

# Meeting Notes

The Wakefield committers may hold off-line discussions from time-to-time. Since these meetings are just for the committers and invited wayland experts, summary minutes will be recorded here as time permits

## Online Zoom Meeting 9am PDT 25th May 2023

Attendees: Victor D'yakov, Alexey Ushakov, Phil Race, Kevin Rushforth, Maxim Kartashev, Olivier Fourdan, Jonas Adahl, Alexander Zvegintsev

Alexander:

*we have pushed several changes related to the X11 compatibility :*

[JDK-8307779](#) - Relax the java.awt.Robot specification

[JDK-8280993](#) - [XWayland] Popop is not closed on click outside of area controlled by XWayland

[CSR JDK-8307456](#) approved for the java.awt.Robot taking screenshots issue with Screencast. This fix is also targeted to JDK 21.

*Drag and Drop does not work in java -> wayland app direction: while this DND fix has been released on the system side(1, 2), changes are still required on the JDK side.*

*Our code is quite picky for the drop target, and requires it to be a toplevel with WM\_STATE property, whereas the Wayland server uses dummy windows without this property set.*

*I already have a fix for this on the JDK side that I want to put in 21, but first I want to make sure I didn't break anything. In any case, it will be active when the Wayland session is active.*

Phil:

*We still have test failures, some of them are tests that really was not so stable, and the timing thing on Wayland vs Xorg makes them fail.*

*Interesting categories of failures are number of crashed that seem to be in pipewire code.*

*It is a mixture on the thread that attached to the JDK where we actually calling something and occasionally some threads that pipewire kick of on its own.*

*We have seen at least one that says "stack smash", terminating.*

*And there is one that is not a crash, with one particular test on particular systems, which we still need to analyze, related to be not able to get the screen data.*

*Some of the crashes are in tests that do a lot of captures in a loop.*

*There is one test which repeatedly moves a mouse cursor and taking a pixel color at mouse position. It makes around ~11000 of getPixelColor requests, which takes no time on Xorg, but can time out with the new API. Alexander said that one request can be 3000x slower on his machine comparing to X11 implementation. So anything that captures pixels in a loop over and over again is a potential problem.*

*Sometimes when a test fails, test harness makes its own screencapture from another VM, so we wondering about the concurrency of all these libraries we are using now.*

*It could be a mixture of bugs in our code, bugs in pipewire that are fixed in a later version, or bugs are not yet fixed.*

Jonas:

*Some comments about threads, I would first check that you have handled the processes callback from pipewire, and also check that you have not enabled the real time pipewire thread(It is normally not enabled, as you should explicitly enable it).*

*Regarding performance, you are using the screencasting not for screencasting, not the way it is intended, so it is not strange that it takes so much longer time, due to security checks, sessions processing, streams and all that stuff.*

*The workaround for this particular test case is to keep this stream open, reducing the number of re-negotiation, permission checking, etc.*

Alexey:

*It reminds me one optimization, that we did for metal pipeline, we have display sync thread which started and stopped on each redraw request.*

So the optimization was to have a keepalive counter that allows us to keep at alive enough to catch subsequent requests.

Probably this approach can be used here.

Phil:

I do not want to overblow this, there are few tests/use cases which really suffers from this, most cases do screen capture a couple of times. So this enhancement could be done later, I don't think that we should rush on that and redone all of that in JDK 21.

All of the TCK tests pass, it is a good step that we pass all the conformance tests. It is the some of functional tests failing.

The plan is to ship this screencast code in 21, and if somebody happens to run on Wayland it should work modular this crashes. We are not gonna say it is supported, we hoping it should practically for most cases.

Olivier:

XTest will be wired to libei, which will actually move your mouse pointer using the Wayland compositor.

The good news is that [libei 1.0.0rc1](#) was released a couple of days ago so we are working hard to try get everything in place, and it is moving nicely.

Maxim:

I fixed a few bugs in our implementation of Wayland toolkit regarding scaling multiscreen environment with different scaling.

What I did was that I tried to keep the buffer associated with a window in sync with the scaling of the window of the output where the window is on(use maximum). So if I move window from an output with 200% scale to an output with 100% scale, the buffer actually shrinks, but now I am not sure that I really have to do this, because the window to my does not change at all.

Wayland seems to scale the buffer correctly. Do I really need to go through the hassle of adjusting the buffer size and have the buffer scaled to the highest available scale of outputs and leave all work to Wayland?

Jonas:

I would suggest changing the buffer whenever it makes sense. If you don't do this, there are two issues:

1. when the compositor scales down fonts will look slightly more blurry and will be less sharp
2. If you're working on two monitors (say, 100% and 200%) on a battery-powered machine, and your main program is always on a lower-scaled monitor, all that drawing at 200% and scaling down work will increase power consumption and decrease battery life.

Right now it means that you always check at what outputs you are actually on, an take the maximum scale.

There is an extension for the fractional scaling that give you preferred scaling. (Gnome 44+)

Alexey:

We are in progress of implementing Vulkan rendering for pure Wayland prototype

Alexander:

I recall that there was a [HiDPI fix for X11 compatibility mode](#) on Wakefield mailing list.

