Subject: GLCtrl in GTK
Posted by mirek on Sun, 25 Nov 2018 19:47:50 GMT
View Forum Message <> Reply to Message

GTK GLCtrl is now refactored with single context (just as Win32 implementation) and is functionally equivalent.

Subject: Re: GLCtrl in GTK
Posted by ptkacz on Mon, 26 Nov 2018 03:35:57 GMT
View Forum Message <> Reply to Message

Hi Mirek,

Does this resolve the issue, [BUG] GL canvas gone missing...?

Is this included in the latest nightly build?

Peter

Subject: Re: GLCtrl in GTK
Posted by ptkacz on Wed, 28 Nov 2018 04:43:47 GMT
View Forum Message <> Reply to Message

Hi Mirek,

I downloaded and built upp-x11-src-12584. Looks like there's an issue with U++ & OpenGL, since when I build and run the "GLDrawDemo" from the U++ "reference" section, nothing renders.

The first issue that I ran into was the following, "ld: cannot find -libgtkglext-x11-1.0". Once addressed, I rebuilt and attempted to run the "GLDrawDemo" without luck.

Attached will be a tar.gz file consisting of three screen shots as follows:

1. What one gets when one builds the app and runs the demo. The application frame renders blank!
2. Making some adjustments so as not to make use of GTK, which I was doing previously, specified in U++ issue 1481.
3. Rebuilding and running the demo with GUI.NOGTK specified. The background gets rendered beneath the GL Window.

Unlike before, (msg 48505) no heap errors were encountered when running the U++ demo! Unfortunately, the fix (issue 1481) targeted for the next major release of U++ (2018.2) no longer

can be applied since, "hasdhctrl" is no longer defined.

Peter

File Attachments

1) GLScreenShots.tar.gz, downloaded 347 times

Subject: Re: GLCtrl in GTK
Posted by mirek on Wed, 28 Nov 2018 08:39:33 GMT
View Forum Message <> Reply to Message

ptkacz wrote on Wed, 28 November 2018 05:43Hi Mirek,

I downloaded and built upp-x11-src-12584. Looks like there's an issue with U++ & OpenGL, since when I build and run the "GLDrawDemo" from the U++ "reference" section, nothing renders.

The first issue that I ran into was the following, "ld: cannot find -libgtkglext-x11-1.0". Once addressed, I rebuilt and attempted to run the "GLDrawDemo" without luck.

Attached will be a tar.gz file consisting of three screen shots as follows:

1. What one gets when one builds the app and runs the demo. The application frame renders blank!
2. Making some adjustments so as not to make use of GTK, which I was doing previously, specified in U++ issue 1481.
3. Rebuilding and running the demo with GUI.NOGTK specified. The background gets rendered beneath the GL Window.

Unlike before, (msg 48505) no heap errors were encountered when running the U++ demo! Unfortunately, the fix (issue 1481) targeted for the next major release of U++ (2018.2) no longer can be applied since, "hasdhctrl" is no longer defined.

Peter

OK, looks like it is more fragile than expected... For what is worth, it works with old Intel integrated GPU and with Radeon, albeit there are some new leaks to resolve.

What are specs of your system?

BTW, NOGTK is not supported for GLCtrl at this moment.

Subject: Re: GLCtrl in GTK
Posted by Oblivion on Wed, 28 Nov 2018 09:03:15 GMT
View Forum Message <> Reply to Message

Hello Mirek,

I also have an additional (old) machine with AMD FX 6100, Radeon R9 270, Linux 4.18, GNOME/GTK 3.30, with Xorg Radeon drivers.

I can confirm Peter's observations.

Reference examples (GLDraw demo and GLCtrl demo) work fine on GTK but they leak memory on exit. I've noticed that there is a GLCtrl::Destroy() method declared for GTK build, but it is not defined nor used anywhere. Maybe related?

Best regards,
Obiivion

Subject: Re: GLCtrl in GTK
Posted by mirek on Wed, 28 Nov 2018 09:41:49 GMT
View Forum Message <> Reply to Message

Oblivion wrote on Wed, 28 November 2018 10:03Hello Mirek,

I also have an additional (old) machine with AMD FX 6100, Radeon R9 270, Linux 4.18, GNOME/GTK 3.30, with Xorg Radeon drivers.

I can confirm Peter's observations.

Reference examples (GLDraw demo and GLCtrl demo) work fine on GTK but they leak memory on exit. I've noticed that there is a GLCtrl::Destroy() method declared for GTK build, but it is not defined nor used anywhere. Maybe related?

Best regards,
Obiivion

Well, you confirm mine (and klugier's) observation that radeon produces leaks. We are working on it, but the leak seems to be in OpenGL drivers and it is 'stable'. Looks like the driver is in C++ and calls 'new' for some global static (as in 'singleton') data, but never bothers to release that memory on exit - we think this because the number of leaks is always the same number regardless what we do (perhaps you can test that theory too?). I hope that well placed IgnoreMemoryLeaksBlock will resolve that.

Peter's observation is blank screen. Thats sort of more serious problem :)

Subject: Re: GLCtrl in GTK
Posted by Oblivion on Wed, 28 Nov 2018 12:24:01 GMT
View Forum Message <> Reply to Message

Hello Mirek,

Quote:
we think this because the number of leaks is always the same number regardless what we do (perhaps you can test that theory too?).

