Red Green Refactor

Aleksander Zubala
@alekzubala

Red Green Refactor

- TDD Mantra
- Development in short, repeatable cycles
- Constantly forming hypotheses and checking them
- Gives confidence to refactor and experiment
- Improves the code quality

RED

- Think for a while what piece of code move your project towards completion
- Write a short test:
  - tested object might not exist
  - method might not be implemented
  - AppCode helps to quickly implement/create missing classes or methods
- Execute tests, check if the test is failing (IMPORTANT)

GREEN

- Write a production code in your project:
  - the previous test must pass
  - do not focus on the quality of the code
  - you can even hardcode to achieve the passing test
- Execute test, check the result of the test (IMPORTANT)
- Now you have a proof that the test is testing the right thing

REFACTOR

- Take a deep breath, all of your tests are passing:)
- Go back to the code you've just written, see what can be improved
- Don't be afraid to change the code, tests will quickly catch mistakes
- Focus on the code duplication (DRY) and 'smelly' parts
- No idea how to improve, leave it for a while, then go back
- REPEAT!

What you get?

- Baby steps - a lot of small cycles
- Most of the time spend on refactoring
- When something goes wrong, it’s easy to identify what was the cause
- You have to think about design of the code
- Once feature is done, you can move to next task

Common mistakes

- Too much time spend red/green cycle
- Attempt to achieve fully functional feature in single cycle
- Refactor stage is ignored

Demo

Q & A

Thanks!