Unit Testing in Groovy

Jeremy Jarrell
www.JeremyJarrell.com
@jeremyjarrell

pluralsight
hardcore dev and IT training

Why Should We Unit Test?

Finds problems **early** in development

Preserves **key** behavior of the system

Documents the code

What Makes a Good Unit Test

- **F**ast
- **I**solated
- **R**epeatable
- **S**elf-verifying
- **T**imely

Test-driven development (TDD)

Writing your unit tests *before* you write the production code.

Red-Green-Refactor

✓

Watch the test

Fail

Red-Green-Refactor

✓

Watch the test
Fail

✗

Write just enough code
for the test to **Pass**

Red-Green-Refactor

✓

Watch the test
Fail

✗

Write just enough code
for the test to **Pass**

↻

Refactor the code

Benefits of TDD

- Results in clean APIs

Benefits of TDD

- Results in clean APIs
- Testable code is flexible code

Benefits of TDD

- Results in clean APIs
- Testable code is flexible code
- Results in much less production code

Benefits of TDD

- Results in clean APIs
- Testable code is flexible code
- Results in much less production code
- No need to schedule ‘unit-testing’ time

Disadvantages of TDD

- Increases initial development time

Disadvantages of TDD

- Increases initial development time
- Has a significant learning curve

Disadvantages of TDD

- Increases initial development time
- Has a significant learning curve
- Difficult without full team buy-in

Unit Testing Support in Groovy

JUnit is built **directly** into the Groovy runtime

Mocking APIs are **included** in the Groovy platform

Advanced testing frameworks are also available

Groovy Is Dynamic

The compiler doesn’t enforce type-checking

Unit tests become your safety net
Especially on a large team

Mock Object

A simulated object that mimics the behavior of a real object in a controlled way.

Mocks Versus Stubs

Stubs

- Can be used as stand-in objects
- Will return canned values when asked

Mocks

- Supports all the functionality of stubs
- Can verify a member was called a certain number of times
- Can assert a specific interaction occurred

Summary

- Unit testing provides a multitude of benefits to software developers

Summary

- Unit testing provides a multitude of benefits to software developers
- Functionality for unit testing is built directly into the Groovy platform

Summary

- Unit testing provides a multitude of benefits to software developers
- Functionality for unit testing is built directly into the Groovy platform
- Mocking allows you to isolate and control the dependencies of your code