Test-Driven Development Practices in Java

Overview of Practices, Principles, and Tools

Mike Nolan
mnolanjr@gmail.com

Course Overview

- **Overview of Test-Driven Development and Java frameworks to help support the principles**

- **Dive into details of these frameworks**
  - Mockito
  - DBUnit
  - PowerMockito

Test-Driven Development Practices in Java

Overview of Practices, Principles, and Tools

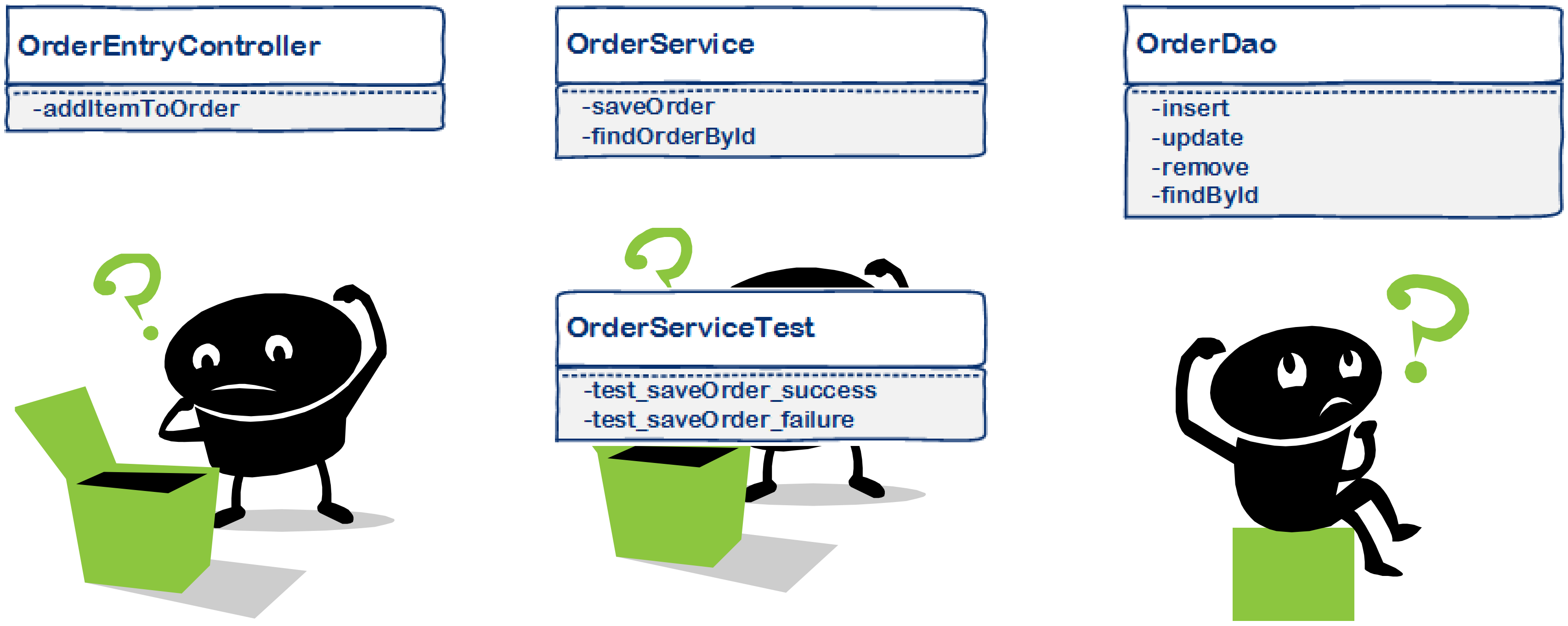
Module Overview

- Overview of Test-Driven Development
- Benefits of using this approach & common myths encountered
- Brief overview of supporting frameworks covered throughout this course

Overview of Test-Driven Development

What is Test-Driven Development (TDD)

- Development approach keeping tests one step ahead of your code
- Test-Driven Development ➔ Test-Driven Design
- Early practices established anti-patterns



Problems with the traditional approach

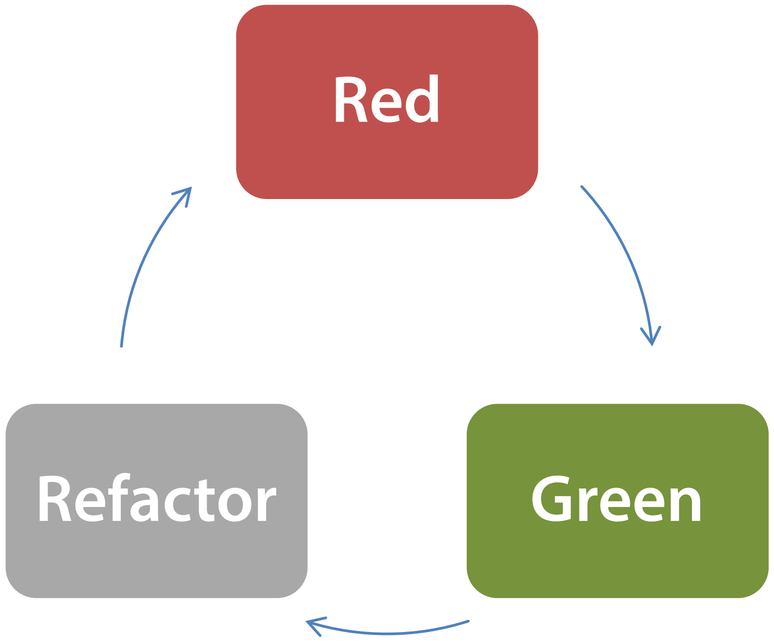
- Omission of thought in automated unit testing approach
- Benefits of automation missed
- Time crunch encountered in projects
- Mountain of work to implement the automated unit test