Is there any chance that it could be targeted for 21?

Maxim:

It depends on a fix I [merged](#) yesterday, so there is no chance that it will go to 21.

## Online Zoom Meeting 9am PDT 27th Apr 2023

Attendees: Victor D'yakov, Alexander Zvegintsev, Olivier Fourdan, Niels de Graef, Phil Race, Maxim Kartashev, Kevin Rushforth

Victor:

Alexander, I know you submitted a link to a proposed change to the wakefield sandbox repo, and our plan is to go to openjdk mainline code review as well, right?

Please make it much more verbose, as your last submission was really poor on explanation.

Alexander:

Yes, I posted it to the Wakefield sandbox a few weeks ago as a preliminary review of code changes(1, 2) several weeks ago. No work has been done in the last two weeks as we are now doing a test sprint.

I will post it to the openjdk repo next week to get it pushed out before the fork of JDK21. It will have more explanations.

Maxim:

Nikita Gubarkov is doing the HiDPI, it relies on me integrating [cached bounds and insets for GraphicsDevice into the mainline](#), and that has not happened yet.

Phil:

Does his code overlap with Alexander's or are they complementary?

Maxim:

I didn't have a chance to check the Alexander's code, but from the descriptions they probably are complementary.

Phil:

We also need to understand the risks of anything that has to do with changes to the X11 compatibility mode, we've had some discussions about how we're going to test all this.

We need to be sure that whatever we put in there is not going to break anything else.

But the clock is ticking, because RDP1, feature freeze is June 8, so we have 6 weeks.

Victor:

We'll need a CSR to carefully review, it will also require Joe Darcy to jump on it.

Alexander:

There will be a separate PR for changes: code changes and spec changes to make it easier to review(dependent on each other). I'll post a head-up message to wakefield mailing list.

Niels:

About libE1, progress is still being made, and we are getting closer to something stable and ready to merge . One of the latest protocol changes , to not depend on protobuf anymore.

We are still planning to make it Gnome 45 and Fedora 39.

Olivier:

For libE1 we are expecting last few API tweaks, obviously we want to do all that before we declare the API stable.

Niels:

How is the pure Wayland stuff going on? I understand that it is not going into JDK21

Maxim:

There was no significant progress in the last few months.

Kevin:

*The spec changes that are going into JDK21 should be the same as needed as for the pure Wayland port. There could be some additional things needing to relax.*

Phil:

*Regarding TCK, there are lots of discussions about how we actually adequately test this from the functional perspective on different distros.*

Alexander:

*and we still have in issue with Oracle Linux 9.1 that xdg-desktop-portal-gnome is too old, so it keeps asking the user permission for every screenshot.*

Niels:

*9.2 going to be released in couple of weeks, normally you could try it on CentOS and it should work on RHEL as well.*

## Online Zoom Meeting 9am PDT 30th Mar 2023

Attendees: Maxim Kartashev, Alexey Ushakov, Alexander Zvegintsev, Victor D'yakov, Kevin Rushforth, Olivier Fourdan, Jonas Adahl

Alexander:

*xdg-desktop-portal-gnome package was missing in 9.1, but was present in 9.0.*

*It now comes with 9.1 only with fresh install, system update of an old 9.1 installation didn't work for me, have to install it manually.*

*While the system now successfully reports version 4 of org.freedesktop.portal.ScreenCast. It is minimum required for restore\_token functionality.*

*xdg-desktop-portal-gnome package comes outdated for both 9.0 and 9.1 (latest available 41.2), to be able make screenshot without confirmation for every attempt we need at least 42+ version..*

*Added handling of negative scenarios when the user partially or completely denies access to screen data.*

*Now we throw a security exception only if the user denied screen capture at all.*

*If user is allowed to access only some displays, image data of others will be displayed in black, no exceptions or warnings will be thrown.*

*If user wants to change previously made decision, newly added Robot.resetRestoreToken() method can be used.*

*Changed the storage location of restore\_token, now instead of a simple file, we use Preferences API.*

*It's still a file, but now it's a standardized XML file, still storing the token in plaintext.*

*I don't think this is a problem though.*

*I also checked probably the most popular product using ScreenCast API - OBS, Open Broadcast Software:*

*They also use store this token in plaintext in their configuration files.*

*I also checked how the prototype behaves with Plasma Desktop Session.*

*When I get the screen data, everything goes without errors, but the data itself is black.*

*Apparently, it's because I have an Nvidia graphics card.*

*Another interesting thing is that in response to a call to the Start method, I get a stream that has size, but no position.*

*This can be a problem, for example when we have two displays with the same resolution.*

*It is not clear how to distinguish the stream of one screen from the other and thus get the data from the requested display.*

Victor:

*We have plans to make it, including introduction of header files, in JDK 21 LTS.*

*We will know more on our next meeting in 4 weeks from now.*

*We are usually not making any too risky changes 1-2 weeks before the fork(June 8th), we have only just two months for all this work.*

*It is not our ultimate goal, but a good opportunity.*

Alexey:

