luv2code

JUnit Lifecycle Methods

Lifecycle Methods

- When developing tests, we may need to perform common operations
- Before each test
  - Create objects, set up test data
- After each test
  - Release resources, clean up test data

Lifecycle Method Annotations

Annotation	Description
`@BeforeEach`	Method is executed before each test method. *Useful for common set up code: creating objects, setting up test data*
`@AfterEach`	Method is executed after each test method. *Useful for common clean up code: releasing resources, cleaning up test data*
...	...

Execution Sequence

@BeforeEach

@Test Method One

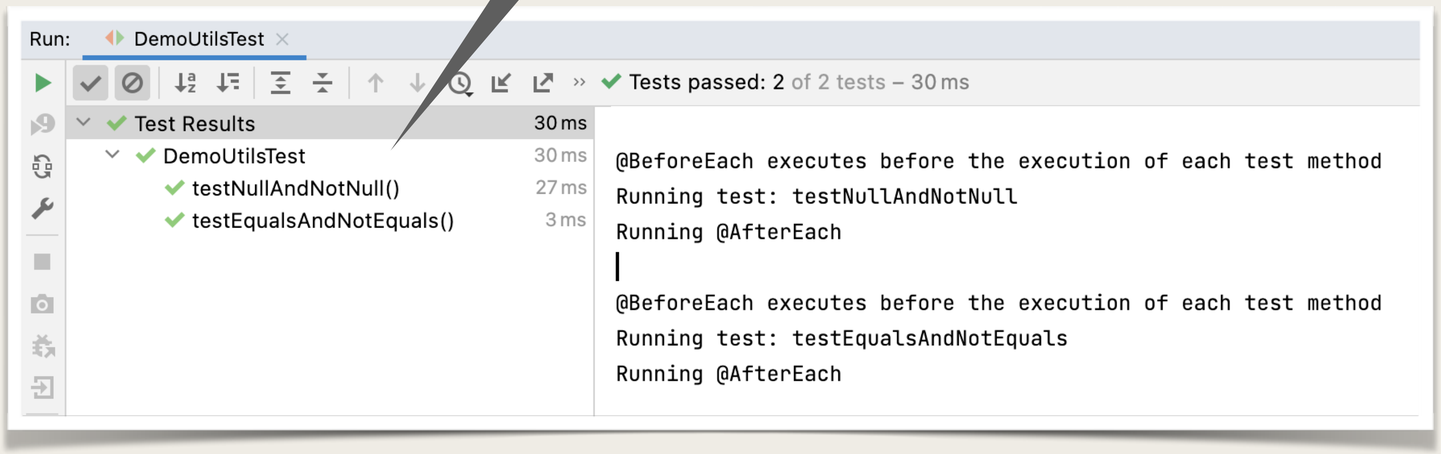
@AfterEach

@BeforeEach

@Test Method Two

@AfterEach

Don't worry about order of test methods.
We'll cover that later.




```
@BeforeEach
void setupBeforeEach() {
    // set up
    demoUtils = new DemoUtils();
    System.out.println("@BeforeEach executes before the execution of each test method");
}

@AfterEach
void tearDownAfterEach() {
    System.out.println("Running @AfterEach\n");
}

@Test
void testNullAndNotNull() {

    System.out.println("Running test: testNullAndNotNull");


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

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

...
```

DemoUtilsTest.java

BEFORE

```
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() {

        // set up
        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");
    }

}
```

DemoUtilsTest.java

AFTER

```
package com.luv2code.junitdemo;

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

class DemoUtilsTest {

    DemoUtils demoUtils;

    @BeforeEach
    void setupBeforeEach() {
        // set up
        demoUtils = new DemoUtils();
        System.out.println("@BeforeEach executes before the execution of each test method");
    }

    @Test
    void testEqualsAndNotEquals() {

        // 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() {
        String str1 = null;
        String str2 = "luv2code";

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

Create object before each test

No need to create object ... handled by @BeforeEach

No need to create object ... handled by @BeforeEach

DemoUtilsTest.java

```
package com.luv2code.junitdemo;

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

class DemoUtilsTest {

    DemoUtils demoUtils;

    @BeforeEach
    void setupBeforeEach() {
        // set up
        demoUtils = new DemoUtils();
        System.out.println("@BeforeEach executes before the execution of each test method");
    }

    @AfterEach
    void tearDownAfterEach() {
        System.out.println("Running @AfterEach\n");
    }

    @Test
    void testEqualsAndNotEquals() {

        // 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() {
        String str1 = null;
        String str2 = "luv2code";

