Tema 8 Distribución del código

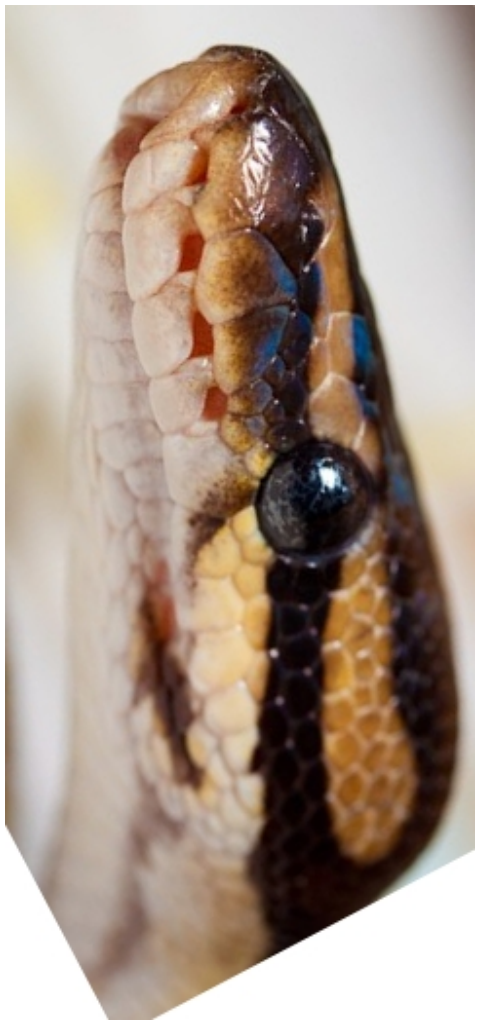
Curso de Python Avanzado

Juan Pedro Bolívar Puente

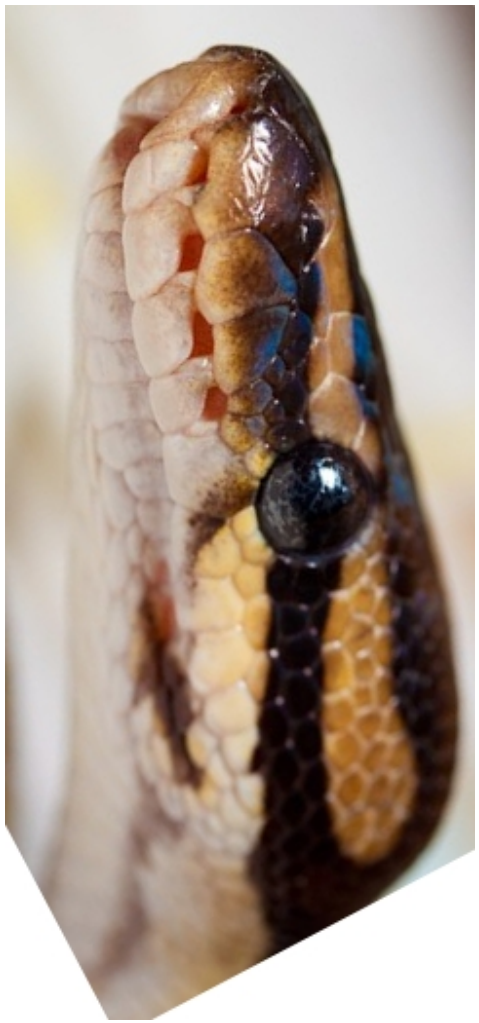
Instituto Andaluz de Astrofísica

Mayo de 2011

Índice



Índice



Introducción

`distutils` = utilidades para la distribución del código

Permite

- Gestionar los metadados del paquete
- Gestionar la compilación de módulos sencillos
- Gestionar las instalación del software
- Crear paquetes para su distribución

