Python packaging: pip, conda, venv

Matthew Rich

Northwestern IT Research Computing Services

Pip, what is it?

- “Pip installs packages”
- Pip is the official package manager for the Python language and PyPI (the Python Package Index).
- Pip downloads and installs Python packages from the official PyPI or alternative package indexes.
  - Packages can be in source or “wheel” format (a fancy zip file).

Installing with pip

```
$ pip install <package>
```

Re-Installing with pip

- Pip can save the set of installed packages for a particular Python environment into a text file (normally called `requirements.txt`).
- Pip can read a requirements.txt file and install all the packages listed.

Re-Installing with pip

```
$ pip freeze >requirements.txt
$ pip install -r requirements.txt
```

Pip and dependencies

- Pip does not have a dependency resolver.
- IE It has no way to figure out what software any package needs installed to work properly.
- Package maintainers can specify their dependencies in a special file that Pip reads, so it knows what other packages to download.

Virtualenv, what and why

- Virtualenv is a tool to create isolated Python environments.
- What if you want to use software with conflicting dependencies?
- E.g. You use Scipy 1.4 but you want to use a package that need functions from scipy.linalg that were removed in v1.0.0? Create separate virtualenvs.

Virtualenv makes your life easier

- Virtualenv allows users to install packages without needing to write to system locations or install Python themselves.
- Virtualenv lets you try out a package in its own environment, which you can easily delete
  - (pip is not good at uninstalling)

How virtualenv works

- Virtualenv creates a Python environment with its own physically separate installation directories
  - Nothing shared with other virtualenv environments
  - Nothing shared with system Python environment
- Virtualenvs need to be “activated”
  - This is basically just setting a couple environment variables that Python looks for.

Using virtualenv

```
# Create environment called “My-env”
$ virtualenv my-env
# same but specify which python to use:
$ virtualenv –p /path/to/python my-env
```

Using virtualenv

```
# activate for current session
$ source my-env/bin/activate
# disable for current session
$ deactivate
```

Conda, what and why

- Conda is a tool that manages environments (like virtualenv) and installs software (like pip).
- It can do this for any language, not just Python.
  - (but it's usually used for Python)
- Conda installs from "channels".
  - "default", but also "conda-forge", "bioconda", etc.
- Conda is especially useful on Windows!

Conda vs. Pip

- Conda can only install packages into conda environments, unlike Pip (which can install into any Python environment).
- Conda can install Python packages with external (IE, non-Python) dependencies, unlike Pip.
- Conda has a dependency resolver, unlike Pip.
- You can use pip inside a conda environment, but (in general) cannot use conda inside a virtualenv.

Using conda

```
# Create environment called “My-env”
$ conda create -n my-env python=3.8
# activate for current session
$ source activate my-env
# install a package
$ conda install <package>
# deactivate for current session
$ deactivate
```

Which tool should I use?

- Do you have an existing system Python installation and you want to install "pure-Python" packages that will use it?

- Use pip + virtualenv.

Which tool should I use?

- Do you need to install Python packages with external, non-python dependencies?

- use conda.

Which tool should I use?

- Do you need to use a different Python version than your system Python installation (or a different language altogether)?

- use conda.

Which tool should I use?

- Do you not have a system Python at all (e.g., Windows users)?

- use conda.

Which tool should I use?

- Do you want to install pure-Python packages into an isolated environment (e.g. to preserve a set of mutually-compatible packages)?

- Conda and pip + virtualenv are pretty much interchangeable.

Which tool should I use?

- In the scientific and data science Python communities, conda is the preferred standard.
- Pip+virtualenv are more often used by general business users, web developers, and hobbyists.

Thank you!

Hit me up with questions:

m-rich@northwestern.edu