

From Script to Open Source Project

Python standards, tools and continuous integration

Michał Karzyński • EuroPython 2019



europython
July 8-14 2019 BASEL

So, you wanna be a

ROCK STAR

Step 1



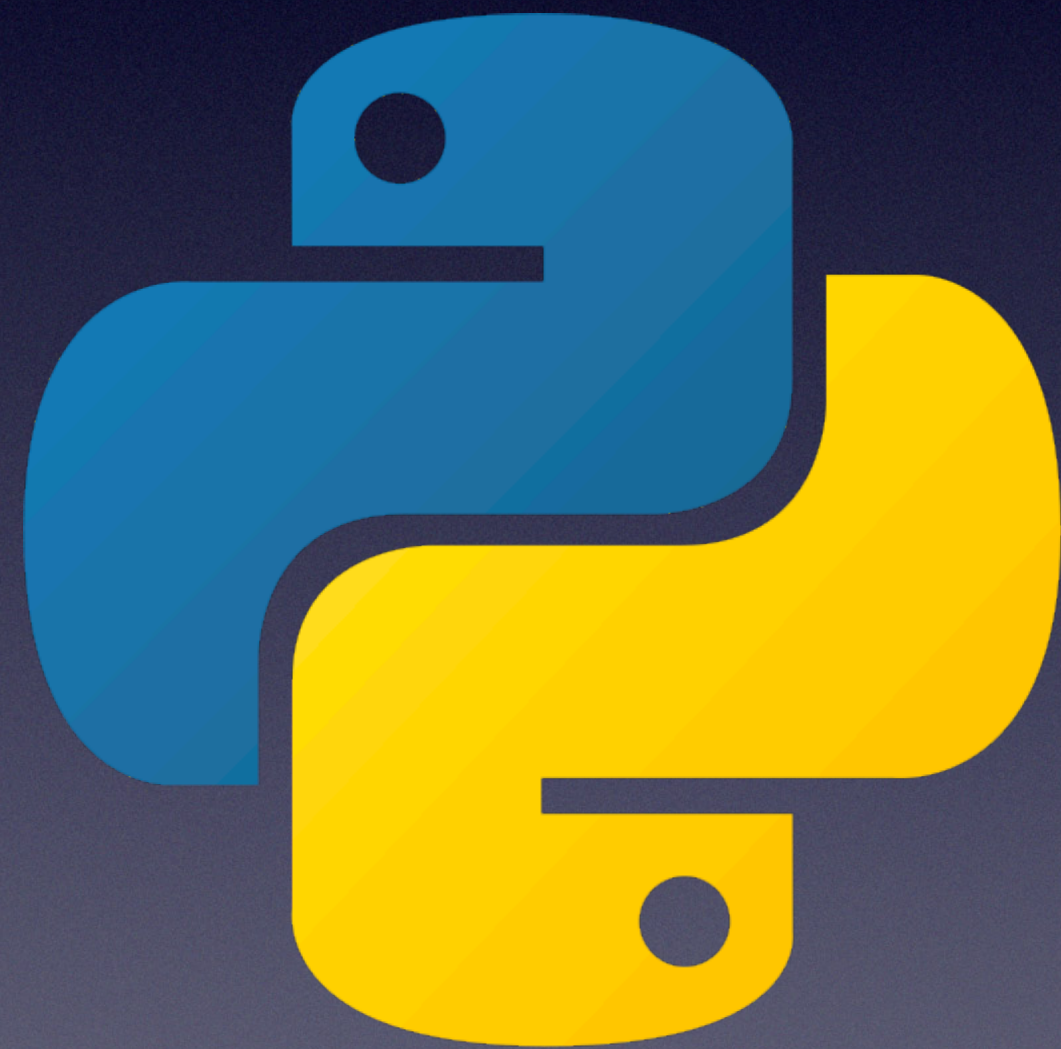
Master your instrument

Step 2



Learn to play in a band

What can help you play together better?



- Standards
- Best practices
- Tools

About me

- Michał Karzyński (@postrational)
- Full stack geek (C++, Python, JavaScript)
- I blog at michal.karzynski.pl
- I'm an architect at  working on



Open AI Gym Demo





```
1. gym-demo SpaceInvaders-ram-v4 (Python)

Environment: SpaceInvaders-ram-v4

Observation Space:
Box(128,)
Low values:
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
High values:
[255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
 255 255]

Action Space:
Discrete(6)
Action meanings: ['NOOP', 'FIRE', 'RIGHT', 'LEFT', 'RIGHTFIRE', 'LEFTFIRE']

Running environment demonstration ...
Unique environment information is output to standard out:
2019-02-22 18:36:52.149 Python[68647:1852448] ApplePersistenceIgnoreState: Existing state will not b
e touched. New state will be written to (null)
Reward: 0.0, Done: False, Info: {'ale.lives': 3}
Reward: 5.0, Done: False, Info: {'ale.lives': 3}
Reward: 0.0, Done: False, Info: {'ale.lives': 3}
Reward: 10.0, Done: False, Info: {'ale.lives': 3}
Reward: 0.0, Done: False, Info: {'ale.lives': 3}
```


Stages	Code Prep	Automate	CI	
Specs	 PEP8	 GNU/POSIX	 PyPA	 PyCQA
pip install	pytest virtualenv	mypy wheel	docopt black	setuptools pre-commit tox flake8
Services	↗ GitHub ↗ codeclimate.com	↗ TravisCI ↗ codacy.com	↗ coveralls.io ↗ mergify.io	↗ Dependabot ↗ pyup.io

