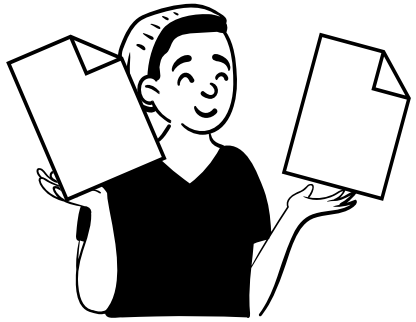
The Python packaging ecosystem

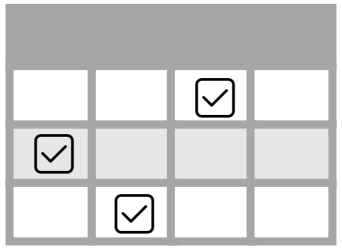
– Simple guidelines for packaging

PyCon Sweden
2023-11-09
Yngve Mardal Moe

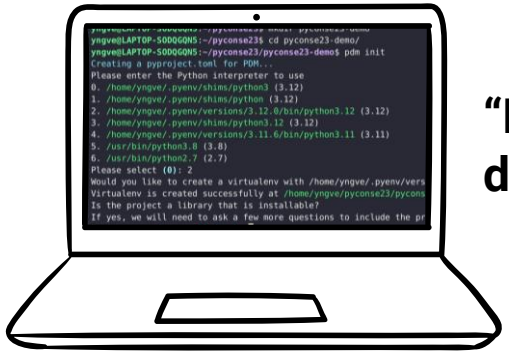
bouvet



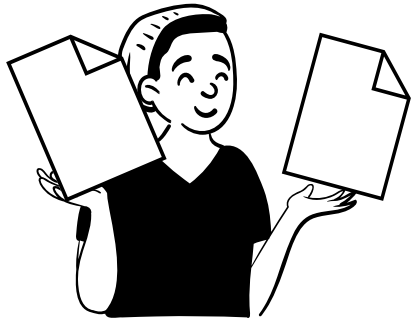
Introduction and brief overview of packaging



Comparison of packaging tools



“Live” demo

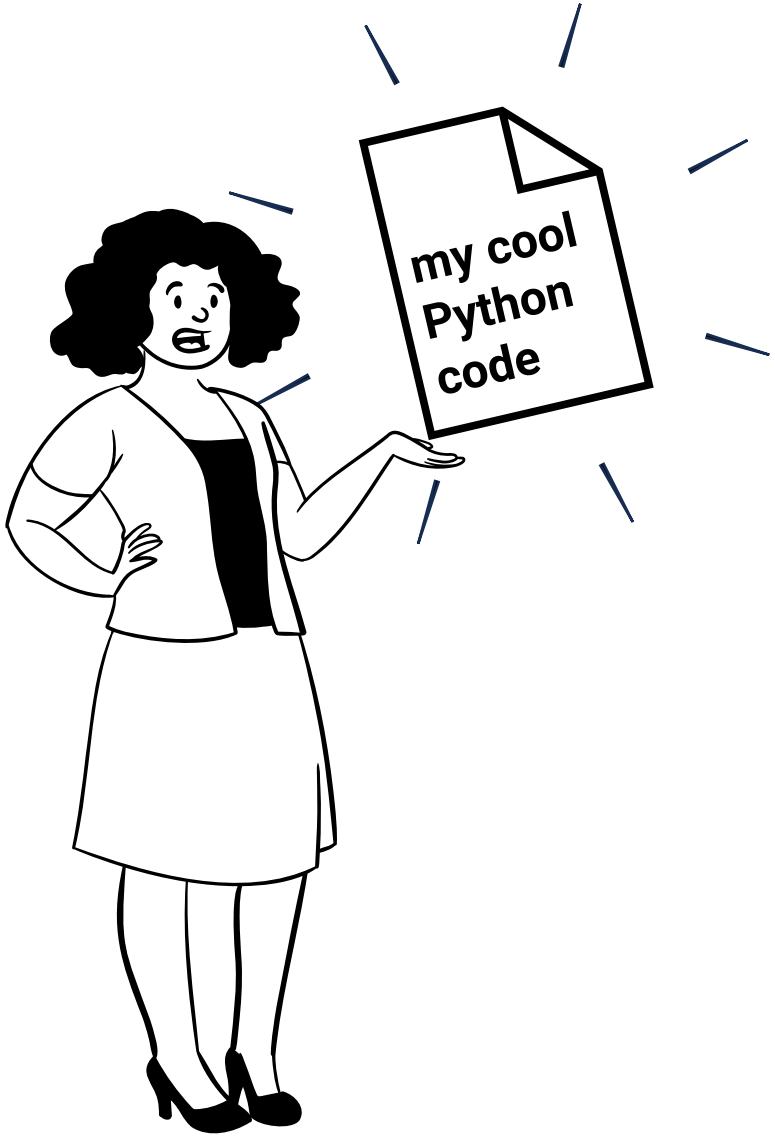


Introduction and brief overview of packaging

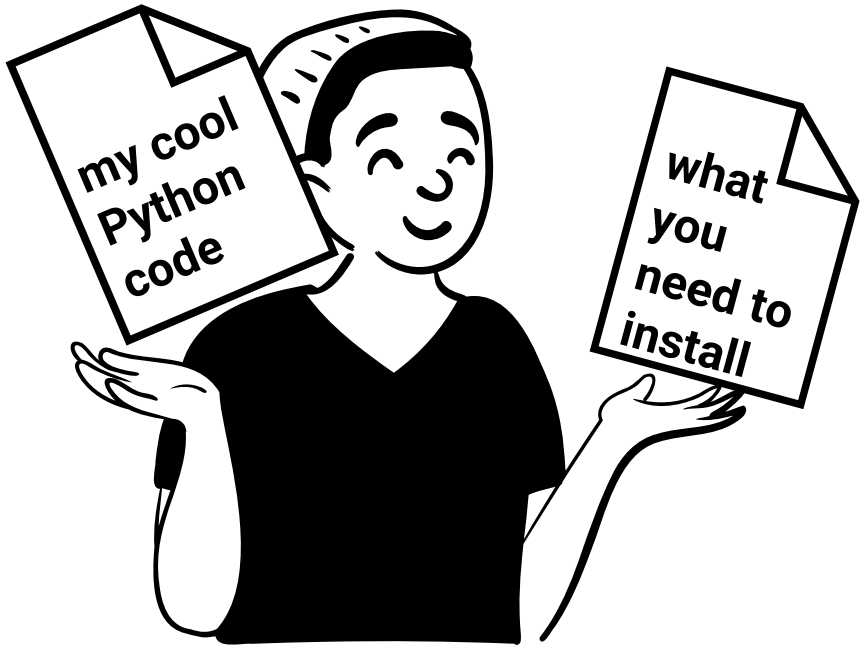
Comparison of packaging tools

“Live” demo

Packaging is about preparing your code to be shared with others



The easiest way to package your python project is to bundle it with a list of software requirements




```
├── my_cool_project
│   ├── __init__.py
│   ├── main.py
│   └── utils.py
└── requirements.txt
```

We want to make our code installable by package managers like pip or conda

pip and conda are designed to solve different problems

pip

Install python packages
and python dependencies

pip and conda are designed to solve different problems

pip

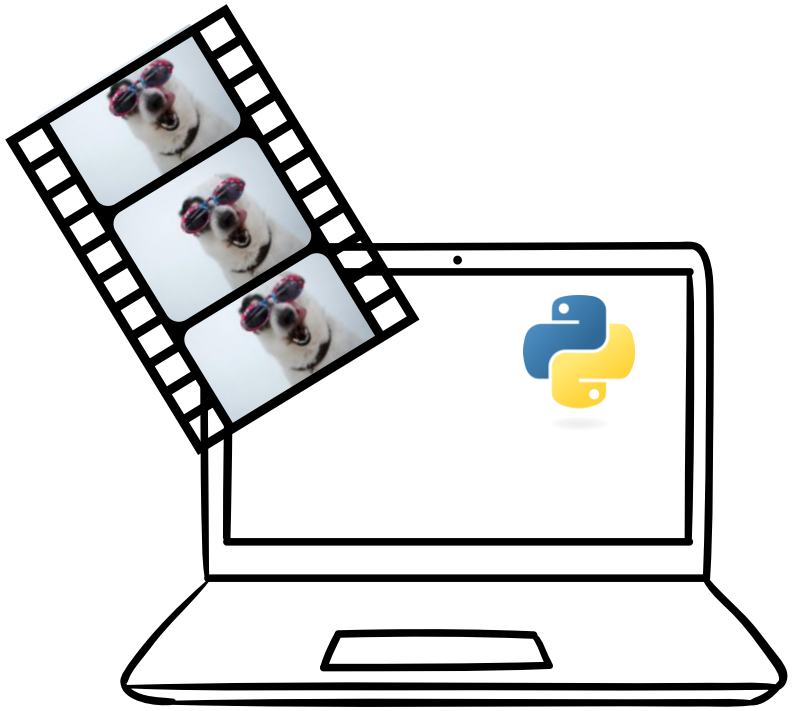
Install python packages
and python dependencies

Install general dependencies including non-python
packages, system dependencies and applications

pip and conda are designed to solve different problems

Install general dependencies including non-python packages, system dependencies and applications

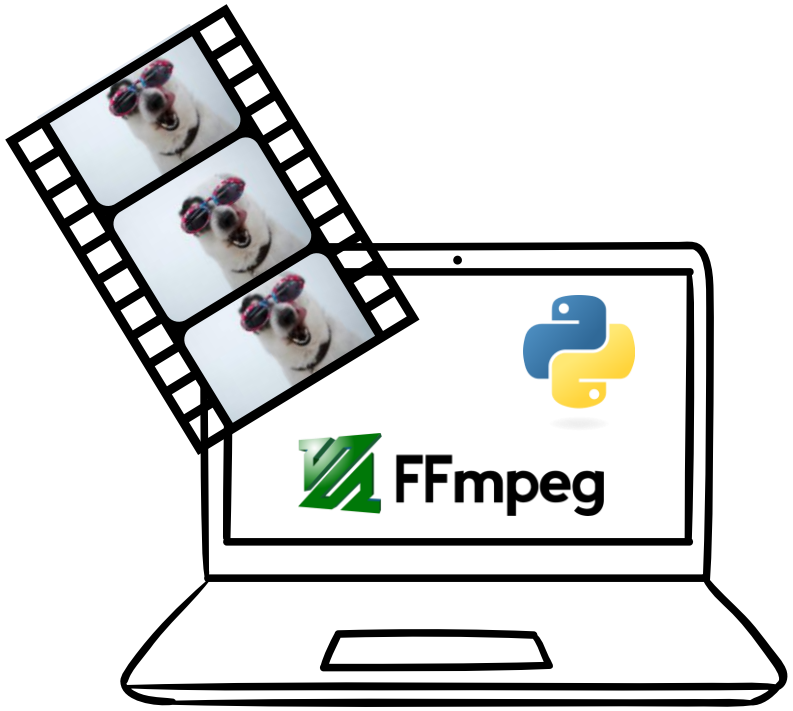
pip and conda are designed to solve different problems



