

## GammaLib - Feature #1490

### Build binary conda packages for Gammalib and ctools

06/30/2015 01:15 PM - Deil Christoph

<b>Status:</b>	Closed	<b>Start date:</b>	09/27/2017
<b>Priority:</b>	Normal	<b>Due date:</b>	10/31/2017
<b>Assigned To:</b>	Knödseder Jürgen	<b>% Done:</b>	100%
<b>Category:</b>		<b>Estimated time:</b>	0.00 hour
<b>Target version:</b>	1.5.0		
<b>Description</b> Conda is a popular cross-platform package manager: <a href="http://conda.pydata.org/docs/">http://conda.pydata.org/docs/</a>  Anaconda / Miniconda has become the most popular Scientific Python distribution:  <ul style="list-style-type: none"><li>• <a href="https://store.continuum.io/cshop/anaconda/">https://store.continuum.io/cshop/anaconda/</a></li><li>• <a href="http://continuum.io/downloads">http://continuum.io/downloads</a></li><li>• <a href="http://conda.pydata.org/miniconda.html">http://conda.pydata.org/miniconda.html</a></li></ul> LSST is using it. Astropy and CIAO Sherpa are using it.  So ideally before 1.0, we should have Conda binary packages for Linux and Mac. Once we have packages, we can distribute them via <a href="https://anaconda.org/">https://anaconda.org/</a>  I have to investigate how to build the packages.  A common way is to use travis-ci as a free build service to build Mac / Linux packages and AppVeyor to build Windows packages. Example: <a href="https://github.com/astropy/conda-builder-affiliated">https://github.com/astropy/conda-builder-affiliated</a>  Step 1: I'll start investigating how this works and write down the commands and some docs. Step 2: Jürgen can add a make target for this and integrate the binary package build on the CI Mac / Linux servers? (probably better than relying on travis-ci?)			
<b>Subtasks:</b>			
Action # 2196: Validate GammaLib installation through conda			<b>Closed</b>
Action # 2211: Prepare packages with conda			<b>Closed</b>

#### History

##### #1 - 07/03/2015 10:28 AM - Knödseder Jürgen

- Target version deleted (1.0.0)

##### #2 - 07/03/2015 05:41 PM - Deil Christoph

Building binary conda packages is not as simple as I'd hoped.  
I'll continue investigating / trying to make this work for Gammalib / ctools.

Another example: [https://github.com/mjuric/lsst\\_packaging\\_conda](https://github.com/mjuric/lsst_packaging_conda)

##### #3 - 01/11/2016 10:17 PM - Knödseder Jürgen

- Start date deleted (06/30/2015)

- Release set to gammalib-1.1.0

##### #4 - 01/18/2016 03:24 PM - Knödseder Jürgen

- Target version set to 1.1.0

- Start date set to 01/18/2016

#### #5 - 06/23/2016 12:31 AM - Knödlseider Jürgen

- Assigned To changed from Deil Christoph to Knödlseider Jürgen
- Target version changed from 1.1.0 to 1.2.0

#### #6 - 03/03/2017 10:16 AM - Knödlseider Jürgen

- Target version changed from 1.2.0 to 1.3.0

#### #7 - 03/08/2017 12:03 PM - Kosack Karl

I've created conda packages for both ctools and gammalib, and they seem to work (at least in my quick tests). Could anyone with more ctools experience do some more complex tests?

So far I have only built packages for **macOS 64 bit with Python-3.6** (though it's trivial to do others, once it is tested). So you need an anaconda distribution with python 3.6. The easiest way to test is to run the following (again only on a mac!):

### create a virtual environment to test in and switch to it:

```
> conda create -n ctoolstest python=3.6 anaconda
> source activate ctoolstest
```

you can omit "anaconda" from the first line if you want to have a smaller env with only python + ctools for testing, otherwise you get the full anaconda dist with all scientific packages

### install the packages

(the packages are so far are only in my "kosack" channel on anaconda.org, until they are tested and then I will add them to "cta-observatory"):

```
> conda install -c kosack ctools
Fetching package metadata .....
Solving package specifications: .
```

Package plan for installation in environment /Users/kosack/anaconda/envs/ctools:

The following NEW packages will be INSTALLED:

```
cfitsio: 3.410-2 openastronomy
ctools: 1.2.0-py36_2 kosack
gammalib: 1.2.0-py36_1 kosack
openssl: 1.0.2k-1
pip: 9.0.1-py36_1
python: 3.6.0-0
readline: 6.2-2
setuptools: 27.2.0-py36_0
sqlite: 3.13.0-0
tk: 8.5.18-0
wheel: 0.29.0-py36_0
xz: 5.2.2-1
zlib: 1.2.8-3
```

Proceed ([y]/n)?

