

Environment Variable Conditions - @DisabledIfEnvironmentVariable and @EnabledIfEnvironmentVariable


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

We'll cover the following

- @DisabledIfEnvironmentVariable and @EnabledIfEnvironmentVariable

@DisabledIfEnvironmentVariable and @EnabledIfEnvironmentVariable

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 use system environment properties and specified regex to disable or enable tests are - `@DisabledIfEnvironmentVariable` and `@EnabledIfEnvironmentVariable`. Let's take a look at a demo.

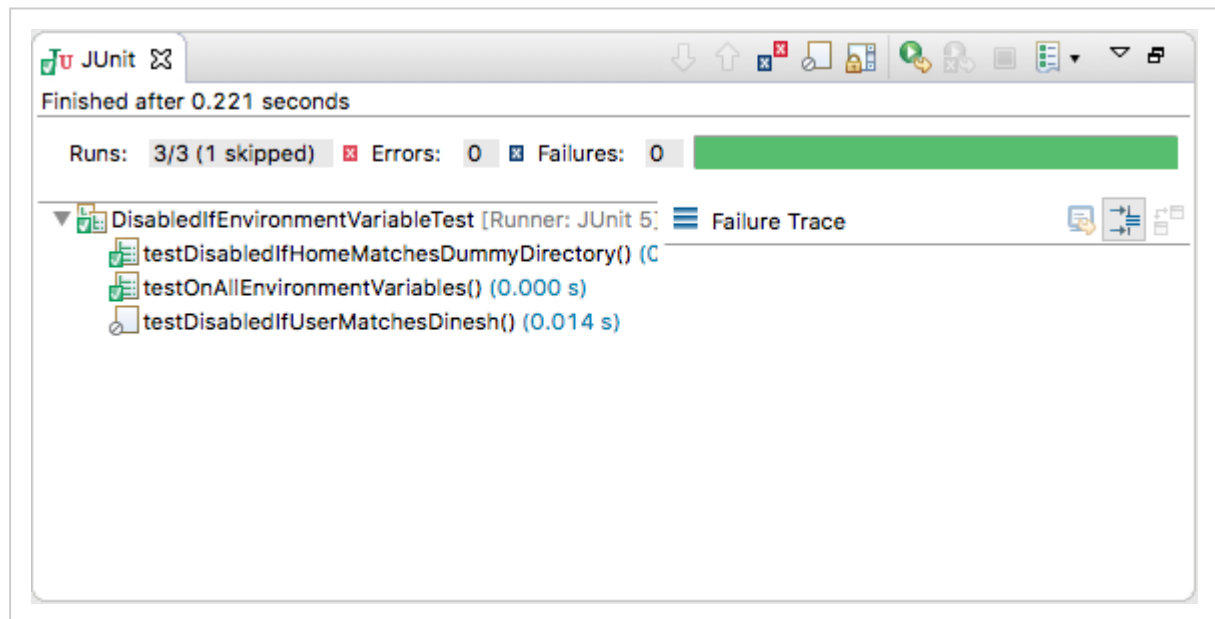
 DisabledIfEnvironmentVariableTest.java

```
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.DisabledIfEnvironmentVariable;
8
9 public class DisabledIfEnvironmentVariableTest {
10
11     @Test
12     void testOnAllEnvironmentVariables() {
13         assertTrue(3 > 0);
14     }
15
16     @DisabledIfEnvironmentVariable(named="USER", matches="dinesh")
17     @Test
```

```

18     void testDisabledIfUserMatchesDinesh() {
19         assertFalse(0 > 4);
20     }
21
22     @DisabledIfEnvironmentVariable(named="HOME", matches="/dummies/home")
23     @Test
24     void testDisabledIfHomeMatchesDummyDirectory() {
25         assertFalse(10 > 40);
26     }
27 }
28

```



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

1. `testDisabledIfUserMatchesDinesh()` - Here, `@DisabledIfEnvironmentVariable` annotation takes in two attributes such as `named` and `matches`. The value provided to `named` attribute is `"USER"` and value provided to `matches` attribute is `"dinesh"`. It makes the test method skip to execute if environment variable by name "USER" matches "dinesh".
2. `testDisabledIfHomeMatchesDummyDirectory()` - Here, `@DisabledIfEnvironmentVariable` annotation takes in two attributes such as `named` and `matches`. The value provided to `named` attribute is `"HOME"` and value provided to `matches` attribute is `"/dummies/home"`. It makes the test method skip to execute if environment variable by name "HOME" matches `"/dummies/home"`.

