

Publishing your First PyPI Package by/for the Absolute Beginner

The process for packaging and publishing Python packages is subject to much community discussion and frequent change. That leads to a lot of outdated information about it floating around the interwebs.

As part of [my PyCon 2017 talk](#) preparations I wrote [a driver for a barcode reader](#) that was part of the live demo in my presentation. Having never published a package to PyPI before¹, I thought this would be a good opportunity to learn about the Python Package Index (PyPI) and how to use it.

It only took me a quick five minutes to realize that the top search results contain piles of obsolete and contradictory information. Case in point: Even the [instructions linked to on pypi.python.org](#) are not the “state of the art” and are, in fact, actively discouraged elsewhere.²

Not discouraged by having to do a little bit of internet archeology, I had a draft of this blog post ready in June 2017. Because of [reasons] I left it in the draft folder all summer. In September 2017 I did a quick final review before publishing, or so I thought. Instead, I found that my own instructions from June no longer worked in September because of a major update to PyPI that had happened in the meantime!

I updated the post and confirmed that the following instructions work on September 13, 2017. No guarantees about whether they still work on September 14.

Don't copy-and-paste this!

```
# Seriously, don't use these:  
python setup.py register  
python setup.py sdist upload
```

These two commands for registering and uploading a package to PyPI are considered obsolete by most recent sources I could find. The [Python Packaging Authority tutorial](#) explains why:

These methods of registering and uploading a package are strongly discouraged as it may use a plain text HTTP or unverified HTTPS connection on some Python versions, allowing your username and password to be intercepted during transmission.

“test PyPI” and “real PyPI”

Because all write operations to `pypi.org` are publicly visible and (mostly) not reversible, I was glad to find that there is a sandbox environment at `test.pypi.org`.

`test.pypi.org` is the exact same software as `pypi.org`, but pointed at a separate database (no shared user accounts, package data, etc).

`test.pypi.org` is publicly visible (but nobody cares to look at it), and allegedly gets reset occasionally³.

The obvious process to follow for the absolute beginner is:

- Register only for `test.pypi.org` and go through all steps there.
- Only once everything works, repeat all steps (including registration) with `pypi.org`.

pypi.org and pypi.python.org

Currently (September 2017), PyPI is being migrated from an old software (called “Cheeseshop”) running at `pypi.python.org` to a new software (called “Warehouse”) running on `pypi.org`.

While the pages look very different, they point to the same database. The

sunsetting of `pypi.python.org` has been announced in [this mailing list post](#).

Some of the steps below still work with the old URL, and some don't. Theoretically, there is no reason to *not* use the new URL. Practically, the new software has bugs that necessitate using the old URL as a workaround. More on that below.

Let's get started with uploading that package.

Step 0: Have a Package to Upload

I assume that you have a package ready to be published. If you don't, the "More Reading" section below points to instructions for how to create a Python package.

tl;dr: Run `setup.py dist` in the root directory of your package to get a `dist/package-0.1.2.tar.gz` file. That's the file we will upload to PyPI. `package` is the name users will later use to `pip install package`, `0.1.2` is the version number, both are specified in `setup.py`.

Step 1: Create a User Account on PyPI

Head to <https://testpypi.python.org/pypi/> and click the "Register" link to get to the [registration form](#).⁴

Check your inbox for an email from `admin@mail.pypi.python.org` and click the link in the email. It will take you to

`https://pypi.python.org/pypi?action=user&otk=[your_token]`. This page contains terms and conditions for PyPI which aren't otherwise available via a direct link on either old or new website, but you can get read them [in the Warehouse source code](#).

This brief legal blurb is worth reading because it reminds you that you own the content you upload to PyPI. And, by extension, you also own all the legal trouble your content might cause. Two bits specifically caught my curious eye:

I further warrant that I do not know of any applicable unlicensed patent or trademark rights that would be infringed by my uploaded Content.

You probably authored the Python source code you are uploading yourself. But what about content that's included? Including a trademark protected logo as a graphical asset seems like an easy way to accidentally infringe someone else's

rights.

I further affirm that any Content I provide is not intended for use by a government end-user engaged in the manufacture or distribution of items or services controlled on the Wassenaar Munitions List as defined in part 772 of the United States Export Administration Regulations, or by any other embargoed user.

This is quite interesting because it seems so foreign and unrelated—but isn't. If your Python package is at all related to cryptography or security research, consider googling these terms! ⁵

Also: Make sure your package includes a license. [Click here if you're not sure why](#).

Step 2: `~/.pypirc`

Now (Test)PyPI is ready for your package. Next step is to configure your development machine to talk to (Test)PyPI. That's what `~/.pypirc` is for:

```
# ~/.pypirc

[distutils]
index-servers =
    pypitest

[pypitest]
repository: https://test.pypi.org/legacy/
username: YOUR_USERNAME_GOES_HERE
```

Replace `YOUR_USERNAME_GOES_HERE` with the username you create an account for in Step 1. You can also include your password, with the obvious security implications:

```
# append to end of file, if you wish
password: SECRET!
```

In case you are wondering why the URL contains `legacy`, then [you're not](#)

alone. This is to signal, that the new Warehouse software emulates the Cheeseshop's API, for now. In future, there will, presumably, be a different API.

Step 3: Install twine

Packages are sent to PyPI as [tarballs](#) via a web API. You could use cURL, but why would you? You could use `setup.py upload`, but I already told you above why not to.

Use `twine` instead, it's the recommended (by the Python Packaging Authority contributors) PyPI client.

