Test Driven Development & Code Coverage

Marta Julia González Padrón - alu0101254487
Vanessa Valentina Villalba Pérez - alu0101265704

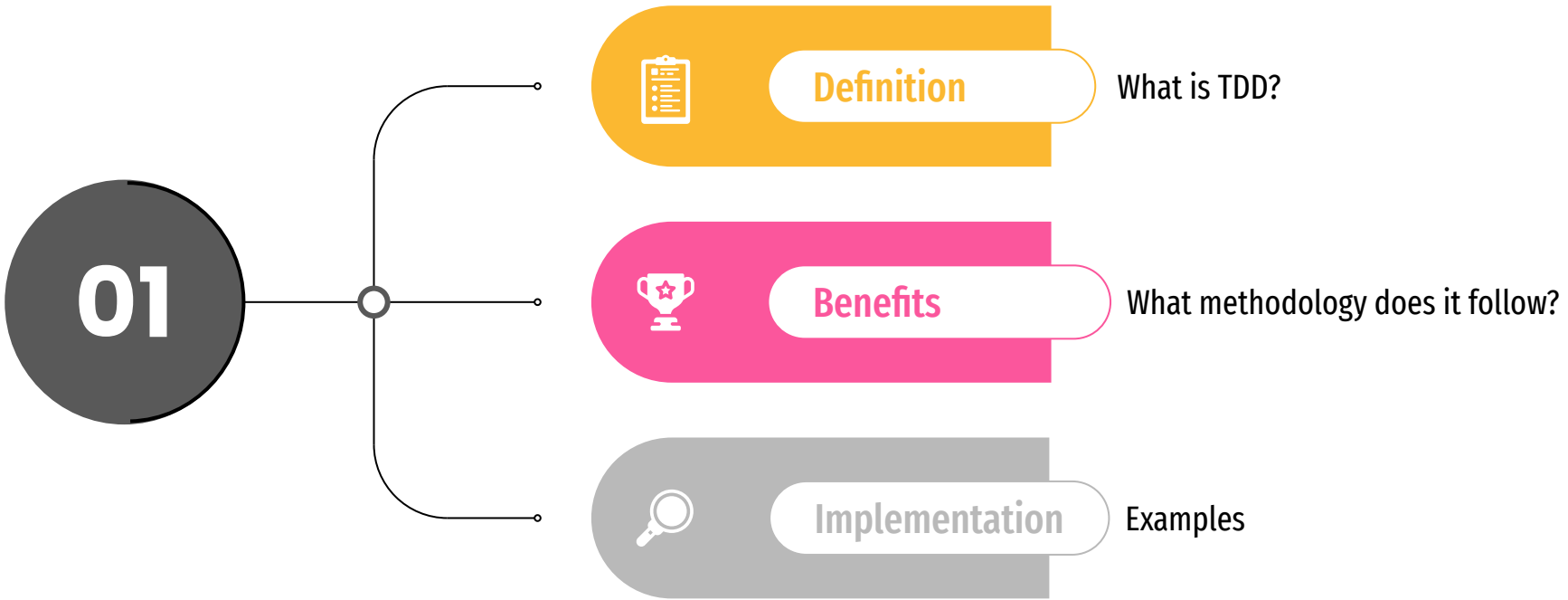
Table Of Contents

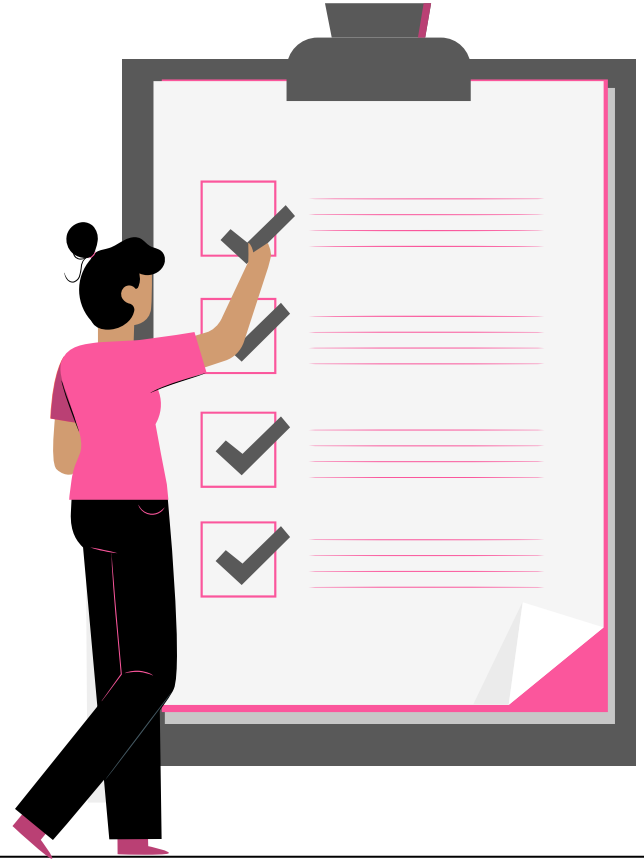
01 Test Driven Development

02 Code Coverage

03 CodeCov

Test Driven Development

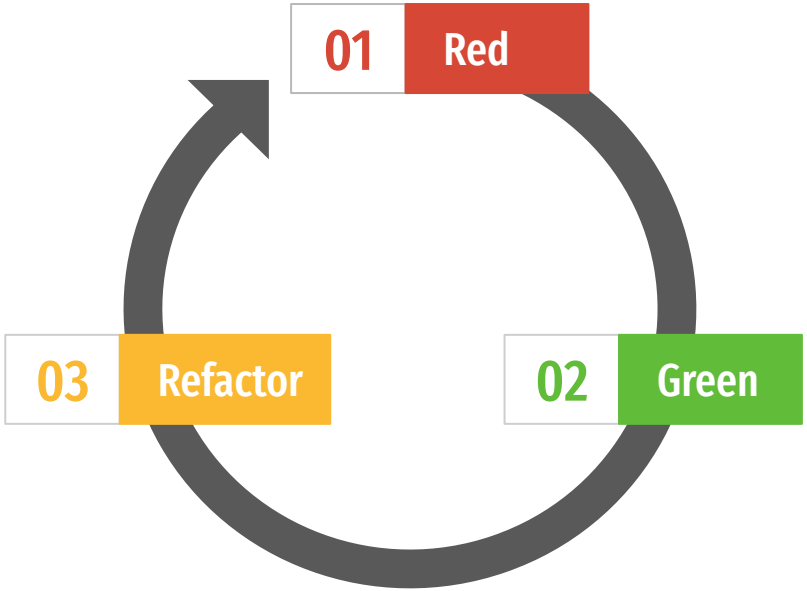




What is TDD?

Software development process which combines test-first development and refactoring

Methodology



Implementation

01 Red

```
describe('Yacht', () => {
  it('Yacht', () => {
    expect(score([5, 5, 5, 5, 5], 'yacht')).to.equal(50);
  });

  it('Not Yacht', () => {
    expect(score([1, 3, 3, 2, 5], 'yacht')).to.equal(0);
  });
});
```

```
Yacht
    1) Yacht
    2) Not Yacht


  0 passing (13ms)
  2 failing

  1) Yacht
       Yacht:
     AssertionError: expected undefined to equal 50
      at Context.<anonymous> (test/yatch.spec.js:9:48)
      at processImmediate (internal/timers.js:461:21)

  2) Yacht
       Not Yacht:
     AssertionError: expected undefined to equal 0
      at Context.<anonymous> (test/yatch.spec.js:13:48)
      at processImmediate (internal/timers.js:461:21)
```

OBJ

Implementation

02 Green

```
const calculatePointsYacht = function(points) {
  let repetitions = 0;
  for (const element of points) {
    if (element === points[0]) {
      repetitions++;
    }
  }
  if (repetitions === 5) {
    return 50;
  }
  return 0;
};
```

```
> mocha test/yatch.spec.js


  Yacht
    ✓ Yacht
    ✓ Not Yacht


  2 passing (9ms)
```

