luv2code

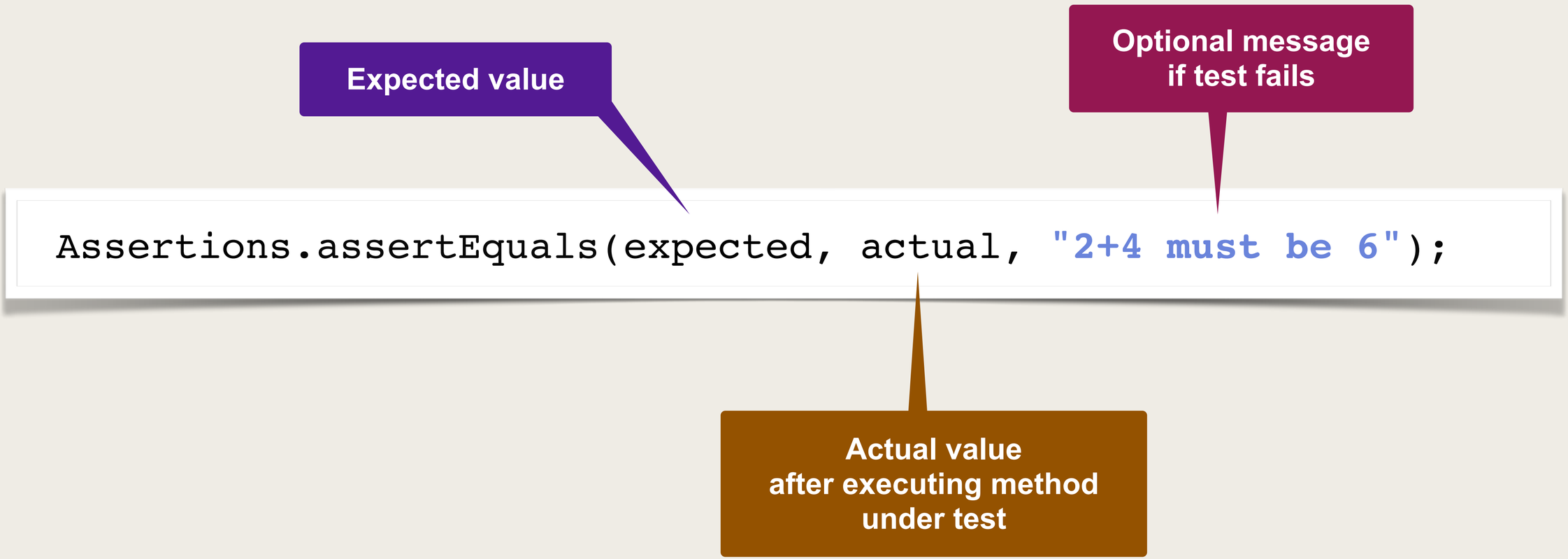
JUnit Assertions

JUnit Assertions

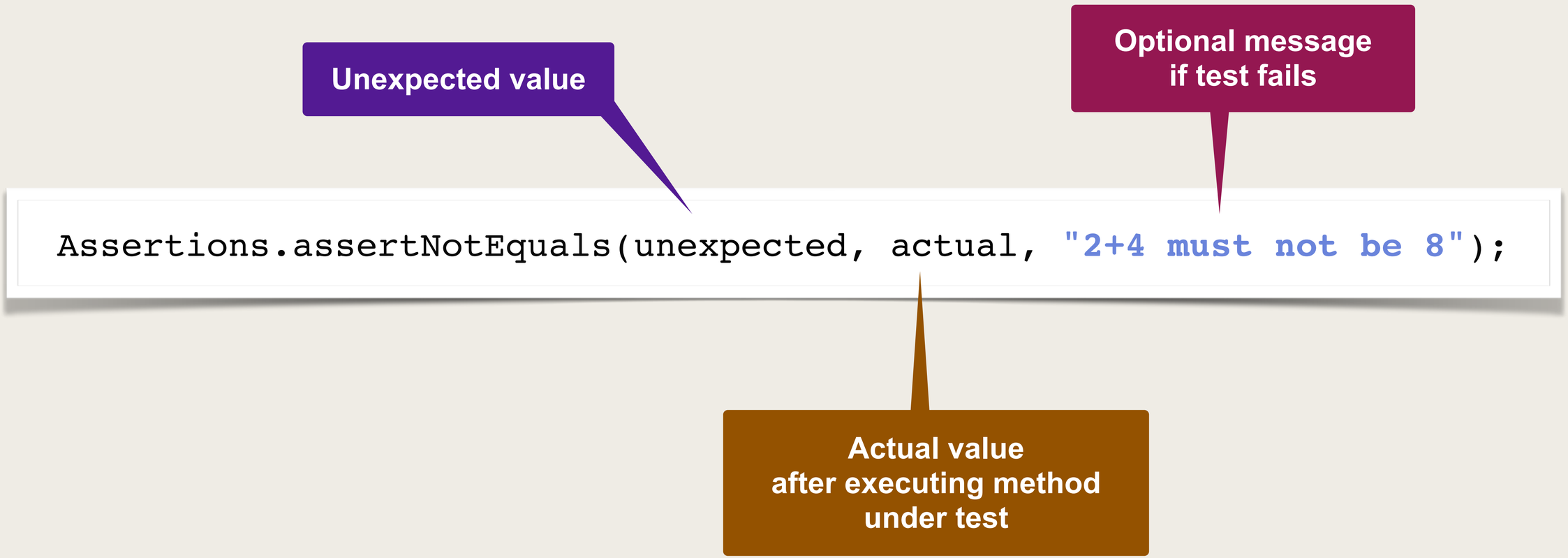
- JUnit has a collection of assertions
- Defined in class: `org.junit.jupiter.api.Assertions`