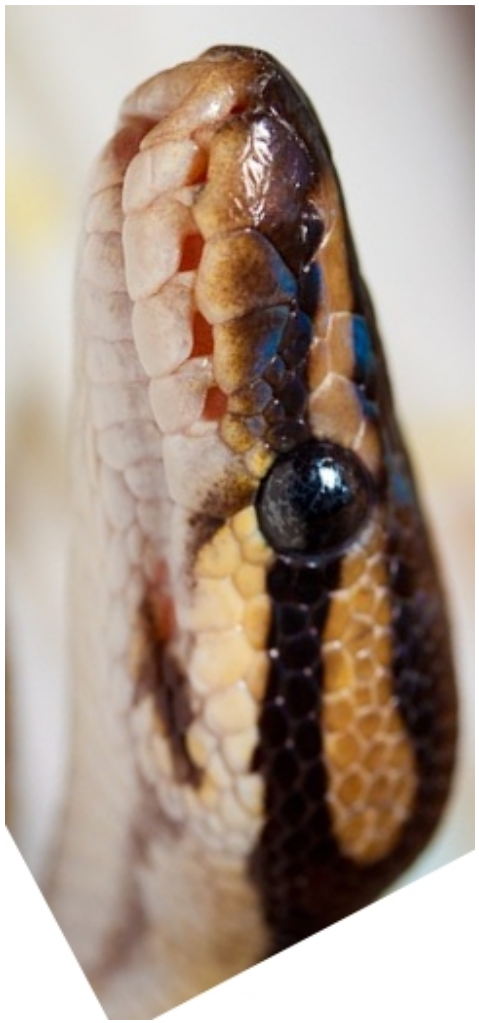
Ejemplo de programa

```
$ python setup.py --help
$ python setup.py --help-commands
$ python setup.py --author
$ python setup.py --provides
$ python setup.py install
```

Después de instalar podemos usarlo en cualquier sitio

```
from crazymod import mod
mod.crazyfunc ()
```

Índice



La base de distutils

La base de distutils es la función `distutils.core.setup`

- Ejecuta el script que hemos visto
- Parámetros por clave modifican su comportamiento

Ejemplo

`packages` indica una lista de paquetes ...

```
from distutils.core import setup

setup (
    name         = 'CrazySoft',
    version      = '1.3',
    description  = 'Crazy package that is ..',
    author       = 'Crazy Author',
    author_email = 'raskolnikov@gnu.org',
    url          = 'http://www.iaa.es',
    packages     = ['crazymod'],
    )
```

Buscando automáticamente ...

`find_packages` los busca automáticamente ...

```
from distutils.core import setup
from distutils.core import find_packages

setup (
    name         = 'CrazySoft',
    version      = '1.3',
    description  = 'Crazy package that is ..',
    author       = 'Crazy Author',
    author_email = 'raskolnikov@gnu.org',
    url          = 'http://www.iaa.es',
    packages     = find_packages (),
    )
```

Encontrando paquetes

Aunque es mejor ser explícito ...

Cuando nombre del paquete $\neq$ nombre del directorio ...

```
package_dir = {'': 'lib'}
```

```
package_dir = {'foo' : 'lib'}
```

Módulos

En programas pequeños hay módulos sueltos ...

```
py_modules = ['mod1', 'pkg.mod2']
```

¡Los paquetes tienen que tener su `__init__.py` de todas formas!

Metadatos

author	Lista con los autores del proyecto
author_email	Lista con los correos de los autores, si quieren recibir correos
contact	Persona de contacto
contact_email	Debes proveerlo
description	Descripcion corta, idealmente de < 80 caracteres
long_description	Descripcion completa
fullname	Nombre completo
keywords	Etiquetas

Metadatos

license	¡Poner siempre!
maintainer	Poner si es diferente del autor
maintainer_email	Y su correo
name	Nombre como ¡idenficador válido!
platforms	Lista de plataformas donde crees que funciona...
url	Página oficial
version	Versión del proyecto
classifiers	Lista de Trove classifiers

Clasificadores

http://pypi.python.org/pypi?%3Aaction=list_classifiers

```
'Development Status :: 4 - Beta',
'Environment :: Console',
'Environment :: Web Environment',
'Intended Audience :: End Users/Desktop',
'Intended Audience :: Developers',
'Intended Audience :: System Administrators',
'License :: OSI Approved :: Python Software Foundation License',
'Operating System :: MacOS :: MacOS X',
'Operating System :: Microsoft :: Windows',
'Operating System :: POSIX',
'Programming Language :: Python',
'Topic :: Communications :: Email',
'Topic :: Office/Business',
'Topic :: Software Development :: Bug Tracking',
```

Clasificadores

http://pypi.python.org/pypi?%3Aaction=list_classifiers

```
'Development Status :: 4 - Beta',
'Environment :: Console',
'Environment :: Web Environment',
'Intended Audience :: End Users/Desktop',
'Intended Audience :: Developers',
'Intended Audience :: System Administrators',
'License :: OSI Approved :: Python Software Foundation License',
'Operating System :: MacOS :: MacOS X',
'Operating System :: Microsoft :: Windows',
'Operating System :: POSIX',
'Programming Language :: Python',
'Topic :: Communications :: Email',
'Topic :: Office/Business',
'Topic :: Software Development :: Bug Tracking',
```

