Multiview APIA black background with a black square

AI-generated content may be incorrect.

This section provides a complete reference for the Netvio MV-H2-0401-10 Multiview API commands.

**Serial Connection and Command Format**

The multiviewer can be controlled via RS-232 using ASCII commands. Connect a PC to the device’s RS-232 port (3-pin Phoenix connector) using an appropriate cable. Use serial settings **115200 baud, 8 data bits, 1 stop bit, no parity** (8-N-1).

Each command must end with a ! character. In the command syntax below, placeholders like x and y represent variables (e.g. numeric values). The prefix s denotes a **set** command (to configure a setting) and r denotes a **read** command (to query the current status).

Below is a comprehensive list of available commands, grouped by functional section. For each command, the **Description** explains its function, **Variables** detail the parameters and their possible values, **Example** provides a usage example, **Feedback** shows the device’s response, and **Default** gives the default value or state (when applicable).

## System Settings

### power z!

**Description:** Powers the device on or off.

**Variables:**

* z (0 or 1) – Power state to set: 0 turns the power **off**, and 1 turns the power **on**.

**Example:** power 1!

**Feedback:** Returns power on (if the device is now on) or power off (if turned off).

**Default:** *(N/A – device retains last power state or powers on upon startup.)*

### r power!

**Description:** Queries the current power state of the device.

**Example:** r power!

**Feedback:** Returns the current state, either power on or power off.

**Default:** *(N/A)*

### reboot!

**Description:** Reboots the device. This command will restart the system.

**Example:** reboot!

**Feedback:** The device replies with a reboot confirmation (e.g. Reboot...) and then restarts.

**Default:** *(N/A)*

### help!

**Description:** Lists all available RS-232 commands. This is useful for retrieving a quick reference of command syntax from the device.

**Example:** help!

**Feedback:** The device will output a list of all supported commands and their brief usage.

**Default:** *(N/A)*

### r type!

**Description:** Reports the device model information.

**Example:** r type!

**Feedback:** Returns the model identifier, for example: HDC-SWB41MVK (actual model name may vary).

**Default:** *(N/A)*

### r fw version!

**Description:** Retrieves the firmware version(s) of the device.

**Example:** r fw version!

**Feedback:** Returns the firmware version details, for example:

MCU FW version x.xx.xx

SCALER FW version x.xx.xx

SUB\_MCU FW version x.xx.xx

KVM\_MCU FW version x.xx.xx

(each component’s firmware version is listed).

**Default:** *(N/A)*

### reset!

**Description:** Resets the device to factory default settings. Use this command to restore all configuration to defaults.

**Example:** reset!

**Feedback:** The device returns a confirmation (e.g. Reset to factory defaults) and then resets its settings.

**Default:** *(N/A – this command itself invokes loading defaults.)*

## Output Settings

### s output res x!

**Description:** Sets the HDMI output resolution of the multiviewer.

**Variables:**

* x (1–15) – Output resolution selection:
  + **1** – 4096×2160p60
  + **2** – 4096×2160p50
  + **3** – 3840×2160p60
  + **4** – 3840×2160p50
  + **5** – 3840×2160p30
  + **6** – 3840×2160p25
  + **7** – 1920×1200p60 (Reduced Blanking)
  + **8** – 1920×1080p60
  + **9** – 1920×1080p50
  + **10** – 1360×768p60
  + **11** – 1280×800p60
  + **12** – 1280×720p60
  + **13** – 1280×720p50
  + **14** – 1024×768p60
  + **15** – AUTO (automatically select the optimal resolution)

**Example:** s output res 3!

**Feedback:** The device confirms the new setting, e.g. out resolution: 3840x2160p60.

**Default:** **3840×2160p60** is the default output resolution.

### r output res!

**Description:** Retrieves the current HDMI output resolution setting.

**Example:** r output res!

**Feedback:** Returns the active output resolution, e.g. out resolution: 3840x2160p60.

**Default:** *(See default for* ***s output res x!*** *above.)*

### s output hdcp x!

**Description:** Sets the HDCP compliance level for the HDMI output. This controls which HDCP version the output will use or if a custom mode is used.