CONDA

Install general dependencies including non-python packages, system dependencies and applications

pip and conda are designed to solve different problems



CONDA

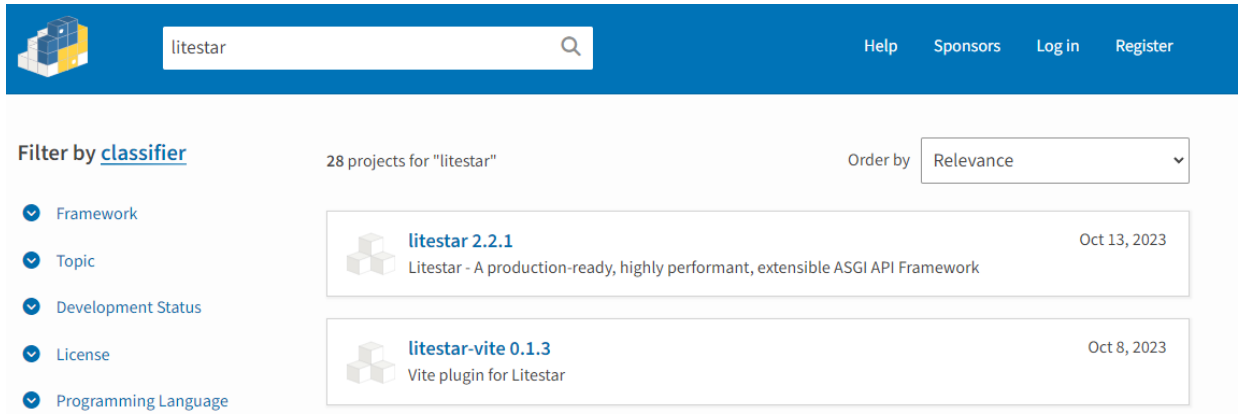
Install general dependencies including non-python packages, system dependencies and applications

Since pip has a smaller scope, it is also easier to understand how it works

```
$ pip install litestar
```

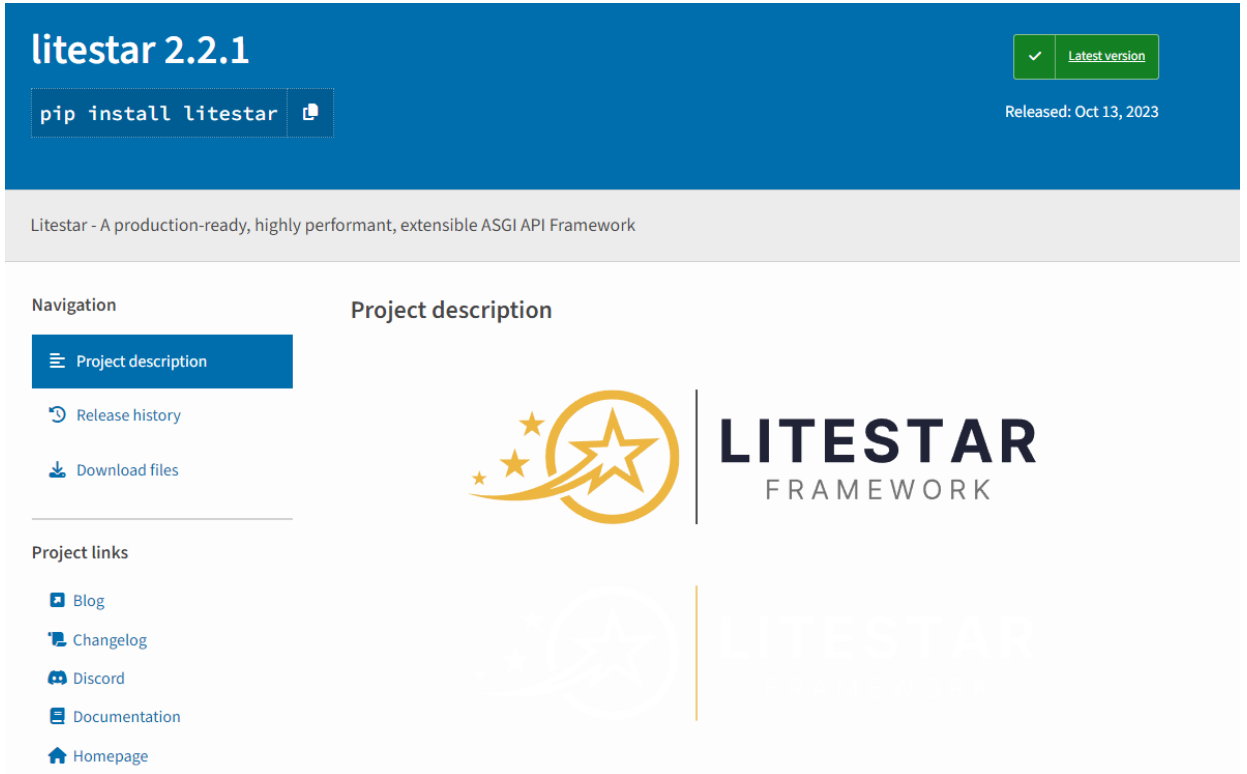
Pip starts by searching PyPI for available packages

```
$ pip install litestar
Collecting litestar
```



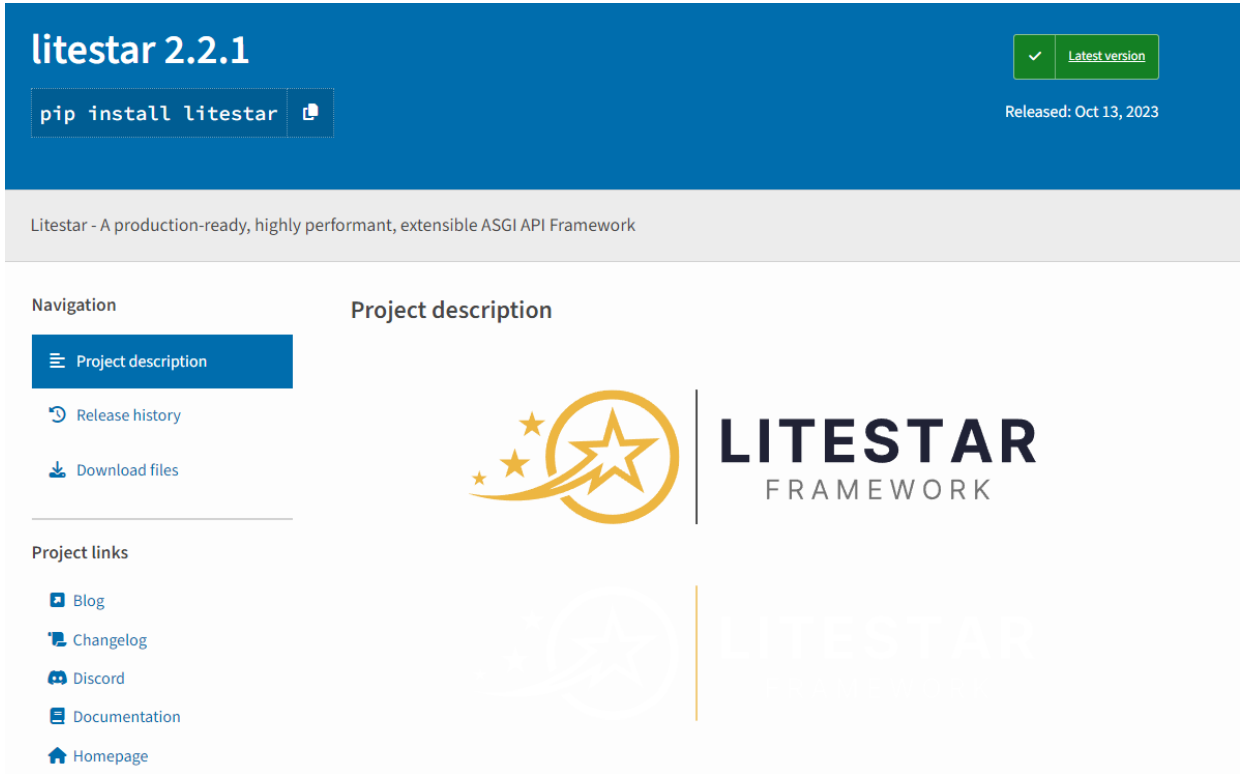
Once pip finds a suitable package, it downloads the package metadata

```
$ pip install litestar
Collecting litestar
 Downloading litestar-2.2.1-py3-none-
any.whl.metadata (82 kB)

82.7/82.7 kB 1.2 MB/s eta 0:00:00
```



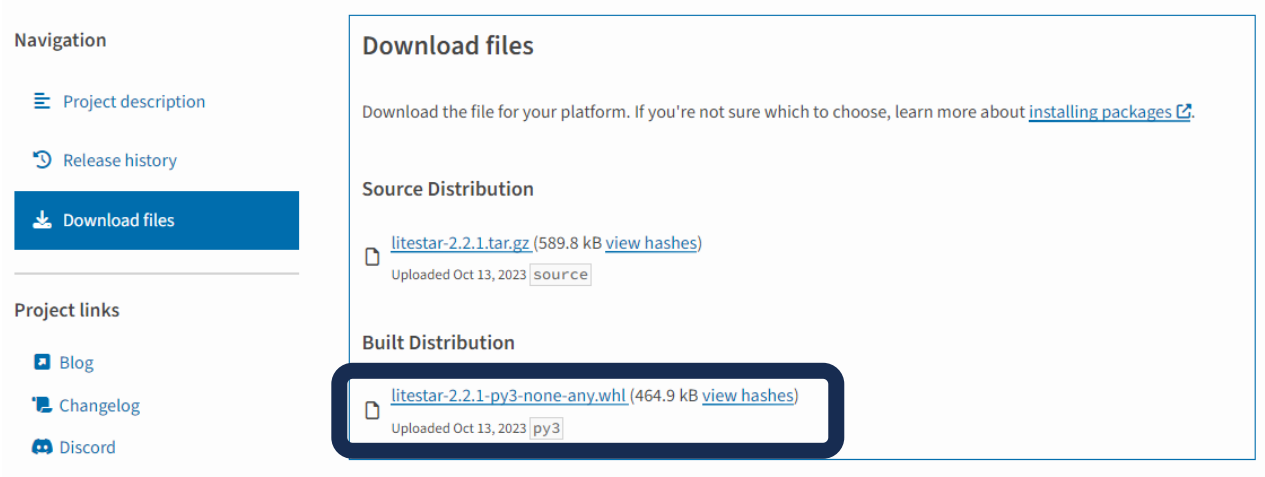
Then, pip installs the required dependencies

```
$ pip install litestar
Collecting litestar
 Downloading litestar-2.2.1-py3-none-
any.whl.metadata (82 kB)

     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 82.7/82.7 kB 1.2 MB/s eta 0:00:00
Requirement already satisfied:
anyio>=3 in
/home/yngve/.pyenv/versions/3.12.0/lib
/python3.12/site-packages
[…]
```



