COLLABORA

Cluttering WebKitGTK+

Gustavo Noronha
gustavo.noronha@collabora.co.uk

Last year…

WebKit Clutter

Built for an R&D IVI platform using Clutter

Heavily based on WebKitGTK+

Future uncertain so never upstreamed

Allowed for multiple Clutter frontends

Had AC quite early on!

Apertis

Last year…

An IVI platform

Based on Debian and Gnome tech

~~In the process of being made public~~

Collabora works on some components

Apertis

https://wiki.apertis.org/

https://git.apertis.org/

Last year…

“WebkitClutterGTK+”

Use as much GTK+ infrastructure as possible

~~clutter-gtk wrapper for ease of use~~

Input handling also done through actor

Allows API users to leverage PanAction

Last year…

Under the hood

Yoon’s threaded compositor

Fast zooming animations

Some features ported from WebKit Clutter

Otherwise pretty much vanilla WebKit2GTK+

Upstream!

Not so fast…

Project derailed by pivot in priorities

Had to whip up a proof of concept in record time, dropped everything and ran for it

The good news, though…

Quite a bit of perf investigation

Back on track to upstreaming

“WebkitClutterGTK+”

clutter-gtk had terrible cost/benefit

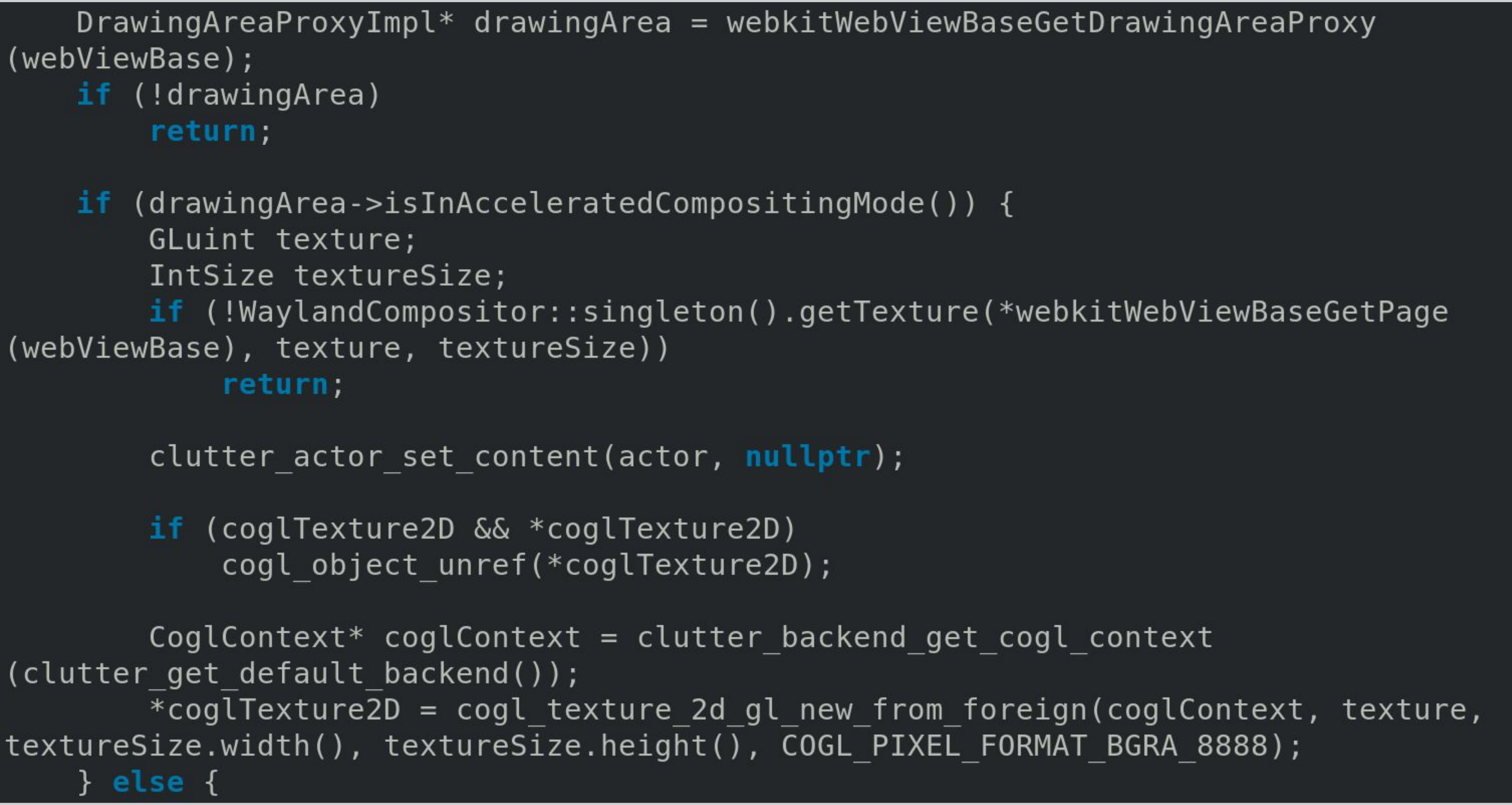
GTK+ widget hosted on GtkOffscreenWindow

Rendering, ViewState, focus handling disabled

WebKitWebClutterView takes over

“WebkitClutterGTK+”

WebClutterView gets texture from WaylandCompositor, wraps it on a CoglTexture2D and adds it on paint_node()



