --name MeetingRoom
```

Response:

【None】

The device name will change to MeetingRoom.

2.1.2 Query device name

【Format】

Request	`gbconfig -s name`

Response	*DeviceName*
Description	Obtain the device name.

【Example】

Request:

```
gbconfig -s name
```

Response:

```
MeetingRoom
```

2.1.3 Query device information

【Format】

Request	`gbconfig -s device-info`
Response	The device prints its model and firmware version.
Description	Obtain the information about the device model and firmware version.

【Example】

Request:

```
gbconfig -s device-info
```

Response:

```
VL-BYOD200

V1.1.0
```

2.1.4 Reboot device

【Format】

Request	`gbcontrol --reboot`
Response	【No text response message】
Description	Reboot the device manually

【Example】

Request:

```
gbcontrol --reboot
```

Response:

【None】

The device starts to reboot.

2.1.5 Reset device factory default

【Format】

Request	`gbcontrol --reset-to-default [guide]`
Response	【No text response message】
Description	This command makes the device restore its factory default; to archive this, the device will reboot to enter recovery or safe mode. If the request message carries the optional parameter guide, it means to restore the default guide picture only.

【Example】

Request:

```
gbcontrol --reset-to-default
```

Response:

【None】

The device will start to restore all factory defaults.

2.1.6 Configure OSD display

【Format】

Request	`gbconfig --show-osd-item { devicename | ipaddress | wifipwd | accesscode } { y | n }`
Response	【No text response message】
Description	Set the device whether to show an OSD item. The first parameter designates a specified OSD item, the listed four values correspond device name, IP address, Wi-Fi password and access code respectively. The second parameter determines whether to show the OSD item. As the factory default, every OSD item is shown, the value of the second parameter is y. **Note:** This feature does not impact the existing feature show all OSD items when **RESET** button is pressed.

【Example】

To hide the device name OSD item:

Request:

```
gbconfig --show-osd-item devicename n
```

Response:

【none】

2.1.7 Query OSD display

【Format】

Request	`gbconfig -s show-osd-item [ devicename | ipaddress | wifipwd | accesscode ]`

Response	`{ y | n }`
	`or`
	`devicename { y | n }`
	`ipaddress { y | n }`
	`wifipwd { y | n }`
	`accesscode { y | n }`

Description	Get the configuration whether an OSD item is enabled to be shown on the screen. if the request message does not carry parameter, the device responds with the configurations of all OSD items row by row.

【Example】

Request:

```
gbconfig -s devicename
```

Response:

```
n
```

In the above example, the device name OSD item is hidden.

2.2 Input & output management

2.2.1 Configure output resolution

【Format】

Request	`gbconfig --output-resolution { Resolution | auto }`
Response	【No text response message】
Description	The device will change its output resolution as the command designates or automatically. If you assign auto as the parameter, the device will select a best resolution according to the display's EDID. The list of all available timings is below: 3840x2160P@60 3840x2160P@50 3840x2160P@30 3840x2160P@25 3840x2160P@24 1920x1080P@60 1920x1080P@50 1920x1080P@30 1920x1080P@25 1920x1080P@24 1680x1050P@60 1600x1200P@60 1440x900P@60 1366x768P@60 1280x1024P@60 1280x720P@60 1280x720P@50 1024x768P@60 800x600P@60 720x480P@60 640x480P@60 As the factory default, the output resolution is set to auto.

【Example】

To use 4K@60 resolution:

Request:

```
gbconfig --output-resolution 3840x2160P@60
```

Response:

【None】

The output resolution will change to 3840x2160P@60Hz.

2.2.2 Query output resolution

【Format】

Request	`gbconfig -s output-resolution`
Response	`{ Resolution | disconnected } [ auto ]`
Description	The device prints the resolution information. The first field is the actual resolution. The second field auto is optional which means the corresponding configuration is set auto. The term disconnected indicates that the corresponding output resolution is set auto but the output is not connected to a display.

【Example】

Request:

```
gbconfig -s output-resolution
```

Response:

```
3840x2160@60 auto
```

In the above example, the output resolution is 4K@60 which is chosen automatically.

2.2.3 Configure output HDCP

【Format】

Request	`gbconfig --hdcp-enable { y | n }`
Response	【No text response message】
Description	Configure whether the HDCP feature is enabled on the outputs. If it is enabled, HDCP 2.2 will be used on the output when the connected display support HDCP 2.2, or else HDCP 1.4 will be used. If this feature is disabled, the output content will not be protected by HDCP. As the factory default, HDCP feature is enabled, the corresponding configuration is y.

【Example】

To disable HDCP feature:

Request:

```
gbconfig --hdcp-enable n
```

Response:

【None】

The output video will not be protected by HDCP.

2.2.4 Query output HDCP

【Format】

Request	`gbconfig -s hdcp-enable`
Response	`{ y | n }`
Description	The device prints y means that the HDCP is enable, or else disabled.

【Example】

Request:

```
gbconfig -s hdcp-enable
```

Response:

```
n
```

2.2.5 Report input signal status

【Format】

Request	【No request message】
Response	`[Event] VideoSource` *`VideoName`* `{ NoSignal | { VideoTiming { YUV444 | YUV422 | RGB888 | MJPEG | H.264 | H.265 }}`
Description	When the signal status of a certain input changes, the device actively sends the response message. It has at least two arguments. The first is the name of the video source whose state has changed. The rest arguments vary on the new state: ➢ If the video source lost signal, a word NoSignal is used as the second argument. ➢ If the video source became valid, the second and third seconds provide the timing and format respectively.

【Example】

Request:

【None】

Response:

```
[Event] VideoSource hdmi 3840x2160P@30 RGB888
```

2.2.6 Query input video format

【Format】

Request	`gbconfig --input-state [` *`VideoName`* `]`
	or
	`gbconfig -s input-video [` *`VideoName`* `]`

Response	`{ invalid |` *`VideoFormat`* `}` or

	`usbc { invalid | VideoFormat }`
	`hdmi { invalid | VideoFormat }`
	`ariplay1 VideoFormat`
	`miracast3 VideoFormat`
	`...`

Description	If the request carries a parameter, it means to query the video format of the specific source, the device prints invalid to indicate the source has not valid signal, or else the device prints the *VideoFormat* whose format is:
	`{ Width x Height[p],FrameRate;HDRInfo;ColorSpace;ColorDepth }`
	The value and meaning of every field are as follows:
	✧ *Width*, *Height* The width and height of the video. In the actual message, there is no space character between the Width, height and the letter 'x', for an example, 1920x1080
	✧ P It is optional to indicate progressive video
	✧ *FrameRate* Frame rate
	✧ *HDRInfo* Whether HDR format, now the device does not support HDR, so its value is always none hdr.
	✧ *ColorSpace* Color space, available values are rgb、ycbcr444、ycbcr422
	✧ *ColorDepth* Color depth, *now only* 8bit
	An actual example of *VideoFormat*:
	`1920x1080p;60;none hdr;rgb;8bit`
	if the request message does not carry parameter, the device prints the video format information of all wired sources and all connected BYOD sources.

【Example】

Request:

```
gbconfig --input-state usbc
```

Response:

```
3840x2160p;60;none hdr;rgb;8bit
```

2.3 Video switching

2.3.1 Configure automatic switching

【Format】

Request	`gbconfig --auto-switch { y | n }`
Response	【No text response message】
Description	Configure the automatic switching feature. When it is enabled, the device supports the following functions: 1. Automatically switch wired source ✧ When a wired source is connected, starts displaying the video source automatically ✧ When a wired source is disconnected, stops displaying the video source automatically **Note**: The setting of whether automatically switch the video source only works for wired sources. Due to the nature of BYOD sources (no persistent physical interface), the device always automatically switches to BYOD sources. 2. Automatically switch layout If the multiview feature is enabled, when only display is connected to the device, the device also can: ✧ When starts displaying a video source (due to automatic switching or API triggering), if there is no idle child window (not occupied by any video source) in the current layout, try switching to a layout with more child windows. ✧ When stops displaying a video source (due to automatic switching or API triggering), if multiview layout is being used, try switching to a layout with less child windows.

	3. Roll back switching When the device is in full screen (multiview layout is unused) state, if the currently displayed video source is automatically stopped, it will automatically switch to display the last displayed video source. This feature is only for automatic switching scenarios. For API commands that have the function of "stopping displaying a certain video source", rollback switching is NOT performed. The reason is: rollback switching mechanism is designed to improve user experience in automatic switching scenarios, but it cannot accurately meet customer needs. When users use the API in a certain way, they can accurately realize their intentions through subsequent API calls, and rollback switching is unnecessary. As the factory default, automatic switching is enabled, the configuration value is set y.

【Example】

To disable automatic switching:

Request:

```
gbconfig --auto-switch n
```

Response:

【none】

The automatic switching feature is disabled.

2.3.2 Query automatic switching

【Format】

Request	`gbconfig -s auto-switch`
Response	`{ y| n }`
Description	Query the state of the automatic switching configuration

【Example】

Request:

```
gbconfig -s auto-switch
```

Response:

```
n
```

In the above example, the automatic switching is disabled.

2.3.3 Configure multiview feature

【Format】

Request	`gbconfig --multiview { y	n }`
Response	【No text response message】
Description	Configure whether the multiview feature is eanbled. When disabled, the device displays the video source with full screen or single view mode. **Note**: Even the configuration is enabled, the multiview feature can be actually activated only when only one display is connected to the device. As the factory default, multiview feature is enabled, the configuration value is set y.

【Example】

To disable multiview feature:

Request:

```
gbconfig --multiview n
```

Response:

【none】

The multiview feature is disabled.

2.3.4 Query multiview feature

【Format】

Request	`gbconfig -s multiview`
Response	`{ y| n }`
Description	Query the state of the multiview configuration

【Example】

Request:

```
gbconfig -s multiview
```

Response:

