JUNIT

dasturlash.uz

Junit definition

- JUnit is a unit testing framework for Java programming language.
- JUnit has been important in the development of test-driven development, and is one of a family of unit testing frameworks collectively known as xUnit, that originated with JUnit.

- JUnit Java dasturlash tili uchun birlik test tizimidir.

Junit lessons

- This guide gives an introduction into unit testing with the JUnit framework using JUnit 5. It focus on the usage of the framework.

dasturlash.uz

Junit dependency

```
<dependency>
      <groupId>org.junit.jupiter</groupId>
      <artifactId>junit-jupiter-engine</artifactId>
      <version>5.4.0</version>
      <scope>test</scope>
  </dependency>
```

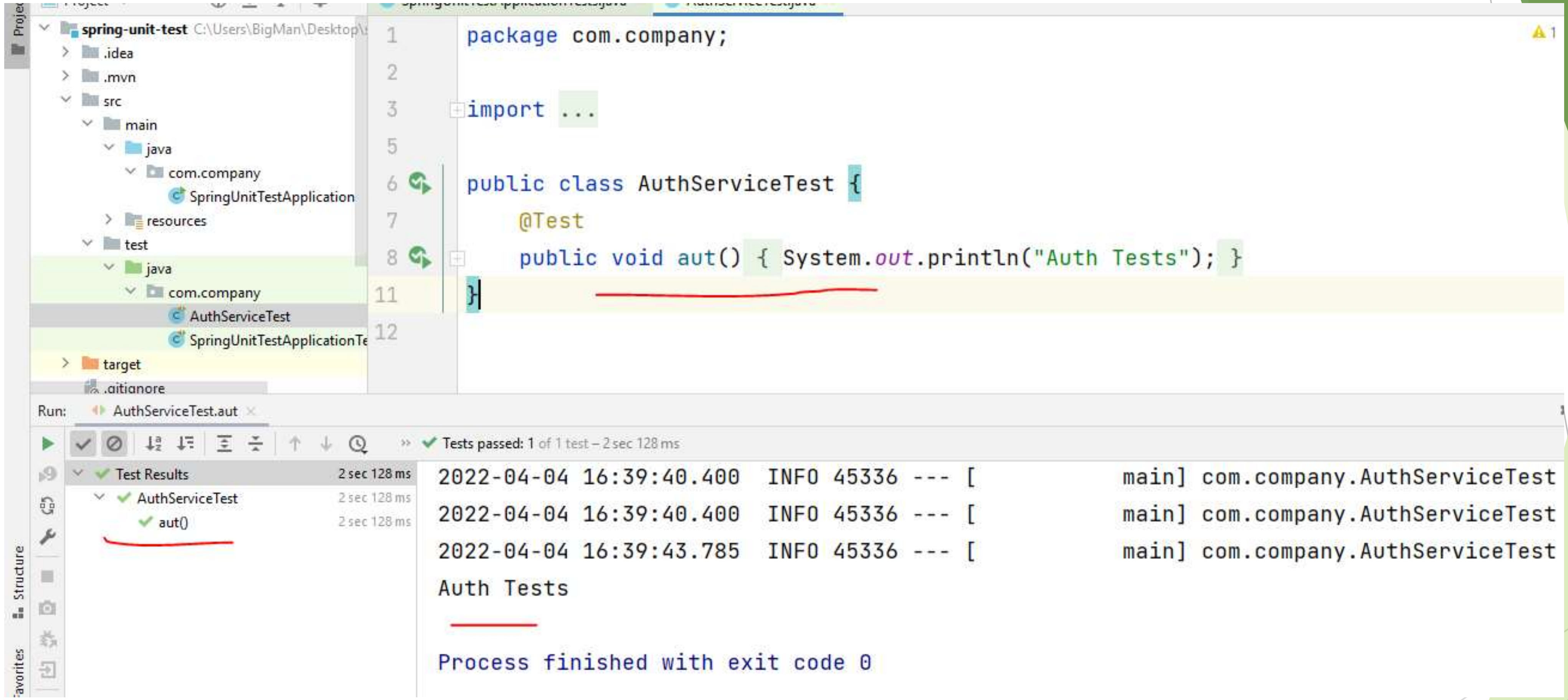
How to define a test in JUnit? - @Test

- A JUnit *test* is a method contained in a class which is only used for testing. This is called a *Test class*.To mark a method as a test method, annotate it with the @Test annotation. This method executes the code under test.