Implementation

03 Refactor

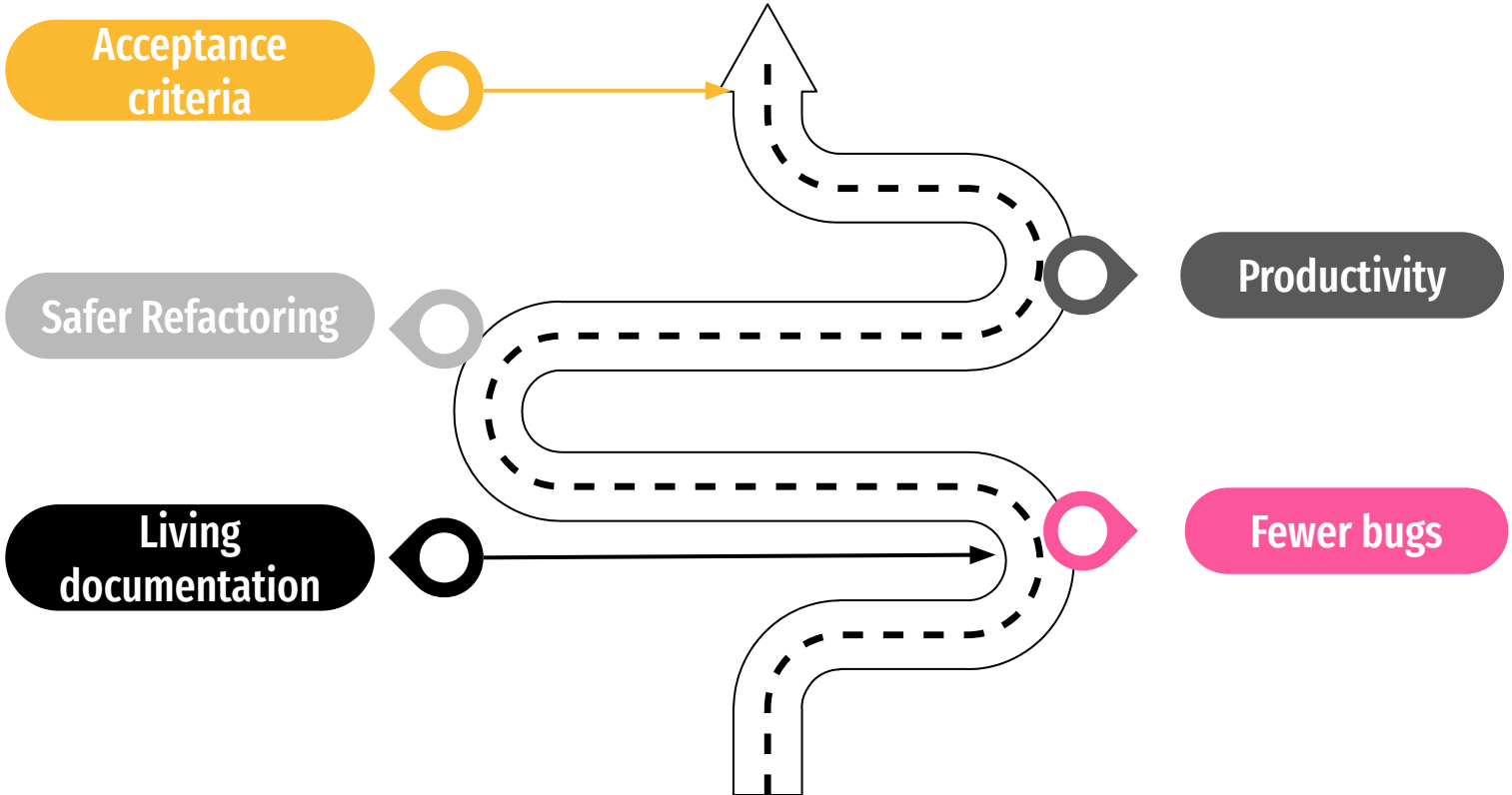
```
const calculatePointsYacht = function(points) {
  if (points.every((score) => score === points[0])) {
    return 50;
  }
  return 0
};
```

```
> mocha test/yatch.spec.js


  Yacht
    ✓ Yacht
    ✓ Not Yacht


  2 passing (10ms)
```

Benefits


Acceptance criteria
Safer Refactoring
Living documentation
Productivity
Fewer bugs

