TEST DRIVEN DEVELOPMENT

Java Clean code

Outline

Any fool can write code that *a* **computer can understand.** *Good programmers* **write code** that *humans* **can understand.**
Martin Fowler

Laws for TDD by Kent Beck

- Don't write a line of new code unless you first have a failing automated test
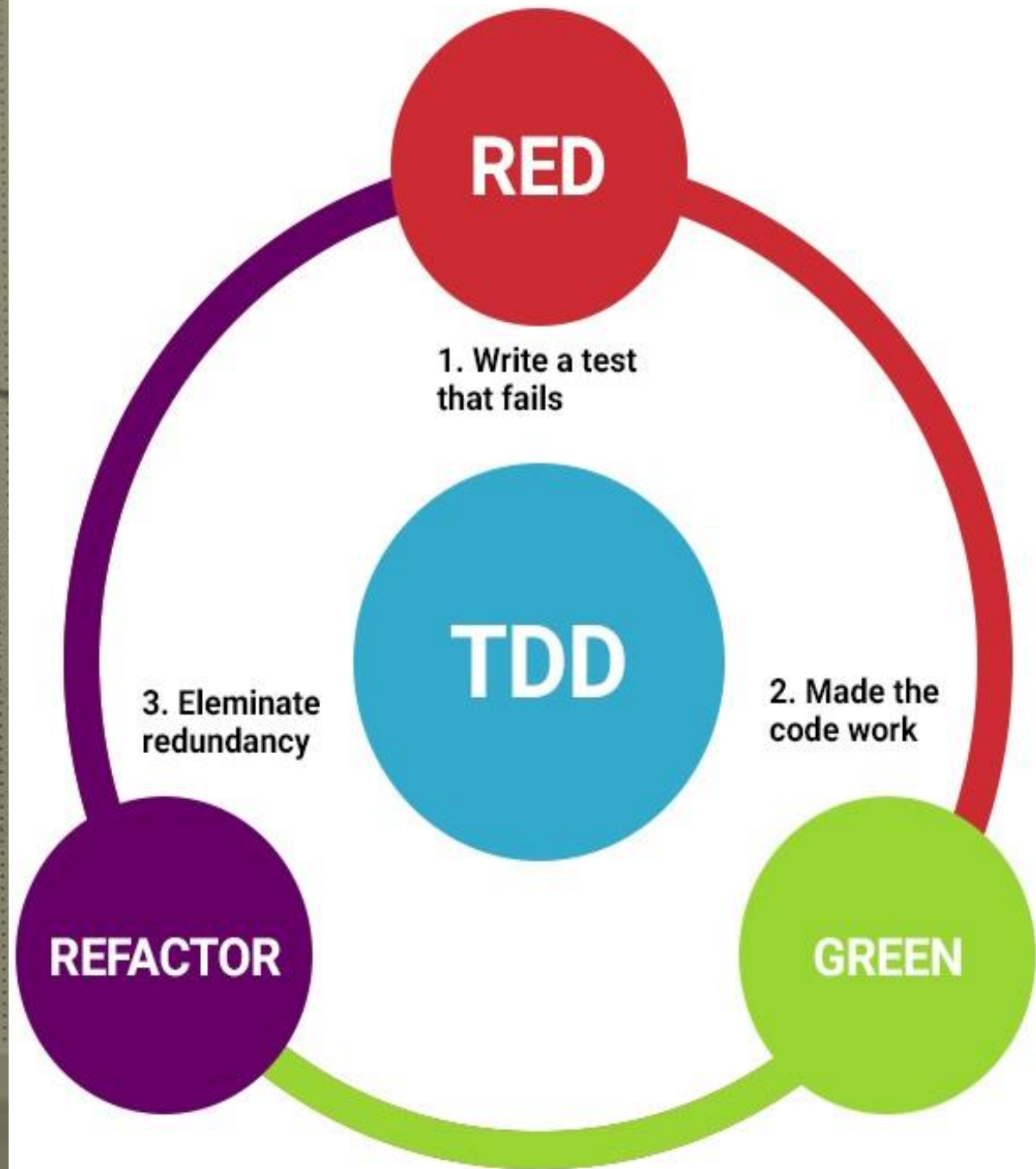
- Eliminate duplication

Laws for TDD by Uncle Bob

○ You must write a failing test before you write any production code.

○ You must not write more of a test than is sufficient to fail, or fail to compile.

○ You must not write more production code than is sufficient to make the currently failing test pass.

Laws for TDD by Martin Fowler

- Write a test for the next bit of functionality you want to add.

- Write the functional code until the test passes.

- Refactor both new and old code to make it well structured.

RED
1. Write a test that fails

TDD

3. Eleminate redundancy

2. Made the code work

REFACTOR

GREEN

Test Driven Development

- Enough to fail

- Enough to pass

- Keep going

Test Driven Development

- Debug time

- Design documents

- Code change

- Not perfect

TDD legacy code

- Catch 22 or deadlock

- QA should not find bugs

- 85% code coverage

TDD Summary

- Debug time

- Reliable documentation

- Code change

Course Progress

Lesson 11	Lesson 12	Lesson 13
Components	Architecture	Test Driven Development

FURTHER READING

- https://martinfowler.com/bliki/TestDrivenDevelopment.html
- https://blog.cleancoder.com/uncle-bob/2014/12/17/TheCyclesOfTDD.html

THANK YOU!