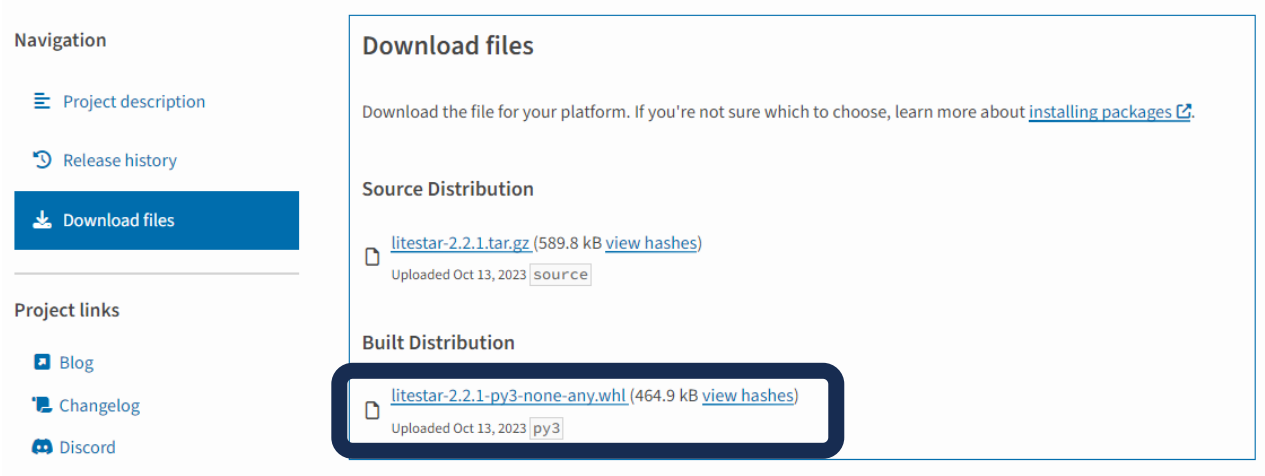
Once all dependencies are installed, pip downloads the package itself

```
$ pip install litestar
Collecting litestar
 Downloading litestar-2.2.1-py3-none-
any.whl.metadata (82 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 82.7/82.7 kB 1.2 MB/s eta 0:00:00
Requirement already satisfied:
anyio>=3 in
/home/yngve/.pyenv/versions/3.12.0/lib
/python3.12/site-packages
[…]
Downloading litestar-2.2.1-py3-none-
any.whl (474 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 474.8/474.8 kB 4.2 MB/s eta 0:00:00
```



Finally, pip extracts the wheel file to the site-packages directory

```
$ pip install litestar
Collecting litestar
 Downloading litestar-2.2.1-py3-none-
any.whl.metadata (82 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 82.7/82.7 kB 1.2 MB/s eta 0:00:00
Requirement already satisfied:
anyio>=3 in
/home/yngve/.pyenv/versions/3.12.0/lib
/python3.12/site-packages
[…]
Downloading litestar-2.2.1-py3-none-
any.whl (474 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 474.8/474.8 kB 4.2 MB/s eta 0:00:00
Installing collected packages:
litestar
Successfully installed litestar-2.2.1
```

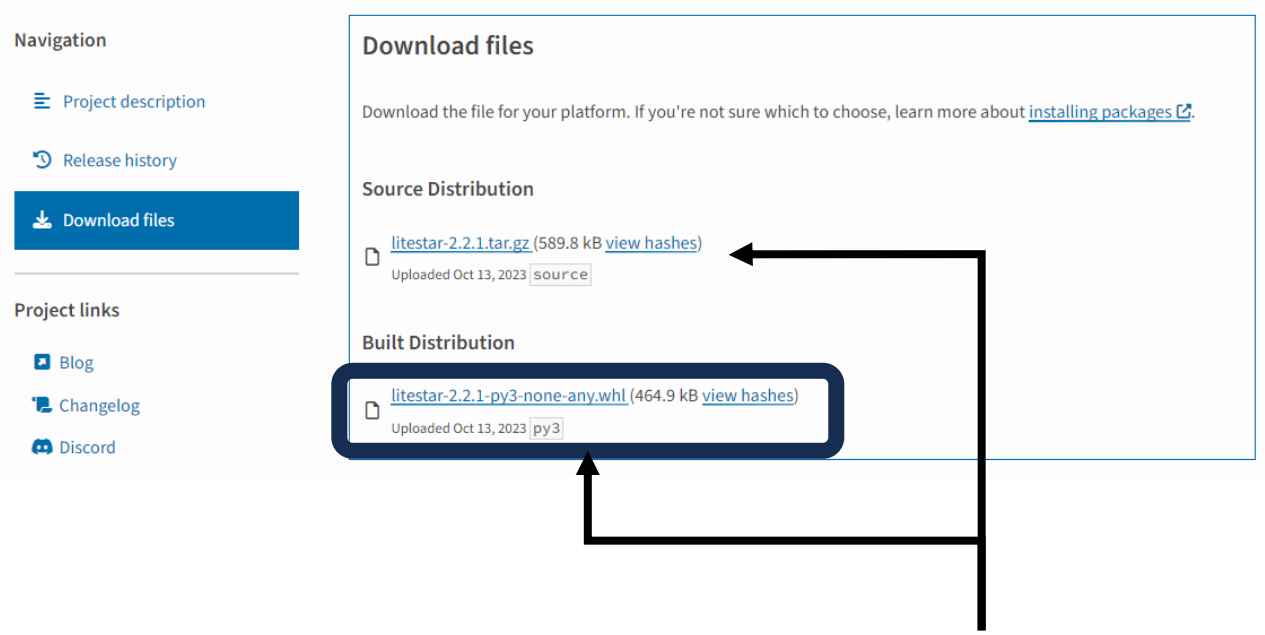


Extract to

`python-3.11.3/Lib/site-packages`

An important question is how we create wheels and source distributions

```
$ pip install litestar
Collecting litestar
 Downloading litestar-2.2.1-py3-none-
any.whl.metadata (82 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 82.7/82.7 kB 1.2 MB/s eta 0:00:00
Requirement already satisfied:
anyio>=3 in
/home/yngve/.pyenv/versions/3.12.0/lib
/python3.12/site-packages
[…]
Downloading litestar-2.2.1-py3-none-
any.whl (474 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 474.8/474.8 kB 4.2 MB/s eta 0:00:00
Installing collected packages:
litestar
Successfully installed litestar-2.2.1
```

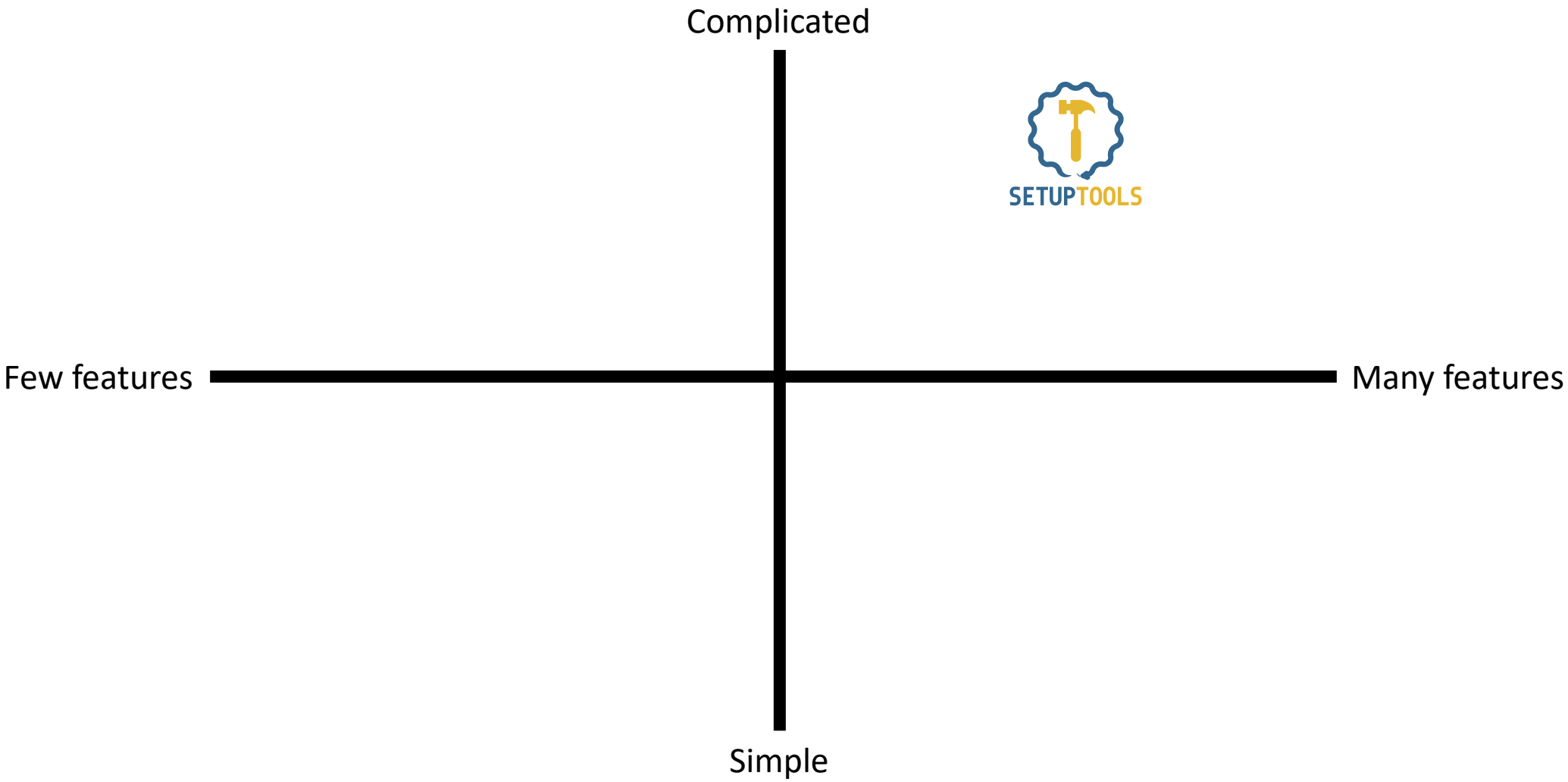


How do we make these?