The above test methods are executed on environment variables as -

1. USER - dinesh

2. HOME - /Users/dinesh

Thus, output shows that 1 test method marked as,

`@DisabledIfEnvironmentVariable(named="USER", matches="dinesh")` is skipped for execution.

EnabledIfEnvironmentVariableTest.java

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

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

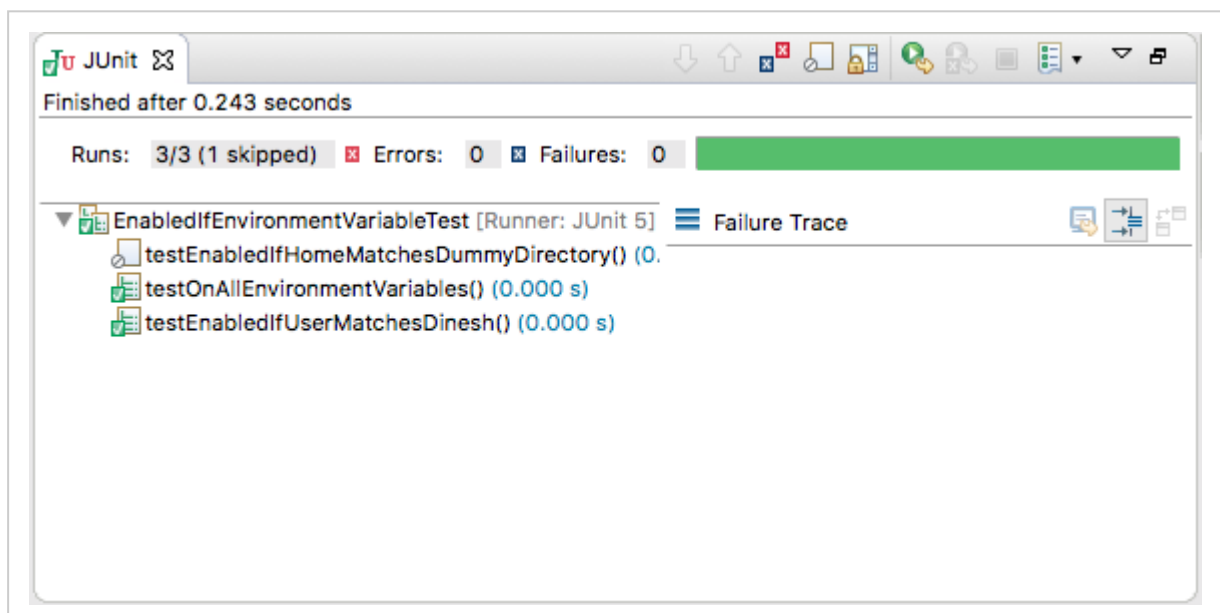
import org.junit.jupiter.api.Test;
import org.junit.jupiter.api.condition.EnabledIfEnvironmentVariable;

public class EnabledIfEnvironmentVariableTest {

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

    @EnabledIfEnvironmentVariable(named="USER", matches="dinesh")
    @Test
    void testEnabledIfUserMatchesDinesh() {
        assertFalse(0 > 4);
    }

    @EnabledIfEnvironmentVariable(named="HOME", matches="/dummies/home")
    @Test
    void testEnabledIfHomeMatchesDummyDirectory() {
        assertFalse(10 > 40);
    }
}
```



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

1. `testEnabledIfUserMatchesDinesh()` - Here, `@EnabledIfEnvironmentVariable` annotation takes in two attributes such as `named` and `matches`. The value provided to `named` attribute is `"USER"` and value provided to `matches` attribute is `"dinesh"`. It makes the test method enable to execute if environment variable by name `"USER"` matches `"dinesh"`.
2. `testEnabledIfHomeMatchesDummyDirectory()` - Here, `@EnabledIfEnvironmentVariable` annotation takes in two attributes such as `named` and `matches`. The value provided to `named` attribute is `"HOME"` and value provided to `matches` attribute is `"/dummies/home"`. It makes the test method enable to execute if environment variable by name `"HOME"` matches `"/dummies/home"`.

The above test methods are executed on environment variables as -

1. **USER - dinesh**
2. **HOME - /Users/dinesh**

Thus, output shows that 1 test method marked as,

`@EnabledIfEnvironmentVariable(named="HOME", matches="/dummies/home")` is skipped for execution.

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