Mais quelle galère que la distribution de programmes Python packagés !

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 10 décembre 2011

https://www.bortzmeyer.org/setup-distutils-egg-python.html

Je sais, je ferais mieux d'être constructif, au lieu de râler, et de dire du mal des autres programmeurs. Mais c'est trop énervant, il faut que je me défoule : lorsqu'on écrit un programme en Python, et qu'il dépend de plusieurs paquetages (dont certains peuvent être installés et d'autres pas), c'est très galère. Il n'existe aucun mécanisme standard pour indiquer dans la distribution quels paquetages sont nécessaires.

Démonstration avec la bibliothèque seenthis-python <https://github.com/bortzmeyer/seenthis-python>, qui permet d'accéder au service de court-bloguage et de partage de liens SeenThis <https://www.bortzmeyer.org/seenthis.html>. seenthis-python dépend de deux paquetages, SimpleTAL <http://www.owlfish.com/software/simpleTAL/>, un système de gabarits, pour générer du XML propre, et FeedParser <http://code.google.com/p/feedparser/>, pour analyser les messages en Atom que renvoie SeenThis. Dans la distribution de seenthis-python, comment indiquer (à part par du texte libre dans le README) qu'on dépend de ces paquetages ?

Commençons par le grand classique, car c'est le seul distribué avec Python, dans la bibliothèque standard, et sur lequel tout le monde peut compter : distutils <http://docs.python.org/distutils/>. Le principe est qu'on écrit un fichier `setup.py` qui va indiquer un certain nombre de choses sur le programme :

```
from distutils.core import setup

setup(name='SeenThis',
      version='0.0',
      description='Use the SeenThis API',
      license='BSD',
      author='Stephane Bortzmeyer',
      author_email='stephane+seenthis@bortzmeyer.org',
      url='https://github.com/bortzmeyer/seenthis-python',
      )
```

Ce fichier peut ensuite être exécuté avec des options qui vont dire par exemple d'installer le paquetage (`python setup.py install`) ou de l'enregistrer dans le "*Python Package Index (PyPI)*" <http://pypi.python.org/> (`python setup.py register`).

distutils a des tas de limites. Par exemple, il ne permet pas de faire des "*eggs*", des paquetages binaires pour Python. Mais, surtout, il ne permet pas d'exprimer des dépendances : aucun moyen dans le script ci-dessus de dire qu'on a besoin de SimpleTAL et de FeedParser. Le script `setup.py` s'exécutera et installera la bibliothèque, même si les pré-requis manquent. En lisant la documentation <http://docs.python.org/distutils/setupscript.html>, on a l'impression qu'on peut ajouter :

```
requires=['SimpleTAL', 'FeedParser']
```

mais c'est purement décoratif. La documentation ne le dit pas mais cette variable est complètement ignorée par distutils.

Au passage, le langage d'expression des dépendances (qui ne sert à rien, on l'a vu) permet également d'indiquer un numéro de version, par exemple le fait que, pour FeedParser, en raison de la bogue #91 <http://code.google.com/p/feedparser/issues/detail?id=91> (« `sgmllib.SGMLParseError: unexpected '\xe2' char in declaration` »), il faut au moins la version 5.

Lorsque le débutant en Python se plaint de cet état de chose, les vieux experts blanchis sous le harnais lui rétorquent que tout le monde le sait, et que personne n'utilise le système standard (alors, pense le débutant naïf, pourquoi est-ce que ce truc reste dans la bibliothèque standard ?). Et c'est là que les choses se compliquent, car il existe plusieurs remplaçants possibles.

D'abord, le premier, setuptools <http://pypi.python.org/pypi/setuptools> (le lien pointe vers PyPi puisque, à partir de là, on n'est plus dans la bibliothèque standard, ce qui oblige l'utilisateur qui va installer la bibliothèque à installer un autre programme d'abord, alors que le but était de lui simplifier la vie). Il existe une bonne documentation pour débutant <http://packages.python.org/an_example_pypi_project/setuptools.html>. En gros, on remplace la première ligne de code du `setup.py` par :

```
from setuptools import setup
```

et on a alors des fonctions et variables supplémentaires comme `install_requires` qui permet d'indiquer des dépendances. Tout content, le programmeur Python naïf édite le README pour indiquer à ses futurs utilisateurs qu'il va falloir installer setuptools d'abord avant de faire le `python setup.py build`. Mais il va vite être déçu : même si SimpleTAL est installé, le `setup.py` va tenter de le télécharger (et échouer, car il n'est pas dans PyPi). En effet, setuptools ne teste pas si un module Python donné est disponible (`import simpletal`), il regarde s'il y a un "*egg*" du même nom. Ce qui n'est pas le cas. Les dépendances de setuptools sont donc vers les "*eggs*", pas vers les modules, ce qui ôte une bonne partie de leur intérêt.