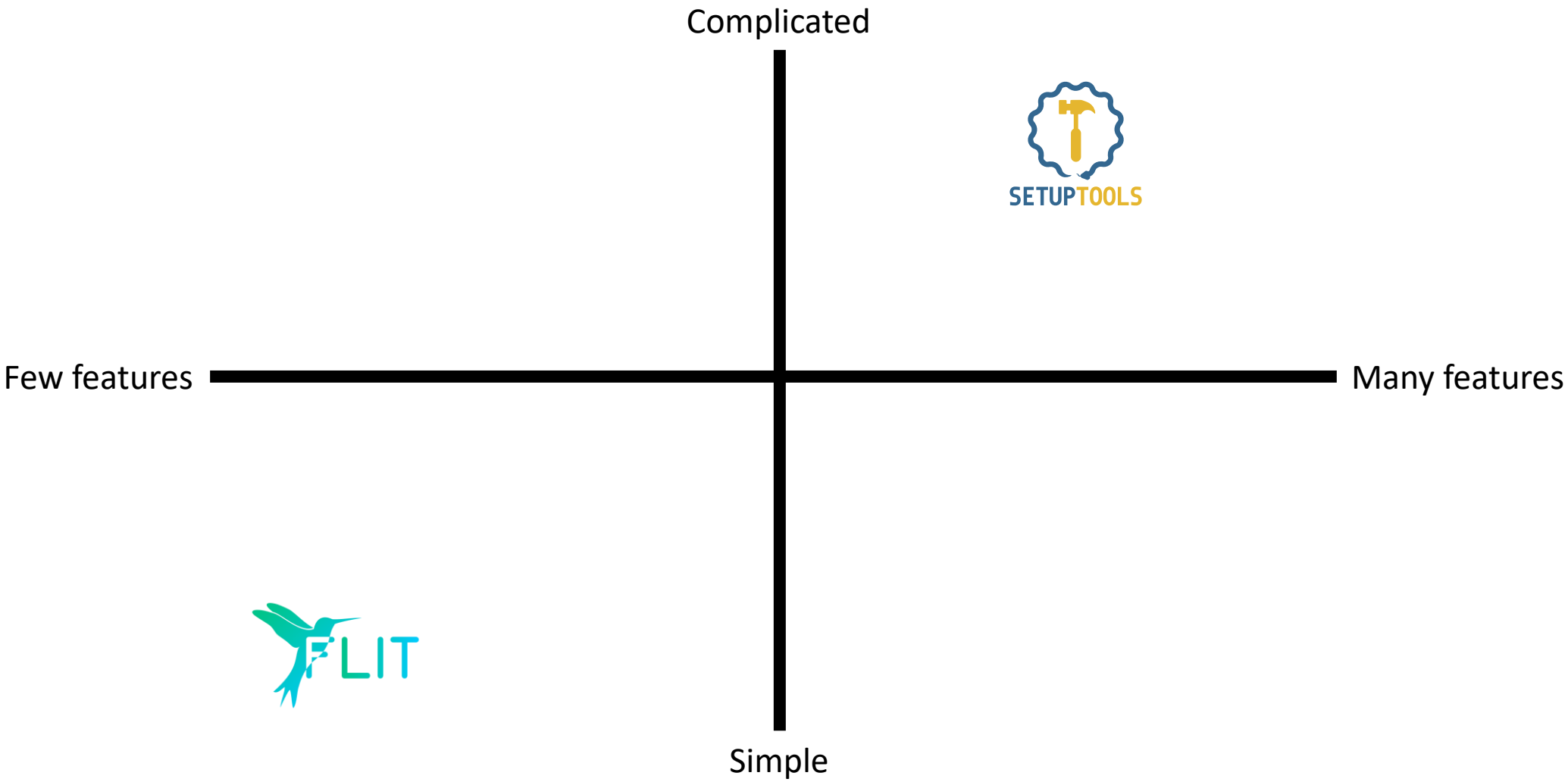
Extract to

`python-3.11.3/Lib/site-packages`

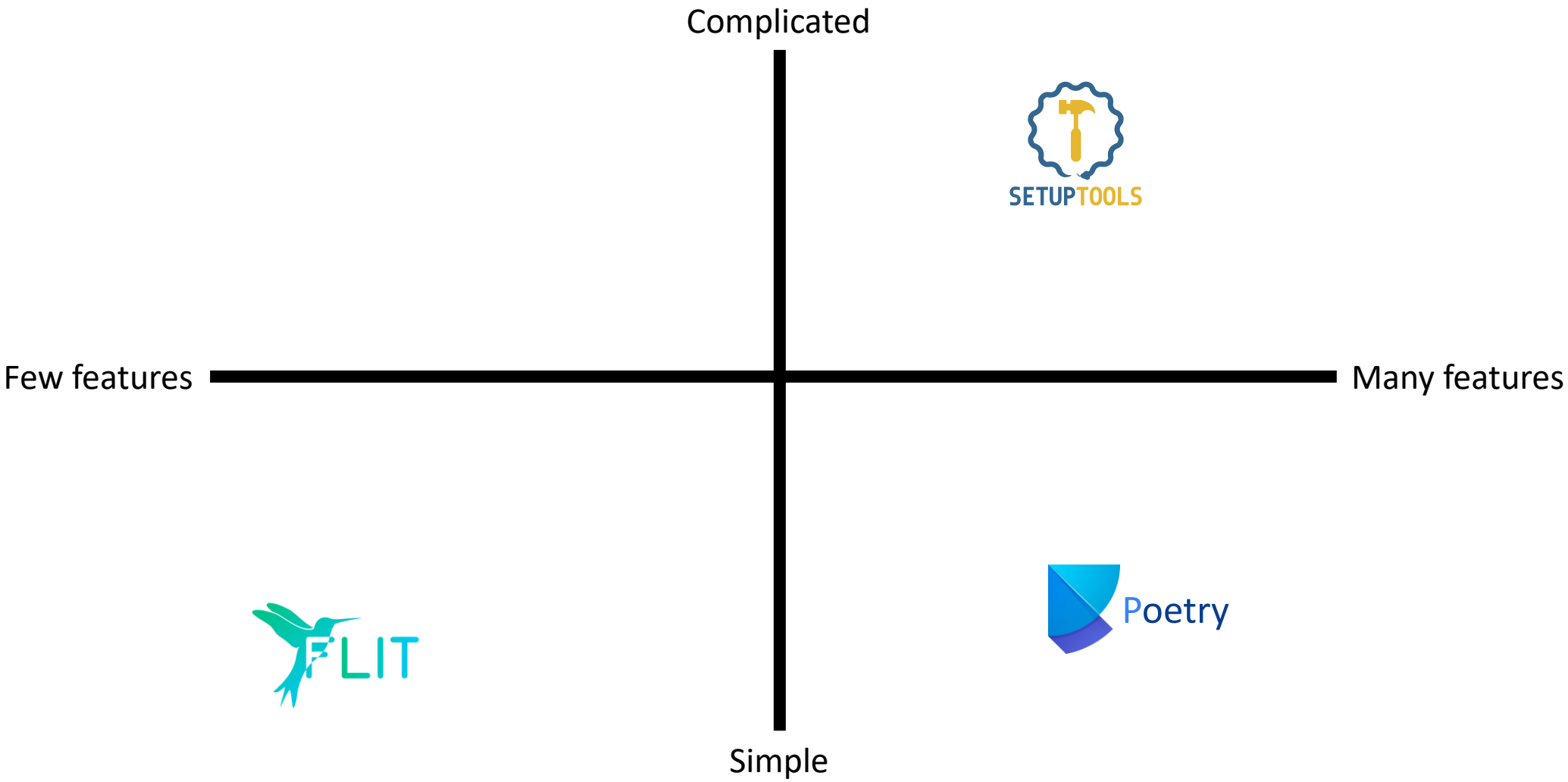
Setuptools is an old and powerful packaging tool



Flit makes it very easy to package and distribute *pure python packages*



Poetry is an easy-to-use project manager and packaging tool with many useful features



PEP 621 specifies how project metadata should be specified in a pyproject.toml file

```
[project]
name = "spam"
version = "2020.0.0"
description = "Lovely Spam! Wonderful Spam!"
readme = "README.rst"
requires-python = ">=3.8"
authors = [
  {name = "Pradyun Gedam", email =
"pradyun@example.com"},
]
dependencies = [
  "httpx",
  "django>2.1; os_name != 'nt'",
  "django>2.0; os_name == 'nt'",
]
...
```

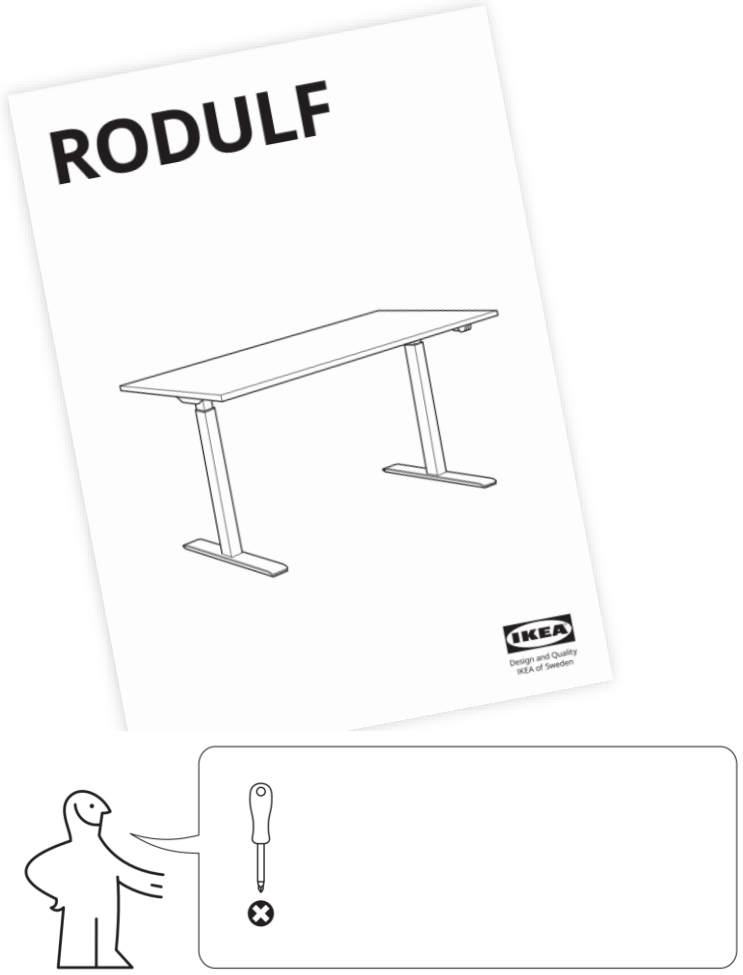
PEP 517 and 518 specifies a package should be built

```
[build-system]
# Defined by PEP 518:
requires = ["flit_core >=3.2,<4“]

# Defined by PEP-517:
build-backend =
"flit_core.buildapi"
```

We need flit to build this project

We use this entry-point to build the package

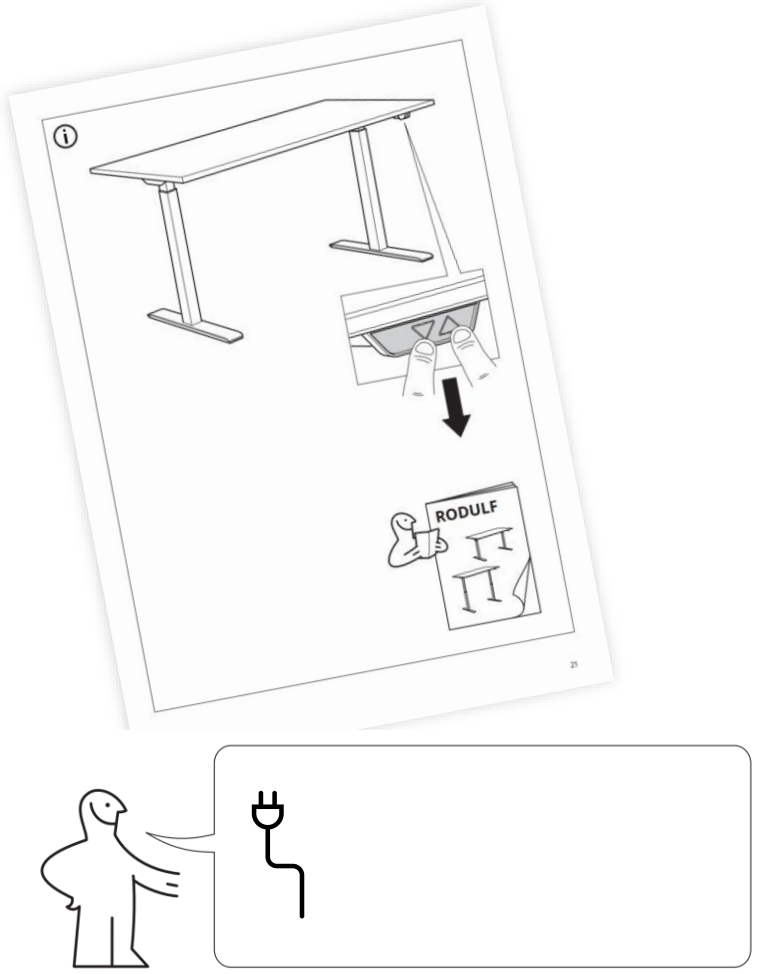
The build specification is how Python makes your library installable



