

assumeTrue() and assumeFalse() method

This lesson demonstrates how to use `assumeTrue` and `assumeFalse` methods in JUnit 5 to make conditional assumptions.

WE'LL COVER THE FOLLOWING ^

- `assumeTrue()`
- Demo
- `assumeFalse()`
- Demo

`assumeTrue()`

Assumptions API in JUnit 5 has a static utility method called as, `assumeTrue()`. It validates the given assumption to true.

- if the assumption is **true** then test proceeds to execution.
- if the assumption is **false** then test execution is aborted.

There are basically three useful overloaded methods for `assumeTrue`.

```
// boolean assumption to validate
public static void assumeTrue(boolean assumption) throws TestAbortedException
public static void assumeTrue(boolean assumption, Supplier<String> messageSupplier) throws TestAbortedException
public static void assumeTrue(boolean assumption, String message) throws TestAbortedException

// BooleanSupplier to provide boolean assumption to validate
public static void assumeTrue(BooleanSupplier assumptionSupplier) throws TestAbortedException
public static void assumeTrue(BooleanSupplier assumptionSupplier, String message) throws TestAbortedException
public static void assumeTrue(BooleanSupplier assumptionSupplier, Supplier<String> messageSupplier) throws TestAbortedException
```

Demo

Let's look into the usage of the above methods.

```
package io.educative.junit5;
```

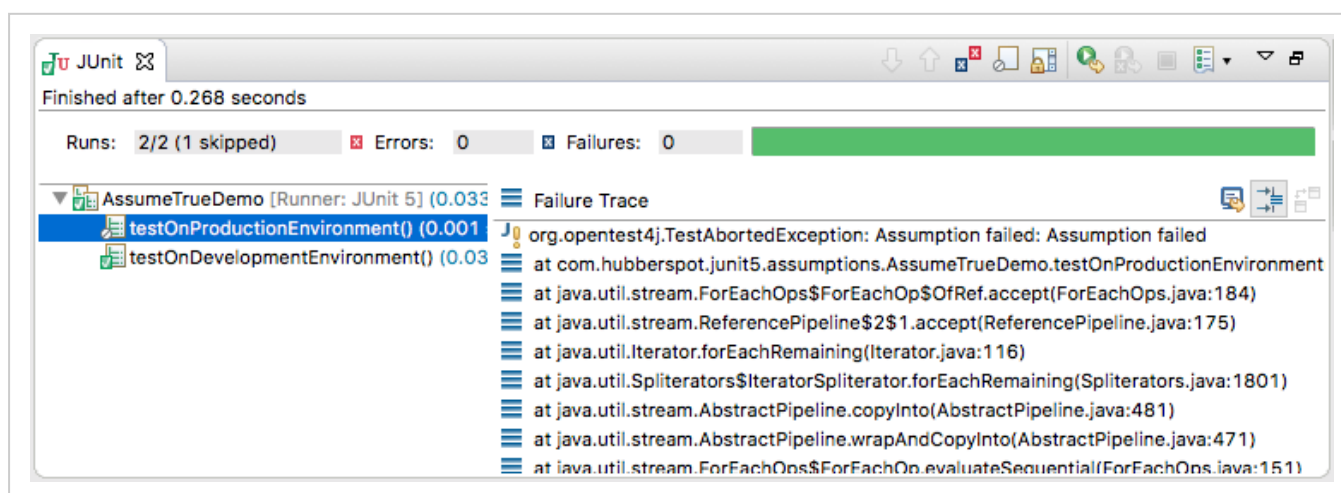
```
import static org.junit.jupiter.api.Assertions.assertTrue;

import org.junit.jupiter.api.Test;

public class AssumeTrueDemo {

    @Test
    void testOnDevelopmentEnvironment() {
        System.setProperty("ENV", "DEV");
        assertTrue("DEV".equals(System.getProperty("ENV")));
        //remainder of test will proceed
    }

    @Test
    void testOnProductionEnvironment() {
        System.setProperty("ENV", "PROD");
        assertTrue("DEV".equals(System.getProperty("ENV")), "Assumption failed");
        // remainder of test will be aborted
    }
}
```



assumeFalse()

Assumptions API in JUnit 5 has a static utility method called as `assumeFalse()`. It validates the given assumption to false.

