Programming Advanced java

JUnit





Software testing – voor- en nadelen

PROS	CONS
Code quality	More code needs more time (at first)
Development time (less debugging)	Bad test = false idea of working code
Regressive	
Loosely coupled	
Descriptive	
Descriptive	



Software testing in java

- Testing framework
- Alternatief in java = TestNG

- Kent Beck
 - Agile software development
 - Test driven development



JUnit

- Test-driven development
 - First testing then coding
 - Testing → coding → testing → coding → ...
- Method annotations
- JUnit test = a method contained in a class which is only used for testing
- Assert / Assert Statements
- Naming conventions
 - Use descriptive method names, describe what you're testing
 - Use the word "should" in your method name
 - Ex. shouldFindExistingOrder of it_should_find_existing_order



JUnit

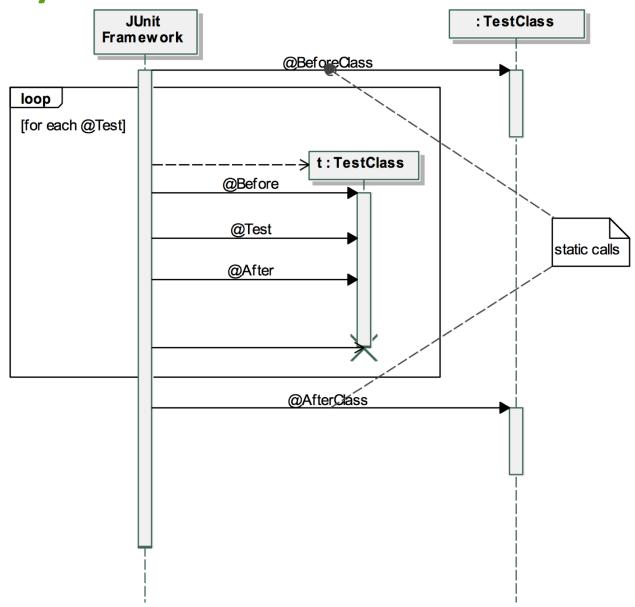
```
import org.junit.Test;
                                   Requires JUnit library
public class MyTests {
                                        Test class
        @Test ___
                       JUnit method annotation
        public void multiplicationOfZeroIntegersShouldReturnZero() {
                                                                                    Descriptive method name
                 MyClass tester = new MyClass();
                                                             Functional class to test
                 // assert statements
                 assertEquals("10 x 0 must be 0", 0, tester.multiply(10, 0));
                 assertEquals("0 x 10 must be 0", 0, tester.multiply(0, 10));
                 assertEquals("0 x 0 must be 0", 0, tester.multiply(0, 0));
                 Assertion / state test
                                                           Functionality to test
```

JUnit – test methods

- @Test annotation
- public
- return type void
- geen argumenten
- eender welke exception mag
- method naam is technisch gezien niet belangrijk



Junit - lifecycle



JUnit - Assertions

Methode	Betekenis
assertEquals()	Evalueert de gelijkheid van 2 waarden. De test slaagt als beide waarden gelijk (equal) zijn.
assertFalse()	Evaluatie van een booleaanse uitdrukking. De test slaagt indien de uitdrukking false is.
assertTrue()	Evaluatie van een booleaanse uitdrukking. De test slaagt indien de uitdrukking true is.
assertNotNull()	Vergelijkt een object referentie met null. De test slaagt indien de object referentie niet null is.
assertNull()	Vergelijkt een object referentie met null. De test slaagt indien de object referentie null is.
assertSame()	Vergelijkt het geheugenadres van twee object referenties (gebruik maken van == operator). De test slaagt indien beide object referenties naar hetzelfde object verwijzen.
fail()	Zorgt ervoor de de huidige test zal fallen. Wordt regelmatig gebruikt bij het testen van exception handling.



JUnit - Exceptions

@Test(expected=...Exception.class)