*We have just begun to continue our work on pure Wayland prototype.*

*Nikita Gubarkov has some important changes that could be useful for compatibility mode,*

*he was able to support HiDPI for different monitors using access for underlying wayland subsystem.*

*He said he will be able to port this changes, and probably, to include them into JDK 21 LTS release.*

*It is already integrated in our runtime shipped with our products, looks like all the issues resolved so it is time to move to OpenJDK*

Victor:

*If this will requires CSR, please do not submit it right before the fork, give it an extra 1-2 weeks.*

Maxim:

*We could implement some limited support for screenshots for pure wayland.*

*General idea: as long as we can make at least some tests run just a single window, and don't rely on their absolute position,*

*but calculate their offsets just from the top left corner of just one window, we can implement robot color picking, and even screenshoting functionality for this window,*

*by grabbing pixels from associated Wayland buffer.*

*This will work for some tests, especially performance measuring will be more accurate(e.g. color picking at some points frequently).*

*Alexander, can we get recent changes for taking screenshots with ScreenCast API, before they reach JDK 21?*

Alexander:

*pipewire headers requires legal approval before placing them at openjdk github, but I can make a version to use headers files stored in system and share it.*

## Online Zoom Meeting 9am PDT 2nd Mar 2023

Attendees: Alexey Ushakov, Alexander Zvegintsev, Victor D'yakov, Kevin Rushforth, Phil Race, Olivier Fourdan, Jonas Adahl

Alexander :

Pushed the stable version of screencast code to the sand box

Fix for when focus is outside of xserver window

2 places where window focus transfer does not happen and popup remains open

Tried several 3rd party apps and they behave the same so this OK (is it ? - phil)

Working on build changes to build OpenJDK using screencast on OL 7

Have dynamic loading of functions working. Still relying on the pipewire and spa header files.

On RHEL/OL 8 it does not work when try to run screenshot -outdated pipewire

missing xdg portal support.

So probably can't support such older releases as they will never get all the fixes needed.

Last time when trying to take screenshot on RHEL/OL 9 found does not have xdg portal -

Jonas : said xdg portal gnome backend was there in RHEL 9.0 but lost in 9.1

but this was fixed (9.2) ? Alexander would have to check what version he was using

but this is probably the issue

Alexander: we should talk to jreg folks about not cleaning out various desktop related variables from the environment that become more important for wakefield

(DBUS\_SESSION\_BUS\_ADDRESS, XDG\_SESSION\_TYPE and WAYLAND\_DISPLAY)

Phil: agreed but our CI system cleans some of these out too, before jtreg even has a chance.  
So multiple things to address.

Jonas : how important is it to support restore tokens in persist mode on RHEL9.x/OL9.x

Phil: This is the support to allow permissions to do screen capture to persist across sessions.  
It is very much needed for our automated CI testing systems which reboot after every test task.  
Would be hard to support a release where it isn't possible.

RHEL bug about backport restore token portal support: [https://bugzilla.redhat.com/show\\_bug.cgi?id=2174946](https://bugzilla.redhat.com/show_bug.cgi?id=2174946)

On Ubuntu 22.04 have pipewire-0.3 and spa-0.2 - are these API stable ?

Jonas (or was it Olivier?) replied pipewire was api stable as of v 0.3.1.X. So probably fine.

Phil:

The build work Alexander is doing is important so that we can build on our CI systems  
Otherwise a blocker to integrating even xwayland compatibility support into wayland  
We build one binary for many distros and versions .. so there's a LCD requirement.

Alexey :

Talked with Nikita Gubarkov who has been working on hidpi on wayland  
(see posts on list at <https://mail.openjdk.org/pipermail/wakefield-dev/2023-February/000081.html>)

Where should he push this ?

(1) Probably into its own branch at <https://github.com/openjdk/wakefield>

but

(2) he isn't a project committer so you will need to do it for him

Maxim will hopefully be back soon and will pick up where he left off on wakefield work.

## Online Zoom Meeting 9am PDT 2nd Feb 2023

Attendees: Alexey Ushakov, Alexander Zvegintsev, Niels de Graef, Victor D'yakov , Laurent Bourges, Kevin Rushforth, Phil Race, Olivier Fourdan, Jonas Adahl

Alexander:

Updated and slightly redesigned the [Known problems and solutions](#).

The ScreenCast prototype has been redesigned to open/close screencast session during a single call for each screenshot

Some of these changes are already pushed, some will be soon.

It also runs faster than the previous solution.

Alexey:

Did you tried to run some tricky regression tests that require a quick screenshot between some states of controls?

Alexander:

I tried a bunch(they are fine), but haven't run all.

However, some tests that require you to take a screenshot of an icon in the system tray area may fail because an orange"screen is being shared" icon appears while you are taking the screenshot.

Some questions about full support for keyboard/mouse event emulation:

1. What is the status of libEI,
2. The proposed RemoteDesktop solution doesn't provide a concept for retaining user permission between sessions, such as `restore_token` in ScreenCast. When can we expect this to be addressed?
3. Which solution will become usable for our needs first?

