

Packaging Python Projects

Christian Külker

2020-05-16

Contents

1	Example Test Package	1
1.1	Create Account	2
1.2	Create A Project	2
1.3	Install Dependencies	3
1.4	Build The Project	3
1.5	Upload The Project	5
1.6	Install The Project	6
2	Real Project	7
3	Excursus On Python3 vs PyPy	8
4	History	9
5	Disclaimer of Warranty	9
6	Limitation of Liability	9

This document describes briefly the process to package Python projects. It follows the Python Software Foundation [packaging tutorial](#) that uses [setuptools](#). Setuptools is a collection of enhancements to the Python distribution utilities that allow you to more easily build and distribute Python distributions, especially ones that have dependencies on other packages. For details on how to use the [setuptools](#) see the [documentation of setuptools](#).

1 Example Test Package

1.1 Create Account

Because [TestPyPI](#) has a separate database from the live PyPI, you'll need a separate user account for specifically for TestPyPI. [Create an account on test.pypi.org](#) to [register an account](#) with a valid mail address.

Go to the [token section](#) of your account and create a new API token; don't limit its scope to a particular project, since you are creating a new project.

Since we will use `twine` copy the API token to `$HOME/.pypirc`. However this will not hinder to be asked about it. So maybe it is useless, but mentioned in your token section of your test account.

```
1 [pypi]
2   username = __token__
3   password = pypi-.....
```

1.2 Create A Project

We create the project structure:

```
mkdir -p packaging_tutorial/example_pkg/tests
touch packaging_tutorial/example_pkg/__init__.py
touch packaging_tutorial/{LICENSE,README.md,setup.py}
cd packaging_tutorial
echo -e "# Example Package\n\nThis is a simple example package.">README.md
vim setup.py
```

The contents of `setup.py`, replace `YOUR-USERNAME-HERE` with your user name. And probably also update others, like URL to your repository and e-mail address. (That is a guess, the tutorial gives no information about this).

```
1 import setuptools
2
3 with open("README.md", "r") as fh:
4     long_description = fh.read()
5
6 setuptools.setup(
7     name="example-pkg-YOUR-USERNAME-HERE", # Replace with your own username
8     version="0.0.1",
9     author="Example Author",
10    author_email="author@example.com",
11    description="A small example package",
12    long_description=long_description,
13    long_description_content_type="text/markdown",
14    url="https://github.com/pypa/sampleproject",
15    packages=setuptools.find_packages(),
```

```
16     classifiers=[
17         "Programming Language :: Python :: 3",
18         "License :: OSI Approved :: MIT License",
19         "Operating System :: OS Independent",
20     ],
21     python_requires='>=3.6',
22 )
```

1.3 Install Dependencies

Even though it is recommended to install the latest version of `setuptools` and `wheel`, if you like to maintain the security management supported by your Linux distribution, you probably will prefer the package manager.

```
aptitude install python3-setuptools python3-wheel twine python3-pip
```

