U++ - Bug #988
KDE - GTK apps hangs in debug mode (GTK Backend!)

02/18/2015 08:05 PM - Zbigniew Rebacz

Status:	Approved	**Start date:**	02/18/2015
Priority:	High	**Due date:**	
Assignee:	Zbigniew Rebacz	**% Done:**	0%
Category:	CtrlCore	**Estimated time:**	0.00 hour
Target version:		**Spent time:**	0.00 hour

Description

To reproduce his bug. You will need to use KDE with oxygen-gtk theme (use by default on almost all KDE distributions).

1. Compile app in debug mode.
2. Run app in debuger (defualt process - CTRL + F5)
3. Use menu bar.
4. Turn off app.

It is hard to say from debug information what causing this problem.

- CRITICAL!

History

#1 - 02/20/2015 01:51 PM - Miroslav Fidler

Actuall, this is funny (and desperately sad). It looks like KDE/Qt leaves memory leaks, which are then reported by U++, using Gtk "panic" dialog, which then crashes in Qt (probably because heap is inactive at that point).

You can verify that:

- there is a nice function "Debug/Copy backtrace". Puts the stack on clipboard. Paste it here/examine

- Examine app log (Alt+L) whether you would see there memory leaks (I do...)

Mirek

#2 - 02/20/2015 01:51 PM - Miroslav Fidler

- Assignee changed from Miroslav Fidler to Zbigniew Rebacz

#3 - 02/20/2015 02:32 PM - Zbigniew Rebacz

- Assignee changed from Zbigniew Rebacz to Miroslav Fidler

ALT + L

```
* /home/klugier/upp.out/tutorial/GCC.Debug.Debug_Full.Gui.Shared/Gui07 20.02.2015 14:30:11, user: klugier

Heap leaks detected:

--memory-breakpoint__ 4156 : Memory at 0x0x7fffef639ca0, size 0x40 = 64
  +0 0x00007FFFEF639CA0 79 04 80 02 00 00 00 00 7B 04 80 02 00 00 00 00     y.......{.......
```

+16 0x00007FFFEF639CB0 7D 04 80 02 00 00 00 00 7F 04 80 02 00 00 00 00 }.....?.....
+32 0x00007FFFEF639CC0 81 04 80 02 00 00 00 00 83 04 80 02 00 00 00 00 ....
+48 0x00007FFFEF639CD0 85 04 80 02 00 00 00 00 87 04 80 02 00 00 00 00 ....

--memory-breakpoint__ 4152 : Memory at 0x0x7fffef639d10, size 0x40 = 64
+0 0x00007FFFEF639D10 69 04 80 02 00 00 00 00 6B 04 80 02 00 00 00 00 i.....k.....
+16 0x00007FFFEF639D20 6D 04 80 02 00 00 00 00 6F 04 80 02 00 00 00 00 m.....o.....
+32 0x00007FFFEF639D30 71 04 80 02 00 00 00 00 73 04 80 02 00 00 00 00 q.....s.....
+48 0x00007FFFEF639D40 75 04 80 02 00 00 00 00 77 04 80 02 00 00 00 00 u.....w.....
********** PANIC: Heap leaks detected!

