How to be properly lazy

Like, really lazy!

Ellert van der Velden

ADACS Astro Hack Week 2020

Code coverage

Code coverage

- “The art of convincing yourself that you are testing everything.”

Code coverage: What?

Code coverage: What?

```
---------- coverage: platform win32, python 3.6.6-final-0 ----------
Name                                Stmts   Miss  Cover   Missing
-----------------------------------------------------------------
example_scripts\__init__.py             0      0   100%
example_scripts\downsampler.py         91     69    24%   39-103, 141-172, 215-217, 222-225, 228-231, 263-273
example_scripts\galaxy.py              58     21    64%   51-52, 149, 168-169, 206-219, 245-254
-----------------------------------------------------------------
TOTAL                                 149     90    40%
```

Code coverage: Why?

Code coverage: Why?

- Write near-exhaustive tests;

Code coverage: Why?

- Write near-exhaustive tests;
- Check for code redundancy;

Code coverage: Why?

- Write near-exhaustive tests;
- Check for code redundancy;
- Find non-covered code;

Code coverage: Why?

- Write near-exhaustive tests;
- Check for code redundancy;
- Find non-covered code;
- Special test-cases help in the future.

Code coverage: How?

Code coverage: How?

- Aim for 100% coverage, including branch coverage;

Code coverage: How?

- Aim for 100% coverage, including branch coverage;

```
main_code()
if A or B:
    do_action()
main_code_continued()
```

Code coverage: How?

- Aim for 100% coverage, including branch coverage;
- If that is not possible, ask yourself why (maybe use *pragma: no cover*);

Code coverage: How?

- Aim for 100% coverage, including branch coverage;
- If that is not possible, ask yourself why (maybe use *pragma: no cover*);

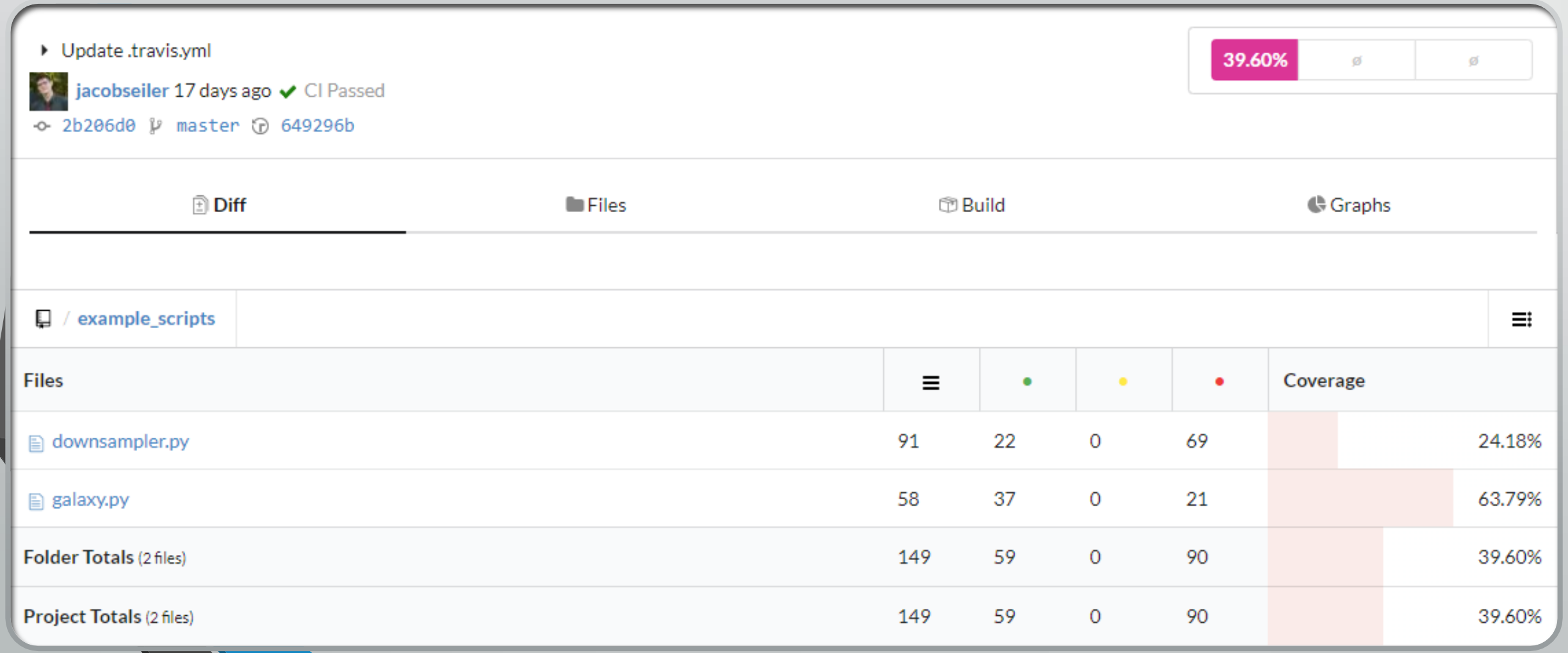
```
main_code()
if flag:  # pragma: no cover
    do_action()
    do_another_action()
main_code_continued()
```