Yes, same on my machine. Always the same number (4) and size (48b, 48b, 16b, 16b) of memory. Both with reference examples and a simple glctrl app I've created.

Pattern:

Heap leaks detected:

```
--memory-breakpoint__ 13578 : Memory at 0x0x7fe4901bf600, size 0x30 = 48
    +0 0x00007FE4901BF600 01 00 00 00 46 72 65 65 68 F3 1B 90 E4 7F 00 00     ....Freeh....•..
   +16 0x00007FE4901BF610 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00     ................
   +32 0x00007FE4901BF620 00 72 65 65 46 72 65 65 60 89 9E 03 D3 55 00 00
.reeFree`....U..

--memory-breakpoint__ 13577 : Memory at 0x0x7fe4901bf360, size 0x30 = 48
    +0 0x00007FE4901BF360 46 72 65 65 46 72 65 65 00 00 00 00 46 72 65 65
FreeFree....Free
   +16 0x00007FE4901BF370 00 F6 1B 90 E4 7F 00 00 00 F6 1B 90 E4 7F 00 00     .....•.......•..
   +32 0x00007FE4901BF380 00 F6 1B 90 E4 7F 00 00 01 00 00 00 00 00 00 00     .....•..........

--memory-breakpoint__ 13576 : Memory at 0x0x7fe4901ab6e0, size 0x10 = 16
    +0 0x00007FE4901AB6E0 80 97 95 03 D3 55 00 00 00 00 00 00 01 72 65 65     .....U.......ree

--memory-breakpoint__ 6757 : Memory at 0x0x7fe498a9fba0, size 0x10 = 16
    +0 0x00007FE498A9FBA0 08 0E EE 8B E4 7F 00 00 00 72 65 65 00 00 00 00     .....•...ree....
****************** PANIC: Heap leaks detected!
```

Quote:
Peter's observation is blank screen. Thats sort of more serious problem

Yes, I get this blank/black screen with NOGTK only. I was referring to that. But as you pointed out NOGTK for GLCtrl is broken. :)

Best regards,
Oblivion

Subject: Re: GLCtrl in GTK
Posted by mirek on Wed, 28 Nov 2018 16:25:52 GMT
View Forum Message <> Reply to Message

Oblivion wrote on Wed, 28 November 2018 13:24Hello Mirek,

Quote:
we think this because the number of leaks is always the same number regardless what we do (perhaps you can test that theory too?).

Yes, same on my machine. Always the same number (4) and size (48b, 48b, 16b, 16b) of memory. Both with reference examples and a simple glctrl app I've created.

Pattern:

Heap leaks detected:

```
--memory-breakpoint__ 13578 : Memory at 0x0x7fe4901bf600, size 0x30 = 48
    +0 0x00007FE4901BF600 01 00 00 00 46 72 65 65 68 F3 1B 90 E4 7F 00 00     ....Freeh....•..
   +16 0x00007FE4901BF610 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00     ................
   +32 0x00007FE4901BF620 00 72 65 65 46 72 65 65 60 89 9E 03 D3 55 00 00
.reeFree`....U..

--memory-breakpoint__ 13577 : Memory at 0x0x7fe4901bf360, size 0x30 = 48
    +0 0x00007FE4901BF360 46 72 65 65 46 72 65 65 00 00 00 00 46 72 65 65
FreeFree....Free
   +16 0x00007FE4901BF370 00 F6 1B 90 E4 7F 00 00 00 F6 1B 90 E4 7F 00 00     .....•.......•..
   +32 0x00007FE4901BF380 00 F6 1B 90 E4 7F 00 00 01 00 00 00 00 00 00 00     .....•..........

--memory-breakpoint__ 13576 : Memory at 0x0x7fe4901ab6e0, size 0x10 = 16
    +0 0x00007FE4901AB6E0 80 97 95 03 D3 55 00 00 00 00 00 00 01 72 65 65     .....U.......ree

--memory-breakpoint__ 6757 : Memory at 0x0x7fe498a9fba0, size 0x10 = 16
    +0 0x00007FE498A9FBA0 08 0E EE 8B E4 7F 00 00 00 72 65 65 00 00 00 00     .....•...ree....
********** PANIC: Heap leaks detected!
```

Quote:
Peter's observation is blank screen. Thats sort of more serious problem

Yes, I get this blank/black screen with NOGTK only. I was referring to that. But as you pointed out

NOGTK for GLCtrl is broken. :)

Best regards,
Oblivion

Perhaps you could try to add some "IgnoreMemoryLeaksBlock __;" into various parts of GLCtrl and see if it resolves it?

Mirek

Subject: Re: GLCtrl in GTK
Posted by Oblivion on Wed, 28 Nov 2018 17:54:23 GMT
View Forum Message <> Reply to Message

Hopefully, I narrowed down the area to investigate. (Sorry, I don't have time now, but I'll be able to look into it tomorrow.)

```
void GLCtrl::Paint(Draw& w)
{
 Size sz = GetSize();
 if(!s_GLXContext || sz.cx == 0 || sz.cy == 0)
  return;

       MemoryIgnoreLeaksBlock __;      // This supresses all. Note: Putting this below
glXMakeCurrent() leads to heap leaks message only when the ctrl is resized.

 glXMakeCurrent(s_Display, win, s_GLXContext);

 ONCELOCK {
  glewInit();
 }



 DoGLPaint();

 if(doubleBuffering)
  glXSwapBuffers(s_Display, win);
 else
  glFlush();

 glXMakeCurrent(s_Display, None, NULL);
}
```

Edit:

Apparently, GlCtrl::Sync() (resize) causes extra leaks.