Olivier:

There is active work on libEI, its protocol is being actively developed. At current stage it is already works.

Once it finalized and stabilised the follow up work will be to support it on Mutter, xdg-desktop-portal, xdg-desktop-portal-gnome, libportal and libwayland as well.

libEI is planned to ship with Gnome 45(comes with Fedora 39).

Jonas:

[xdg-desktop-portal/issues/850](#) priority can be bumped. At the earliest it can be shipped with Gnome 44(comes with Fedora 38) if it will be fixed soon.

It should be much easier comparing to libE1.

Alexey:

There is a slowdown on pure Wayland port work. More engineers are expected to join this work. Hopefully Maxim will be back this spring.

Phil:

X11 compatibility mode, what are the blockers?

Alexander:

We don't have any.

All of them can be worked around by changing the documentation to reflect the current behavior, modified/excluded tests, even some crashes can be avoided by modifying tests.

Olivier:

[xorg/xserver/-/issues/1222](#) posted a possible fix some time ago, but didn't get any traction

Phil:

All documentation changes should go to the OpenJDK 21 LTS release, even if the work on X11 compatibility mode will not complete by this time. This will allow us to do backports later.

Alexey:

We were able to fix some issues with HiDPI in X11 compatibility mode using some Wayland api besides X11, so probably we will be able to provide our solution.

## Online Zoom Meeting 9am PDT 30th June 2022

Attendees: Alexey Ushakov, Olivier Fourdan, Niels de Graef, Jonas Adahl, Kevin Rushforth, Victor Dyakov, Alexander Zvegintsev

Alexander:

Played with [gdbus-org.freedesktop.portal.RemoteDesktop](#) API, it looks suitable for mouse/keyboard control for AWT Robot.

Unfortunately it does not support session restoration like [gdbus-org.freedesktop.portal.ScreenCast](#) do.

Without it user has to provide "Allow remote interaction" confirmation each time the application is launched..

Is there any plans to add ability to restore session for RemoteDesktop API?

If so, will it come earlier than libE1?

Jonas:

There are such plans, furthermore, RemoteDesktop API will use libE1.

So we can either use RemoteDesktop API or libei directly at some point.

However dates are not clear yet.

Alexey:

A few crashes have been fixed in pure Wayland prototype, and text rendering support has been added.

Software rendering performance is quite good: it is close to 60 FPS( display refresh rate), but it can be worse on a less powerful hardware.

It looks like we can proceed without implementing hardware accelerated pipelines for now.

SwingSet2 works fine, no glitches observed, however input events are not implemented, so there is no way to control the app yet.

We do not expect any several issues with implementing input support.

## Online Zoom Meeting 9am PDT 5th May 2022

Attendees: Phil, Kevin, Alexander Zvegintsev, Victor, Zhengyu Gu, Niels de Graef, Maxim Kartashev, Jonas Adahl, Olivier Fourdan, Mario Torres

Alexander : Ubuntu 2204 - no difference from previous releases

Looking at bug where mouse exit event not coming but works fine on native wayland app

Maxim : Nikita has done some work so fonts are being rendered although without hints and AA and sub-pixel but basic rendering working

Also (Maxim) made a preliminary prototype of how we want to start testing pure wayland - custom extension ..

He wasn't able to merge into openjdk repo [Kevin : Maxim missed a step in his project acceptances]

Code available through JB runtime repo

<https://github.com/JetBrains/JetBrainsRuntime/tree/wayland-dev/src/java.desktop/share/native/libwakefield>

Maxim gave a quick demo of test framework

Lanuch a custom weston instance with Maxim's plugin so you aren't updating the running

desktop so no admin permissions required

The plugin gives access to weston internals -

provides wayland protocols which could be implemented for some other server

<https://github.com/mkartashev/wakefield/blob/main/protocol/wakefield.xml>

location of surface in abs coords

set location ..

grab portion of screen

get pixel color

would like someone to code review - Java side OK but plugin code is less obvious

Why doesn't work it with weston 10 when it works with weston 9 .. once you get into accessing internals you really need to build against the exact version of weston you are targetting.

This testing effort is intended to be a stop-gap until we have a proper solution to running tests on the wayland desktop - not imagined as a solution we can deliver as part of OpenJDK

## Online Zoom Meeting 9am PDT 7th April 2022



Attendees: Alexey Ushakov, Olivier Fourdan, Niels de Graef, Jonas Adhal, Sergey Bylokohov, Alexander Zvegintsev, Zhengyu Gu, Phil Race, Kevin Rushforth, Victor Dyakov

Alexey :

Working on pure wayland, can show AWT and Swing frames with some simple rendering in paintComponent  
Nikita working on font rendering. Will push to branch on openJDK when ready.

3rd thing is AWT robot - Maxim is providing custom extension to Weston instance to get it working.

Alexander : does extension cover screen shot + key press : yes - works even for multimon ...

this is useful for testing but not for production

Sergey : how difficult to install this extension ?

Alexander: Reported a bug about XOR rendering but it could be a driver bug ?