Code Coverage

02

Definition

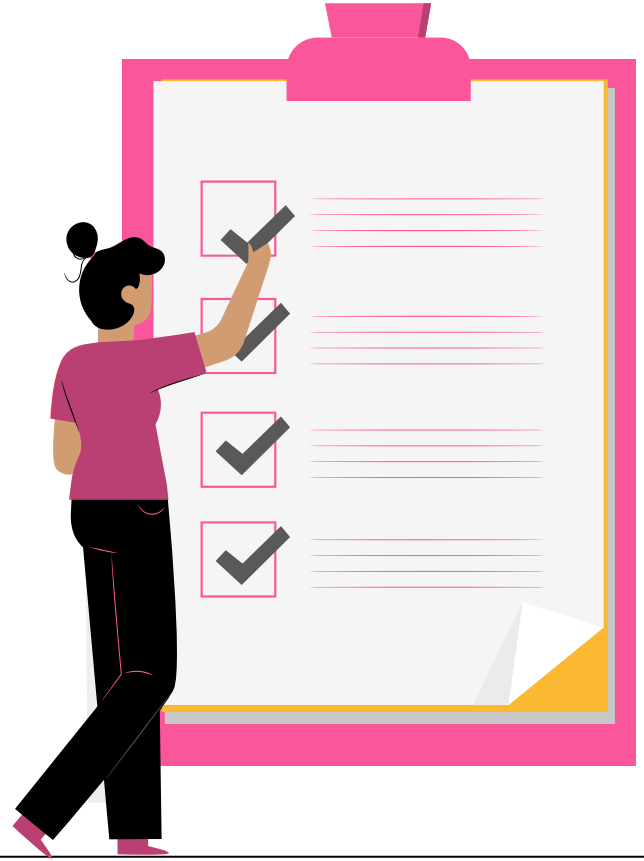
What is code coverage?

Benefits

What does code coverage provide?

Criteria

Which criteria does code coverage have?



What is it?

Percentage of the code that has been tested.

Benefits

Missed paths

Untested code

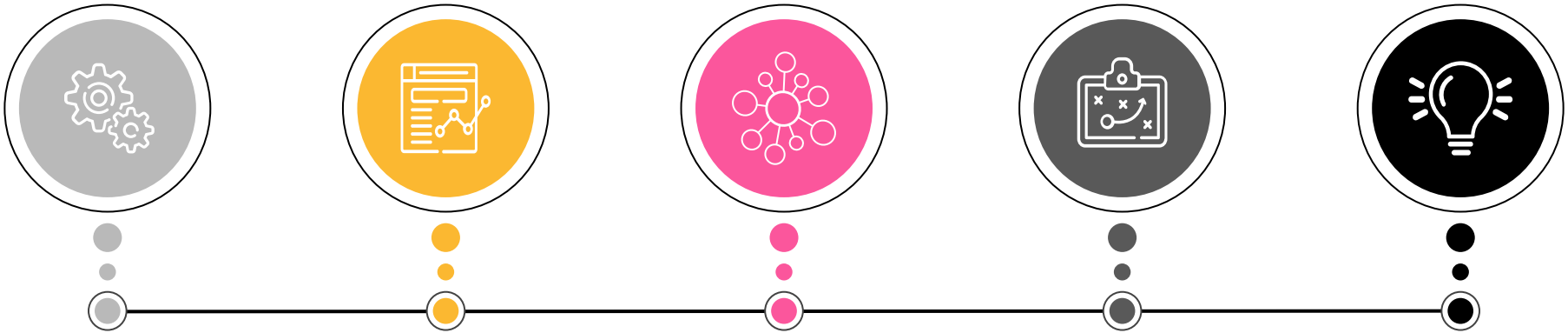
Reliable code

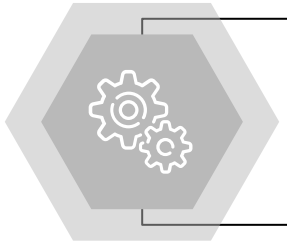
Add tests

Refactor

Confidence

Coverage Criteria





Function Coverage

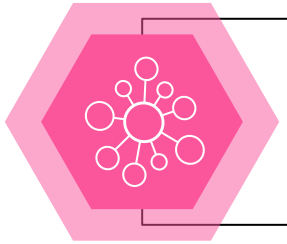
```
const uselessFunction = function(x, y) {
    let z = 0;
    if ((x > 0) || (y > 0)) {
        z = x;
    }
    return z;
};
```

```
it('Function is being called', () => {
   expect(uselessFunction(1, 1)).to.eql(1);
});
```

Statement Coverage

```
const uselessFunction = function(x, y) {
    let z = 0;
    if ((x > 0) || (y > 0)) {
        z = x;
    }
    return z;
};
```

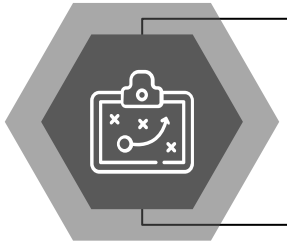
```
it('A statement is not being executed', () => {
   expect(uselessFunction(-1, -1)).to.eql(0);
});
```