That should fetch ctools and gammalib (1.2.0), install them to the virtual environment.

Make sure you do **not** set paths for PYTHON\_PATH or GAMMALIB, etc, since they are not needed and will interfere with the installation.

Now, you should have a full ctools/gammalib install with python bindings! no gammalib-init.sh is needed.

### Try it:

```
> ctbin --help
```

```
> which ctbin
/Users/kosack/anaconda/envs/ctoolstest/bin/ctbin

> python
PYTHON> from gammalib import cta
PYTHON> help(cta)
```

#### #8 - 03/08/2017 12:12 PM - Kosack Karl

- Status changed from New to In Progress

if this looks ok, it would be nice for somebody to modify the ctools/gammalib continuous integration system to build the packages automatically. The package definitions are in: <https://github.com/cta-observatory/cta-conda-recipes>

The procedure would be something like this on each machine type (macOS, Linux):

```
git clone https://github.com/cta-observatory/cta-conda-recipes
cd cta-conda-recipes
```

```
for pyver in 2.7 3.4 3.5 3.6; do conda build gammalib-1.2.0.conda --python=$pyver; done
for pyver in 2.7 3.4 3.5 3.6; do conda build ctools-1.2.0.conda --python=$pyver; done
```

```
anaconda upload <path-to-conda-build-dir> gammalib.* ...
```

#### #9 - 03/08/2017 12:27 PM - Kosack Karl

one more comment: you may need the cfitsio package, since I require it in the definition. You can get it via: `conda install -c OpenAstronomy cfitsio`. Eventually that can be added to the same cta-observatory channel, so only one channel needs to be added.

#### #10 - 03/08/2017 02:55 PM - Kosack Karl

A few issues:

- the user shouldn't have to set \$CTOOLS or \$GAMMALIB variables, since it is installed in a common place, but so far the code expects them to be there. Setting them both to \$CONDA\_PREFIX works, but this should not be necessary (though it seems to be a fundamental problem of how gammalib/ctools expect to be installed). That means the user must manually still set some variables after installing the package (unlike any other package)
- even with CTOOLS set correctly, no help info is found (e.g. `ctbin --help` says "no help available"). Seems to be that the manpages don't get included in the package. Is that where tools get their help info? or do they look for some other file?

**#11 - 03/08/2017 03:00 PM - Knödlseider Jürgen**

I guess the point is that the setup scripts need to be run, typically:

```
export GAMMALIB=/usr/local/gamma
source $GAMMALIB/bin/gammalib-init.sh
export CTOOLS=/usr/local/gamma
source $CTOOLS/bin/ctools-init.sh
```

**#12 - 03/09/2017 10:50 AM - Kosack Karl**

guess the point is that the setup scripts need to be run, typically:

Yes, we just need to think how to do it correctly, since there is no "gamma" directory anymore, and the files can move around. With the packages, the software is installed in either `/path/to/anaconda/{lib,share,bin,include}` (if installed in the base environment) or in `$CONDA_PREFIX/{lib,share,bin,include}` if it is in a virtual environment, where `CONDA_PREFIX` depends on the name of the environment the user gave. It's the same as if you had an RPM that installs it in `/usr/`. Therefore the location depends on the virtual environment name.

No other packages require you to set environment variables to find their support data (and packages know nothing about conda internally), so there must be a nicer mechanism to do it without having the user type `"export CTOOLS=$CONDA_PREFIX"` every time they want to use it.

I know at least `PATH`, `LD_LIBRARY_PATH`, etc get updated automatically, but I'm not sure if there is some standard for finding out if support files are in `/usr/share` or `/usr/local/share`, etc.

**#13 - 06/06/2017 10:30 PM - Knödlseider Jürgen**

- Target version changed from 1.3.0 to 1.4.0
- % Done changed from 0 to 50

**#14 - 07/31/2017 11:08 PM - Knödlseider Jürgen**

- Target version changed from 1.4.0 to 1.5.0

**#15 - 11/23/2017 05:32 PM - Knödlseider Jürgen**

- Status changed from In Progress to Closed

I solved the problem with the environment variables (conda in facts supports environment variables, this can be done via an activation script in

`./etc/conda/activate.d.`

Conda package generation is now included in the release pipeline (#2278)