**Variables:**

* x (1–3) – HDCP mode selection:
  + **1** – HDCP 1.4
  + **2** – HDCP 2.2
  + **3** – USER Mode (use custom HDCP setting)

**Example:** s output hdcp 2! *(sets output to HDCP 2.2 mode)*

**Feedback:** Confirms the new HDCP setting, e.g. output HDCP: HDCP 2.2.

**Default:** **HDCP 1.4** is the default HDCP setting for the output.

### r output hdcp!

**Description:** Queries the current HDCP setting of the HDMI output.

**Example:** r output hdcp!

**Feedback:** Returns the HDCP status of the output, e.g. output HDCP: HDCP 1.4.

**Default:** *(See default for* ***s output hdcp x!*** *above.)*

### s output vka x!

**Description:** Sets the “video keep-alive” pattern on the HDMI output. This pattern is displayed when no active input is present (to maintain an active signal).

**Variables:**

* x (1 or 2) – Output blank screen pattern:
  + **1** – Black screen (blank output with black image)
  + **2** – Blue screen (solid blue image)

**Example:** s output vka 1!

**Feedback:** Confirms the pattern set, e.g. output VKA pattern: black screen.

**Default:** **Black screen** is the default keep-alive pattern.

### r output vka!

**Description:** Reads the current “video keep-alive” pattern setting.

**Example:** r output vka!

**Feedback:** Returns the pattern currently configured, e.g. output VKA pattern: black screen.

**Default:** *(See default for* ***s output vka x!*** *above.)*

### s output itc x!

**Description:** Sets the output video mode between standard video timing or PC timing. (ITC stands for InfoFrame Type Code, which often relates to video vs PC mode color space handling.)

**Variables:**

* x (1 or 2) – Output video mode:
  + **1** – Video mode (intended for standard video signals)
  + **2** – PC mode (intended for PC graphics signals)

**Example:** s output itc 1! *(sets output to Video mode)*

**Feedback:** Confirms the mode, e.g. output ITC: video mode.

**Default:** **Video mode** (1) is the default output mode.

### r output itc!

**Description:** Queries whether the output is in video mode or PC mode.

**Example:** r output itc!

**Feedback:** Returns the current setting, e.g. output ITC: video mode.

**Default:** *(See default for* ***s output itc x!*** *above.)*

## EDID Settings

### s input EDID x!

**Description:** Selects the EDID profile that will be used on the HDMI input ports. This effectively sets what EDID information is reported to the input sources.

**Variables:**

* x (1–19) – EDID mode selection for inputs:
  1. **4K2K60\_444, Stereo Audio 2.0**
  2. **4K2K60\_444, Dolby/DTS 5.1**
  3. **4K2K60\_444, HD Audio 7.1**
  4. **4K2K30\_444, Stereo Audio 2.0**
  5. **4K2K30\_444, Dolby/DTS 5.1**
  6. **4K2K30\_444, HD Audio 7.1**
  7. **1080p, Stereo Audio 2.0**
  8. **1080p, Dolby/DTS 5.1**
  9. **1080p, HD Audio 7.1**
  10. **1920×1200, Stereo Audio 2.0**
  11. **1680×1050, Stereo Audio 2.0**
  12. **1600×1200, Stereo Audio 2.0**
  13. **1440×900, Stereo Audio 2.0**
  14. **1360×768, Stereo Audio 2.0**
  15. **1280×1024, Stereo Audio 2.0**
  16. **1024×768, Stereo Audio 2.0**
  17. **720p, Stereo Audio 2.0**
  18. **Copy from HDMI Out** (reads the EDID of the connected output display and uses it for the inputs)
  19. **USER1** (a custom user-defined EDID profile)

**Example:** s input EDID 1! *(applies EDID preset #1 to all inputs)*

**Feedback:** Confirms the EDID selection by showing the active EDID name, e.g. input EDID: 4K2K60\_444, Stereo Audio 2.0.