Installing `twine` is straightforward and strangely meta:

```
pip install twine
```

Step 4: Use twine to upload your package

~~Register your new package with PyPI~~⁶

Upload your tarball⁷:

```
$ twine upload -r pypitest dist/package-0.1.2.tar.gz
Uploading distributions to https://test.pypi.org/legacy/
Enter your password:
Uploading package-0.1.2.tar.gz
```

Next, go to <https://test.pypi.org/project/package/> to confirm that your package shows up on (Test)PyPI. If it does, you are ready to graduate to the "real" PyPI!

Final: Repeat with pypi.org

1. Register on pypi.python.org ([direct link to registration form](#))

2. Click confirmation link in email and accept terms
3. Update your `~/.pypirc` to include a `[pypi]` section:

```
# ~/.pypirc

[distutils]
index-servers =
    pypi
    pypitest

[pypi]
repository: https://pypi.org/legacy/
username: YOUR_USERNAME_GOES_HERE

[pypitest]
repository: https://test.pypi.org/legacy/
username: YOUR_USERNAME_GOES_HERE
```

4. `twine upload -r pypi dist/package-name-0.1.2.tar.gz`
5. Check whether it worked:
`https://test.pypi.org/project/package-name/`

Hurray! You uploaded your first Python package!

More Reading

- The official reference for all things Python Packaging is provided by the Python Packaging Authority (PyPA) at <https://packaging.python.org/>.
- Hynek Schlawack wrote a “[A completely incomplete guide to packaging a Python module and sharing it with the world on PyPI](#)”. In May, this post confused me because I didn’t realize that there were substantial changes to publishing to PyPI since 2013 (when Hynek first published this post). However, he updated the post in June 2017 making it a better superset of what I wrote here :) (Second) however, the sunset of `pypi.python.org` in favor of `pypi.org` render his instructions outdated again :(
- David Forgas’s PyCon 2017 talk “[Share Your Code! Python Packaging Without Complication](#)” ([video](#)) covers approximately the same material as Hynek’s blog post, including what’s covered by me here.
- Kenneth Reitz (who might or might not have a trademark on the pattern

“[Python related term] for humans”) created a [setup.py template](#) which includes code to provide a `setup.py publish` command that uploads the package to Python (using `twine` under the hood). He calls it, you guessed it, “setup.py for humans”.

Other Things I Learned

I’ve been procrastinating over a half-written version of this blog post for several months. Knowing I’d eventually finish it, I kept taking notes on everything PyPI related. These bullet points aren’t related to the title of the blog post, but were interesting enough to include anyway:

- Some people pronounce it “pie-pie” which leads to confusion with [PyPy](#). Call it “pie-pee-eye” instead.
- There is a story for why the software running [pypi.python.org](#) is called “Cheese Shop” [on Wikipedia](#)
- [Talk Python episode 64](#) is an interview with [Donald Stufft](#) who, together with a very small group of contributors, keeps PyPI running.
- [Donald’s blog post](#) makes clear just how little resources and manpower goes into PyPI. The best part is in footnote 5: “[Cheese Shop] was a weekend hack for a proof of concept that was intended to get quickly replaced by the *real* code.”
- It is possible to donate specifically to the Python Packaging Workgroup (who maintains and develops PyPI) [here](#). While personal donations help, convincing your employer who derives monetary value from the Python ecosystem to contribute regularly might be more impactful.

Footnotes

1. I have uploaded packages to a company-internal [devpi](#) server. While similar in principle, the steps for doing so are entirely different than working with the public PyPI service. [↩](#)
2. You might wonder: “That page is a wiki, why not just edit it?” – Because editing is disabled for everyone but the [editors group](#). [↩](#)
3. “Allegedly” because I know for sure that Test PyPI did not get erased between May 2017 and September 2017 and I suspect it never happens. That’s confirmed by Donald Stufft who writes in a [Github issue](#): “TestPyPI basically never gets purged. I think it’s happened once or twice ever and it is an entirely manual process.” [↩](#)
4. Note how this step uses the old URL `pypitest.python.org` instead of the new URL

test.pypi.org. That's because the new Warehouse software fails to send a confirmation email after registration, making it impossible to activate a new user account, ultimately preventing the new user from uploading packages. This bug is tracked [in Github here](#). ↩

5. The Wassenaar Agreement (part of which is the "Munitions List") covers some (very specific) software. If you are planning to publish software that falls under "security research", consider reading up on the details. [This 2015 Verge article](#) is a brief summary of why there's been some uproar related to the Wassenaar Agreement. ↩
6. Until July 2017 packages had to be registered before uploading for the first time. The API of the new "Warehouse" PyPI kindly points out that "project pre-registration is no longer required or supported, so continue directly to uploading files." ↩
7. It is common practice to not only upload a tarball of the Python source of your package to PyPI but also one or more Python wheels. You can read the docs about Python wheels [here](#), but note that the examples there use the discouraged `setup.py upload` method of uploading to PyPI. If you have wheels to upload to PyPI, you can either call `twine upload` multiple times in a row, once for each file in your `dist` folder, or just use `twine upload dist/*.tar.gz` . ↩

[TABLE OF CONTENTS](#) ↩

[JONASNEUBERT.COM](#) ↩


[TWITTER.COM/JONEMO](#) ↩

[ARCHIVE](#) ↩

2019  ↩

2018  ↩

2017  ↩

2014  ↩

2012  ↩

2011  ↩

