luv2code

Spring Boot support for Unit Testing

What do you need for Spring Boot unit testing?

- Access to the Spring Application Context
- Support for Spring dependency injection
- Retrieve data from Spring application.properties
- Mock object support for web, data, REST APIs etc ...

Unit Testing support in Spring Boot

- Spring Boot provides rich testing support

- **`@SpringBootTest`**
  - Loads the application context
  - Support for Spring dependency injection
  - You can access data from Spring application.properties
  - …

Spring Boot Starter - for Testing support

- Add Maven dependency

pom.xml

```
<dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-test</artifactId>
   <scope>test</scope>
</dependency>
```

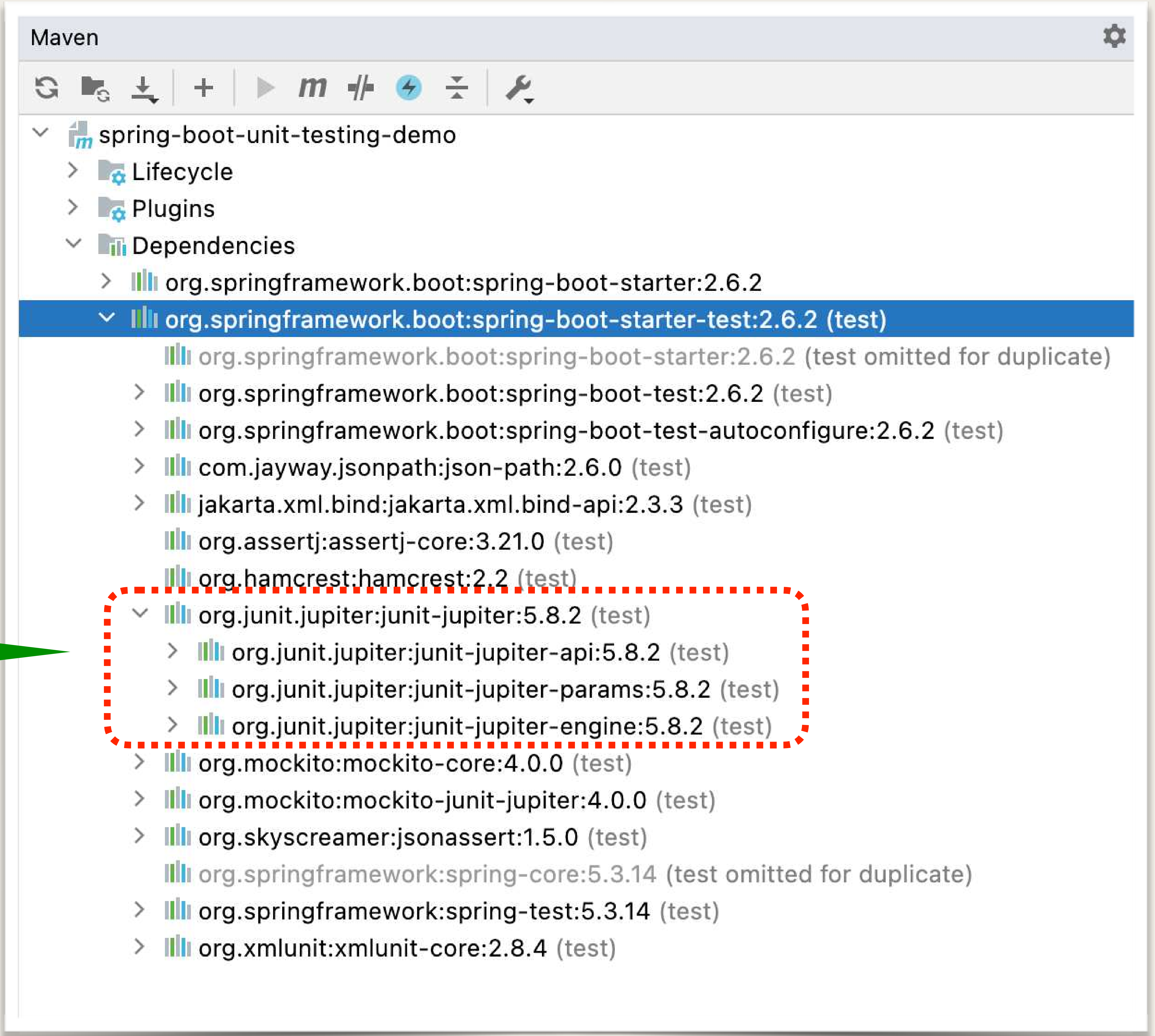
Starter includes a transitive dependency on JUnit 5

