

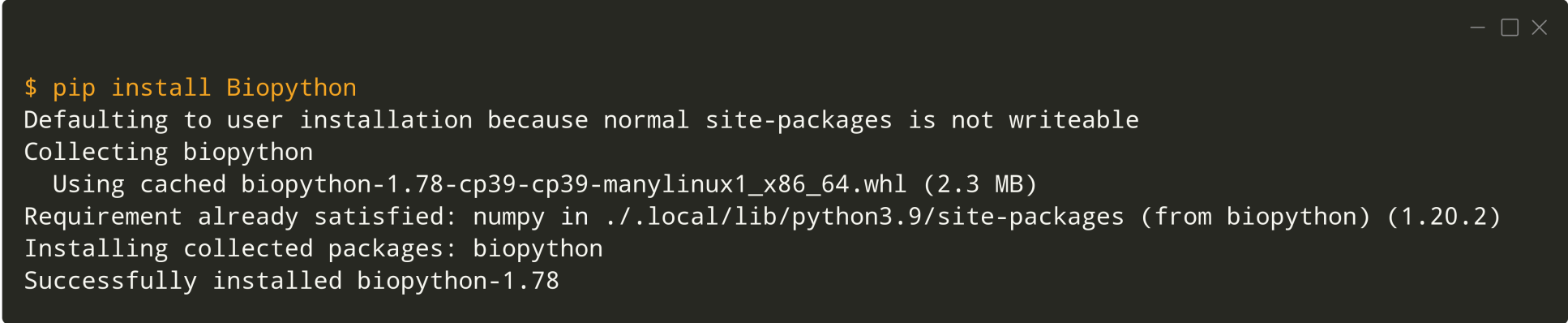
Python Packaging and PyPI

Martin Larralde

Python User Group

25-05-21

How do you install something in Python ?



```
$ pip install Biopython
```

```
Defaulting to user installation because normal site-packages is not writeable
```

```
Collecting biopython
```

```
Using cached biopython-1.78-cp39-cp39-manylinux1_x86_64.whl (2.3 MB)
```

```
Requirement already satisfied: numpy in ./local/lib/python3.9/site-packages (from biopython) (1.20.2)
```

```
Installing collected packages: biopython
```

```
Successfully installed biopython-1.78
```

Overview

- What happens when you install a package?
- How are Python packages distributed?
- How do you create a distribution for your package?
- How do you publish a distribution to PyPI?
- How do you expose a CLI script from your package ?
- How do you include data in your package?

The Python Package Index (PyPI)



<https://pypi.org/project/biopython/>

A screenshot of the biopython 1.78 project page on the Python Package Index (PyPI). The page has a blue header with a search bar and links for Help, Sponsors, Log in, and Register. The main content area shows the project name "biopython 1.78" with a green "Latest version" badge. Below this is a button to "pip install biopython" and a note that it was "Released: Sep 4, 2020". A description states it is "Freely available tools for computational molecular biology." The page is divided into sections: "Navigation" with links to Project description, Release history, and Download files (highlighted with an orange box and a blue arrow); "Project links" with links to Homepage, Tracker, Source, and Documentation; and "Statistics" showing GitHub statistics: Stars: 2,705, Forks: 1,296, and Open issues/PRs: 433. The "Project description" section features a banner with a blue and yellow DNA double helix and the word "biopython" below it. Above the banner are several status badges: pypi v1.78, conda-forge v1.78, build passing, build failing, coverage 84%, and Depsy 100th percentile.

PyPI-hosted files

Navigation

[Project description](#)

[Release history](#)

[Download files](#)

Project links

[Homepage](#)

[Tracker](#)

[Source](#)

[Documentation](#)

Statistics

GitHub statistics:

★ Stars: 2,705

🍴 Forks: 1,296

📢 Open issues/PRs: 433

View statistics for this project via [Libraries.io](#), or by using [our public dataset on Google BigQuery](#)

Download files

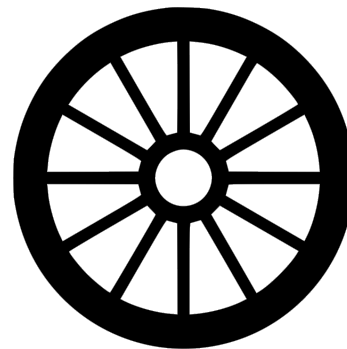
Download the file for your platform. If you're not sure which to choose, learn more about [installing packages](#).