- if the assumption is **false** then test proceeds to execution.
- if the assumption is **true** then test execution is aborted.

There are basically three useful overloaded methods for `assumeFalse`.

```
// boolean assumption to validate
public static void assumeFalse(boolean assumption) throws TestAbortedException
public static void assumeFalse(boolean assumption, Supplier<String> messageSupplier) throws TestAbortedException
public static void assumeFalse(boolean assumption, String message) throws TestAbortedException
```

```
// BooleanSupplier to provide boolean assumption to validate
public static void assumeFalse(BooleanSupplier assumptionSupplier) throws TestAbortedException {
    assumeFalse(assumptionSupplier, "Assumption failed");
}

public static void assumeFalse(BooleanSupplier assumptionSupplier, String message) throws TestAbortedException {
    assumeFalse(assumptionSupplier, Supplier<String> messageSupplier);
}

public static void assumeFalse(BooleanSupplier assumptionSupplier, Supplier<String> messageSupplier) throws TestAbortedException {
    if (!assumptionSupplier.getAsBoolean()) {
        throw new TestAbortedException(messageSupplier.get());
    }
}
```

Demo

Let's look into the usage of the above methods.

```
package io.educative.junit5;

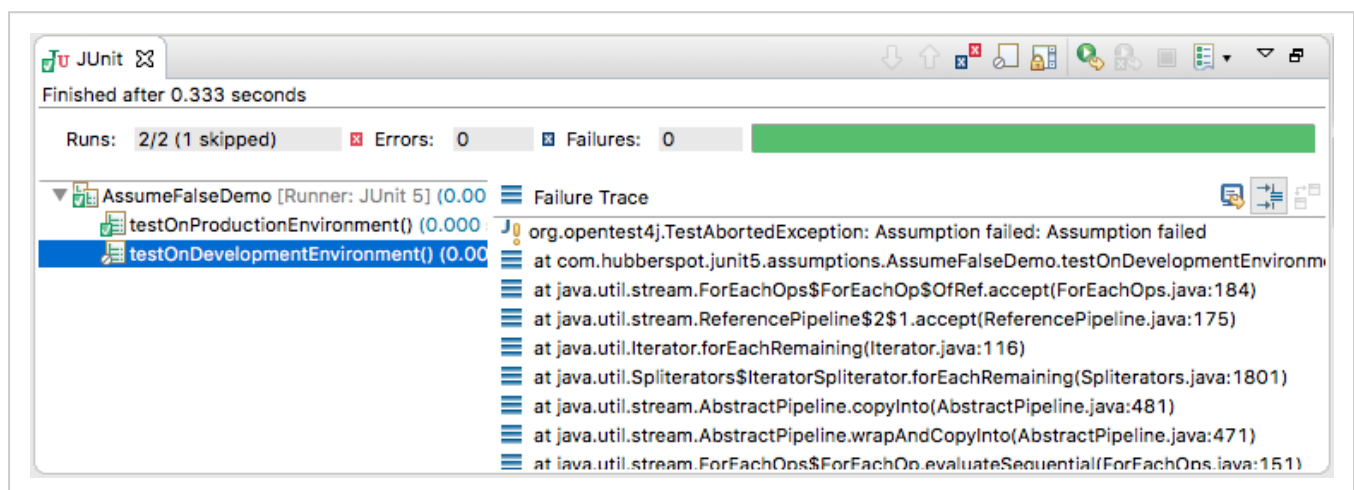
import static org.junit.jupiter.api.Assertions.assumeFalse;

import org.junit.jupiter.api.Test;

public class AssumeFalseDemo {

    @Test
    void testOnDevelopmentEnvironment() {
        System.setProperty("ENV", "DEV");
        assumeFalse("DEV".equals(System.getProperty("ENV")), "Assumption failed");
        //remainder of test will be aborted
    }

    @Test
    void testOnProductionEnvironment() {
        System.setProperty("ENV", "PROD");
        assumeFalse("DEV".equals(System.getProperty("ENV")));
        // remainder of test will proceed
    }
}
```



In the next lesson we will learn about `assumingThat()` method.

