IN28MINUTES

JUnit Tutorial for Beginners in 5 Steps

- Git Repository - https://github.com/in28minutes/getting-started-in-5-steps/tree/master/junit-in-5-steps
- Pre-requisites
    - Java & Eclipse - https://www.youtube.com/playlist?list=PLBBog2r6uMCSmMVTW_QmDLyASBvovyAO3
- We will use embedded maven in Eclipse

JUnit is the most popular Java Unit testing framework

Here is an overview of what we would learn in this section

- Step 1 : What is JUnit and Unit Testing?
- Step 2 : First JUnit Project and Green Bar
- Step 3 : First Code and First Unit Test
- Step 4 : Other assert methods
- Step 5 : Important annotations

Step 1 : What is JUnit and Unit Testing?

- What is JUnit?
- What is Unit Testing?
- Advantages of Unit Testing

We typically work in large projects - some of these projects have more than 2000 source files or sometimes it might be as big as 10000 files with one million lines of code.

Before unit testing, we depend on deploying the entire app and checking if the screens look great. But that's not very efficient. And it is manual.

Unit Testing focuses on writing automated tests for individual classes and methods.

JUnit is a framework which will help you call a method and check (or assert) whether the output is as expected.

The important thing about automation testing is that these tests can be run with continuous integration - as soon as some code changes.

Step 2 : First JUnit Project and Green Bar

- What is JUnit?
- First Project with JUnit
- First JUnit Class
- No Failure is Success
- MyMath class with sum method

```
package com.in28minutes.junit;

public class MyMath {
	int sum(int[] numbers) {
		int sum = 0;
		for (int i : numbers) {
			sum += i;
		}
		return sum;
	}
}
```

Step 3 : First Code and First Unit Test

- Unit test for the sum method

```
package com.in28minutes.junit;

import static org.junit.Assert.assertEquals;

import org.junit.After;
import org.junit.AfterClass;
import org.junit.Before;
import org.junit.BeforeClass;
import org.junit.Test;

public class MyMathTest {
	MyMath myMath = new MyMath();

	// MyMath.sum
	// 1,2,3 => 6
	@Test
	public void sum_with3numbers() {
		System.out.println("Test1");
		assertEquals(6, myMath.sum(new int[] { 1, 2, 3 }));
	}

	@Test
	public void sum_with1number() {
		System.out.println("Test2");
		assertEquals(3, myMath.sum(new int[] { 3 }));
	}
}
```

Step 4 : Other assert methods

- assertTrue and assertFalse methods

```
package com.in28minutes.junit;

import static org.junit.Assert.assertEquals;
import static org.junit.Assert.assertTrue;

import org.junit.Test;

public class AssertTest {

	@Test
	public void test() {
		boolean condn = true;
		assertEquals(true, condn);
		assertTrue(condn);
		// assertFalse(condn);
	}

}
```

Step 5 : Important annotations

- @Before @After annotations
  - Run before and after every test method in the class
- @BeforeClass @AfterClass annotations
  - Static methods which are executed once before and after a test class

```
package com.in28minutes.junit;

import static org.junit.Assert.assertEquals;

import org.junit.After;
import org.junit.AfterClass;
import org.junit.Before;
import org.junit.BeforeClass;
import org.junit.Test;

public class MyMathTest {
	MyMath myMath = new MyMath();

	@Before
	public void before() {
		System.out.println("Before");
	}

	@After
	public void after() {
		System.out.println("After");
	}

	@BeforeClass
	public static void beforeClass() {
		System.out.println("Before Class");
	}

	@AfterClass
	public static void afterClass() {
		System.out.println("After Class");
	}
```

```
        // MyMath.sum
        // 1,2,3 => 6
        @Test
        public void sum_with3numbers() {
                System.out.println("Test1");
                assertEquals(6, myMath.sum(new int[] { 1, 2, 3 }));
        }

        @Test
        public void sum_with1number() {
                System.out.println("Test2");
                assertEquals(3, myMath.sum(new int[] { 3 }));
        }
}
```

Complete Code Example

/src/com/in28minutes/junit/MyMath.java

```
package com.in28minutes.junit;

public class MyMath {
        int sum(int[] numbers) {
                int sum = 0;
                for (int i : numbers) {
                        sum += i;
                }
                return sum;
        }
}
```

/test/com/in28minutes/junit/AssertTest.java

```
package com.in28minutes.junit;

import static org.junit.Assert.assertEquals;
import static org.junit.Assert.assertTrue;

import org.junit.Test;

public class AssertTest {

        @Test
        public void test() {
                boolean condn = true;
                assertEquals(true, condn);
                assertTrue(condn);
                // assertFalse(condn);
        }

}
```

/test/com/in28minutes/junit/MyMathTest.java

```
package com.in28minutes.junit;

import static org.junit.Assert.assertEquals;

import org.junit.After;
import org.junit.AfterClass;
import org.junit.Before;
import org.junit.BeforeClass;
import org.junit.Test;

public class MyMathTest {
	MyMath myMath = new MyMath();

	@Before
	public void before() {
		System.out.println("Before");
	}

	@After
	public void after() {
		System.out.println("After");
	}

	@BeforeClass
	public static void beforeClass() {
		System.out.println("Before Class");
	}

	@AfterClass
	public static void afterClass() {
		System.out.println("After Class");
	}

	// MyMath.sum
	// 1,2,3 => 6
	@Test
	public void sum_with3numbers() {
		System.out.println("Test1");
		assertEquals(6, myMath.sum(new int[] { 1, 2, 3 }));
	}

	@Test
	public void sum_with1number() {
		System.out.println("Test2");
		assertEquals(3, myMath.sum(new int[] { 3 }));
	}
}
```