Your command-line interface (CLI)

```
$ gym-demo
```

```
Usage: gym-demo [--steps=NN --no-render --observations] ENV_NAME
```

```
$ gym-demo --help
```

```
$ gym-demo --steps=5 --no-render Pendulum-v0
```

```
$ gym-demo -ns 5 Pendulum-v0
```


Your command-line interface (CLI)

```
#!/usr/bin/env python
```

```
"""Usage: gym-demo [--steps=NN --no-render --observations] ENV_NAME
```

```
Show a random agent playing in a given OpenAI environment.
```

```
Arguments:
```

```
    ENV_NAME           Name of the Gym environment to run
```

```
Options:
```

```
    -h --help
```

```
    -s --steps=<STEPS> How many iteration to run for. [default: 5000]
```


```
    -n --no-render      Don't render the environment graphically.
```

```
    -o --observations   Print environment observations.
```

```
"""
```


Your command-line interface (CLI)

```
import docopt

arguments = docopt(__doc__) 

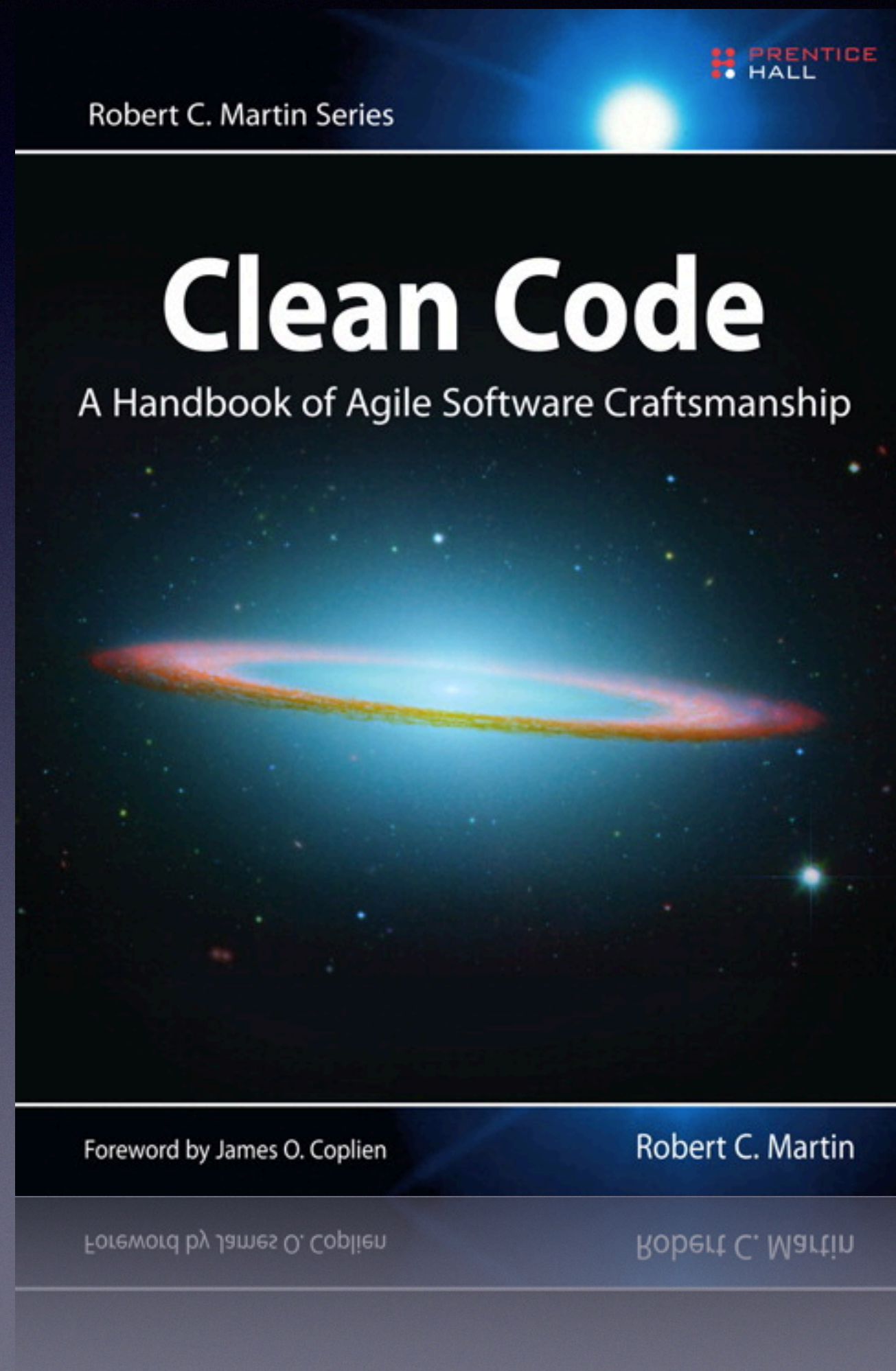
print_observations = arguments.get("--observations")
steps = int(arguments.get("--steps"))
render_env = not arguments.get("--no-render")
```


Code directory layout

```
package-name
├── LICENSE
├── README.md
├── main_module_name
│   ├── __init__.py
│   ├── helpers.py
│   └── main.py
├── docs
│   ├── conf.py
│   └── index.rst
├── tests
│   └── test_main.py
├── requirements.txt
└── setup.py
```

src

Code structure



- meaningful names
- single responsibility
- up to 2 parameters
- preferably no side-effects
- write unit tests



[@unclebobmartin](https://twitter.com/unclebobmartin)

Code Prep

Define your main function

```
if __name__ == "__main__":  
    main()
```


Preparing your setup.py file

```
#!/usr/bin/env python
import os
from setuptools import setup

setup(
    name="gym-demo",
    version="0.2.1",
    description="Explore OpenAI Gym environments.",
    long_description=open(
        os.path.join(os.path.abspath(os.path.dirname(__file__)), "README.md")
    ).read(),
    long_description_content_type="text/markdown",
    author="Michał Karzynski",
    packages=["gym_demo"],
    install_requires=["setuptools", "docopt"],
)
```


