Operating System Conditions - @DisabledOnOs and @EnabledOnOs

This lesson demonstrates how to disable or enable the test method or a complete test class using OS-level conditions.

We'll cover the following

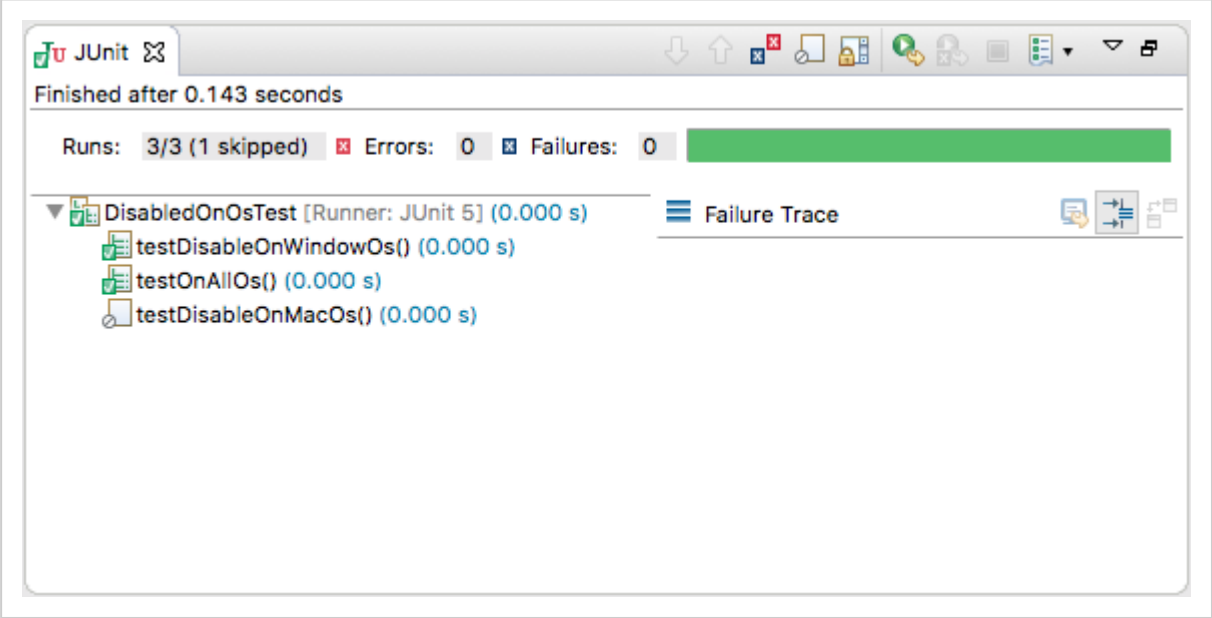
@DisabledOnOs and @EnabledOnOs

Junit 5 helps us to disable or enable test cases using various conditions. JUnit Jupiter API provides annotations in `org.junit.jupiter.api.condition` - https://junit.org/junit5/docs/5.3.2/api/org/junit/jupiter/api/condition/package-summary.html

package to enable/disable tests based on a certain condition. The annotations provided by API can be applied to test methods as well as the class itself. The two annotations which are applied to disable/enable tests based on Operating system are - `@DisabledOnOs` and `@EnabledOnOs`. Let's take a look at a demo.

```
package io.educative.junit5;

import static org.junit.jupiter.api.Assertions.assertFalse;
import static org.junit.jupiter.api.Assertions.assertTrue;

import org.junit.jupiter.api.Test;
import org.junit.jupiter.api.condition.DisabledOnOs;
import org.junit.jupiter.api.condition.OS;

public class DisabledOnOsTest {

    @Test
    void testOnAllOs() {
        assertTrue(3 > 0);
    }

    @DisabledOnOs(OS.MAC)
    @Test
    void testDisableOnMacOs() {
        assertFalse(0 > 4);
```

```
    }

    @DisabledOnOs(OS.WINDOWS)
    @Test
    void testDisableOnWindowOs() {
        assertFalse(10 > 40);
    }
}
```



Above test program has 3 test methods and `@DisabledOnOs` is applied on 2 test methods as -

