Conda Documentation

Release latest

August 09, 2015

Contents

Conda is a cross-platform, Python-agnostic binary package manager. It is the package manager used by Anaconda installations, but it may be used for other systems as well. Conda makes environments first-class citizens, making it easy to create independent environments even for C libraries. Conda is written entirely in Python, and is BSD licensed open source.

CHAPTER 1

Installation

Conda is a part of the Anaconda distribution. You can also download a minimal installation that only includes conda and its dependencies, called Miniconda.

CHAPTER 2

Getting Started

If you install Anaconda, you will already have hundreds of packages installed. You can see what packages are installed by running

```
$ conda list
```

to see all the packages that are available, use

```
$ conda search
```

and to install a package, use

```
$ conda install <package-name>
```

The real power of conda comes from its ability to manage environments. In conda, an environment can be thought of as a completely separate installation. Conda installs packages into environments efficiently using hard links by default when it is possible, so environments are space efficient, and take seconds to create.

The default environment, which `conda` itself is installed into is called `root`. To create another environment, use the `conda create` command. For instance, to create an environment with the IPython notebook and NumPy 1.6, which is older than the version that comes with Anaconda by default, you would run

```
$ conda create -n numpy16 ipython-notebook numpy=1.6
```

This creates an environment called `numpy16` with the latest version of the IPython notebook, NumPy 1.6, and their dependencies.

We can now activate this environment, use

```
# On Linux and Mac OS X
$ source activate numpy16

# On Windows
> activate numpy16
```

This puts the bin directory of the `numpy16` environment in the front of the `PATH`, and sets it as the default environment for all subsequent conda commands.

To go back to the root environment, use

```
# On Linux and Mac OS X
$ source deactivate

# On Windows
> deactivate
```

CHAPTER 3

Building Your Own Packages

You can easily build your own packages for conda, and upload them to anaconda.org, a free service for hosting packages for conda, as well as other package managers. To build a package, create a recipe. See http://github.com/conda/conda-recipes for many example recipes, and http://docs.continuum.io/conda/build.html for documentation on how to build recipes.

To upload to anaconda.org, create an account. Then, install the anaconda-client and login

```
$ conda install anaconda-client
$ anaconda login
```

Then, after you build your recipe

```
$ conda build <recipe-dir>
```

you will be prompted to upload to anaconda.org.

To add your anaconda.org channel, or the channel of others to conda so that `conda install` will find and install their packages, run

```
$ conda config --add channels https://conda.anaconda.org/username
```

(replacing `username` with the user name of the person whose channel you want to add).

CHAPTER 4

Getting Help

The documentation for conda is at http://conda.pydata.org/docs/. You can subscribe to the conda mailing list. The source code and issue tracker for conda are on GitHub.

CHAPTER 5

Contributing

Contributions to conda are welcome. Just fork the GitHub repository and send a pull request.

To develop on conda, the easiest way is to use `python setup.py develop` in your root conda environment. This will install a link to the local conda source code, so that any change you make to conda will be instantly available. To undo this, run `python setup.py develop -u`. If you are worried about breaking your conda installation, you can install a separate instance of Miniconda and work off it. This is also the only way to test conda in both Python 2 and Python 3, as conda can only be installed into a root environment.

Run the conda tests by `conda install pytest` and then running `py.test` in the conda directory. The tests are also run by Travis CI when you make a pull request.