Using your setup.py file

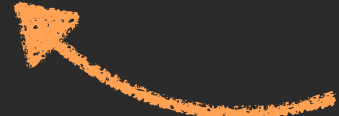
```
$ python setup.py sdist          # Prepare a source package
$ python setup.py bdist_wheel    # Prepare a binary wheel for distribution

# Start local development in a Virtualenv:
$ source my_venv/bin/activate
(my_venv)$ python setup.py develop
or
(my_venv)$ pip install -e .
```


Add entry_points to setup.py

 setup.py

```
setup(  
    # other arguments here...  
  
    # my_module.main:main points to the method main in my_module/main.py  
    entry_points={"console_scripts": ["my-command = my_module.main:main"]},  
)
```



Create a requirements.txt file

 requirements.txt

```
colorful==0.5.0  
docopt==0.6.2  
gym==0.12.5  
another_package>=1.0,<=2.0  
git+https://myvcs.com/some_dependency@sometag#egg=SomeDependency
```



```
$ pip freeze > requirements.txt
```

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

```
$ pip install -r requirements_test.txt
```


Use Black to format your code

```
(my_venv) $ black my_module  
All done! ✨ 🍰 ✨  
1 file reformatted, 7 files left unchanged.
```

 PEP8

black

Automate

Use pre-commit to run formatters

 `.pre-commit-config.yaml`

```
repos:
- repo: https://github.com/ambv/black
  rev: stable
  hooks:
    - id: black
```

```
(my_venv) $ pre-commit install
(my_venv) $ git commit
black..... Failed
hookid: black
```

Files were modified by this hook.
Additional output:

```
reformatted gym_demo/demo.py
All done! ✨ 🍰 ✨
1 file reformatted.
```


Use flake8 to check your code

 requirements_test.txt

```
flake8
flake8-blind-except
flake8-bugbear
flake8-builtins
flake8-comprehensions
flake8-debugger
flake8-docstrings
flake8-isort
flake8-quotes
flake8-string-format
```

 tox.ini

```
[flake8]
max-line-length=88
max-complexity=6
inline-quotes=double
; ignore:
; C812 - Missing trailing comma
; D104 - Missing docstring in package
ignore=C812,D104
```

```
(my_venv) $ flake8
./my_package/my_module.py:1:1: D100 Missing docstring in public module
```


Use MyPy for static type analysis

```
from typing import List, Text, Mapping, Union, Optional

def greeting(name: Text) -> Text:
    return "Hello {}".format(name)

def my_function(name: Optional[Text] = None) -> Mapping[str, Union[int, float]]:
    ...
```

```
(my_venv) $ mypy --config-file=tox.ini my_module
my_module/main.py:43:27: error: Argument 1 to "my_function" has incompatible
type "int"; expected "List[str]"
```


Use tox to test all the things

 **tox.ini**

```
[tox]
envlist=py35,py36,py37

[testenv]
deps=
    -Urrequirements.txt
    -Urrequirements_test.txt
commands=
    flake8
    pytest tests/

[pytest]
timeout=300
```

```
$ tox -e py37
GLOB sdist-make: .../setup.py
py37 create: .../.tox/py37
py37 installdeps: -Urrequirements.txt
py37 inst: gym-demo-0.2.2.zip
py37 run-test: commands[1] | flake8
py37 run-test: commands[4] | pytest
...

_____ summary _____
py37: commands succeeded
congratulations :)
The command exited with 0.
```


Write unit tests

 test/test_main.py

```
"""Test suite for my-project."""
import pytest
from my_project import my_function

def test_my_function():
    result = my_function()
    assert result == "Hello World!"
```

```
$ pytest
===== test session starts =====
rootdir: my-project, inifile: tox.ini
plugins: timeout-1.3.3, cov-2.7.1
timeout: 300.0s
timeout method: signal
timeout func_only: False
collected 7 items

tests/test_main.py ..... [100%]

===== 7 passed in 0.35 seconds =====
```


Set up a Git repository

Skip this step if you're importing an existing repository.

☒ **Initialize this repository with a README**

This will let you immediately clone the repository to your computer.

Add .gitignore: Python ▼

Add a license: MIT License ▼



Create repository

Create repository

```
$ git init
$ git remote add origin https://github.com/you/your-project.git
$ git pull origin master

$ git add --all
$ git commit -m 'First commit'
$ git push -u origin master
```

➤ GitHub

➤ gitignore.io

➤ choosealicense.com

Code Prep

Set up continuous integration

Travis CI About Us Blog Status Documentation Help [Sign in with GitHub](#)

Help make Open Source a better place and start building better software today!

postrational / gym-demo build passing

Current Branches Build History Pull Requests > Build #63 Job #63.1 More options

✓ Pull Request #16 Add colorful output #63.1 passed

Commit df15daf #16: Add colorful output Branch master

Michał Karzyński

Python: 3.4

Job log View config

Raw log

```
1 Worker information worker_info
6 Build system information system_info
413 3.4 is not installed; attempting download
414 Downloading archive: https://travis-ci-python-
415 archives.global.ssl.fastly.net/binaries/ubuntu/14.04/x86_64/python-3.4.tar.bz2
416 $ curl -sSf -o python-3.4.tar.bz2 ${archive_url} 0.50s
417 $ sudo tar xjf python-3.4.tar.bz2 --directory / 8.23s
418
419 $ git clone https://github.com/postrational/gym-demo.git postrational/gym-demo git.checkout 0.71s
436
437 $ source ~/virtualenv/python3.4/bin/activate 0.00s
438 Setting up build cache cache.1
449
450 $ python --version
450 2.7.12 Ubuntu 14.04.2 LTS
451
452 Setting up build cache cache.1
453 $ source ~/virtualenv/python3.4/bin/activate 0.00s
454
455 $ git clone https://github.com/postrational/gym-demo.git postrational/gym-demo 0.71s
```