**Default:** **4K2K60\_444, Stereo Audio 2.0** (EDID mode 1) is the default EDID used for inputs.

### r input EDID!

**Description:** Reports which EDID mode is currently selected for the HDMI inputs.

**Example:** r input EDID!

**Feedback:** Returns the active EDID name/mode, e.g. input EDID: 4K2K60\_444, Stereo Audio 2.0.

**Default:** *(See default for* ***s input EDID x!*** *above.)*

### r edid user1!

**Description:** Reads the contents of the custom **USER1** EDID memory. This returns the EDID data bytes that have been stored for the user-defined EDID profile.

**Example:** r edid user1!

**Feedback:** Returns the full EDID data for USER1 in hex format, e.g.:

user1 EDID data:

00 FF FF FF FF FF FF 00 ...

(the entire 256-byte EDID hex string will be displayed).

**Default:** *(N/A – this is a query of stored data.)*

### s edid user1 <EDID\_Data>!

**Description:** Writes a custom EDID to the **USER1** memory slot. This allows uploading a user-defined EDID by providing the 256-byte EDID hex string.

**Variables:**

* <EDID\_Data> – The 256-byte EDID content to store, represented as a continuous hex string (typically 512 hex characters). This should include the full EDID including header, descriptors, checksum, etc.

**Example:** s edid user1 00FFFFFFFFFFFF000...? (where the string of hex values represents the EDID bytes).

**Feedback:** After setting, the device will store the EDID and typically respond with a confirmation. You can verify by using the **r edid user1!** command to read back the data.

**Default:** *(N/A – the factory default USER1 EDID may be empty or a copy of a standard EDID until overwritten.)*

## Audio Settings

### s output audio x!

**Description:** Selects the audio source for the HDMI output. In multiview mode, audio can follow a specific window’s input, or it can be fixed to a particular HDMI input.

**Variables:**

* x (0–4) – Audio source selection:
  + **0** – Follow Window 1’s selected source (the audio will automatically follow whatever video source is displayed in window 1)
  + **1** – HDMI 1 input audio
  + **2** – HDMI 2 input audio
  + **3** – HDMI 3 input audio
  + **4** – HDMI 4 input audio

**Example:** s output audio 0! *(sets audio to follow window 1’s video source)*

**Feedback:** Confirms the selection, e.g. output audio: follow window 1 selected source (for option 0) or output audio: HDMI 2 input audio (if a specific input is chosen).

**Default:** **Follow window 1 selected source** (audio follows window 1 by default).

### r output audio!

**Description:** Queries the current audio source setting for the HDMI output.

**Example:** r output audio!

**Feedback:** Returns the active audio source, e.g. output audio: follow window 1 selected source.

**Default:** *(See default for* ***s output audio x!*** *above.)*

### s output audio vol x!

**Description:** Sets the output audio volume to a specific level.

**Variables:**

* x (0–100) – Volume level as a percentage (%), where 0 is mute (no audio) and 100 is the maximum volume.

**Example:** s output audio vol 30! *(sets audio volume to 30%)*

**Feedback:** Confirms the new volume setting, e.g. output audio volume: 30.

**Default:** **100** (volume is at 100% by default).

### r output audio vol!

**Description:** Queries the current output audio volume level.

**Example:** r output audio vol!

**Feedback:** Returns the volume level, e.g. output audio volume: 30.

**Default:** *(See default for* ***s output audio vol x!*** *above.)*

### s output audio vol+!

**Description:** Increases the output audio volume by one step. Each use of this command will raise the volume slightly (by 1 increment on the 0–100 scale).

**Example:** s output audio vol+!

**Feedback:** Returns the new volume after incrementing, e.g. output audio volume: 50 (if it was 49% prior, for example).

**Default:** *(N/A – this command adjusts the current volume upward.)*

### s output audio vol-!

**Description:** Decreases the output audio volume by one step. Each use of this command will lower the volume slightly (by 1 increment on the 0–100 scale).

