

**format the text from the given links:**

<https://docs.nvidia.com/jetson/archives/r34.1/DeveloperGuide/text/SD/WindowingSystems/WestonWayland.html>  
<https://docs.nvidia.com/jetson/archives/r34.1/DeveloperGuide/text/SD/WindowingSystems/XWindowSystem.html>

Here is a **structured, sectioned summary** of the NVIDIA Jetson documentation for **Weston (Wayland)** and the **X Window System**, with all relevant commands and configuration details clearly listed for easy chatbot parsing and information extraction.

**1. Weston (Wayland) on Jetson**

**1.1 Overview**

* **Wayland** is a protocol for communication between a display server (compositor) and its clients (GUI apps).
* **Weston** is the reference implementation of a Wayland compositor.
* **Weston** manages:
  + Display composition
  + Input device events (touch, mouse, keyboard)
  + Display settings (wallpaper, resolution, multi-monitor, etc.)
* **Weston** is lightweight and fast, suitable for embedded/mobile use.

**1.2 Architecture**

* **Wayland server libraries**: Implement the Wayland protocol for Weston.
* **Weston libraries**: Implement the compositor, use Kernel Mode Setting (KMS), OpenGLES, and Direct Rendering Manager (DRM), manage input devices.
* **Wayland clients**: Communicate with the compositor using the Wayland protocol (can be EGL apps, rootless X servers, etc.).

**1.3 Shells**

* **Shell plugins** provide different GUIs:
  + **Desktop shell**: Modern desktop (like X11), implemented by desktop-shell.so, with a client weston-desktop-shell for wallpaper, panels, etc.
  + **IVI shell**: For In-Vehicle Infotainment (IVI), implemented by ivi-shell.so, provides GENIVI Layer Manager API.

**1.4 Configuration**

* Configure Weston via /etc/xdg/weston/weston.ini.
* See the weston.ini man page for options.

**1.5 Environment Variables**

* WAYLAND\_DEBUG: Set to any value to print live protocol to stderr.
* XDG\_RUNTIME\_DIR: Directory for Weston's socket/lock files. Weston and clients use this for communication.

**1.6 Running Weston**

**1.6.1 Using a Script**

$ nvstart-weston.sh

* Launches Weston with desktop-shell by default.
* To run a Wayland client:

$ weston-simple-egl

**1.6.2 Manual Launch**

1. **Stop X server if running:**

$ sudo service gdm stop; sudo pkill -9 Xorg

1. **Load DRM driver:**
   * For Jetson AGX Orin:

$ sudo modprobe nvidia-drm modeset=1

* + For Jetson AGX Xavier/NX:

$ sudo modprobe tegra-udrm modeset=1

1. **Symlink libgbm.so.1:**

$ sudo ln -sf /usr/lib/aarch64-linux-gnu/tegra/libnvgbm.so /usr/lib/aarch64-linux-gnu/libgbm.so.1

1. **Set up environment:**

$ unset DISPLAY  
$ mkdir /tmp/xdg  
$ chmod 700 /tmp/xdg  
$ export WESTON\_TTY=1

1. **Launch Weston as root:**

$ sudo XDG\_RUNTIME\_DIR=/tmp/xdg weston --tty="$WESTON\_TTY" --idle-time=0 &

* + Optional: Set protocol debug output:

$ export WAYLAND\_DEBUG=server

* + Optional: Use EGLStream:

$ sudo XDG\_RUNTIME\_DIR=/tmp/xdg weston --tty="$WESTON\_TTY" --use-egldevice &

1. **Launch as non-root (with weston-launch):**

$ sudo groupadd weston-launch  
$ sudo usermod -a -G weston-launch $USER  
$ sudo chown root /usr/bin/weston-launch  
$ sudo chmod +s /usr/bin/weston-launch  
$ weston-launch &

1. **Run a Wayland client:**

$ sudo XDG\_RUNTIME\_DIR=/tmp/xdg weston-simple-egl

1. **Run Weston tests:**

$ sudo XDG\_RUNTIME\_DIR=/tmp/xdg weston --modules=/usr/lib/aarch64-linux-gnu/tegra/weston/weston-test.so

1. **Launch with IVI shell and controller:**

$ weston --shell=ivi-shell.so --modules=ivi-controller.so

**1.7 Multi-Display Support**

* Weston supports multiple display heads and overlays.
* Configure display options in weston.ini:

name=HDMI-A-1  
mode=1920x1080@60.0  
transform=90

* Applications launch on the display with the mouse pointer.
* Hot-plugging is supported; applications move to active displays if a monitor is removed.

