SOFTWARE PACKAGING

Workshop on Advanced Techniques for Scientific Programming and Management of Open Source Software Packages

Beraldo Leal

April 21, 2015

"Talk is cheap. Show me the code."

- Linus Torvalds

INTRODUCTION

IF YOU WANNA SHARE:

- You need a package!

IF YOU WANNA SHARE:

- You need a package!
- Installation process should be as painless as possible for your user / contributor;

IF YOU WANNA SHARE:

- You need a package!
- Installation process should be as painless as possible for your user / contributor;
- Use a version control system (choose a modern one);

IF YOU WANNA SHARE:

- You need a package!
- Installation process should be as painless as possible for your user / contributor;
- Use a version control system (choose a modern one);
- Don’t send your code by email (or host at dropbox);

WHAT IS A PACKAGE ?

- We are not talking about python import package;
- A compressed (or not) file containing other files in a directory structure that will be installed on the target system;
- On steroids:
  - pre/post (un)install scripts;
  - documentation;
  - dependencies and libraries;
  - etc.

WHAT IS A PACKAGE ?

- Like a tarball file ?

WHAT IS A PACKAGE ?

- Like a tarball file ?
- Well, let's use something more sophisticated!

WHAT IS A PACKAGE ?

- Like a tarball file ?
- Well, let's use something more sophisticated!

DISTRIBUTIONS PACKAGE MANAGEMENT SYSTEM

- They try to follow the Filesystem Hierarchy Standard (FHS);

DISTRIBUTIONS PACKAGE MANAGEMENT SYSTEM

- They try to follow the Filesystem Hierarchy Standard (FHS);
- Can be binary or source package;

DISTRIBUTIONS PACKAGE MANAGEMENT SYSTEM

- They try to follow the Filesystem Hierarchy Standard (FHS);
- Can be binary or source package;
- Each distro has a different review process;

DISTRIBUTIONS PACKAGE MANAGEMENT SYSTEM

- They try to follow the Filesystem Hierarchy Standard (FHS);
- Can be binary or source package;
- Each distro has a different review process;
- .deb: any debian-like distro;

DISTRIBUTIONS PACKAGE MANAGEMENT SYSTEM

- They try to follow the Filesystem Hierarchy Standard (FHS);
- Can be binary or source package;
- Each distro has a different review process;
- .deb: any debian-like distro;
- .rpm: RedHat Package Manager;

DISTRIBUTIONS PACKAGE MANAGEMENT SYSTEM

- They try to follow the Filesystem Hierarchy Standard (FHS);
- Can be binary or source package;
- Each distro has a different review process;
- .deb: any debian-like distro;
- .rpm: RedHat Package Manager;
- ABS: Arch Build System;

