NOKIA

Reviewing code using Codecov

LINCS Python Academy

Quentin Lutz

15-01-2020

Code reviewing

- Pretty much **any review** not provided by the author of some code. It aims at:
  - Finding and correcting errors/flaws
  - Proposing better solutions
  - Improving readability/maintainability of the code
  - Increasing some sense of responsibility

- Very **broad term**:
  - Your IDE reviews your code
  - Git users review your code

- Here, our goal will be to make code reviewing easier.

Context

- Say you are **developing a package** (in pretty much any major language or even multiple languages at once).
- You have written an **adequate test suite** and are satisfied with it.
- However, as your package grows, your code may become increasingly **difficult to test** accurately.
- In particular, **how do you assess how relevant your tests are?**

Code coverage a.k.a. Line count

- **Counting and marking lines** of codes that are read during tests is a possible indicator of test quality.

- Pros:
  - You can easily know what parts of your code are left out.
  - The count metric is easily increased.

- Cons:
  - A high line count is no guarantee for relevant tests.

Codecov

- Codecov does not compute this metric, it formats externally-generated coverage reports in a user-friendly way.
- There are many alternatives to Codecov that do the same thing.
- Many Python packages use Codecov.

scikit-learn

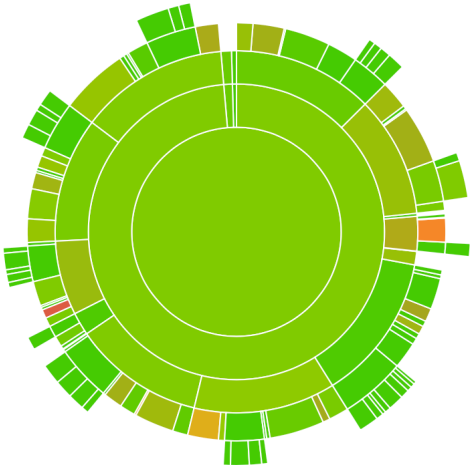
Welcome to the GitHub repository of CherryPy!

Codecov features

- Seamless integration with CI tools
- Automatic merging of all the build reports for each commit
- Marketing arguments:
  - A new badge

  - Synthetic graphics



Codecov features

- All the **major languages are supported**, multiple languages can be used at once.
- Automatic notifications can be set up for quick pull requests evaluations.

Codecov workflow

On your machine:

- Run your tests using a coverage tool (for Python: coverage). This generates a coverage report.
- Upload this report to codecov.io .

Using a CI service:

- Run your tests using a coverage tool.
- Let codecov upload each build’s report, gather them and merge them.

Codecov setup

Setup depends on your CI and language. See https://docs.codecov.io/docs/quick-start

If you created a package using F. Durand's My Toy Package, setup is as follows:

- In your .travis.yml:

```
install:
    - pip install -U tox-travis
    - pip install codecov
```

```
after_success:
  - codecov
```

- In your tox.ini:

```
commands =
    pip install -U pip
    py.test --basetemp={envtmpdir} --doctest-modules --cov-report=xml --cov=your_projects_name
```

- In your requirements_dev.txt:

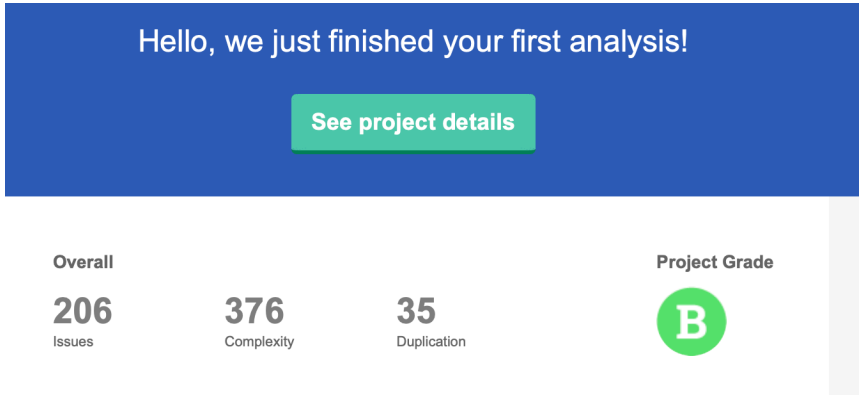
```
pytest==4.6.5
pytest-runner==5.1
pytest-cov==2.8.1
```

Codecov interface

- Go to scikit-network's Codecov and Travis

Comparison with another service: Codacy

- Codacy needs no additional code in the package.
- It provides IDE-like remarks on the code but also a few higher-level mostly security-related advice (SQL injections, weak keys, …).
- Attempts to aggregate more metrics to give a more educated guess of the package's quality.
- It lacks a (free) local API.



NOKIA

Takeaway

Codecov has advantages...

- ... as regards the developer:
  - Overview of the code coverage
  - Easy integration
  - Incentive for covering tests

- ... as regards the package presentation:
  - Badge

However, it gives no strong guarantee about code functionality.

References

- Bacchelli, A; Bird, C (May 2013). "Expectations, outcomes, and challenges of modern code review" . *Proceedings of the 35th IEEE/ACM International Conference On Software Engineering (ICSE 2013).*

- https://codecov.io

- http://codacy.com