Edge Coverage

```
const uselessFunction = function(x, y) {
    let z = 0;
    if ((x > 0) || (y > 0)) {
        z = x;
    }
    return z;
};
```

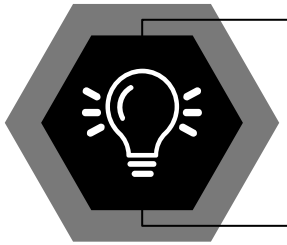
```
it('A path is not being executed', () => {
   expect(uselessFunction(-1, 1)).to.eql(1);
});
```



Branch Coverage

```
const uselessFunction = function(x, y) {
    let z = 0;
    if ((x > 0) || (y > 0)) {
        z = x;
    }
    return z;
};
```

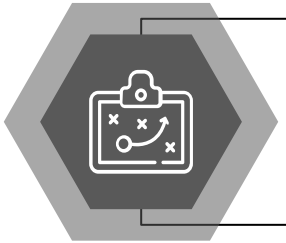
```
it('A branch is not being executed', () => {
   expect(uselessFunction(1, 1)).to.eql(1);
});
```



Condition Coverage

```
const uselessFunction = function(x, y) {
    let z = 0;
    if ((x > 0) || (y > 0)) {
        z = x;
    }
    return z;
};
```

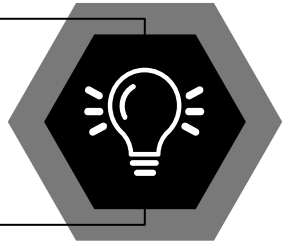
```
it('Boolean sub-expressions evaluated both to true and false', () => {
   expect(uselessFunction(-1, 1)).to.eql(1);
   expect(uselessFunction(1, -1)).to.eql(1);
});
```



Branch Coverage

VS

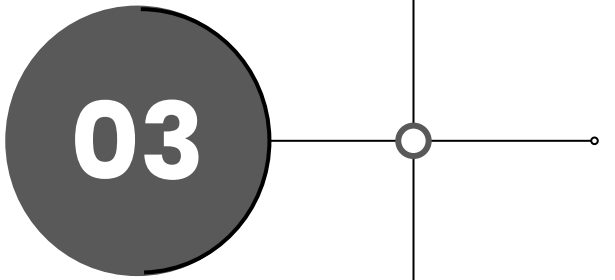
Condition Coverage



```
const anotherUselessFunction = function(a, b) {
  if ((a > 0) && (b > 0)) {
      ...
    }
};
```

```
it('Condition satisfied but Branch unsatisfied', () => {
   expect(anotherUselessFunction(-1, 1)).to.eql(undefined);
   expect(anotherUselessFunction(1, -1)).to.eql(undefined);
});
```

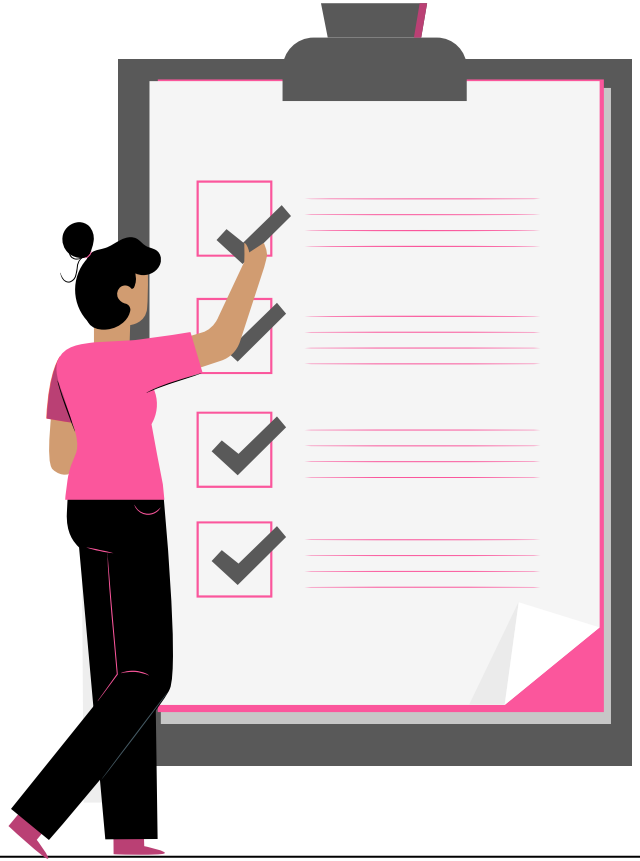
CodeCov



What is CodeCov.io?