Instalando scripts

```
scripts=['scripts/xmlproc_parse',
         'bin/xmlproc_val']
```

- Se instalan en `$PREFIX/bin`
- Si la primera linea es `#!` y contiene la palabra `python` entonces se cambia por el interprete de Python local durante la instalación

  `--excutable` permite al usuario cambiarlo a mano

Ficheros de datos

Datos asociados a un paquete

```
package_data={'mypkg': ['data/*.dat']},
```

- Los ficheros contienen globs
- Están relativos al directorio del paquete
- Las jerarquías de directorios las crea el instalador

Ficheros de datos

Fichero que de los que el código no depende tanto

```
data_files=[('bitmaps', ['bm/b1.gif', 'bm/b2.gi
            ('config', ['cfg/data.cfg']),
            ('/etc/init.d', ['init-script'])]
```

- Las rutas destino relativas lo son a `sys.prefix`
- Están relativos al directorio actual
- Del fuente sólo se mantiene el nombre del fichero

Dependencias

Podemos especificar las dependencias con `depends`

Se pueden poner la versión con comparadores

```
setup (
   depends = ['pyfits',
              'pygtk >= 2.12']
)
```

Dependencias

`provides` y `obsoletes` permiten definir qué dependencias proveemos

```
setup (
   provides = ['mymod', 'mypkg (3.2)'],
   obsoletes = ['oldmod < 2.0']
)
```

Extensiones

Automatizamos la compilación e instalación de extensiones

```
from distutils.core import setup, Extension
setup(...,
   ext_package='pkg',
   ext_modules=[
     Extension('foo', ['f1.c', 'f2.c']),
     Extension('subpkg.bar', ['bar.c'])],
)
```

También soporta `.i` para SWIG

Extensiones

A `Extension` le podemos pasar parámetros

include_dirs Directorios de los *includes*

define_macros Macros del preprocesador

undef_macros Que desdefinir

library_dirs Para buscar bibliotecas

libraries Con las que enlazar

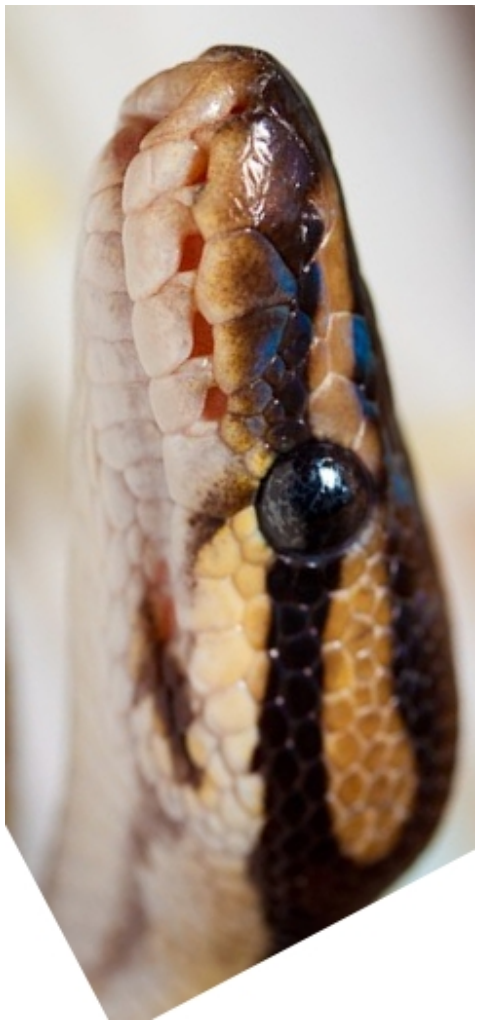
El asunto de Ctypes

¿Cuidado con ctypes?

Mi hack personal :)

```
def load_ctypes_extension (name):
    lib = None
    for x in sys.path:
        try:
            lib = cdll.LoadLibrary(path.join(x,name))
            break
        except OSError:
            pass
    if not lib:
        raise ImportError, "Could␣not␣find␣C␣lib␣..."
    return lib
```

Índice



Distribuyendo el software

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

Genera un tarball que incluye ...

- Los fuentes de los `py_modules`
- Los ficheros C de los `ext_modules`
- Los scripts de `scripts`
- Las pruebas de `test/test*.py`
- `README`, `README.txt`, `setup.py`, `seutp.cfg`

Distribuyendo el software

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

A partir de Python 2.7 además mete ...

- Los datos de `package_data`
- Los datos de `data_files`

¡Hacen falta más datos!