<https://gitlab.freedesktop.org/xorg/xserver/-/issues/1333>

Screencast still not usable because have to configure it manually every time you start a session

So weston extension looks more promising (for testing)

Need some scripting support

Jonas: maybe not running in a sandbox or something - should be possible to make it remember what you selected in a dialog.

Maybe be missing a way for portal to associate request with application ID

Screencasting portal configures a token that should make it possible

Screenshot portal doesn't remember that `_yet_` though

You can follow this issue for the latter: <https://github.com/flatpak/xdg-desktop-portal/issues/649>

Niels De Graef : Note that a wl extension protocol that disables security features like that probably won't fly in any compositor

( Jonas & Niels?) : AI: Are there any particular JDK requirements around input methods - there is a slow moving project to update in update flawed gtk3 4 protocol .. Phil will have to get back on that after asking around.

## Online Zoom Meeting 9am PST 10th Mar 2022

Attendees : Maxim, Olivier Jonas, Alexey, Niels De Graef, Phil, Alexander, Kevin, Victor

Alexander : has been doing testing of current JDK on Oracle Linux 8.5  
This uses gnome 3.32 - old gnome shell - so not getting recent fixes  
So just a point of reference

Getting some new failures (tests that passed before ) on Ubuntu 21.04 and 21.10 - likely this is due to updated OpenGL drivers

Some JDK screen shot tests failing but need to implement something so not an unexpected problem

Olivier : update on Drag and Drop - the upstream fixes have landed in gnome 42 and 40

Niels :

[X.org](#) not likely to be removed any time soon by any distro as most are only just switching to wayland as the default. So we have time. Perhaps some cutting edge distro will be first but not soon.

Alexey : Up date on pure wayland work - trying to support just one AWT button but needed to bring in whole AWT hierarchy - meaning a lot of the equivalent of the Xawt toolkit.

Kind of working but faced with problem of placing location of windows on desktop  
can only provide coords for popup surfaces We've talked before about this being part of the design philosophy of wayland  
Surely this is needed even for applications that try to place their windows relative to each other ?

Jonas: this has come up in discussions about whether it makes sense ..  
GIMP might need this ? Well maybe not GIMP because it is moving away from multiple images but some medical applications have a need .. but it may be more like session management  
1st time launch rearrange your windows. then the extension protocol can be used to save this.  
[https://gitlab.freedesktop.org/wayland/wayland-protocols/-/merge\\_requests/18](https://gitlab.freedesktop.org/wayland/wayland-protocols/-/merge_requests/18)  
[ Editor : I'm not sure if this would work for JDK needs and how would we expose that protocol to apps ?  
It can't just be "java" as that's the platform not the application.]

Alexey is focused on the rendering side. Dmitry Batrak will be looking at event handling.  
And Maxim is thinking about Robot - prototyping some new approach to automated testing ... run tests in a weston instance on top of X11 ?  
Only useful for testing. Using a custom protocol extension.

Peter Hutterer working on libei and has made a 0.2 release Jonas working on it too.  
Concentrating on 2 parts : capturing events and other is sending input events

feature where input captured from compositor - like Synergy which forked into Barrier and then again  
That project is to share input devices across multiple computers mouse events go to the other computer when the mouse goes off edge of screen and emulate events on other computer  
API mostly written down but not yet stable.  
Peter Hutterer made a 0.2 pre-release.

Blog here :

<http://who-t.blogspot.com/2022/03/libei-adding-support-for-passive.html>

In the world of sandboxing and flatpak not guaranteed that they have same version of a library.  
So not going to make it into any distro even as a pre-release until API and wire format are stable.

## Online Zoom Meeting 9am PST 10th Feb 2022

Attendees: Kevin, Victor, Olivier, Mario, Aleksandr, Alexey Ushakov, Maxim Kartashev, Sergey, Zdenek, Zhengyu, Dalibor

Discussion:

We went through all of the open issues on the Wiki. Aleksandr will update the bugs with the latest status. The suggestion was made to group them by their status into 2 or three groups (unresolved, fix in progress, resolved). For the unresolved issues, they could be prioritized.

1. [JDK-8280982](#) : java.awt.Robot taking screenshots

Needs to be solved by the Portal team. Not limited to Wayland. This is the number one issue in terms of importance. There is a bug filed against Portal in the flatpak project.

We spent some time discussing the current limitations. What we ultimately need is something that only needs to be configured once by an administrator, and will not produce a popup. Another limitation that needs to be overcome is that the current workarounds only capture the entire screen, which is not sufficient for an API that is often used to read small regions (or a single pixel). This will be too slow. Another idea was discussed to throw a security exception in the screen capture API, but that would require a spec change. It's also not what Java2D does on other platforms (e.g., Mac) when the proper permissions haven't been granted.

2. [JDK-8280983](#) : java.awt.Robot emulating keyboard/mouse events

No new information. Solving this using libEI will be best, and will also solve one of the below issues, but there is no time frame as to when we might expect a fix.

Additionally, using libEI should fix following:

[JDK-8280995](#) : X11 compatibility: Robot.mouseMove does not visually move mouse cursor

[JDK-8280990](#) : X11 compatibility: XTest emulated mouse click does not bring window to front.

