

# INSTALLATION INSTRUCTIONS

[gist.github.com/stephsamson](https://gist.github.com/stephsamson)

# Package and Dependency Management with Poetry

Steph Samson  
[hello@stephsamson.com](mailto:hello@stephsamson.com)  
[github.com/stephsamson](https://github.com/stephsamson)



Mercedes-Benz .iO



- 🌟 timeline of Python packaging
- 🌟 configuration files
- 🌟 PEP 517 & PEP 518
- 🌟 Poetry

# A Brief Timeline of Python Packaging History

# pre-2000

```
17437
17438 =====
17439 ==> Release 0.9.0 (February 1991) <==
17440 =====
17441
17442 Original posting to alt.sources.
```

BinSearch -- Newsgroup infor... x BinSearch -- Newsgroup infor... x +

https://www.binsearch.info/groupinfo.php Zoeken

## Groups sorted by size

Only the most popular groups are shown.  
Alternatively, you can [view the other groups indexed](#)

	Newsgroup	Current retention	Number of files	Number of parts	Total size of files
1.	<a href="#">alt.binaries.boneless</a>	1341 days	109965309	10434730503	4.9 PB
2.	<a href="#">alt.binaries.hdtv</a>	1341 days	15763165	4042336871	1.96 PB
3.	<a href="#">alt.binaries.mom</a>	1341 days	27025531	4270740001	1.61 PB
4.	<a href="#">alt.binaries.cores</a>	1341 days	23753654	4237395623	1.55 PB
5.	<a href="#">alt.binaries.nl</a>	1341 days	23366853	3380009284	1.28 PB
6.	<a href="#">alt.binaries.ath</a>	1341 days			
7.	<a href="#">alt.binaries.erotica</a>	1341 days			
8.	<a href="#">alt.binaries.dvd</a>	1341 days			
9.	<a href="#">alt.binaries.multimedia</a>	1341 days			
10.	<a href="#">alt.binaries.test</a>	1341 days			
11.	<a href="#">alt.binaries.bloaf</a>	1341 days			
12.	<a href="#">alt.binaries.misc</a>	1341 days			
13.	<a href="#">alt.binaries.x</a>	1341 days			
14.	<a href="#">alt.binaries.warez</a>	1341 days			
15.	<a href="#">alt.binaries.movies</a>	1341 days			
396.	<a href="#">alt.binaries.ftn.music</a>	915 days			
397.	<a href="#">alt.binaries.pictures.anime</a>	550 days	106	114	15.59 MB
398.	<a href="#">alt.binaries.multimedia.cartoons.repost</a>	1170 days	6	11	2.28 MB

3128. [alt.binaries.dvdr.asian](#)

<< < 3128 records > >>

### All groups

Total number of files in database: 374822647  
Total number of parts (messages) in database: 26795248915  
Total size of files: 10.35 PB  
Copyright © 2006-2011 binsearch - [disclaimer](#)

### All groups

Total number of files in database: 740351623  
Total number of parts (messages) in database: 56749108979  
Total size of files: 23.6 PB  
Copyright © 2006-2011 binsearch - [disclaimer](#)

23.6PB = Most popular groups =  
<https://www.binsearch.info/groupinfo.php>  
10.35PB = Other groups =  
<https://www.binsearch.info/groupinfo.php?server=2>  
33.95PB = Total



# Vaults of Parnassus

## Notes on the History of the Site:

Frustration is a cousin's nephew's father's aunt of invention. Truly, I just got tired of searching [USENET](#) for things---recalling seeing things posted, and then often not being able to find them again (at least not easily). Others suffer this too, I notice. Various projects to collect python resources have been tried; I figured another can't hurt, can it? Who knows. Who am I to say? It is for you to decide. It is for fate to determine.

It may not be an original idea, and the implementation may be quirky (but hopefully not too flaky), still I comfort myself with the thought that if nothing else, this is the only one with a "*random* resource" function. Is that not reason enough? I think so. That is my way.