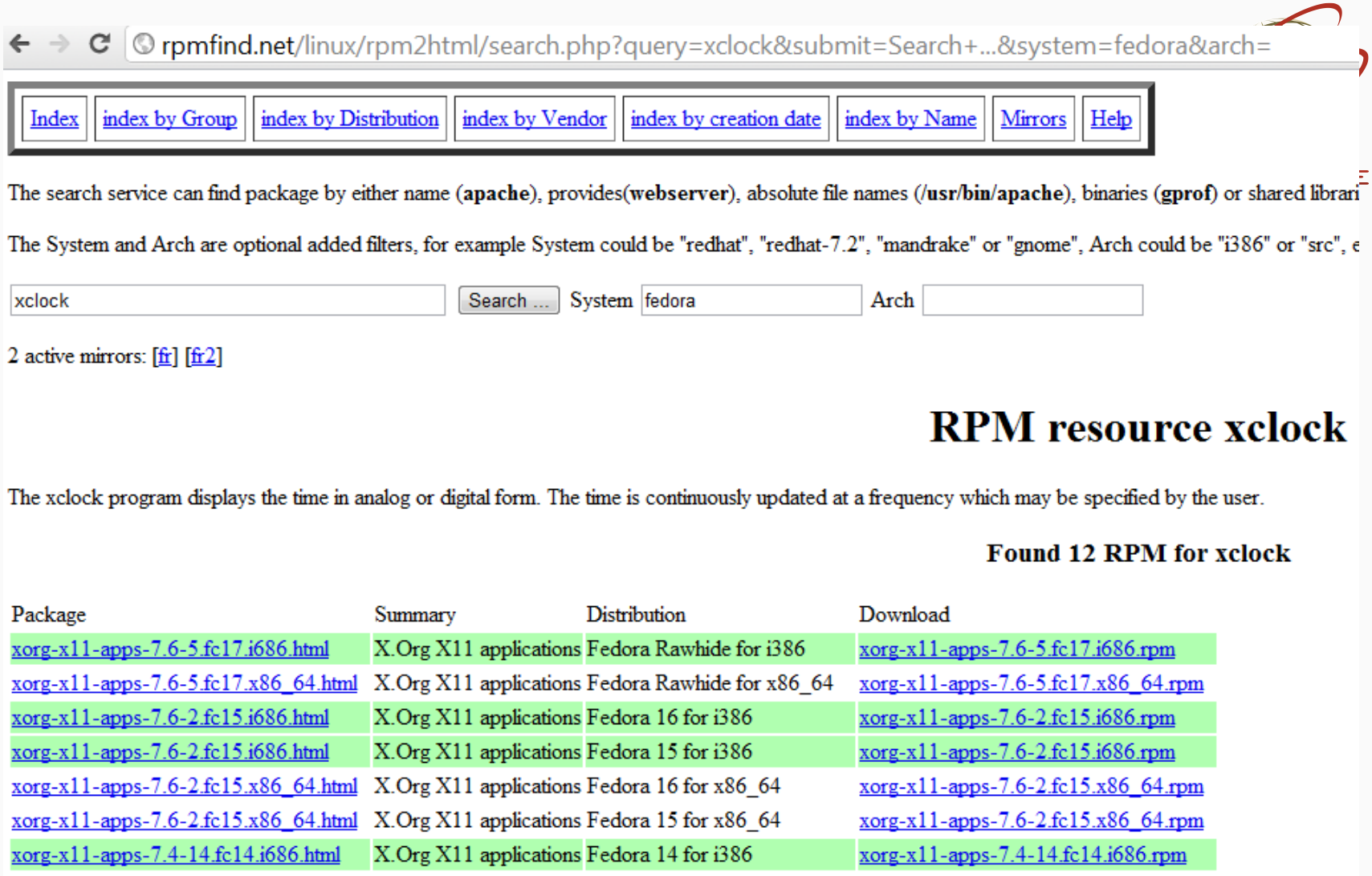
DISTRIBUTIONS PACKAGE MANAGEMENT SYSTEM

- They try to follow the Filesystem Hierarchy Standard (FHS);
- Can be binary or source package;
- Each distro has a different review process;
- .deb: any debian-like distro;
- .rpm: RedHat Package Manager;
- ABS: Arch Build System;
- Ports-like from BSD;

REPOSITORIES

- Did you rememeber rpmfind.net ?

REPOSITORIES



DEBIAN REPOSITORIES EXAMPLE

- three releases:
  - stable: contains the latest officially released distribution of Debian;
  - testing: contains packages that haven't been accepted into a "stable" release yet, but they are in the queue for that;
  - unstable (sid): is where active development of Debian occurs.

And about Python packages?

PYTHON PACKAGES: HISTORY

- 2000: distutils was added to the Python standard library in Python 1.6;

