GammaLib - Bug #1563

GammaLib Python check does not work on El Capitan (Mac OS 10.11)

10/30/2015 06:11 PM - Knödlseder Jürgen

Status:	Closed	**Start date:**	10/30/2015
Priority:	Normal	**Due date:**	
Assigned To:	Knödlseder Jürgen	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

This bug has been reported earlier by Fabio Acero how got the following during the Python test:

```
FAIL: test_python.py
====================

Traceback (most recent call last):
  File "./test_python.py", line 21, in <module>
    from gammalib import *
  File "/Users/facero/Documents/Work/Program/ctools/gammalib/pyext/gammalib/__init__.py", line 4, in <module>
    from gammalib.app import *
  File "/Users/facero/Documents/Work/Program/ctools/gammalib/pyext/gammalib/app.py", line 28, in <module>
    _app = swig_import_helper()
  File "/Users/facero/Documents/Work/Program/ctools/gammalib/pyext/gammalib/app.py", line 20, in swig_import_helper
    import _app
ImportError: dlopen(/Users/facero/Documents/Work/Program/ctools/gammalib/pyext/build/gammalib/_app.so, 2): Library not loaded:
 libreadline.6.2.dylib
  Referenced from: /Users/facero/Documents/Work/Program/ctools/gammalib/pyext/build/gammalib/_app.so
  Reason: image not found
```

Fabio works under Mac OS 10.10.

I have encountered the same problem on a fresh version on El Capitan (10.11) after installing the following using Homebrew:

```
brew install automake
brew install libtool
brew install cfitsio
brew install swig
```

My understanding is that the Python binary is a universal binary (i386 and x86_64 targets), but the GammaLib C++ library is only compiled using the x84_64 target. The GammaLib Python extension is however built for the x86_64 and i386 targets since Python was built for these targets:

```
c++ -bundle -undefined dynamic_lookup -arch i386 -arch x86_64 -Wl,-F. build/temp.macosx-10.11-intel-2.7/gammalib/app_wrap.o
 -L../src/.libs -L../src/.libs -lgamma -lcfitsio -lreadline -lncurses -o build/lib.macosx-10.11-intel-2.7/gammalib/_app.so
ld: warning: ignoring file ../src/.libs/libgamma.dylib, file was built for x86_64 which is not the architecture being linked (i386):
 ../src/.libs/libgamma.dylib
ld: warning: ignoring file /usr/local/lib/libcfitsio.dylib, file was built for x86_64 which is not the architecture being linked (i386):
 /usr/local/lib/libcfitsio.dylib
building '_base' extension
```

Now trying to compile GammaLib for both targets works but cannot use cfitsio, as cfitsio installed through Homebrew is only for the x86_64 target.
Seems to be a tricky problem.

History

#1 - 10/30/2015 06:37 PM - Knödlseder Jürgen

Contrary to my suspicion, the problem was is the different targets, it is the fact that somehow it expects to find the gammalib library in /Users/cta/test/install:

```
$ otool -L /Users/cta/test/gammalib/pyext/build/gammalib/_app.so
/Users/cta/test/gammalib/pyext/build/gammalib/_app.so:
    /Users/cta/test/install/lib/libgamma.0.dylib (compatibility version 1.0.0, current version 1.0.0)
    /usr/local/opt/cfitsio/lib/libcfitsio.2.dylib (compatibility version 2.0.0, current version 2.3.37)
    /usr/lib/libedit.3.dylib (compatibility version 2.0.0, current version 3.0.0)
    /usr/lib/libncurses.5.4.dylib (compatibility version 5.4.0, current version 5.4.0)
    /usr/lib/libc++.1.dylib (compatibility version 1.0.0, current version 120.1.0)
    /usr/lib/libSystem.B.dylib (compatibility version 1.0.0, current version 1225.1.1)
```

I'm not sure why he wants the install directory. Initially this does not exist, hence he complains. After typing make install, make test works.

#2 - 10/30/2015 06:38 PM - Knödlseder Jürgen

- Status changed from New to In Progress

- Assigned To set to Knödlseder Jürgen

- % Done changed from 0 to 10

#3 - 10/30/2015 08:48 PM - Knödlseder Jürgen

- % Done changed from 10 to 20

Interestingly, the GammaLib C++ library itself is looked for in the install directory:

```
$ otool -L src/.libs/libgamma.0.dylib
src/.libs/libgamma.0.dylib:
    /Users/cta/test/install/lib/libgamma.0.dylib (compatibility version 1.0.0, current version 1.0.0)
    /usr/lib/libedit.3.dylib (compatibility version 2.0.0, current version 3.0.0)
    /usr/lib/libncurses.5.4.dylib (compatibility version 5.4.0, current version 5.4.0)
    /usr/local/opt/cfitsio/lib/libcfitsio.2.dylib (compatibility version 2.0.0, current version 2.3.37)
    /usr/lib/libSystem.B.dylib (compatibility version 1.0.0, current version 1225.1.1)
```

How can then the unit tests work?