```
n
```

In the above example, the multiview feature is disabled.

2.3.5 Start displaying video source

【Format】

Request	`gblayout --start-video { VideoName | guide } [ WinNo ]`
Response	【No text response message】
Description	Instruct the device to start displaying a video source. Please refer to the chapter 1.4.3 to get the details of the parameters *VideoName* and guide. The optional parameter *WinNo* is used to specify which window of the multiview layout is used to display the video source, its value range is from 1 to the quantity of the child windows of current layout. The specific behavior is as follows: 1. If instruct to display the guide screen, the guide screen will be displayed in full screen. 2. If the specified video source is displayed already, the device continues to judge: ✧ If a multiview layout is being used, the request message carries the *WinNo* parameter and it is different from the child window currently occupied by the video source, the video source is moved to the specified child window for display (the video source originally displayed in the child window is dropped), and the child window originally occupied by the video source is left blank without displaying any video source. ✧ In other cases, no action is performed and the response message is returned directly. 3. As the third situation, the device executes the follow logic: ✧ If the multiview feature is disabled, the specified video source will be displayed in full screen. ✧ If the multiview feature is enabled and the request message does not carry the *WinNo* parameter, the device will start displaying the video source according to the multiview switching logic. Furthermore, if the automatic switching is enabled, the device may switch layout when necessary. ✧ If the multiview feature is enabled and the request message carries the *WinNo* parameter, the device will start displaying the video source in the designated child window. The video source originally displayed in the child window will be dropped.

【Example】

Request:

```
gblayout --start-video usbc 2
```

Response:

【none】

The device starts displaying usbc (correspond to the first USB-C IN input interface) in the second child window of the current multiview layout.

2.3.6 Stop displaying video source

【Format】

Request	`gblayout --stop-video VideoName`

Response	【No text response message】
Description	Instruct the device to start displaying a video source. The specific behavior is as follows: ✧ If the video source is not actually displayed, no action is performed. ✧ If the multiview feature is disabled, the device stops displaying the video source and switch to the guide screen. ✧ If the multiview feature is enabled. the device stops displaying the video source according to the multiview logic (if the automatic switching is enabled, the device may switch layout when necessary). If the stopped video source is the only video source currently displayed, the device switch to the guide screen after stopping displaying.

【Example】

Request:

```
gblayout --stop-video hdmi
```

Response:

【none】

The device stops displaying hdmi.

2.3.7 Display video source in full screen

【Format】

Request	`gbconfig --video-source { VideoName | guide }`
Response	【No text response message】
Description	Instruct the device to display the video source in full screen, if the parameter value is guide, the device display the guide screen.

【Example】

Request:

```
gbconfig --video-source  hdmi
```

Response:

【none】

The device displays hdmi in full screen.

2.3.8 Query video output status

【Format】

Request	`gbconfig -s video-source`
Response	`{ guide | disconnected | { { VideoName1 | null } { VideoName2 | null }... } }`
Description	Query which video sources are displayed, there are several situations: ✧ disconnected means the output is not connected to a display. ✧ guide means the output does not display any video source, guide screen is displayed on it. ✧ If the multiview layout is being used, the device returns all video sources information displayed on the output in descending order of the child windows. ✧ null indicates the corresponding child windows is idle (no video source is displayed in it). The null at the end of the response message can be omitted.

【Example】

Request:

```
gbconfig -s video-source
```

Response:

```
usbc null hdmi miracast1
```

In the above example, the child windows 1, 3, 4 are used to display usbc, hdmi and miracast respectively, the child window 2 is idle.

2.3.9 Query layout list

【Format】

Request	`gblayout --list`

Response	`Layout #  Name`
	`LayoutID1` `[ LayoutName1 ]`
	`LayoutID2` `[ LayoutName2 ]`
	`LayoutID3` `[ LayoutName3 ]`
	`...`

Description	Query the layout list. The device returns the ID and name of all layouts

【Example】

Request:

```
gblayout --list
```

Response:

```
Layout #        Name:
0x100           layout0
0x101           layout1
```

2.3.10 Query layout

【Format】

Request	`gblayout --get [ LayoutID | LayoutName ]`

Response	*`LayoutNo`* `[ LayoutName ]`
	or
	`Layout #:` *`LayoutNo`* `Name:` *`LayoutName`* *`WinNum`* `windows`
	`1*`*`Win1X Win1Y Win1W Win1H`*
	`2` *`Win2X Win2Y Win2W Win2H`*
	`3` *`Win3X Win3Y Win3W Win3H`*
	`...`

Description	This command has 2 uses, described as follows: 1. When the request does not carry parameter, the device returns the ID and name (if the layout has it) of the layout being used currently. 2. When the request specifies an ID or name of a layout, the device returns the position and size of all child windows of the specified layout. For the position and size data, they use a virtual coordinate where the screen resolution is always 16000x9000 so make them independent of the actual screen resolution. The asterisk '*' indicates the main window.

【Example1】

Request:

```
gblayout --get
```

Response:

```
0x101 SidebySide_DualView
```

In the above example, the ID of the current layout is 0x101 and the name is SidebySide_DualView.

【Example2】

Request:

```
gblayout --get layout1
```

Response:

```
Layout #: 0x101 Name:layout1    2 windows

 1        0        2250     8000     4500

2        8000     2250     8000     4500
```

In the above example, the queried layout has two child windows.

2.3.11 Switch layout

【Format】

Request	gblayout --set { *LayoutID* | *LayoutName* }
Response	【No text response message】
Description	Switch the layout (of the child windows) used by multiview mode. described as follows: ✧ *LayoutID or LayoutName* is used to specified a layout. ✧ This API does not work when the multiview feature is disabled. ✧ Multiview feature can actually work only when only one display is connected to the device, so the request message needs not to specify the output. As the boot default, single full screen layout is used, the corresponding *LayoutID* is 0x100

【Example】

Request:

```
gblayout --set 0x101
```

Response:

【none】

The device switch to the side-by-side dual view layout.

2.4 Audio & auxiliary signal switching

2.4.1 Configure output audio source

【Format】

Request	`gbconfig --audio-source { VideoName | auto }`
Response	【No text response message】
Description	Switch the output audio source, the specific behavior of the device depends on the specific parameters: 1. If the parameter value is `auto`, it means automatic switching, always playing the audio of the most recently connected video source. If the input is disconnected, it will fall back and play the audio of the last connected video source. 2. If the parameter designates a wired source, as long as the corresponding video source is connected, the device will play the audio of this video source always. If the video source is disconnected, the device will switch the audio source according to the `auto` logic mentioned in the previous item. 3. If the parameter designates a BYOD source, the behavior is as follows: ✧ If the specified BYOD source is invalid (disconnected), nothing happens. ✧ If the specified BYOD source is valid, the device will play the audio of this BYOD source, but does not modify the saved configuration. ✧ When the specified BYOD source becomes invalid, the device will load the saved configuration and behave

	accordingly. In short, playing a BYOD source's audio is a temporary and one-time change. As the factory default, the configuration value is set `auto`.

【Example】

To play the audio of USB-C IN:

Request:

```
gbconfig --audio-source usbc
```

Response:

【none】

The device starts playing the audio of usbc1 once it is connected.

2.4.2 Query output audio source

【Format】

Request	`gbconfig -s audio-source`
Response	`{ VideoName | auto } { VideoName | null }`
Description	Query the audio source being played currently, the details are as follows: ✧ In the returned message, the first parameter is the saved configuration. When switch to a BYOD source, the device does not modify the save configuration, so the value of this parameter can only be a wired source or `auto`. ✧ The second parameter indicates the actually played audio source. If the device is displaying guide screen and no audio source is played, it is represented by `null`.

【Example】

Request:

```
gbconfig -s audio-source
```

Response:

```
auto miracast1
```

In the above example, the configuration is auto, the device is playing the audio of the source miracast1.

2.4.3 Configure USB host

【Format】

Request	`gbconfig --usb-host { USBHost | auto }`
Response	【No text response message】
Description	Configure the output interface of the USB switcher, the details are as follows: 1. The parameter *USBHost* is used to designate a certain USB host interface. The value and meaning are as follows: ✧ `usbcin` The USB channel of the USB-C IN port ✧ `usbhost` The USB HOST interface ✧ `wireless` Connect the USB switcher to the main SoC for wireless conference and other purpose. 2. The parameter value `auto` means automatic switching, the USB host interface is always connected to the most recently connected USB host interface and following the fall-back principle. As the factory default, the configuration value is set `auto`.

【Example】

To play the audio of USB-C IN:

Request:

```
gbconfig --usb-host usbcin
```

Response:

【none】

The output of the USB switcher is connected to the USB channel of the USB-C IN port.

2.4.4 Query USB host

【Format】

Request	`gbconfig -s usb-host`
Response	*USBHost* [auto]
Description	Query the current USB host interface, the optional auto means the USB switcher is configured as automatic switching.

【Example】

Request:

```
gbconfig -s usb-host
```

Response:

```
usbcin auto
```

In the above example, the configuration is auto and the current USB host is usbcin (USB-C IN).

2.5 Standby related & peripheral control

2.5.1 Configure automatic standby

【Format】

Request	`gbconfig --auto-standby { y | n }`
Response	【No text response message】
Description	Configure whether the automatic standby feature is enabled. As the factory default, automatic standby feature is enabled, the configuration value is set to y.

【Example】

To disable automatic standby feature:

Request:

```
gbconfig --auto-standby n
```

Response:

【none】

The automatic standby feature is disabled.

2.5.2 Query automatic standby

【Format】

Request	`gbconfig -s auto-standby`
Response	{ y| n }
Description	Query the state of the automatic standby configuration

【Example】

Request:

```
gbconfig -s auto-standby
```

Response:

```
n
```

In the above example, the automatic standby feature is disabled.

2.5.3 Configure automatic standby timeout

【Format】

Request	`gbconfig --auto-standby-time` *`TimeOut`*
Response	【No text response message】
Description	The parameter *`Timeout`* indicates how long the device will be idle (namely, displaying guide screen) before entering standby mode, its unit is second. The value `0` means entering standby mode immediately when the device starts being idle. Apparently, this configuration takes effect only when the automatic standby feature is enabled. The value range is from `0` to `3600`. As the factory default, automatic standby timeout is 2 minutes, the configuration value is set `120`.

【Example】

Request:

```
gbconfig --auto-standby-time 360
```

Response:

【none】

The automatic standby timeout is set 6 minutes.

2.5.4 Query automatic standby timeout

【Format】

Request	`gbconfig -s auto-standby-time`
Response	`TimeOut`
Description	Query the value of the automatic standby timeout

【Example】

Request:

```
gbconfig -s auto-standby-time
```

Response:

```
360
```

In the above example, the automatic standby timeout is 6 minutes.

2.5.5 Configure CEC command

【Format】

Request	`gbconfig --cec-cmd { on | off }` *`CECCode`*
Response	【No text response message】
Description	Configure the stored CEC command used to power on/off the display. The first parameter indicates which command will be changed. `on` and `off` correspond to power on and off respectively. The second parameter *`CECCode`* is the actual CEC message string with hexadecimal format, no space between the

	adjacent bytes. As the factory default, the power on message is `4004` and power off message is `ff36`.

【Example】

Request:

```
gbconfig --cec-cmd on 4004
```

Response:

【none】

The power on CEC message is set `4004`.

2.5.6 Query CEC command

【Format】

Request	`gbconfig -s cec-cmd [ on | off ]`
Response	`CECCode` or `on` *`CECCode1`* `off` *`CECCode2`*
Description	Query a CEC command configuration. If the request message request has no parameter, the device returns all two CEC messages in two rows.

【Example】

Request:

```
gbconfig -s cec-cmd on
```

Response:

```
4004
```

In the above example, the query result of CEC power on message is 4004.

2.5.7 Configure RS232 work mode

【Format】

Request	`gbconfig --rs232-role { api | com | passthrough }`
Response	【No text response message】
Description	The RS232 port supports different work modes, the value and meanings of the parameter are as follows: ✧ `api` Receive the API commands sent by the external controller ✧ `com` Conventional communication, control the peripherals such as display, participate in standby related power on/off display operations. ✧ passthrough Connect the RS232 port to the RS232 channel of the HDBT port so that the RS232 data is transparently transmitted between the local RS232 port and remote HDBT RX. As the factory default, the RS232 port is used to control the peripherals the configuration value is `com`.

【Example】

Request:

```
gbconfig --rs232-role api
```

Response:

【none】

The RS232 port is configured for API commands.

2.5.8 Query RS232 work mode

【Format】

Request	`gbconfig -s rs232-role`
Response	`{ api | com | passthrough }`
Description	Query the work mode of the RS232 port.

【Example】

Request:

```
gbconfig -s rs232-role
```

Response:

```
api
```

In the above example, the query result of RS232 port work mode is api.

2.5.9 Configure RS232 settings

【Format】

Request	`gbconfig --rs232-param` *`RS232Settings`*
Response	【No text response message】
Description	Modify the RS232 port settings, the parameter *`RS232Settings`* has the format link `9600-8n1`. The settings are applicable to all scenarios using RS232 port, unless the operation specifies the RS232 settings directly. As the factory default, the configuration is `115200-8n1`.

【Example】

Request:

```
gbconfig --rs232-param 9600-8n1
```

Response:

【none】

The RS232 port settings are set `9600-8n1`.

2.5.10 Query RS232 setting

【Format】

Request	`gbconfig -s rs232-param`
Response	*`RS232Settings`*
Description	Query the settings of the RS232 port.

【Example】

Request:

```
gbconfig -s rs232-param
```

Response:

```
115200-8n1
```

In the above example, the query result of RS232 port settings is 115200-8n1.

2.5.11 Configure RS232 baud rate

【Format】

Request	`gbconfig --rs232-baudrate` *`BaudRate`*
Response	【No text response message】
Description	Configure the RS232 port baud rate separately, the available values are `9600`, `19200`, `38400`, `57600`, `115200`. As the factory default, the baud rate is `115200`.

【Example】

Request:

```
gbconfig --rs232-baudrate 9600
```

Response:

【none】

The RS232 port baud rate is set `9600`.

2.5.12 Query RS232 baud rate

【Format】

Request	`gbconfig -s rs232-baudrate`
Response	*`BaudRate`*
Description	Query the baud rate of the RS232 port.

【Example】

Request:

```
gbconfig -s rs232-baudrate
```

Response:

```
115200
```

In the above example, the query result of RS232 port baud rate is 115200.

2.5.13 Configure RS232 command

【Format】

Request	`gbconfig --rs232-cmd { on | off |` *`CmdName`* `} [ { hex | str } ` *`CmdStr`* `]`
Response	【No text response message】
Description	Configure the message string sent to the peripherals through the RS232 port. The values and meanings of the parameters are as follows: 1. The first parameter indicates which command to be configured. Except `on` and `off` (used to power on / off the display), the caller can define new command such as `volumeup`, `volumedown`, but the command name can't contain space. If the specified command has not been set before, it will be added, otherwise the existing command will be modified. 2. The second parameter indicates the format of the format of the subsequent third parameter *`CmdStr`*. `hex` means hexadecimal and `str` means printable string. 3. The third parameter is the actual command string. ✧ For hexadecimal format, there is no space between the adjacent bytes. For example: `112233445566`

	✧ For string format, the spaces among the words will be regarded as the part of the command string. For example, `Power on` 4. The second and third parameters are optional, if they are omitted, it means to delete a command defined already. As the factory default, no RS232 message is defined.

【Example1】

Request:

```
gbconfig --rs232-cmd on str Poweron
```

Response:

【none】

The power on RS232 message is set `Power on` with string format.

【Example2】

Request:

```
gbconfig --rs232-cmd on
```

Response:

【none】

The power on RS232 message is deleted.

2.5.14 Query RS232 command

【Format】

Request	`gbconfig -s rs232-cmd [ on | off | CmdName ]`

Response	`{ hex | str } CmdStr1`
	or
	`On { hex | str } CmdStr1`
	`off { hex | str } CmdStr2`
	`CmdName3 { hex | str } CmdStr3`
	`...`

Description	Query a RS232 command configuration. If the command request has no parameter, the device returns all defined RS232 messages row by row.

【Example】

Request:

```
gbconfig -s rs232-cmd on
```

Response:

```
str Poweron
```

In the above example, the query result of RS232 power on message is Power on with string format.

2.5.15 Configure power on/off display via RS232

【Format】

Request	`gbconfig --sinkpower-mode { cec | both }`
Response	【No text response message】

Description	When the device enters or exits standby mode, in additions to powering on/off the display through CEC messages, the device also can send the corresponding commands through RS232 port. This API is used to configure whether to send RS232 command when powering on/off the display. **Note**: The actual activation of sending power on/off command through RS232 port requires three conditions to be met: ✧ The RS232 port work mode is `com`, namely, control peripherals ✧ The RS232 `on` or `off` message is defined ✧ This configuration is `both` As the factory default, this configuration is set to `cec`, when the device enters or exits standby mode, the device sends power on/off command through CEC messages only.

【Example】

Request:

```
gbconfig --sinkpower-mode both
```

Response:

【none】

Enable the feature of power on/off display via RS23 port.

2.5.16 Query power on/off display via RS232

【Format】

Request	`gbconfig -s sinkpower-mode`
Response	`{ cec | both }`
Description	Query the configuration.

【Example】

Request:

```
gbconfig -s sinkpower-mode
```

Response:

```
cec
```

In the above example, the query result of is cec.

2.5.17 Enter or exit standby mode

【Format】

Request	`gbcontrol --sinkpower { on | off }`
	or
	`gbconfig --sinkpower { on | off }`

Response	【No text response message】
Description	Manually instruct the device to enters or exits standby mode. `on` means power on (exiting standby mode) and `off` means power off (entering standby mode).

【Example】

Request:

```
gbconfig --sinkpower off
```

Response:

【none】

The device enters standby mode.

2.5.18 Query standby mode

【Format】

Request	`gbconfig -s sinkpower`
Response	`{ on | off }`
Description	Query whether the device is in standby mode.

【Example】

Request:

```
gbconfig -s sinkpower
```

Response:

```
off
```

In the above example, the device is in standby mode.

2.5.19 Report entering or exiting standby mode

【Format】

Request	【No request message】
Response	`[Event] WorkMode { Normal | Sleep }`
Description	When the device enters or exits standby mode, it actively sends the response message. The parameter is the new state.

【Example】

Request:

【None】

Response:

```
[Event] WorkMode Sleep
```

In the above example, the device enters standby mode.

2.5.20 Send CEC or/and RS232 command

【Format】

Request	`gbcontrol --send-cmd { on | off | CmdName } [ rs232 | cec | both ]`
Response	【No text response message】
Description	Instruct the device to send CEC or/and RS232 command. The first parameter is the name of sent command, the second optional parameter indicates the sending method. If the sending method is not specified, the configuration of `gbconfig --sinkpower-mode` is used. If the RS232 work mode is not `com` or a specified RS232 command is not defined, the device sends CEC command only.

【Example】

Request:

```
gbcontrol --send-cmd on both
```

Response:

【none】

The device sends power on command through both CEC and RS232 interfaces.

2.5.21 Send RS232 data

【Format】

Request	`gbcontrol --serial [ -b RS232Settings ] [ -r { on | off }] [ -h { on | off }] [ -t timeout ] CmdStr`
Response	`Response Size [ xx xx xx xx… ]`
Description	Send some data through the RS232 port and receive the response data. The details of the parameters are below: 1. `-b` parameter is used to set the RS232 port settings which contains baud rate, data bits, parity and stop bits. By default, `9600-8n1` is used. 2. `-r { on | off }` is used to set whether to add a carriage return at the end of the data . The default is `off`. 3. `-h { on | off }` is used to indicate whether to fifth parameter *`CmdStr`* is in hexadecimal format. The default is `off`. So the *`CmdStr`* will be sent by its printable ASCII format. If the value is `on`, *`CmdStr`* will be interpret as hexadecimal characters. 4. `-t` *`timeout`* is used to designate the timeout in which this command will return. When the command returns, all data received from the RS232 port will be print as hexadecimal format. The unit is mini-second and its default is 0, it means that the command returns immediately and no response data will be received. 5. *`CmdStr`* is the data to be sent. 6. *`Size`* is the size of the response data received before the command returns. **Note**: This is a low-level command for testing and debugging mainly. The device executes this command uncontionally, regardless of any configuration. The caller must ensure that the command is called at the appropriate time.

【Example1】

Request:

```
gbcontrol --serial Hello
```

Response:

```
Response 0
```

The device sends printable string Hello through the RS232 port and returns immediately.

【Example2】

Request:

```
gbcontrol --serial -b 115200-8n1 -h on -t 500 67 65 74 20 73 74 61 74 65
```

Response:

```
Response 4 67 6F 6F 64
```

Configure the RS232 port settings as 115200-8n1, send the hexadecimal string `67 65 74 20 73 74 61 74 65`, then keep receiving the response data for 500ms. During the waiting period, the peripheral returns four bytes `67 6F 6F 64`.

2.6 Network & BYOD

2.6.1 Configure network mode

【Format】

Request	`gbconfig --network-isolation { y | n }`
Response	【No text response message】
Description	Configure whether to enable the network isolation mode. The device has five Gigabit Ethernet interfaces: three physical RJ45 network interfaces on the rear panel, each of the two

	USB-C input ports has a GbE network adapter. The device can use these Ethernet interfaces in two different modes: ✧ Transparent All Ethernet interfaces are interconnected together and their actual functions are also equal. To use this mode with this API, the parameter value should be `n`. ✧ Isolated Only the Gigabit Ethernet interface marked **CONTROL** can be used to control the device (via Web UI or API). The rest can be used for the conventional service features, such as BYOD or network accessing. To use this mode with this API, the parameter value should be `y`. To get the details, please refer to the PRD or user manual. As the factory default, the configuration value is `n`.

【Example】

Request:

```
gbconfig --network-isolation y
```

Response:

【none】

The device network switches to isolated mode.

2.6.2 Query network mode

【Format】

Request	`gbconfig -s network-isolation`
Response	`{ y | n }`
Description	Query the current network mode.

【Example】

Request:

```
gbconfig -s network-isolation
```

Response:

```
n
```

In the above example, the query result is n, namely, transparent mode.

2.6.3 Configure IP address

【Format】

Request	`gbconfig --ip-address { control | service } { dhcp | static` *`IPAddress NetMask`* `[` *`Gateway`* `[` *`DNSServer`* `] ] }`
Response	【No text response message】
Description	Configure the IP address. Here are the details: 1. In isolated network mode, the device has two sets of IP address, one for control and one for conventional service, the first parameter specifies which IP address to be set. 2. The second parameter gives the IP mode. Now DHCP and static are supported. 3. For static IP address, a series additional parameters are used to provide IP address, subnet mask, optional gateway and DNS server. As the factory default, DHCP is used.

【Example】

Request:

```
gbconfig --ip-address service static 192.168.1.88 255.255.255.0 192.168.1.1
```

Response:

【none】

The service IP address is configured as static, the configuration takes effect in isolated network mode.

2.6.4 Query IP address

【Format】

Request	`gbconfig -s ip-address [ control | service ]`

Response	`{ dhcp | static} IPAddress NetMask [ Gateway [ DNSServer ] ]`
	or
	`control { dhcp | static} IPAddress NetMask [ Gateway [ DNSServer ] ]`
	`service { dhcp | static} IPAddress NetMask [ Gateway [ DNSServer ] ]`

Description	Query the IP address. If the request message does not carry parameter, the device returns the information of all two IP address in two rows.

【Example】

Request:

```
gbconfig -s ip-address control
```

Response:

```
dhcp 169.254.2.155 255.255.0.0
```

In the above example, the control IP address is DHCP mode, the IP address is 169.254.2.155 and the subnet mask is 255.255.0.0.

2.6.5 Configure USB-C network adapter

【Format】

Request	`gbconfig --nic-enable { y | n }`
Response	【No text response message】
Description	The USB-C input port has a GbE network adapter which can be disabled to improve network security. This API configure whether to enable the aforementioned network adapter. As the factory default, USB-C network adapters are enabled, the configuration value is `y`.

【Example】

Request:

```
gbconfig --nic-enable n
```

Response:

【none】

The USB-C network adapter is disabled.

2.6.6 Query USB-C network adapter

【Format】

Request	`gbconfig -s nic-enable`
Response	`{ y | n }`
Description	Query the state whether the USB-C network adapter is enabled.

【Example】

Request:

```
gbconfig -s nic-enable
```

Response:

```
n
```

In the above example, the USB-C network adapter is disabled.

2.6.7 Configure Wi-Fi mode

【Format】

Request	`gbconfig --wifi-mode { 2 | 5 } { WifiChannel | auto }`
Response	【No text response message】
Description	Configure the radio band and channel used by the Wi-Fi module. The first parameter designates the band, `2` means 2.4G and `5` means 5G. The second parameter *`WifiChannel`* designates the channel, its values range varies on the band: Band: 2.4G — Values Range of Channel: `1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11` Band: 5G — Values Range of Channel: `36, 40, 44, 48, 149, 153, 157, 161` The value `auto` means the device chooses a best channel automatically. As the factory default, the device uses 5G band and chooses a channel automatically.

【Example】

Request:

```
gbconfig --wifi-mode 2 1
```

Response:

【none】

Configure 2.4G band and channel 1.

2.6.8 Query Wi-Fi mode

【Format】

Request	`gbconfig -s wifi-mode`
Response	`{ 2 | 5 } WifiChannel [ auto ]`
Description	Query the Wi-Fi mode, the optional `auto` word means the channel is chosen by the device automatically.

【Example】

Request:

```
gbconfig -s wifi-mode
```

Response:

```
5 149 auto
```

In the above example, the Wi-Fi module works on 5 GHz band and the 149 channel is chosen automatically.

2.6.9 Configure Wi-Fi hotspot

【Format】

Request	`gbconfig --softap-enable { y | n }`
Response	【No text response message】
Description	Configure whether the hotsport (soft AP) is enabled.

	Note: The soft AP is designed for the BYOD applications mainly. Its performance limited, if your purpose is to access the LAN through Wi-Fi connection, we suggest to deploy a standalone Wi-Fi AP to archive better experience. As the factory default, the soft AP is enabled, the configuration is y.

【Example】

Request:

```
gbconfig --softap-enable n
```

Response:

【none】

The soft AP is disabled.

2.6.10 Query Wi-Fi hotspot

【Format】

Request	`gbconfig -s softap-enable`
Response	`{ y | n }`
Description	Query whether the soft AP is enabled

【Example】

Request:

```
gbconfig -s softap-enable
```

Response:

```
y
```

In the above example, the soft AP is enabled.

2.6.11 Configure Wi-Fi password

【Format】

Request	`gbconfig --softap-password` *`WifiPassword`*
Response	【No text response message】
Description	Configures the password of the soft AP. The password must be 8~20 characters in length, furthermore, it must include only letters, numbers and two special character ('_' and '-'). As the factory default, the soft AP password is `12345678`.

【Example】

Request:

```
gbconfig --softap-password 88888888
```

Response:

【none】

The soft AP password is set `88888888`.

2.6.12 Query Wi-Fi password

【Format】

Request	`gbconfig -s softap-password`
Response	*`WifiPassword`*

Description	Query the Wi-Fi password.

【Example】

Request:

```
gbconfig -s softap-password
```

Response:

```
12345678
```

In the above example, the query result of the Wi-Fi password is 12345678.

2.6.13 Configure soft router

【Format】

Request	`gbconfig --softap-router { y | n }`
Response	【No text response message】
Description	Configure whether to enable the soft router. Basing on the soft AP, the device can launch a built in NAT module with which a device connected to the soft AP can access the LAN/WAN through the device's wired network interface. To make soft router work, please make sure: ✧ Soft AP feature is enabled because it is the basis of the soft router. ✧ The DNS server is configured correctly. If static IP is used, please specify a valid DNS server. As the factory default, the soft router is enabled, the configuration is `y`.

【Example】

Request:

```
gbconfig --softap-router n
```

Response:

【none】

The soft router is disabled.

2.6.14 Query soft router

【Format】

Request	`gbconfig -s softap-router`
Response	`{ y | n }`
Description	Query whether the soft router is enabled

【Example】

Request:

```
gbconfig -s softap-router
```

Response:

```
y
```

In the above example, the soft router is enabled.

2.6.15 Configure BYOD

【Format】

Request	`gbconfig --byod-enable { y | n }`

Response	【No text response message】
Description	Configure whether to enable the BYOD (sink), currently this configuration works for Airplay Mirroring, Miracast and Google Cast. The Dongle casting is not impact by this configuration, always enabled. As the factory default, the BYOD is enabled, the configuration is `y`.

【Example】

Request:

```
gbconfig --byod-enable n
```

Response:

【none】

The BYOD feature is disabled.

2.6.16 Query BYOD

【Format】

Request	`gbconfig -s byod-enable`
Response	`{ y | n }`
Description	Query whether the BYOD feature is enabled

【Example】

Request:

```
gbconfig -s byod-enable
```

Response:

```
y
```

In the above example, the BYOD feature is enabled.

2.6.17 Configure BYOD access code

【Format】

Request	`gbconfig --access-code { AccessCode | auto | none }`
Response	【No text response message】
Description	Configure BYOD access code (PIN). The parameter *AccessCode* consists of 4 digits. `auto` means the device automatically generates the access code and changes it dynamically. `none` means there is no access code. Now access code works for Airplay Mirroring and Miracast only. It has no use for Google Cast and Dongle. Note: • Many devices can't support access code well, To ensure the best screen casting experience, we recommend setting an access code only when absolutely necessary. • As the factory default, no access code is configured, the configuration is `none`.

【Example】

Request:

```
gbconfig --access-code 1234
```

Response:

【none】

The BYOD access code is set `1234`.

2.6.18 Query BYOD access code

【Format】

Request	`gbconfig -s access-code`
Response	`{ AccessCode [ auto ] | none }`
Description	Query the BYOD access code. If the word `auto` follows the *`AccessCode`*, it meas the access code is generated by the device dynamically. `none` means no access code.

【Example】

Request:

```
gbconfig -s access-code
```

Response:

```
8947 auto
```

In the above example, the access code is 8947 generated by the device dynamically.

2.6.19 Configure SSID Broadcast

【Format】

Request	`gbconfig --broadcast-ssid { y | n }`
Response	【No text response message】
Description	Configure whether the Wi-Fi AP module sends the SSID broadcast actively. When this feature is disabled, it is commonly known as a hidden SSID or hidden Wi-Fi, in this situation, the Wi-Fi hotsport will not normally be discovered by the Wi-Fi scanning mechanisms of the mobile or laptop. However, a user can still input the SSID name manually to connect to the Wi-Fi AP normally. As the factory default, the feature is enabled, the configuration is `y`. **Note:** Hidden SSID is not mean it cannot be discovered absolutely, it can still be discovered by a certain tools or approaches. However, it is a little more secure than those SSIDs that are not hidden.

【Example】

Request:

```
gbconfig --broadcast-ssid n
```

Response:

【none】

The SSID becomes hidden.

2.6.20 Query SSID Broadcast

【Format】

Request	`gbconfig -s broadcast-ssid`
Response	`{ y | n }`
Description	Query whether the device sends SSID broadcast.

【Example】

Request:

```
gbconfig -s broadcast-ssid
```

Response:

```
n
```

In the above example, the SSID broadcast is disabled.

3 Appendix

[To be added]

4 Revision History

Version	Prepared by	Verified by	Date	Description
V1.0	Hornet		2024.09.05	Original version.
V1.1	Hornet		2024.09.21	• Updated the screenshot to reflect changes to the Telnet server welcome message. • Added support for `LayoutName` as a parameter for the `gblayout --get` and `gblayout --set` commands.
V1.2	Hornet		2026.08.10	• Updated the `gbconfig -s video-source` command to indicate that trailing `null` values in the response message can be omitted. • Added Chapter 1.3, which describes how to access the API via RS232.