

Python Packaging

Jakub Wasielak



<http://blog.pykonik.org/>



<https://pl.pycon.org/2017/>

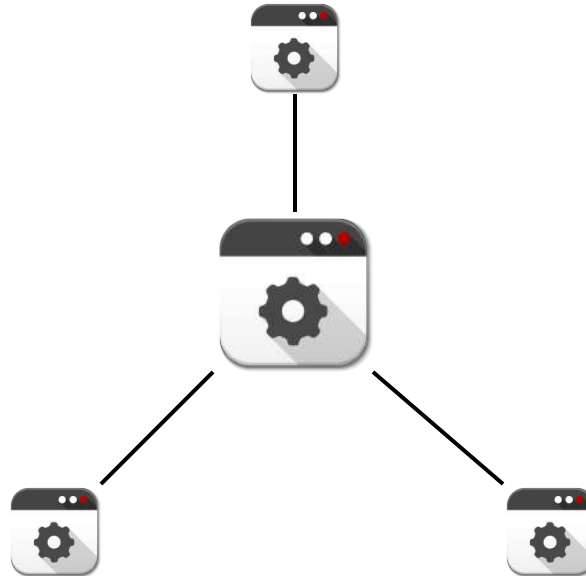


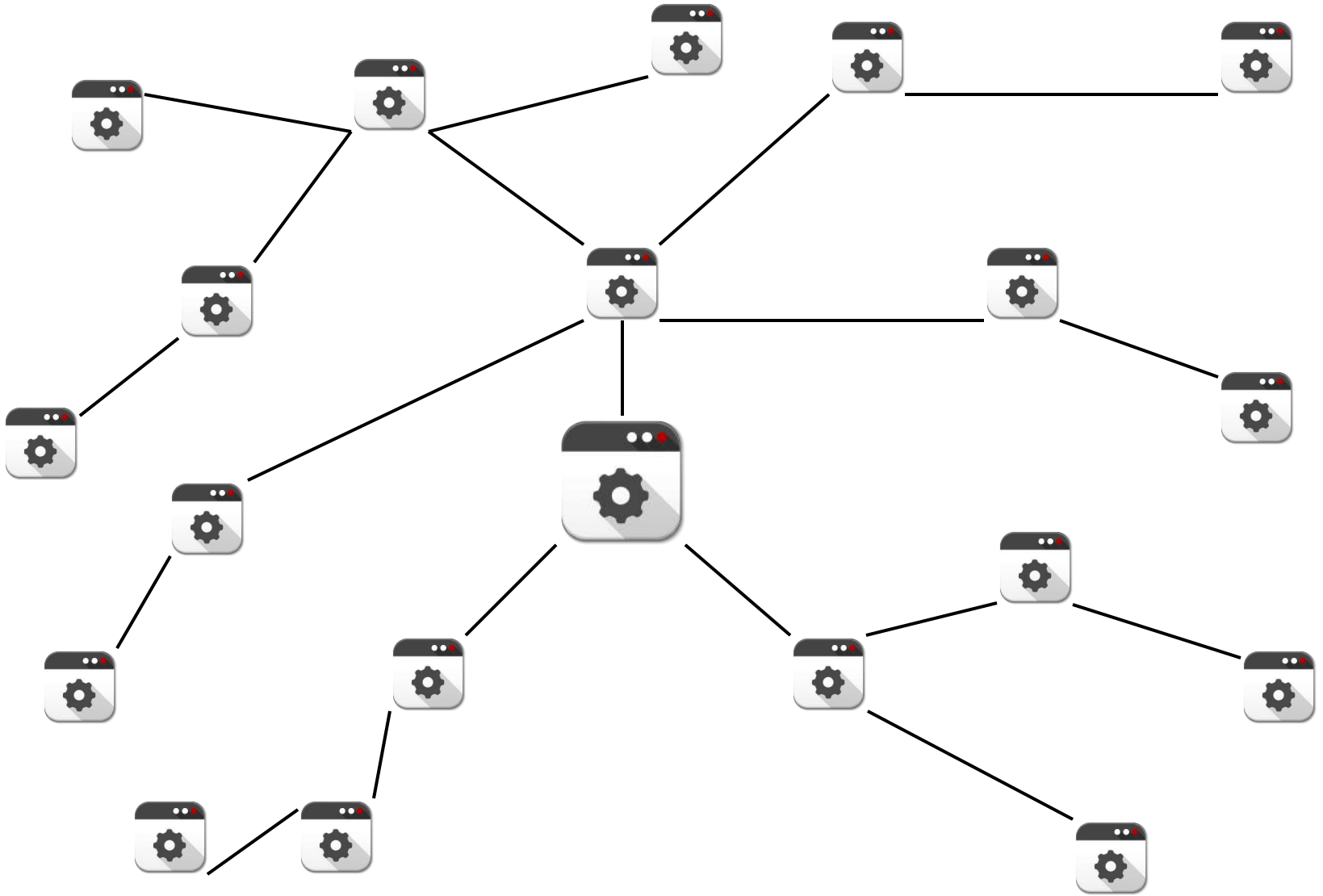
<http://koderek.edu.pl/>
<facebook.com/startechkrk>