Build: PEP 517 & 518

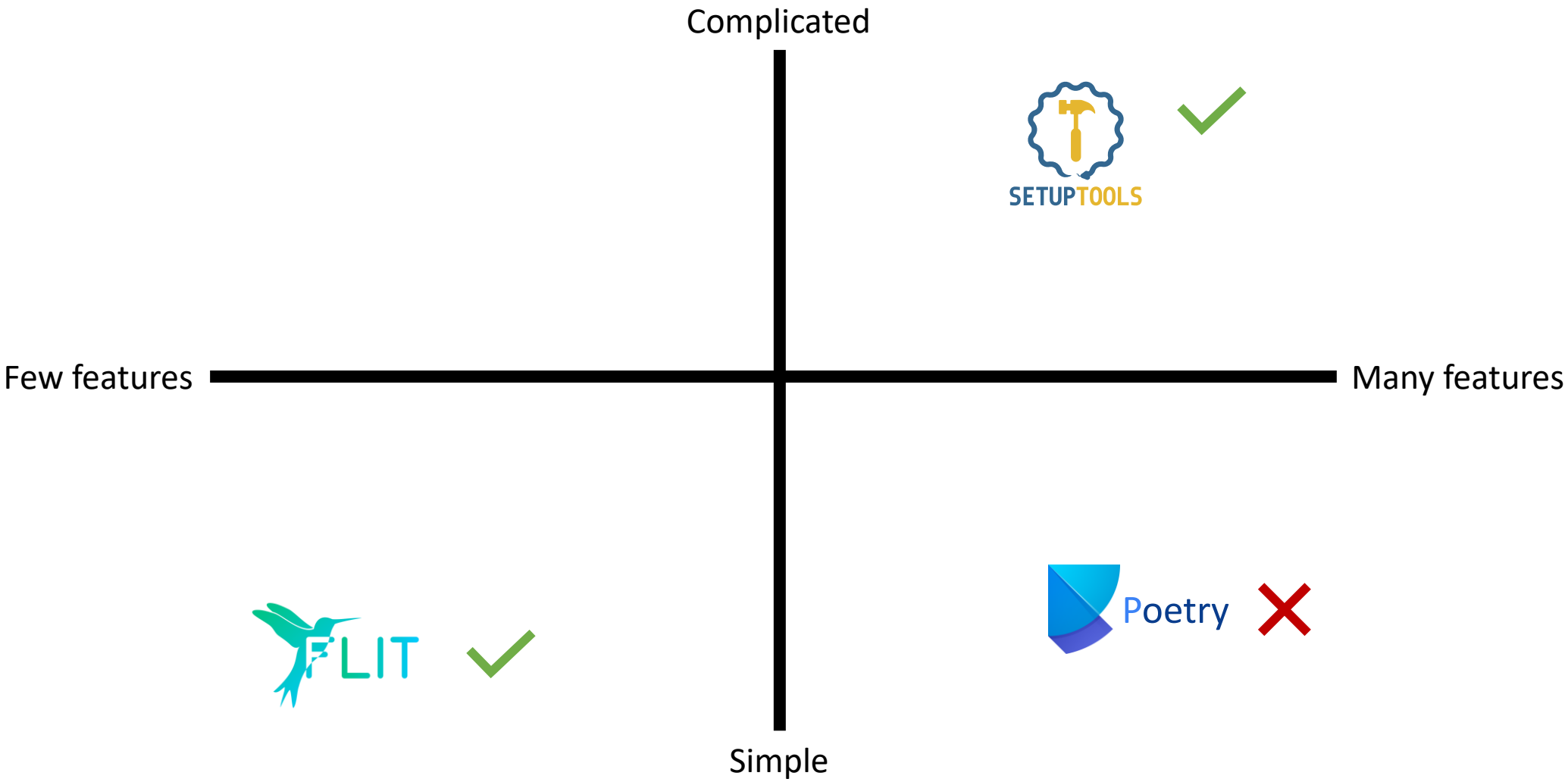
The project metadata contains information about the project itself

Build: PEP 517 & 518

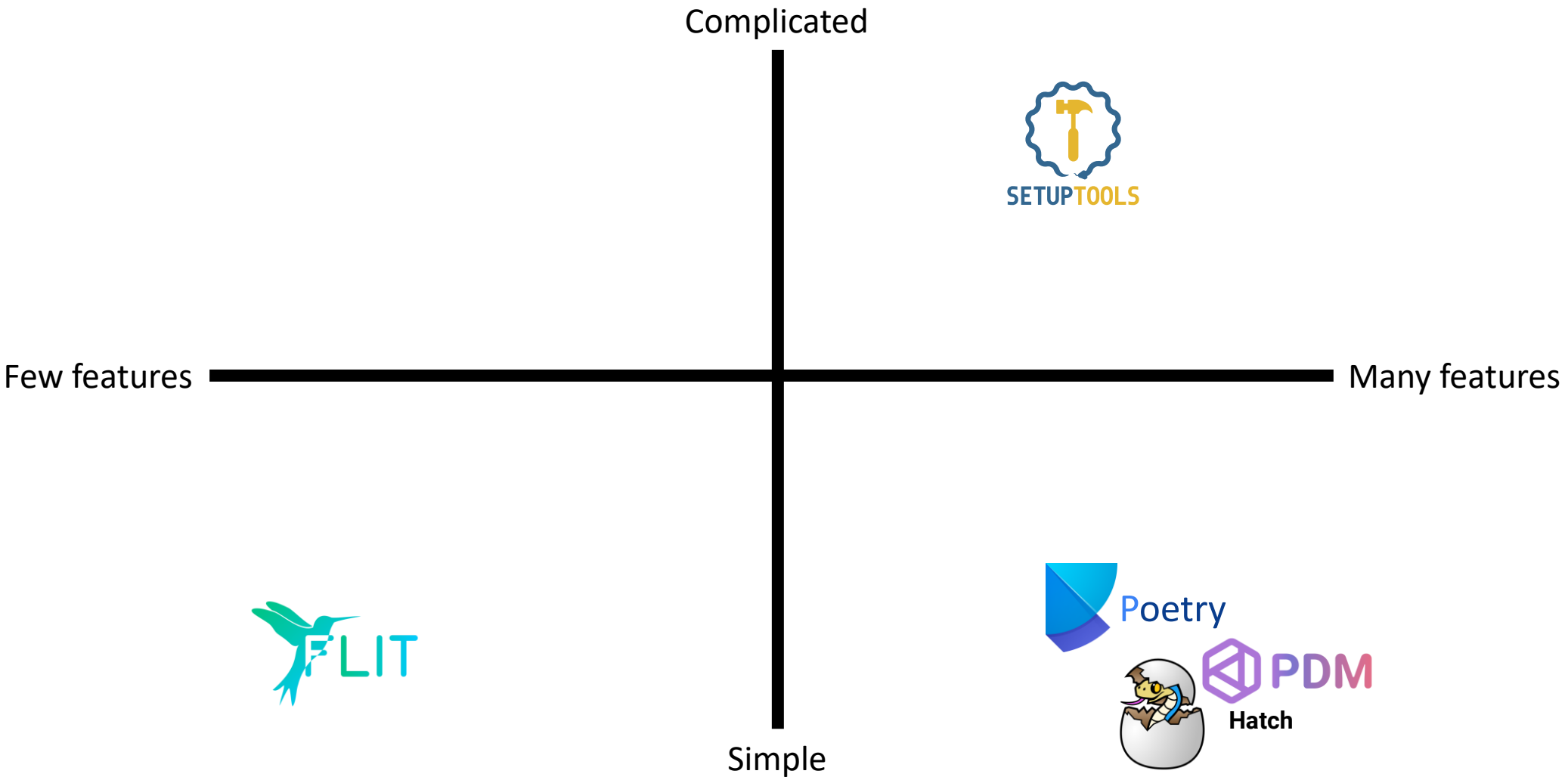


Run: PEP 621

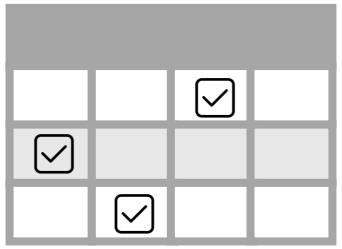
Most packaging tools have adopted these standards



After the standard, we also got new packaging tools that adhere to it



Introduction and brief overview of packaging



Comparison of packaging tools

“Live” demo

All tools except poetry follows PEP 621

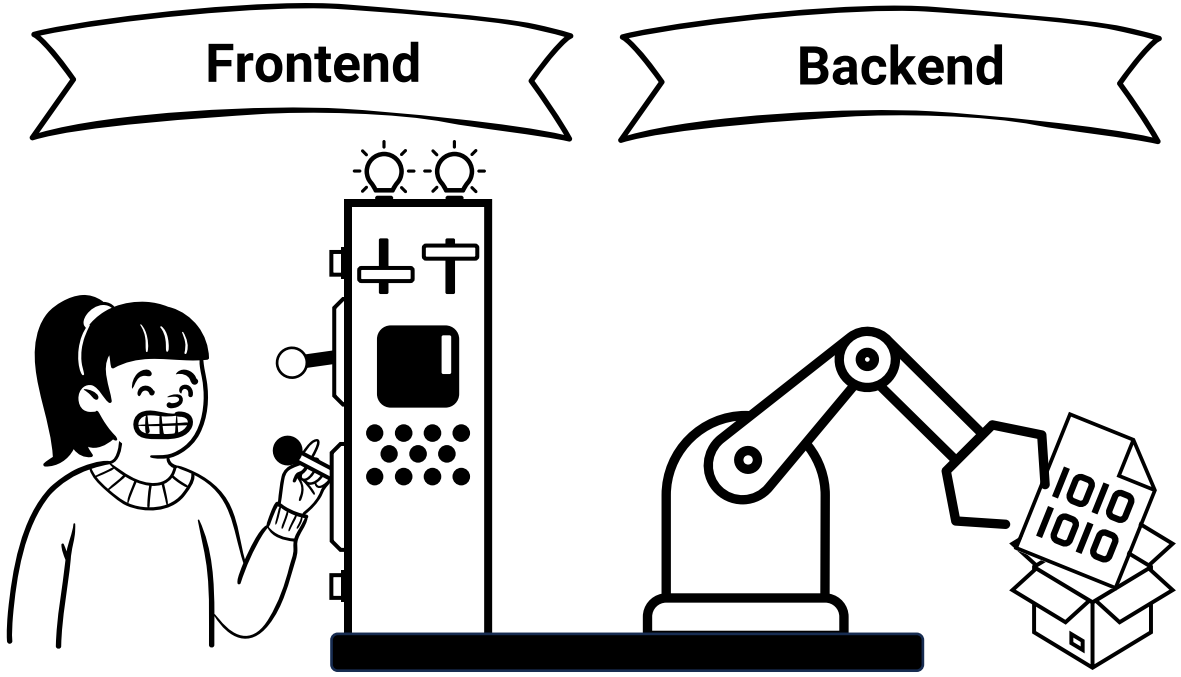
Feature	 Hatch	PDM	 Poetry	 Flit	 Setuptools
Follows PEP 621	☑	☑	✗	☑	☑