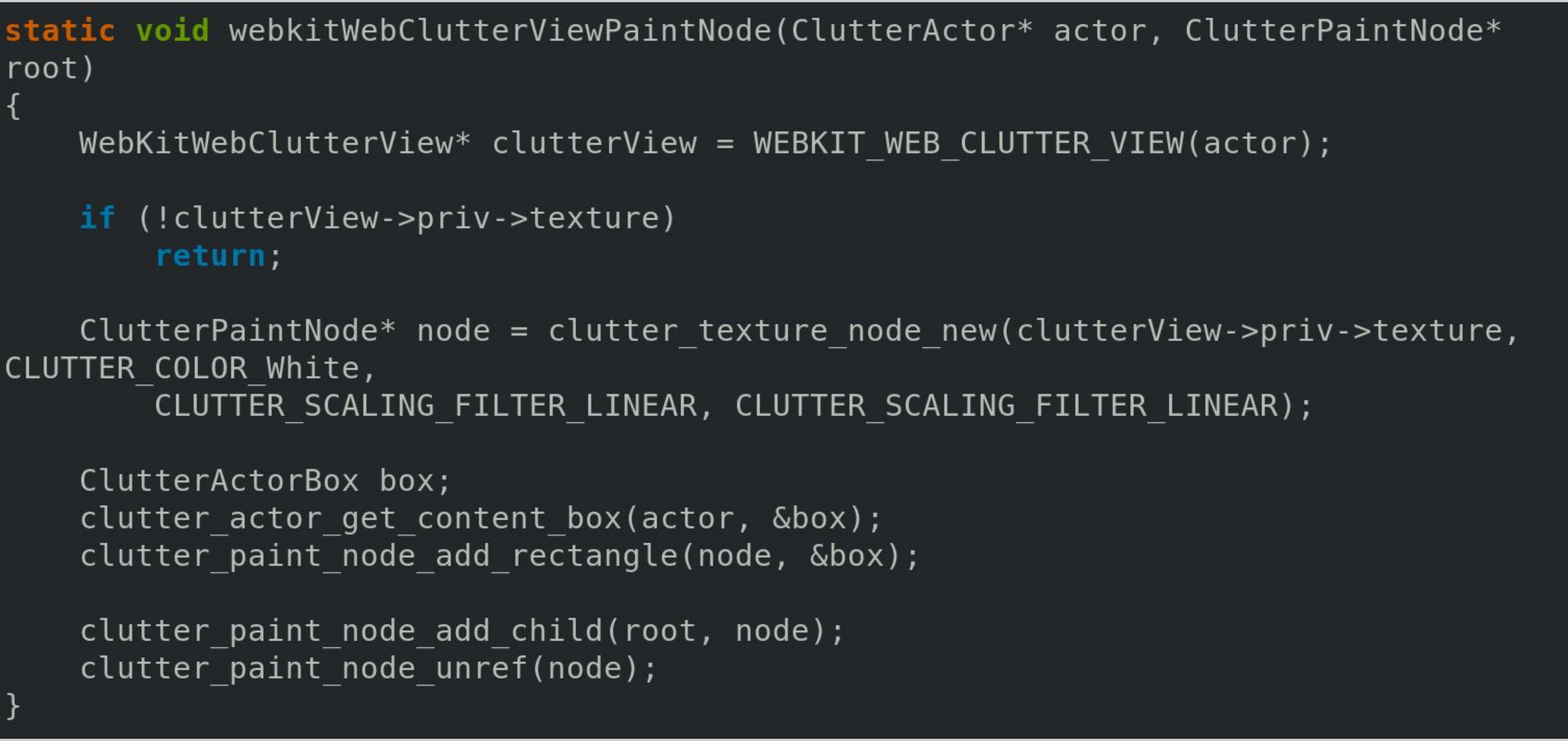
```
    DrawingAreaProxyImpl* drawingArea = webkitWebViewBaseGetDrawingAreaProxy
(webViewBase);
    if (!drawingArea)
        return;

    if (drawingArea->isInAcceleratedCompositingMode()) {
        GLuint texture;
        IntSize textureSize;
        if (!WaylandCompositor::singleton().getTexture(*webkitWebViewBaseGetPage
(webViewBase), texture, textureSize))
            return;

        clutter_actor_set_content(actor, nullptr);

        if (coglTexture2D && *coglTexture2D)
            cogl_object_unref(*coglTexture2D);

        CoglContext* coglContext = clutter_backend_get_cogl_context
(clutter_get_default_backend());
        *coglTexture2D = cogl_texture_2d_gl_new_from_foreign(coglContext, texture,
textureSize.width(), textureSize.height(), COGL_PIXEL_FORMAT_BGRA_8888);
    } else {
```



```
static void webkitWebClutterViewPaintNode(ClutterActor* actor, ClutterPaintNode*
root)
{
    WebKitWebClutterView* clutterView = WEBKIT_WEB_CLUTTER_VIEW(actor);

    if (!clutterView->priv->texture)
        return;

    ClutterPaintNode* node = clutter_texture_node_new(clutterView->priv->texture,
CLUTTER_COLOR_White,
        CLUTTER_SCALING_FILTER_LINEAR, CLUTTER_SCALING_FILTER_LINEAR);

    ClutterActorBox box;
    clutter_actor_get_content_box(actor, &box);
    clutter_paint_node_add_rectangle(node, &box);

    clutter_paint_node_add_child(root, node);
    clutter_paint_node_unref(node);
}
```

Why use the GTK+ widget?

- We want to duplicate as little as possible
- It nicely manages sizes, notifications
- We do duplicate some of the signals and APIs for convenience

Thanks!

gustavo.noronha@collabora.co.uk

emanuele.aina@collabora.co.uk

andre.magalhaes@collabora.co.uk

http://git.collabora.co.uk/