**Example:** s output audio vol-!

**Feedback:** Returns the new volume after decrementing, e.g. output audio volume: 50 (if it was 51% prior, for example).

**Default:** *(N/A – this command adjusts the current volume downward.)*

### s output audio mute x!

**Description:** Sets the audio mute on or off for the HDMI output.

**Variables:**

* x (0 or 1) – Mute setting: 0 = **Mute Off** (unmute, audio on), 1 = **Mute On** (audio is silenced).

**Example:** s output audio mute 0! *(ensures the audio is unmuted)*

**Feedback:** Returns the new mute status, e.g. output audio mute: off (when unmuted) or output audio mute: on (when muted).

**Default:** **Mute Off** (audio is unmuted by default).

### r output audio mute!

**Description:** Queries whether the output audio is currently muted or not.

**Example:** r output audio mute!

**Feedback:** Returns the current mute status, e.g. output audio mute: off or output audio mute: on.

**Default:** *(See default for* ***s output audio mute x!*** *above.)*

## Single Screen Mode Settings

*(Single Screen Mode refers to the scenario where the multiviewer is showing only one source full-screen, essentially acting as a simple switcher. These commands control input selection and auto-switch features in that mode.)*

### s in source x!

**Description:** Routes a specific HDMI input to the output in single-screen mode. This selects which single source is displayed full-screen on the output.

**Variables:**

* x (1–4) – The input port number to route to the output:
  + **1** – HDMI 1
  + **2** – HDMI 2
  + **3** – HDMI 3
  + **4** – HDMI 4

**Example:** s in source 1! *(switches the output to display HDMI 1)*

**Feedback:** The device confirms the routed source, e.g. HDMI 1 (meaning HDMI 1 is now the active output).

**Default:** **HDMI 1** is the default input routed to the output (at startup or in absence of other selection).

### r in source!

**Description:** Reads which HDMI input is currently routed to the output (in single-screen mode).

**Example:** r in source!

**Feedback:** Returns the active input source name/number, e.g. HDMI 1.

**Default:** *(See default for* ***s in source x!*** *above.)*

### s auto switch x!

**Description:** Enables or disables the **Auto Switch** feature. When enabled, the device will automatically switch the output to an active input based on signal detection (e.g., if the current source is lost, it may switch to another with a valid signal). This is only applicable in single-screen mode.

**Variables:**

* x (0 or 1) – Auto-switch setting: 0 = **Disable** auto-switching, 1 = **Enable** auto-switching.

**Example:** s auto switch 0! *(turns auto-switch off)*

**Feedback:** Confirms the setting, e.g. auto switch off (if disabled) or auto switch on (if enabled).

**Default:** **Auto Switch Off** (the auto switching feature is disabled by default).

### r auto switch!

**Description:** Checks whether the Auto Switch feature is currently enabled or disabled.

**Example:** r auto switch!

**Feedback:** Returns the status, auto switch on or auto switch off.

**Default:** *(See default for* ***s auto switch x!*** *above.)*

## Multi-viewer Mode Settings

*(Multi-viewer Mode allows multiple source windows to be displayed simultaneously on the screen in various layouts: PIP, PBP, triple, quad, etc. The following commands control multi-view layouts and related features.)*

### s multiview x!

**Description:** Selects the multi-viewer display mode (window layout). This command changes the output to one of the multi-window configurations or back to single screen.

**Variables:**

* x (1–5) – Multi-view layout mode:
  + **1** – Single Screen (one input full-screen)
  + **2** – PIP (Picture-in-Picture, one full-screen with a second small window)
  + **3** – PBP (Picture-by-Picture, two sources side by side)
  + **4** – Triple Screen (three sources)
  + **5** – Quad Screen (four sources in a 2x2 grid)

**Example:** s multiview 2! *(switches to PIP mode)*

**Feedback:** The device confirms the mode by name, e.g. PIP or quad screen, etc. For example, single screen if x=1.

**Default:** **Single screen** mode is the default output mode.

### r multiview!

**Description:** Queries the current multi-viewer display mode.

**Example:** r multiview!

**Feedback:** Returns the active mode name, e.g. single screen, PIP, quad screen, etc.

**Default:** *(See default for* ***s multiview x!*** *above.)*

### s window x in y!

**Description:** Assigns an input source to a specific window in the current multi-view layout. For example, in quad mode with four windows, this command selects which HDMI input appears in a given window number.