Code coverage: How?

- Aim for 100% coverage, including branch coverage;
- If that is not possible, ask yourself why (maybe use *pragma: no cover*);
- Make sure to write a single test for a single coverage case (e.g., do not cover multiple exception cases in the same test).

Code coverage: Where?

Code coverage: Where?



Update .travis.yml

jacobseiler 17 days ago ✔ CI Passed

2b206d0 master 649296b

39.60% ø ø

Diff Files Build Graphs

/ example_scripts

Files	≡	●	●	●	Coverage
downsampler.py	91	22	0	69	24.18%
galaxy.py	58	37	0	21	63.79%
Folder Totals (2 files)	149	59	0	90	39.60%
Project Totals (2 files)	149	59	0	90	39.60%

Code coverage: Where?

Code coverage: Where?

- Once you have your coverage reports, you can upload them to CodeCov (https://codecov.io);

Code coverage: Where?

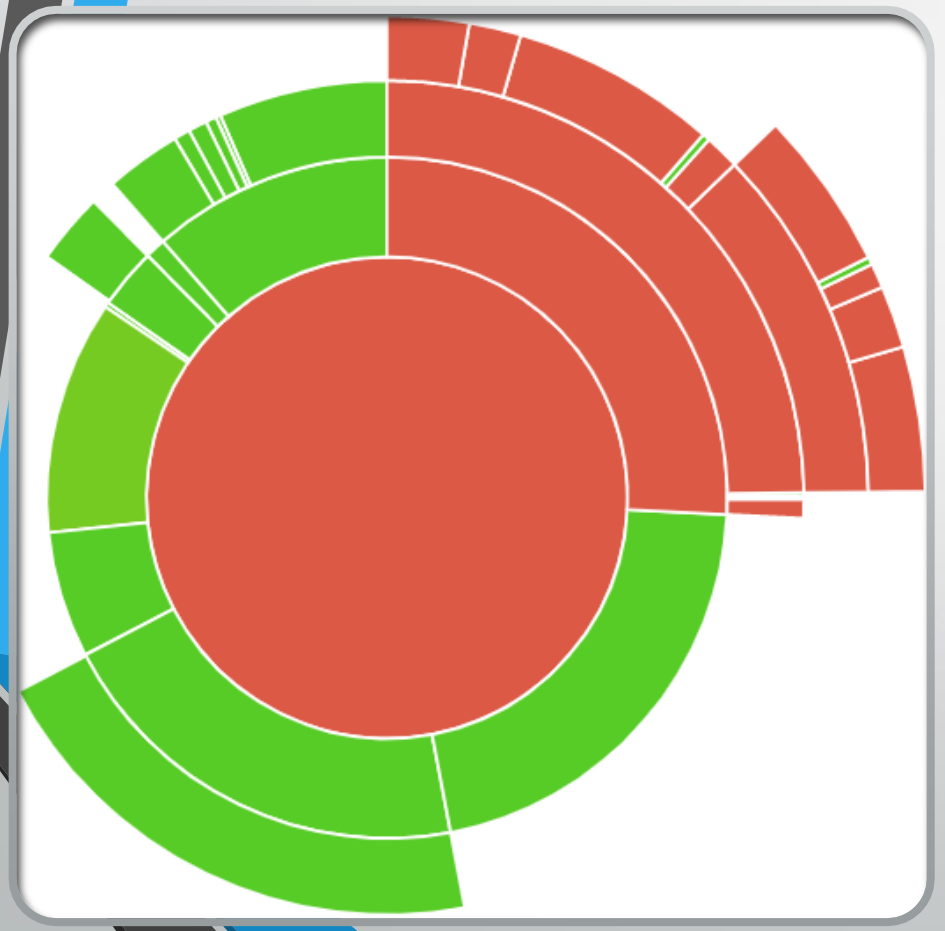
- Once you have your coverage reports, you can upload them to CodeCov (https://codecov.io);
- CodeCov keeps track of your code coverage;

