

Pure Wayland toolkit prototype

XWayland server provides limited capabilities for X11 desktop applications (see [X11 Application Support](#) and [JDK-8269245](#)). In order to get full support of the desktop features, we need to implement a pure Wayland client toolkit for java. Wayland architecture in [many ways](#) differs from X11, so we cannot reuse XAWT even for basic capabilities. The new toolkit should be implemented from scratch. Here are some major chunks of work:

- Event handling
 - Dispatch native events on EDT, to avoid potential race conditions when the state is updated both from EDT and toolkit thread. See the full proposal [here](#)
- Graphics devices support
 - Onscreen/offscreen Wayland surface management
 - Adopt OGL pipeline for rendering on Wayland surfaces
 - Implement a new rendering pipeline based on Vulkan (for better performance)
- `java.awt.Robot`
 - Sending input events
 - Reading screen data (at least current java application windows)
- Client-side decorations for windows
 - Swing internal frames rendering code can be reused

The prototype implementation of the pure Wayland toolkit ([JDK-8281970](#)) can be found in the Wakefield repository ([pure_wl_toolkit](#) branch). It uses software rendering loops to draw geometry primitives on Wayland surfaces. It supports the rendering of AWT Frame and Swing JFrame.

```
java -Dawt.toolkit.name=WLToolkit JFTest
```

[blocked URL](#)

The `java.awt.Robot` functionality is implemented in a somewhat roundabout way because it contradicts several basic principles of Wayland such as not exposing pixels after they have been composited. As a short- to mid-term solution, we intend to run tests in an instance of Weston (one such instance per test) that runs on top of X11. That instance of Weston requires a custom plugin `libwakefield.so` (source code is under [src/java.desktop/share/native/libwakefield](#)). A [Java wrapper](#) is available that helps to fire up and tear down Weston and set up `WAYLAND_DISPLAY` for the test. There is also a [sample test](#) `test/jdk/java/awt/wakefield/ScreenCapture.java` that can be run roughly as follows:

```
export LIBWAKEFIELD=../build/wakefield/libwakefield.so
jtest -e:XDG_RUNTIME_DIR -e:LIBWAKEFIELD -testjdk:... test/jdk/java/awt/wakefield/
```

This was verified to work with Weston 9 on Ubuntu 21.10.

Text rendering has been implemented ([JDK-8281970](#))

```
java -Dawt.toolkit.name=WLToolkit components.TextSamplerDemo
```

[blocked URL](#)

Keyboard and mouse input has been implemented:

[blocked URL](#)

Also, `WLToolkit` now has support for

- Client-side decorations,
- Interactive window resize,
- Maximize/minimize/fullscreen, restricting min/max window size.