Indeed, calling directly

```
$ ./test/.libs/test_CTA
dyld: Library not loaded: /Users/cta/test/install/lib/libgamma.0.dylib
  Referenced from: /Users/cta/test/gammalib/./test/.libs/test_CTA
  Reason: image not found
Trace/BPT trap: 5
```

leads to the same problem as for the Python test, but looking into the test_CTA it turns out that the system sets first the DYLD_LIBRARY_PATH before calling the binary, and indeed

```
Mac-OS-X-1011:gammalib cta$ export DYLD_LIBRARY_PATH="/Users/cta/test/gammalib/src/.libs:$DYLD_LIBRARY_PATH"
```

```
Mac-OS-X-1011:gammalib cta$ ./test/.libs/test_CTA

******************************************
* CTA instrument specific class testing *
******************************************
Test response: ..... ok
Test effective area: .. ok
Test PSF: ..... ok
Test King profile PSF: ..... ok
Test integrated PSF: ..... ok
Test energy dispersion: ..... ok
```

works.

#4 - 10/30/2015 09:48 PM - Knödlseder Jürgen

So somehow this is linked to the way how DYLD_LIBRARY_PATH is handled by El Capitan, we don't seem the be the only ones having trouble with that:

- https://community.lsst.org/t/thinking-of-upgrading-to-el-capitan/266
- https://forums.developer.apple.com/thread/7935
- https://forums.developer.apple.com/thread/13161

The last thread mentions explicitly the problem:

When a process is started, the kernel checks to see whether the main executable is protected on disk or is signed with an special system entitlement. If either is true, then a flag is set to denote that it is protected against modification. … Any dynamic linker (dyld) environment variables, such as DYLD_LIBRARY_PATH, are purged when launching protected processes.

Since all system-provided script interpreters (bash, perl, python, etc.) are protected processes, dyld environment variables are purged when you run any script (even if the script itself is not protected). Personally, I think this part is a bad policy or an oversight, but such is life.

Another important thing is mentioned on

- https://github.com/rust-lang/cargo/issues/1816

You can still use DYLD_LIBRARY_PATH if it is used in the same shell, but you can no longer export it to subshells. It is a bit hard to describe. So calling the script directly using the ruby binary works. But spawning ruby via a shebang doesn't. So you can still use DYLD_LIBRARY_PATH as long as you don't use shell wrappers, or you need to set it from within the shell wrappers.

This may explain why the C++ tests work that are started using exec while the trick does not work for the Python test.

See also:

- https://forums.developer.apple.com/message/31148
- https://github.com/kubo/fix_oralib_osx

#5 - 10/30/2015 09:50 PM - Knödlseder Jürgen

Note that this problem only affects the unit test of the Python module when the code is not yet installed. After installing GammaLib there is no problem anymore. This is a workaround for the problem that should be put in the Known Problems section of the documentation.

#6 - 11/19/2015 12:45 AM - Knödlseder Jürgen

- % Done changed from 20 to 80

I finally found a solution that seems to work.

I added the following code to the pyext/Makefile.am that makes use of install_name_tool and that uses the rpath symbol instead of an absolute path. Both the install path but also the path for the uninstalled version are added using the add_rpath option. Note that the IS_ELCAPITAN symbol is used to do the post processing only in case that El Capitan has been detected (or newer). This symbol is set in configure.ac:

```
# Build the gammalib extension module
build: $(BUILT_SOURCES)
	@PYTHON_BUILD_PREFIX@ $(PYTHON) setup.py build_ext
if IS_ELCAPITAN
	for f in build/gammalib/_*.so; do \
		echo "Post process module "$$f; \
		install_name_tool -change $(libdir)/libgamma.0.dylib @rpath/libgamma.0.dylib $$f; \
		install_name_tool -add_rpath $(libdir) $$f; \
		install_name_tool -add_rpath $(abs_top_srcdir)/src/.libs $$f; \
	done
endif
```

Note that I also adjusted setup.py.in to add the -headerpad_max_install_names option to the linker (this is needed as otherwise there may not be enough space in the library for path replacement):

```
if '@IS_ELCAPITAN_TRUE@' != '#':
    extra_link_args.append('-headerpad_max_install_names')
```

#7 - 11/19/2015 09:36 AM - Knödlseder Jürgen

- Status changed from In Progress to Closed

- % Done changed from 80 to 100

This is now fixed, also for ctools. Merged into devel.