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


Run after each test

DemoUtilsTest.java

```java
package com.luv2code.junitdemo;

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

class DemoUtilsTest {

    DemoUtils demoUtils;

    @BeforeEach
    void setupBeforeEach() {
        // set up
        demoUtils = new DemoUtils();
        System.out.println("@BeforeEach executes before the execution of each test method");
    }

    @AfterEach
    void tearDownAfterEach() {
        System.out.println("Running @AfterEach\n");
    }

    @Test
    void testEqualsAndNotEquals() {

        System.out.println("Running test: testEqualsAndNotEquals");

        // 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() {

        System.out.println("Running test: testNullAndNotNull");


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

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

Adding println(...) for diagnostics

Adding println(...) for diagnostics

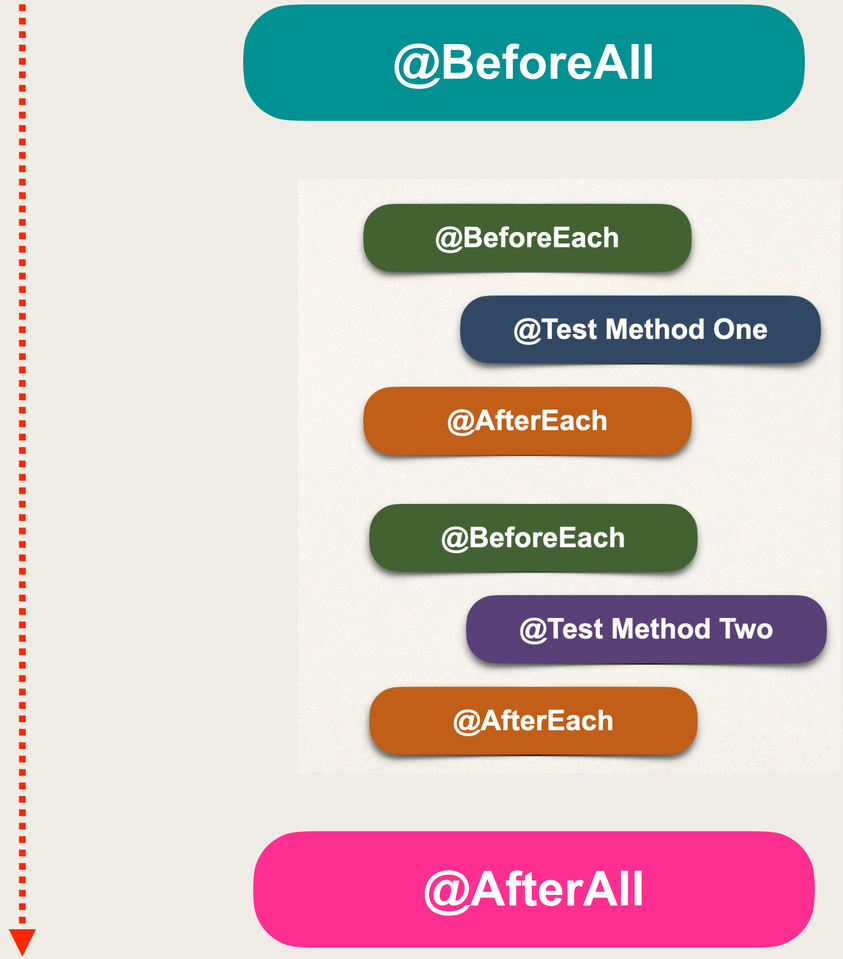
Lifecycle Methods

- When developing tests, we may need to perform one-time operations
- One-time set up before all tests
  - Get database connections, connect to remote servers ...
- One-time clean up after all tests
  - Release database connections, disconnect from remote servers ...

Lifecycle Method Annotations

Annotation	Description
`@BeforeAll`	Method is executed only once, before all test methods. *Useful for getting database connections, connecting to servers ...*
`@AfterAll`	Method is executed only once, after all test methods. *Useful for releasing database connections, disconnecting from servers ...*
...	...

Execution Sequence



DemoUtilsTest.java

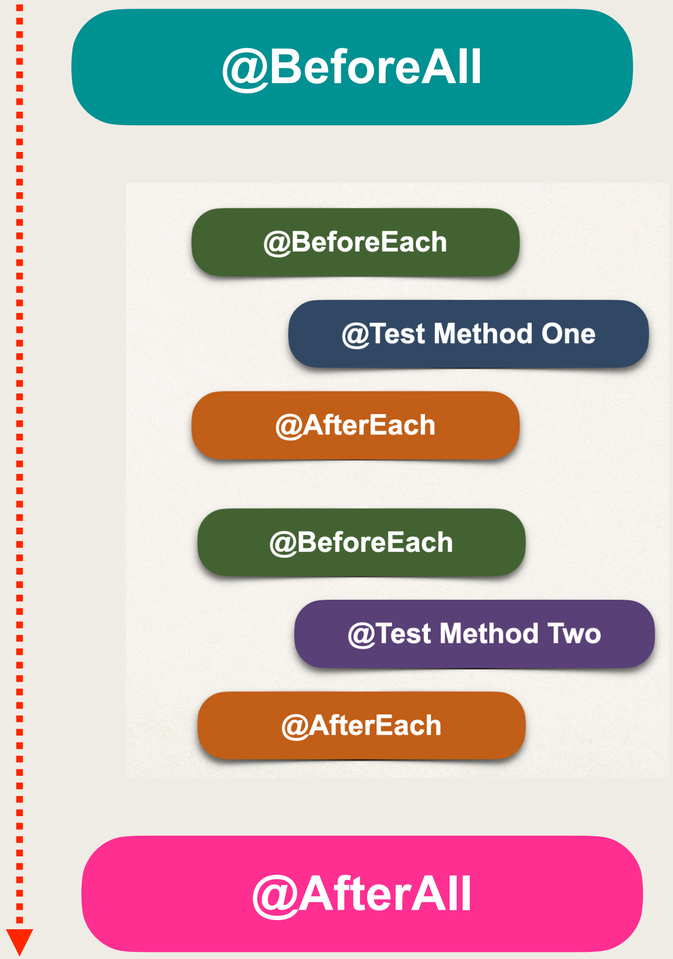
```java
package com.luv2code.junitdemo;