D'autant plus que, on l'a vu, des paquetages fréquemment répandus, comme SimpleTAL, ne sont pas forcément disponibles dans PyPi... (SimpleTAL est arrivé plus tard.)

Bon, troisième tentative : distribute <http://packages.python.org/distribute/>, qui prétend remédier aux problèmes de setuptools. Il reprend en partie son nom (ce qui contribue beaucoup à la confusion des programmeurs, par exemple le paquetage Debian de distribute se nomme setuptools) :

---

https://www.bortzmeyer.org/setup-distutils-egg-python.html

```
from distribute_setup import use_setuptools
use_setuptools()
from setuptools import setup

[puis setup.py comme avant]
```

Comme setuptools, il permet de déclarer des dépendances <http://packages.python.org/distribute/setuptools.html#declaring-dependencies>, mais il les gère tout aussi mal :

```
# python setup.py install
...
Searching for simpletal
Reading http://pypi.python.org/simple/simpletal/
Reading http://www.owlfish.com/software/simpleTAL/index.html
No local packages or download links found for simpletal
error: Could not find suitable distribution for Requirement.parse('simpletal')

%  python
Python 2.6.6 (r266:84292, Dec 27 2010, 00:02:40)
[GCC 4.4.5] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import simpletal
>>>
```

(La dernière commande montre que SimpleTAL est pourtant bien installé.)

Bref, on a trois systèmes concurrents et dont aucun ne permet d'assurer une tâche aussi simple que « vérifie que simpletal est bien disponible ». Pire, rien dans la documentation n'indique leurs limites. Ainsi tous les programmeurs Python semblent savoir que distutils est très limité mais cela n'apparaît pas dans la documentation de la bibliothèque standard. Et ce n'est pas fini. Comme l'écrit Victor Stinner «Le packaging Python est une série à la Dallas avec ses drames et ses rebondissements. Le dernier événement marquant est que la réécriture de distutils, "distutils2", a été intégrée dans Python 3.3 sous le nom "packaging". Sa version pour Python 2.5 - 3.2 sera maintenue en dehors de Python sous le nom "distutils2". » Il y a aussi des compétiteurs comme Bento <http://cournape.github.com/Bento/> ou plus exotiques comme zc.buildout <http://pypi.python.org/pypi/zc.buildout>.

Le problème du "*packaging*" Python est discuté avec bien plus de détails techniques (et moins de râleries pas constructives) dans l'excellent article "*Python packaging*" de Tarek Ziadé dans le livre "*Architecture of Open Source Software*" <http://www.aosabook.org/en/index.html>, (co-dirigé par Greg Wilson et Amy Brown).

Un petit truc : si on veut regarder la variété des fichiers "*setup.py*" possibles, on peut demander à Google <http://www.google.com/codesearch?hl=en&start=10&sa=N&q=requires+lang:python+file:setup.py> (ici, en se limitant à ceux qui contiennent `requires`).

Pour terminer, une bonne série de lectures, due à Nicolas Chauvat :

- "*The Configuration Management Problem*" <http://www.logilab.org/blogentry/9860>,
- "*Looking for a Windows Package Manager*" <http://www.logilab.org/blogentry/9861>,
- "*Virtualenv - Play safely with a Python*" <http://www.logilab.org/blogentry/22498>,
- "*A comprehensive guide through Python packaging (a.k.a. setup scripts)*" <http://foobar.lu/wp/2012/05/13/a-comprehensive-step-through-python-packaging-a-k-a-setup-scripts/>, un bon document d'introduction pour ceux qui veulent souffrir le moins possible,
- "*PyPi is not a software distribution*" <http://www.logilab.org/blogentry/60163>,
- "*Sprint Distutils2*" <http://www.logilab.org/blogentry/57611> et la suite <http://www.logilab.org/blogentry/60262>,
- Un PEP qui spécifie un format de fichier qui décrit les dépendances d'un module python PEP345 <http://www.python.org/dev/peps/pep-0345/>, et qui, si un outil s'en servait, simplifierait formidablement la vie de ceux qui font des paquets pour les distributions.