*“Various projects to collect Python resources have been tried; I figured another can’t hurt, can it?”*

*- Vaults of Parnassus author*





# Vaults of Parnassus : Python Resources.



[About](#) [Feedback](#) [Latest](#) [Random](#) [Stats](#) [Submit](#) [Tree](#)

Vault:

Find:

➤	<b>Applications</b> - Ready to run programs written in, or using, Python.	(5148)
➤	<b>Database</b> - Database related.	(73)
➤	<b>Games</b> - And game systems.	(45)
➤	<b>Graphics</b> - Graphics processing, manipulation, display, etc.	(1164)
➤	<b>Info/Books/Tutorials</b> - In a word: reading material.	(310)
➤	<b>Internet</b> - Specifically Internet related networking stuff.	(6124)
➤	<b>Lost/Broken Links</b> - Broken Links and things that need updating. Can you help?	(26)
➤	<b>Math</b> - Mmmmm, crunchy.	(1168)
➤	<b>Miscellany</b> - Odds and ends, and otherwise uncategorised things.	(51)
➤	<b>Networking</b> - Networks, distributed computing, sockets, etc.	(3132)
➤	<b>O/S Support</b> - Modules and tools specifically targeting various operating systems.	(1136)
➤	<b>Programming Tasks</b> - Datatypes, parsing, algorithms, files, etc.	(414)
➤	<b>Python Tools/Extensions</b> - Patches and other enhancements for making Python behave the way you want.	(2162)
➤	<b>Sound/Audio</b> - More than a slight rustling in the grass.	(45)
➤	<b>User Interfaces</b> - Text and GUI interfaces, widgets and controls for use with python.	(218)
➤	<b>Web Sites</b> - Python related web sites (personal, user group, non-english, collections, etc).	(3126)

Parnasus Totals: 1386 items in 49 categories.

Find:

in

Title/Short



sorted by

Date



-



matching

All Strings



[download]





1998 / distutils-sig

2000 / distutils

```
$ python setup.py build
```

# 2001 / PEP 241 – Metadata for Python Software Packages



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

# Source Distributions

**sdist**

# Built Distributions

**bdist**



2004 / setuptools, easy\_install,  
and .egg 🐍🥚

2005 / package files hosted on PyPI  
for the first time 🚀

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

2005

✓ / packaging index

✓ / package installer

✓ / packaging distribution tools



2005

- ✓ / packaging index
- ✓ / package installer
- ✓ / packaging distribution tools
- × / package version management

2005

- ✓ / packaging index
- ✓ / package installer
- ✓ / packaging distribution tools
- × / package version management
  - × / package uninstaller

