Testing frameworks

- junit
- mockito

junit

https://github.com/junit-team/junit4

```
<dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <version>4.12</version>
</dependency>
```

JUnit is a unit testing framework for the Java programming language which

- instantiate the test class,
- calls each method annotated with @Test.

Annotations

@Test	Identifies a method as a test method.
@Before	Executed before each test.
@After	Executed after each test.
@BeforeClass	Executed once, before the start of all tests.
@AfterClass	Executed once, after all tests have been finished.
@Ignore("Why disabled")	Marks that the test should be disabled.
@Test (expected = Exception.class)	Fails if the method does not throw the named exception.
@Test(timeout=100)	Fails if the method takes longer than 100 milliseconds.

Assert statements

```
import static org.junit.Assert.*;
```

fail(message)	Let the method fail.
assertTrue([message,] boolean condition)	Checks that the boolean condition is true.
assertFalse([message,] boolean condition)	Checks that the boolean condition is false.
assertEquals([message,] expected, actual)	Tests that two values are the same.
assertEquals([message,] expected, actual, tolerance)	Test that float or double values match.
assertNull([message,] object)	Checks that the object is null.
assertNotNull([message,] object)	Checks that the object is not null.
assertSame([message,] expected, actual)	Checks that both variables refer to the same object.
assertNotSame([message,] expected, actual)	Checks that both variables refer to different objects.

JUnit test suites

If you have several test classes, you can combine them into a test suite. Running a test suite executes all test classes in that suite in the specified order. A test suite can also contain other test suites.

Parameterized test

JUnit allows you to use parameters in a tests class. This class can contain one test method and this method is executed with the different parameters provided.

JUnit Rules

Via JUnit rules you can add behavior to each tests in a test class. You can annotate fields of type TestRule with the @Rule annotation.

mockito

https://github.com/mockito/mockito

```
<dependency>
  <groupId>org.mockito</groupId>
  <artifactId>mockito-all</artifactId>
  <version>1.10.19</version>
</dependency>
```

Mockito is an open source testing framework for Java which allows the creation of test double objects (mock objects) in automated unit tests.

Test doubles

name	description
dummy	Passed around but never used, i.e., its methods are never called. Such an object can for example be used to fill the parameter list of a method.
fake	Have working implementations, but are usually simplified. For example, they use an in memory database and not a real database.
stub	Partial implementation for an interface or class with the purpose of using an instance of this stub class during testing. Stubs usually don’t respond to anything outside what’s programmed in for the test. Stubs may also record information about calls.
mock	Dummy implementation for an interface or a class in which you define the output of certain method calls. Mock objects are configured to perform a certain behavior during a test. They typical record the interaction with the system and test can validate that.

Mockito allows you to create and configure mock objects.

Mockito usage:

- Mock away external dependencies and insert the mocks into the code under test
- Execute the code under test
- Validate that the code executed correctly

Mockito cannot mock static methods and private methods.

Examples