[JDK-8280988](#) : X11 compatibility: Click on title to request focus test failures

The last of the above two can be worked around by test changes, but using libEI is a better solution.

3. [JDK-8280993](#) X11 compatibility: Popup menu dismiss in X11 compatibility mode.

This one should be fixable.

4. [JDK-8280991](#) X11 compatibility: fake configure event for XRandR emulation

Should already be fixed in latest version

5. [JDK-8280985](#) X11 compatibility: Wayland crash on huge window

Already fixed

6. [JDK-8280987](#) X11 compatibility: crash when mapping a lot of windows

There are two issues here: one is that we are running out of file descriptors, the other is that this will fill up the Wayland buffers. The first is a simple fix (In progress). The second fix in Wayland itself, not XWayland. It's also in progress, but is a complex fix.

7. [JDK-8280992](#) X11 compatibility: Custom colored cursor might be displayed in black and white

Aleksandr will file a bug if he can come up with a simple reproducer

There are additional issues specific to pure Wayland mode that still need to be filed (this needs more discussion).

## Online Zoom Meeting 9am PST 27th Jan 2022

Attendees : Olivier, Phil, Kevin, Aleksandr,, Victor, Niels De Graaf, Alexey Ushakov, Jonas, Sergey, Zdenek, Zhengyu

Aleksandr :

Ubuntu 20.04 - some test problems - tests passed on 20.10 and 21.04

XOR

Memory leak tests - need to increase xMax heap

Mouse enter/exit events

Not always clear if Java problem or Wayland

Let's not spend/waste time on older OSes even LTS versions they'll never have all the support we need anyway.

Working on migrating bugs / issues just documented on internal wiki to JBS using release==internal

Olivier : DnD also fixed in new OSes

Xwayland is not involved in DnD or copy/paste - standard x atoms

Wayland compositor or applications - x app knows nothing about wayland native so it is the wayland compositor which does the translation

So doing DnD mutter has to map X11 window and use that to translate

Upstream Gnome 42 fix not yet backported to gnome 41 used by Fedora 35 ..

<https://gitlab.gnome.org/GNOME/mutter/-/issues/2042>

[https://gitlab.gnome.org/GNOME/mutter/-/merge\\_requests/2259](https://gitlab.gnome.org/GNOME/mutter/-/merge_requests/2259)

Can you use latest mutter from main on older OS ? Olivier or Jonas (?) - possible but sounded like a bunch of steps he will find docs on this

Ed : I think this is the doc meant : <https://gitlab.gnome.org/exalm/jhbuild-steps/-/wikis/JHBuild-on-Fedora>

Alexey : trying native wayland port rendering using J2D s/w rendering.

Need to fit wayland model to our surface architecture ? Will create a branch in project repo.

Also (A) will update wiki with info on this

Niels : updated wiki to more clearly describe JDK issues where there are gaps in wayland support

Do AWT Robot docs offer a "way out".

<https://docs.oracle.com/en/java/javase/17/docs/api/java.desktop/java/awt/Robot.html>

Note that some platforms require special privileges or extensions to access low-level input control. If the current platform configuration does not allow input control, an `AWTException` will be thrown when trying to construct Robot objects. For example, X-Window systems will throw the exception if the XTEST 2.2 standard extension is not supported (or not enabled) by the X server.

Applications that use Robot for purposes other than self-testing should handle these error conditions gracefully.

These words are not intended to say you can be conformant - more about advice that you may need to update settings add packages to mke your desktop supported. Analogies such as makung sure you have required X11 packages installed and headful support is available.

Zdenek : Fedora rawhide - dev version of Fedora may be a way of getting VERY latest ahead of a Fedora release

## Online Zoom Meeting 9am PST 13th Jan 2022

Attendees : Olivier, Phil, Kevin, Aleksandr, Dalibor, Victor, Niels De Graaf, Alexey Ushakov, Jonas, Mario

he topic of the day was a discussion around Alexey's email to the mailing list ..

<https://mail.openjdk.java.net/pipermail/wakefield-dev/2021-December/000025.html>

proposing what we might do for the "full" wayland native port and the various points it raised

Note: Alexey will add a fleshed out version on the project wiki.

The proposal suggests we'd code to the low level wayland APIs rather than using a high level toolkit (ie GTK4)

As previously observed we should explore both. Probably we could get bootstrapped a lot faster with GTK4 and it would give flexibility for backends so we'd probably not need to create an OGL and Vulkan backend .. if Vulkan was an option on that distro then we'd expect the underlying Cairo (?) based renderer to be using it or have it as an option.

Likely there are pros and cons to both options. GTK4 might make assumptions or so things in a way that isn't compatible with some other functionality. A low-level port might mean a lot of re-inventing the wheel and having to use functionality that is distro-specific but would have been hidden / abstracted by the platform GTK 4.

So due diligence on both options, and not expecting completely smooth sailing for either. Ease of implementation and maintenance and portability are strong arguments, but so are flexibility and internal consistency in behaviour and behavioural compatibility with the existing X11 implementation. So if doing it "right" is more work, we'll have to absorb that, and we want a stable base so we aren't in a perpetual maintenance game.