2007 / virtualenvs

2008 / pip



1990 / 🐍

2000 / distutils

2003 / PyPI 🚀

2004 / setuptools

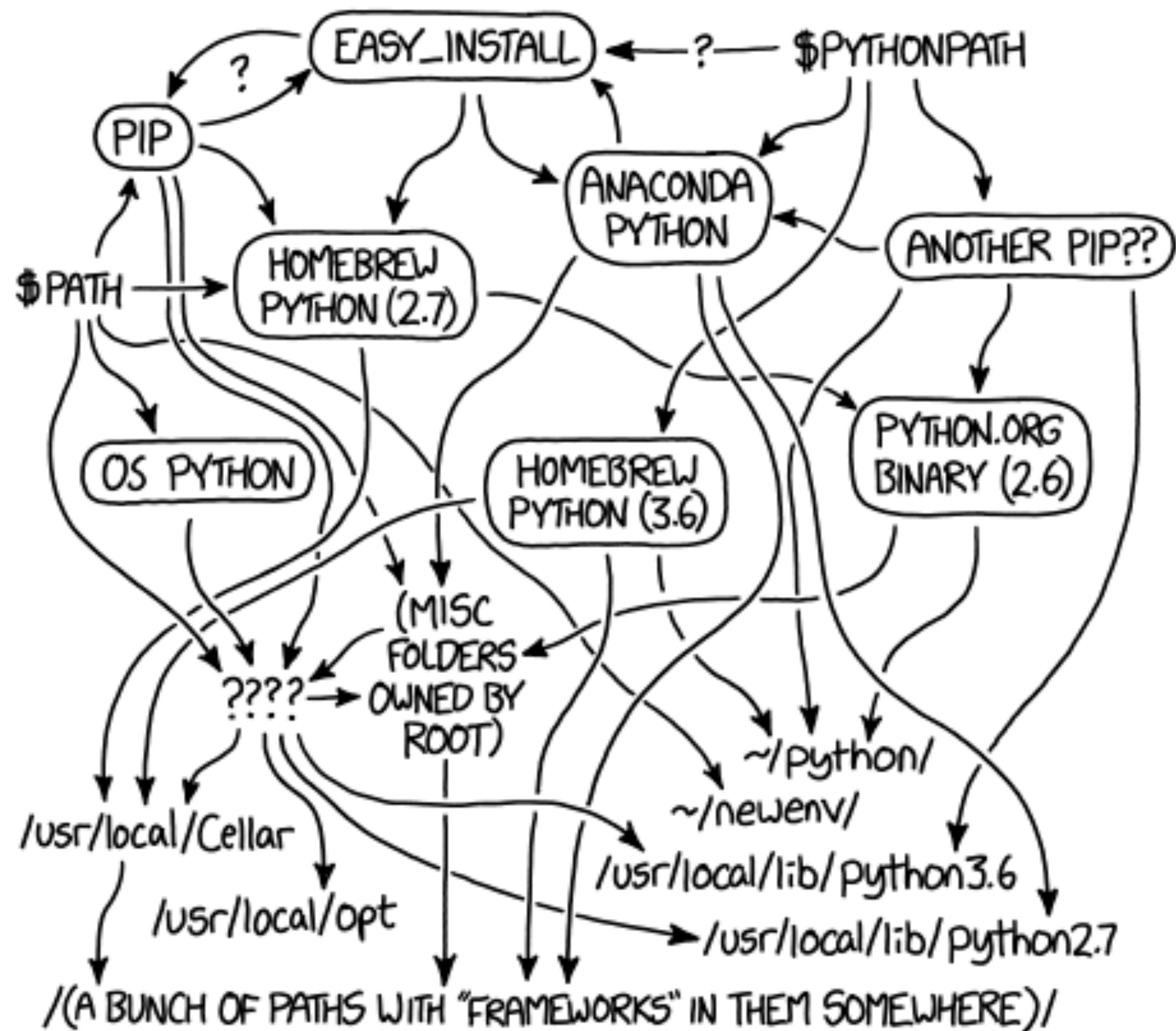
2005 / 1<sup>st</sup> 📦 on PyPI

2007 / virtualenvs

2008 / pip

2013 / wheel 

*Great! Now I can package libraries,  
install them, and even distribute  
them! 🎉*



MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED  
THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

**pip**

**virtualenv**

**\* pip-tools (or Pipenv)**



pip

virtualenv

\* pip-tools (or Pipenv)

setuptools

twine

**With many dependencies,  
comes many config files.**

requirements.txt

requests==2.22.0  
pytest==4.5.0

```
pip install -r requirements.txt
```

**setup.py**



```
# src: https://github.com/pytest-dev/pytest/blob/master/setup.py
```

```
from setuptools import setup
```

```
INSTALL_REQUIRES = ["py>=1.5.0", "six>=1.10.0", # ...]
```

```
setup(  
    # ...  
    package_dir={"": "src"},  
    extras_require={  
        "testing": [  
            "argcomplete",  
            "hypothesis>=3.56",  
            "nose",  
            "requests",  
            "mock;python_version=='2.7'",  
        ],  
    },  
    install_requires=INSTALL_REQUIRES,  
)
```

setup.cfg

```
[metadata]
```

```
name = awesomepackage
```

```
version = 0.1.0
```

```
description = Something awesome 🌟
```

```
author = Steph Samson
```

```
license = MIT
```

```
url = https://awesomepackage.com
```

```
[bdist_wheel]
```


```
universal = 1
```

**MANIFEST.in**

```
include README.md  
include requirements.txt  
reverse-include data *
```

# Recap





- MANIFEST.in
- requirements.txt
- setup.cfg
- setup.py



- MANIFEST.in
- requirements.txt
- setup.cfg
- setup.py
- Pipfile





**IGHT IMMA HEAD OUT**





PYTHON PACKAGING AND DEPENDENCY MANAGEMENT MADE EASY

Poetry

Sebastián Eustace  
[poetry.eustace.io](https://poetry.eustace.io) | [sebastien@eustace.io](mailto:sebastien@eustace.io)

# HOW STANDARDS PROLIFERATE: (SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC)

SITUATION:  
THERE ARE  
14 COMPETING  
STANDARDS.

