CODING SHUTTLE

Spring Testing

Understanding Junit and AssertJ

Common Junit Annotations

- **@Test**: Marks a method as a test method. JUnit will execute this method when running tests.
- **@DisplayName:** Sets a custom display name for the test class or test method. This name is used in test reports and IDEs.
- **@Disabled:** Disables a test class or test method. Disabled tests are not executed.

Junit Annotations

- **@BeforeEach**: Indicates that the annotated method should be executed before each test method. These can be used to reset each test case conditions.
- **@AfterEach**: Indicates that the annotated method should be executed after each test method.
- **@BeforeAll**: Indicates that the annotated method should be executed once before all test methods in the class. The method must be static. (Executed once)
- **@AfterAll**: Indicates that the annotated method should be executed once after all test methods in the class. The method must be static. (Executed once)

More JUnit Functionalities: JUnit5 Doc

JUnit vs AssertJ

JUnit is one of the most widely used testing frameworks in the Java ecosystem. JUnit provides a simple and standardized way to write test cases, execute them, and report the results.

AssertJ, on the other hand, is not a testing framework but rather a library that complements testing frameworks like JUnit. It focuses on providing fluent and expressive assertions, enhancing the readability and maintainability of your test code.

JUnit and AssertJ are both popular tools used in Java for testing, but they serve different purposes and have distinct features.

Common AssertJ Methods

1. Numbers:

```
assertThat(5).isEqualTo(5).isNotEqualTo(10).isGreaterThan(4);
```

2. String:

```
assertThat("hello").startsWith("he").endsWith("lo").contains("ell");
```

3. Boolean:

```
assertThat(true).isTrue(); OR assertThat(false).isFalse();
```

4. List/Array:

```
assertThat(List.of("apple", "banana")).contains("apple")
.doesNotContain("orange").hasSize(2);
```

Common AssertJ Methods

5. Exceptions:

```
assertThatThrownBy(() -> {
        throw new IllegalArgumentException("Invalid argument");
    }).isInstanceOf(IllegalArgumentException.class)
      .hasMessage("Invalid argument")
      .hasStackTraceContaining("ExampleTest");
```

More AssertJ Methods: https://assertj.github.io/doc/