Subject: Raster::Line segfaults ... sometimes.
Posted by rainbowsally on Mon, 22 Dec 2014 07:04:36 GMT
View Forum Message <> Reply to Message

Adding a static counter somehow fixed the issue (for me). And then considering that something might be choking on the cpu usage in this thread, I tried a 'usleep()' which also works, and seems to be more consistent.

I am multithreading in linux. You might not have this problem.

In uppsrc/Draw/Raster.cpp I was getting seg faults ("illegal memory access") so I put a counter in the function noted below. And just adding a static counter fixed the problem.

Huh????

[Update: Well, it fixed it a little more than half the time.]

If you are having trouble with RasterTest try this. I can't see how it could do anything but add a delay (unless there's a bug in BLITZ), but I don't want to experiment any more with this at this time.

File: uppsrc/Draw/RasterLine.cpp

```
void Raster::Line::MakeRGBA() const
{
// -rs added these two lines
//  static int cnt;
//  cnt++;
// -rs replaced the above with this ONE line.. another "experiment".
    usleep(500); // half a millisecond

 ASSERT(fmtdata && raster);
 int cx = raster->GetWidth();
// ...
```

Tested, 5 times, no crashes. I'm done for now.

The half millisecond delay gives the OS plenty of time to pass a time slice to other threads. Dunno what's causing this or where the best place to deal with this is, but it might be that a "io pause" something like this should be in the main event loop. At least in Linux.

May look into this later.

Subject: Re: Raster::Line segfaults ... sometimes.

Posted by Didier on Mon, 22 Dec 2014 19:34:31 GMT
View Forum Message <> Reply to Message

Hello Rainbowsally,

Quote:And just adding a static counter fixed the problem.
Huh????.
When you encounter such behaviour : modification of an unrelated code that causes a correction (or a crash), this is mostly due to uninitialized variables (somewhere in the code ....).
I recommend launching valgrind on this : info should be very instructive.

Subject: Re: Raster::Line segfaults ... sometimes.
Posted by rainbowsally on Tue, 23 Dec 2014 08:24:12 GMT
View Forum Message <> Reply to Message

Hi Didier.

Didier wrote on Mon, 22 December 2014 20:34Hello Rainbowsally,

Quote:And just adding a static counter fixed the problem.
Huh????.
When you encounter such behaviour : modification of an unrelated code that causes a correction (or a crash), this is mostly due to uninitialized variables (somewhere in the code ....).
I recommend launching valgrind on this : info should be very instructive.

I'm using Linux Mint 32-bit with 64 bit dual core CPU.

The problem was intermittent after I included the static variable and counter (to use it so it couldn't be optimized out).

I ran it in an external debugger (kdbg) and it never crashed. This lack of crashing in the debugger but crashing from the command line or when clicked with a mouse suggested to me that it might be a problem with thread timing.

Thank you for the note and an opportunity to describe the situation more fully.

IF you guys are experiencing intermittent crashes with multithreading apps, do consider passing control back to the operating system from time to time using a simple Sleep(1) or usleep(N) inside the thread's "run" loop.

Easier to disable multi-threading? True. But that may not be necessary.

The Raster example code has been around for so long it seems very very unlikely that it's a memory leak, although that could account for the variables being set up by previous calls if the memory is not re-initialized. But that then begs the question of how the first app crashed and left

its variables set and the second app failed to overwrite them.

Subject: Re: Raster::Line segfaults ... sometimes.
Posted by mirek on Wed, 24 Dec 2014 11:18:12 GMT
View Forum Message <> Reply to Message

rainbowsally wrote on Mon, 22 December 2014 08:04Adding a static counter somehow fixed the issue (for me). And then considering that something might be choking on the cpu usage in this thread, I tried a 'usleep()' which also works, and seems to be more consistent.

I am multithreading in linux. You might not have this problem.

In uppsrc/Draw/Raster.cpp I was getting seg faults ("illegal memory access") so I put a counter in the function noted below. And just adding a static counter fixed the problem.

Please, such post is sort of useless. Look here

http://www.ultimatepp.org/www$uppweb$community$en-us.html

You are not giving me a single clue about what problem you have encountered.

(Just to check: Are you aware that you have to use Mutexes and such to serialize access to shared resources?)

Mirek

Subject: Re: Raster::Line segfaults ... sometimes.
Posted by rainbowsally on Thu, 25 Dec 2014 14:03:55 GMT
View Forum Message <> Reply to Message

Mirek, you couldn't have duplicated the problem because I wasn't doing MT right.

Still, it's interesting in that it is possible that the X message que overflowed.

I don't know why it would cause a memory access violation in another thread... but that's what I will attempt to figure out if that is indeed what happened.

It's not a bug report. I'm not asking for help solving it.

Subject: Re: Raster::Line segfaults ... sometimes.
Posted by mirek on Thu, 25 Dec 2014 20:00:29 GMT
View Forum Message <> Reply to Message

rainbowsally wrote on Thu, 25 December 2014 15:03Mirek, you couldn't have duplicated the problem because I wasn't doing MT right.

Still, it's interesting in that it is possible that the X message que overflowed.

I don't know why it would cause a memory access violation in another thread... but that's what I will attempt to figure out if that is indeed what happened.

It's not a bug report. I'm not asking for help solving it.

Well, sounded like pretty serious issue... I cannot afford to leave such things (like "something segfaults") around