Method name	Description
`assertEquals`	Assert that the values are equal
`assertNotEquals`	Assert that the values are not equal
`assertNull`	Assert that the value is null
`assertNotNull`	Assert that the value is not null
...	...

assertEquals



assertNotEquals



DemoUtilsTest.java

```java
package com.luv2code.junitdemo;

import org.junit.jupiter.api.Test;
import org.junit.jupiter.api.Assertions;

class DemoUtilsTest {

   @Test
   void testEqualsAndNotEquals() {

       // set up
       DemoUtils demoUtils = new DemoUtils();

       int expected = 6;
       int unexpected = 8;

       // execute
       int actual = demoUtils.add(2, 4);

       // assert
       Assertions.assertEquals(expected, actual, "2+4 must be 6");
       Assertions.assertNotEquals(unexpected, actual, "2+4 must not be 8");

   }

}
```

Create instance of the class to test

Call the method you want to test

Check the *actual* result and
verify that it is NOT the *unexpected* result

Static Import

- Static import: a short-cut for referencing static methods & fields in a class

BEFORE

```
import org.junit.jupiter.api.Assertions;

   ...

   Assertions.assertEquals(expected, actual, "2+4 must be 6");
   Assertions.assertNotEquals(unexpected, actual, "2+4 must not be 8");
```

Class name

Method name

Short-cut for static methods & fields

Static import

```
import static org.junit.jupiter.api.Assertions.assertEquals;
import static org.junit.jupiter.api.Assertions.assertNotEquals;

   ...

   assertEquals(expected, actual, "2+4 must be 6");
   assertNotEquals(unexpected, actual, "2+4 must not be 8");
```

Just give method name

Static Import

- Can also use wildcard for static import

BEFORE

```
import static org.junit.jupiter.api.Assertions.assertEquals;
import static org.junit.jupiter.api.Assertions.assertNotEquals;

   ...

   assertEquals(expected, actual, "2+4 must be 6");
   assertNotEquals(unexpected, actual, "2+4 must not be 8");
```

AFTER

```
import static org.junit.jupiter.api.Assertions.*;

   ...

   assertEquals(expected, actual, "2+4 must be 6");
   assertNotEquals(unexpected, actual, "2+4 must not be 8");
```

Wildcard

Just give method name

Short-cut for static methods & fields

Static import

DemoUtilsTest.java

```
package com.luv2code.junitdemo;

import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.*;

class DemoUtilsTest {

   @Test
   void testEqualsAndNotEquals() {

       // set up
       DemoUtils demoUtils = new DemoUtils();

       int expected = 6;
       int unexpected = 8;

       // execute
       int actual = demoUtils.add(2, 4);

       // assert
       assertEquals(expected, actual, "2+4 must be 6");
       assertNotEquals(unexpected, actual, "2+4 must not be 8");

   }

}
```

