Subject: U++ GTK Wayland Port
Posted by Klugier on Sun, 15 Sep 2024 21:33:28 GMT
View Forum Message <> Reply to Message

Hello,

I would like to announce that I am currently working on porting U++ to Wayland using our GTK backend. I did some progress in this area:
- SSD (Server Side Decoration) variant works just fine with minor issues. Please, keep in mind that SSD is not supported on many distros, currently KDE support it properly
- CSD (Client Side Decoration) variant works, but there are more problems there in comparison with SSD variant
- In general the responsiveness of the app is much higher in comparison to X11

However, there are some problems as well:
- When backing to previous window after closing dialog window, focus is not being properly restored with CSD
- Menubar positioning is not optimal for example when you spawn menu near the right ennd of the sceen it will be rendered outside of the screen instead of inside
- Second menubar on Gnome doesn't work for some reasons - CSD variant
- Drag and drop needs to be properly implemented on both CSD and SSD
- OpenGL doesn't work (This is out of scope for initial release)
- Window positioning is not optimal. However, it depends on the compositor since it is very powerful in this area for example in comparison with X11
- Probably more...

I attached screenshot from Ubuntu 24.04, where U++ is running in Wayland session and windowing is done using Wayland, not X11:

If you want play with it you can try my branch klugier/gtk-wayland-initial-version in main u++ repo. Also, once running, please make sure that CtrlCore/GtkApp.cpp file has following backend initialization order:

```
#if GTK_CHECK_VERSION(3, 10, 0)
 gdk_set_allowed_backends("wayland,x11"); // this fixes some wayland issues
#endif
```

Alternatively, you can add WAYLAND compilation flag and it should force prioritizing Wayland backend over X11.

Klugier

## File Attachments

1) WaylandCSDDemo.png, downloaded 334 times

Subject: Re: U++ GTK Wayland Port
Posted by Tom1 on Mon, 16 Sep 2024 08:31:55 GMT
View Forum Message <> Reply to Message

Hi Klugier,

I'm pleased to see that you're working on this subject! Keep up the good work!

Best regards,

Tom

Subject: Re: U++ GTK Wayland Port
Posted by Oblivion on Tue, 17 Sep 2024 20:05:13 GMT
View Forum Message <> Reply to Message

Hello Klugier,

Great and much needed work. Thank you for your efforts!

I am willing to test it and help, but unfortunately I cant get it compiled. The compiler gives the following errors:

unknown type name 'XDisplay'

initialize return object of type 'int *' with an rvalue of type 'Display *' (aka '_XDisplay *')
 (): 31 |  return  G DK_DISPLAY_XDISPLAY(gdk_display_get_default()) ;
/usr/include/gtk-3.0/gdk/x11/gdkx11display.h (59): note: expanded from macro 'GDK_DISPLAY_XDISPLAY'
 (): 59 |  #define GDK_DISPLAY_XDISPLAY(display)  ( gdk_x11_display_get_xdisplay (display))

Any ideas why I'm getting X errors? (I am on wayland, on up-to-date archlinux with the latest gtk/gnome)

Do I have to manually set any flags?

Best regards,
Oblivion

Subject: Re: U++ GTK Wayland Port
Posted by Klugier on Tue, 17 Sep 2024 20:17:47 GMT
View Forum Message <> Reply to Message

Hello Oblivion,

I have this compilation error from time to. I think it is somehow related to BLITZ. Just change (CtrlCore/GtkX11Util.cpp - line 29):

```
XDisplay *Xdisplay()
{
 return GDK_DISPLAY_XDISPLAY(gdk_display_get_default());
}
```

to

```
XDisplay *Xdisplay()
{
 return NULL;
}
```

If you will have again this error, just back to the previous version, so I usually comment and recomment

```
 return NULL;//GDK_DISPLAY_XDISPLAY(gdk_display_get_default());
```

I know that it sound stupid, but I didn't have time to investigate BLITZ problem. I hope it will help. In case of any error or success, please let me know.

Please keep in mind that even if you are on Wayland, X11 modules for GTK are still there. So, you can call both X11 and Wayland functions.

Klugier

Subject: Re: U++ GTK Wayland Port
Posted by Oblivion on Tue, 17 Sep 2024 20:41:57 GMT
View Forum Message <> Reply to Message

Yep, that worked, thanks!

Now I can start testing.

First issue I've noticed:

I've noticed that some elements (in this case a SplitterFrame) do not paint its background and the whole background becomes transparent (actually, if this can be utilized in the final form, it would be very cool to have transparent or blurred background on some apps :) )

Screenshot:

Best regards,
Oblivion

File Attachments
1) Ekran Görüntüsü - 2024-09-17 23-38-48.png , downloaded 282 times

---

Subject: Re: U++ GTK Wayland Port
Posted by Oblivion on Thu, 19 Sep 2024 20:37:54 GMT
View Forum Message <> Reply to Message

Hello Klugier,

The below approach you've recommended didn't work:

return NULL;//GDK_DISPLAY_XDISPLAY(gdk_display_get_default());

However, undefining the preprocessor flag at the top level (at the application level, before including U++ headers) did:

#undef GDK_WINDOWING_X11

This way, I've even successfully compiled and run Bobcat on Wayland backend with no apparent issues.

Best regards,
Oblivion

---

Subject: Re: U++ GTK Wayland Port
Posted by Klugier on Sun, 22 Sep 2024 13:05:29 GMT
View Forum Message <> Reply to Message

Hello Oblivion,

Thanks for letting me know that Wayland backend cooperates with Bobcat. For the random compilation issue, I managed to fix it by replacing XDisplay with _XDisplay. So, you shouldn't observe the issue anymore on the test branch. Undefing GDK_WINDOWING_X11 is bad idea in this case. We would like to have X11 backed alongiste Wayland one in single application. Thanks to that the application will be able to work in both X11 and Wayland and the compilation for concrete display won't be needed.

BTW, The branch name was changed to klugier/gtk-wayland-initial-version. So, if you encourage any pull issues, just switch to this particular branch.

Klugier

---

Subject: Re: U++ GTK Wayland Port
Posted by Oblivion on Sun, 22 Sep 2024 18:05:32 GMT
View Forum Message <> Reply to Message

Hello Klugier,

I've checked the latest version and it works fine. There are some small issues, which I'll report back (or come up with pull requests) to you in the following days.

Thank you for your efforts.

Best regards,
Oblivion

---