**Variables:**

* x (1–4) – Window number in the multi-view layout to assign (depending on the mode, not all windows may be visible; e.g., PIP uses windows 1 and 2).
* y (1–4) – HDMI input number to display in that window.

**Example:** s window 1 in 3! *(displays HDMI 3 input in window #1 of the multi-view layout)*

**Feedback:** Confirms the window assignment, e.g. window 1 select HDMI 3. If all windows (x=0) are targeted and supported, the device may set all windows to the specified input (if such a feature exists using x=0).

**Default:** *No specific default window-to-input mapping (in Single Screen mode, window 1 defaults to HDMI 1 as noted above). In multi-view modes, by default window 1 may show HDMI 1, window 2 HDMI 2, etc., unless changed.*

### r window x in!

**Description:** Reports which input source is currently assigned to a given window in the multi-view layout.

**Variables:**

* x (0–4) – Window number to query: 1–4 for a specific window, or **0** to query **ALL** windows at once.

**Example:** r window 2 in!

**Feedback:** Returns the source for that window, e.g. window 2 select HDMI 4. If x=0 (ALL), the device will list the source for each window in the layout (each on a new line), for example:

window 1 select HDMI 1

window 2 select HDMI 2

window 3 select HDMI 3

window 4 select HDMI 4

(depending on mode and windows available).

**Default:** *See* ***s window x in y!*** *default note above.*

### s multiview x roaming switch y!

**Description:** Enables or disables the **USB mouse roaming** feature for a given multi-view mode. When mouse roaming is enabled for a mode, the user can seamlessly move the mouse cursor across that multi-view layout’s windows to switch USB control focus (useful in KVM scenarios). This setting can be applied per layout or to all layouts.

**Variables:**

* x (0–5) – Target layout mode: 0 = **ALL** modes, 1 = Single Screen, 2 = PIP, 3 = PBP, 4 = Triple, 5 = Quad.
* y (0 or 1) – Roaming switch: 0 = **Off** (disable mouse roaming for the selected mode(s)), 1 = **On** (enable mouse roaming).

**Example:** s multiview 5 roaming switch 1! *(enables mouse roaming in Quad mode)*

**Feedback:** Confirms the new setting for the specified mode. For example, quad screen roaming switch on. If x=0 (all), it may confirm each mode’s new state or a general confirmation.

**Default:** By default, **Single Screen** mode’s roaming is **Off** (since only one window), and multi-window modes (PIP/PBP/Triple/Quad) have roaming **On** by default.

### r multiview x roaming switch!

**Description:** Queries the mouse roaming feature status for a given multi-view mode or all modes.

**Variables:**

* x (0–5) – Mode to query (0 = ALL, 1 = Single, 2 = PIP, 3 = PBP, 4 = Triple, 5 = Quad).

**Example:** r multiview 1 roaming switch!

**Feedback:** Returns the status for the queried mode. For example, single screen roaming switch off or PIP roaming switch on. If x=0 (ALL), the device will list each mode’s status on separate lines. For example:

single screen roaming switch off

PIP roaming switch on

PBP roaming switch on

triple screen roaming switch on

quad screen roaming switch on

**Default:** *See defaults under* ***s multiview x roaming switch y!*** *above.*

### r PIP position!

**Description:** Retrieves the current position of the PIP (Picture-in-Picture) sub-window on the screen.

**Example:** r PIP position!

**Feedback:** Returns the PIP window placement, e.g. PIP on right top (meaning the smaller PIP window is currently at the top-right corner).

**Default:** **Right Top** is the default PIP window position.

### s PIP position x!

**Description:** Sets the position of the PIP window on the screen to one of the preset corners or to user-defined coordinates.