1.4 Build The Project

Build distribution

```
python3 setup.py sdist bdist_wheel
running sdist
running egg_info
creating example_pkg_ckuelker.egg-info
writing example_pkg_ckuelker.egg-info/PKG-INFO
writing dependency_links to
  example_pkg_ckuelker.egg-info/dependency_links.txt
writing top-level names to example_pkg_ckuelker.egg-info/top_level.txt
writing manifest file 'example_pkg_ckuelker.egg-info/SOURCES.txt'
reading manifest file 'example_pkg_ckuelker.egg-info/SOURCES.txt'
writing manifest file 'example_pkg_ckuelker.egg-info/SOURCES.txt'
running check
creating example-pkg-ckuelker-0.0.1
creating example-pkg-ckuelker-0.0.1/example_pkg
creating example-pkg-ckuelker-0.0.1/example_pkg_ckuelker.egg-info
copying files to example-pkg-ckuelker-0.0.1...
copying README.md -> example-pkg-ckuelker-0.0.1
copying setup.py -> example-pkg-ckuelker-0.0.1
copying example_pkg/__init__.py -> example-pkg-ckuelker-0.0.1/example_pkg
copying example_pkg_ckuelker.egg-info/PKG-INFO ->
  example-pkg-ckuelker-0.0.1/
example_pkg_ckuelker.egg-info
copying example_pkg_ckuelker.egg-info/SOURCES.txt -> \
```

```
example-pkg-ckuelker-0.0.1/example_pkg_ckuelker.egg-info
copying example_pkg_ckuelker.egg-info/dependency_links.txt -> \
example-pkg-ckuelker-0.0.1/example_pkg_ckuelker.egg-info
copying example_pkg_ckuelker.egg-info/top_level.txt -> \
example-pkg-ckuelker-0.0.1/example_pkg_ckuelker.egg-info
Writing example-pkg-ckuelker-0.0.1/setup.cfg
creating dist
Creating tar archive
removing 'example-pkg-ckuelker-0.0.1' (and everything under it)
running bdist_wheel
running build
running build_py
creating build
creating build/lib
creating build/lib/example_pkg
copying example_pkg/__init__.py -> build/lib/example_pkg
installing to build/bdist.linux-x86_64/wheel
running install
running install_lib
creating build/bdist.linux-x86_64
creating build/bdist.linux-x86_64/wheel
creating build/bdist.linux-x86_64/wheel/example_pkg
copying build/lib/example_pkg/__init__.py ->
  build/bdist.linux-x86_64/wheel/\
example_pkg
running install_egg_info
Copying example_pkg_ckuelker.egg-info to build/bdist.linux-x86_64/wheel/\
example_pkg_ckuelker-0.0.1.egg-info
running install_scripts
adding license file "LICENSE" (matched pattern "LICEN[CS]E*")
creating
  build/bdist.linux-x86_64/wheel/example_pkg_ckuelker-0.0.1.dist-info/\
WHEEL
creating 'dist/example_pkg_ckuelker-0.0.1-py3-none-any.whl' and adding \
'build/bdist.linux-x86_64/wheel' to it
adding 'example_pkg/__init__.py'
adding 'example_pkg_ckuelker-0.0.1.dist-info/LICENSE'
adding 'example_pkg_ckuelker-0.0.1.dist-info/METADATA'
adding 'example_pkg_ckuelker-0.0.1.dist-info/WHEEL'
adding 'example_pkg_ckuelker-0.0.1.dist-info/top_level.txt'
adding 'example_pkg_ckuelker-0.0.1.dist-info/RECORD'
removing build/bdist.linux-x86_64/wheel
```

```
$ tree dist
dist
├── example_pkg_ckuelker-0.0.1-py3-none-any.whl
└── example-pkg-ckuelker-0.0.1.tar.gz
```

Altogether the project tree looks like this.

```
$ tree
.
├── build
│   ├── bdist.linux-x86_64
│   └── lib
│       └── example_pkg
│           └── __init__.py
├── dist
│   ├── example_pkg_ckuelker-0.0.1-py3-none-any.whl
│   └── example-pkg-ckuelker-0.0.1.tar.gz
├── example_pkg
│   ├── __init__.py
│   └── tests
├── example_pkg_ckuelker.egg-info
│   ├── dependency_links.txt
│   ├── PKG-INFO
│   ├── SOURCES.txt
│   └── top_level.txt
├── LICENSE
├── README.md
└── setup.py
```

1.5 Upload The Project

Upload the distribution (using username `__token__` and the token for password):

```
python3 -m twine upload --repository testpypi dist/*
Enter your username: __token__
/usr/lib/python3/dist-packages/twine/utils.py:238: UserWarning: Failed to
  ↳ open
keyring: org.freedesktop.DBus.Error.NoReply: Did not receive a reply.
  ↳ Possible
causes include: the remote application did not send a reply, the message
  ↳ bus
security policy blocked the reply, the reply timeout expired, or the
  ↳ network
```

```
connection was broken..
  warnings.warn(str(exc))
Enter your password:
Uploading distributions to https://test.pypi.org/legacy/
Uploading example_pkg_ckuelker-0.0.1-py3-none-any.whl
100%|████████████████████████████████████████| 16.7k/16.7k [00:01<00:00,
  ↪ 10.2kB/s]
Uploading example-pkg-ckuelker-0.0.1.tar.gz
100%|████████████████████████████████████████| 4.29k/4.29k [00:00<00:00,
  ↪ 4.89kB/s]
```

The package should be visible at <https://test.pypi.org/project/example-pkg-YOUR-USERNAME-HERE>

1.6 Install The Project

It can be installed via `virtualenv` and `pip` (in this case python2):

```
virtualenv /tmp/python-packaging-tutorial-example-package
Running virtualenv with interpreter /usr/bin/python2
New python executable in
  ↪ /tmp/python-packaging-tutorial-example-package/bin/\
python2
Also creating executable in
  ↪ /tmp/python-packaging-tutorial-example-package/\
bin/python
Installing setuptools, pkg_resources, pip, wheel...done.
source /tmp/python-packaging-tutorial-example-package/bin/activate
```

For `zsh` you know that the environment is active when your `zsh` prompt is messed up as it prints (DIR) in front of your prompt.