- Junit test bu faqat test qilish uchun ishlatiladigan class ning ichida yaratilgan metod xolos. Bu class Test Class deyiladi. Methodni test metodi qilib ishlatish uchun unga @Test annotatsiyasini yozing. Bu annotatsiya shu metodni ishlatadi.

```
public class AuthServiceTest {
    @Test
    public void aut() {
        // mazgi qandaydir test
    }
}
```

@Test - result



spring-unit-test C:\Users\BigMan\Desktop\
.idea
.mvn
src
main
java
com.company
SpringUnitTestApplication
resources
test
java
com.company
AuthServiceTest
SpringUnitTestApplicationTe
target
.gitignore
Run: AuthServiceTest.aut
Tests passed: 1 of 1 test – 2 sec 128 ms
Test Results 2 sec 128 ms
AuthServiceTest 2 sec 128 ms
aut() 2 sec 128 ms

```java
package com.company;

import ...

public class AuthServiceTest {
    @Test
    public void aut() { System.out.println("Auth Tests"); }
}
```

```
2022-04-04 16:39:40.400  INFO 45336 --- [           main] com.company.AuthServiceTest
2022-04-04 16:39:40.400  INFO 45336 --- [           main] com.company.AuthServiceTest
2022-04-04 16:39:43.785  INFO 45336 --- [           main] com.company.AuthServiceTest
Auth Tests

Process finished with exit code 0
```

dasturlash.uz

@SpringBootTest

- Start Class all test method

```
@SpringBootTest
public class AuthServiceTest {
    @Test
    public void aut() {
        System.out.println("Auth Tests");
    }

    @Test
    public void registration() {
        System.out.println("Registration Test");
    }
}
```

Junit Test - Idea

- Xullas methodda boshqa bir class (Service,controller,…) larni test qilamiz va ular return qilgan qiymatni biz kutgan qiymat bilan solishtiramiz. Agar kelgan qiymat biz solishtirgan qiymat bilan teng bolsa test Success bo'lmasa Fail bo'ladi.
- Tushunganskiy.

Davom ettirganskiy?

Junit example - 1

```
public class Calculator {
    public int multiply(int a, int b) {
        return a * b;
    }
}
```

- Test for this method

```
@Test
   void testMultiply() {
       Calculator calculator = new Calculator();
       assertEquals(20, calculator.multiply(4, 5));
   }
```

Junit – *assertEquals*

- *assertEquals – method used to compare two values. If they are equal it test will pass. If not the test will fail.*
- *AssertEquls – metod 2ta qiymatni solishtirish uchun ishlatiladi. Agar ular teng bo’lsa test mofaqiyatli bo’ladi. Agar teng bo’lmasa mofaqiyatsiz bo’ladi.*

- *Internar implemantation (ichki realizatsiyasi.)*
  - **assertEquals(expected, actual)**
    - **expected.equals(actual)**
  - *assertEquals(expected,actual,message)*

dasturlash.uz

JUnit Assert methods

dasturlash.uz

JUnit Assert methods

- Boolean
  - assertTrue(condition)
  - assertFalse(condition)
- Null object
  - assertNull(object)
  - assertNotNull(object)
- Identical
  - assertSame(expected, actual), It will return true if **expected == actual**
  - assertNotSame(expected, actual)
- Assert Equals
  - assertEquals(expected, actual)
- Assert Array Equals
  - assertArrayEquals(expected, actual)
-

JUnit Assert methods - Boolean

- If you want to test the boolean conditions (true or false), you can use following assert methods
  - **1. assertTrue(condition)**
  - **2. assertFalse(condition)**
- Here the condition is a boolean value.