**Variables:**

* x (1–5) – PIP position preset:
  + **1** – Left Top
  + **2** – Left Bottom
  + **3** – Right Top
  + **4** – Right Bottom
  + **5** – User (custom position)

**Example:** s PIP position 3! *(moves the PIP window to the top-right corner)*

**Feedback:** Confirms the new PIP location, e.g. PIP on right top. If **User** mode is selected (x=5), the PIP window will move to the custom coordinates defined by the user (see **s PIP Hstart Vstart Hsize Vsize!** below).

**Default:** **Right Top** (option 3) is the default PIP position.

### s PIP Hstart Vstart Hsize Vsize!

**Description:** Defines a custom PIP window size and position when using the **User** PIP mode. This command is used to precisely place and size the PIP window. (Effective only if PIP position is set to 5, *User*).

**Variables:**

* Hstart – Horizontal start position (1–100), as a percentage of the screen width from the left.
* Vstart – Vertical start position (1–100), as a percentage of the screen height from the top.
* Hsize – Horizontal size (width) of the PIP window (1–100), as a percentage of full screen width.
* Vsize – Vertical size (height) of the PIP window (1–100), as a percentage of full screen height.
  + *Note:* The values must not cause the window to overflow the screen. The device expects Hstart + Hsize <= 101 and Vstart + Vsize <= 101 to ensure the PIP window fits entirely on screen.

**Example:** s PIP 10 10 20 20! *(places the PIP window starting at 10% from left and top, with a width 20% and height 20% of the screen)*

**Feedback:** Confirms the custom PIP geometry, echoing the values, e.g. PIP 10 10 20 20 (indicating the PIP window’s X, Y position and size).

**Default:** By default, the device uses preset positions/sizes. The custom values take effect only when **User** mode is selected. (No specific default user values beyond the constraints.)

### s PIP size x!

**Description:** Sets the size preset for the PIP window. This controls how large the PIP sub-window is relative to the screen when using a preset size.

**Variables:**

* x (1–4) – PIP window size preset:
  + **1** – Small
  + **2** – Middle (medium)
  + **3** – Large
  + **4** – User (custom size)

**Example:** s PIP size 3! *(sets the PIP window to the large preset size)*

**Feedback:** Confirms the selection, e.g. PIP size: large. If **User** is selected (x=4), the actual PIP size will be determined by user-defined values (if such a command exists for size; note that custom sizing might be managed via the user-defined position command above if it also covers size).

**Default:** **Large** (option 3) is the default PIP window size.

### r PIP size!

**Description:** Queries the current preset size setting of the PIP window.

**Example:** r PIP size!

**Feedback:** Returns the size status, e.g. PIP size: large.

**Default:** *(See default for* ***s PIP size x!*** *above.)*

### s PBP mode x!

**Description:** Sets the Picture-by-Picture (PBP) layout mode. There are two predefined layouts for dual-window PBP mode.

**Variables:**

* x (1 or 2) – PBP layout selection:
  + **1** – PBP Mode 1 (one arrangement of the two windows)
  + **2** – PBP Mode 2 (an alternate arrangement of the two windows)

**Example:** s PBP mode 1!

**Feedback:** Confirms the layout, e.g. PBP mode 1. (Each mode represents a specific side-by-side configuration of the two sources.)

**Default:** **PBP mode 1** is the default PBP layout.

### r PBP mode!

**Description:** Queries which PBP layout mode is currently active.

**Example:** r PBP mode!

**Feedback:** Returns the mode, e.g. PBP mode 1.

**Default:** *(See default for* ***s PBP mode x!*** *above.)*

### s PBP aspect x!

**Description:** Sets the aspect ratio mode for the PBP windows. This determines how the two PBP windows are scaled (full screen stretch or with 16:9 aspect).