Filename, size	File type	Python version	Upload date	Hashes
biopython-1.78-cp36-cp36m-macosx_10_9_x86_64.whl (2.2 MB)	Wheel	cp36	Sep 4, 2020	View
biopython-1.78-cp36-cp36m-manylinux1_i686.whl (2.2 MB)	Wheel	cp36	Sep 4, 2020	View
biopython-1.78-cp36-cp36m-manylinux1_x86_64.whl (2.3 MB)	Wheel	cp36	Sep 4, 2020	View
biopython-1.78-cp36-cp36m-manylinux_2_17_aarch64.manlinux2014_aarch64.whl (2.2 MB)	Wheel	cp36	May 11, 2021	View
biopython-1.78-cp36-cp36m-win32.whl (2.2 MB)	Wheel	cp36	Sep 4, 2020	View
biopython-1.78-cp36-cp36m-win_amd64.whl (2.3 MB)	Wheel	cp36	Sep 4, 2020	View
biopython-1.78-cp37-cp37m-macosx_10_9_x86_64.whl (2.2 MB)	Wheel	cp37	Sep 4, 2020	View
biopython-1.78.tar.gz (16.9 MB)	Source	None	Sep 4, 2020	View
biopython-1.78-cp37-cp37m-manylinux1_x86_64.whl (2.3 MB)	Wheel	cp37	Sep 4,	View

Distribution types



python setup.py bdist_wheel



- Source distribution
(.zip, .tar.gz)
- Contains everything
needed to build a wheel

- Binary distribution
(.whl)
- Either universal or
platform-specific

Distribution contents



- Contains source files



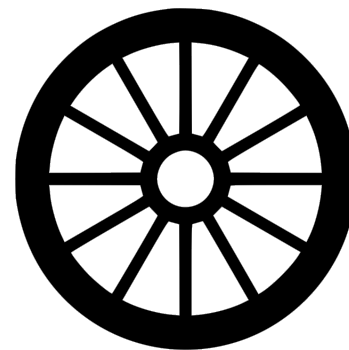
.py



.c



.pyx



- Contains compiled files



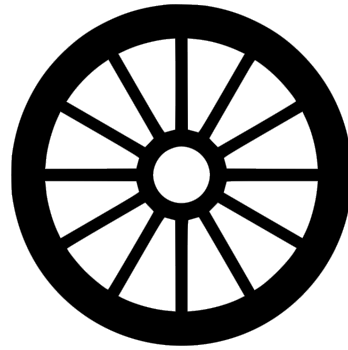
.py



.so
.dll

Universal wheels

- A wheel is universal if it does not contain platform-specific or version-specific files



.py

- In practice, universal wheels are wheels that don't contain compiled extensions (only Python files and static data files)

How do I setup my project so I can distribute it?

- What I'll be showing is a typical project setup to be installed with *setuptools* (Python standard library)
- There are alternative packagers that I won't cover (<https://python-poetry.org/>)
- The structure is very flexible but I'm going to show what I consider the *best practices*
- The project can be found on the EMBL git server (<https://git.embl.de/larralde/epug-package>)

Our Python Package



Python code

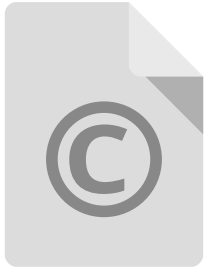
- Organise your code in modules that can be imported

```
$ tree
```

```
.  
├── epug_package  
│   ├── __init__.py  
│   ├── model.py  
│   └── utils.py
```

<https://docs.python-guide.org/writing/structure/>

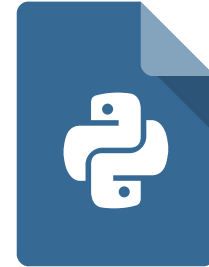
Our Python Package



LICENSE

- **Your code should have a license**, and the license file should be available somewhere

<https://choosealicense.com/>



setup.py

- The `setuptools` file, which will be used to build and package the library

Our Python Package



setup.py

Old-fashioned: all metadata in the setup.py file

```
from setuptools import setup

setup(
    name="biopython",
    version="1.78",
    author="The Biopython Contributors",
    author_email="biopython@biopython.org",
    url="https://biopython.org/",
    description="Freely available tools for computational molecular biology.",
)
```