import org.junit.jupiter.api.Test;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.AfterEeach;
import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.AfterAll;
import static org.junit.jupiter.api.Assertions.*;

class DemoUtilsTest {

    DemoUtils demoUtils;

    @BeforeEach
    void setupBeforeEach() {
        // set up
        demoUtils = new DemoUtils();
        System.out.println("@BeforeEach executes before the execution of each test method");
    }

    @AfterEach
    void tearDownAfterEach() {
        System.out.println("Running @AfterEach\n");
    }

    @BeforeAll
    static void setupBeforeEachClass() {
        System.out.println("@BeforeAll executes only once before all test methods execution in the class\n");
    }

    @AfterAll
    static void tearDownAfterAll() {
        System.out.println("@AfterAll executes only once after all test methods execution in the class");
    }

    @Test
    void testEqualsAndNotEquals() {
        System.out.println("Running test: testEqualsAndNotEquals");
        ...
    }

    @Test
    void testNullAndNotNull() {
        System.out.println("Running test: testNullAndNotNull");
        ...
    }
}
```

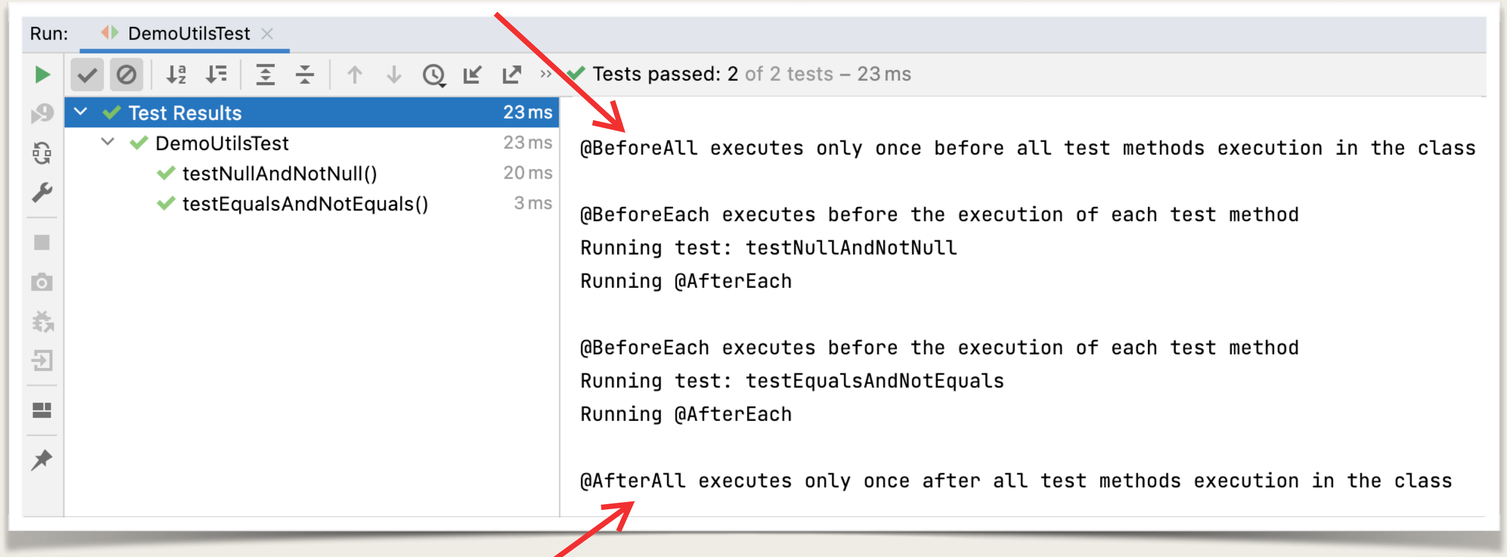
Executed only once, before all test methods

Executed only once, after all test methods

By default, methods must be static

Execution Sequence






```java
@BeforeAll
static void setupBeforeEachClass() {
    System.out.println("@BeforeAll executes only once before all test methods execution in the class\n");
}

@AfterAll
static void tearDownAfterAll() {
    System.out.println("@AfterAll executes only once after all test methods execution in the class");
}
```