All tools except Flit support building C-extensions

Feature	Hatch	PDM	Poetry	Flit	Setuptools
Follows PEP 621	☑	☑	✗	☑	☑
Supports C-extensions	☑ (through plug-ins)	☑	☑ (undocumented)	✗	☑

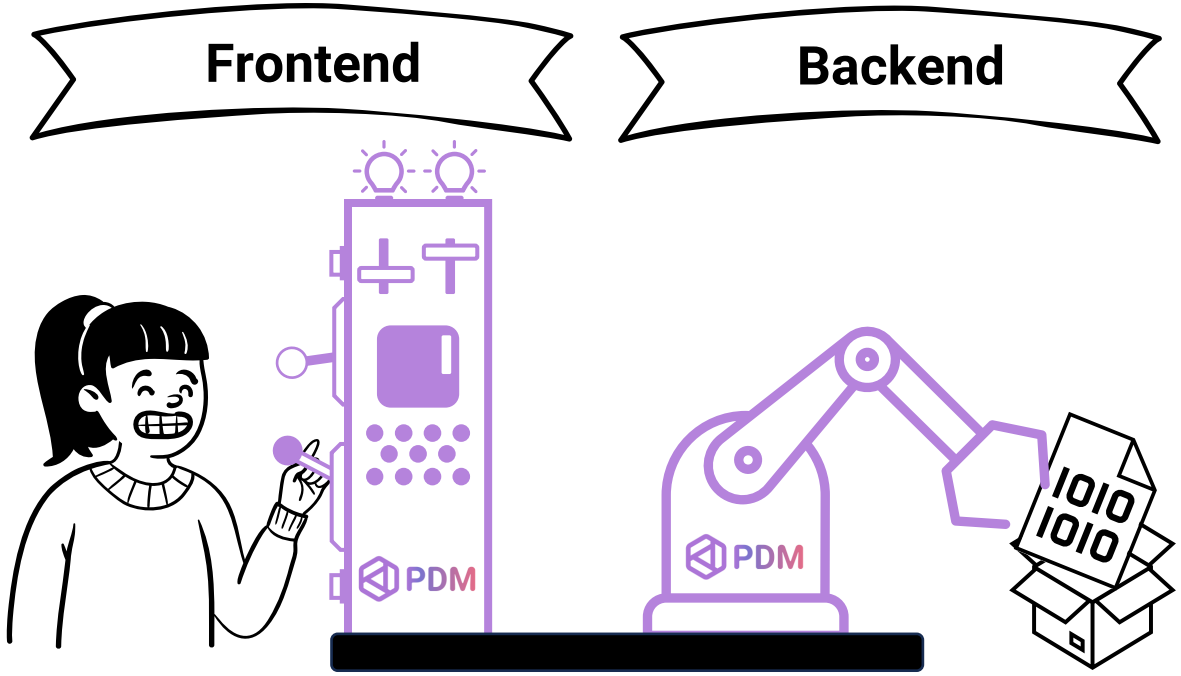
PDM is the only tool that supports other build backends

Feature	Hatch	PDM	Poetry	Flit	Setuptools
Follows PEP 621	☑	☑	✗	☑	☑
Supports C-extensions	☑ (through plug-ins)	☑	☑ (undocumented)	✗	☑
Use other build-backends	✗	☑	✗	✗	✗



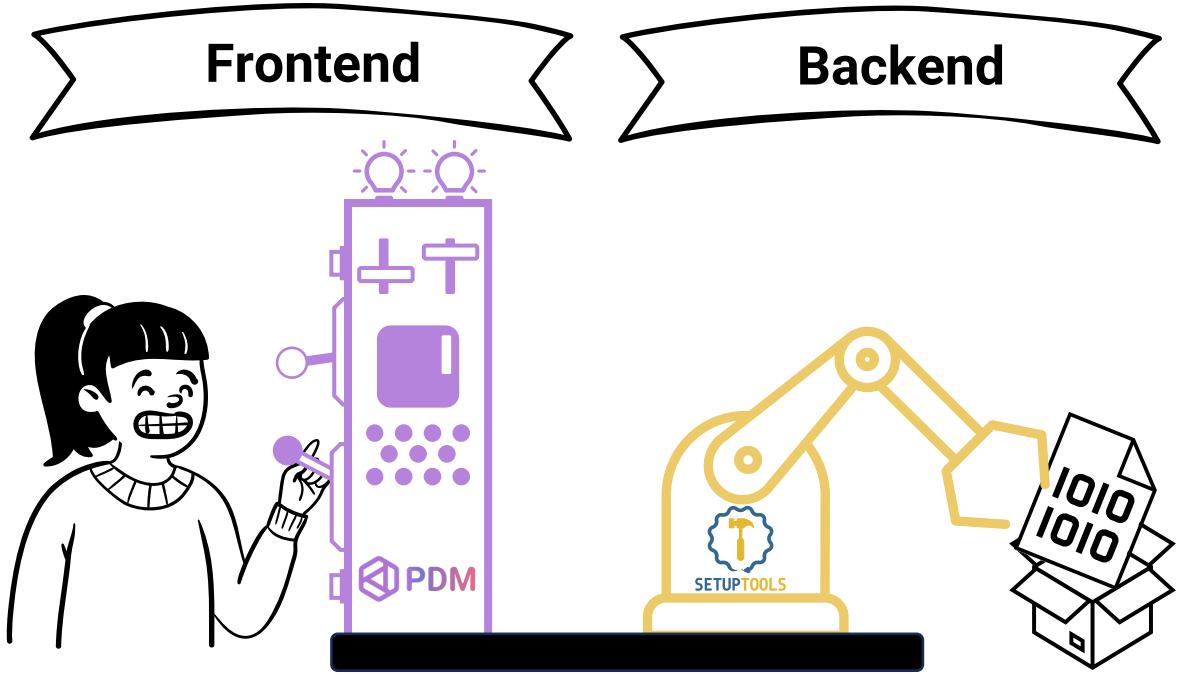
PDM is the only tool that supports other build backends

Feature	Hatch	PDM	Poetry	Flit	Setuptools
Follows PEP 621	✓	✓	✗	✓	✓
Supports C-extensions	✓ (through plug-ins)	✓	✓ (undocumented)	✗	✓
Use other build-backends	✗	✓	✗	✗	✗



PDM is the only tool that supports other build backends

Feature	Hatch	PDM	Poetry	Flit	Setuptools
Follows PEP 621	✓	✓	✗	✓	✓
Supports C-extensions	✓ (through plug-ins)	✓	✓ (undocumented)	✗	✓
Use other build-backends	✗	✓	✗	✗	✗



PDM and poetry both support managing your dependencies

Feature	 Hatch	PDM	Poetry	Flit	Setuptools
Follows PEP 621	☑	☑	✗	☑	☑
Supports C-extensions	☑ (through plug-ins)	☑	☑ (undocumented)	✗	☑
Use other build-backends	✗	☑	✗	✗	✗
Dependency management	✗	☑	☑	✗	✗

Hatch excels at environment management

Feature	Hatch	PDM	Poetry	Flit	Setuptools
Follows PEP 621	☑	☑	✗	☑	☑
Supports C-extensions	☑ (through plug-ins)	☑	☑ (undocumented)	✗	☑
Use other build-backends	✗	☑	✗	✗	✗
Dependency management	✗	☑	☑	✗	✗
Environment management	☑ 🎉	☑	☑	✗	✗

Hatch excels at environment management

Feature	Hatch	PDM	Poetry	Flit	Setuptools
Follows PEP 621	✓	✓	✗	✓	✓
Supports C-extensions	✓ (through plug-ins)	✓	✓ (undocumented)	✗	✓
Use other build-backends	✗	✓	✗	✗	✗
Dependency management	✗	✓	✓	✗	✗
Environment management	✓	✓	✓	✗	✗

```
[tool.hatch.envs.test.overrides]
matrix.auth.features = [
  { value = "oauth", if = ["oauth2"] },
  { value = "kerberos", if = ["krb5"] },
]

[[tool.hatch.envs.test.matrix]]
python = ["2.7", "3.8"]
auth = ["oauth2", "krb5", "noauth"]
```

Poetry and Setuptools are the only projects with more than one developer

Feature	Hatch	PDM	Poetry	Flit	Setuptools
Follows PEP 621	✓	✓	✗	✓	✓
Supports C-extensions	✓ (through plug-ins)	✓	✓ (undocumented)	✗	✓
Use other build-backends	✗	✓	✗	✗	✗
Dependency management	✗	✓	✓	✗	✗
Environment management	✓	✓	✓	✗	✗
Multiple developers	✗	✗	✓	✗	✓

Feature	Hatch	PDM	Poetry	Flit	Setuptools
Follows PEP 621	☑	☑	✗	☑	☑
Supports C-extensions	☑ (through plug-ins)	☑	☑ (undocumented)	✗	☑
Use other build-backends	✗	☑	✗	✗	✗
Dependency management	✗	☑	☑	✗	✗
Environment management	☑ 🎉	☑	☑	✗	✗
Multiple developers	✗	✗	☑	✗	☑

Any of these tools will probably serve your packaging needs

Feature	Hatch	PDM	Poetry	Flit	Setuptools
Follows PEP 621	☑	☑	✗	☑	☑
Supports C-extensions	☑ (through plug-ins)	☑	☑ (undocumented)	✗	☑
Use other build-backends	✗	☑	✗	✗	✗
Dependency management	✗	☑	☑	✗	✗
Environment management	☑ 🎉	☑	☑	✗	✗
Multiple developers	✗	✗	☑	✗	☑

Any of these tools will probably serve your packaging needs

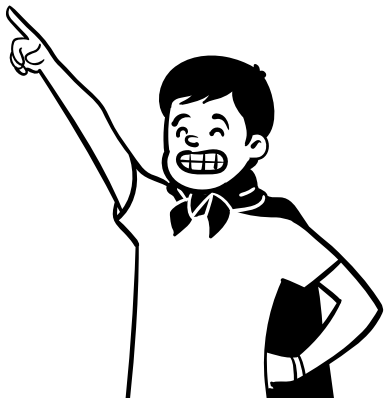
Feature	Hatch	PDM	Poetry	Flit	Setuptools
Follows PEP 621	☑	☑	✗	☑	☑
Supports C-extensions	☑ (through plug-ins)	☑	☑ (undocumented)	✗	☑
Use other build-backends	✗	☑	✗	✗	✗
Dependency management	✗	☑	☑	✗	✗
Environment management	☑ 🎉	☑	☑	✗	✗
Multiple developers	✗	✗	☑	✗	☑

