

# Java Runtime Environment Conditions - @DisabledOnJre and @EnabledOnJre

This lesson demonstrates how to disable or enable test methods or a complete test class using JRE level conditions.

## We'll cover the following



- @DisabledOnJre and @EnabledOnJre

## @DisabledOnJre and @EnabledOnJre #

JUnit 5 helps us to disable or enable test cases using various conditions. JUnit Jupiter API provides annotations in `org.junit.jupiter.api.condition` 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 the particular version of the Java Runtime Environment (JRE) are - `@DisabledOnJre` and `@EnabledOnJre`. Let's take a look at a demo.

### DisabledOnJreTest.java

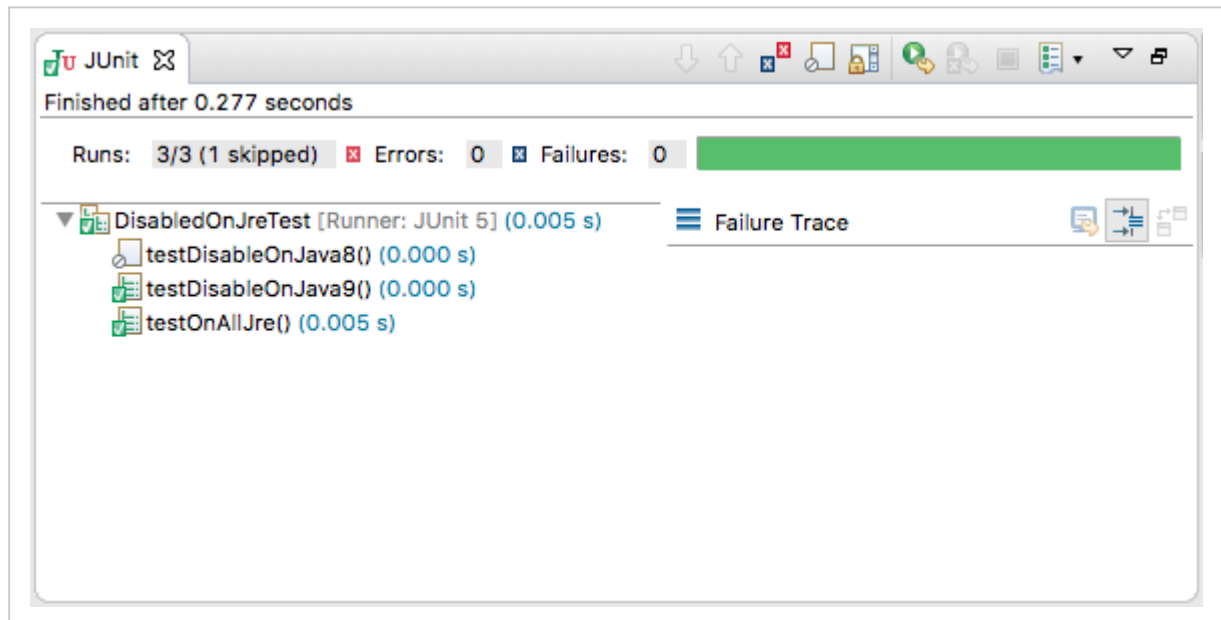
```
1 package com.hubberspot.junit5.disabled;
2
3 import static org.junit.jupiter.api.Assertions.assertFalse;
4 import static org.junit.jupiter.api.Assertions.assertTrue;
5
6 import org.junit.jupiter.api.Test;
7 import org.junit.jupiter.api.condition.DisabledOnJre;
8 import org.junit.jupiter.api.condition.JRE;
9
10 public class DisabledOnJreTest {
11
12     @Test
13     void testOnAllJre() {
14         assertTrue(3 > 0);
15     }
16
17     @DisabledOnJre(JRE.JAVA_8)
18     @Test
19     void testDisableOnJava8() {
20         assertFalse(0 > 4);
```



```

21     }
22
23     @DisabledOnJre(JRE.JAVA_9)
24     @Test
25     void testDisableOnJava9() {
26         assertFalse(10 > 40);
27     }
28 }
29

```



Above test program has 3 test methods and `@DisabledOnJre` is applied on 2 test methods as.