Code coverage: Where?

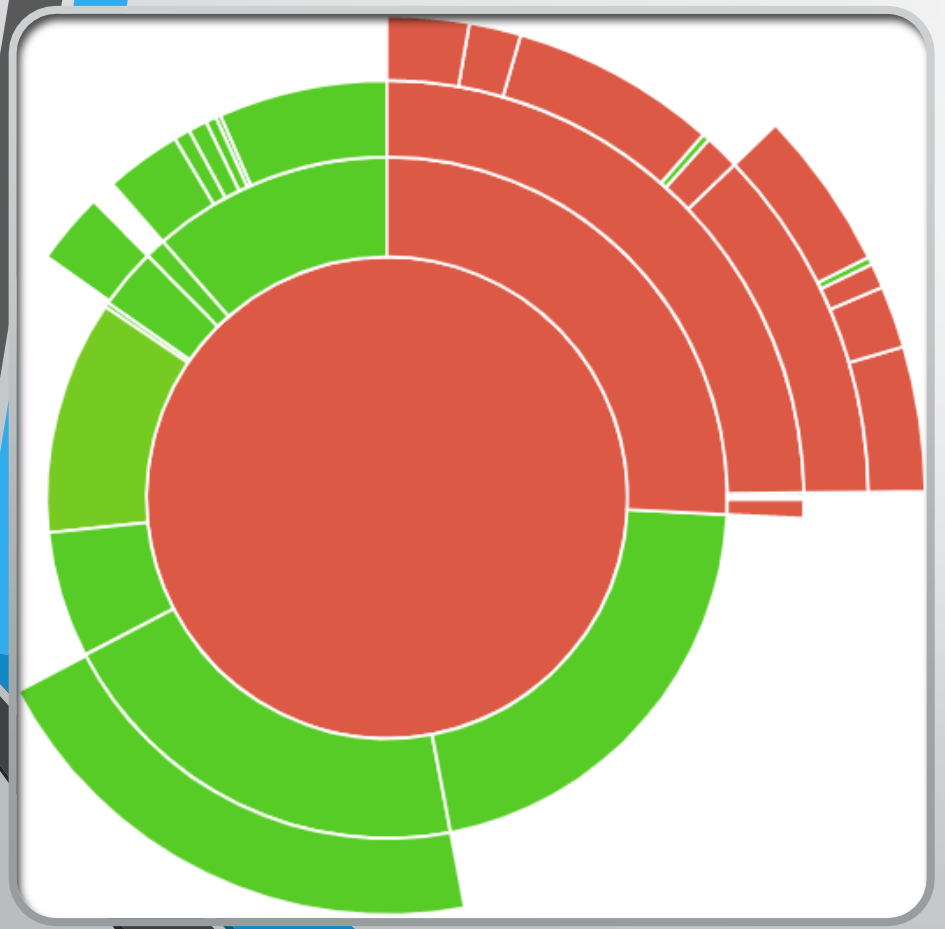
- Once you have your coverage reports, you can upload them to CodeCov (https://codecov.io);
- CodeCov keeps track of your code coverage;
- It can also provide commit status messages.