14?! RIDICULOUS!  
WE NEED TO DEVELOP  
ONE UNIVERSAL STANDARD  
THAT COVERS EVERYONE'S  
USE CASES.



SOON:

SITUATION:  
THERE ARE  
15 COMPETING  
STANDARDS.

# pypproject.toml

<https://snarky.ca/clarifying-pep-518/> | <https://www.python.org/dev/peps/pep-0518/>



# pypproject.toml

## PEP 517 -- A build-system independent format for source trees

### Abstract

While `distutils` / `setuptools` have taken us a long way, they suffer from three serious problems: (a) they're missing important features like usable build-time dependency declaration, autoconfiguration, and even basic ergonomic niceties like [DRY](#)-compliant version number management, and (b) extending them is difficult, so while there do exist various solutions to the above problems, they're often quirky, fragile, and expensive to maintain, and yet (c) it's very difficult to use anything else, because `distutils/setuptools` provide the standard interface for installing packages expected by both users and installation tools like `pip`.

Previous efforts (e.g. `distutils2` or `setuptools` itself) have attempted to solve problems (a) and/or (b). This proposal aims to solve (c).

The goal of this PEP is get `distutils`-sig out of the business of being a gatekeeper for Python build systems. If you want to use `distutils`, great; if you want to use something else, then that should be easy to do using standardized methods. The difficulty of interfacing with `distutils` means that there aren't many such systems right now, but to give a sense of what we're thinking about see [flit](#) or [bento](#). Fortunately, wheels have now solved many of the hard problems here -- e.g. it's no longer necessary that a build system also know about every possible installation configuration -- so pretty much all we really need from a build system is that it have some way to spit out standard-compliant wheels and sdist.

We therefore propose a new, relatively minimal interface for installation tools like `pip` to interact with package source trees and source distributions.

## PEP 518 -- Specifying Minimum Build System Requirements for Python Projects

### Abstract

This PEP specifies how Python software packages should specify what build dependencies they have in order to execute their chosen build system. As part of this specification, a new configuration file is introduced for software packages to use to specify their build dependencies (with the expectation that the same configuration file will be used for future configuration details).

### Rationale

When Python first developed its tooling for building distributions of software for projects, `distutils` [\[1\]](#) was the chosen solution. As time went on, `setuptools` [\[2\]](#) gained popularity to add some features on top of `distutils`. Both used the concept of a `setup.py` file that project maintainers executed to build distributions of their software (as well as users to install said distribution).

Using an executable file to specify build requirements under `distutils` isn't an issue as `distutils` is part of Python's standard library. Having the build tool as part of Python means that a `setup.py` has no external dependency that a project maintainer needs to worry about to build a distribution of their project. There was no need to specify any dependency information as the only dependency is Python.

requirements.txt  
setup.py  
setup.cfg  
MANIFEST.in



pypproject.toml

requirements.txt

setup.py

setup.cfg

MANIFEST.in



pypproject.toml

```
[tool.poetry.dependencies]  
python = "^3.7"  
requests = "2.22.0"
```

```
[tool.poetry.dev-dependencies]  
pytest = "^4.5"  
tox = "^3.0"
```

requirements.txt

setup.py

setup.cfg

MANIFEST.in



pypproject.toml

```
[tool.poetry]  
# ...  
packages = [{ include = "pytest", from = "src" }]
```

```
[tool.poetry.dependencies]  
python = "^3.7"  
requests = "2.22.0"
```

```
[tool.poetry.dev-dependencies]  
pytest = "^4.5"  
tox = "^3.0"
```



requirements.txt

setup.py

setup.cfg

MANIFEST.in



pypproject.toml

```
[tool.poetry]
name = "awesomepackage"
version = "0.1.0"
description = "Something awesome 🌟"
authors = ["stephsamson <hello@stephsamson.com>"]
license = "MIT"
# ...
```

requirements.txt

setup.py

setup.cfg

**MANIFEST.in**



pypproject.toml

```
[tool.poetry]
# ...
readme = "README.rst"
include = ["package/**/*.py", "package/**/*.c"]
exclude = ["package/excluded.py"]
```

# Scripts for the Command Line

```
# src: https://poetry.eustace.io/docs/pyproject/#scripts
```

```
[tool.poetry.scripts]  
poetry = 'poetry:console.run'
```

