@BeforeAll and @AfterAll Annotation

This lesson demonstrates working of two more Lifecycle methods annotated with - @BeforeAll and @AfterAll Annotation.

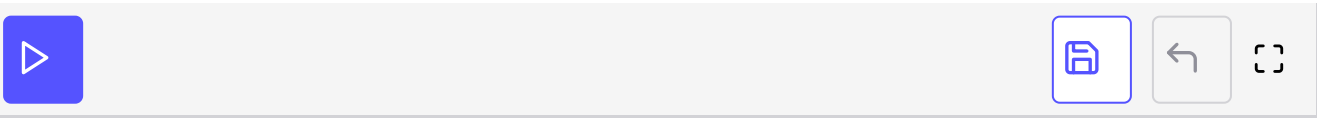
WE'LL COVER THE FOLLOWING

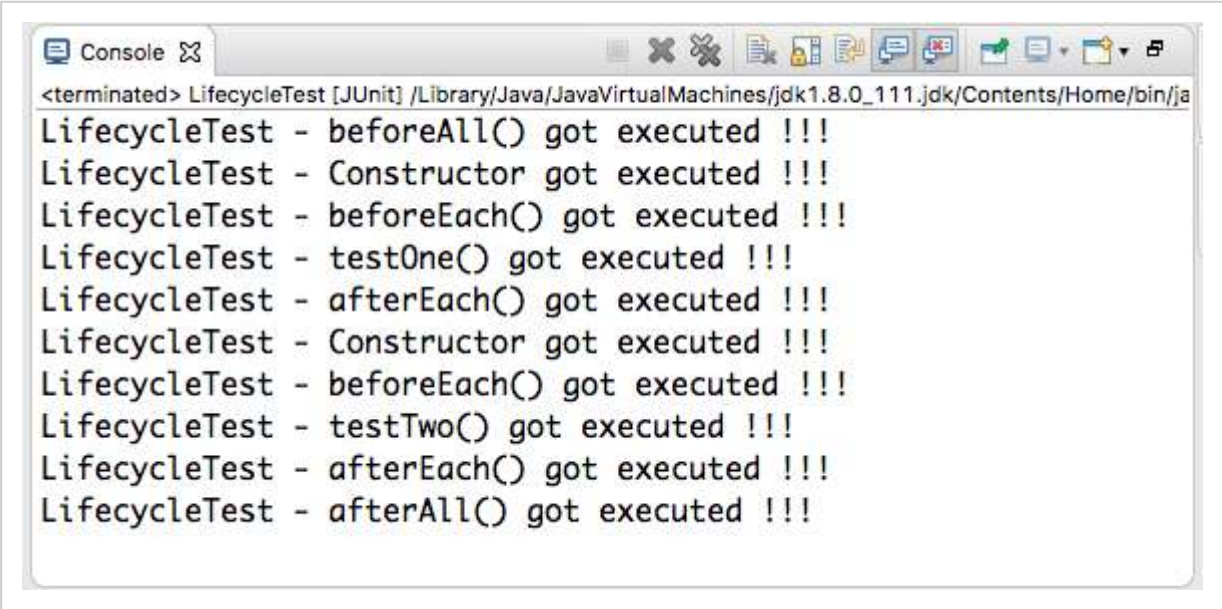
@BeforeAll and @AfterAll

Methods annotated with `@BeforeAll` and `@AfterAll` are `static` methods because, as the name suggest they are called once before all and once after all `@Test` methods. So, if in a test class there are two @Test methods, the @BeforeAll method will be called once before all test methods and similarly, the @AfterAll method will be called once, just after all @Test method gets finished. Let's look at demo taking previous lesson test class:-

```
package io.educative.junit5;

import org.junit.jupiter.api.AfterAll;
import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

public class LifecycleTest {

	@BeforeAll
	public static void beforeAll() {
		System.out.println("LifecycleTest - beforeAll() got executed !!!");
	}

	public LifecycleTest() {
		System.out.println("LifecycleTest - Constructor got executed !!!");
	}

	@BeforeEach
	public void beforeEach() {
		System.out.println("LifecycleTest - beforeEach() got executed !!!");
	}

	@Test
```

```
        public void testOne() {
                System.out.println("LifecycleTest - testOne() got executed !!!");
        }


        @Test
        public void testTwo() {
                System.out.println("LifecycleTest - testTwo() got executed !!!");
        }

        @AfterEach
        public void afterEach() {
                System.out.println("LifecycleTest - afterEach() got executed !!!");
        }

        @AfterAll
        public static void afterAll() {
                System.out.println("LifecycleTest - afterAll() got executed !!!");
        }

}
```





Output of the test

You can see in the output of the above test class, that `@BeforeAll` and `@AfterAll` has the following properties:-

1. Both are static methods.
2. Both are called once in a test lifecycle.
3. `@BeforeAll` annotated method being a class level method, it gets called even before the constructor.
4. `@AfterAll` annotated method being a class level method, it gets called after all methods get executed.

5. `@BeforeAll` annotated method is used where expensive resource initialization needs to be done such as database connection, server start etc. This resources that can be used by test methods.
6. `@AfterAll` annotated method is used where expensive resource clean up needs to be done such as, database connection terminates, server stop etc. So that these expensive resources can be cleaned up.

In the next chapter, we will discuss about `@DisplayName`.