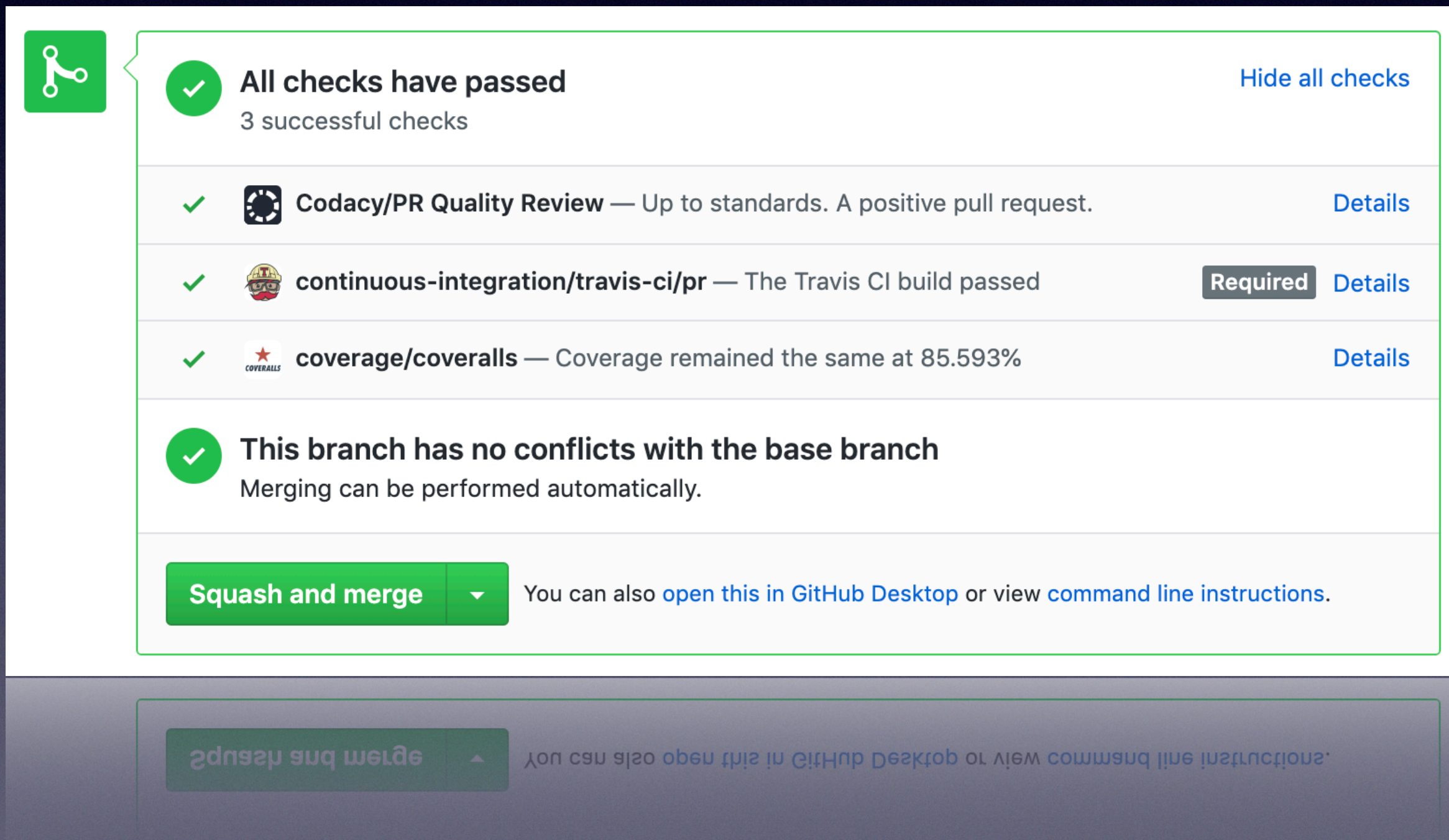
```
 .travis.yml

language: python
os: linux
install:
  - pip install tox
script:
  - tox
git:
  depth: false
branches:
  only:
    - "master"
cache:
  directories:
    - $HOME/.cache/pip
```

➔ TravisCI


CI


Set up continuous integration




A screenshot of a GitHub pull request interface. At the top, a green box with a checkmark icon and the text "All checks have passed" and "3 successful checks" is displayed, with a "Hide all checks" link. Below this, a list of checks is shown, each with a green checkmark, an icon, a title, a description, and a "Details" link. The checks are: "Codacy/PR Quality Review — Up to standards. A positive pull request.", "continuous-integration/travis-ci/pr — The Travis CI build passed" (marked as "Required"), and "coverage/coveralls — Coverage remained the same at 85.593%". At the bottom, a green "Squash and merge" button is shown, followed by the text "You can also open this in GitHub Desktop or view command line instructions."

✓ All checks have passed
3 successful checks [Hide all checks](#)


✓  Codacy/PR Quality Review — Up to standards. A positive pull request. [Details](#)

✓  continuous-integration/travis-ci/pr — The Travis CI build passed **Required** [Details](#)

✓  coverage/coveralls — Coverage remained the same at 85.593% [Details](#)

✓ This branch has no conflicts with the base branch
Merging can be performed automatically.