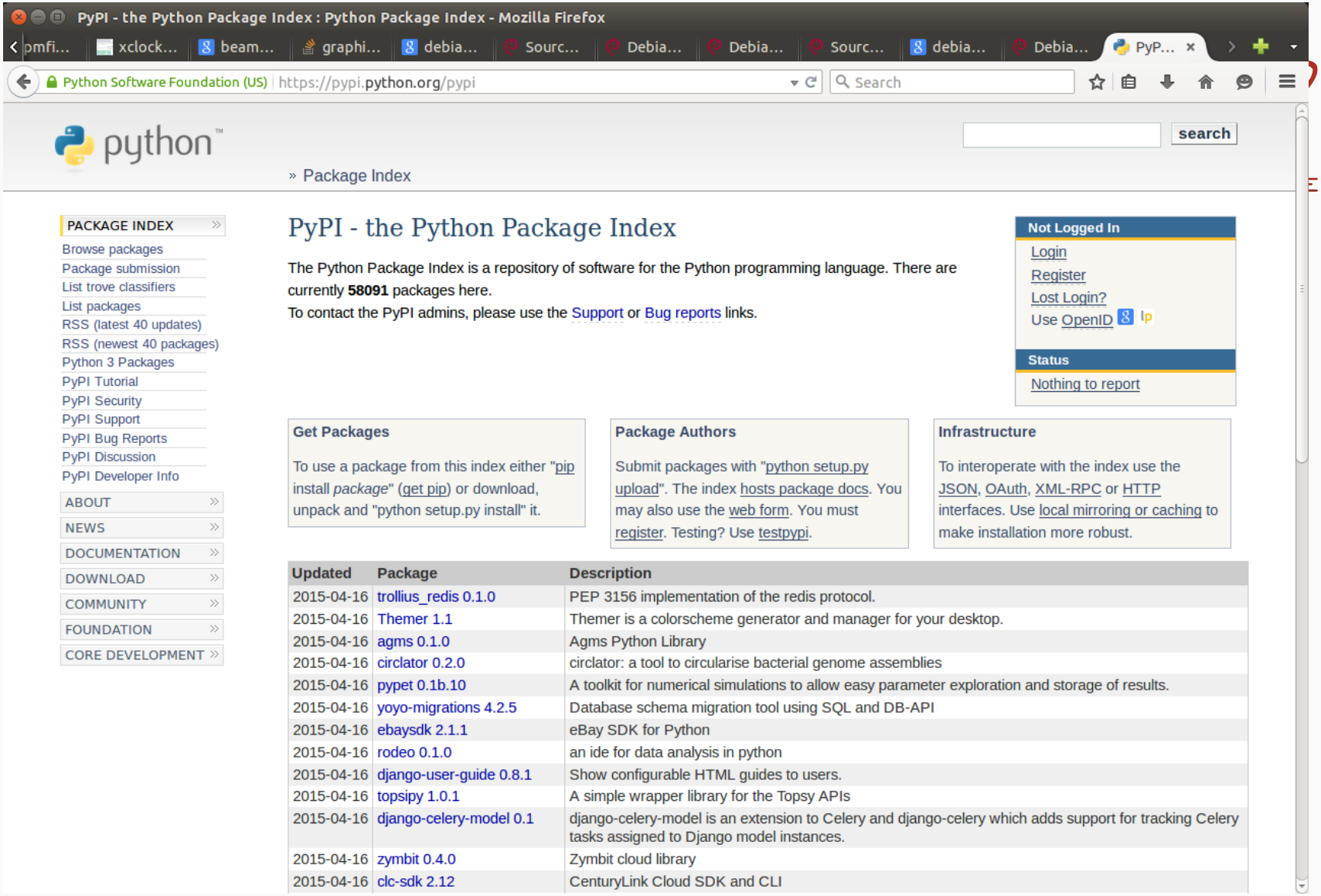
PYTHON PACKAGES: HISTORY

- 2000: distutils was added to the Python standard library in Python 1.6;
- 2003: PyPI was up and running;

PYTHON PACKAGES: HISTORY

- 2000: distutils was added to the Python standard library in Python 1.6;
- 2003: PyPI was up and running;
- To not be confuse: ”distribution” vs ”python package”.