Just give method name

Restructuring code

DemoUtilsTest.java

```
package com.luv2code.junitdemo;

import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.*;

class DemoUtilsTest {

    @Test
    void testEqualsAndNotEquals() {

        DemoUtils demoUtils = new DemoUtils();

        // execute and assert
        assertEquals(6, demoUtils.add(2, 4), "2+4 must be 6");
        assertNotEquals(8, demoUtils.add(1, 9), "1+9 must not be 8");

    }

}
```

Unexpected value

Optional message if test fails

Actual value after executing method under test

Code to Test

DemoUtils.java

```
package com.luv2code.junitdemo;

public class DemoUtils {

   public Object checkNull(Object obj) {

        if (obj != null) {
            return obj;
        }
        return null;

   }

}
```

Assertions: Null and NotNull

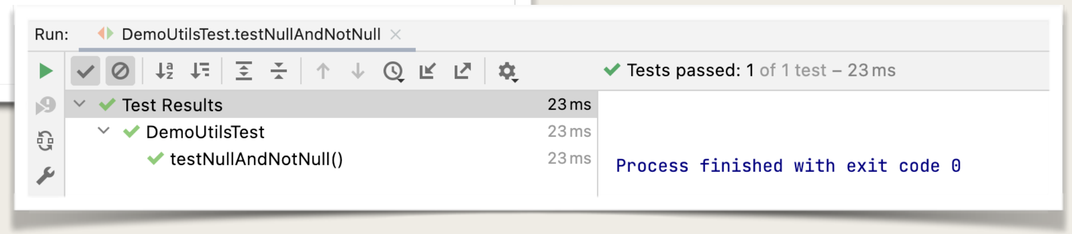
Method name	Description
`assertNull`	Assert that the value is null
`assertNotNull`	Assert that the value is not null
…	…

DemoUtilsTest.java

```
package com.luv2code.junitdemo;

import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.*;

class DemoUtilsTest {

    @Test
    void testNullAndNotNull() {

        DemoUtils demoUtils = new DemoUtils();

        String str1 = null;
        String str2 = "luv2code";

        assertNull(demoUtils.checkNull(str1), "Object should be null");
        assertNotNull(demoUtils.checkNull(str2), "Object should not be null");

    }

}
```

Optional message if test fails

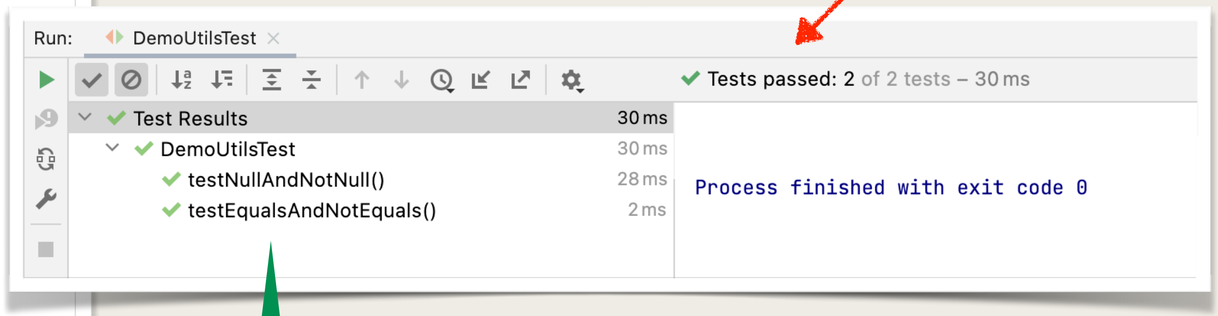
Actual value after executing method under test



Multiple Tests in One Class

DemoUtilsTest.java

```java
package com.luv2code.junitdemo;

import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.*;

class DemoUtilsTest {

    @Test
    void testEqualsAndNotEquals() {

        // set up
        DemoUtils demoUtils = new DemoUtils();

        // execute and assert
        assertEquals(6, demoUtils.add(2, 4), "2+4 must be 6");
        assertNotEquals(8, demoUtils.add(1, 9), "1+9 must not be 8");

    }

    @Test
    void testNullAndNotNull() {

        DemoUtils demoUtils = new DemoUtils();

        String str1 = null;
        String str2 = "luv2code";

        assertNull(demoUtils.checkNull(str1), "Object should be null");
        assertNotNull(demoUtils.checkNull(str2), "Object should not be null");

    }

}
```



Both tests passed

More Details

- JUnit User Guide

https://junit.org/junit5/docs/current/user-guide/#writing-tests