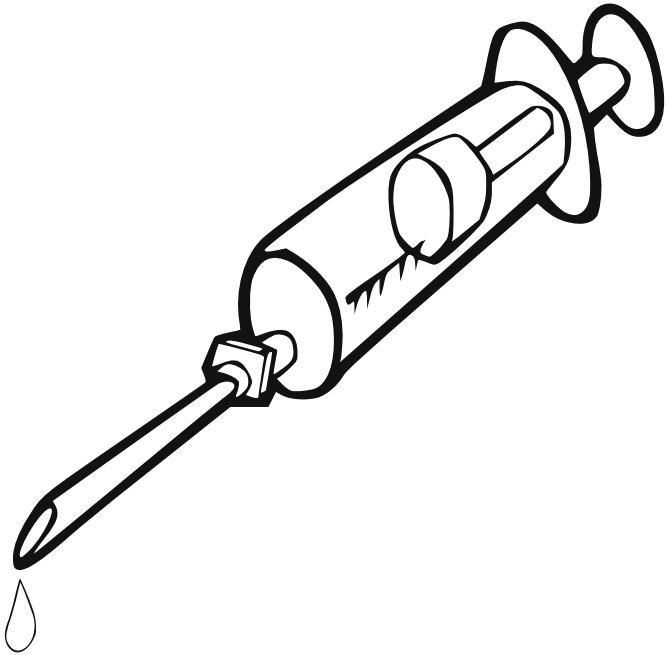
Packaging - où en est-on ?

PYCONFR STRASBOURG 26-29 OCTOBRE 2013

UNLISH

Christophe de Vienne
@cmdevienne

Piqûre de rappel



A quoi ça sert ?

faciliter la réutilisation

faciliter la diffusion

Vocabulaire

Dans l'écosystème python, "package" n'a pas tout à fait le sens habituel, ce qui est une source de confusion.

Paquet ("package")

Ensemble de modules python :

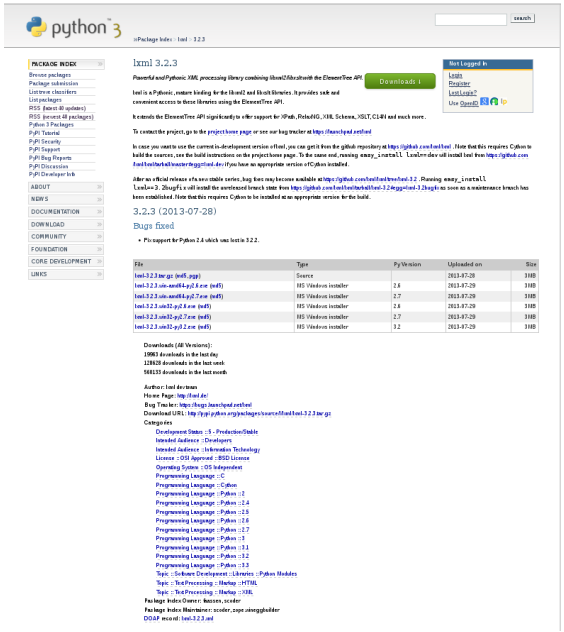
- Un module qui contient plusieurs modules
- Un ensemble de modules qu'on versionne et qu'on distribue sous un même nom - Twisted, lxml, pyramid...

Distribution source ou compilée ("source distribution" et "built distribution")

Archive permettant la diffusion et l'installation d'un ou plusieurs package(s).

PyPI - Python Package Index

https://pypi.python.org/pypi c'est + de 36000 packages



Sur PyPI, chaque version de chaque paquet a sa propre page. On y retrouve :

- la description longue
- Les distributions sources et compilées

File	Type	Py Version	Uploaded on	Size
lxml-3.2.3.tar.gz (md5, pgp)	Source		2013-07-28	3MB
lxml-3.2.3.win-amd64-py2.6.exe (md5)	MS Windows installer	2.6	2013-07-29	3MB
lxml-3.2.3.win-amd64-py2.7.exe (md5)	MS Windows installer	2.7	2013-07-29	3MB

- Les informations détaillées

Author: lxml dev team
Home Page: http://lxml.de/
Bug Tracker: https://bugs.launchpad.net/lxml
Download URL: http://pypi.python.org/packages/source/l/lxml/lxml-3.2.3.tar.gz
Categories

- Development Status :: 5 - Production/Stable
- Intended Audience :: Developers
- Intended Audience :: Information Technology
- License :: OSI Approved :: BSD License
- Operating System :: OS Independent
- Programming Language :: C

Packager

Comment packager votre SuperLib.

