Unit and Integration Tests With Maven and JUnit Categories

by John Dobie MVB · **May. 09, 12 · Java Zone · Not set**

Heads up...this article is old!

Technology moves quickly and this article was published 6 years ago. Some or all of its contents may be outdated.

This example shows how to split unit and integration tests using Maven and JUnit categories. It is especially useful for existing test suites and can be implemented in minutes.

Why use this?

My previous post showed how we to use a maven profile to split unit and integration

annotation.http://kentbeck.github.com/junit/javadoc/latest/org/junit/experimental/categories/Categories.html

Define the Marker Interface

The first step in grouping a test using categories is to create a marker interface. This interface will be used to mark all of the tests that you want to be run as integration tests.

```
public interface IntegrationTest {}
```

Mark your test classes

Add the category annotation to the top of your test class. It takes the name of your new interface.

```
import org.junit.experimental.categories.Category;
@Category(IntegrationTest.class)
public class ExampleIntegrationTest{

 @Test
 public void longRunningServiceTest() throws Exception {

 }
}
```

```
  <version>2.12</version>
 </dependency>
</dependencies>
<configuration>
 <includes>
  <include>**/*.class</include>
 </includes>
 <excludedgroups>com.test.annotation.type.IntegrationTest</excludedgr
</configuration>
</plugin>
```

There are 2 very important parts. The first is to configure surefire to exclude all of the integrations tests.

```
<excludedgroups>com.test.annotation.type.IntegrationTest</excludedgro
```

Surefire will run all of your tests, except those marked as an integration test. The other important part is to make sure the surefire plugin uses the correct JUnit provider. The JUnit47 provider is needed to correctly detect the categories.

```
<dependencies>
 <dependency>
  <groupid>org.apache.maven.surefire</groupid>
  <artifactid>surefire-junit47</artifactid>
```

```
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
```

Configure Maven Integration Tests

Again the configuration for this is very simple. We use the standard failsafe plugin and configure it to only run the integration tests.

```
<plugin>
 <artifactid>maven-failsafe-plugin</artifactid>
 <version>2.12</version>
 <dependencies>
  <dependency>
   <groupid>org.apache.maven.surefire</groupid>
   <artifactid>surefire-junit47</artifactid>
   <version>2.12</version>
  </dependency>
 </dependencies>
 <configuration>
  <groups>com.test.annotation.type.IntegrationTest</groups>
```

And again the JUnit provider must be correctly configured.

```
<dependencies>
 <dependency>
  <groupid>org.apache.maven.surefire</groupid>
  <artifactid>surefire-junit47</artifactid>
  <version>2.12</version>
 </dependency>
</dependencies>
```

That's it!

Running the integration tests

We can now run the whole build.

```
mvn clean install
```

This time as well as the unit test running, the integration tests are run during the integration-test phase.

```
-------------------------------------------------------
 T E S T S
```