**Variables:**

* x (1 or 2) – PBP aspect ratio setting:
  + **1** – Full screen (each PBP window is stretched to fill its half of the screen)
  + **2** – 16:9 (each PBP window maintains 16:9 aspect ratio, may letterbox/pillarbox as needed)

**Example:** s PBP aspect 1! *(sets PBP windows to fill each half fully)*

**Feedback:** Confirms the aspect setting, e.g. PBP aspect: full screen.

**Default:** **Full screen** is the default PBP aspect ratio setting.

### r PBP aspect!

**Description:** Queries the current aspect ratio mode set for PBP windows.

**Example:** r PBP aspect!

**Feedback:** Returns the setting, e.g. PBP aspect: full screen.

**Default:** *(See default for* ***s PBP aspect x!*** *above.)*

### s triple mode x!

**Description:** Sets the layout mode for Triple-window display. Two preset triple-window arrangements are available.

**Variables:**

* x (1 or 2) – Triple-window layout selection:
  + **1** – Triple Mode 1
  + **2** – Triple Mode 2

*(Each mode corresponds to a different arrangement of the three windows on screen.)*

**Example:** s triple mode 1!

**Feedback:** Confirms the layout selection, e.g. triple mode 1.

**Default:** **Triple mode 1** is the default triple-window layout.

### r triple mode!

**Description:** Queries which triple-window layout mode is currently active.

**Example:** r triple mode!

**Feedback:** Returns the mode, e.g. triple mode 1.

**Default:** *(See default for* ***s triple mode x!*** *above.)*

### s triple aspect x!

**Description:** Sets the aspect ratio handling for Triple-window mode. Similar to PBP aspect, this determines if windows fill their slots or maintain 16:9 aspect.

**Variables:**

* x (1 or 2) – Triple mode aspect setting:
  + **1** – Full screen (windows fill each allocated area completely)
  + **2** – 16:9 (windows maintain 16:9 aspect ratio within their areas)

**Example:** s triple aspect 1!

**Feedback:** Confirms the aspect ratio setting, e.g. triple aspect: full screen.

**Default:** **Full screen** is the default aspect mode for triple-window layouts.

### r triple aspect!

**Description:** Queries the current aspect ratio setting for Triple-window mode.

**Example:** r triple aspect!

**Feedback:** Returns the setting, e.g. triple aspect: full screen.

**Default:** *(See default for* ***s triple aspect x!*** *above.)*

### s quad mode x!

**Description:** Sets the layout mode for Quad-window display. Two preset quad-window arrangements are available.

**Variables:**

* x (1 or 2) – Quad-window layout selection:
  + **1** – Quad Mode 1
  + **2** – Quad Mode 2

*(Each mode corresponds to a different 2×2 arrangement of the four windows — typically these might differ in ordering of windows or scaling.)*

**Example:** s quad mode 1!

**Feedback:** Confirms the layout selection, e.g. quad mode 1.

**Default:** **Quad mode 1** is the default quad-window layout.

### r quad mode!

**Description:** Queries which quad-window layout mode is currently active.

**Example:** r quad mode!

**Feedback:** Returns the mode, e.g. quad mode 1.

**Default:** *(See default for* ***s quad mode x!*** *above.)*

### s quad aspect x!

**Description:** Sets the aspect ratio handling for Quad-window mode. This determines how each of the four windows is scaled within its quadrant.

**Variables:**

* x (1 or 2) – Quad mode aspect setting:
  + **1** – Full screen (each window fills its quadrant completely)
  + **2** – 16:9 (each window maintains 16:9 aspect ratio within its quadrant, possibly with letterboxing)

**Example:** s quad aspect 1!

**Feedback:** Confirms the aspect setting, e.g. quad aspect: full screen.

**Default:** **Full screen** is the default aspect mode for quad-window layouts.

### r quad aspect!

**Description:** Queries the current aspect ratio setting for Quad-window mode.

**Example:** r quad aspect!

**Feedback:** Returns the setting, e.g. quad aspect: full screen.

**Default:** *(See default for* ***s quad aspect x!*** *above.)*

### r mousekeys!

**Description:** Queries the status of the **Mouse Keys (Mouse Roaming)** feature for triple-window mode (and related functionality). This feature, when enabled, allows using the mouse to switch control between windows (especially in triple layout, where roaming might involve moving across screens).

