QGIS Application - Bug report #6505
Segfault on Mac OS X with QgsAtlasComposition unit test

2012-10-12 01:44 PM - Larry Shaffer

Status:	Closed		
Priority:	High		
Assignee:			
Category:			
Affected QGIS version:	master	**Regression?:**	No
Operating System:	Mac OS X	**Easy fix?:**	No
Pull Request or Patch supplied:	No	**Resolution:**	
Crashes QGIS or corrupts data:	Yes	**Copied to github as #:**	15724

Description

Crash of Python executable when running tests that include QgsAtlasComposition::prepareForFeature() method:

specifically the provider->featureAtId() at line #166 source:src/core/composer/qgsatlascomposition.cpp#L166

Unit test log output: http://dash.orfeo-toolbox.org/testDetails.php?test=12507606&build=98288

Crash report attached.

Associated revisions

Revision 332ce329 - 2012-10-23 01:30 AM - Larry Shaffer

Fix #6505, set typedef for Mac size_t in bindings

History

#1 - 2012-10-15 12:18 AM - Hugo Mercier

Do you have the same problem with the C++ unit test ? (output/bin/qgis_atlascompositiontest)

#2 - 2012-10-15 01:55 PM - Larry Shaffer

- Status changed from Open to Feedback

Hi Larry,

Thanks for your bug report.

Do you have the same problem with the C++ unit test ?

Unfornataly I do not have any Mac computer to test on.
I've looked at the code carefully but cannot see anything that may lead to that problem.
If you could add these three lines, that could help diagnose (line 188 of qgsatlascomposition.cpp):

```
QgsDebugMsg( QString( "provider: %1" ).arg( (long)provider, sizeof(void*), 16 ) );
QgsDebugMsg( QString( "feature_I: %1" ).arg( featureI ) );
QgsDebugMsg( QString( "feature id: %1" ).arg( mFeatureIds[featureI] ) );
```

> *(adding #include "qgslogger.h" at the beginning of the file)*
>
> *Thanks*
>
> --
> *Hugo Mercier*
> *Oslandia*

Hi Hugo,

The crash happens at line# 167 (i.e., provider->featureAtId(...)), so I placed the QgsDebugMsg lines before that call. Everything looks fine with the C++ test (for first feature of 4):

23: QDEBUG : TestQgsAtlasComposition::filename() src/core/composer/qgsatlascomposition.cpp: 167: (prepareForFeature) provider: 7fcbafaee8d0
23: QDEBUG : TestQgsAtlasComposition::filename() src/core/composer/qgsatlascomposition.cpp: 168: (prepareForFeature) feature_I: 0
23: QDEBUG : TestQgsAtlasComposition::filename() src/core/composer/qgsatlascomposition.cpp: 169: (prepareForFeature) feature id: 0

However, for the Python test the feature id is way off:

51: src/core/composer/qgsatlascomposition.cpp: 167: (prepareForFeature) provider: 7f97b4bef990
51: src/core/composer/qgsatlascomposition.cpp: 168: (prepareForFeature) feature_I: 4294967296

Doesn't produce the 3rd line of debug output, because mFeatureIds[featureI] is probably out of index. :^) So, it crashes there.

Something to do with the **typedef unsigned int size_t**, as defined in python/core/qgsgeometry.sip not being compatible with Mac. Hinted to here:

http://lists.osgeo.org/pipermail/qgis-developer/2008-November/005182.html

I hunted down the typedef for size_t (in system header) for Mac OS X 10.6, 10.7, 10.8 (the currently supported OS versions) and it's the same on all (32 or 64 bit):

```
MacOSX10.x.sdk/usr/include/i386/_types.h

#if defined(__GNUC__) && defined(__SIZE_TYPE__)
typedef __SIZE_TYPE__        __darwin_size_t;    /* sizeof() */
#else
typedef unsigned long        __darwin_size_t;    /* sizeof() */
#endif
```

Even though its under i386, it applies to 64 bit Macs as well (in include/machine/_types.h). Guessing that __SIZE_TYPE__ is not generally pre-defined, and defaults to an unsigned long, I changed the following in python/core/qgsgeometry.sip:

```
%If (WS_MACX)
typedef unsigned long size_t;
%End
%If (!WS_MACX)
typedef unsigned int size_t;
%End
```

I added some debug code to cmake/SIPMacros.cmake to see what -t flags were being sent to sip binary for building the bindings and found the

'WS_MACX' Platform tag.

http://www.riverbankcomputing.co.uk/static/Docs/sip4/directives.html#directive-%Platforms

Now both tests pass fine. However, **the Python test took 10 times longer to complete** (15 seconds for C++ and 155 for Python, on an older Mac Pro). That is by far the longest running test on the Mac. Not sure why it is so much longer under Python??

Need to test more on other Mac platforms and talk to some sip gurus before committing this.

Basically, it doesn't look like your code is causing the problem, just showing the symptoms of what a appears to be a longstanding issue. It just didn't show up until some of the Python bindings resolved to the size_t ctype on Mac.

Thanks for helping with this.

#3 - 2012-10-18 03:55 PM - Larry Shaffer

After some more research, I have found two ways of fixing this:

A) Pass in CMake platform varialbes (as -t tags) to the sip command for generating bindings, and use that in the following manner:

In SIPMacros.cmake

```
SET(_sip_tags)
IF (APPLE)
  LIST(APPEND _sip_tags -t APPLE)
ELSEIF (UNIX)
  LIST(APPEND _sip_tags -t UNIX_NOTAPPLE)
ELSEIF (WIN32)
  LIST(APPEND _sip_tags -t WIN32)
ENDIF (APPLE)
```

In python/core/qgsgeometry.sip

```
%Platforms {UNIX_NOTAPPLE APPLE WIN32}
%If (APPLE)
typedef unsigned long size_t;
%End
%If (!APPLE)
typedef unsigned int size_t;
%End
```

This would bypass the PyQt tags and use a set of already-known-to-be-valid platform tags from CMake. Although, the Platforms defined by PyQt4 or sip could also be used.

B) Delete typedef unsigned int size_t; from python/core/qgsgeometry.sip and in cmake/FindSIP.cmake run a Python script that figures out what size_t and ssize_t are and generate a sip file to typedef them appropriately. It can be done with Python ctypes package with basic comparison tests (on my Mac with Py 2.7.1):

```
import ctypes
```

```
ctypes.c_size_t == ctypes.c_ulong
# returns True
ctypes.c_size_t == ctypes.c_uint
# returns False
```

Whether this 'size_t.sip' file should be included early in the bindings (so it applies to all) is up for debate. I don't know enough about whether it will only limit the size_t for the qgsgeometry.sip file, or all sip files included after it.

See this PyQt mailing list thread for more details: http://python.6.n6.nabble.com/Platforms-directive-or-tags-or-both-td4992626.html

This brings up a question: **does size_t really need defined there at all?** Martin mentioned it was put in place because the '32bit integer gives us a limit of 4GB per geometry's wkb.' If the user tries to load a larger geometry through Python, won't QGIS crash without any indication as to why? How would that be different than not imposing a limit and having the app crash *if* it can't handle the size?

#4 - 2012-10-22 04:31 PM - Larry Shaffer

- Status changed from Feedback to Closed

Fixed in changeset commit:"332ce329afb61609f1f224eec41b552535437dfe".

Files

Python_2012-10-12-083547_larrys-imac-crash.txt	54.4 KB	2012-10-12	Larry Shaffer