- Boolean holatlarni tekshirish uchun yuqoridagi metodlardan foydalansak bo’ladi.
- condition bu yerda boolean qiymat.
- assertTrue(condition) – agar condition true bo’lmasa test dan o’tmaydi.
- assertFalse(condition) – agar condition false bo’lmasa test dan o’tmaydi.

dasturlash.uz

JUnit Assert methods - Null object

- If you want to check the initial value of an object/variable, you have the following methods:
- **1. assertNull(object)**
- **2. assertNotNull(object)**
- Null qiymatni tekshirmoqchi bo'lsangiz yuqoridagi methodlarni ishlating.

JUnit Assert methods - Identical

- If you want to check whether the objects are identical (i.e. comparing two references to the same java object), or different.
  - **1. assertSame(expected, actual),** It will return true if **expected == actual**
  - **2. assertNotSame(expected, actual)**
- Ob'ektlarni yoki qiymatlarni tengligini solishtirish uchun yuqoridagi methodlarni ishlating.

JUnit Assert methods - Assert Equals

- If you want to test equality of two objects, you have the following methods
  - **assertEquals(expected, actual)**
  - **assertEquals(expected,actual,message)**
- It will return true if: **expected.equals(actual)** returns true.
-

JUnit Assert methods - Assert Array Equals

- If you want to test equality of arrays, you have the following methods as given below:
  - **assertArrayEquals(expected, actual)**
- Above method must be used if arrays have the same length, for each valid value for **i**, you can check it as given below.
- Tepadagi method birxil uzunlikdagi arraylarni solishtirish uchun ishlatiladi. Agar arrayni i chi elementini solishtirish uchun quyidagi methodlarni ishlatish mumkin.
  - **assertEquals(expected[i],actual[i])**
  - **assertArrayEquals(expected[i],actual[i])**
-

dasturlash.uz

JUnit Assert methods - assertThrows() 1

- There are two ways of exception testing in JUnit 5, both of which we can implement using the *assertThrows()* method:
- First verifies validates the type of exception, and the second one validates the details of the thrown exception.

- JUnit 5 da exception larni test qilishning 2ta usuli mavjut. Ammo ikkalasi ham assertThrows() meodi orqali bajariladi.
- Birinchisi Exception ni turini tekshiradi, ikkinchisi uni messagini tekshiradi.
  - assertThrows(Class<T> expectedType, Executable executable)
  - assertThrows(Class<T> expectedType, Executable executable, String message)
- Executable interface:

```
public interface Executable {
    void execute() throws Throwable;
}
```

dasturlash.uz

JUnit Assert methods - assertThrows() 2

- Method

```
public int division(int a, int b) {
      if (b == 0) {
          throw new ArithmeticException("Division by 0");
      }
      return a / b;
  }
```

- Test throwed exception class

- Test

```
@Test
void exceptionTest() {
    Calculator calculator = new Calculator();
    assertThrows(ArithmeticException.class, () -> {
        calculator.division(3, 0);
    });
}
```

JUnit Assert methods - assertThrows() 3

- Method

```
public int division(int a, int b) {
       if (b == 0) {
           throw new ArithmeticException("Division by 0");
       }
       return a / b;
   }
```

- Test

```
@Test
  void exceptionMessageTest() {
      Calculator calculator = new Calculator();
      Throwable exception = assertThrows(ArithmeticException.class, () -> {
          calculator.division(3, 0);
      });
      assertEquals(exception.getMessage(), "Division by 0");
  }
```

- Test throwed exception message

JUnit Assert methods - multiple assertions

- In case you want to ensure that all asserts are checked you can assertAll.
- Bitta methodni ichida bir nechta shartlarni tekshirish uchun assertAll methodni ishlating.

```
@Test
void groupedAssertions() {
    Address address = new Address();
    assertAll("address name",
        () -> assertEquals("John", address.getFirstName()),
        () -> assertEquals("User", address.getLastName())
    );
}
```

JUnit Assert methods - timeouts

- If you want to ensure that a test fails, if it isn’t done in a certain amount of time you can use the assertTimeout() method. This assert fails the method if the timeout is exceeded.
- Methodni bajarilish vaqtini test qilish uchun assertTimeOut() metodini ishlating.

```
@Test
void timeoutNotExceeded() {
    assertTimeout(ofMinutes(1), () -> service.doBackup());
}

// if you have to check a return value
@Test
void timeoutNotExceededWithResult() {
    String actualResult = assertTimeout(ofSeconds(1), () -> {
        return restService.request(request);
    });
    assertEquals(200, request.getStatus());
}
```