Préparer les sources

- Créer le setup.py

Distribution source

```
$ python setup.py sdist
```

=> `dist/SuperLib-0.1.tar.gz`

Distribuer

Sur PyPI

```
$ python setup.py sdist upload
```

=> http://pypi.python.org/pypi/SuperLib/0.1

Installer

A la main

```
$ tar xzf SuperLib-0.1.tar.gz
$ cd SuperLib-0.1
$ python setup.py install
```

Avec pip

Depuis PyPI :

```
$ pip install SuperLib
```

Depuis l'archive :

```
$ pip install SuperLib-0.1.tar.gz
```

Oui mais...

-> Le téléchargement depuis PyPI peut être long !

-> l'extension doit être compilée, il faut un compilateur !

Distribution compilée

A l'ancienne

```
$ python setup.py bdist
```

=> `dist/SuperLib-0.1.linux-x86_64.tar.gz`

Une archive qui contient ce qu'aurait installé "python setup.py install"...

- pas tellement portable
- pas d'outil pour installer ou gérer les dépendances

Setuptools

Pendant longtemps, la seule alternative a été setuptools, avec les distributions .egg et easy_install.

Préparer

On importe setuptools dans setup.py

Fabriquer

```
$ python setup.py bdist_egg
```

=> dist/SuperLib-0.1-py2.7-linux-x86_64.egg

Installer

```
$ easy_install SuperLib-0.1-py2.7-linux-x86_64.egg
```

- gère les dépendances

Oui mais...

- Format pas standard
- Maintenance de setuptools longtemps coincée
- ne tient pas compte des implémentations de python (CPython, pypy, jython...)
- les .egg sont à la fois des distributions et des paquets chargeables par l'interpréteur (comme le son les .jar en java)
- `pip` ne peut pas les installer
- easy_install a ses problèmes

Arrive distribute

Un fork de setuptools qui visait à :

- corriger les problèmes
- porter sur python 3
- documenter proprement
- et devenir le standard

Un travail nécessaire et qui débloque bien des situations mais :

- distribute fait quelques acrobaties pour se faire passer pour setuptools.
- setuptools est encore là : la confusion est totale pour bien des développeurs

Au bout du compte...


Domen Kožar
@iElectric
Abonné
There are some very dirty jobs. And one of them is fixing #python packaging
Voir la traduction
Répondre
Retweeter
Ajouté aux favoris
Plus
1
FAVORI
7:49 PM - 5 Sept, 13

Ce qui a changé

setuptools + distribute = setuptools 1.0

En mars 2013, la maintenance de setuptools est rendue plus communautaire, et conjointement avec les développeurs de distribute les deux outils sont fusionnés dans setuptools.

PIP par défaut

L'outil de gestion / installation de paquets "pip" a largement fait ses preuves.

- Il est désormais l'installeur par défaut dans l'écosytème python
- Formalisé dans la PEP 453 - "Explicit bootstrapping of pip in Python installations" - accepté le 22 octobre
- Officiellement recommandé dans la documentation de Python 2.7 et Python 3.3 et ultérieurs.
- Installé par défaut avec Python 3.4

PyPI - Exploration des liens

Le problème

Pip explorait systématiquement tous les liens trouvés sur ces pages.

Beaucoups de liens ou liens vers des sites lents / peu disponibles => de grosses lenteurs pour un simple téléchargement.

La solution

La PEP 428 "Transitioning to release-file hosting on PyPI".

=> "Hosting Mode" définit pour chaque paquet comment chercher les téléchargements :

- simple - fichiers présents sur pypi et lien de téléchargement.
- hybride - simple + liens de la description longue.
- complet - comportement historique.

=> acceptée le 19 mai 2013

=> Dans la foulée, tous les mainteneurs ont été sollicités pour paramétrer leur paquet.

Conséquences

Temps de recherche des distributions beaucoup plus court et fiable.

PyPI - CDN

Pour faire face à la demande, PyPI utilisait des mirroirs. La sélection du mirroir se faisait via leur nom. [a..z].pypi.python.org.

Le Problème

-> L'utilisation des mirroirs n'est pas totalement transparente

-> Une indisponibilité de PyPI a des conséquences pour BEAUCOUP de monde

La solution

-> Un CDN hébergé par Fastly

-> Activé le 26 mai 2013

Conséquences

-> Téléchargements beaucoups plus rapides

-> Plus grande disponibilité générale

-> Délai de mise à disposition d'un fichier plus long

-> Comptage du nombre de téléchargements encore moins fiable qu'avant

-> Les mirroirs [a..z].pypi.python.org seront définitivement désactivé le 14 février prochain

Wheel

Le besoin couvert par les .eggs ne pouvait être ignoré.