1. `testDisableOnJava8()` - Here, `@DisabledOnJre` annotation takes in value as `JRE.JAVA_8`. It makes the test method skip to execute on Java 8 runtime environment. It will not skip on other Java runtime environments.
2. `testDisableOnJava9()` - Here, `@DisabledOnJre` annotation takes in value as `JRE.JAVA_9`. It makes the test method skip to execute on Java 9 runtime environment. It will not skip on other Java runtime environments.

The above test methods are executed on the `Java 8` runtime environment. Thus, the output shows that 1 test method marked as, `@DisabledOnJre(JRE.JAVA_8)` is skipped for execution.

 EnabledOnJreTest.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.EnabledOnJre;
import org.junit.jupiter.api.condition.JRE;

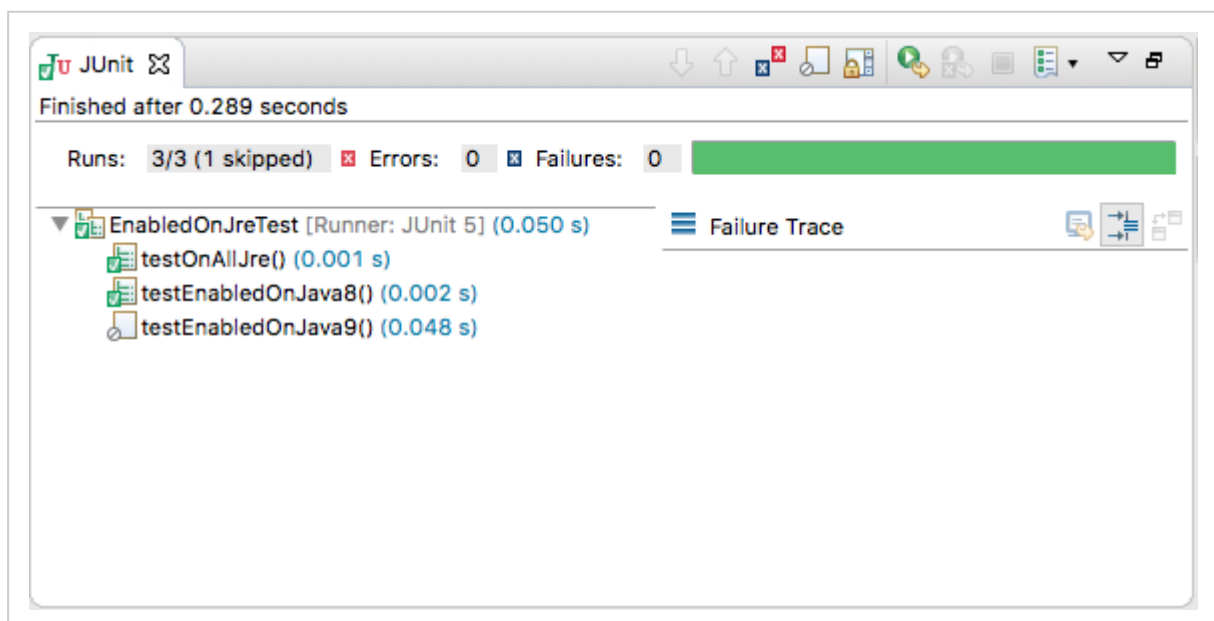
public class EnabledOnJreTest {

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

    @EnabledOnJre(JRE.JAVA_8)
    @Test
    void testEnabledOnJava8() {
        assertFalse(0 > 4);
    }

    @EnabledOnJre(JRE.JAVA_9)
    @Test
    void testEnabledOnJava9() {
        assertFalse(10 > 40);
    }
}

```



Output of tests

Above test program has 3 test methods and `@EnabledOnJre` annotation is applied on 2 test methods as.

1. `testEnabledOnJava8()` - Here, `@EnabledOnJre` annotation takes in value as - `JRE.JAVA_8`. It makes the test method enabled to execute only on Java 8 runtime environment. It will get skipped on other Java runtime environments.
2. `testEnabledOnJava9()` - Here, `@EnabledOnJre` annotation takes in value as - `JRE.JAVA_9`. It makes the test method enabled to execute only on Java 9 runtime environment. It will get skipped on other Java runtime environments.

runtime environment. It will get skipped on other java runtime environments.

The above test methods are executed on `Java 8` runtime environment. Thus, the output shows that 1 test method marked as, `@EnabledOnJre(JRE.JAVA_9)` is skipped for execution. It will execute when tests are run on Java 9 runtime environment.