Or you use `virtualenv` with python3

```
virtualenv -p /usr/bin/python3
  ↪ /tmp/python-packaging-tutorial-example-package
Already using interpreter /usr/bin/python3
Using base prefix '/usr'
New python executable in
  ↪ /tmp/python-packaging-tutorial-example-package/bin/\
python3
Also creating executable in
  ↪ /tmp/python-packaging-tutorial-example-package/\
bin/python
```

```
Installing setuptools, pkg_resources, pip, wheel...done.  
source /tmp/python-packaging-tutorial-example-package/bin/activate
```

Why `virtualenv` is complaining about 'Already ...' is unclear, as it certainly did use python2 before. Does this software have some attitude issues?

Installing the new test software:

```
python3 -m pip install --index-url https://test.pypi.org/simple/ --no-deps  
↩ \  
example-pkg-YOUR-USERNAME-HERE  
Looking in indexes: https://test.pypi.org/simple/  
Collecting example-pkg-ckuelker  
  Downloading https://test-files.pythonhosted.org/packages/bb/83/\  
db02a09e5963f60e3ef79e9c491d9aaf772d3e9a60d7b628d5434e9d6f26/\  
example_pkg_ckuelker-0.0.1-py3-none-any.whl (13 kB)  
Installing collected packages: example-pkg-ckuelker  
Successfully installed example-pkg-ckuelker-0.0.1
```

Testing the installation with the command line.

```
python3  
import example_pkg
```

Since the example package does nothing there is nothing more to say on this. However one might add a non trivial package to test.pypi.org to better understand this process.

2 Real Project

Real projects should be tested on [\[test.pypi.org\]](https://test.pypi.org), however at some point you might upload it to the real pypi.org. Here only some short differences to the example project section above.

- Chose a unique name for your project (you should not add your user name to the project name)
- [Create an account on pypi.org](#) (not on the test.pypi.org)
- Use `twine upload dist/*` without `--repository` option
- Test installation with `pip install` (No `--index-url`, no `--no-deps`)

3 Excursus On Python3 vs PyPy

```

aptitude search setuptools
p  pypy-setuptools          - PyPy Distutils Enhancements
p  pypy-setuptools-scm      - blessed package to manage your
                             versions by scm tags for PyPy
p  python-setuptools        - Python Distutils Enhancements
p  python-setuptools-doc    - Python Distutils Enhancements
                             (documentation)
p  python-setuptools-git    - plugin for setuptools that enables
  ↪ git                    integration
p  python-setuptools-scm    - blessed package to manage your
                             versions by scm tags for Python 2
v  python2.7-setuptools-git -
i A python3-setuptools      - Python3 Distutils Enhancements
p  python3-setuptools-git   - plugin for setuptools that enables
  ↪ git                    integration
p  python3-setuptools-scm   - blessed package to manage your
                             versions by scm tags for Python 3
p  python3-setuptools-scm-git-archive - Plugin for setuptools_scm to add
                             support for git archives

```

From a distance Debian offers 2 packages: `python3-setuptools` and `pypy-setuptools`. However it seems they are the same.

```

aptitude show pypy-setuptools
Package: pypy-setuptools
Version: 40.8.0-1
State: not installed
Multi-Arch: foreign
Priority: optional
Section: python
Maintainer: Matthias Klose <doko@debian.org>
Architecture: all
Uncompressed Size: 1,353 k
Depends: pypy-pkg-resources (= 40.8.0-1), pypy
Suggests: python-setuptools-doc
Description: PyPy Distutils Enhancements
 Extensions to the python-distutils for large or complex distributions.
Homepage: https://pypi.python.org/pypi/setuptools

```

```
aptitude show python3-setuptools
Package: python3-setuptools
Version: 40.8.0-1
State: installed
Automatically installed: yes
Multi-Arch: foreign
Priority: optional
Section: python
Maintainer: Matthias Klose <doko@debian.org>
Architecture: all
Uncompressed Size: 1,353 k
Depends: python3-pkg-resources (= 40.8.0-1), python3-distutils,
        python3:any
Suggests: python-setuptools-doc
Description: Python3 Distutils Enhancements
Extensions to the python-distutils for large or complex distributions.
Homepage: https://pypi.python.org/pypi/setuptools
```

4 History

Version	Date	Notes
0.1.1	2022-05-26	+History, Change shell to bash
0.1.0	2020-05-16	Initial release

5 Disclaimer of Warranty

THERE IS NO WARRANTY FOR THIS INFORMATION, DOCUMENTS AND PROGRAMS, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE INFORMATION, DOCUMENT OR THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE INFORMATION, DOCUMENTS AND PROGRAMS IS WITH YOU. SHOULD THE INFORMATION, DOCUMENTS OR PROGRAMS PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

6 Limitation of Liability

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE INFORMATION, DOCUMENTS OR PROGRAMS AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE INFORMATION, DOCUMENTS OR PROGRAMS (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE INFORMATION, DOCUMENTS OR PROGRAMS TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.