How you benefit

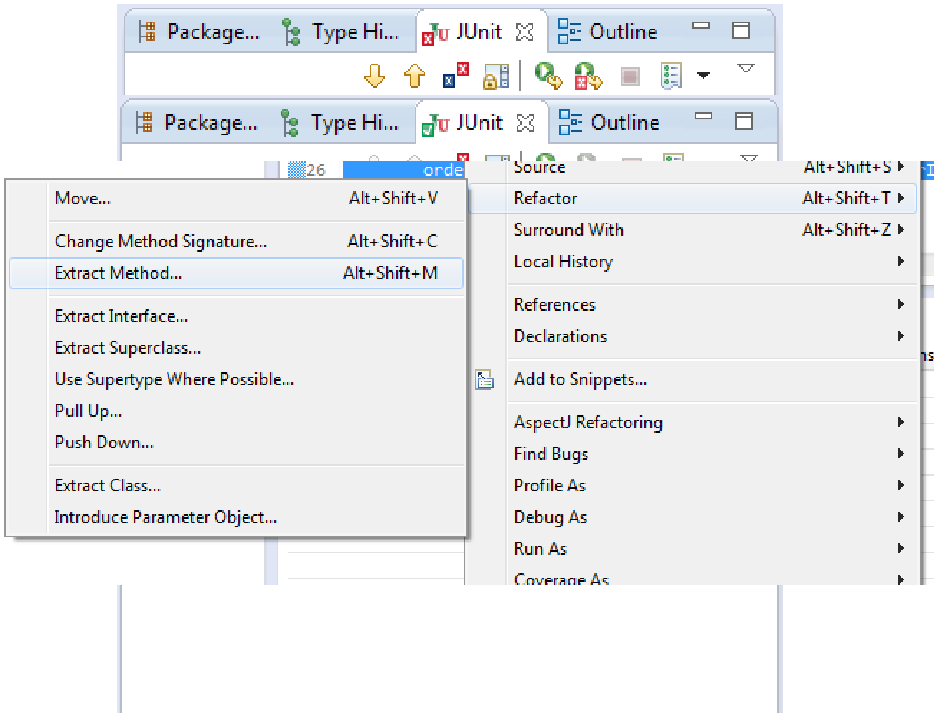
- Balance between automated testing and functional coding during development sprints
- Consistent repeatability of test execution
- Up-front design with testing in mind
- More even gauge of progress between testing and coding

Red / Green / Refactor

Red / Green / Refactor

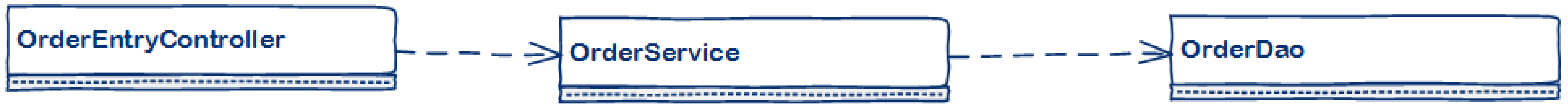


Write a test



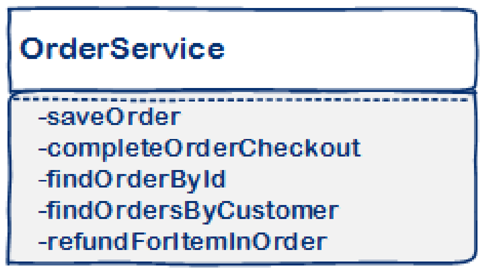
Development Task Focus

Create functional slice to add Item to an order
(focus on all layers for single story)



Vs.

Create all operations of Order Service
(focus on single layer)



Terminology

xUnit Testing

Class-Under-Test

Method-Under-Test

Test Fixture

Common myths about TDD

Myth – Coding effort increases significantly

- “Doesn’t all this testing double the development?”
- Question you should be asking – “If you are not automating your unit tests, then how are you executing the unit tests in a <u>repeatable</u> and <u>consistent</u> manner?”
- Over time, you will never manually unit test all cases in a repeatable manner with less effort than creating the test up-front; you are sacrificing quality
- The following link contains some analysis and calculations - http://c2.com/cgi/wiki?UnitTestingCostsBenefits

Myth – All tests are written before any code

- **How do I write all my tests before I’ve written a line of functional code?**

- **You don’t!**
  - Follow red/green/refactor
  - Stub out methods for important conditions
  - Add tests as bugs are discovered

Preview of supporting Java testing frameworks

Testing frameworks

- Mockito
- DBUnit
- PowerMockito

Summary

- Basic principles & benefits of Test-Driven Development
- Basic tenants of the Red/Green/Refactor approach
- Explored common myths
- Brief framework introduction