El fichero MANIFEST

`sdist` incluye todo lo que hay en `MANIFEST`

- `MANIFEST` contiene un fichero por linea
- Si no existe lo genera de `MANIFEST.in`

Ejemplo ...

```
include *.txt
recursive-include examples *.txt *.py
prune examples/sample?/build
```

¡El orden es importante!

- El conjunto estandar se añade antes
- `build`, `RCS`, `CVS`, `.svn`, `.hg`, `.git`, `.bzr` y `_darcs` se excluyen después

El lenguaje de `MANIFEST.in`

include pat1 pat2 ... Incluye ficheros de los patrones

exclude pat1 pat2 ... Excluye ficheros de los patrones

recursive-include dir pat1 pat2 ... Incluye los ficheros de dir que satisfacen los patrones

recursive-include dir pat1 pat2 ... Igual pero excluye

global-include pat1 pat2 ... Equivalente a recursive-include . pat1 pat2 ...

global-include pat1 pat2 ... Equivalente a recursive-exclude . pat1 pat2 ...

prune dir Excluye todo es directorio

graf dir Incluye todo el directorio

Configuración

Fichero de configuración sirve como punto intermedio entre el script de `setup` y los parámetros de consola

- Valores por defecto
- Opciones específicas de comandos

```
#comentario

[comando]
opciona = valor
opcionb = valor2
```

Configuración

El comando especial `global` se aplica siempre

Ejemplo...

```
[global]
verbose=0
```

Cuidado al meter estas cosas en un paquete, mejor hacerlo localmente...

Orden ficheros de configuración

Cada fichero puede sobreescribir los valores del anterior

1. Configuración global: *prefix*/lib/python*ver*/distutils/distutils.cfg
2. De usuario: $HOME/.pydistutils.cfg
3. Local: setup.cfg
4. Parámetros pasados directamente

Ejemplo

Aquí es mejor idea cambiar `[global]`

Ejemplo de `.pydistutils.cfg`

```
[global]
prefix=/home/raskolnikov/usr
verbose=0
```

Otras formas de distribución ...

Crear un binario "tonto"

```
$ python setup.py bdist $ python setup.py
bdist --help
```

Crear un binario RPM

```
$ python setup.py bdist --formats=rpm
$ python setup.py bdist_rpm
```

Crear un instalador de Windows

```
$ python setup.py bdist_wininst
```

Ejemplo ...

Ejemplo para para el fichero “spec” de RPM

```
[bdist_rpm]
release = 1
packager = Jogn Doe <john@doe.net>
doc_files = CHANGES.txt
            README.txt
            USAGE.txt
            doc/
            examples/
```

Paquetes para Debian/Ubuntu/Triskel...

Necesitamos python-stdeb

```
$ python setup.py \
    --command-packages=stdeb.command \
    bdist_deb
```

Sugerencia .pydistutils.cfg

```
[global]
command-packages: stdeb.command
```

Paquetes para Debian/Ubuntu/Triskel...

Necesitamos python-stdeb

```
$ python setup.py bdist_deb
```

- Esto sólo garantiza que el paquete funcionará en tu versión de Debian
- Especialmente malo con extensiones

Paquetes para Debian/Ubuntu/Triskel...

Mejor generar un fuente Debian (dsc)

```
$ py2dsc paquetito-0.1.0.tar.gz
$ python setup.py sdist_dsc
```

Y en la carpetita del fuente Debian:

```
dpkg-buildpackage -rfakeroot -uc -us
```

En Ubuntu:
https://help.launchpad.net/Packaging/PPA

Configurando el paquete Debian

En el fichero `setup.cfg`

```
[sdist_dsc]
maintainer: Chuck Norris
```

Hay que propagar las opciones a los comandos ...

```
$ python setup sdist_dsc \
    --debian-version IIA0 \
    bdist_deb
```