Our Python Package



setup.py

Newer: move metadata out, keep minimal code

```
from setuptools import setup  
  
setup()
```

Our Python Package



README.md

- The README, supporting Markdown syntax to render nicely on PyPI



setup.cfg

- A config file containing the project metadata (and configuration for some tool, if desired)

The setup configuration file



setup.cfg

```
[metadata]
name = epug-package
version = attr: epug_package.__version__
author = Martin Larralde
author_email = martin.larralde@embl.de
url = https://git.embl.de/larralde/epug_package
description = A toy package to be shown at the EMBL Python User Group
long_description = file: README.md
long_description_content_type = text/markdown
license = MIT
license_file = LICENSE
platform = any
keywords = example, template, toy-package
classifiers =
    Development Status :: 3 - Alpha
    Intended Audience :: Developers
    License :: OSI Approved :: MIT License
    Operating System :: OS Independent
    Programming Language :: Python :: 3
    Topic :: Software Development :: Libraries :: Python Modules
project_urls =
    Bug Tracker = https://git.embl.de/larralde/epug-package/-/issues
```

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

The setup configuration file

```
[options]
zip_safe = false
packages = find:
include_package_data = true
python_requires = >= 3.6
setup_requires =
    setuptools >=39.2
    wheel >=0.30
install_requires =
    biopython ~=1.78
    dataclasses ~=0.8           ; python_version < '3.7'
    psutil ~=5.8               ; sys_platform == 'win32'
```

```
[bdist_wheel]
universal = true
```

The Final Project Structure

```
$ tree
```

```
.  
├── docs  
├── epug_package  
│   ├── __init__.py  
│   ├── model.py  
│   └── utils.py  
├── LICENSE  
├── README.md  
├── setup.cfg  
├── setup.py  
└── tests
```

Building a source distribution

```
$ python setup.py sdist
running sdist
running egg_info
creating epug_package.egg-info
writing epug_package.egg-info/PKG-INFO
writing dependency_links to epug_package.egg-info/dependency_links.txt
writing requirements to epug_package.egg-info/requires.txt
writing top-level names to epug_package.egg-info/top_level.txt
writing manifest file 'epug_package.egg-info/SOURCES.txt'
reading manifest file 'epug_package.egg-info/SOURCES.txt'
writing manifest file 'epug_package.egg-info/SOURCES.txt'
running check
creating epug-package-0.1.0
creating epug-package-0.1.0/epug_package
creating epug-package-0.1.0/epug_package.egg-info
copying files to epug-package-0.1.0...
copying LICENSE -> epug-package-0.1.0
copying README.md -> epug-package-0.1.0
copying setup.cfg -> epug-package-0.1.0
copying setup.py -> epug-package-0.1.0
copying epug_package/__init__.py -> epug-package-0.1.0/epug_package
copying epug_package/model.py -> epug-package-0.1.0/epug_package
copying epug_package/utils.py -> epug-package-0.1.0/epug_package
copying epug_package.egg-info/PKG-INFO -> epug-package-0.1.0/epug_package.egg-info
copying epug_package.egg-info/SOURCES.txt -> epug-package-0.1.0/epug_package.egg-info
copying epug_package.egg-info/dependency_links.txt -> epug-package-0.1.0/epug_package.egg-info
copying epug_package.egg-info/not-zip-safe -> epug-package-0.1.0/epug_package.egg-info
copying epug_package.egg-info/requires.txt -> epug-package-0.1.0/epug_package.egg-info
copying epug_package.egg-info/top_level.txt -> epug-package-0.1.0/epug_package.egg-info
Writing epug-package-0.1.0/setup.cfg
creating dist
Creating tar archive
removing 'epug-package-0.1.0' (and everything under it)
```