I've noted one more thing: DrawImage(), for some reason, doesn't work. Image is not painted onto screen.

Best regards,
Oblivion

Subject: Re: GLCtrl in GTK
Posted by ptkacz on Thu, 29 Nov 2018 03:54:55 GMT
View Forum Message <> Reply to Message

Hi Merik,

My system: AMD Threadripper, GTX 1080 video card, Linux Mint 64-bit latest version. I also have 3D modelling software running in Linux.

Peter

Subject: Re: GLCtrl in GTK
Posted by ptkacz on Thu, 29 Nov 2018 04:25:23 GMT
View Forum Message <> Reply to Message

Offending code identified. ...I decided to roll the dice and went back to the fix for issue # 1481. Below is the code changes that I made to CtrlDraw.cpp. For compiling the OpenGL demo, NOGTK was specified and vola, the demo worked!

Changes made to CtrlDraw.cpp, starting at line 595:

```
void Ctrl::UpdateArea0(SystemDraw& draw, const Rect& clip, int backpaint)
{
 GuiLock __;
 LTIMING("UpdateArea");
 LLOG("========== UPDATE AREA " << UPP::Name(this) << " " << clip << " ==========");
//      ExcludeDHCtrls(draw, GetRect().GetSize(), clip);
 if(globalbackbuffer) {
  CtrlPaint(draw, clip);
  LLOG("========== END (TARGET IS BACKBUFFER)");
  return;
 }
/*
 if(backpaint == FULLBACKPAINT || globalbackpaint) {
```

```
  ShowRepaintRect(draw, clip, LtRed());
  BackDraw bw;
  bw.Create(draw, clip.GetSize());
  bw.Offset(-clip.TopLeft());
  bw.SetPaintingDraw(draw, clip.TopLeft());
  CtrlPaint(bw, clip);
  bw.Put(draw, clip.TopLeft());
  LLOG("========== END (FULLBACKPAINT)");
  return;
 }
*/
 if(backpaint == TRANSPARENTBACKPAINT) {
  LLOG("TransparentBackpaint");
  Vector<Rect> area;
  GatherTransparentAreas(area, draw, GetRect().GetSize(), clip);
```

Line 600, "ExcludeDHCtrls(..." was commented out as specified in issue ticket, 1481.

As for the commented block of code, this was a gamble, a bit of a hunch, since in issue ticket 1481, the following change had originally been specified:

```
- if(backpaint == FULLBACKPAINT || globalbackpaint/* && !hasdhctrl && !dynamic_cast<DHCtrl
*>(this)*/) {
+ if(backpaint == FULLBACKPAINT || globalbackpaint && !hasdhctrl && !dynamic_cast<DHCtrl
*>(this)) {
```

Since if I recall that hasdhctrl is no longer defined, I wondered if just commenting out that block of code would make a difference. It appears to have done so. I'd attach a screen shot of the working demo, but have a cap of one attachment per message.

Peter

Subject: Re: GLCtrl in GTK
Posted by mirek on Thu, 29 Nov 2018 07:39:40 GMT
View Forum Message <> Reply to Message

ptkacz wrote on Thu, 29 November 2018 05:25Offending code identified. ...I decided to roll the dice and went back to the fix for issue # 1481. Below is the code changes that I made to CtrlDraw.cpp. For compiling the OpenGL demo, NOGTK was specified and vola, the demo worked!

Changes made to CtrlDraw.cpp, starting at line 595:

```
void Ctrl::UpdateArea0(SystemDraw& draw, const Rect& clip, int backpaint)
{
```

```
 GuiLock __;
 LTIMING("UpdateArea");
 LLOG("========== UPDATE AREA " << UPP::Name(this) << " " << clip << " ==========");
//      ExcludeDHCtrls(draw, GetRect().GetSize(), clip);
 if(globalbackbuffer) {
  CtrlPaint(draw, clip);
  LLOG("========== END (TARGET IS BACKBUFFER)");
  return;
 }
/*
 if(backpaint == FULLBACKPAINT || globalbackpaint) {
  ShowRepaintRect(draw, clip, LtRed());
  BackDraw bw;
  bw.Create(draw, clip.GetSize());
  bw.Offset(-clip.TopLeft());
  bw.SetPaintingDraw(draw, clip.TopLeft());
  CtrlPaint(bw, clip);
  bw.Put(draw, clip.TopLeft());
  LLOG("========== END (FULLBACKPAINT)");
  return;
 }
*/
 if(backpaint == TRANSPARENTBACKPAINT) {
  LLOG("TransparentBackpaint");
  Vector<Rect> area;
  GatherTransparentAreas(area, draw, GetRect().GetSize(), clip);
```

Line 600, "ExcludeDHCtrls(..." was commented out as specified in issue ticket, 1481.

As for the commented block of code, this was a gamble, a bit of a hunch, since in issue ticket 1481, the following change had originally been specified:

```
- if(backpaint == FULLBACKPAINT || globalbackpaint/* && !hasdhctrl && !dynamic_cast<DHCtrl
*>(this)*/) {
+ if(backpaint == FULLBACKPAINT || globalbackpaint && !hasdhctrl && !dynamic_cast<DHCtrl
*>(this)) {
```

Since if I recall that hasdhctrl is no longer defined, I wondered if just commenting out that block of code would make a difference. It appears to have done so. I'd attach a screen shot of the working demo, but have a cap of one attachment per message.