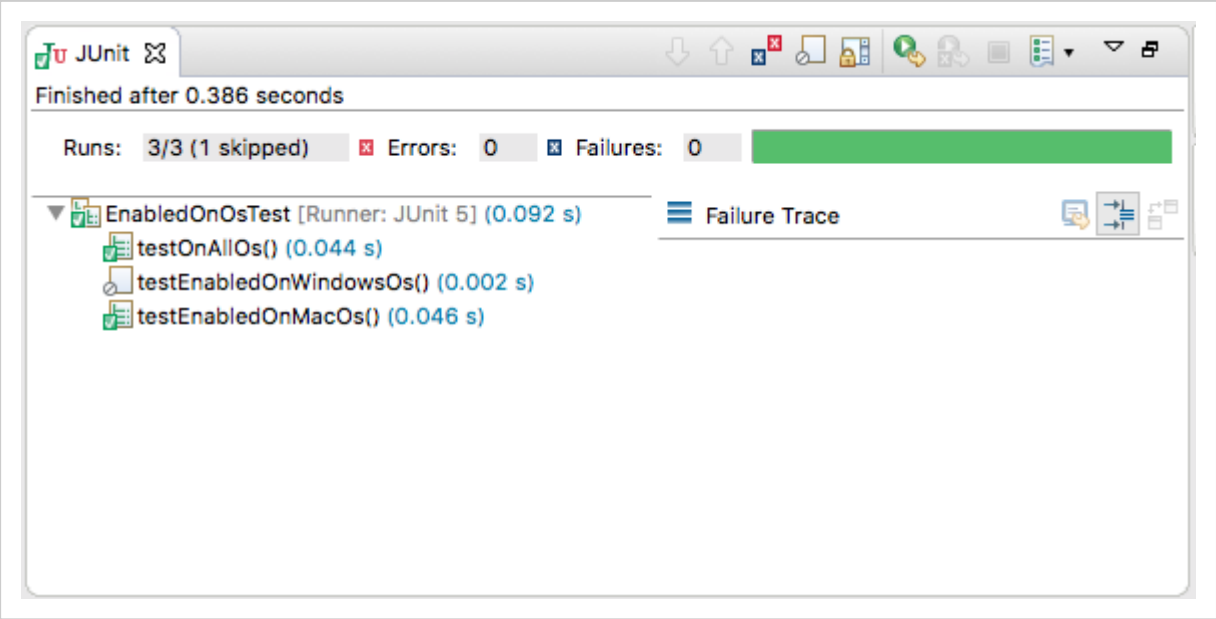
1. `testDisableOnMacOs()` - Here, `@DisabledOnOs` annotation takes in value as - `OS.MAC`. It makes the test method skip to execute on Mac operating system. It will not skip on other operating systems.
2. `testDisableOnWindowOs()` - Here, `@DisabledOnOs` annotation takes in value as - `OS.WINDOWS`. It makes the test method skip to execute on Windows operating system. It will not skip on other operating systems.

The above test methods are executed on the `Mac` operating system. Thus, the output shows that 1 test method marked as, `@DisabledOnOs(OS.MAC)` is skipped for execution.

EnabledOnOsTest.java

```
package com.hubberspot.junit5.disabled;
```

```
import static org.junit.jupiter.api.Assertions.assertFalse;
import static org.junit.jupiter.api.Assertions.assertTrue;

import org.junit.jupiter.api.Test;
import org.junit.jupiter.api.condition.EnabledOnOs;
import org.junit.jupiter.api.condition.OS;

public class EnabledOnOsTest {

    @Test
    void testOnAllOs() {
        assertTrue(3 > 0);
    }

    @EnabledOnOs(OS.WINDOWS)
    @Test
    void testEnabledOnWindowsOs() {
        assertFalse(0 > 4);
    }

    @EnabledOnOs(OS.MAC)
    @Test
    void testEnabledOnMacOs() {
        assertFalse(10 > 40);
    }
}
```



Output of tests

Above test program has 3 test methods and `@EnabledOnOs` annotation is applied on 2 test methods as -

1. `testEnabledOnWindowsOs()` - Here, `@EnabledOnOs` annotation takes in value as - `OS.WINDOWS`. It makes the test method enabled to execute only on the Windows operating system. It will get skipped on other operating systems.
2. `testEnabledOnMacOs()` - Here, `@EnabledOnOs` annotation takes in value as -

`OS.MAC`. It makes the test method enabled to execute only on Mac operating system. It will get skipped on other operating systems.

The above test methods are executed on the `Mac` operating system. Thus, the output shows that 1 test method marked as, `@EnabledOnOs(OS.WINDOWS)` is skipped for execution. It will execute when tests are run on Windows operating system.

In the next lesson we will learn about `DisabledOnJre` and `EnableOnJre` based conditions.