# Using Poetry



[poetry.eustace.io](https://poetry.eustace.io)

# INSTALLATION INSTRUCTIONS

[gist.github.com/stephsamson](https://gist.github.com/stephsamson)

```
pip install poetry --index-url http://10.66.5.80:8080/  
--trusted-host 10.66.5.80 poetry
```

username: **steph**

password: **poetry**

# virtualenvs

<https://poetry.eustace.io/docs/basic-usage/#poetry-and-virtualenvs>

Projects are *Always* Isolated

```
poetry config settings.virtualenvs.create false
```

poetry new awesomepackage

```
awesomепackage/  
├── README.rst  
├── awesomепackage  
│   ├── __init__.py  
├── pyproject.toml  
├── tests  
│   ├── __init__.py  
│   └── test_awesomepackage.py
```



```
cd existingproject && poetry init
```

```
existingpackage/  
├── README.rst  
├── src  
├── pypproject.toml  
└── tests
```

poetry install

```
awesompackage/  
├── README.rst  
├── awesompackage  
│   └── __init__.py  
├── poetry.lock  
├── pyproject.toml  
└── tests  
    ├── __init__.py  
    └── test_awesomepackage.py
```

poetry add pendulum

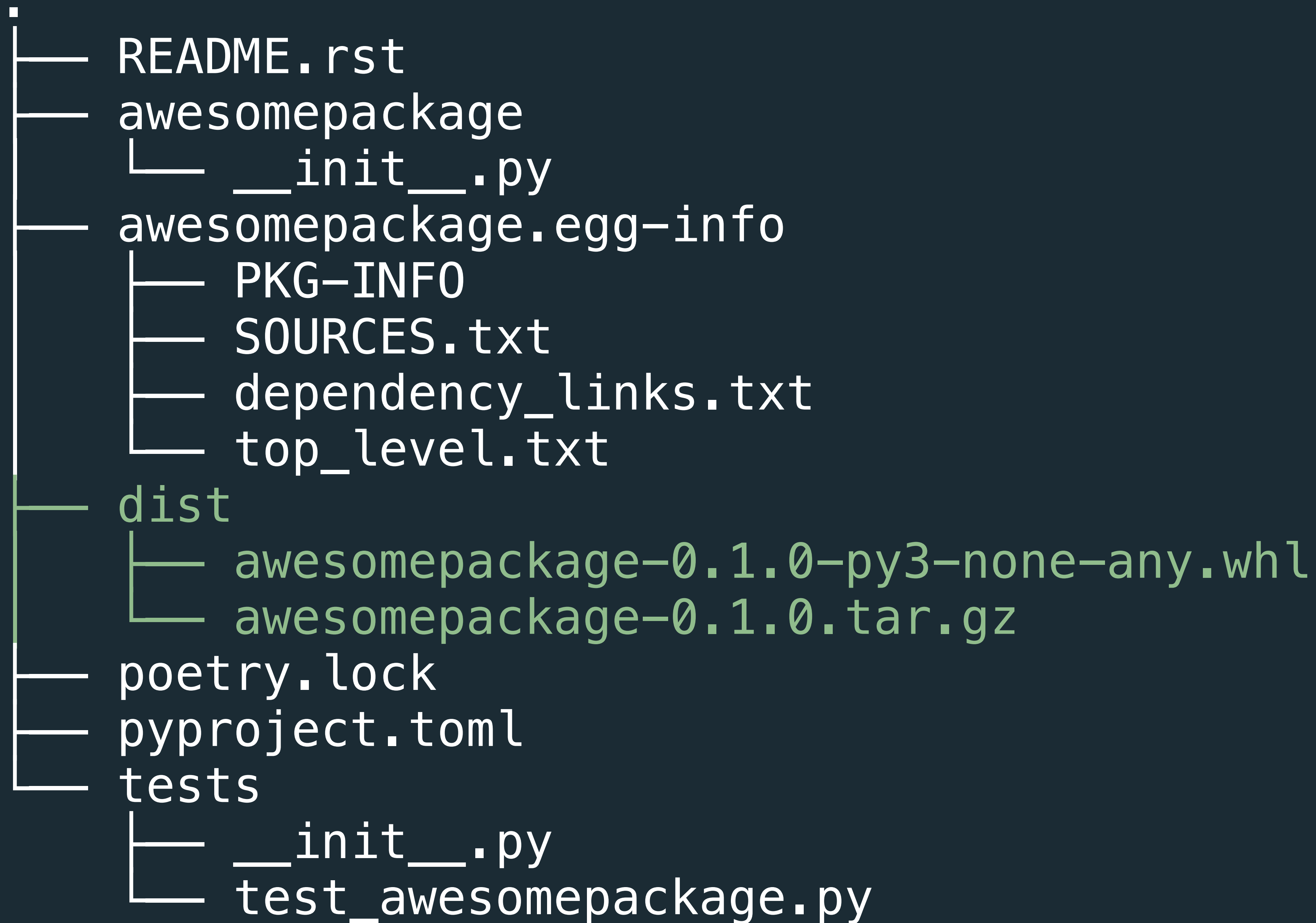
```
[tool.poetry]  
# ...  
packages = [{ include = "pytest", from = "src" }]
```

```
[tool.poetry.dependencies]  
python = "^3.7"  
requests = "2.22.0"  
pendulum = "^2.0"
```

```
[tool.poetry.dev-dependencies]  
pytest = "^4.5"  
tox = "^3.0"
```