We get it for free :-)

Spring Boot Starter - Transitive Dependency for JUnit 5

pom.xml

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
</dependency>
```



```
Maven
spring-boot-unit-testing-demo
  Lifecycle
  Plugins
  Dependencies
    org.springframework.boot:spring-boot-starter:2.6.2
    org.springframework.boot:spring-boot-starter-test:2.6.2 (test)
      org.springframework.boot:spring-boot-starter:2.6.2 (test omitted for duplicate)
      org.springframework.boot:spring-boot-test:2.6.2 (test)
      org.springframework.boot:spring-boot-test-autoconfigure:2.6.2 (test)
      com.jayway.jsonpath:json-path:2.6.0 (test)
      jakarta.xml.bind:jakarta.xml.bind-api:2.3.3 (test)
      org.assertj:assertj-core:3.21.0 (test)
      org.hamcrest:hamcrest:2.2 (test)
      org.junit.jupiter:junit-jupiter:5.8.2 (test)
        org.junit.jupiter:junit-jupiter-api:5.8.2 (test)
        org.junit.jupiter:junit-jupiter-params:5.8.2 (test)
        org.junit.jupiter:junit-jupiter-engine:5.8.2 (test)
      org.mockito:mockito-core:4.0.0 (test)
      org.mockito:mockito-junit-jupiter:4.0.0 (test)
      org.skyscreamer:jsonassert:1.5.0 (test)
      org.springframework:spring-core:5.3.14 (test omitted for duplicate)
      org.springframework:spring-test:5.3.14 (test)
      org.xmlunit:xmlunit-core:2.8.4 (test)
```

Starter includes a transitive dependency on JUnit 5

We get it for free :-)

Spring Boot Starter - Transitive Dependency for JUnit 5

At command-line, type:

```
mvn dependency:tree

[INFO] Scanning for projects...
[INFO]
[INFO] -------------< com.luv2code:spring-boot-unit-testing-demo >-------------
[INFO] Building spring-boot-unit-testing-demo 1.0.0
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-dependency-plugin:3.2.0:tree (default-cli) @ spring-boot-unit-testing-demo ---
[INFO] com.luv2code:spring-boot-unit-testing-demo:jar:1.0.0
...
[INFO] \- org.springframework.boot:spring-boot-starter-test:jar:2.6.2:test
[INFO]    \- org.junit.jupiter:junit-jupiter:jar:5.8.2:test
[INFO]       +- org.junit.jupiter:junit-jupiter-api:jar:5.8.2:test
[INFO]       +- org.junit.jupiter:junit-jupiter-params:jar:5.8.2:test
[INFO]       \- org.junit.jupiter:junit-jupiter-engine:jar:5.8.2:test

...
```

Starter includes a transitive dependency on JUnit 5

We get it for free :-)

Spring Boot Test

ApplicationExampleTest.java

```java
import org.junit.jupiter.api.Test;
import org.springframework.boot.test.context.SpringBootTest;

@SpringBootTest
public class ApplicationExampleTest {

    @Test
    void basicTest() {
        // ...
    }

}
```

Loads Spring Application Context

Inject Spring Beans

```java
@Component
public class StudentGrades {

  ...

