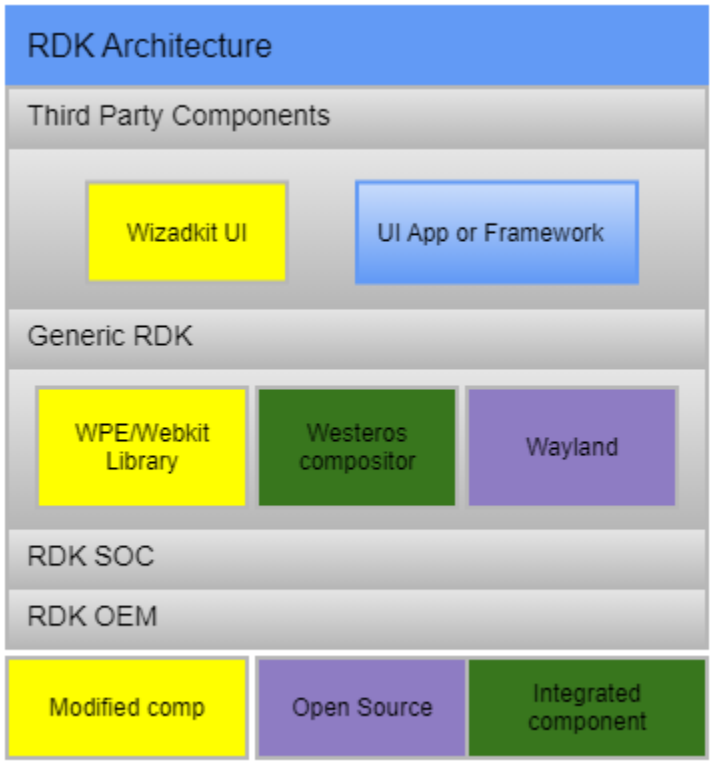
Westeros

Overview

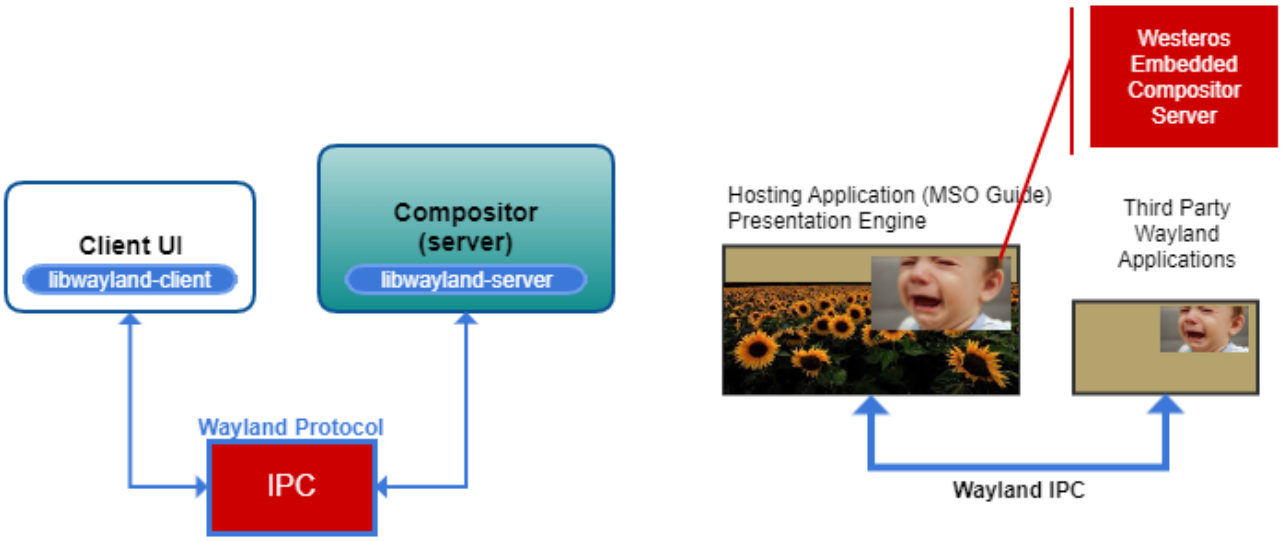
Westeros is a lightweight Wayland Compositor library that allows users to create Wayland displays and also allows nesting and embedding displays that contain third-party applications. The WPE integration with Westeros enables a better video experience with increased smoothness and enhanced browser responsiveness. Westeros is designed for embedded systems and is a replacement for Weston.

- Westeros is small and simple making it easier to understand and maintain.
- Westeros caters to the needs of embedded systems over traditional desktop computing.
- Westeros is a shared library that provides an API for creating and operating a compositor.
- Westeros includes sample compositor app OR you can implement a custom compositor.
- Westeros allows an application to create a Wayland display within itself- referred to as an embedded compositor. The main UI can then control the embedded application window and lifecycle.

Architecture



Westeros Use Case



The hosting application (MSO Guide) has control over the presentation & composition of third-party applications.

Advantages of Westeros over Weston Compositor

There are two main sets of attributes of Westeros that make it a good choice:

First

- Westeros is small, simple, and targets embedded systems.
- Its small code size makes it easier to understand and maintain.
- Westeros concentrates on functionality needed for embedded systems rather than trying to include features that are related to traditional desktop computing.

Second

- The functionality of Wayland composition is in a shared library that provides an API for creating and operating a compositor.
- You can use the included sample compositor app, or you can use the sample as a reference for implementing a custom compositor.
- The use of embedded compositors allows applications to create additional Wayland display within itself.

A system could have a primary user interface which allows other third party applications to provide additional functionality such as Netflix, and the primary UI can control the size and position of the third party UI to provide seamless integration. If, on the other hand, you want to implement a system where the user moves, resizes, minimizes, and maximizes various windows, and does drag and drop operations etc., then Weston is a more appropriate choice.

Westeros as Wayland Library

- Westeros implements the Wayland protocols and is compatible with applications that are built to use Wayland compositors.
- Westeros supports the creation of normal, nested, and embedded Wayland Compositors.
- Westeros includes memory management primitives in order to provide a better video experience.

Build Steps

Bitbake recipes for building wpewebkit, which is integrated with Westeros, are available in the `meta-metrological` layer in the CMF space. Changes which are required to build and run Westeros-integrated applications on the Raspberry Pi platform, have been merged into `meta-cmf-raspberrypi` CMF Layer.

Steps to run a Westeros-enabled Application

Since the Westeros compositor is integrated with the WPE Browser, any application/link using WPE can be used to demonstrate a Westeros compositor.

To run a Westeros-enabled application:

1. Set the following environment variables prior to running a compositor-enabled application:

```
$ export XDG_RUNTIME_DIR=/run/user/0/
$ export WAYLAND_DISPLAY=WPE
```

2. Run the Westeros compositor for WPE display server:

```
$ westeros --renderer /usr/lib/libwesteros_render_gl.so.0.0.0 --display WPE
```

3. Run the wizardkit UI Application:

```
$ WPELauncher http://<RaspberryPi Hybrid Ip>:80/wizardkit/ui/guide.html
```

4. Run the pre-built Westeros bits included with the test case:

```
$ westeros_test --display WPE
```