Configurando el paquete Debian

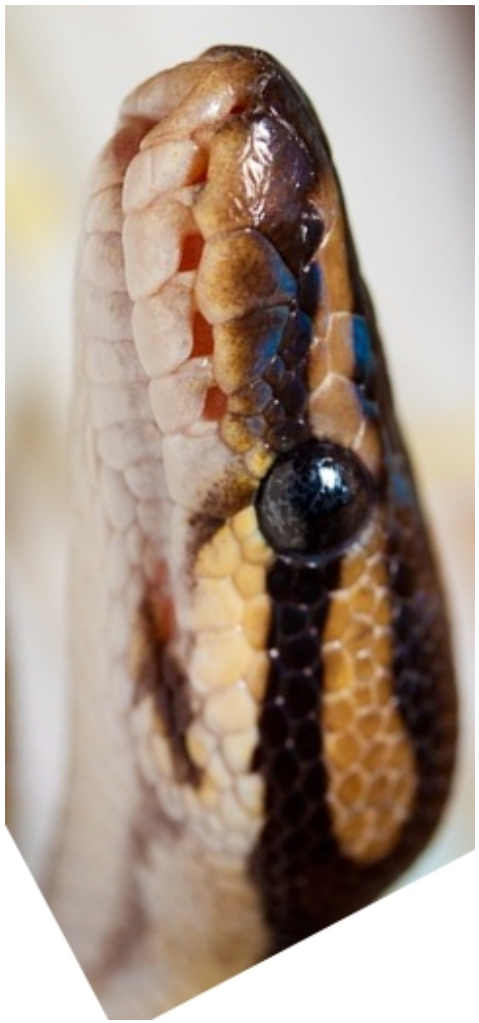
En el fichero `stdeb.cfg`

```
[DEFAULT]
Maintainer: Jimmy Page
Depends: python-numpy
XS-Python-Version: >= 2.6
```

Es mejor porque...

- Se lee siempre
- Opciones estilo `debian/control`

Índice



Python Package Index

Repositorio de software en Python
aka Cheese Shop

- Software distribuido con distutils
- Utilidad asociada: Easy Install
- http://pypi.python.org/pypi
- http://www.youtube.com/watch?v=B3KBuQHHKx0

En Debian...

Un regalito de `python-stdeb`

```
$ pypi-install bpython
```

¡Descarga, compila, genera paquete deb e instala!

Registrando un paquete

```
python setup.py register
```

```
running register
We need to know who you are, so please choose either:
    1. use your existing login,
    2. register as a new user,
    3. have the server generate a new password for you
       (and email it to you), or
    4. quit
Your selection [default 1]:
```

Via web: `http://pypi.python.org/pypi?%3Aaction=register_form`

Configurando los servidores ...

El fichero `.pypirc` podemos configurar otros repositorios

```
[distutils]
index-servers =
    pypi

[pypi]
repository: <repository-url>
username: <username>
password: <password>
```

Subiendo ficheros al repositorio

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

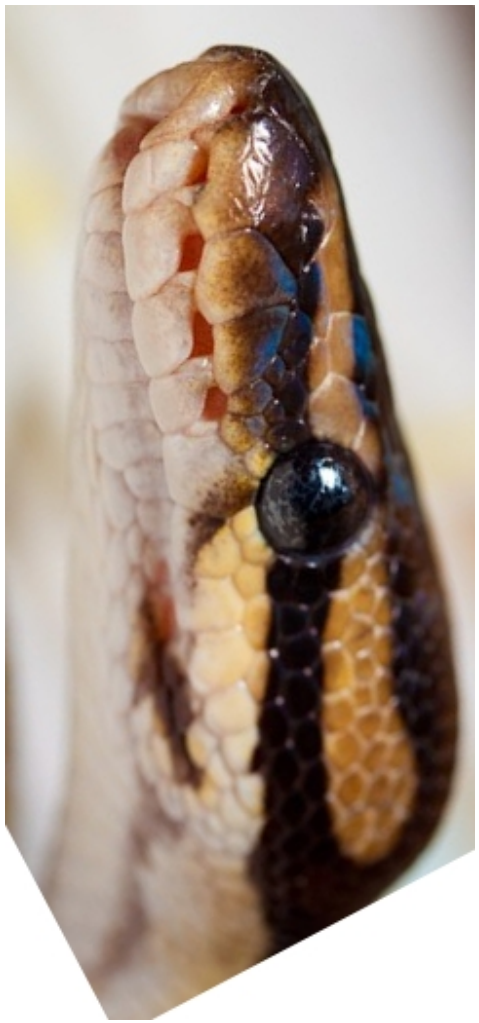
Opciones...

- `-r repo` Repositorio alternativo
- `--sign` Firmar con GPG

La portada en PyPi es
la `long_description` en reStructuredText

```
$ python setup.py --long-description | \
      rst2html.py > output.html
```

Índice



Conclusiones

- Para distribuir usad `distutils`
- Para instalar usad el sistema de paquetes de la distro
- Existen alternativas...
  - SCons (¡En Python!)
  - Waf (¡En Python!)
  - Autotools
  - Cmake

Recursos adicionales

¿Preguntas?

Muchas gracias por su atención.