Fail Message

- If you want to throw any assertion error, you have **fail()** that always results in a fail verdict.
  - **fail(message);**
- faile() metodi orqali sodir bo’ladigan hatolikni xabarini o’zgartirish mumkin.
- You can have assertion method with an additional **String** parameter as the first parameter. This string will be appended in the failure message if the assertion fails. E.g. **fail(message)** can be written as:
  - **assertEquals(expected, actual, message)**
- Bundan tashqari ko’pgina Asser metodlarida 3chi parametr sifatida xabarni berish mumkin. Bu xabar hatolik sodir bo’lsa ko’rinadigan xabardir.
-

Junit Annotations

dasturlash.uz

Junit Annotations - @*Test*

- *@Test* – The Test annotation tells JUnit that the public void method to which it is attached can be run as a test case.
- @Test – annotatsiyasi Junitga bu method public void metod va uni test qilish uchun ishlatish kerakligini bildiradi.

Junit Annotation - @Disabled

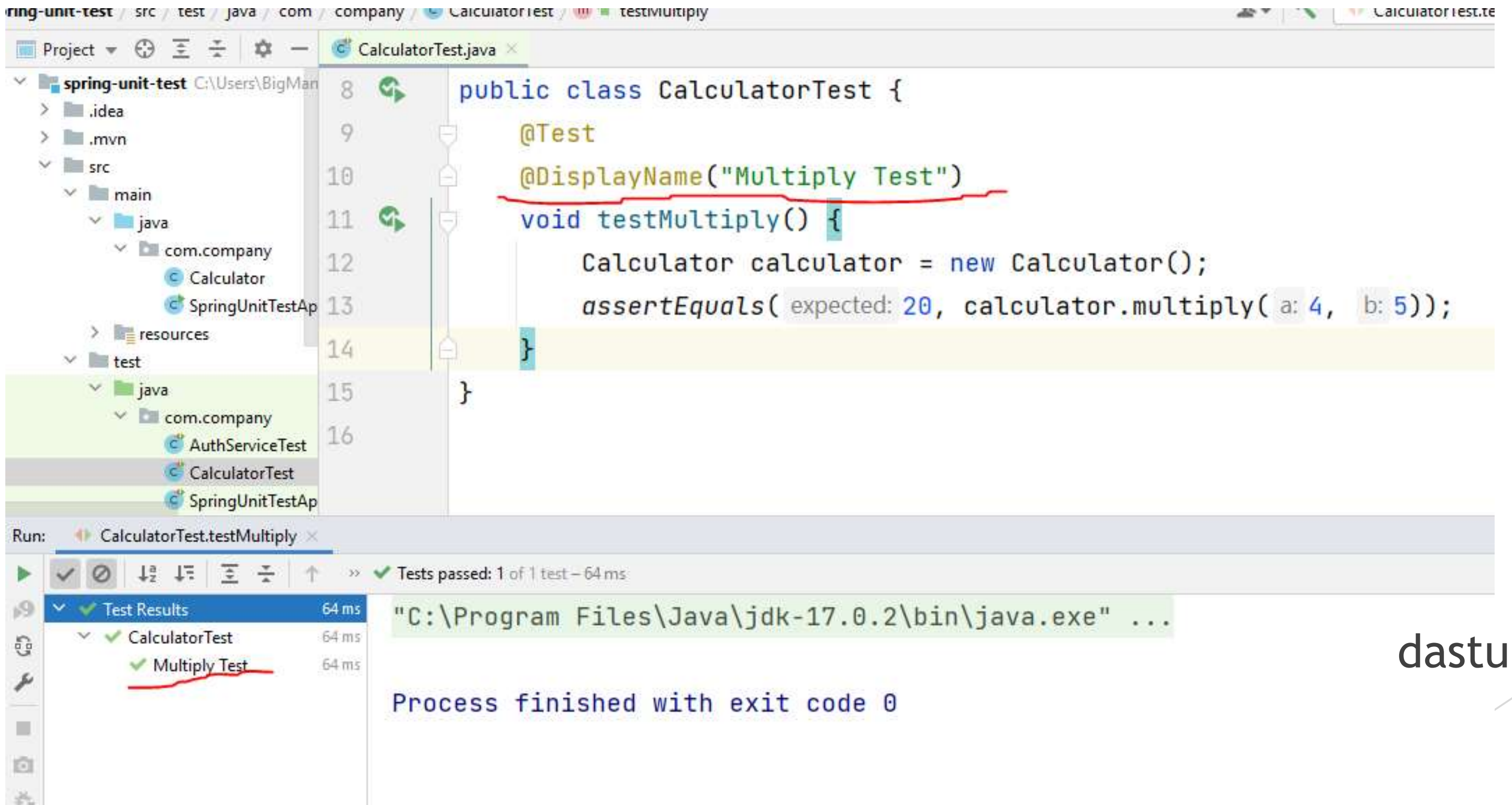
- The @Disabled or @Disabled("Why disabled") annotation marks a test to be disabled.
- @Disabled – yozilgan test metodni o’chirib qo’yadi. Yani test ishlamaydi.

```
@Test
 @Disabled("Method o'zgargan")
 void exceptionMessageTest() {

 }
```