**Example:** r mousekeys!

**Feedback:** Returns the status of the mouse roaming/hotkey feature, e.g. mousekeys switch: on or mousekeys switch: off.

**Default:** *(N/A – see* ***s mousekeys x!*** *for default state.)*

### s mousekeys x!

**Description:** Enables or disables the **Mouse Hotkey Switch** feature. This feature allows the user to use a mouse movement or hotkey to seamlessly switch USB control between source windows (particularly relevant in multi-view modes).

**Variables:**

* x (0 or 1) – Mouse hotkey switch: 0 = **Disable** the mouse hotkey/roaming feature, 1 = **Enable** it.

**Example:** s mousekeys 1! *(enables the mouse hotkey/roaming control feature)*

**Feedback:** Confirms the new state, e.g. mousekeys switch: on (if enabled) or mousekeys switch: off.

**Default:** **On** – The mouse hotkey/roaming feature is enabled by default for multi-view modes (so the user can switch control between windows out-of-the-box).

### s window source osd x!

**Description:** Turns the window **Source OSD (On-Screen Display)** labels on or off. When on, each window in a multi-view layout will display a small label showing which input source is being shown.

**Variables:**

* x (0 or 1) – Source label OSD: 0 = **Off** (no source labels on windows), 1 = **On** (show input source name/number on each window).

**Example:** s window source osd 1! *(enables on-screen source labels for each window)*

**Feedback:** Confirms the setting, e.g. window source osd: on or window source osd: off.

**Default:** **On** – Source OSD labels are on by default (window labels will be shown).

### r window source osd!

**Description:** Queries whether the window source labels (OSD) are currently enabled or disabled.

**Example:** r window source osd!

**Feedback:** Returns the status, e.g. window source osd: on (if labels are currently displayed).

**Default:** *(See default for* ***s window source osd x!*** *above.)*

### r window usb border!

**Description:** Queries the status of the **USB Control Border** indicator. In multi-view KVM operation, the active window (the one currently controlled via USB) can be highlighted with a border. This command checks if that feature is on or off.

**Example:** r window usb border!

**Feedback:** Returns the status, e.g. window usb border: on or window usb border: off.

**Default:** *(See default for* ***s window usb border x!*** *below.)*

### s window usb border x!

**Description:** Enables or disables the **USB Control Border** around the active window. When enabled, the currently selected window for USB/KVM control will be outlined (to clearly indicate which source is being controlled by keyboard/mouse).

**Variables:**

* x (0 or 1) – USB border display: 0 = **Off** (no border highlighting), 1 = **On** (border around active window).

**Example:** s window usb border 1! *(turns on the USB control border highlight)*

**Feedback:** Confirms the setting, e.g. window usb border: on (when enabled) or similar for off. If a border was on, after disabling it might simply stop showing without a textual response aside from confirmation.

**Default:** **On** – The USB control border feature is on by default, so the active window is highlighted.

### r usb select win!

**Description:** Queries which window is currently selected for USB (KVM) control. In multi-viewer modes, the user can assign the USB hub (keyboard/mouse) to control one of the source devices. This command tells which window (source) is currently active for KVM.

**Example:** r usb select win!

**Feedback:** Returns the window number that has USB focus, e.g. usb in win 1 (meaning window 1’s source is currently being controlled via USB).

**Default:** *(N/A – see* ***s usb select win x!*** *default.)*

### s usb select win x!

**Description:** Assigns the USB control (keyboard/mouse) to a specific window’s source. Use this to manually switch KVM control from one source to another in multi-view mode.

**Variables:**

* x (1–4) – Window number whose source should get USB control. (Only one window/source can be controlled at a time through the USB hub.)

**Example:** s usb select win 1! *(grants USB control to the source in window 1)*

**Feedback:** Confirms the selection, e.g. usb in win 1 (indicating USB is now controlling the device shown in window 1).

**Default:** **Window 1** is the default USB-controlled window (the keyboard/mouse control focuses on window 1’s source by default).