Mario observed that we need to not get ahead of ourselves because we still have the list of technical challenges that need to be resolved that apply to X compat mode and to a native port too.

We'll work on those but whilst they work through the eco-system over 2 years (?) we could be doing something useful ..

One element of the proposal not directly related to wayland is we could merge EDT and Toolkit thread since we no longer need these separate for applets. Possibly, but we kick off a new thread for modal dialogs too .. could this be special cased ? Might depend on whether you can limit what happens in such a case but I'm not sure you can.

We do have these documented on the wiki but it needs another pass to have a consistent format

- what the problem is
- what JDK can do
- what we need from the wayland ecosystem
- how much of a blocker it is
- can we "tweak" the spec - at the cost of making backports harder

We should look at this soon so we can get started on spec tweaks we think may be unavoidable but acceptable ..

Eg What's acceptable for screenshots ? We can't deprecate but can we make something optional ?

Alex Z: Is the SplashScreen API supportable ?

Apparently splash screens display in top lefe (the window positioning problem)

A splash screen (output only) surface role could probably be proposed upstream. it'd just need a fallback implementation for compositors that doesn't support it, e.g. a dialog

Maybe similar to this PiP request : [https://gitlab.freedesktop.org/wayland/wayland-protocols/-/merge\\_requests/132](https://gitlab.freedesktop.org/wayland/wayland-protocols/-/merge_requests/132)

For more info on roles: <https://wayland-book.com/surfaces/roles.html>

Add input to this existing issue : <https://gitlab.freedesktop.org/wayland/wayland-protocols/-/issues/67>

Accessibility ?

Is there a library ? Like ATK ?

There are existing DBUS based A11Y APIs ..

For the accessibility protocol Jonas is talking about: <https://www.freedesktop.org/wiki/Accessibility/AT-SPI2/>

For a quick overview of what Emmanuele did for accessibility in GTK4: <https://blog.gtk.org/2020/10/21/accessibility-in-gtk-4/>

## Online Zoom Meeting 9am PST 16th Dec 2021

Attendees : Olivier, Phil, Sergey B., Kevin, Aleksandr, Dalibor, Victor, Niels De Graaf, Alexey Ushakov

Previously discussed upstream DnD bug is fixed upstream - maybe backported to gnome 41 branch ?

On Ubuntu 21.04 using JDK in Xwayland compat. mode DnD from one java app to another (eg) JEdit gets dropped on native terminal window instead.

DnD is handled by the wayland compositor acting as proxy - Olivier has seen similar issues if you have wayland native under X11 then it hits the window underneath this should be fixed if you try Fedora 35 - Ubuntu 21.04 slightly older and presumably doesn't have this fix.

Sergey asked Aleksandr if his DnD testing was using the Robot API or by hand and whether it is even theoretically possible if wayland doesn't report global screen coords for a window so you can't move the mouse to a screen position of another windows.  
i.e can't move mouse from window 1 -> windows 2 because don't know where window 2 is ...

In other words we have wiggle room in the APIs - and code in the implementation - such that SetBounds calls for a top-level might be ignored or adjusted, but we rely on correct answers from GetBounds

Olivier : this should actually be OK because in compat mode the apps are talking to an [X.org](#) server albeit one acting as a bridge (ie Xwayland)

Some similar discussion where you just want to move the focus from Button A in Window 1, to Button B in Window 2 and you need to know the position

In wayland you could set the position by saying loc (x1,y1) in Window 2 even if you don't know the real position of Window 2

Not clear (to me) what this would look like for DnD as you want to move the mouse and see it cross the screen outside window boundaries .. (some of the above needs verifying - outcome of the discussion was not 100% clear to me)

Screencast API is used for automated testing by wayland devs

Would we need to provide new APIs ? And respecify so that existing APIs are "optional".

Could get confusing because other platforms will not have the same issue or model.

Assumes that xwayland compat mode would not need this respecifying to give us backport options.

All too early to know.

Noted that GTK4 was re-specified in a similar way so that things that don't work on wayland are no longer part of the API

How is multi-mon handled in this scenario where you can't position/get position ? Full screen too ?

Full screen is a special case, and you may be able to specify the monitor for a window but that's about it.

Sergey : How does it work for custom decoration on pure wayland

Olivier: Some history here expected client-side decorations was always the answer - so custom decorations would naturally fall from that.

But KDE folks not happy - wanted server side decorations for consistency on the desktop so there is an optional protocol for a wayland compositor but the client needs to be able to fall back to client-side in case that protocol is not supported on a compositor.

Is there a deadline to ship xwayland compat JDK ?

None yet - still working through these issues - a deadline might be when (say) RHEL shipped without [x.org](#) .. which would be a real problem if we didn't have all the answers ready.

## Online Zoom Meeting 9am PST 2nd Dec 2021

Some progress on issues previously seen - some upstream bugs filed and/or fixed.