}
```

ApplicationExampleTest.java

```java
import org.junit.jupiter.api.Test;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.beans.factory.annotation.Autowired;

@SpringBootTest
public class ApplicationExampleTest {

    @Autowired
    StudentGrades studentGrades;

    @Test
    void basicTest() {
        // ...
    }
```

Injection

Access Application Properties

application.properties

```
info.school.name=luv2code
info.app.name=My Super Cool Gradebook
```

ApplicationExampleTest.java

```java
import org.junit.jupiter.api.Test;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.beans.factory.annotation.Value;

@SpringBootTest
public class ApplicationExampleTest {

    @Value("${info.school.name}")
    private String schoolName;

    @Value("${info.app.name}")
    private String appInfo;

    @Test
    void basicTest() {
        // ...
    }

}
```

Access data from application.properties

Access Application Context

ApplicationExampleTest.java

```java
import org.junit.jupiter.api.Test;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.ApplicationContext;

@SpringBootTest
public class ApplicationExampleTest {

     @Autowired
     ApplicationContext context;

     @Test
     void basicTest() {
          // ...
     }

}
```

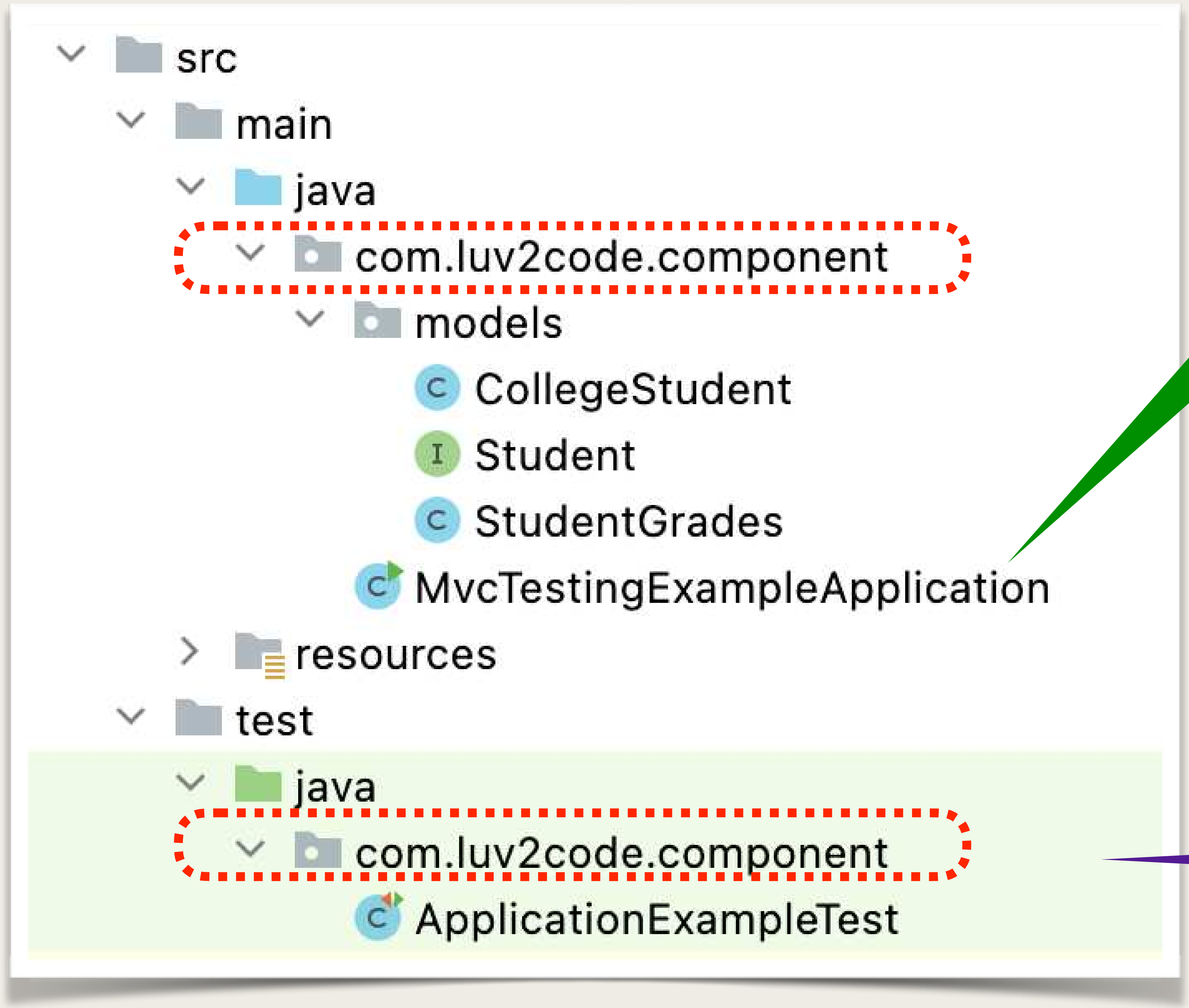
Access Spring Application Context

@SpringBootTest configuration

Place your test class in `test` package same as your `main` package

- This implicitly defines a base search
  - Allows you to leverage default configuration
  - No need to explicitly reference the main Spring Boot application class

More on Configuration