PYPI


PyPI - the Python Package Index : Python Package Index - Mozilla Firefox
Python Software Foundation (US) https://pypi.python.org/pypi
Search
python
search
» Package Index
PACKAGE INDEX
Browse packages
Package submission
List trove classifiers
List packages
RSS (latest 40 updates)
RSS (newest 40 packages)
Python 3 Packages
PyPI Tutorial
PyPI Security
PyPI Support
PyPI Bug Reports
PyPI Discussion
PyPI Developer Info
ABOUT
NEWS
DOCUMENTATION
DOWNLOAD
COMMUNITY
FOUNDATION
CORE DEVELOPMENT
PyPI - the Python Package Index
The Python Package Index is a repository of software for the Python programming language. There are currently 58091 packages here.
To contact the PyPI admins, please use the Support or Bug reports links.
Not Logged In
Login
Register
Lost Login?
Use OpenID
Status
Nothing to report
Get Packages
To use a package from this index either "pip install package" (get pip) or download, unpack and "python setup.py install" it.
Package Authors
Submit packages with "python setup.py upload". The index hosts package docs. You may also use the web form. You must register. Testing? Use testpypi.
Infrastructure
To interoperate with the index use the JSON, OAuth, XML-RPC or HTTP interfaces. Use local mirroring or caching to make installation more robust.
Updated Package Description
2015-04-16 trollius_redis 0.1.0 PEP 3156 implementation of the redis protocol.
2015-04-16 Themer 1.1 Themer is a colorscheme generator and manager for your desktop.
2015-04-16 agms 0.1.0 Agms Python Library
2015-04-16 circlator 0.2.0 circlator: a tool to circularise bacterial genome assemblies
2015-04-16 pypet 0.1b.10 A toolkit for numerical simulations to allow easy parameter exploration and storage of results.
2015-04-16 yoyo-migrations 4.2.5 Database schema migration tool using SQL and DB-API
2015-04-16 ebaysdk 2.1.1 eBay SDK for Python
2015-04-16 rodeo 0.1.0 an ide for data analysis in python
2015-04-16 django-user-guide 0.8.1 Show configurable HTML guides to users.
2015-04-16 topsipy 1.0.1 A simple wrapper library for the Topsy APIs
2015-04-16 django-celery-model 0.1 django-celery-model is an extension to Celery and django-celery which adds support for tracking Celery tasks assigned to Django model instances.
2015-04-16 zymbit 0.4.0 Zymbit cloud library
2015-04-16 clc-sdk 2.12 CenturyLink Cloud SDK and CLI

PYTHON PACKAGES: DISTUTILS


```
FooBar/
  setup.py
  foobar/
    __init__.py
    foo.py
    bar.py
    subfoo/
      __init__.py
      blah.py
```

```
1 from distutils.core import setup
2 setup(name='foobar',
3       version='1.0',
4       author='tarek',
5       author_email='foo@bar.org',
6       url='http://example.com',
7       packages=['foobar',
8                 'foobar.subfoo'],
9       )
```

PYTHON PACKAGES: DISTUTILS


```
# Build source distribution
$ python setup.py sdist
$ ls -l dist/
total 4
-rw-r--r-- 1 user user  491 Apr 17 14:22 hello-1.0.tar.gz
```

PYTHON PACKAGES: DISTUTILS


```
# Manual installation
$ tar zxvf hello-1.0.tar.gz
$ cd hello-1.0
$ sudo python setup.py install
```

PYTHON PACKAGES: DISTUTILS


```
# Register at PyPi and upload
$ python setup.py register
$ python setup.py sdist upload
```

PYTHON PACKAGES: DISTUTILS


```
# Install from PyPi
$ sudo pip install hello
```

PYTHON PACKAGES: DISTUTILS PROBLEMS

- But...

PYTHON PACKAGES: DISTUTILS PROBLEMS

- But...
- distutils is not really designed for compiled packages: Ex: `numpy.distutils` fork;

PYTHON PACKAGES: DISTUTILS PROBLEMS

- But...
- distutils is not really designed for compiled packages: Ex: `numpy.distutils` fork;
- no uninstalling;

PYTHON PACKAGES: DISTUTILS PROBLEMS

- But...
- distutils is not really designed for compiled packages: Ex: `numpy.distutils` fork;
- no uninstalling;
- You cannot use built in dependencies!