Especially these tools

My favourite is PDM

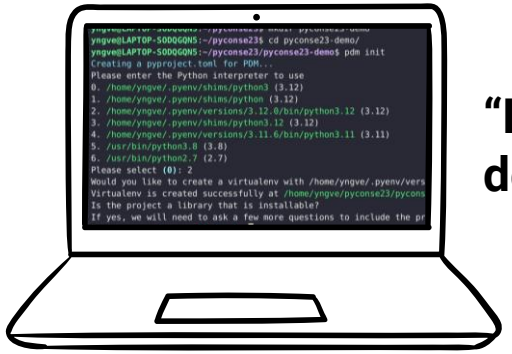
Feature	Hatch	PDM	Poetry	Flit	Setuptools
Follows PEP 621	☑	☑	✗	☑	☑
Supports C-extensions	☑ (through plug-ins)	☑	☑ (undocumented)	✗	☑
Use other build-backends	✗	☑	✗	✗	✗
Dependency management	✗	☑	☑	✗	✗
Environment management	☑ 🎉	☑	☑	✗	✗
Multiple developers	✗	✗	☑	✗	☑

Especially these tools



Introduction and brief overview of packaging

Comparison of packaging tools



“Live” demo

```
yngve@LAPTOP-SODQGQN5:~/pyconse23$ mkdir pyconse23-demo
yngve@LAPTOP-SODQGQN5:~/pyconse23$ cd pyconse23-demo/
yngve@LAPTOP-SODQGQN5:~/pyconse23/pyconse23-demo$
```

```
2. /home/yngve/.pyenv/versions/3.12.0/bin/python3.12 (3.12)
3. /home/yngve/.pyenv/shims/python3.12 (3.12)
4. /home/yngve/.pyenv/versions/3.11.6/bin/python3.11 (3.11)
5. /usr/bin/python3.8 (3.8)
6. /usr/bin/python2.7 (2.7)
Please select (0): 2
Would you like to create a virtualenv with /home/yngve/.pyenv/versions/3.12.0/bin/python3.12? [y/n] (y):
Virtualenv is created successfully at /home/yngve/pyconse23/pyconse23-demo/.venv
Is the project a library that is installable?
If yes, we will need to ask a few more questions to include the project name and build backend [y/n] (n): y
Project name (pyconse23-demo):
Project version (0.1.0):
Project description (): Sample project to download the weather forecast for Stockholm
Which build backend to use?
0. pdm-backend
1. setuptools
2. flit-core
3. hatchling
Please select (0):
License(SPDX name) (MIT):
Author name (): Yngve Mardal Moe
Author email (): yngve.moe@bouvet.no
Python requires('*' to allow any) (>=3.12): >=3.11
Project is initialized successfully
yngve@LAPTOP-SODQGQN5:~/pyconse23/pyconse23-demo$
```

```
Project version (0.1.0):
Project description (): Sample project to download the weather forecast for Stockholm
Which build backend to use?
0. pdm-backend
1. setuptools
2. flit-core
3. hatchling
Please select (0):
License(SPDX name) (MIT):
Author name (): Yngve Mardal Moe
Author email (): yngve.moe@bouvet.no
Python requires('*' to allow any) (>=3.12): >=3.11
Project is initialized successfully
yngve@LAPTOP-SODQGQN5:~/pyconse23/pyconse23-demo$ pdm install
Lock file does not exist
Updating the lock file...
  Lock successful
Changes are written to pdm.lock.
All packages are synced to date, nothing to do.
Installing the project as an editable package...
  ✔ Install pyconse23-demo 0.1.0 successful

  All complete!

yngve@LAPTOP-SODQGQN5:~/pyconse23/pyconse23-demo$
```

```
Lock file does not exist
Updating the lock file...
  Lock successful
Changes are written to pdm.lock.
All packages are synced to date, nothing to do.
Installing the project as an editable package...
  ✔ Install pyconse23-demo 0.1.0 successful

  All complete!

yngve@LAPTOP-SODQGQN5:~/pyconse23/pyconse23-demo$ ls -la
total 48
drwxr-xr-x 7 yngve yngve 4096 Nov  2 09:23 .
drwxr-xr-x 4 yngve yngve 4096 Nov  2 09:22 ..
-rw-r--r-- 1 yngve yngve 3102 Nov  1 09:26 .gitignore
drwx------ 3 yngve yngve 4096 Nov  2 09:23 .pdm-build
-rw-r--r-- 1 yngve yngve   53 Nov  2 09:22 .pdm-python
drwxr-xr-x 4 yngve yngve 4096 Nov  2 09:22 .venv
-rw-r--r-- 1 yngve yngve   17 Nov  2 09:23 README.md
drwxr-xr-x 2 yngve yngve 4096 Nov  2 09:23 __pycache__
-rw-r--r-- 1 yngve yngve  248 Nov  2 09:23 pdm.lock
-rw-r--r-- 1 yngve yngve  372 Nov  2 09:23 pyproject.toml
drwxr-xr-x 3 yngve yngve 4096 Nov  2 09:23 src
drwxr-xr-x 3 yngve yngve 4096 Nov  2 09:23 tests
yngve@LAPTOP-SODQGQN5:~/pyconse23/pyconse23-demo$
```

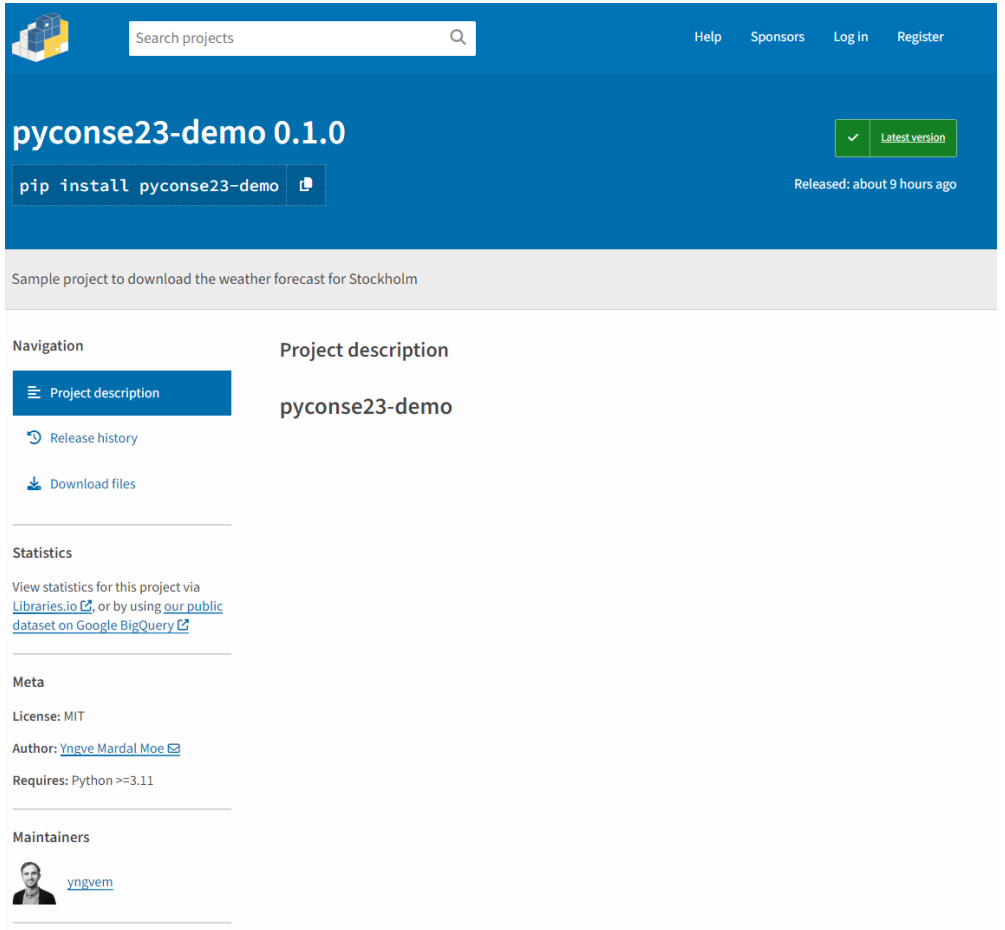
```
Updating the lock file...
  Lock successful
Changes are written to pdm.lock.
All packages are synced to date, nothing to do.
Installing the project as an editable package...
  ✔ Install pyconse23-demo 0.1.0 successful

  All complete!

yngve@LAPTOP-SODQGQN5:~/pyconse23/pyconse23-demo$ ls -la
total 48
drwxr-xr-x 7 yngve yngve 4096 Nov  2 09:23 .
drwxr-xr-x 4 yngve yngve 4096 Nov  2 09:22 ..
-rw-r--r-- 1 yngve yngve 3102 Nov  1 09:26 .gitignore
drwx------ 3 yngve yngve 4096 Nov  2 09:23 .pdm-build
-rw-r--r-- 1 yngve yngve   53 Nov  2 09:22 .pdm-python
drwxr-xr-x 4 yngve yngve 4096 Nov  2 09:22 .venv
-rw-r--r-- 1 yngve yngve   17 Nov  2 09:23 README.md
drwxr-xr-x 2 yngve yngve 4096 Nov  2 09:23 __pycache__
-rw-r--r-- 1 yngve yngve  248 Nov  2 09:23 pdm.lock
-rw-r--r-- 1 yngve yngve  372 Nov  2 09:23 pyproject.toml
drwxr-xr-x 3 yngve yngve 4096 Nov  2 09:23 src
drwxr-xr-x 3 yngve yngve 4096 Nov  2 09:23 tests
yngve@LAPTOP-SODQGQN5:~/pyconse23/pyconse23-demo$ vim src/pyconse23_demo/__init__.py
yngve@LAPTOP-SODQGQN5:~/pyconse23/pyconse23-demo$
```

```
-rw-r--r-- 1 yngve yngve  248 Nov  2 09:23 pdm.lock
-rw-r--r-- 1 yngve yngve  372 Nov  2 09:23 pyproject.toml
drwxr-xr-x 3 yngve yngve 4096 Nov  2 09:23 src
drwxr-xr-x 3 yngve yngve 4096 Nov  2 09:23 tests
yngve@LAPTOP-SODQGQN5:~/pyconse23/pyconse23-demo$ vim src/pyconse23_demo/__init__.py
yngve@LAPTOP-SODQGQN5:~/pyconse23/pyconse23-demo$ pdm add httpx plotille
Adding packages to default dependencies: httpx, plotille
  Lock successful
Changes are written to pyproject.toml.
Synchronizing working set with resolved packages: 8 to add, 0 to update, 0 to remove

  ✔ Install sniffio 1.3.0 successful
  ✔ Install certifi 2023.7.22 successful
  ✔ Install idna 3.4 successful
  ✔ Install plotille 5.0.0 successful
  ✔ Install h11 0.14.0 successful
  ✔ Install httpx 0.25.0 successful
  ✔ Install httpcore 0.18.0 successful
  ✔ Install anyio 4.0.0 successful
Installing the project as an editable package...
  ✔ Update pyconse23-demo 0.1.0+editable -> 0.1.0 successful

  All complete!

yngve@LAPTOP-SODQGQN5:~/pyconse23/pyconse23-demo$
```

```
  ✔ Install anyio 4.0.0 successful
  ✔ Install httpx 0.25.0 successful
  ✔ Install httpcore 0.18.0 successful
Installing the project as an editable package...
  ✔ Update pyconse23-demo 0.1.0+editable -> 0.1.0 successful

  All complete!

yngve@LAPTOP-SODQGQN5:~/pyconse23/pyconse23-demo$ pdm run python src/pyconse23_demo/__init__.py
                               Temperature forecast in Stockholm [°C]
(Air temp) ^
11.2200000 |
10.1160000 |
9.01200000 |
7.90800000 |
6.80400000 |
5.70000000 |
4.59600000 |                                                                            ......
3.49200000 |                                                  ...........................
2.38800000 |                                  ................
1.28400000 |       ............................
0.18000000 | ......
-----------|-|---------|---------|---------|---------|---------|---------|---------|---------|-> (Time)
           | 07:00:00  08:00:00  09:00:00  10:00:00  11:00:00  12:00:00  13:00:00  14:00:00  15:00:00
yngve@LAPTOP-SODQGQN5:~/pyconse23/pyconse23-demo$
```

```
  All complete!

yngve@LAPTOP-SODQGQN5:~/pyconse23/pyconse23-demo$ pdm run python src/pyconse23_demo/__init__.py
                              Temperature forecast in Stockholm [°C]
(Air temp) ^
11.2200000 |
10.1160000 |
9.01200000 |
7.90800000 |
6.80400000 |
5.70000000 |
4.59600000 |
3.49200000 |
2.38800000 |
1.28400000 |
0.18000000 |
-----------|-|---------|---------|---------|---------|---------|---------|---------|---------|---------|-> (Time)
           | 07:00:00  08:00:00  09:00:00  10:00:00  11:00:00  12:00:00  13:00:00  14:00:00  15:00:00
yngve@LAPTOP-SODQGQN5:~/pyconse23/pyconse23-demo$ pdm publish
Building sdist...
Built sdist at /home/yngve/pyconse23/pyconse23-demo/dist/pyconse23_demo-0.1.0.tar.gz
Building wheel...
Built wheel at /home/yngve/pyconse23/pyconse23-demo/dist/pyconse23_demo-0.1.0-py3-none-any.whl
```