Backtrace

00-std::less<unsigned int>::operator()(this=0x7ffff0079e10,__x=@0x6565724665657266: <error reading variable>,__y=@0x7fffffffd040: 3957515007) at /usr/include/c++/4.8/bits/stl_function.h:235
01-std::_Rb_tree<unsigned int, std::pair<unsigned int const, Oxygen::ColorUtils::Rgba>, std::_Select1st<std::pair<unsigned int const, Oxygen::ColorUtils::Rgba> >, std::less<unsigned int>, std::allocator<std::pair<unsigned int const, Oxygen::ColorUtils::Rgba> > >::_M_lower_bound(this=0x7ffff0079e10,__x=0x6565724665657246,__y=0x7ffff0079e18,__k=@0x7fffffffd040: 3957515007) at /usr/include/c++/4.8/bits/stl_tree.h:1141
02-std::_Rb_tree<unsigned int, std::pair<unsigned int const, Oxygen::ColorUtils::Rgba>, std::_Select1st<std::pair<unsigned int const, Oxygen::ColorUtils::Rgba> >, std::less<unsigned int>, std::allocator<std::pair<unsigned int const, Oxygen::ColorUtils::Rgba> > >::find(this=0x7ffff0079e10,__k=@0x7fffffffd040: 3957515007) at /usr/include/c++/4.8/bits/stl_tree.h:1792
03-std::map<unsigned int, Oxygen::ColorUtils::Rgba, std::less<unsigned int>, std::allocator<std::pair<unsigned int const, Oxygen::ColorUtils::Rgba> > >::find(this=0x7ffff0079e10,__x=@0x7fffffffd040: 3957515007) at /usr/include/c++/4.8/bits/stl_map.h:822
04-Oxygen::SimpleCache<unsigned int, Oxygen::ColorUtils::Rgba>::find(this,key) at /home/klugier/Pobrane/oxygen-gtk/src/oxygencache.h:237
05-Oxygen::ColorUtils::backgroundTopColor(color) at /home/klugier/Pobrane/oxygen-gtk/src/oxygencolorutils.cpp:137
06-Oxygen::StyleHelper::verticalGradient(this,base,height) at /home/klugier/Pobrane/oxygen-gtk/src/oxygenstylehelper.cpp:355
07-Oxygen::Style::renderBackgroundGradient(this,context,window,widget,clipRect,x,y,w,h,options,isMaximized) at /home/klugier/Pobrane/oxygen-gtk/src/oxygenstyle.cpp:454
08-Oxygen::Style::renderWindowBackground(this,context,window,widget,clipRect,x,y,w,h,options,isMaximized) at /home/klugier/Pobrane/oxygen-gtk/src/oxygenstyle.cpp:275
09-Oxygen::Style::renderWindowBackground(this,window,widget,r,x,y,w,h,o) at /home/klugier/Pobrane/oxygen-gtk/src/oxygenstyle.h:149
10-Oxygen::draw_flat_box(style,window,state,shadow,clipRect,widget,detail,x,y,w,h) at /home/klugier/Pobrane/oxygen-gtk/src/oxygenstylewrapper.cpp:190
11-?? at <unknown>:<unknown>
12-?? at <unknown>:<unknown>
13-g_closure_invoke at <unknown>:<unknown>
14-?? at <unknown>:<unknown>
15-g_signal_emit_valist at <unknown>:<unknown>
16-g_signal_emit at <unknown>:<unknown>
17-gtk_widget_realize at <unknown>:<unknown>
18-?? at <unknown>:<unknown>
19-?? at <unknown>:<unknown>
20-g_signal_emit_valist at <unknown>:<unknown>
21-g_signal_emit at <unknown>:<unknown>
22-gtk_widget_show at <unknown>:<unknown>
23-gtk_dialog_run at <unknown>:<unknown>
24-Upp::Ctrl::PanicMsgBox(title,text) at /home/klugier/upp/uppsrc/CtrlCore/GtkApp.cpp:32
25-Upp::PanicMessageBox(title,text) at /home/klugier/upp/uppsrc/Core/Util.cpp:29
26-Upp::Panic(msg) at /home/klugier/upp/uppsrc/Core/Util.cpp:99

27-Upp::MemoryDumpLeaks at /home/klugier/upp/uppsrc/Core/heapdbg.cpp:201
28-MemDiagCls::~MemDiagCls(this) at /home/klugier/upp/uppsrc/Core/Core.h:360
29-__run_exit_handlers(status,listp,run_list_atexit) at exit.c:82
30-__GI_exit(status) at exit.c:104
31-__libc_start_main(main,argc,argv,init,fini,rtld_fini,stack_end) at libc-start.c:321
32-_start at <unknown>:<unknown>

P.S.
I build oxygen-gtk from source code (This is why I have debug symbols).

#4 - 02/20/2015 05:45 PM - Miroslav Fidler

Well, so it is the very same as for me.

After further analysis, the real problem is that the leak checker gets run after the last global destructor. Qt being C++, it is quite illegal to be called from GTK at that point.

My solution to the crashing problem is to use 'xmessage' instead of GTK dialog to display the error. That is the problem #1. #2 is to find what call is causing the leak and sanitize it with ignore... (but GTK chameleon calls are already sanitized, so it must be something else, unfortunately).

#5 - 02/21/2015 06:55 PM - Miroslav Fidler

- Status changed from New to Ready for QA
- Assignee changed from Miroslav Fidler to Zbigniew Rebacz

Note: To fix memory leaks, I have sanitized window creation.... (I mean, memory leaks in Create, probably caused by Oxygen, are now ignored).

#6 - 02/21/2015 07:53 PM - Zbigniew Rebacz

- Status changed from Ready for QA to Approved

No errors on simply example. New dialog subsystem works correctly (Tested with kdialog).

Files

KDEGtkAppHangsOnDebug.png	224 KB	02/18/2015	Zbigniew Rebacz