dist



epug-package-0.1.0.tar.gz

Building a wheel distribution

```
$ python setup.py bdist_wheel
running bdist_wheel
running build
running build_py
creating build
creating build/lib
creating build/lib/epug_package
copying epug_package/utils.py -> build/lib/epug_package
copying epug_package/__init__.py -> build/lib/epug_package
copying epug_package/model.py -> build/lib/epug_package
running egg_info
writing epug_package.egg-info/PKG-INFO
writing dependency_links to epug_package.egg-info/dependency_links.txt
writing requirements to epug_package.egg-info/requires.txt
writing top-level names to epug_package.egg-info/top_level.txt
reading manifest file 'epug_package.egg-info/SOURCES.txt'
writing manifest file 'epug_package.egg-info/SOURCES.txt'
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/epug_package
copying build/lib/epug_package/utils.py -> build/bdist.linux-x86_64/wheel/epug_package
copying build/lib/epug_package/__init__.py -> build/bdist.linux-x86_64/wheel/epug_package
copying build/lib/epug_package/model.py -> build/bdist.linux-x86_64/wheel/epug_package
running install_egg_info
Copying epug_package.egg-info to build/bdist.linux-x86_64/wheel/epug_package-0.1.0-py3.9.egg-info
running install_scripts
creating build/bdist.linux-x86_64/wheel/epug_package-0.1.0.dist-info/WHEEL
creating 'dist/epug_package-0.1.0-py3-none-any.whl' and adding 'build/bdist.linux-x86_64/wheel' to it
adding 'epug_package/__init__.py'
adding 'epug_package/model.py'
adding 'epug_package/utils.py'
adding 'epug_package-0.1.0.dist-info/LICENSE'
adding 'epug_package-0.1.0.dist-info/METADATA'
adding 'epug_package-0.1.0.dist-info/WHEEL'
adding 'epug_package-0.1.0.dist-info/top_level.txt'
adding 'epug_package-0.1.0.dist-info/RECORD'
removing build/bdist.linux-x86_64/wheel
```