Peter

OK, now I am officially confused....

Thing is, new GLCtrl for GTK has nothing to do with DHCtrl. And NOGTK variant is not supported at this moment (I might implement that later or not, X11 backend being something between deprecated and fallback for now).

From your posts it is maybe a bit unclear whether you have tested plain GTK mode (that is, without NOGTK)? That is what is supposed to be fixed...

Mirek

Subject: Re: GLCtrl in GTK
Posted by mirek on Thu, 29 Nov 2018 08:21:09 GMT
View Forum Message <> Reply to Message

Oblivion wrote on Wed, 28 November 2018 18:54
I've noted one more thing: DrawImage(), for some reason, doesn't work. Image is not painted onto screen.

Fixed.

Subject: Re: GLCtrl in GTK
Posted by ptkacz on Thu, 29 Nov 2018 12:33:36 GMT
View Forum Message <> Reply to Message

Mirek,

Upon making the code changes and building, I first tested the demo without the NOGTK and was presented with a blank demo window. When specifying NOGTK + GUI application, the demo ran.

Subject: Re: GLCtrl in GTK
Posted by mirek on Mon, 03 Dec 2018 07:54:15 GMT
View Forum Message <> Reply to Message

ptkacz wrote on Thu, 29 November 2018 13:33Mirek,

Upon making the code changes and building, I first tested the demo without the NOGTK and was presented with a blank demo window. When specifying NOGTK + GUI application, the demo ran.

:(

Well, we better have this fixed. I hope you will help me, as your setup seems to be the only one failing.

Can you test this? This is boilerplate GLX code from the internet that was my starting point in developing new GLCtrl incarnation:

```
#include <Core/Core.h>

#include <GL/glx.h>
#include <GL/gl.h>
#include <unistd.h>
#include <iostream>

#include <stdlib.h>
#include <string.h>
#include <stdio.h>

#define GLX_CONTEXT_MAJOR_VERSION_ARB       0x2091
#define GLX_CONTEXT_MINOR_VERSION_ARB       0x2092

typedef GLXContext (*glXCreateContextAttribsARBProc)(Display*, GLXFBConfig, GLXContext,
Bool, const int*);

CONSOLE_APP_MAIN
{
   Display *display = XOpenDisplay(0);

   glXCreateContextAttribsARBProc glXCreateContextAttribsARB = NULL;

   const char *extensions = glXQueryExtensionsString(display, DefaultScreen(display));
   std::cout << extensions << std::endl;

   static int visual_attribs[] =
   {
      GLX_RENDER_TYPE, GLX_RGBA_BIT,
      GLX_DRAWABLE_TYPE, GLX_WINDOW_BIT,
      GLX_DOUBLEBUFFER, true,
      GLX_RED_SIZE, 1,
      GLX_GREEN_SIZE, 1,
      GLX_BLUE_SIZE, 1,
      None
    };

   std::cout << "Getting framebuffer config" << std::endl;
   int fbcount;
   GLXFBConfig *fbc = glXChooseFBConfig(display, DefaultScreen(display), visual_attribs,
&fbcount);
   if (!fbc)
   {
```

```
        std::cout << "Failed to retrieve a framebuffer config" << std::endl;
        return;
    }

    std::cout << "Getting XVisualInfo" << std::endl;
    XVisualInfo *vi = glXGetVisualFromFBConfig(display, fbc[0]);

    XSetWindowAttributes swa;
    memset(&swa, 0, sizeof(swa));
    std::cout << "Creating colormap" << std::endl;
    swa.colormap = XCreateColormap(display, RootWindow(display, vi->screen), vi->visual,
AllocNone);
    swa.border_pixel = 0;
    swa.event_mask = 0;//StructureNotifyMask;

    std::cout << "Creating window" << std::endl;
    Window win = XCreateWindow(display, RootWindow(display, vi->screen), 0, 0, 100, 100, 0,
vi->depth, InputOutput, vi->visual, CWBorderPixel|CWColormap|CWEventMask, &swa);
    if (!win)
    {
        std::cout << "Failed to create window." << std::endl;
        return;
    }

    std::cout << "Mapping window" << std::endl;
    XMapWindow(display, win);

    // Create an oldstyle context first, to get the correct function pointer for
glXCreateContextAttribsARB
    GLXContext ctx_old = glXCreateContext(display, vi, 0, GL_TRUE);
    glXCreateContextAttribsARB =  (glXCreateContextAttribsARBProc)glXGetProcAddress((const
GLubyte*)"glXCreateContextAttribsARB");
    glXMakeCurrent(display, 0, 0);
    glXDestroyContext(display, ctx_old);

    if (glXCreateContextAttribsARB == NULL)
    {
        std::cout << "glXCreateContextAttribsARB entry point not found. Aborting." << std::endl;
        return;
    }

    static int context_attribs[] =
    {
        GLX_CONTEXT_MAJOR_VERSION_ARB, 3,
        GLX_CONTEXT_MINOR_VERSION_ARB, 0,
        None
    };
```

```
/*
    std::cout << "Creating context" << std::endl;
    GLXContext ctx = glXCreateContextAttribsARB(display, fbc[0], NULL, true, context_attribs);
    if (!ctx)
    {
        std::cout << "Failed to create GL3 context." << std::endl;
        return;
    }
*/
//    GLXContext ctx = glXCreateContextAttribsARB(display, fbc[0], NULL, true, context_attribs);
 GLXContext ctx = glXCreateContext(display, vi, NULL, GL_TRUE);

    std::cout << "Making context current" << std::endl;
    glXMakeCurrent(display, win, ctx);

  for(int i = 0; i < 4; i++) {
        glClearColor (0, 0.5, 1, 1);
        glClear (GL_COLOR_BUFFER_BIT);
        glXSwapBuffers (display, win);

        sleep(1);

        glClearColor (1, 0.5, 0, 1);
        glClear (GL_COLOR_BUFFER_BIT);
        glXSwapBuffers (display, win);

        sleep(1);
  }
    ctx = glXGetCurrentContext();
    glXMakeCurrent(display, 0, 0);
    glXDestroyContext(display, ctx);
}
```