```
src
  main
    java
      com.luv2code.component
        models
          CollegeStudent
          Student
          StudentGrades
        MvcTestingExampleApplication
    resources
  test
    java
      com.luv2code.component
        ApplicationExampleTest
```

Main Spring Boot application class

Automatically component scans sub-packages

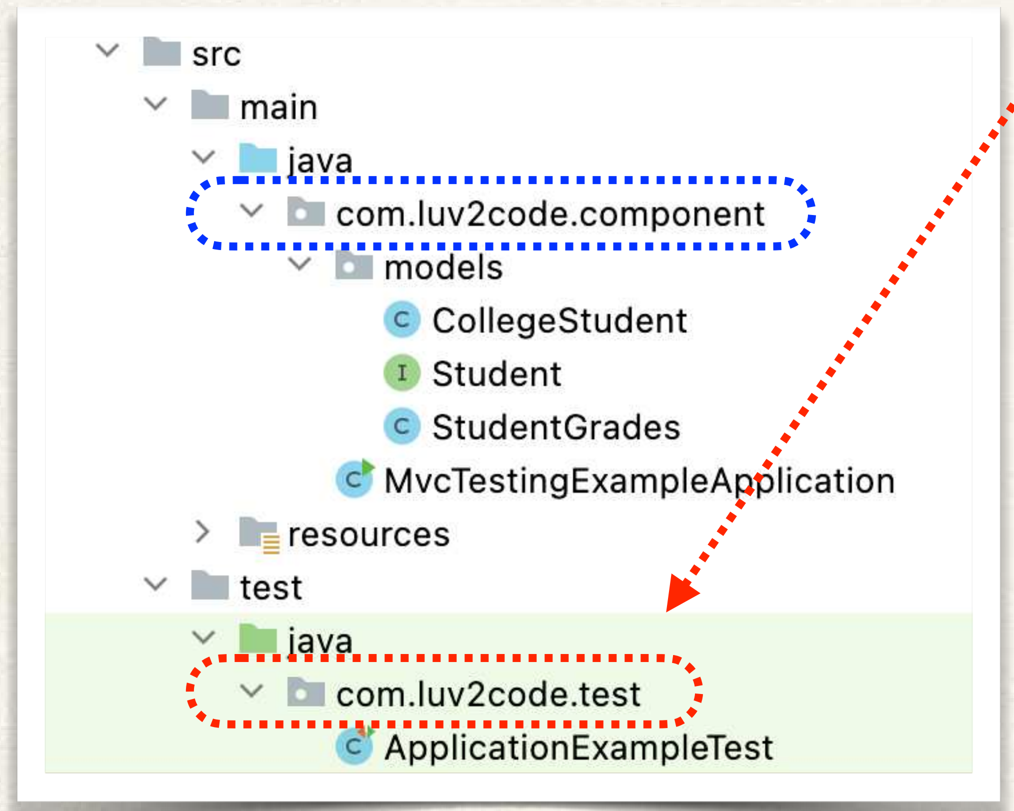
Same package names

... no additional config required ...

More on Configuration

- Default configuration is fine if everything is under
  - **`com.luv2code.component`**

- But what if test class is in a different package???
  - **`com.luv2code.test`**



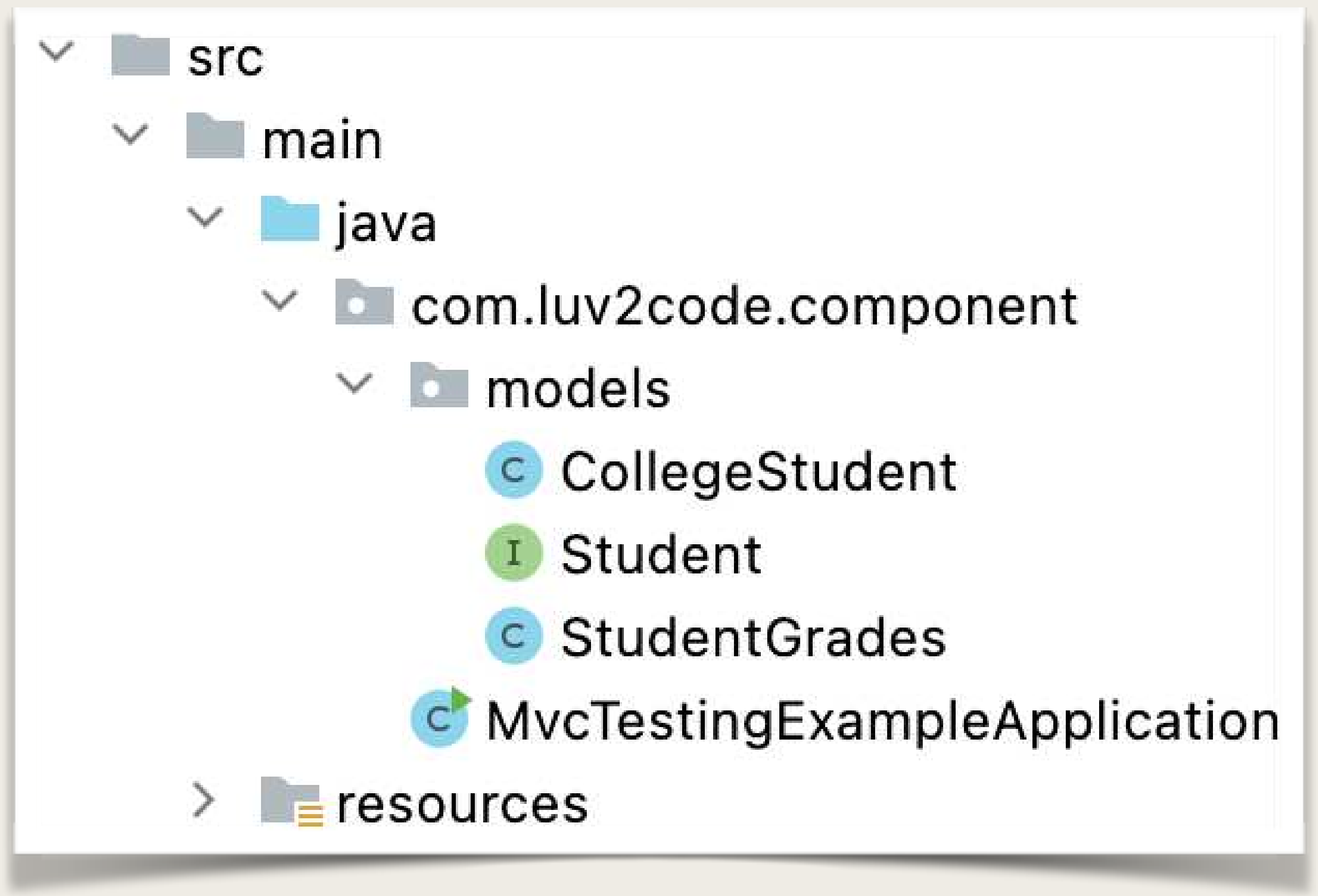
```
src
  main
    java
      com.luv2code.component
        models
          CollegeStudent
          Student
          StudentGrades
        MvcTestingExampleApplication
    resources
  test
    java
      com.luv2code.test
        ApplicationExampleTest
```

Explicitly reference main SpringBoot class

```
package com.luv2code.test;
…
@SpringBootTest(classes = MvcTestingExampleApplication.class)
public class ApplicationExampleTest {
  …
}
```

Our Sample Project

- Student grading system ... code was created by a previous employee
- The code doesn't have any tests ... gasp!
- We've been tasked with developing the unit tests ... yaayy!!!!



```
src
  main
    java
      com.luv2code.component
        models
          CollegeStudent
          Student
          StudentGrades
        MvcTestingExampleApplication
    resources
```