Code coverage: Where?

Code coverage: Where?



Code coverage: Where?



+115 +24 +91

[WIP #27] Added a details option that gives the details overview of an emulator iteration. 85.16% 20.93% -2.55%

1313e 12 days ago dev a9c4a4d CI Passed

+119 +8 +111

[WIP #27] Fixed that the options menu sometimes did not open in the center of the GUI. 87.40% 9.68% ø

1313e 13 days ago dev 04f1ac8

+16 +1 +15

[WIP #27] Made some modifications to the way figures are saved. 87.71% 6.78% -0.51%

1313e 14 days ago dev 60d677c CI Passed

+25 +1 +24

[WIP #27] More improvements to the Projection GUI. 88.21% 20.15% -2.90%

1313e 14 days ago dev 46ad99c CI Passed

+159 +25 +134

[WIP #27] Changed the two main parts of the GUI to be dock widgets, allowing them to be moved by the user. 91.12% 4.00% ø

1313e 14 days ago dev 5e9f1a3

+42 +2 +40

Enforce that PyQt5 is used. 92.04% 25.00% ø

Continuous Integration

Or

Let the Damn Computer Worry About It

Jacob Seiler


AUTOMATE EVERYTHING

State of the Game

- Have a repo which contains tests run using the command ‘pytest’.

State of the Game

- Have a repo which contains tests run using the command ‘pytest’.

- Ensure your code can run on your machine!

State of the Game

- Have a repo which contains tests run using the command ‘pytest’.
- Ensure your code can run on your machine!
- Make sure you run ‘pytest’ before you push a commit!

I’m Lazy and Don’t Like Thinking

- Remembering to run PyTest **every single time** before you push is an absolute nightmare.

I’m Lazy and Don’t Like Thinking

- Remembering to run PyTest **every single time** before you push is an absolute nightmare.

- What happens if you forget and your code is sitting on GitHub broken…?

I’m Lazy and Don’t Like Thinking

- Remembering to run PyTest **every single time** before you push is an absolute nightmare.

- What happens if you forget and your code is sitting on GitHub broken…?

- Also, the tests run only on your machine. What happens if someone is running Windows? Or a different Python version?

I’m Lazy and Don’t Like Thinking

- Remembering to run PyTest **every single time** before you push is an absolute nightmare.
- What happens if you forget and your code is sitting on GitHub broken…?
- Also, the tests run only on your machine. What happens if someone is running Windows? Or a different Python version?
- What if we set it up so our tests automatically run every time we push? And what if we could decide what environment to run these tests on…?

I'm Lazy and Don't [illegible]ng

- Remembering to run [illegible]ush is an absolute night[illegible]
- [illegible]s if someone
- Wh[illegible] automatically run every time we pus[illegible] decide what environment to run these test[illegible]

Continuous Integration

Current | Branches | Build History | Pull Requests | More options

✓ **dev** Use smaller badge for JOSS.