You can now pip install `pyconse23-demo` if you want to get the weather forecast for Stockholm!



These resources are great if you want to learn more about packaging

- **pyOpenSci Python Open Source Package Development Guide:** https://www.pyopensci.org/python-package-guide/
- **Python Packaging User Guide:** https://packaging.python.org/en/latest/
- **Declaring project metadata:** https://packaging.python.org/en/latest/specifications/declaring-project-metadata/
- **Why SciPy builds for Python 3.12 on Windows are a minor miracle:** https://labs.quansight.org/blog/building-scipy-with-flang
- **Various PEPs (somewhat in order of relevance):**
  - **How to store metadata in pyproject.toml files:** PEP 621
  - **Dependency specification:** PEP 508
  - **Build backend metadata and interfaces:** PEP 517 & 518
  - **The wheel standard:** PEP 427
  - **How metadata is stored in installed projects (in PKG-INFO files):** PEP 345, 566 & 643

These resources are great if you want to learn more about packaging

- **pyOpenSci Python Open Source Package Development Guide:** https://www.pyopensci.org/python-package-guide/
- **Python Packaging User Guide:** https://packaging.python.org/en/latest/
- **Declaring project metadata:** https://packaging.python.org/en/latest/specifications/declaring-project-metadata/
- **Why SciPy builds for Python 3.12 on Windows are a minor miracle:** https://labs.quansight.org/blog/building-scipy-with-flang
- **Various PEPs (somewhat in order of relevance):**
  - **How to store metadata in pyproject.toml files:** PEP 621
  - **Dependency specification:** PEP 508
  - **Build backend metadata and interfaces:** PEP 517 & 518
  - **The wheel standard:** PEP 427
  - **How metadata is stored in installed projects (in PKG-INFO files):** PEP 345, 566 & 643

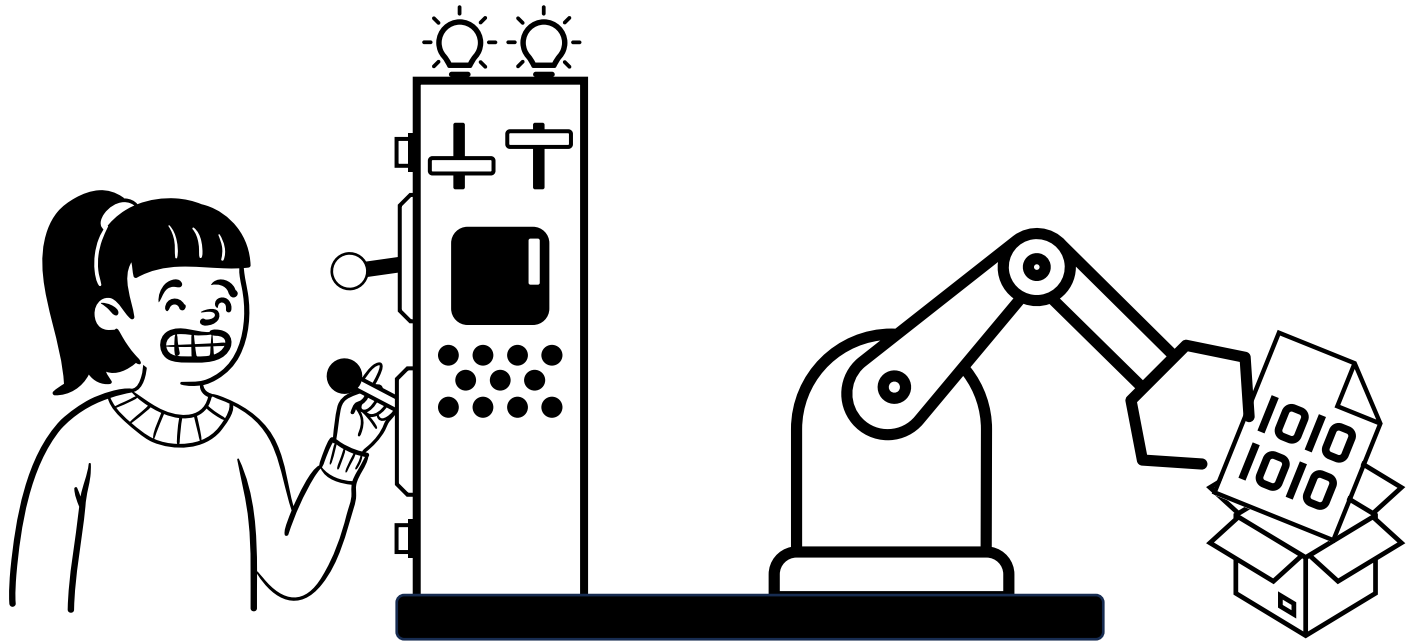
bouvet

These resources are great if you want to learn more about packaging

- **pyOpenSci Python Open Source Package Development Guide:** https://www.pyopensci.org/python-package-guide/
- **Python Packaging User Guide:** https://packaging.python.org/en/latest/
- **Declaring project metadata:** https://packaging.python.org/en/latest/specifications/declaring-project-metadata/
- **Why SciPy builds for Python 3.12 on Windows are a minor miracle:** https://labs.quansight.org/blog/building-scipy-with-flang
- **Various PEPs (somewhat in order of relevance):**
  - **How to store metadata in pyproject.toml files:** PEP 621
  - **Dependency specification:** PEP 508
  - **Build backend metadata and interfaces:** PEP 517 & 518
  - **The wheel standard:** PEP 427
  - **How metadata is stored in installed projects (in PKG-INFO files):** PEP 345, 566 & 643

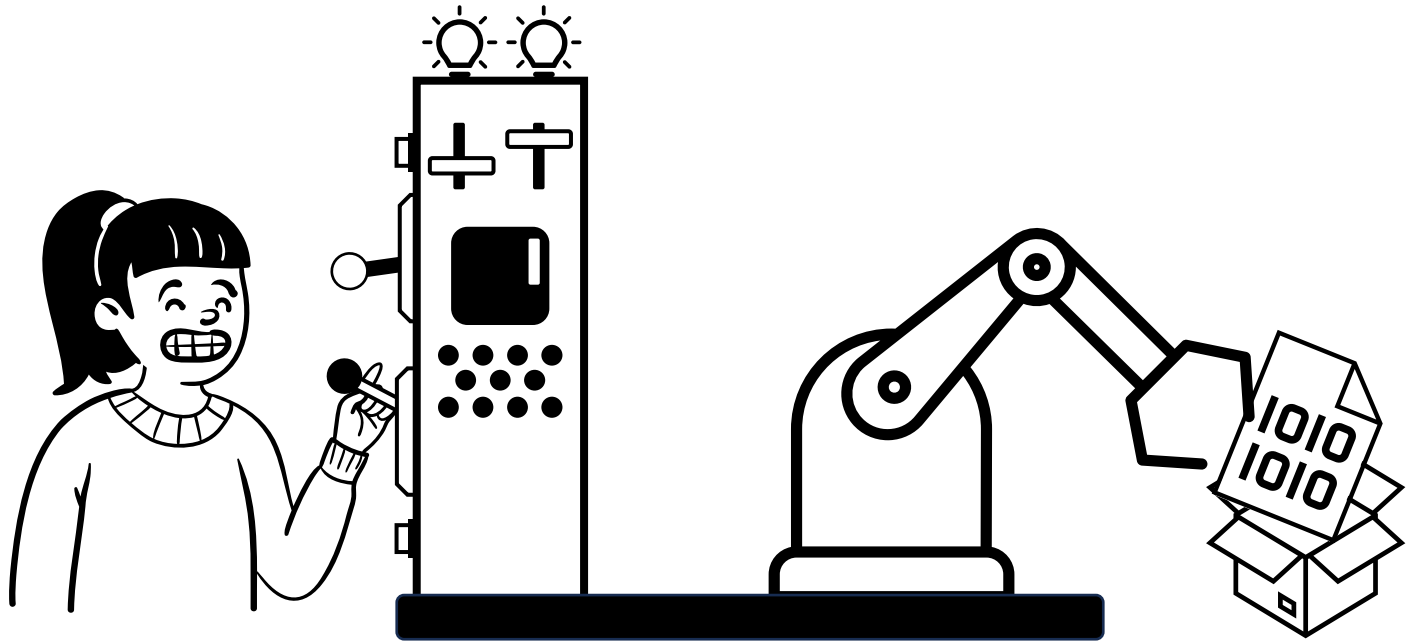
bouvet

To summarise, packaging is the task of preparing your code to be easily used by others, and PDM makes packaging easy



bouvet

To summarise, packaging is the task of preparing your code to be easily used by others, and PDM makes packaging easy



bouvet

Questions?