Un effort de standardisation s'est conclu par la PEP 427 "The Wheel Binary Package Format 1.0".

- Une convention de nommage plus riche qui tient compte de l'implémentation de python
- Un rôle bien défini : Un fichier .whl est destiné à être installé, pas à être importé.
- Implémente les nouveaux standards du packaging python :
  - PEP 376 "Database of Installed Python Distributions" qui définit le dossier '.dist-info' (qui joue le même rôle que .egg-info, mais standardisé).
  - PEP 426 "Metadata for Python Software Packages 2.0" qui définit comment exprimer les dépendances.
- Un format bien documenté.

Et concrètement ?

Fabrication

```
$ python setup.py bdist_wheel
```

=> dist/SuperLib-0.1-cp27-none-linux_x86_64.whl

```
_superlib.so

superlib/utils.py
superlib/__init__.py

SuperLib-0.1.dist-info/DESCRIPTION.rst
SuperLib-0.1.dist-info/pydist.json
SuperLib-0.1.dist-info/top_level.txt
SuperLib-0.1.dist-info/WHEEL
SuperLib-0.1.dist-info/METADATA
SuperLib-0.1.dist-info/RECORD
```

Installation

```
$ pip install SuperLib-0.1-cp27-none-linux_x86_64.whl
```

-> rapide

-> pas besoin de compilateur

-> pas besoin des versions de dev des dépendances

Exemple avec lxml

Sans wheel

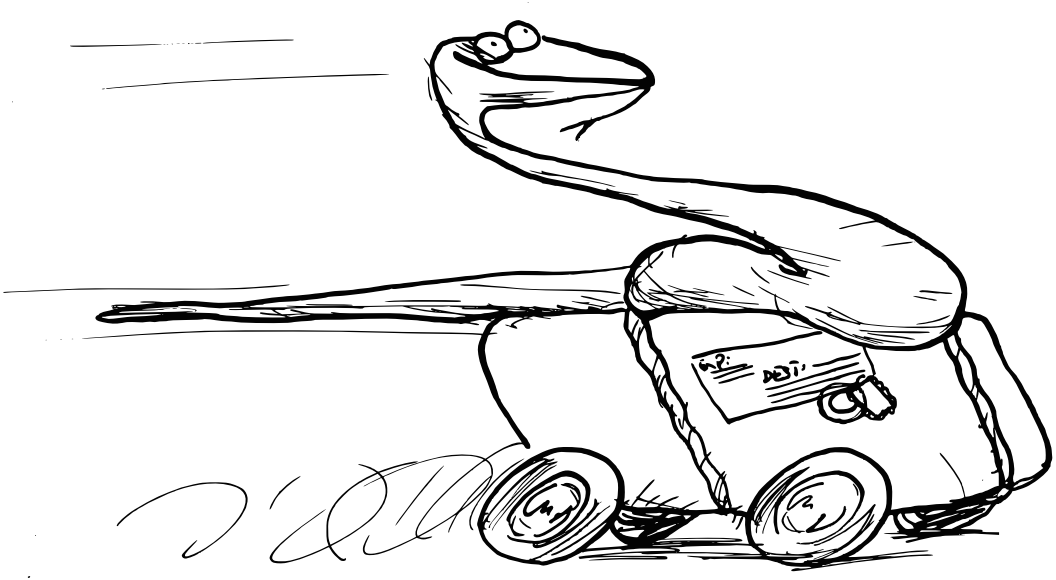
```
$ time pip install lxml
[...]
real    0m34.273s
```

Avec wheel

```
$ time pip install lxml-3.2.3-cp27-none-linux_x86_64.whl
Unpacking ./lxml-3.2.3-cp27-none-linux_x86_64.whl
Installing collected packages: lxml
Successfully installed lxml
Cleaning up...

real    0m0.281s
```

Une solution qui roule



En savoir plus

Wheel

https://pypi.python.org/pypi/wheel

Les PEPs

- PEP 376 "Database of Installed Python Distributions"
- PEP 426 "Metadata for Python Software Packages 2.0"
- PEP 427 "The Wheel Binary Package Format 1.0".
- PEP 428 "Transitioning to release-file hosting on PyPI".
- PEP 453 "Explicit bootstrapping of pip in Python installations"

Illustrations

Nardo

Questions ?

PYCONFR STRASBOURG 26-29 OCTOBRE 2013

UNLISH