What? Why?

Architecture





<https://packaging.python.org/current/>

Installation Tool Recommendations

- pip
- virtualenv

Packaging Tool Recommendations

- setuptools
- bdist_wheel
- twine

Asking for help?

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

```
Standard commands:
```

```
build                build everything needed to install
build_py             "build" pure Python modules (copy to build directory)
build_ext            build C/C++ extensions (compile/link to build directory)
build_clib           build C/C++ libraries used by Python extensions
build_scripts        "build" scripts (copy and fixup #! line)
clean                clean up temporary files from 'build' command
install              install everything from build directory
install_lib          install all Python modules (extensions and pure Python)
install_headers      install C/C++ header files
install_scripts      install scripts (Python or otherwise)
install_data         install data files
sdist                create a source distribution (tarball, zip file, etc.)
register              register the distribution with the Python package index
bdist                create a built (binary) distribution
bdist_dumb           create a "dumb" built distribution
bdist_rpm            create an RPM distribution
bdist_wininst        create an executable installer for MS Windows
upload               upload binary package to PyPI
check                perform some checks on the package
```

```
Extra commands:
```

```
... this one goes on
```

setup.py

```
import os
from setuptools import setup

setup(
    name = "an_example_pypi_project",
    version = "0.0.4",
    author = "Jakub Wasielak",
    author_email = "kuba.wasielak@gmail.com",
    description = ("An demonstration of how to create, document, and publish "
                  "to the cheese shop a5 pypi.org."),
    license = "BSD",
    keywords = "example documentation tutorial",
    url = "http://packages.python.org/an_example_pypi_project",
    packages=['an_example_pypi_project', 'tests'],
    long_description=read('README'),
    classifiers=[
        "Development Status :: 3 - Alpha",
        "Topic :: Utilities",
        "License :: OSI Approved :: BSD License",
    ],
)
```

(source: https://pythonhosted.org/an_example_pypi_project/setuptools.html)

setup.py

```
import os
from setuptools import setup

setup(
    name = "an_example_pypi_project",
    version = "0.0.4",
    author = "Jakub Wasielek",
    author_email = "kuba.wasielek@gmail.com",
    description = ("An demonstration of how to create, document, and publish "
                  "to the cheese shop a5 pypi.org."),
    license = "BSD",
    keywords = "example documentation tutorial",
    url = "http://packages.python.org/an_example_pypi_project",
    packages=['an_example_pypi_project', 'tests'],
    long_description=read('README'),
    classifiers=[
        "Development Status :: 3 - Alpha",
        "Topic :: Utilities",
        "License :: OSI Approved :: BSD License",
    ],
)
```