poetry build





```
poetry config repositories.workshop  
    http://10.66.5.80:8080/
```

```
poetry publish -r workshop
```

**tox: No setup.py?**  
**No Problem**

<https://github.com/sdispater/poetry/issues/222>

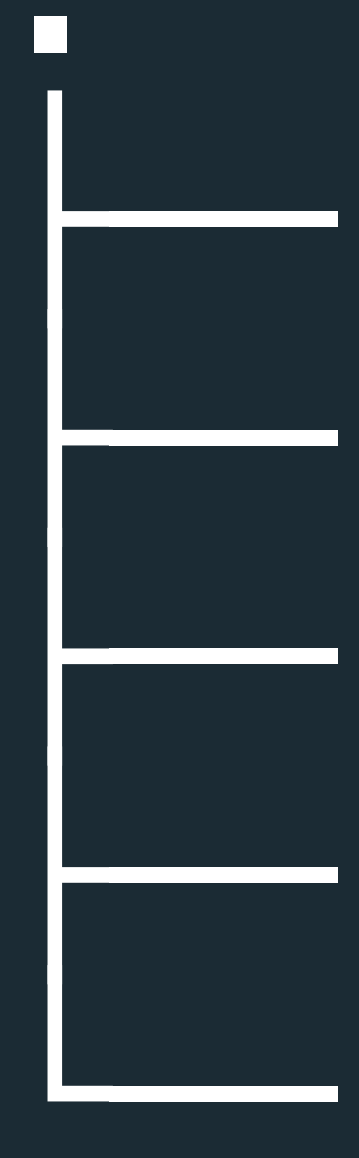
```
[tox]
skipsdist = True
envlist = py27, py35, py36, pypy, pypy3
```

```
[testenv]
whitelist_externals = poetry
skip_install = true
commands =
    poetry install -v
    poetry run pytest tests/
```

```
[testenv:pypy]
whitelist_externals =
    bash
    poetry
skip_install = true
commands =
    poetry install -v
    poetry run pytest tests/
```

```
[tool.tox]
legacy_tox_ini = """
[tox]
skipsdist = True
envlist = py27, py35, py36, pypy, pypy3
```

```
[testenv]
whitelist_externals = poetry
skip_install = true
commands =
    poetry install -v
    poetry run pytest tests/
"""
```



```
graph TD; A[ ] --- B[MANIFEST.in]; A --- C[requirements.txt]; A --- D[setup.cfg]; A --- E[setup.py]; A --- F[tox.ini];
```

MANIFEST.in  
requirements.txt  
setup.cfg  
setup.py  
tox.ini

└─ pypproject.toml

Thanks!

Questions?



# Resources

- Poetry Documentation - [poetry.eustace.io](https://poetry.eustace.io)
- What about Pipenv? - <https://github.com/sdispater/poetry#what-about-pipenv>
- Clarifying PEP 518 (a.k.a. pyproject.toml) - <https://snarky.ca/clarifying-pep-518/>
- Python Packaging User Guide - <https://packaging.python.org/>