It should create a window and blink it 4 times...

Also, can you please .zip your GLCtrl package with changes applied and post here?

Thanks,

Mirek

Subject: Re: GLCtrl in GTK
Posted by ptkacz on Wed, 05 Dec 2018 03:14:10 GMT
View Forum Message <> Reply to Message

Hi Mirek,

I downloaded and built nightly build, 12597 as is. Upon adding the GLCtrl and GLDraw assemblies, building with default settings of GUI & GCC Debug, the application ran cycling between blue and orange for four times. Similarly, with the options of GUI and GCC release.

Peter

Subject: Re: GLCtrl in GTK
Posted by mirek on Thu, 06 Dec 2018 07:10:09 GMT
View Forum Message <> Reply to Message

Thank you. Now let us try adding some LOGs to GLCtrl/GTKGlCtrl.cpp.

Please change it to:

```
#include "GLCtrl.h"

#ifdef GUI_GTK

#include <GL/glx.h>
#include <GL/gl.h>
#include <gdk/gdkx.h>

namespace Upp {

static XVisualInfo *s_XVisualInfo;
static Colormap     s_Colormap;
static GLXContext   s_GLXContext;
static ::Display   *s_Display;

EXITBLOCK {
 if(s_GLXContext)
  glXDestroyContext(s_Display, s_GLXContext);
}

void GLCtrl::Create()
{
 DLOG("Create");
 Ctrl *top = GetTopCtrl();
 if(!top)
  return;

 GdkWindow *gdk = top->gdk();
 if(!gdk)
```

```
 return;

DDUMP(gdk);

Window w = gdk_x11_drawable_get_xid((GdkDrawable *)gdk);

DDUMP(w);

ONCELOCK {
 s_Display = gdk_x11_drawable_get_xdisplay((GdkDrawable *)gdk);
 int samples = numberOfSamples;
 GLXFBConfig *fbc;
 do {
  Vector<int> attr;
  attr << GLX_RGBA << GLX_DEPTH_SIZE << depthSize
       << GLX_STENCIL_SIZE << stencilSize;
  if(doubleBuffering)
   attr << GLX_DOUBLEBUFFER;
  if(samples > 1)
   attr << GLX_SAMPLE_BUFFERS_ARB << 1 << GLX_SAMPLES_ARB << samples;
  attr << 0;

  samples >>= 1;
  int fbcount;
  fbc = glXChooseFBConfig(s_Display, DefaultScreen(s_Display), attr, &fbcount);
 }
 while(!fbc && samples > 0);
 DDUMP(fbc);
 if(!fbc)
  return;
 s_XVisualInfo = glXGetVisualFromFBConfig(s_Display, fbc[0]);
 s_Colormap = XCreateColormap(s_Display, RootWindow(s_Display, s_XVisualInfo->screen),
s_XVisualInfo->visual, AllocNone);
 s_GLXContext = glXCreateContext(s_Display, s_XVisualInfo, NULL, GL_TRUE);

 DDUMP(s_XVisualInfo);
 DDUMP(s_Colormap);
 DDUMP(s_GLXContext);
}

if(!s_GLXContext)
 return;

XSetWindowAttributes swa;
swa.colormap = s_Colormap;
swa.border_pixel = 0;
swa.event_mask = 0;
```

```
 win = XCreateWindow(s_Display, w, 0, 0, 1, 1, 0,
                     s_XVisualInfo->depth, InputOutput, s_XVisualInfo->visual,
                     CWBorderPixel|CWColormap|CWEventMask, &swa);
 visible = false;
 position = Null;

 DDUMP(win);
}

void GLCtrl::Sync()
{
 if(win) {
  Rect r = GetScreenView() - GetTopCtrl()->GetScreenRect().TopLeft();
  bool b = IsVisible() && r.GetWidth() > 0 && r.GetHeight() > 0;
  if(b != visible) {
   visible = b;
   position = Null;
   if(b)
       XMapWindow(s_Display, win);
   else
       XUnmapWindow(s_Display, win);
  }
  if(r != position && visible) {
   position = r;
   XMoveResizeWindow(s_Display, win, r.left, r.top, r.Width(), r.Height());
  }
  DDUMP(visible);
  DDUMP(r);
 }
}

void GLCtrl::State(int reason)
{
 DLOG("State");
 switch(reason) {
 case CLOSE:
  DLOG("Destroy");
  XDestroyWindow(s_Display, win);
  break;
 case OPEN:
  DLOG("Create");
  Create();
 default:
  DLOG("Sync");
  Sync();
  break;
 }
}
```

```
void GLCtrl::Paint(Draw& w)
{
 Size sz = GetSize();
 DLOG("Paint");
 DDUMP(s_GLXContext);
 DDUMP(sz);
 DDUMP(s_Display);

 if(!s_GLXContext || sz.cx == 0 || sz.cy == 0)
  return;

 glXMakeCurrent(s_Display, win, s_GLXContext);

 ONCELOCK {
  glewInit();
 }

 DLOG("DoGLPaint");
 DoGLPaint();

 if(doubleBuffering)
  glXSwapBuffers(s_Display, win);
 else
  glFlush();

 glXMakeCurrent(s_Display, None, NULL);
}

}

#endif
```