**1.8 Hot-Plugging**

* Weston uses udev to monitor DRM hotplug events.
* Handles HDMI/DP connect/disconnect dynamically.
* IVI shell supports hotplug via ivi-controller and ivi-wm protocol.

**1.9 Compositing Modes**

* **Overlay-only mode**: Each surface/view assigned a plane.
* **Mixed mode**: Some surfaces use overlays, others are GL-composited.
* **GL-only mode**: All surfaces composited with GL.
* Force GL-only mode:

$ export WESTON\_FORCE\_RENDERER=1

* Use weston-debug to verify overlay assignment:

$ sudo XDG\_RUNTIME\_DIR=/tmp/xdg ./weston-debug -a

**2. X Window System on Jetson**

**2.1 Overview**

* **X Window System** provides windowing, graphics, and device management for GUI applications.
* **X server** is the standard implementation, supported in Jetson Linux.

**2.2 Starting X Server Manually**

* **Start X server:**

$ sudo -b X -ac -noreset -nolisten tcp

* **Stop X server:**

$ ps aux | grep "X"  
$ sudo kill <pid>

**2.3 Runtime Configuration**

* X server configuration via:
  + Command line options
  + Environment variables
  + xorg.conf and xorg.conf.d files
  + Auto-detection/fallbacks
* **Utilities:**
  + xrandr: Set display properties (part of x11-xserver-utils)
  + nvidia-xconfig: Configure xorg.conf

**2.4 Using xrandr**

* **Query displays/modes:**

$ xrandr

* **Get help:**

$ xrandr --help

**2.5 Modifying Static Configuration**

* **Config files:**
  + /etc/X11/xorg.conf
  + /etc/X11/xorg.conf.d/
  + /usr/share/X11/xorg.conf.d/
* By default, settings are auto-detected; use nvidia-xconfig for manual edits.

**2.6 Using nvidia-xconfig**

* **Help:**

$ nvidia-xconfig --help  
$ nvidia-xconfig --advanced-help

* **Set custom EDID:**

$ nvidia-xconfig --custom-edid=HDMI-<n>:<path>

* **Set color bit-depth:**

$ nvidia-xconfig --depth=<depth>

* **Set display mode:**

$ nvidia-xconfig -mode=<mode>

* **Enable debug mode:**

$ nvidia-xconfig -mode-debug  
$ nvidia-xconfig -no-mode-debug

* **Enable single X screen:**

$ nvidia-xconfig -only-one-x-screen

* **Enable screen mirroring:**

$ nvidia-xconfig --metamode-orientation=clone

* **Enable screen spanning:**

$ nvidia-xconfig --metamode-orientation=<value>

**2.7 Advanced Display Features**

* **Blending/Overlay:**
  + TegraOverlayPriority and TegraOverlayBlendmode options in xorg.conf.
  + Example:

Option "MetaModes" "nvidia-auto-select { TegraOverlayBlendmode = PremultSourceAlphaBlend, TegraOverlayPriority = 0 }"

* + Query/modify at runtime:

$ xrandr --prop  
$ xrandr --output HDMI-0 --set TegraOverlayPriority 0  
$ xrandr --output HDMI-0 --set TegraOverlayBlendmode PremultSourceAlphaBlend

**3. Key Takeaways**

* **Weston/Wayland** is recommended for lightweight, modern, and embedded use cases.
* **X Window System** remains standard for legacy and advanced configurations, with robust runtime and static configuration tools.
* Both systems support multi-display, hot-plug, and advanced display features on Jetson hardware.
* Configuration and debugging are heavily command-driven, with clear support for both automated and manual workflows.

**All commands and configuration options are presented for direct use or scripting, suitable for chatbot extraction and automation.**

⁂