PYTHON PACKAGES: DISTUTILS PROBLEMS

- But...
- distutils is not really designed for compiled packages: Ex: `numpy.distutils` fork;
- no uninstalling;
- You cannot use built in dependencies!
- Only possible with `requirements.txt` file:

PYTHON PACKAGES: DISTUTILS PROBLEMS

- But...
- distutils is not really designed for compiled packages: Ex: `numpy.distutils` fork;
- no uninstalling;
- You cannot use built in dependencies!
- Only possible with `requirements.txt` file:

```
$ cat requirements.txt
FooProject >= 1.2
SomethingWhichVersionIDontCareAbout
BarProject == 1.0

# Install requirements
$ sudo pip install -r requirements.txt
```

PYTHON PACKAGES: HISTORY

- 2004: setuptools was introduced by Phillip Eby, which included the Egg format, and the ability to declare and automatically install dependencies;

PYTHON PACKAGES: HISTORY

- 2004: setuptools was introduced by Phillip Eby, which included the Egg format, and the ability to declare and automatically install dependencies;

```
smr2763Hello/
  setup.py
  ez_setup.py
  smr2763hello/
    __init__.py
    world.py
```

```
1  from ez_setup import use_setuptools
2  use_setuptools()
3  from setuptools import setup
4  from setuptools import find_packages
5  setup(name='smr2763hello',
6        version='0.2',
7        author='me',
8        author_email='foo@bar.org'
9        url='http://example.com',
10       packages = find_packages(),
11       )
```

PYTHON PACKAGES: HISTORY

- 2007: virtualenv was introduced by Ian Bicking, which allowed users to create isolated Python environments based on a central system installation of Python;

PYTHON PACKAGES: HISTORY

- 2007: virtualenv was introduced by Ian Bicking, which allowed users to create isolated Python environments based on a central system installation of Python;

PYTHON PACKAGES: HISTORY

- 2011: The PyPA is created to take over the maintenance of pip and virtualenv from Ian Bicking, led by Carl Meyer, Brian Rosner and Jannis Leidel. Other proposed names were ”ianb-ng”, ”cabal”, ”pack” and ”Ministry of Installation”.

PYTHON PACKAGES: HISTORY

- 2011: The PyPA is created to take over the maintenance of pip and virtualenv from Ian Bicking, led by Carl Meyer, Brian Rosner and Jannis Leidel. Other proposed names were ”ianb-ng”, ”cabal”, ”pack” and ”Ministry of Installation”.
- 2012: The effort to include ”Distutils2/Packaging” in Python 3.3 was abandoned due lack of involvement;

PYTHON PACKAGES: HISTORY

- 2011: The PyPA is created to take over the maintenance of pip and virtualenv from Ian Bicking, led by Carl Meyer, Brian Rosner and Jannis Leidel. Other proposed names were ”ianb-ng”, ”cabal”, ”pack” and ”Ministry of Installation”.
- 2012: The effort to include ”Distutils2/Packaging” in Python 3.3 was abandoned due lack of involvement;
- 2013: PEP425 and PEP427 were accepted. Together, they specify a built-package format for Python called Wheel;

DEMO

SUMMARIZING

TOOL RECOMMENDATIONS:

- Always try to install from your distro repository;

[1] from Python Packaging User Guide

TOOL RECOMMENDATIONS:

- Always try to install from your distro repository;
- But if you need to use PyPI [1]

[1] from Python Packaging User Guide

TOOL RECOMMENDATIONS:

- Always try to install from your distro repository;
- But if you need to use PyPI [1]
  - Use pip to install Python packages from PyPI;

[1] from Python Packaging User Guide

TOOL RECOMMENDATIONS:

- Always try to install from your distro repository;
- But if you need to use PyPI [1]
  - Use pip to install Python packages from PyPI;
  - Use virtualenv, or pyvenv to isolate application specific dependencies from a shared Python installation;

[1] from Python Packaging User Guide

TOOL RECOMMENDATIONS:

- Always try to install from your distro repository;
- But if you need to use PyPI [1]
  - Use pip to install Python packages from PyPI;
  - Use virtualenv, or pyvenv to isolate application specific dependencies from a shared Python installation;
  - Use setuptools to define projects and create Source Distributions.

[1] from Python Packaging User Guide

QUESTIONS?