(source: https://pythonhosted.org/an_example_pypi_project/setuptools.html)

setup.py

```
import os
from setuptools import setup

setup(
    name = "an example pypi_project",
    version = "0.0.4",
    author = "Jakub Wasielesak",
    author_email = "kuba.wasielesak@gmail.com",
    description = ("An demonstration of how to create, document, and publish "
                  "to the cheese shop a5 pypi.org."),
    license = "BSD",
    keywords = "example documentation tutorial",
    url = "http://packages.python.org/an_example_pypi_project",
    packages=['an_example_pypi_project', 'tests'],
    long_description=read('README'),
    classifiers=[
        "Development Status :: 3 - Alpha",
        "Topic :: Utilities",
        "License :: OSI Approved :: BSD License",
    ],
)
```

setuptools_scm

```
setup(  
    name = "an_example_pypi_project",  
    use_scm_version=True,  
    setup_requires=['setuptools_scm'],  
    # ...  
)
```

PEP 440

<https://www.python.org/dev/peps/pep-0440/>

setup.py

```
import os
from setuptools import setup

setup(
    name = "an_example_pypi_project",
    version = "0.0.4",
    author = "Jakub Wasielak",
    author_email = "kuba.wasielak@gmail.com",
    description = ("An demonstration of how to create, document, and publish "
                  "to the cheese shop a5 pypi.org."),
    license = "BSD",
    keywords = "example documentation tutorial",
    url = "http://packages.python.org/an_example_pypi_project",
    packages=['an_example_pypi_project', 'tests'],
    long_description=read('README'),
    classifiers=[
        "Development Status :: 3 - Alpha",
        "Topic :: Utilities",
        "License :: OSI Approved :: BSD License",
    ],
)
```

setup.py

```
import os
from setuptools import setup

setup(
    name = "an_example_pypi_project",
    version = "0.0.4",
    author = "Jakub Wasielesak",
    author_email = "kuba.wasielesak@gmail.com",
    description = ("An demonstration of how to create, document, and publish "
                  "to the cheese shop a5 pypi.org."),
    license = "BSD",
    keywords = "example documentation tutorial",
    url = "http://packages.python.org/an_example_pypi_project",
    packages=['an_example_pypi_project', 'tests'],
    long_description=read('README'),
    classifiers=[
        "Development Status :: 3 - Alpha",
        "Topic :: Utilities",
        "License :: OSI Approved :: BSD License",
    ],
)
```

(source: https://pythonhosted.org/an_example_pypi_project/setuptools.html)

setup.py

```
import os
from setuptools import setup

setup(
    name = "an_example_pypi_project",
    version = "0.0.4",
    author = "Jakub Wasieleski",
    author_email = "kuba.wasieleski@gmail.com",
    description = ("An demonstration of how to create, document, and publish "
                  "to the cheese shop at pypi.org."),
    license = "BSD",
    keywords = "example documentation tutorial",
    url = "http://packages.python.org/an_example_pypi_project",
    packages=['an_example_pypi_project', 'tests'],
    long_description=read('README'),
    classifiers=[
        "Development Status :: 3 - Alpha",
        "Topic :: Utilities",
        "License :: OSI Approved :: BSD License",
    ],
)
```

(source: https://pythonhosted.org/an_example_pypi_project/setuptools.html)

setup.py

```
import os
from setuptools import setup

setup(
    name = "an_example_pypi_project",
    version = "0.0.4",
    author = "Jakub Wasielak",
    author_email = "kuba.wasielak@gmail.com",
    description = ("An demonstration of how to create, document, and publish "
                  "to the cheese shop a5 pypi.org."),
    license = "BSD",
    keywords = "example documentation tutorial",
    url = "http://packages.python.org/an_example_pypi_project",
    packages=['an_example_pypi_project', 'tests'],
    long_description=read('README'),
    classifiers=[
        "Development Status :: 3 - Alpha",
        "Topic :: Utilities",
        "License :: OSI Approved :: BSD License",
    ],
)
```

