Code Coverage

Yegor Bugayenko

Lecture #15 out of 24

80 minutes

Example, Part I

Live Code:

```
1  int fibonacci(int n) {
2    if (n <= 0) {
3      return 0;
4    }
5    if (n <= 2) {
6      return 1;
7    }
8    return fibonacci(n-1)
9      + fibonacci(n-2);
10 }
```

Test Code:

```
1  assert fibonacci(1) == 1;
2  assert fibonacci(2) == 1;
```

$$C = 3/10 = 30\%$$

Example, Part I

Live Code:
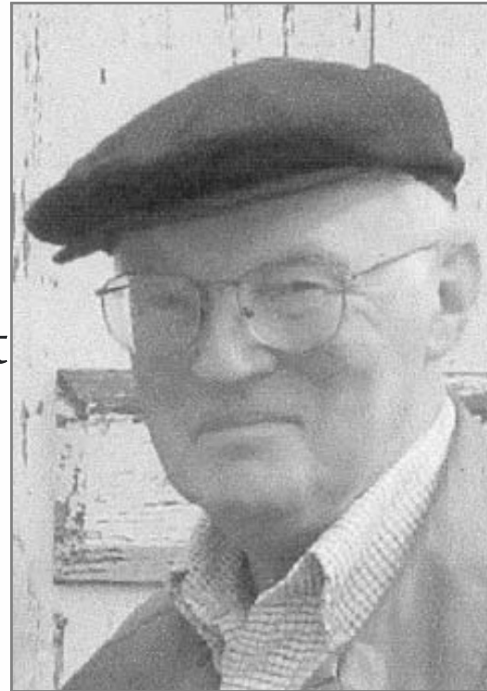
```
1  int fibonacci(int n) {
2    if (n <= 0) {
3      return 0;
4    }
5    if (n <= 2) {
6      return 1;
7    }
8    return fibonacci(n-1)
9      + fibonacci(n-2);
10 }
```

Test Code:

```
1  assert fibonacci(1) == 1;
2  assert fibonacci(2) == 1;
3
4  assert fibonacci(9) == 34;
5  assert fibonacci(10) == 55;
```

$$C = 5/10 = 50\%$$

+2pt

"A disciplined test control process is composed of five steps: 1) establish the intended extent of testing; 2) create a list of functional variations eligible for testing; 3) rank and subset the eligible variations so that test resources can be directed at those with the higher payoff; 4) calculate the test coverage of the test case library; and 5) verify attainment of the planned test coverage."

— William Robert Elmendorf, *Controlling the Functional Testing of an Operating System*, IEEE Transactions on Systems Science and Cybernetics, 5(4), 1969

+2pt

"Coverage numbers (like many numbers) are dangerous because they're <u>objective</u> but <u>incomplete</u>. They too often distort sensible action. Using them in isolation is as foolish as hiring based only on GPA."

— Brian Marick, *How to Misuse Code Coverage*, 1997

+2pt

"Our results show that coverage has an insignificant correlation with the number of bugs that are found after the release of the software at the project level, and no such correlation at the file level."

— *Code Coverage and Postrelease Defects: A Large-Scale Study on Open Source Projects*, Pavneet Singh Kochhar, David Lo, Julia Lawall, Nachiappan Nagappan, IEEE Transactions on Reliability, 66(4), 2017

+2pt

"I would be suspicious of anything like 100% — it would smell of someone writing tests to make the coverage numbers happy, but not thinking about what they are doing."

— Martin Fowler, *Test Coverage*, 1997

+2pt

"Google does not enforce any code coverage thresholds across the entire codebase. Projects (or groups of projects) are free to define their own thresholds and goals. Many projects opt-into a centralized voluntary alerting system that defines five levels of code coverage thresholds."

— *Code Coverage at Google*, Goran Petrović, Marko Ivanković, René Just, Gordon Fraser, Proceedings of the 27th Joint Meeting on ESEC/FSE, 2019

Code Coverage Threshold Levels in Google

Table 2: Coverage levels and corresponding thresholds. Many projects voluntarily set these thresholds as their goal.

