1. JUnit Test Suite

Any class can contain JUnit tests, but by convention the class name ends with Test. such as PurseTest.java, StackTest.java, etc. The test class is usually in the ***same package*** as the class under test.

2. Boilerplate Code for Class

The IDE will generate boilerplate code if you create a new "JUnit Test" instead of a plain Java class. Typical test code is:

```
import static org.junit.Assert.*;  // for assertEquals, assertTrue, etc
import static org.hamcrest.CoreMatchers.*; // Matchers used with assertThat

import org.junit.Before;
import org.junit.Test;

public class PurseTest {
    private static final double TOL = 1.0E-6; // tolerance for comparison
    // a "test fixture" - object to test
    private Purse purse;

    @Before
    public void setUp() throws Exception {
         // common code for each test, such as creating a new test fixture
         purse = new Purse(10);
    }

    /** A test method must be annotated by @Test */
    @Test
    public void testNewPurseIsEmpty() {
        purse = new Purse(20); // this replaces the @Before test fixture
        assertEquals( 0, purse.count() );
        // same thing, using assertThat
        assertThat( purse.count(), is(0) );
        // should not be full
        assertFalse("new purse is not full", purse.isFull());
    }

    /** Creating money with invalid value throws exception */
    @Test(expected=java.lang.IllegalArgumentException.class)
    public void testInvalidMoneyThrowsException() {
        Coin bogus = new Coin(-1, "Baht");
    }

    /** An impossible withdraw. Limit run time to avoid hanging tests. */
    @Test(timeout=100) /* millisecs */
    public void testImpossibleWithdraw() {
        double[] values = {2, 2, 5, 10};
        String currency = "BTC";
        for(double value: values) purse.deposit(new Coin(value, currency));
        assertEquals(19.0, purse.getBalance(currency), TOL);
        Valuable[] wd = purse.withdraw(18.0, currency);
        assertNull(wd);   // withdraw() returns null if can't withdraw
        assertEquals(19.0, purse.getBalance(currency), TOL);
        assertNull( purse.withdraw(10.0, "")); // currency doesn't match
    }
```

3. Common JUnit Assert methods. These are static methods in org.junit.Assert

assertEquals(*expected, actual*)	assertEquals(0, purse.count())
assertEquals("*message*", *expected*, *actual*)	assertEquals("Should be empty", 0, purse.count())
assertEquals(*expect*, *actual*, *tolerance*)	For compariing floating point values, you should specify a tolerance for two values to be considered "equal". Tolerance may be 0. assertEquals(0.0, purse.getBalance(), 1.0E-6)
assertSame(*expected*, *actual*)	Test if two object variables refer to the same object. This is like: assertTrue(expected == actual)
assertTrue(*boolean_expression*)	
assertFalse(*boolean_expression*)	assertFalse(purse.isFull())
assertNull(*variable*)	assertNull(purse.withdraw(1.0E+10, "Bitcoin"))
assertNotNull(*variable*)	assertNotNull(purse.withdraw(1, "Baht"))
assertThat(*expected*, *Matcher*)	assertThat(currency, is("Baht")) assertThat(money.toString(), contains("Coin")) Test results satisfies some condition, specified by a Matcher. See reference for examples.
assertArrayEquals(array1, array2)	Test if each element of two arrays are equal, using the array element's equals method.

If you want to test equality of two values (either objects or primitives) you should use

`assertEquals(expected, actual)`

and not write:

`assertTrue(expected == actual)` or `assertEqual(expected.equals(actual))`

because if the test fails assertEquals will show what the expected and actual values were, which helps you find the error. assertTrue will just report "expected true but result was false".

References

`http://junit.org` JUnit home. Has many examples and how-to.

"*Matchers and assertThat", JUnit Wiki.*

https://github.com/junit-team/junit4/wiki/matchers-and-assertthat

"*Benefit of* assertThat *over other* Assert *Methods",*

https://objectpartners.com/2013/09/18/the-benefits-of-using-assertthat-over-other-assert-methods-in-unit-tests/

Download JUnit from **`junit.org`** to get the JUnit API Javadoc and code samples.