(source: https://pythonhosted.org/an_example_pypi_project/setuptools.html)

setup.py

File	Type
setuptools-36.0.1-py2.py3-none-any.whl (md5)	Python
setuptools-36.0.1.zip (md5)	Source

Author: Python Packaging Authority

Home Page: <https://github.com/pypa/setuptools>

Keywords: CPAN PyPI distutils eggs package management

Categories

[Development Status :: 5 - Production/Stable](#)

[Intended Audience :: Developers](#)

[License :: OSI Approved :: MIT License](#)

[Operating System :: OS Independent](#)

[Programming Language :: Python :: 2](#)

[Programming Language :: Python :: 2.6](#)

[Programming Language :: Python :: 2.7](#)

[Programming Language :: Python :: 3](#)

[Programming Language :: Python :: 3.3](#)

[Programming Language :: Python :: 3.4](#)

[Programming Language :: Python :: 3.5](#)

[Programming Language :: Python :: 3.6](#)

[Topic :: Software Development :: Libraries :: Python Modules](#)

[Topic :: System :: Archiving :: Packaging](#)

[Topic :: System :: Systems Administration](#)

[Topic :: Utilities](#)

Best classifier?

"Private :: Do Not Upload"

setup.py

```
import os
from setuptools import setup

setup(
    name = "an_example_pypi_project",
    version = "0.0.4",
    author = "Jakub Wasielak",
    author_email = "kuba.wasielak@gmail.com",
    description = ("An demonstration of how to create, document, and publish "
                  "to the cheese shop a5 pypi.org."),
    license = "BSD",
    keywords = "example documentation tutorial",
    url = "http://packages.python.org/an_example_pypi_project",
    packages=['an_example_pypi_project', 'tests'],
    long_description=read('README'),
    classifiers=[
        "Development Status :: 3 - Alpha",
        "Topic :: Utilities",
        "License :: OSI Approved :: BSD License",
    ],
)
```

(source: https://pythonhosted.org/an_example_pypi_project/setuptools.html)

setup.py

```
import os
from setuptools import setup, find_packages

PACKAGES = find_packages(where="src")

setup(
    # ...
    packages=PACKAGES,
    # ...
)
```

(source: https://pythonhosted.org/an_example_pypi_project/setuptools.html)

There's more!

```
setup(  
    name = "an_example_pypi_project",  
    # ...  
    install_requires=[  
        "cherrypy==3.5",  
        "lxml",  
        "Pillow>=2.1,<3dev"  
    ],  
)
```

Extras

```
setup(  
    # ...  
    install_requires=[  
        "cherryypy>=3.5,<3.6dev",  
        "lxml",  
        "Pillow>=2.1,<3dev"  
    ],  
    extras_require=dict(  
        doc = ['Sphinx>=1.3'],  
        notebook = ['notebook', 'ipywidgets'],  
        # ... (^ comes from IPython)  
    )  
)
```

```
python setup.py install 'ipython[notebook]'
```

Tests? Why not!

```
setup(  
    # ...  
    install_requires=[  
        "cherryypy>=3.5,<3.6dev",  
        "lxml",  
        "Pillow>=2.1,<3dev"  
    ],  
    tests_require=[  
        'Pyro>=3.16,<4dev',  
        'pytest>=2.3',  
        'selenium'  
    ]  
)
```

python setup.py test

Or...

```
setup(  
    # ...  
    install_requires=[  
        "cherryypy>=3.5,<3.6dev",  
        "lxml",  
        "Pillow>=2.1,<3dev"  
    ],  
    extras_require={  
        'testing': [  
            'Pyro>=3.16,<4dev',  
            'pytest>=2.3',  
            'selenium'  
        ]  
    }  
)
```

