Haaga-Helia

Server Programming
Testing

Juha Hinkula, Jukka Juslin, Tanja Bergius,
Minna Pellikka

Haaga-Helia

Spring Boot - Testing

- Spring Boot provides a number of utilities and annotations to help when testing your application
- When Spring boot project is created you can find from POM.XML test dependency

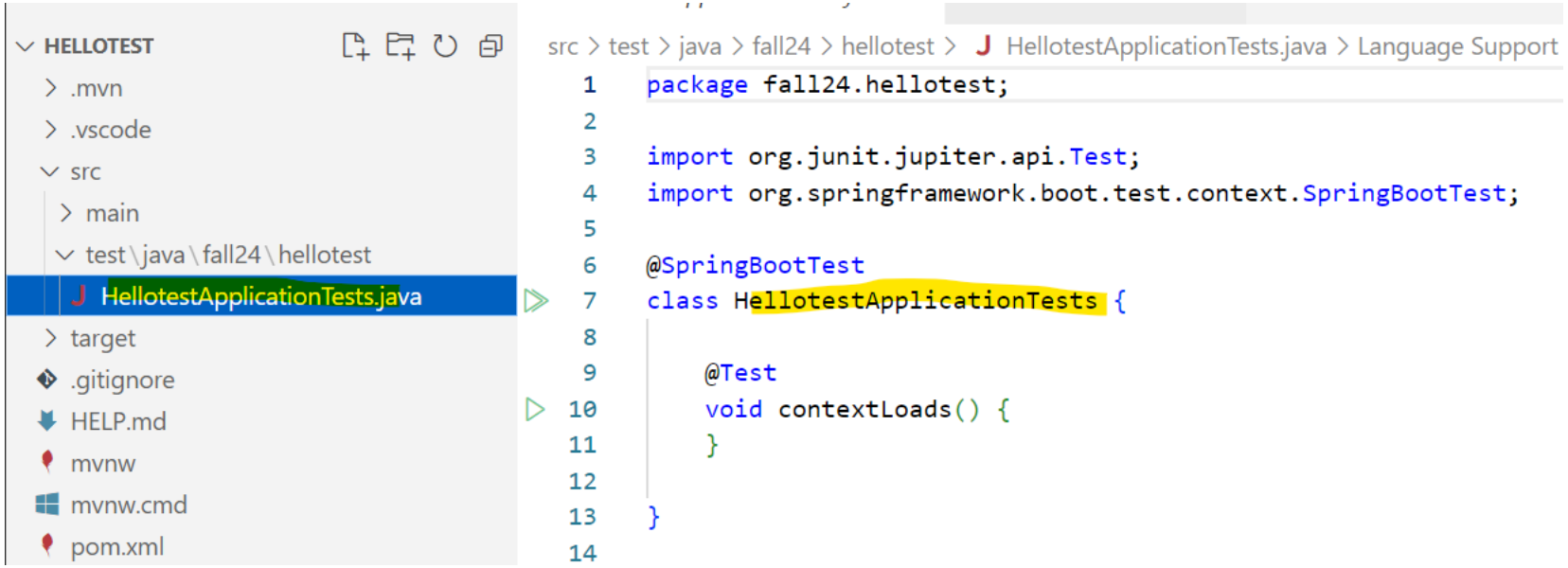
```
<dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
</dependency>
```

- This dependency provides some libraries and tools for testing, for instance Junit.

Haaga-Helia

Spring Boot - Testing

- Spring Boot will create automatically one test class for you



Haaga-Helia

Spring Boot - Testing

```
1   package fall24.hellotest;
2
3   import org.junit.jupiter.api.Test;
4   import org.springframework.boot.test.context.SpringBootTest;
5
6   @SpringBootTest
7   class HellotestApplicationTests {
8
9       @Test
10      void contextLoads() {
11      }
12
13  }
```

This is testing if application context is loaded without problems.

`@SpringBootTest` annotation tells that entire application will be started so that it can be tested

`@Test` annotation defines a method to be tested

Haaga-Helia

Spring Boot - Testing

- Spring Boot test dependency will add AssertJ library for assertions (http://joel-costigliola.github.io/assertj/index.html)
- Syntax

```
assertThat(objectToTest). // code completion -> assertions specific to objectToTest
```

- Examples
  - `assertThat(objectToTest).isNotNull();`
  - `assertThat("Hello World").contains("Hello");`
  - `assertThat(person.getName()).startsWith("M").endWith("s");`

Spring Boot - Testing

- Smoke testing
  - testing the major functions of software before carrying out formal testing

```
1.  import static org.assertj.core.api.Assertions.assertThat;
2.
3.  import org.junit.jupiter.api.Test;
4.  import org.springframework.beans.factory.annotation.Autowired;
5.  import org.springframework.boot.test.context.SpringBootTest;
6.
7.  import fi.haagahelia.course.web.HelloController;
8.
9.
10.  @SpringBootTest
11.  public class HellotestApplicationTests {
12.
13.  @Autowired
14.  private HelloController controller;
15.
16.    @Test
17.    public void contexLoads() throws Exception {
18.      assertThat(controller).isNotNull();
19.    }
20.  }
21.
```


Spring Boot - Testing

- Testing the web layer: Test will start the full Spring application context, but without the server by using Spring's MockMvc.

```
@SpringBootTest
@AutoConfigureMockMvc
public class WebLayerTest {
    @Autowired
    private MockMvc mockMvc;

    @Test
    public void testDefaultMessage() throws Exception {

this.mockMvc.perform(get("/")).andDo(print()).andExpect(status().isOk())
                .andExpect(content().string(containsString("Hello World")));
    }
}
```

Haaga-Helia

Testing the JPA repository / in-memory database

`@DataJpaTest` annotation will be used when testing in-memory database. It also turns on SQL logging

```
@DataJpaTest
public class StudentRepositoryTest {
   @Autowired
   private StudentRepository repository;

   @Test
    public void findByLastnameShouldReturnStudent() {
        List<Student> students = repository.findByLastName("Johnson");
        assertThat(students).hasSize(1);
        assertThat(students.get(0).getFirstName()).isEqualTo("John");
    }

    @Test
    public void createNewStudent() {
       Department department = new Department("BITE");
       drepository.save(department);
       Student student = new Student("Mickey", "Mouse", "mm@mouse.com", department);

       Student student = new Student("Mickey", "Mouse", "mm@mouse.com", new Department("BITE"));
       repository.save(student);
       assertThat(student.getId()).isNotNull();
    }
}
```

Haaga-Helia

JPA Testing on a real database (Postgres)

- When testing with other than development database annotations change and the way to reference category

```
@SpringBootTest(classes = StudentListApplication.class)
@AutoConfigureTestDatabase(replace = AutoConfigureTestDatabase.Replace.NONE) //if you are using
real db
public class StudentRepositoryTest {
    @Autowired
    private BookRepository repository;
    @Autowired
    private CategoryRepository crepository;

    @Test
    public void findByTitleShouldReturnAuthor() {
        List<Book> books = repository.findByTitle("The Old Man And The Sea");
        //assertThat(books).hasSize(1);
        assertThat(books.get(0).getAuthor()).isEqualTo("Ernest Hemingway");
    }

    @Test
    public void createNewBook() {

    Book book = new Book("Paul Trembley", "A Head Full of Ghosts", 2015, 16.30,
"ISBN434345621394", crepository.findByName("Classics").get(0));
    repository.save(book);
    assertThat(book.getId()).isNotNull();
    }
}
```