run reference/OpenGL (without changing anything else) and post a .log file here (Alt+L).

Thanks. We need to figure this out...

Mirek

Subject: Re: GLCtrl in GTK
Posted by ptkacz on Sat, 08 Dec 2018 22:38:04 GMT
View Forum Message <> Reply to Message

Mirek,

Sorry for the delay.

Attached will be the artifacts captured, per your changes. First with GUI & Debug, then GUI NO GTK & Debug.

Peter

File Attachments

1) OpenGL_Artifacts.zip, downloaded 271 times

Subject: Re: GLCtrl in GTK
Posted by mirek on Sun, 09 Dec 2018 11:08:07 GMT
View Forum Message <> Reply to Message

Thank you. Based on logs, I have attempted the fix. Can you please try with trunk, or if you do not want to get dirty with svn nor git, can you please try to replace

```
void GLCtrl::Create()
{
 Ctrl *top = GetTopCtrl();
 if(!top)
  return;

 GdkWindow *gdk = top->gdk();
 if(!gdk)
  return;

 Window w = gdk_x11_drawable_get_xid((GdkDrawable *)gdk);

 ONCELOCK {
  s_Display = gdk_x11_drawable_get_xdisplay((GdkDrawable *)gdk);
  int samples = numberOfSamples;

  do {
   Vector<int> attr;
   attr << GLX_RGBA << GLX_DEPTH_SIZE << depthSize
        << GLX_STENCIL_SIZE << stencilSize;
   if(doubleBuffering)
    attr << GLX_DOUBLEBUFFER;
   if(samples > 1)
    attr << GLX_SAMPLE_BUFFERS_ARB << 1 << GLX_SAMPLES_ARB << samples;
   attr << 0;
   samples >>= 1;
   s_XVisualInfo = glXChooseVisual(s_Display, DefaultScreen(s_Display), attr);
  }
```

```
  while(!s_XVisualInfo && samples > 0);
  if(!s_XVisualInfo)
   return;
  s_Colormap = XCreateColormap(s_Display, RootWindow(s_Display, s_XVisualInfo->screen),
s_XVisualInfo->visual, AllocNone);
  s_GLXContext = glXCreateContext(s_Display, s_XVisualInfo, NULL, GL_TRUE);
 }

 if(!s_GLXContext)
  return;

 XSetWindowAttributes swa;
 swa.colormap = s_Colormap;
 swa.border_pixel = 0;
 swa.event_mask = 0;

 win = XCreateWindow(s_Display, w, 0, 0, 1, 1, 0,
                     s_XVisualInfo->depth, InputOutput, s_XVisualInfo->visual,
                     CWBorderPixel|CWColormap|CWEventMask, &swa);
 visible = false;
 position = Null;
}
```

Subject: Re: GLCtrl in GTK
Posted by ptkacz on Sun, 09 Dec 2018 18:49:07 GMT
View Forum Message <> Reply to Message

Hi Mirek,

I downloaded build, upp-x11-src-12606M and modified, void GLCtrl::Create() as requested. Once I fixed some syntax in GLCtrl.cpp, built the Open GL Demo, the application worked! Attached will be a provided screen shot. Later today, I'll do some more playing around to see how the other demos fair.

The issue with GLCtrl.cpp is as follows:void GLCtrl::DoGLPaint()
{
 glClearDepth(1);
 glClearColor(1, 0, 0, 1);
 glClear(GL_COLOR_B

```
 UFFER_BIT|GL_DEPTH_BUFFER_BIT|GL_STENCIL_BUFFER_BIT);
 glEnable(GL_MULTISAMPLE);
 Size sz = GetSize();
 current_viewport = sz;
 SetCurrentViewport();
 GLPaint();
}
```

Peter

File Attachments

1) ItWorked.png, downloaded 316 times

Subject: Re: GLCtrl in GTK
Posted by mirek on Mon, 10 Dec 2018 07:50:48 GMT
View Forum Message <> Reply to Message

Sorry about that typo...

Good to hear the issue seems to be finally resolved :)

Subject: Re: GLCtrl in GTK
Posted by ptkacz on Tue, 11 Dec 2018 05:09:48 GMT
View Forum Message <> Reply to Message

Mirek,

I went and tried each of the demos and got:

reference/OpenGL: Yes
reference/GLDrawDemo: Yes
examples-bazaar/FTGL_Demo: Fail

When I attempt to compile the FTGL_Demo, I'm running into an issue with FreeType. See the attached screen shot.

In the past I've gotten this project to compile and run.