And tox to install

Entry Points

```
setup(  
    # ...  
    'entry_points': {  
        'console_scripts': ['virtualenv=virtualenv:main'],  
    },  
)
```

```
$ python virtualenv.py my_venv
```

vs.

```
$ virtualenv my_venv
```


setup.cfg

```
[global]
verbose = 1

[bdist_wheel]
universal = 1

[metadata]
license_file = LICENSE

[easy_install]
index_url = https://devpi.company.net/root/sth/+simple/

[tool:pytest]
norecursedirs = build env services *.egg project/lib/test
```

MANIFEST.in

```
include CHANGES.txt  
include project/handlers/*.html  
recursive-include project/static
```

setuptools_scm

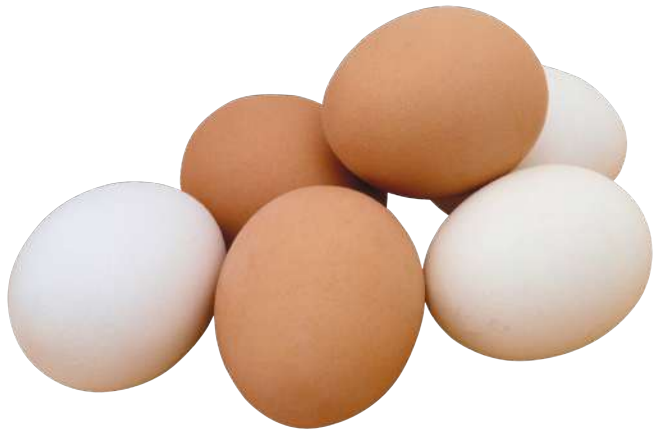
everything not in .gitignore will get used!

install vs. develop

```
$ python setup.py install
...
$ pip freeze | grep project
project==22.2

$ python setup.py develop
...
$ pip freeze | grep project
-e git+ssh://you@your.repo.url/Repositories/Team/project@id_123#egg=project
```

eggs



vs.

wheels

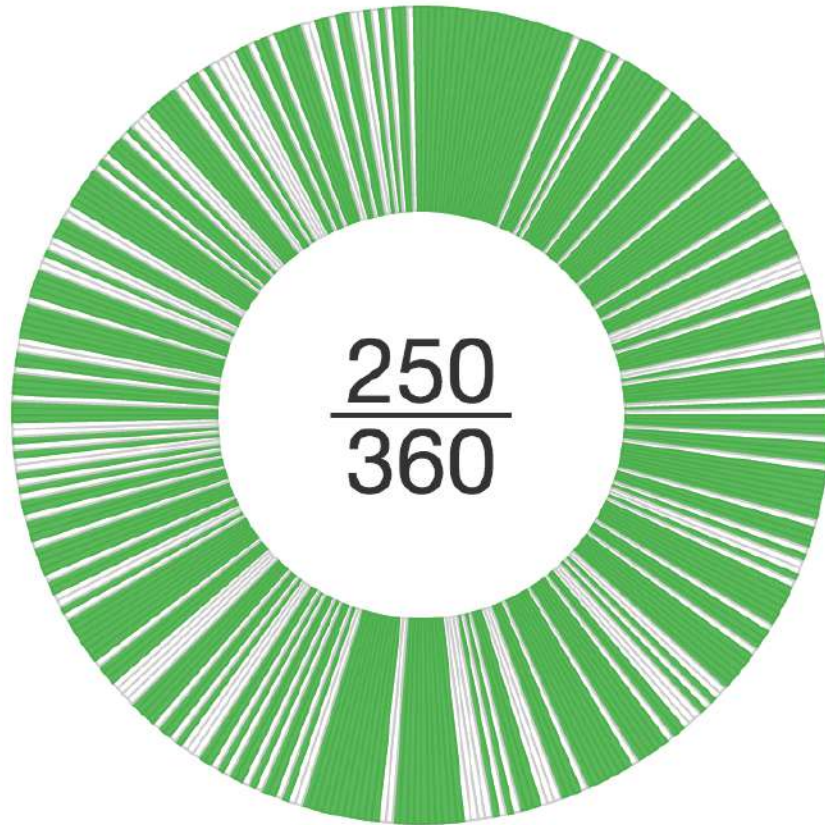


Wheels advantages

distribution-1.0-1-py27-none-any.whl

- official PEP (pep-0427)
- no .pyc files inside (one wheel for both Pythons, .pyc files will be generated upon installation)
- richer file naming convention
- versioning
- installation of C components does not require compiler