Squash and merge ▾ You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

```
 .travis.yml

language: python
os: linux
install:
  - pip install tox
script:
  - tox
git:
  depth: false
branches:
  only:
    - "master"
cache:
  directories:
    - $HOME/.cache/pip
```

➤ TravisCI

CI

Requirements updater

The screenshot shows the GitHub interface for the repository `postrational/gym-demo`. The `Requirements` tab is active, displaying a list of packages from `requirements.txt` and `requirements_test.txt`. The interface includes a search bar, filters for `pyup` (6 updates) and `python 3` (1 blocker), and a table of package details.

File	Package Name	Current Version	Latest Version	Status	Python Version	Permissions	Actions
requirements.txt	colorful	==0.5.0	0.5.0	up-to-date	Python 3	Permissive	
	docopt	==0.6.2	0.6.2	up-to-date	Python 3	Permissive	
	gym	==0.12.5	0.13.0	outdated	Python 3	Unknown	PR
requirements_test.txt	coveralls	==1.8.0	1.8.1	outdated	Python 3	Permissive	PR
	flake8	==3.7.7	3.7.7	up-to-date	Python 3	Permissive	
	flake8-blind-except	==0.1.1	0.1.1	up-to-date	Python 3	Permissive	

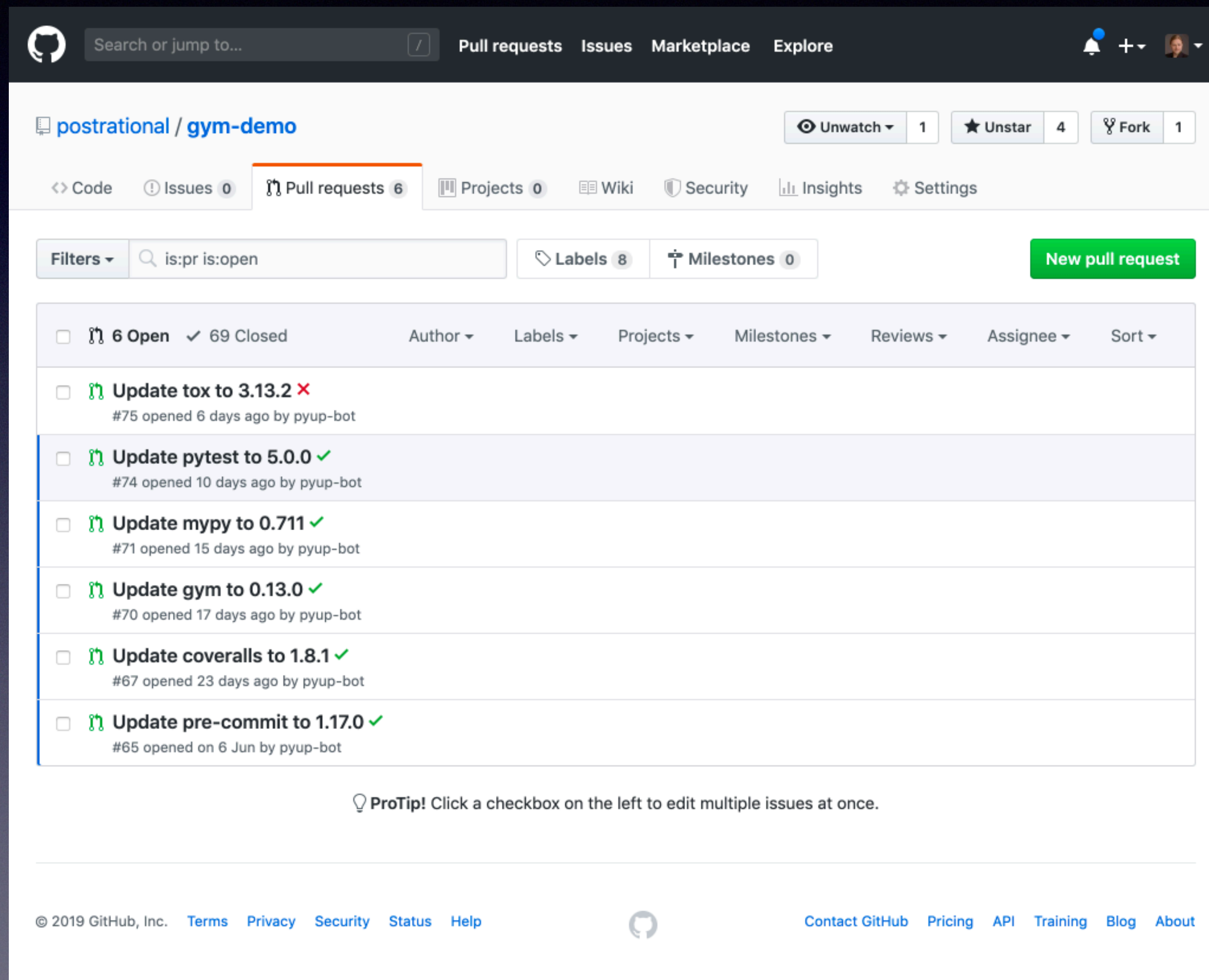
- No configuration
- Just log in with GitHub and give the bot access permissions
- Bot will find your requirements files

➤ Dependabot

➤ pyup.io

CI

Requirements updater



The screenshot shows the GitHub interface for the repository 'postrational / gym-demo'. The 'Pull requests' tab is selected, showing a list of 6 open pull requests. The first pull request, 'Update tox to 3.13.2', is marked with a red 'X' and is the first item in the list. The other five pull requests are marked with green checkmarks and are listed below it. The pull requests are: 'Update tox to 3.13.2', 'Update pytest to 5.0.0', 'Update mypy to 0.711', 'Update gym to 0.13.0', 'Update coveralls to 1.8.1', and 'Update pre-commit to 1.17.0'. All pull requests were opened by 'pyup-bot'. The interface includes a search bar, navigation tabs for Code, Issues, Pull requests, Projects, Wiki, Security, Insights, and Settings. A 'New pull request' button is visible on the right. A 'ProTip!' message at the bottom of the pull request list states: 'Click a checkbox on the left to edit multiple issues at once.'