As reported on the mailing list the DnD issue previously discussed has been fixed upstream

Mapping a single large window : 2 bugs : 1 in wayland 1 in xwayland

Problems with creating a large number of windows is because wayland has fixed buffer sizes for the number of mapped windows

Four, 4K buffers for file descriptors, other requests .. they can fill up

Variable buffer sizes is tricky and a proposed patch is under discussion but not yet accepted.

Some of these problems do not affect running native directly on the hardware with Glimmer and DRM available but only affect running in a VM where shared memory is used instead.

## Online Zoom Meeting 9am PST 18th Nov 2021

We went through some of the issues raised by individuals on the call

Wayland does not provide access to the absolute coordinates of a window on a desktop or allow specifying it. Whereas X window managers don't make guarantees about positioning but generally do honour a request and do allow querying. We may have some work to do in making things work as well as possible with the different approach of wayland  
There's (some) discussion here : <https://gitlab.freedesktop.org/wayland/wayland-protocols/-/issues/72>

There was discussion of ideas such as creating an invisible window that spanned the entire screen and creating other windows as children of that but this may be fighting another battle where child windows are assumed to be transient windows.

Focus issues .. and bringing a window to the front or sending it to the back ?  
Maybe a way to bring a window to the front with focus but not sending to the back  
XDG activation protocol can ask for attention and that usually means being raised to front  
<https://gitlab.freedesktop.org/wayland/wayland-protocols/-/blob/main/staging/xdg-activation/xdg-activation-v1.xml>

Drag and Drop bugs - Alexander having some trouble on Fedora 35

Solutions to some of the challenges are still evolving and different distros may provide different solutions.  
There was a discussion and agreement about the importance of implementing to a level that abstracts away platform-specific code. Coding to something that depends on a particular library that a platform may not choose to deliver makes our task harder and nearly impossible to deliver one OpenJDK binary that could run on multiple distros.

All (or at least most) of this is still applicable to both the xwayland case and the wayland case.  
Focus for now continues to be on the former. Too early to say if GTK4 or lower-level approach will ultimately win out for OpenJDK needs.  
It was observed that GTK and its print dialog drive you to xprint, which may not be what we want.  
Not yet looked into what printing will look like in a native wayland port.

## Online Zoom Meeting 9am PDT 16th Sept 2021

Attending : Wakefield committers from Oracle, Redhat, JetBrains and Amazon as well as Wayland developers from Redhat who were invited by Mario.

The intent of this meeting was to have a productive dialog between the OpenJDK developers and Wayland community developers to come to a better shared understanding of the technical challenges and current state of Wayland and identify avenues of investigation.

Some opening remarks were about the Wakefield infrastructure, including this wiki, the Wakefield mailing list and the Wakefield repo.

Now they are up and running we expect to use the mailing list for most communications and document progress etc on this wiki page.

Branches should be created in the Wakefield repo for all work / experiments etc. Don't use the master branch.

In other words, we should use the project infrastructure for everything and publish contributions and ideas there.

The major topic of conversation was around the options for JDK for capturing screenshots and synthesising input events - both for running in X11 compatibility mode or as a native Wayland client

Screen capture is done today by calling a GTK API which is no longer exposed in GTK 4 because core wayland doesn't support it.

We need an alternative, (JDK has a pure X11 fall back based on xwd but we didn't touch on that - it has the same issue)

A few things were thrown out that might merit investigation

<https://flatpak.github.io/xdg-desktop-portal/portal-docs.html#gdbus-org.freedesktop.portal.Screenshot>

<https://docs.flatpak.org/en/latest/portals-gtk.html>

<https://github.com/flatpak/xdg-desktop-portal#using-portals>

<https://github.com/flatpak/libportal>

The libportal abstraction / helper layer is probably what we'd want.

It is unclear (to me the note taker) if this is readily available on distros alongside wayland - we can't ship it ourselves and it likely needs more than a client-side library anyway

JDK for X11 today needs the functionality of the XTEST X11 extension protocol to support the input event side of things.

The wayland desktop answer for this is probably going to be <https://gitlab.freedesktop.org/libinput/libei> but it is still in development.

It is very unlikely that distros will be ready to ship all the pieces we need in the next 12 months. So the "short term" goal may actually need to wait for somewhat longer than that.

What is the API the clients used for rendering to the wayland client off-screen surface ? GTK defaults to OpenGL (EGL) and falls back to shared memory (ie some software rendering and a copy)

There is also an experimental Vulkan backend.

We use Xrender as the default Linux rendering pipeline today. Maybe we could use OpenGL for the X.org desktop too ? Yes, it is possible that we end up making improvements to the XGL pipeline as a consequence of adding EGL support for Wayland so that we could make OpenGL the default there (X.org) too. But it would be a side-benefit not the main goal.

What are people using for window decoration since the wayland model is that the client does the decoration ? I didn't catch all the options here seems there are a few

There was also a question about hidpi causing fuzziness of X11 apps especially for fractional scaling.

The wayland devs confirmed it is a known issue but difficult to solve without making X11 apps far too small to be legible.

The wayland devs input seems to be essential to us navigating a very complex and evolving landscape.

We agreed we need to go off and study some of these APIs and come back with follow up questions when ready.