Python Wheels



No pycrypto, SQLAlchemy, MySQL-python, tornado

requirements.txt

```
$ pip install requirements-dev.txt
```

```
Collecting requirements-dev.txt
```

```
Installing collected packages: requirements-dev.txt
```

Please reconsider deleting this from pypi #1

 Closed

piquadrat opened this issue on 27 Jan 2016 · 5 comments



piquadrat commented on 27 Jan 2016



I like a good joke as much as the next guy, but I'm really not interested to have "some nasty hacks" running on my machine due to a typo.

Congrats on the idea though!



1

devpi

<http://doc.devpi.net/latest/>

YouGov devpi

Search

[devpi](#) » [root/pypi](#) » [cherrypy](#)

server status ok [How to search](#)

root/pypi/: cherrypy versions

Index	Version
root/pypi	8.1.2
root/pypi	8.1.1
root/pypi	8.1.0
root/pypi	8.0.1
root/pypi	8.0.0
root/pypi	7.1.0
root/pypi	7.0.0
root/pypi	6.2.1
root/pypi	6.2.0
root/pypi	6.1.1
root/pypi	6.1.0
root/pypi	6.0.2
root/pypi	6.0.1
root/pypi	6.0.0
root/pypi	5.6.0

Your projects, your packages

.pypirc

```
[distutils]
index-servers =
    my_devpi

[my_devpi]
repository: https://devpi-master.company.net/root/my_devpi/
username: {your username}
password: {your password}
```

Register

```
python setup.py sdist register -r my_devpi upload -r my_devpi
```

Upload

```
python setup.py sdist upload -r my_devpi
```

Or twine

Create

```
python setup.py sdist bdist_wheel
```

Upload

```
twine upload dist/*
```

Better Upload

```
export TWINE_USERNAME=foo  
export TWINE_PASSWORD=bar  
twine upload dist/*
```

Test your package

```
pip install dist/package-1.0.0.tar.gz  
pip install dist/package-1.0.0-py2.py3-none-any.whl
```

.pypirc

```
[distutils]  
index-servers =  
    test  
  
[test]  
repository: https://testpypi.python.org/pypi  
# repository: https://test.pypi.org/legacy/  
username: {your username}
```

```
pip install -i https://testpypi.python.org/pypi <package_name>
```

PEP 20, last line

Namespaces are one honking great idea
-- let's do more of those!

Namespaces

```
sound/                                Top-level package
  __init__.py                          Initialize the sound package
  formats/                              Subpackage for file format conversions
    __init__.py
    wavread.py
    wavwrite.py
    aiffread.py
    aiffwrite.py
    auread.py
    auwrite.py
    ...
  effects/                              Subpackage for sound effects
    __init__.py
    echo.py
    surround.py
    reverse.py
    ...
  filters/                              Subpackage for filters
    __init__.py
    equalizer.py
    vocoder.py
    karaoke.py
    ...
```

Namespaces

```
import sound.effects.echo
import sound.effects.surround
from sound.effects import *
```

pip cool features

(that easy_install doesn't have)

- easy_install can finish up with a partially completed installation
- better console output
- reasons for actions are kept
- native support for git, mercurial, bazaar
- uninstallation of packages
- pip freeze
- pip install -r requirements.txt

What's next?