Checkbox	Icon	Title	Status	Author	Opened
<input type="checkbox"/>	🐛	Update tox to 3.13.2	❌	pyup-bot	#75 opened 6 days ago
<input type="checkbox"/>	🐛	Update pytest to 5.0.0	✅	pyup-bot	#74 opened 10 days ago
<input type="checkbox"/>	🐛	Update mypy to 0.711	✅	pyup-bot	#71 opened 15 days ago
<input type="checkbox"/>	🐛	Update gym to 0.13.0	✅	pyup-bot	#70 opened 17 days ago
<input type="checkbox"/>	🐛	Update coveralls to 1.8.1	✅	pyup-bot	#67 opened 23 days ago
<input type="checkbox"/>	🐛	Update pre-commit to 1.17.0	✅	pyup-bot	#65 opened on 6 Jun

- The bot will start making update PRs
- Which your CI process will test

➤ Dependabot

➤ pyup.io

CI

Test coverage checker

```
$ pytest --cov=my_module tests/
===== test session starts =====

tests/test_main.py ..... [100%]

----- coverage: platform darwin, python 3.7.2-final-0 -----
Name                               Stmts  Miss  Cover
-----
my_module/__init__.py                0      0   100%
my_module/main.py                   77     17    78%
my_module/utils.py                   41      0   100%
-----
TOTAL                               118     17    86%

===== 255 passed in 1.25 seconds =====
```

pytest

pytest-cov

Automate

Test coverage checker

```
$ pytest --cov=my_module \
      --cov-report=html tests/
```



```
$ open htmlcov/index.html
```

```
Coverage for gym_demo/demo.py : 78%
77 statements  60 run  17 missing  0 excluded

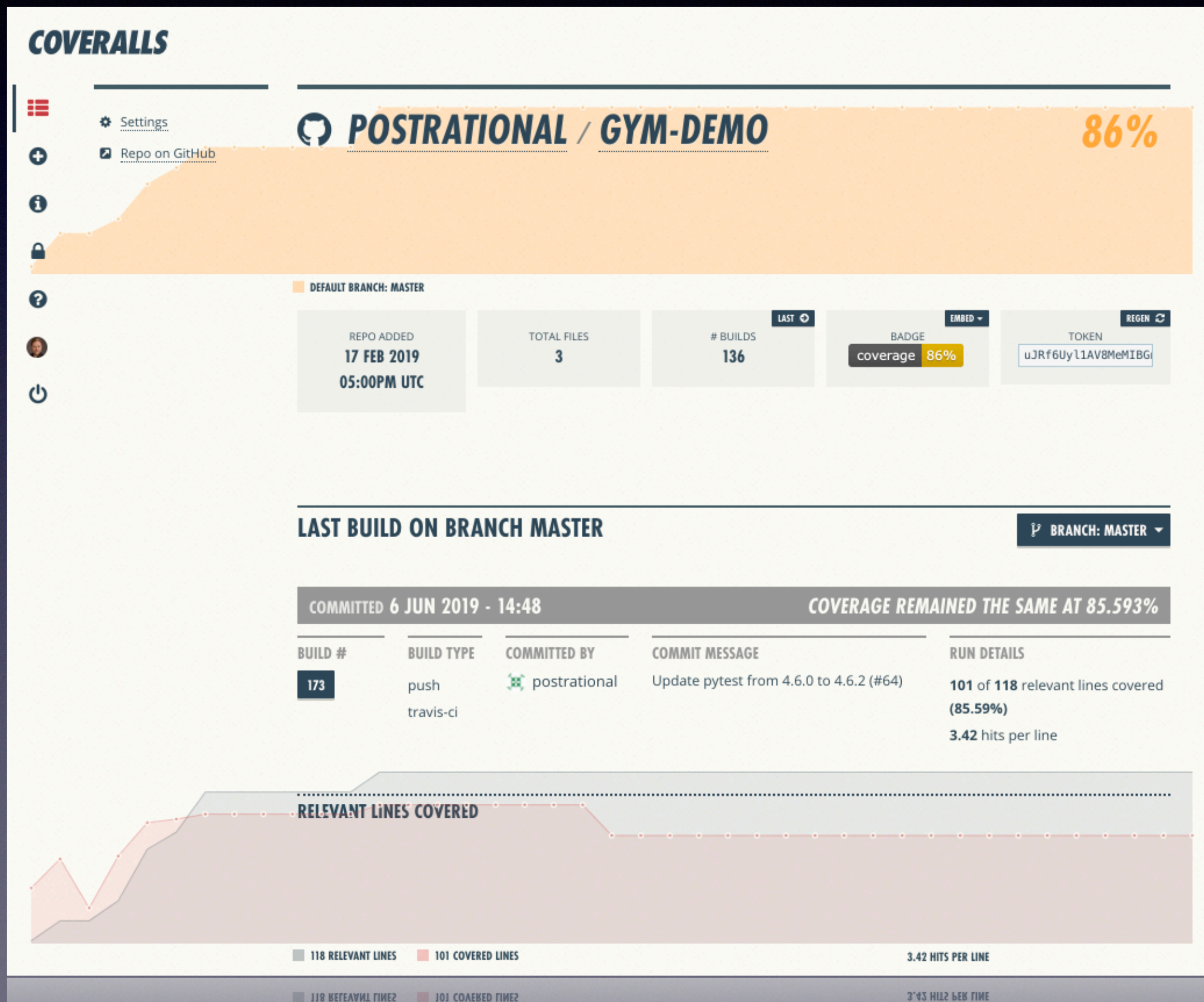
1  #!/usr/bin/env python
2
3  """Usage: gym_demo.py [--steps=NN --no-render --observations] ENV_NAME
4
5  Show a random agent playing in a given OpenAI environment.
6
7  Arguments:
8    ENV_NAME      Name of the Gym environment to run
9
10 Options:
11   -h --help
12   --steps=<STEPS>  How many iteration to run for. [default: 5000]
13   --no-render      Don't render the environment graphically.
14   --observations   Print environment observations.
15
16 """
17 import re
18 from time import sleep
19 from typing import List, Text
20
21 import gym
22 from docopt import docopt
23
24 from gym_demo.formatting import list_to_columns, print_error, print_header
25
26
27 def get_environment_names() -> List[Text]:
28     """Return a list of names of registered Open AI Gym environments."""
29     return sorted(spec.id for spec in gym.envs.registry.all())
30
31
32 def group_environments(env_names: List[Text]) -> List[Text]:
33     """Group a sorted list of environment names into families."""
34     steps = int(arguments.get("--steps"))
35     render_env = not arguments.get("--no-render")
36     print_observations = arguments.get("--observations")
37     env_name = arguments.get("ENV_NAME")
38     env_names = arguments.get("ENV_NAME")
39     render_env = not arguments.get("--no-render")
40     print_observations = arguments.get("--observations")
41     steps = int(arguments.get("--steps"))
42
43     """Group a sorted list of environment names into families."""
44     return sorted(spec.id for spec in gym.envs.registry.all())
```

