Argelander-Institut für Astronomie

Setting up your own Python environment

Reinhold Schaaf

Argelander-Institut für Astronomie
der Universität Bonn

AIfA Technical Seminar – Oct. 24, 2014

Why would I need that?

- Python module not installed systemwide or
- You need newer (or older) version of a systemwide installed module
- AlfA's computer group will install Python modules only if available in Ubuntu repositories
- Example: PyYAML
  - Installed: 3.10
  - Available: 3.11

The hard way: setup.py

- Download source from somewhere
- Hopefully source comes with setup.py
  - Most modules do
  - Otherwise it may be harder
- `python setup.py` will try systemwide installation but
- `python setup.py --user` will install in `~/.local`
- `Sys.path` will be adjusted automatically
- Full story: docs.python.org/2/install

Easier: pip

- pypi.python.org hosts ~50.000 Python modules
- pip is installation tool for these modules
  - handles dependencies among modules
  - allows specification of version:

```
pip install PyYAML==3.11
```

- Full story: pip.pypa.io

Easier: pip

- `pip install` will try systemwide installation but
- `pip install --user` will install in `~/.local` (and adjust `sys.path`)
- To install alternative version of already installed module use

  `pip install --ignore-installed --user`
- `pip freeze` gives list of installed modules incl. version numbers

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

Most flexible: virtualenv

- virtualenv creates isolated Python environment including
  - Python executable
  - Setup tools (pip)
  - Library path where to install modules
- Allows use of well-defined, isolated environments side by side
- Full story: virtualenv.readthedocs.org

Most flexible: virtualenv

- `virtualenv <envname>` creates environment
  - `--python=<executable>` selects Python version
- Usage:
  - Run activate script to start environment
  - Use pip to install modules in the environment accessible only in there
- iPython:
  - `pip install ipython` in environment