Similarly, in another project, I'm now getting errors like:

```
In file included from /usr/include/GL/glx.h:30:0,
                 from /home/ptkacz/upp/uppsrc/GLCtrl/GLCtrl.h:25,
                 from /home/ptkacz/project/dev/FractGenOne/ConsoleDialog.h:6,
                 from /home/ptkacz/project/dev/FractGenOne/FractGen.h:8,
                 from /home/ptkacz/project/dev/FractGenOne/main.cpp:1,
                 from
/home/ptkacz/upp.out/dev/FractGenOne/GCC.Debug.Debug_Full.Gui.Main.Shared/$blitz.cpp:3:
/usr/include/X11/Xlib.h:199:9: error: reference to 'Font' is ambiguous
        Font font;        /* default text font for text operations */
        ^~~~
In file included from /usr/include/X11/Xlib.h:44:0,
                 from /usr/include/GL/glx.h:30,
                 from /home/ptkacz/upp/uppsrc/GLCtrl/GLCtrl.h:25,
                 from /home/ptkacz/project/dev/FractGenOne/ConsoleDialog.h:6,
                 from /home/ptkacz/project/dev/FractGenOne/FractGen.h:8,
                 from /home/ptkacz/project/dev/FractGenOne/main.cpp:1,
                 from
/home/ptkacz/upp.out/dev/FractGenOne/GCC.Debug.Debug_Full.Gui.Main.Shared/$blitz.cpp:3:
/usr/include/X11/X.h:100:13: note: candidates are: typedef XID Font
 typedef XID Font;
```

Peter

File Attachments

1) FreeTypeIssue.png, downloaded 402 times

Subject: Re: GLCtrl in GTK
Posted by ptkacz on Tue, 11 Dec 2018 05:19:00 GMT
View Forum Message <> Reply to Message

Just did some quick checking the last screen shot that I attached. The FTGL_Demo by default only has the option of GUI NOGTK.

As for an application that I'm working on, if I specify GUI, I receive a bunch of compilation errors. If I specify GUI.NOGTK, the compilation errors go away and I am able to run the application. Unfortunately, it looks like I'll have to go back and comment out some U++ code in order to get the Open GL code to display as before...

It looks like the issue has not gone away...

Subject: Re: GLCtrl in GTK
Posted by mirek on Tue, 11 Dec 2018 09:47:12 GMT
View Forum Message <> Reply to Message

OK, I think I might have found the reason, can you please try with trunk?

For what is worth, FTGL_Demo now works (without NOGTK).

Subject: Re: GLCtrl in GTK
Posted by ptkacz on Wed, 12 Dec 2018 03:43:17 GMT
View Forum Message <> Reply to Message

Mirek,

I downloaded and successfully built nightly build, upp-x11-src-12610M. Compiling of the FTGL_Demo was unsuccessful. Attached will be feedback generated during a couple of compilation attempts. The same issues encountered last night, appear to be resulting. No adjustments we made to any U++ C++ library code.

Peter

File Attachments
1) build-12610M-FTGL-Demo.zip, downloaded 309 times

Subject: Re: GLCtrl in GTK

Posted by mirek on Wed, 12 Dec 2018 08:10:13 GMT
View Forum Message <> Reply to Message

ptkacz wrote on Wed, 12 December 2018 04:43Mirek,

I downloaded and successfully built nightly build, upp-x11-src-12610M. Compiling of the FTGL_Demo was unsuccessful. Attached will be feedback generated during a couple of compilation attempts. The same issues encountered last night, appear to be resulting. No adjustments we made to any U++ C++ library code.

Peter

12610 is not enough, changes are in 12611. Probably it was too soon for nightly or nightly build has failed.

I suggest checking out svn trunk (or git) and keeping it in sync. After you have install svn, the process is more or less automated from theide - check Setup menu, the last item.

Subject: Re: GLCtrl in GTK
Posted by ptkacz on Sat, 15 Dec 2018 03:56:05 GMT
View Forum Message <> Reply to Message

Mirek,

Utilizing Git, I downloaded the mirror. One question, where is the Makefile?

Peter

Subject: Re: GLCtrl in GTK
Posted by mirek on Sat, 15 Dec 2018 10:48:41 GMT
View Forum Message <> Reply to Message

ptkacz wrote on Sat, 15 December 2018 04:56Mirek,

Utilizing Git, I downloaded the mirror. One question, where is the Makefile?

Peter

Nowhere (that is generated for nightly only). But you do not need one.

Simply create new assemblies for the mirror. Or replace uppsrc with downloaded one.

Mirek

Subject: Re: GLCtrl in GTK
Posted by ptkacz on Sun, 16 Dec 2018 04:14:03 GMT
View Forum Message <> Reply to Message

Hi Mirek,

Just an FYI, within the Documentation section of this website, the link, POSIX (Linux, BSD, ...) setup is broken.