pytest

pytest-cov

Automate

Test coverage checker



tox.ini

```
[testenv]
...
commands=
    ...
    pytest --cov=my_module tests/
    - coveralls
```



coveralls.io

coveralls

pytest-cov


CI


Automated code review


 codacy-bot reviewed 13 days ago [View changes](#)

onnx-tutorial.md Outdated



```
12 + * [Use pre-built packages](#install-pre-built-packages)
13 + * [Build nGraph from source](#build-from-source)
14 + * [Examples](#examples)
15 + * [Run inference on a model](#run-inference-on-a-model)
```

 codacy-bot 13 days ago + 😊 ...

 Issue found: [\[list-item-bullet-indent\]](#) Incorrect indentation before bullet: remove 2 spaces



[Resolve conversation](#)

 codacy-bot reviewed 13 days ago [View changes](#)

onnx-tutorial.md Outdated

```
7 + This Get Started tutorial is divided into two parts: a) installation
8 + and b)
9 + examples of how to use nGraph with ONNX.
10 + * [Installation](#installation)
```

```
10 + * [Installation](#installation)
9 +
8 + examples of how to use nGraph with ONNX.
```

↗ codeclimate.com

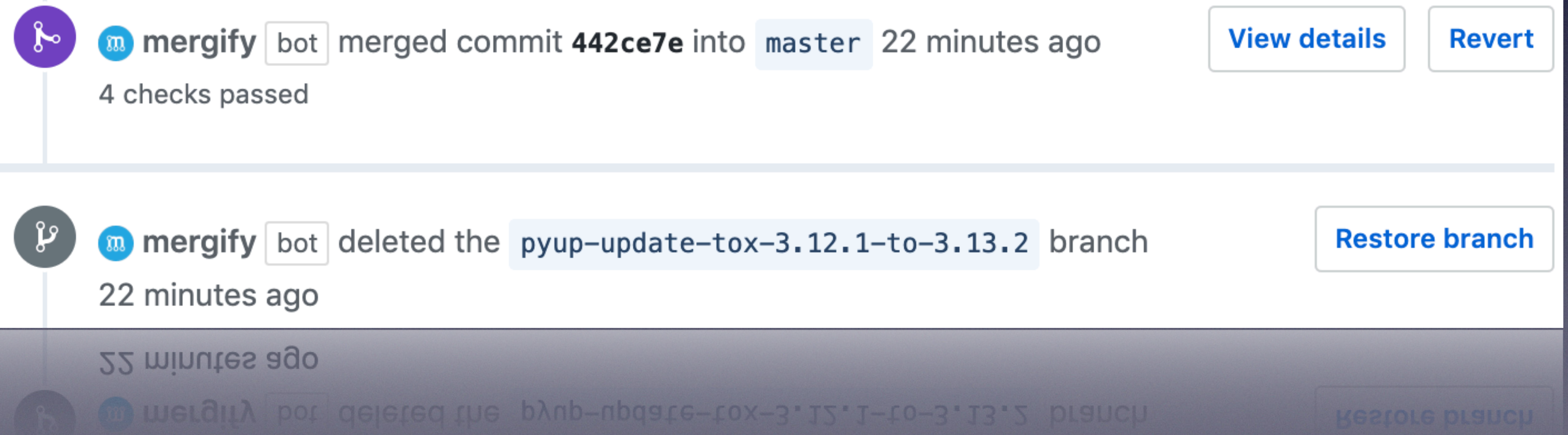
↗ codacy.com

CI



Automated PR merge



 .mergify.yml



```
pull_request_rules:  
  - name: merge when CI passes and 1 positive review  
    conditions:  
      - "#approved-reviews-by>=1"  
      - status-success=continuous-integration/travis-ci/pr  
      - base=master  
    actions:  
      merge:  
        method: squash  
        strict: true
```



