NODE.JS NIGHTS

STRV

TESTING & CI/CD

Martin Galajda, Backend Developer at STRV

STRV

Lecture outline

- Testing
  - What, why and how?
- Continuous integration/continuous delivery (CI/CD)
- Practical example on setting up few tests
- Deploying API using Travis CI

Software testing

- Making sure the software meets specified requirements
  - Functional
  - Non-functional (i.e. technical)
- Essential part of software development process

Testing your API

Manual

- Repetitive work
- Doesn’t support CI/CD
- Doesn’t scale

Automated

- Supports CI/CD
- Scales well
- Can serve as documentation
- Essential in any real production app
- Allows refactoring

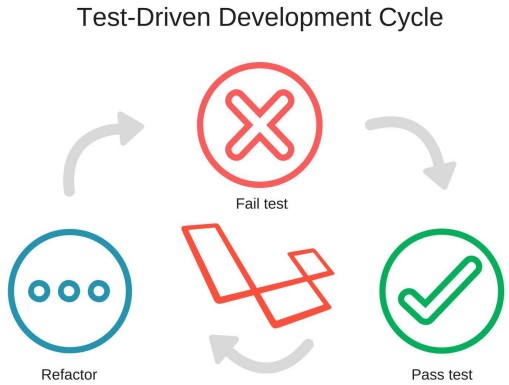
Testing approaches

Test driven development

- Start with failing test
- Make test pass
- Iterate

Behavior driven development

- Focus on testing behavior
- Don’t care about implementation details



Tests

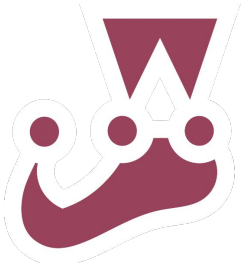
- **Unit**
  - **Verify functionality of one unit of code**
  - Usually function (or class in object oriented world)
- **Integration**
  - **Verify that our units of code work together**
  - Interaction between different components (modules)
  - API testing

- *End to end tests*
- *Performance, penetration tests*
- *Smoke tests, and many more...*

Most popular test frameworks in Node.js

Jest

- Created by Facebook
- Running tests in parallel
- Snapshot testing



Mocha

- More mature
- Stable
- Very flexible

Code coverage

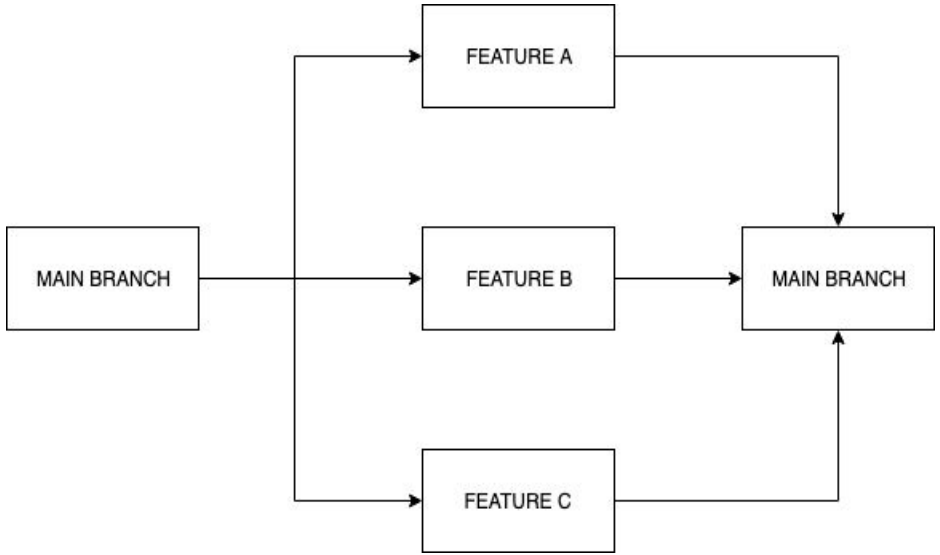
- Measure for how much is your code tested
  - Line coverage - Percentage of lines executed
  - Statement coverage - Percentage of statements executed
  - Function coverage - Percentage of functions called
  - Branch coverage - Percentage of all possible branches executed
- Can be integrated into CI
- **Rule of thumb: 90 % coverage**
- Note: Even 100% test coverage does not ensure that code is bug-free

CI & CD

- Software engineering practices
- Continuous integration
  - Developers integrate code in shared repository
  - Automated build & tests (to verify integration)
- Continuous delivery
  - Small build cycle with short sprint
  - Main branch deployable at any given time
- Continuous deployment
  - Code automatically deployed to production
- Many tools to help with CI/CD
  - **Travis**, Drone, Circle, GitHub Actions, GitLab ...

CI - Guidelines

1. Merge features into the main branch
2. Feature from the main branch
3. Done when confirmed on the main branch
4. Test both feature and the main branch



Tools that we will use

Tests

- Mocha - test framework
- Nyc - code coverage
- Chai - assertion library
- Sinon - stubbing/mocking dependencies

CI / Deployment

- Travis
- Heroku

QUESTIONS

HOMEWORK

STRV

STRV