dist



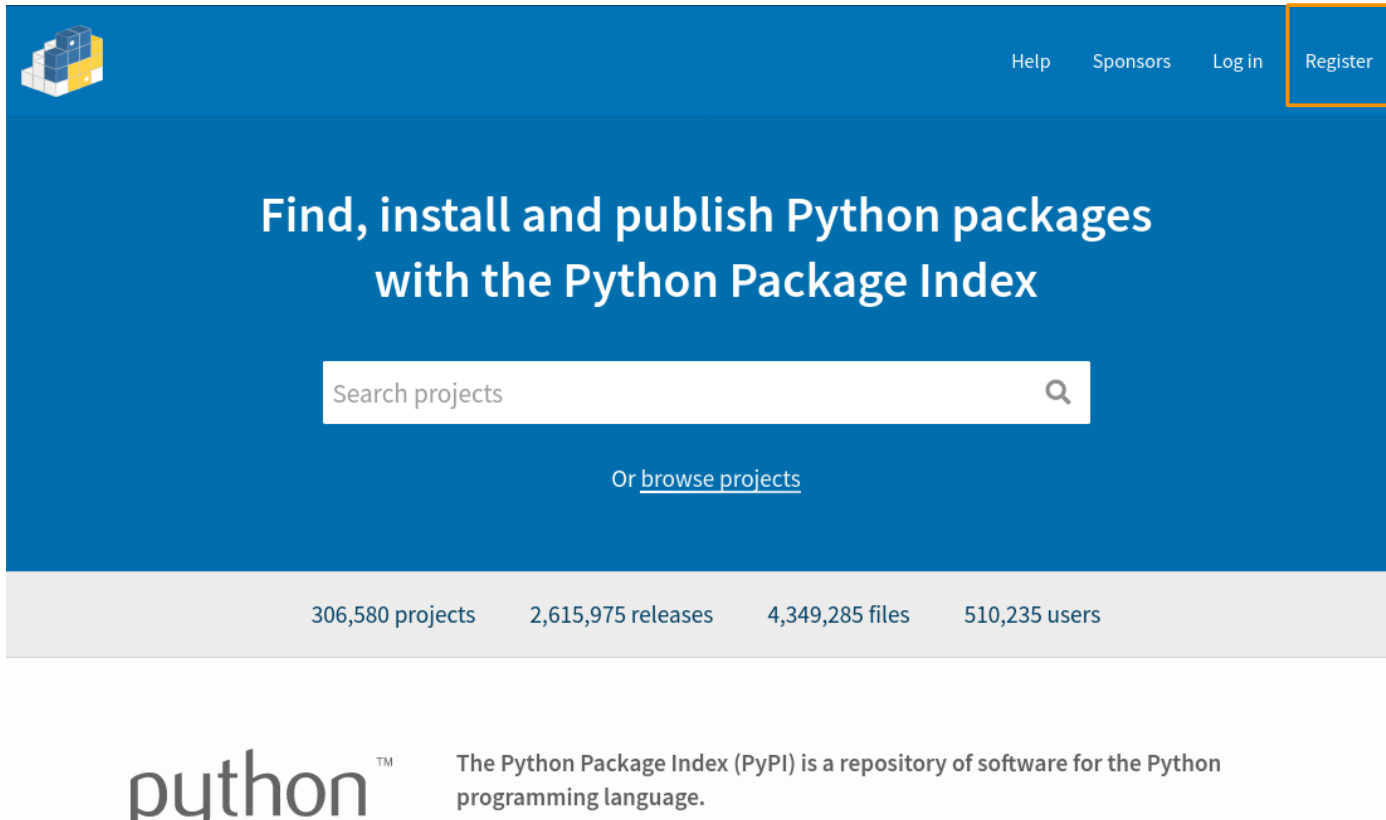
epug_package-0.1.0-py3-none-any.whl

Installing a local file

```
$ pip install dist/epug_package-0.1.0-py3-none-any.whl
Defaulting to user installation because normal site-packages is not writeable
Processing ./dist/epug_package-0.1.0-py3-none-any.whl
Requirement already satisfied: biopython~=1.78 in /home/althonos/.local/lib/python3.9/site-packages
(from epug-package==0.1.0) (1.78)
Requirement already satisfied: numpy in /home/althonos/.local/lib/python3.9/site-packages (from
biopython~=1.78->epug-package==0.1.0) (1.20.2)
Installing collected packages: epug-package
Successfully installed epug-package-0.1.0
```

```
>>> import epug_package
>>> epug_package.say_hello()
Hello, world!
```

Publishing to PyPI



<https://pypi.org/>

Publishing to PyPI

twine 3.4.1

`pip install twine`



[Latest version](#)

Released: Mar 17, 2021

Collection of utilities for publishing packages on PyPI

Navigation

Project description

Release history

Download files

Project links

Project description

pypi v3.4.1 python 3.6 | 3.7 | 3.8 | 3.9 docs passing build passing coverage 94%

twine

Twine is a utility for [publishing](#) Python packages on [PyPI](#).

It provides build system independent uploads of source and binary [distribution artifacts](#) for both new and existing [projects](#).



<https://pypi.org/project/twine>

Publishing to PyPI

```
$ twine upload dist/*
Uploading distributions to https://test.pypi.org/legacy/
Enter your username: althonos
Enter your password:
Uploading epug_package-0.1.0-py3-none-any.whl
100%|████████████████████████████████████████████████████████████████████████████████| 9.00k/9.00k [00:01<00:00, 4.80kB/s]
Uploading epug-package-0.1.0.tar.gz
100%|████████████████████████████████████████████████████████████████████████████████| 16.7k/16.7k [00:01<00:00, 11.7kB/s]


View at:
https://test.pypi.org/project/epug-package/0.1.0/
```



Our published package

 Search projects 

Help Sponsors Log in Register

epug-package 0.1.0

 Latest version

`pip install -i https://test.pypi.org/simple/ epug-package` 

Released: 6 minutes ago

A toy package to be shown at the EMBL Python User Group.

Navigation

Project description

Release history

Download files

Project links

Homepage

Bug Tracker

Project description



A toy package to be shown at the EMBL Python User Group.

license MIT source EMBL

Table of Contents

- [Overview](#)
- [Installing](#)
- [Examples](#)
- [API Reference](#)

Project links

Homepage

Bug Tracker

Statistics

View statistics for this project via [Libraries.io](#), or by using [our public dataset on Google BigQuery](#)

Meta

License: MIT License (MIT)

Author: [Martin Larralde](#)

example, template, toy

Requires: Python >=3.6

Maintainers



[althonos](#)

Classifiers

Development Status

- [3 - Alpha](#)

using TestPyPI – a separate instance

License

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

Operating System

- [OS Independent](#)

Programming Language

- [Python :: 3](#)

Topic


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

Our published package



althonos ▾

Your account

 [Your projects](#)

 [Account settings](#)

Your projects (1)



epug-package SOLE OWNER

Last released 10 minutes ago

A toy package to be shown at the EMBL Python User Group.