Commit `0ae646e`

Compare `8e481dd..0ae646e`

Branch dev

Ellert van der Velden

#61 passed

Ran for 5 min 4 sec

Total time 13 min 49 sec

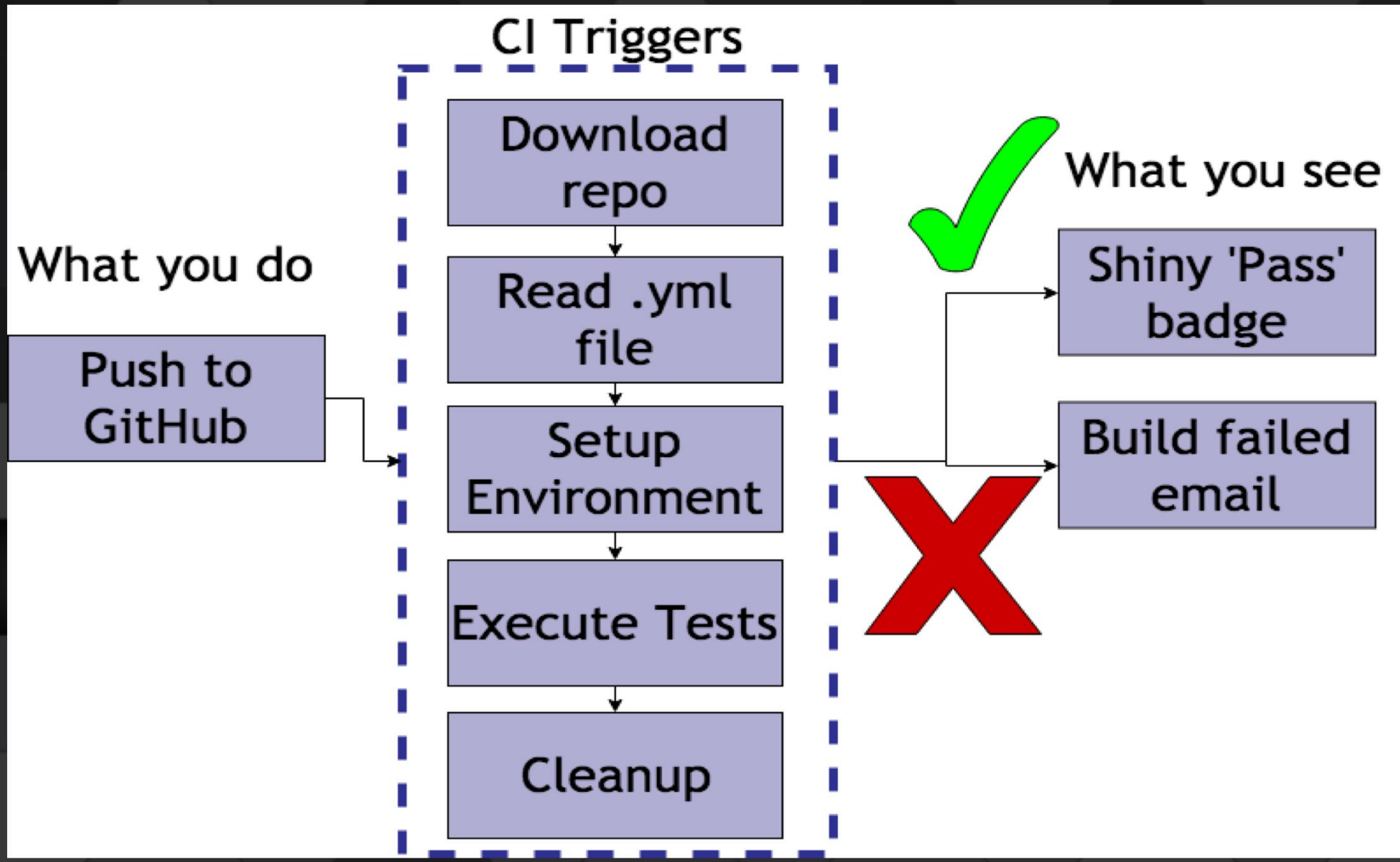
3 days ago

Restart build

Build jobs | View config

✓	# 61.1	AMD64	Python: 2.7	no environment variables set	5 min 4 sec
✓	# 61.2	AMD64	Python: 3.5	no environment variables set	1 min 20 sec
✓	# 61.3	AMD64	Python: 3.6	no environment variables set	1 min 22 sec
✓	# 61.4	AMD64	Python: 3.7	no environment variables set	1 min 12 sec
✓	# 61.5	AMD64	Python: 3.7-dev	no environment variables set	1 min 18 sec
✓	# 61.6	AMD64	Python: 3.8	no environment variables set	1 min 22 sec
✓	# 61.7	AMD64	Python: 3.8-dev	no environment variables set	1 min 30 sec

Continuous Integration



The .travis.yml File

The .travis.yml File

The .travis.yml File

- What Python versions or OS should be tested?