The image shows a snippet of a GitHub commit history. It features three entries, each with a purple merge icon, a blue 'm' icon for the 'mergify' bot, and a 'bot' label. The first entry states 'merged commit 442ce7e into master 22 minutes ago' with a '4 checks passed' status and buttons for 'View details' and 'Revert'. The second entry states 'deleted the pyup-update-tox-3.12.1-to-3.13.2 branch 22 minutes ago' with a 'Restore branch' button. The third entry is a duplicate of the second, appearing below it.

  mergify bot merged commit **442ce7e** into **master** 22 minutes ago [View details](#) [Revert](#)
4 checks passed

  mergify bot deleted the **pyup-update-tox-3.12.1-to-3.13.2** branch 22 minutes ago [Restore branch](#)

  mergify bot deleted the **pyup-update-tox-3.12.1-to-3.13.2** branch 22 minutes ago [Restore branch](#)

[↗ mergify.io](#)

CI

Bots working for you

- PyUP bot finds updates on PyPI
- Travis CI tests your code against the new package version
- Mergify merges the PR if tests pass

Publish your project on PyPI

```
(my_venv) $ python setup.py bdist_wheel
```

```
(my_venv) $ python setup.py sdist
```

```
(my_venv) $ twine check dist/*
```

```
Checking distribution dist/my-package-0.2.2-py3-none-any.whl
```

```
Checking distribution dist/my-package-0.2.2.tar.gz
```

```
(my_venv) $ twine upload dist/*
```

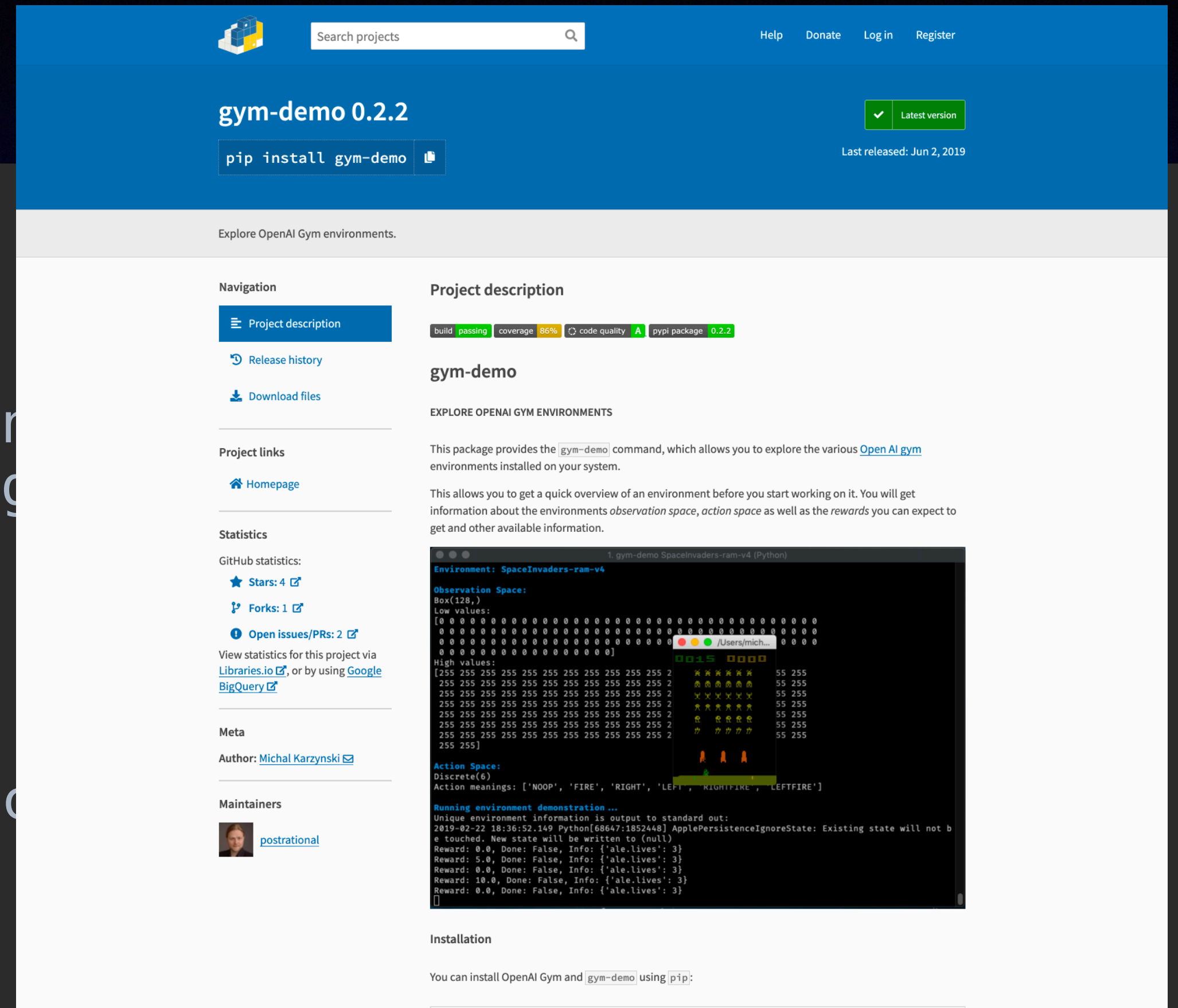
```
Enter your username: my_username
```

```
Enter your password:
```

```
Uploading distributions to https://pypi.org/legacy/
```

```
Uploading my_package-0.2.2-py3-none-any.whl
```

```
Uploading my-package-0.2.2.tar.gz
```



PyPI

wheel

twine

Publish!

More details available on my blog:

michal.karzynski.pl

THANK YOU