As for applying the master's version's U++ source directory on top of the last nightly build's U++ source directory, I received the following errors when making:
...
_out/ide/Builders/GCCMK-Gcc-Gui-Linux-Posix-Shared/Builders.a(GccBuilder.o): In function `GccBuilder::Link(Upp::Vector<Upp::String> const&, Upp::String const&, bool)':
GccBuilder.cpp:(.text._ZN10GccBuilder4LinkERKN3Upp6VectorINS0_6StringEEERKS2_b+0x84) : undefined reference to `GccBuilder::CocoaAppBundle()'
collect2: error: ld returned 1 exit status
Makefile:255: recipe for target 'ide.out' failed
make[1]: *** [ide.out] Error 1
...
...
...
_out/ide/Builders/GCCMK-Gcc-Linux-Posix-Shared/Builders.a(GccBuilder.o): In function `GccBuilder::Link(Upp::Vector<Upp::String> const&, Upp::String const&, bool)':
GccBuilder.cpp:(.text._ZN10GccBuilder4LinkERKN3Upp6VectorINS0_6StringEEERKS2_b+0x84) : undefined reference to `GccBuilder::CocoaAppBundle()'
collect2: error: ld returned 1 exit status
uMakefile:84: recipe for target 'umk.out' failed
make[1]: *** [umk.out] Error 1
make[1]: Leaving directory '/home/ptkacz/Downloads/upp-x11-src-12610M/uppsrc'
DOMAKE WARNING: Can't find uppsrc/ide.out binary
Makefile:5: recipe for target 'all' failed
make: *** [all] Error 4

Subject: Re: GLCtrl in GTK
Posted by mirek on Sun, 16 Dec 2018 07:11:34 GMT
View Forum Message <> Reply to Message

ptkacz wrote on Sun, 16 December 2018 05:14Hi Mirek,

Just an FYI, within the Documentation section of this website, the link, POSIX (Linux, BSD, ...)

setup is broken.

As for applying the master's version's U++ source directory on top of the last nightly build's U++ source directory, I received the following errors when making:
[code]...
_out/ide/Builders/GCCMK-Gcc-Gui-Linux-Posix-Shared/Builders. a(GccBuilder.o): In function `GccBuilder::Link(Upp::Vector<Upp::String> const&, Upp::String const&, bool)':
GccBuilder.cpp:(.text._ZN10GccBuilder4LinkERKN3Upp6VectorINS0_6StringEEERKS2_b+0x84): undefined reference to `GccBuilder::CocoaAppBundle()'
collect2: error: ld returned 1 exit status
Makefile:255: recipe for target 'ide.out' failed
make[1]: *** [ide.out] Error 1

I am sorry for not explaining this fully.. By "you do not need makefile" I really mean "you do not need to use make".

The only situation when you are supposed to use make is to build theide. After that, you can use theide to build things.

If you replace sources, you are done and you can then retry your code (in theide).

Mirek

Subject: Re: GLCtrl in GTK
Posted by ptkacz on Sun, 16 Dec 2018 18:04:52 GMT
View Forum Message <> Reply to Message

I copied over the master u++ source directory over the installed u++ source directory and was successfully able to get the OpenGL and GLDrawDemo apps to compile and run. As for my application, while able to build and run, the window with the OpenGL canvas appeared with a red background, that has never happened before.

As for the examples-bazaar/FTGL_Demo app., with either the NO.GTK or GUI options, I was not able to get the app to build, too many warning were generated.

Peter

Subject: Re: GLCtrl in GTK
Posted by mirek on Sun, 16 Dec 2018 19:54:29 GMT
View Forum Message <> Reply to Message

ptkacz wrote on Sun, 16 December 2018 19:04
As for the examples-bazaar/FTGL_Demo app., with either the NO.GTK or GUI options, I was not

able to get the app to build, too many warning were generated.
Peter

Can you send a list of those warnings?

(After rebuild all, you can switch theide to console and Ctrl+A select the list and then Ctrl+C copy it, paste to the new file, then post that file as attachement here).

Mirek

Subject: Re: GLCtrl in GTK
Posted by ptkacz on Mon, 17 Dec 2018 03:44:07 GMT
View Forum Message <> Reply to Message

I like the neat feature for CTRL+A and CTRL+C and will have to test in other applications... ;)

Attached will be the requested Console logs.

Peter

File Attachments
1) FTGL_Demo_logs.zip, downloaded 273 times

Subject: Re: GLCtrl in GTK
Posted by mirek on Thu, 20 Dec 2018 09:12:41 GMT
View Forum Message <> Reply to Message

I think that the problem is that you have replaced just 'uppsrc'. Please replace 'bazaar' too.

Mirek

Subject: Re: GLCtrl in GTK
Posted by ptkacz on Sat, 22 Dec 2018 17:31:03 GMT
View Forum Message <> Reply to Message

Hi Mirek,

I did as requests and same issue. I also downloaded nightly build 12639 (upp-x11-src-12639.tar.gz) and ran into the same issue.

Attached will be the output from the build console window.

The other the OpenGL demos built and ran with no issue.

Peter

File Attachments

1) FTGL-Open-Build-Failing.txt, downloaded 315 times

Subject: Re: GLCtrl in GTK
Posted by koldo on Sun, 07 Apr 2019 10:06:38 GMT
View Forum Message <> Reply to Message

OpenGL demo still fails with memory leaks.
I have monitored the first of them and is produced in main.cpp, inside OnPaint(), line 46, in glPopMatrix().

Subject: Re: GLCtrl in GTK
Posted by mirek on Sun, 07 Apr 2019 10:25:34 GMT
View Forum Message <> Reply to Message

Well, if so, then that is leak in OpenGL...

Mirek

Subject: Re: GLCtrl in GTK
Posted by koldo on Sun, 07 Apr 2019 19:25:22 GMT
View Forum Message <> Reply to Message

In this case, it could be better to set a MemoryIgnoreLeaksBlock __; in the beginning of OnPaint().