<https://www.pypa.io/en/latest/roadmap/>

The screenshot displays the PyPA Roadmap page, which is organized into several sections, each detailing a project milestone. Each section includes a summary, the project name (PYP or PEP), the issue number, and the status. The sections are as follows:

- Python 3.9**: The next release in the Python 3.x series, currently scheduled for 2020. It is expected to be released in the first half of the year. The release will include several new features and improvements to the standard library.
- Core Standards**: A series of standards that define the core of the Python ecosystem. These include PEP 569 (Standard Library Modules), PEP 570 (Standard Library Packages), and PEP 571 (Standard Library Subpackages).
- Python 3.8 Updates**: A series of updates to the Python 3.8 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 3.7 Updates**: A series of updates to the Python 3.7 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 3.6 Updates**: A series of updates to the Python 3.6 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 3.5 Updates**: A series of updates to the Python 3.5 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 3.4 Updates**: A series of updates to the Python 3.4 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 3.3 Updates**: A series of updates to the Python 3.3 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 3.2 Updates**: A series of updates to the Python 3.2 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 3.1 Updates**: A series of updates to the Python 3.1 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 3.0 Updates**: A series of updates to the Python 3.0 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 2.7 Updates**: A series of updates to the Python 2.7 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 2.6 Updates**: A series of updates to the Python 2.6 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 2.5 Updates**: A series of updates to the Python 2.5 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 2.4 Updates**: A series of updates to the Python 2.4 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 2.3 Updates**: A series of updates to the Python 2.3 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 2.2 Updates**: A series of updates to the Python 2.2 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 2.1 Updates**: A series of updates to the Python 2.1 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 2.0 Updates**: A series of updates to the Python 2.0 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 1.9 Updates**: A series of updates to the Python 1.9 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 1.8 Updates**: A series of updates to the Python 1.8 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 1.7 Updates**: A series of updates to the Python 1.7 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 1.6 Updates**: A series of updates to the Python 1.6 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 1.5 Updates**: A series of updates to the Python 1.5 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 1.4 Updates**: A series of updates to the Python 1.4 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 1.3 Updates**: A series of updates to the Python 1.3 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 1.2 Updates**: A series of updates to the Python 1.2 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 1.1 Updates**: A series of updates to the Python 1.1 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.
- Python 1.0 Updates**: A series of updates to the Python 1.0 series, currently scheduled for 2020. These updates will include several new features and improvements to the standard library.

Pipfile

```
[[source]]
url = 'https://pypi.python.org/simple'
verify_ssl = true

[requires]
python_version = '2.7'

[packages]
requests = { extras = ['socks'] }
Django = '>1.10'
pinax = { git = 'git://github.com/pinax/pinax.git', ref = '1.4', editable = true }

[dev-packages]
nose = '*'
```

Using TOML (Tom's Obvious, Minimal Language)

PyPI?

<https://pypi.python.org/pypi>

<https://pypi.org/>

⚠ This is a pre-production deployment of [Warehouse](#). Changes made here affect the production instance of PyPI (pypi.python.org).

Help us improve Python packaging - [Donate today!](#)



[Help](#) [Donate](#) [Login](#) [Register](#)

Find, install and publish Python packages
with the Python Package Index

Search

Or browse projects.

111,816 Projects

737,204 Releases

940,329 Files

221,740 Users

Warehouse

<https://github.com/pypa/warehouse>



Recommended reading

- **Tool Recommendations:**

<https://packaging.python.org/current/>

- **Wheel vs. Egg:**

https://packaging.python.org/wheel_egg/

- **Getting Started With setuptools and setup.py:**

https://pythonhosted.org/an_example_pypi_project/setuptools.html

- **Sharing Your Labor of Love: PyPI Quick and Dirty:**

<https://hynek.me/articles/sharing-your-labor-of-love-pypi-quick-and-dirty/>

- **PyPA**

<https://www.pypa.io>

THANKS!

QUESTIONS?

<https://about.me/jakub.wasielak>