Name : Sujeewa Sandeepa
Topic : Conditional executions and assumptions
Date : 2021-07-12

We can use different conditional executions to control where and how our tests run.

```
@EnabledOnOS(OS.LINUX)
```

If we use this one, the particular test is gonna run only on Linux. In other operating systems it’s just going to be disabled.

```
@EnabledOnJre(JRE.JAVA_11)
```

Here we can use this conditional execution to make some test cases only enabled in particular JREs, this can come in handy too. Here in this case the particular test code is only gonna run in jre version 11.

And there are other conditional executions like

```
@EnabledIf
@EnabledIfSystemProperty
@EnabledIfEnvironmentVariable
```

We can use assumptions too, but in a little different way.

Here let’s say if we are assuming that a value is true, it’s need to be that way for the test to be valid, we can say it like this.

```
@Test
void testDivide() {
     assumeTrue(value);
     assertThrows(…..);
}
```

So here we assume that the value is true and runs the test, but in some way if the value isn’t true, this test won’t run. It won’t fail, it just won’t run.

And we can use AssertAll to run a bunch of assertions in just one statement.

Using AssertAll

We can use this to run many assertions as I said before. An example will look like this,

```
@Test
@DisplayName(“multiply method”)
void testMultiply() {
     assertAll(
          () → assertEquals(4, mathUtils.multiply(2, 2)),
          () → assertEquals(0, muthUtils.multiply(2, 0)),
          () → assertEquals(-2, muthUtils.multiply(2, -1))
          );
}
```

Writing Nested test classes

And we can write nested tests to arrange everything nicely. We can use @Nested to do so.

```
@Nested
@DisplayName(“Add method”)
class AddTest {

     @Test
     @DisplayName(“Adding two positive numbers”)
     void testAddPostive() {
     < content >
     }

     @Test
     @DisplayName(“Adding two negative numbers”)
     void testAddNegative() {
     < content >
     }
}
```

This is a good way to organize our test code base. This is more clear and much more organized than asserting a lot of things inside one test.