What does this tool provide?

What do we need?



What is CodeCov?

Tool used to measure the test coverage of a codebase.

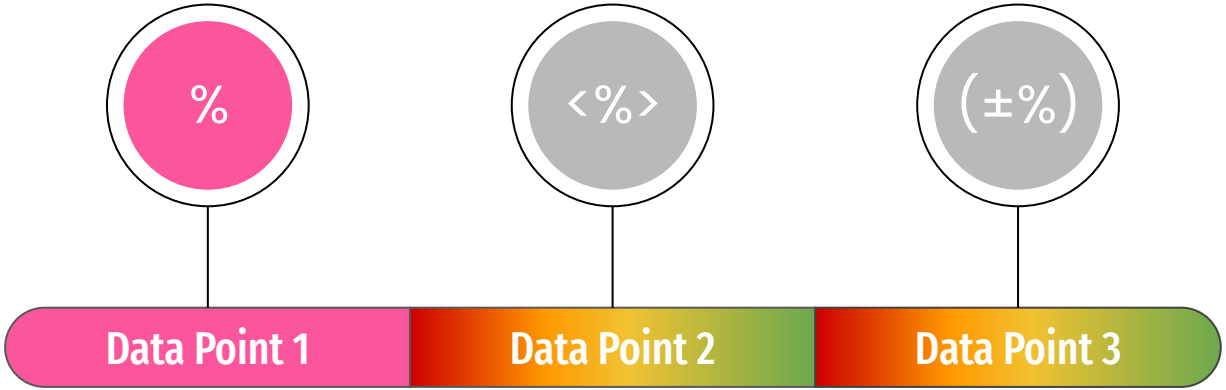
Codecov

Coverage increase after 30 days

(for repositories starting below 70% coverage)

*Based off of open source usage at Codecov

CodeCov Delta



Absolute Scope

The entire project's coverage.

Relative Scope

Coverage concerning only lines adjusted in the commit diff.

Change in coverage

The amount of gain or loss coverage when compared to the commit's parent.

Viewing Source Code

Line covered and executed

Branches that do not have all their paths covered or executed

Line not covered nor executed

Line was covered before the current commit but it is now partially hit

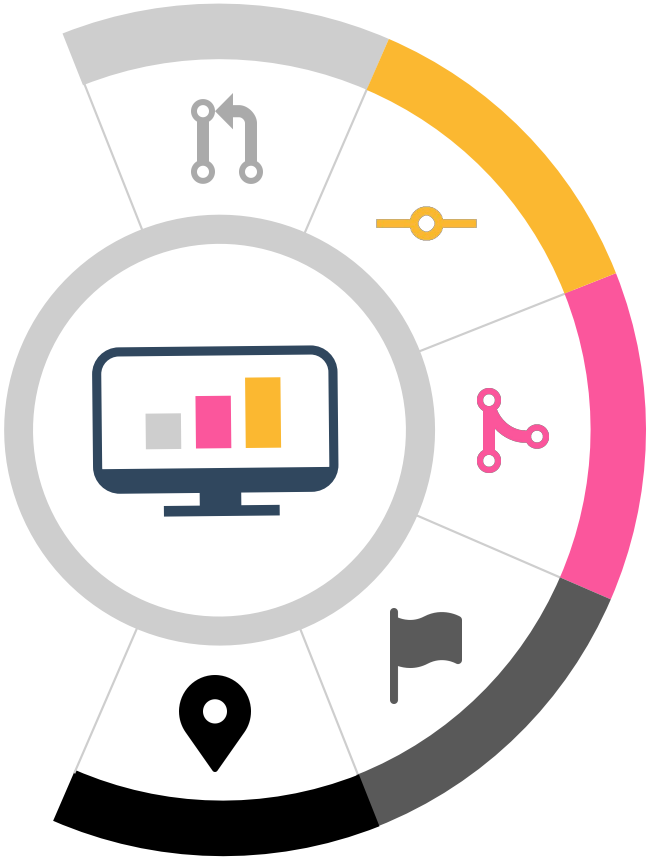
Line was covered before the current commit but it is now missed

Coverage ratio

hits / (sum of hit + partial + miss)

“Codecov delivers or "injects" coverage metrics directly into the modern workflow to promote more code coverage, especially in pull requests where new features and bug fixes commonly occur.”

