Published in Python in Plain English

Daniel Da Costa Follow
Apr 8, 2020 · 3 min read · Listen

Uploading a Python package to PyPi

As a Python developer, you must be familiar with the python packages installation procedure: *pip install <package_name>*. But maybe, you have never asked yourself how this works behind the curtain. This post is to briefly explain how you can upload you own package to PyPi, so that other programmers can use it.

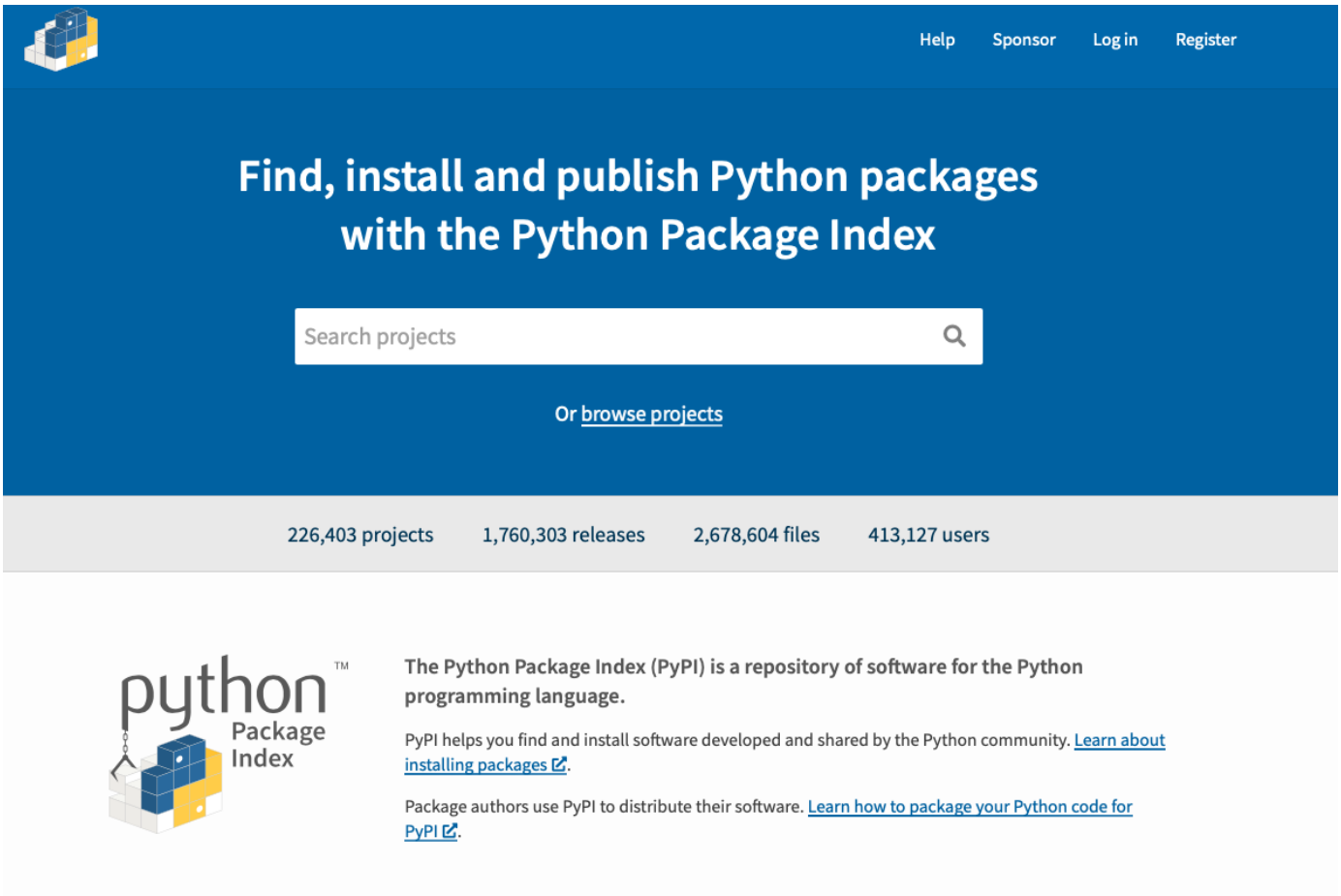

Image taken from: https://pypi.org

Part I: Build a Python package

First, you will have to have your code ready for upload. In this tutorial, I'll be using the package gradient_descent as an example. The code folder tree is organized as follows:

```
gradient-descent
├── Adam.py
├── ExceptionHandler.py
├── GradientDescent.py
├── Momentum.py
```

It's important that your package name matches the package directory name.

Part II: Add __init__.py

This file is responsible for marking the directory a Python package. The code inside the __init__ file gets run whenever you import a package inside a Python program. You can live it empty, but I advise you to use it to export selected portions of the package under a more convenient name.

```
from .GradientDescent import GradientDescent
from .Momentum import Momentum
from .NAG import NAG
from .Adam import Adam
from .RMSprop import RMSprop
```

With the code above, instead of importing the class *Adam* as:

```
from gradient-descent.Adam import Adam
```

You could just:

```
from gradient-descent import Adam
```

Part III: Add setup.py

This file is located at the same level as the *gradient-descent* folder:

```
.
├── gradient_descent
│   ├── Adam.py
│   ├── ExceptionHandler.py
│   ├── GradientDescent.py
│   ├── Momentum.py
│   ├── NAG.py
│   ├── RMSprop.py
│   ├── __init__.py
└── setup.py
```

This file contains information about the package, such as: name, version, description, url and etc. Example:

```
import setuptools

setuptools.setup(
    name="YOUR_PACKAGE_NAME",
    version="0.0.1",
    author="YOUR_NAME",
    author_email="YOUR_EMAIL",
    description="PACKAGE_DESCRIPTION",
    url="PACKAGE_GITHUB",
    packages=setuptools.find_packages(),
    python_requires='>=3.6',
    zip_save=False
)
```

If you want to have a better understanding of the setup.py file, click here.

Here is an example of the MIT license:

```
Copyright <YEAR> <COPYRIGHT HOLDER>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and
associated documentation files (the "Software"), to deal in the Software without restriction, including
without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the
following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial
portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT
LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO
EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR
THE USE OR OTHER DEALINGS IN THE SOFTWARE.
```

You will also have to create a README.md file, where you will add a detailed description of the package and its important informations.

Part V: Setup.cfg

This file provides Python distribution's metadata and build configuration. You will only have to add the following lines inside the file:

```
[metadata]
description-file = README.md
```

The file should be located inside the package directory.

In the end, you will end up with following folder tree:

```
.
├── gradient_descent
│   ├── Adam.py
│   ├── ExceptionHandler.py
│   ├── GradientDescent.py
│   ├── Momentum.py
│   ├── NAG.py
│   ├── README.md
│   ├── RMSprop.py
│   ├── __init__.py
│   ├── license.txt
│   └── setup.cfg
└── setup.py
```

Part VI: PyPi

In order to upload a package to PyPi, you will have to create an account on the following website: https://pypi.org

Once you have created your account, you will have to run the following commands inside the folder with your package files:

```
python setup.py sdist
```

Install the twine package:

Finally:

```
twine upload dist/*
```

It will ask for your username and password of your PyPi account. You can check on the regular PyPi repository website if everything was correctly uploaded.

Congratulations! You've just uploaded your fist package to PyPi. Now you can install it using the simple command:

```
pip install <your_package_name>
```

References

- https://pypi.org
- packaging.python.org