The .travis.yml File

- What Python versions or OS should be tested?
- What should Travis do once it's cloned your repo?

The .travis.yml File

- What Python versions or OS should be tested?
- What should Travis do once it's cloned your repo?
- What happens if the tests pass/fail?

The .travis.yml File

- What Python versions or OS should be tested?
- What should Travis do once it's cloned your repo?
- What happens if the tests pass/fail?
- What happens after your tests have been executed?

The .trav

- What Python ve ed?
- What should Tra r repo?
- What happens i
- What happens a executed?

```
language: python
dist: xenial
os: linux

env:
  global:
    - TWINE_USERNAME=1313e
    - secure: "i8vL+y1qXy37yXkaDx7gdJIgCRo+5ltmJIj3UZ76W
+mGlXKp8lbWrOc3UI+79ABdKg6CgiVrKLaLan/+Oim8jUXa05XrV3qVw
+MItJqJWsZaJNCSAuk7FkeNNOl0wfX3tEsZ9Q0YtN4DAke95LFVLtmbb
+T3qKRFWOWrn/a6ZdQ/5fTtETDpMrp0198AjyhS0L+wBOVnIXq+bclgm
+5vJ3gNyEDJmjDzdkkdlMqvGK8X8Ig+QWP9Wphl+4i6d9M41ohsH
+ZgaEigUkLKNnwbHXQKteYZ1gFzKhFvoikwqpY8F7Oi6adyuMPzpBXRw
+5JR3SpTRtgcXRiym/do5dKwE5StUybXbpzcNA/8FmZYzhb1A="

jobs:
  include:
    - python: 2.7
    - python: 3.5
    - python: 3.6
    - python: 3.7
    - python: 3.7-dev
    - python: 3.8
    - python: 3.8-dev
    - python: 3.9-dev
  allow_failures:
    - python: 3.9-dev

before_install:
  - python -m pip install --upgrade pip setuptools wheel
  - pip install -r requirements_dev.txt
install:
  - check-manifest
  - python setup.py sdist bdist_wheel
  - twine check dist/*
script:
  - pytest
```

Your Turn!

- Log into Travis CI and enable builds for your fork of the workshop repo (https://github.com/1313e/software-dev).

Your Turn!

- Log into Travis CI and enable builds for your fork of the workshop repo (https://github.com/1313e/software-dev).

- Take the template .yml file and adjust it to be ‘correct’.

Your Turn!

- Log into Travis CI and enable builds for your fork of the workshop repo (https://github.com/1313e/software-dev).

- Take the template .yml file and adjust it to be ‘correct’.

- Track the .yml file. Commit. Push!

Your Turn!

- Log into Travis CI and enable builds for your fork of the workshop repo (https://github.com/1313e/software-dev).

- Take the template .yml file and adjust it to be ‘correct’.

- Track the .yml file. Commit. Push!

- Did your tests pass?
  - No? Change the .yml file and commit/push until it does!
  - Yes? Add the shiny ‘pass’ badge to the README!

Your Turn!

- Log into Travis CI and enable builds for your fork of the workshop repo (https://github.com/1313e/software-dev).

- Take the template .yml file and adjust it to be ‘correct’.

- Track the .yml file. Commit. Push!

- Did your tests pass?
  - No? Change the .yml file and commit/push until it does!
  - Yes? Add the shiny ‘pass’ badge to the README!

- All done? Work on getting ‘codecov’ to automatically run and update your results to ‘codecov.io’. Then get the codecov badge!