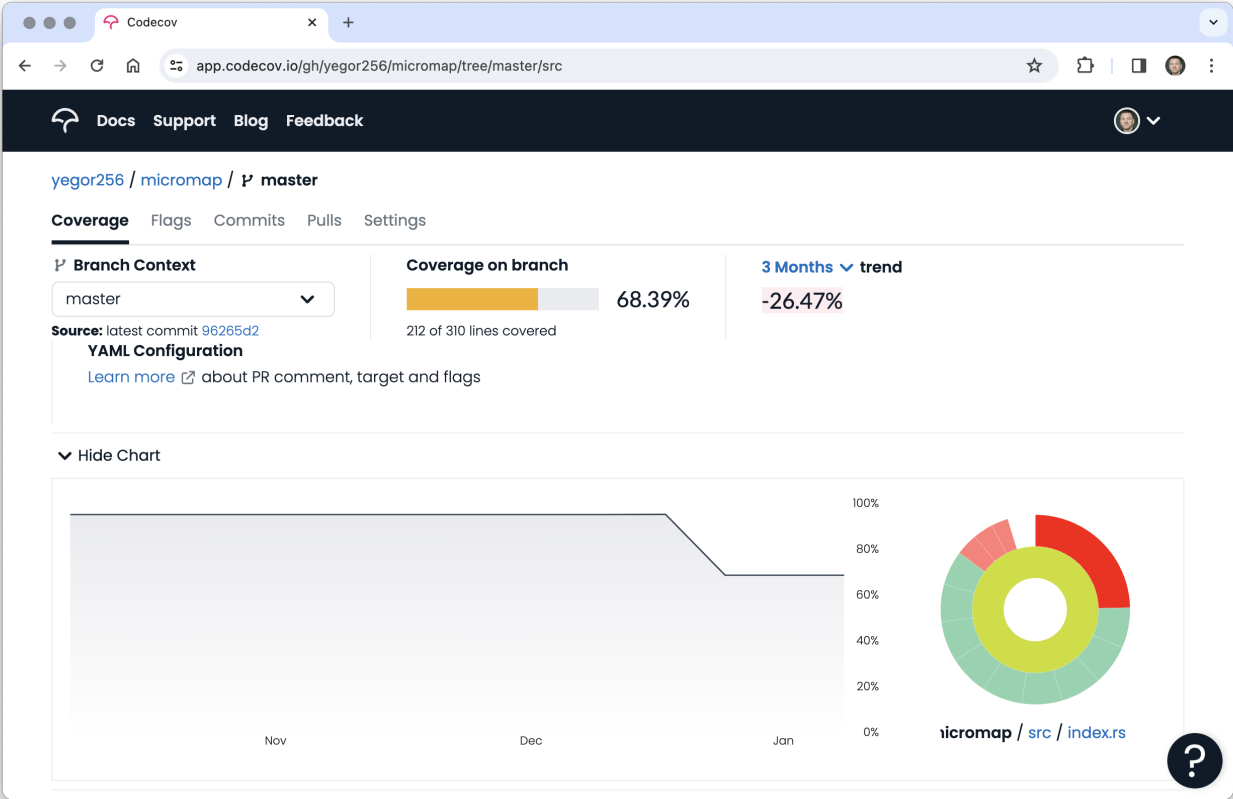| Level	Threshold
Level 1	Coverage automation disabled
Level 2	Coverage automation enabled
Level 3	Project coverage at least 60%; Changelist coverage at least 70%
Level 4	Project coverage at least 75%; Changelist coverage at least 80%
Level 5	Project coverage at least 90%; Changelist coverage at least 90%

+2pt

"Code coverage does not guarantee that the covered lines or branches have been tested correctly, it just guarantees that they have been executed by a test. But a low code coverage number does guarantee that large areas of the product are going completely untested by automation on every single deployment."

— *Code Coverage Best Practices*, Carlos Arguelles, Marko Ivanković, Adam Bender, Google Blog, 2020

Codecov.io

Line Coverage

Tarpaulin for Rust

Code | Blame   23 lines (23 loc) · 551 Bytes   Raw

```yaml
1   ---
2   name: tarpaulin
3   on:
4     push:
5       branches:
6         - master
7   jobs:
8     tarpaulin:
9       runs-on: ubuntu-22.04
10      steps:
11        - uses: actions/checkout@v4
12        - uses: actions-rs/toolchain@v1
13          with:
14            toolchain: stable
15            override: true
16        - uses: actions-rs/tarpaulin@v0.1
17          with:
18            version: '0.22.0'
19            args: '--all-features --exclude-files src/lib.rs -- --test-threads 1'
20        - uses: codecov/codecov-action@v3
21          with:
22            token: ${{ secrets.CODECOV_TOKEN }}
23            fail_ci_if_error: true
```

Code Coverage can be calculated by a few tools:

- JaCoCo for Java

- Istanbul for Javascript

- Gcov for C/C++

- Coverage.py for Python

- Simplecov for Ruby

- Tarpaulin for Rust

Read this:

Code Coverage Best Practices, Carlos Arguelles, Marko Ivanković, Adam Bender, Google Blog, 2020