Testing Your Code

Kevin Jones

@kevinrjones www.rocksolidknowledge.com

Overview

All code should have tests

Unit tests

Integration tests

Mocking

What This Is and Is Not

This is not a class on testing.

What we show is how to structure the application to support different types of tests

“A test is not a unit test if:”

“A test is not a unit test if:
It talks to the database
It communicates across the network”

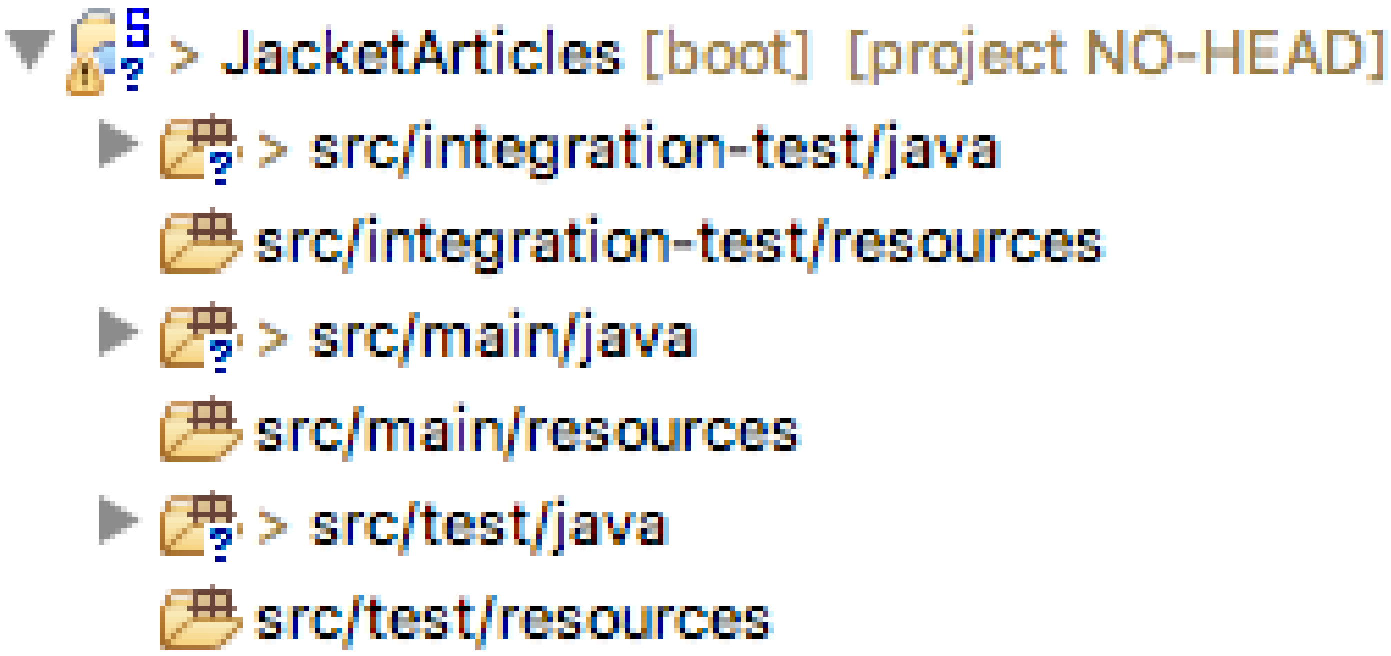
*"A test is not a unit test if:
It talks to the database
It communicates across the network
It touches the file system"*

“A test is not a unit test if:
It talks to the database
It communicates across the network
It touches the file system
It can’t run at the same time as any of your other unit tests”

> *“A test is not a unit test if:*
> *It talks to the database*
> *It communicates across the network*
> *It touches the file system*
> *It can’t run at the same time as any of your other unit tests*
> *You have to do special things to your environment (such as editing config files) to run it.”*

Michael Feathers

Code Structure



```
▼ > JacketArticles [boot] [project NO-HEAD]
    ▶ > src/integration-test/java
        src/integration-test/resources
    ▶ > src/main/java
        src/main/resources
    ▶ > src/test/java
        src/test/resources
```

Unit Testing

Isolate Units

Inject dependencies

These can then be stubbed

Use a mocking framework

Demo

Unit Testing

Integration Testing

Spring DataSource Configuration

```
public abstract class AbstractTest {

  @Bean

        @Profile("test")

        public DataSource dataSource() {}

}
```

Spring Configuration

```
@Transactional
@RunWith(SpringJUnit4ClassRunner.class)
@SpringApplicationConfiguration(classes = ApplicationConfiguration.class)
@Sql({"/insert-data.sql"})
@Sql(scripts = "/clean-up.sql", executionPhase = AFTER_TEST_METHOD)
@ActiveProfiles("test")
public abstract class AbstractTest {}
```

Spring Configuration

```
@Transactional
@RunWith(SpringJUnit4ClassRunner.class)
@SpringApplicationConfiguration(classes = ApplicationConfiguration.class)
@Sql({"/insert-data.sql"})
@Sql(scripts = "/clean-up.sql", executionPhase = AFTER_TEST_METHOD)
@ActiveProfiles("test")
public abstract class AbstractTest {}
```

Spring Configuration

```
@Transactional
@RunWith(SpringJUnit4ClassRunner.class)
@SpringApplicationConfiguration(classes = ApplicationConfiguration.class)
@Sql({"/insert-data.sql"})
@Sql(scripts = "/clean-up.sql", executionPhase = AFTER_TEST_METHOD)
@ActiveProfiles("test")
public abstract class AbstractTest {}
```

Spring Configuration

```
@Transactional
@RunWith(SpringJUnit4ClassRunner.class)
@SpringApplicationConfiguration(classes = ApplicationConfiguration.class)
@Sql({"/insert-data.sql"})
@Sql(scripts = "/clean-up.sql", executionPhase = AFTER_TEST_METHOD)
@ActiveProfiles("test")
public abstract class AbstractTest {}
```

Summary

The code works!

Well…It's tested

One last thing

Security