[Manage](#)

[View](#)

Overview

- What happens when you install a package?
- How are Python packages distributed?
- How do you create a distribution for your package?
- How do you publish a distribution to PyPI?
- How do you expose a CLI script from your package ?
- How do you include data in your package?

Installing a CLI



setup.cfg

```
[options.entry_points]
console-scripts =
    hello-epug = epug_package:say_hello
```

```
$ python setup.py bdist_wheel
# ... #

$ pip install dist/epug_package-0.1.2-py3-none-any.whl
# ... #

$ hello-epug
Hello, world!
```

Size limits



epug-package

A toy package to be shown at the EMBL Python User Group.

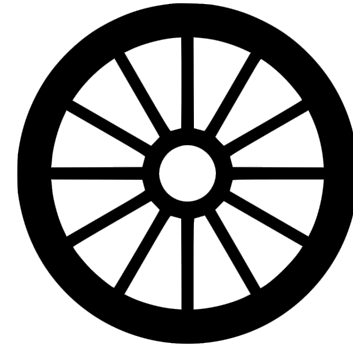
Project settings

- Project size : 9.7 KiB
- Project upload limit : Default (100.0 MiB) [request an increase](#)
- Project total size limit : Default (10.0 GiB) [request an increase](#)

Including data in distributions



MANIFEST.in



setup.cfg

Including data in source distributions



MANIFEST.in

```
include LICENSE
include README.md
include CHANGELOG.md

recursive-include epug_package *.json *.pkl
recursive-exclude tests *
```

Including data in source distributions

```
$ python setup.py sdist
# ... #
copying epug_package/blosum62.json -> epug-package-0.1.0/epug_package
copying epug_package/random_forest.pkl -> epug-package-0.1.0/epug_package
# ... #
Writing epug-package-0.1.0/setup.cfg
Creating tar archive
removing 'epug-package-0.1.0' (and everything under it)
```


Including data in wheels



setup.cfg

```
[options]
include_package_data = true
```

```
[options.package_data]
epug_package = *.json, *.pkl
```

Including data in wheels

```
$ python setup.py bdist_wheel
running bdist_wheel
# ... #
copying build/lib/epug_package/random_forest.pkl -> build/bdist.linux-x86_64/wheel/epug_package
copying build/lib/epug_package/blosum62.json -> build/bdist.linux-x86_64/wheel/epug_package
# ... #
creating 'dist/epug_package-0.1.0-py3-none-any.whl' and adding 'build/bdist.linux-x86_64/wheel' to it
```

Installed data

```
$ pip install dist/epug_package-0.1.0-py3-none-any.whl
Defaulting to user installation because normal site-packages is not writeable
Processing ./dist/epug_package-0.1.0-py3-none-any.whl
Requirement already satisfied: biopython~=1.78 in /home/althonos/.local/lib/python3.9/site-packages
(from epug_package==0.1.0) (1.78)
Requirement already satisfied: numpy in /home/althonos/.local/lib/python3.9/site-packages (from
biopython~=1.78->epug_package==0.1.0) (1.20.2)
Installing collected packages: epug-package
Successfully installed epug-package-0.1.0

$ ls ~/.local/lib/python3.9/site-packages/epug_package
blosum62.json __init__.py model.py __pycache__ random_forest.pkl utils.py
```

Properly loading installed data

```
>>> import importlib.resources
>>> import pickle

>>> with importlib.resources.open_binary("epug_package", "random_forest.pkl") as f:
...     rf = pickle.load(f)

>>> rf
RandomForestClassifier(max_depth=2, random_state=0)
```

<https://docs.python.org/3/library/importlib.html#module-importlib.resources>

Overview

- What happens when you install a package?
- How are Python packages distributed?
- How do you create a distribution for your package?
- How do you publish a distribution to PyPI?
- How do you expose a CLI script from your package ?
- How do you include data in your package?
- How do you use data from an installed package?

Thanks for your attention !

Documentation:

setuptools: <https://setuptools.readthedocs.io>

pip: <https://pip.pypa.io/>

Assets:

Code: <https://carbon.now.sh>

Icons: <https://github.com/numixproject>

Resources:

Repository: <https://git.embl.de/larralde/epug-package>

Package page: <https://test.pypi.org/project/epug-package>