Top 5 Features



01

Pull requests comments

02

Commit status

03

Merging reports

04

Flags

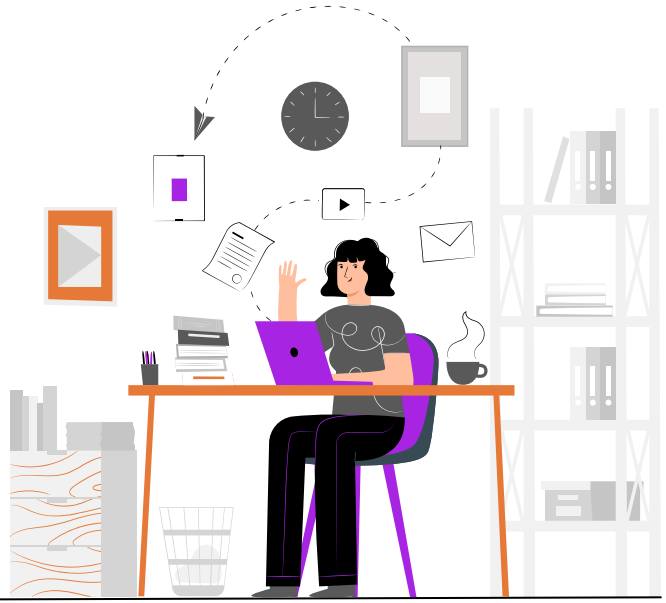
$ bash <(curl https://codecov.io/bash) -f path