Junit Annotation - *@DisplayName*

- @DisplayName can be used to define the name of the test which is displayed to the user



```
public class CalculatorTest {
    @Test
    @DisplayName("Multiply Test")
    void testMultiply() {
        Calculator calculator = new Calculator();
        assertEquals( expected: 20, calculator.multiply( a: 4,  b: 5));
    }
}
```

Run: CalculatorTest.testMultiply

Tests passed: 1 of 1 test – 64 ms

```
Test Results        64 ms
  CalculatorTest    64 ms
    Multiply Test   64 ms
```

```
"C:\Program Files\Java\jdk-17.0.2\bin\java.exe" ...

Process finished with exit code 0
```

dasturlash.uz

Junit Annotation - *@Nested*

- *@Nested* – denotes that the annotated class is a nested, non-static test class
- @Nested – annotatsiyasi inner test class ni bildirish uchun ishlatilaid.

```
class JUnit5NestedExampleTest {
    @Test
    void exceptionTest() {
                                                  //...
    }

     @Nested
    @DisplayName("Tests for the method A")
    class A {
        @Test
        @DisplayName("Example test for method A")
        void sampleTestForMethodA() {
            System.out.println("Example test for method A");
        }
    }
}
```

dasturlash.uz

Junit Annotation - *@BeforeAll*

- *@BeforeAll* – denotes that the annotated method will be executed before all test methods in the current class (previously *@BeforeClass*)
- @BeforeAll – annotatsiyasi berilgan metod class dagi barcha @Test metodlaridan oldin ishga tushadi.

```
public class CalculatorTest {
    static Calculator calculator;

    @BeforeAll
    public static void init() {
        calculator = new Calculator();
    }
}
```

dasturlash.uz

Junit Annotation - *@BeforeEach*

- *@BeforeEach* – denotes that the annotated method will be executed before each test method (previously *@Before*)
- *@BeforeEach - annotatsiya yozilgan method @Test metodi yozilgan har bir methoddan oldin ishga tushadi.*

Junit Annotation - *@AfterAll*

- *@AfterAll* – denotes that the annotated method will be executed after all test methods in the current class (previously *@AfterClass*)
- @AfterAll – annotatsiya yozilgan method Class dagi Barcha @Test annotatsiyani yozilgan methodlardan keyin ishlaydi.

Junit Annotation - *@AfterEach*

- *@AfterEach* – denotes that the annotated method will be executed after each test method (previously *@After*)
- @AfterEach – annotatsiyasi yozilgan method har bir @Test annotatsiyasi yozilgan method dan keyin ishga tushadi.

Links

- https://www.guru99.com/junit-assert.html#3
- https://www.baeldung.com/junit-5#exception-testing
- https://www.guru99.com/junit-tutorial.html