05

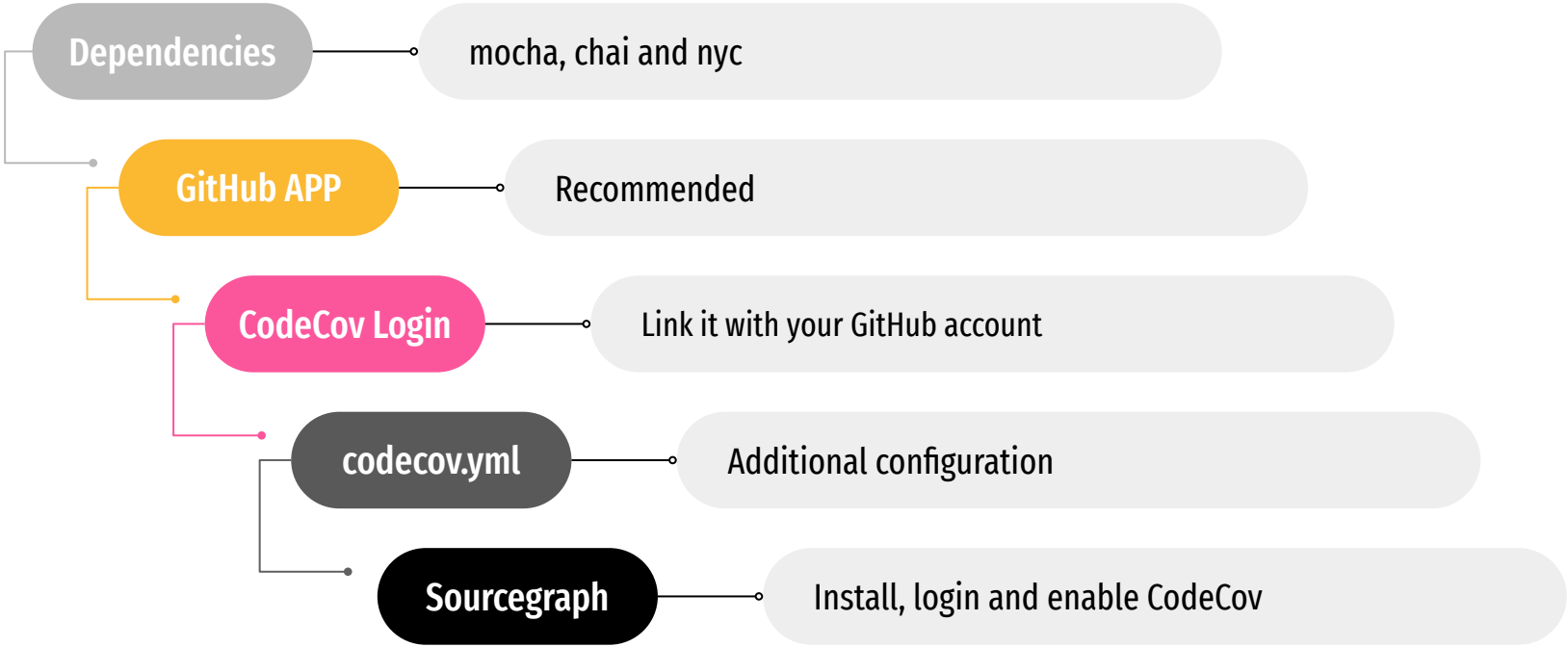
Path Manipulation

Sourcegraph

Overlay reports directly in the user's code host of choice



Quick Start


Dependencies
mocha, chai and nyc
GitHub APP
Recommended
CodeCov Login
Link it with your GitHub account
codecov.yml
Additional configuration
Sourcegraph
Install, login and enable CodeCov

Live Demonstration

Steps to follow

Package.json — Dependencies and script

```
"scripts": {
   "test": "nyc mocha",
   "validate": "cat codecov.yml | curl --data-binary @- https://codecov.io/validate",
   "report": "nyc --reporter=json mocha",
   "codecov": "curl -s https://codecov.io/bash | bash"
 },
 "devDependencies": {
   "chai": "^4.3.4",
   "mocha": "^8.3.2",
   "nyc": "^15.1.0"
 }
```

Steps to follow

codecov.yml

Token and partial lines

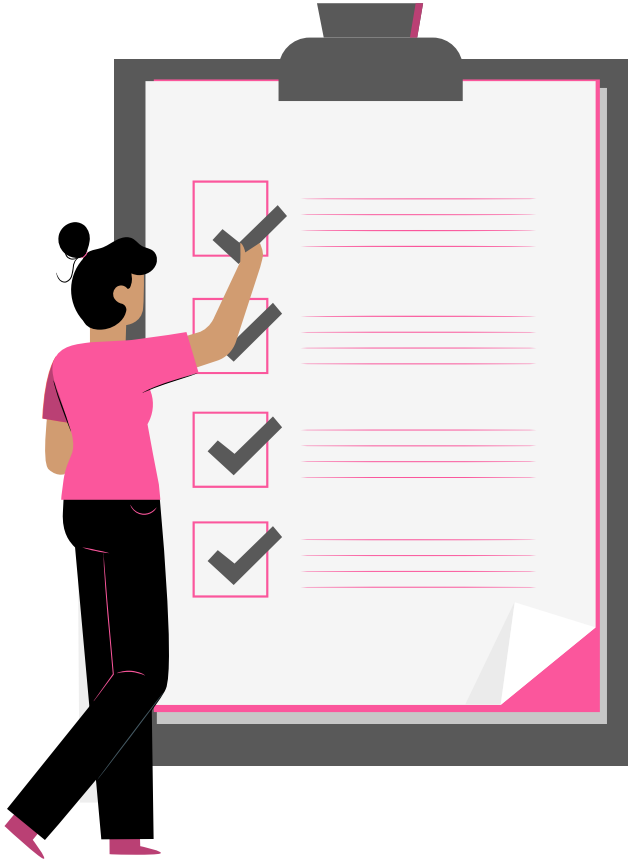
```
codecov:
  token: "_____________________________________"
  bot: "codecov-io"

parsers:
  javascript:
    enable_partials: yes
```

Steps to follow

Upload reports

Always repeat this workflow



Conclusions

- Get used to the TDD methodology.
- Improve your code coverage.
- Always report your test coverage.
- Practice makes perfect.

Bibliography

Test Driven Development

- Introduction to Test Driven Development
- Learn Test Driven Development
- Introduction to Test Driven Development in JavaScript
- Refactoring JavaScript
- 9 Benefits of Test Driven Development
- Testing - Mocha and Chai presentation repository. Adal Diaz, Oscar Moreira

Code Coverage

- Code Coverage
- Everything you need to know about Code Coverage
- Chai Assertion Library Documentation
- Istanbul Test Coverage Tool Documentation
- Mocha Documentation

CodeCov.io

- CodeCov Documentation
- Quick Start
- Codecov Github APP
- 2020 State of Open Source Coverage
- 13 CodeCov alternatives

SourceGraph

- CodeCov Sourcegraph extension
- CodeCov's additional information

FreePik

Slidesgo

Thanks for your attention

Any Questions?

Vanessa Valentina Villalba Pérez
Email
GitHub

Marta Julia González Padrón
Email
GitHub