

```
<!-- Entornos de Desarrollo -->
```

JUnit {

```
<Para="DAM/DAW"/>
```

```
<Prof="Antonio Calabuig"/>
```

}



Configuración en IDEs

01

Visual Studio Code

02

IntelliJ IDEA

Introducción {

JUnit es un framework de testeo de código.

Como todo framework, éste se puede utilizar en todos los IDEs de desarrollo que trabajen con JAVA.

En esta unidad nos centraremos exclusivamente sobre:

- Visual Studio Code
- IntelliJ IDEA

Son los dos IDE's más usados y por tanto aprenderemos como incluir JUnit en ambos IDEs

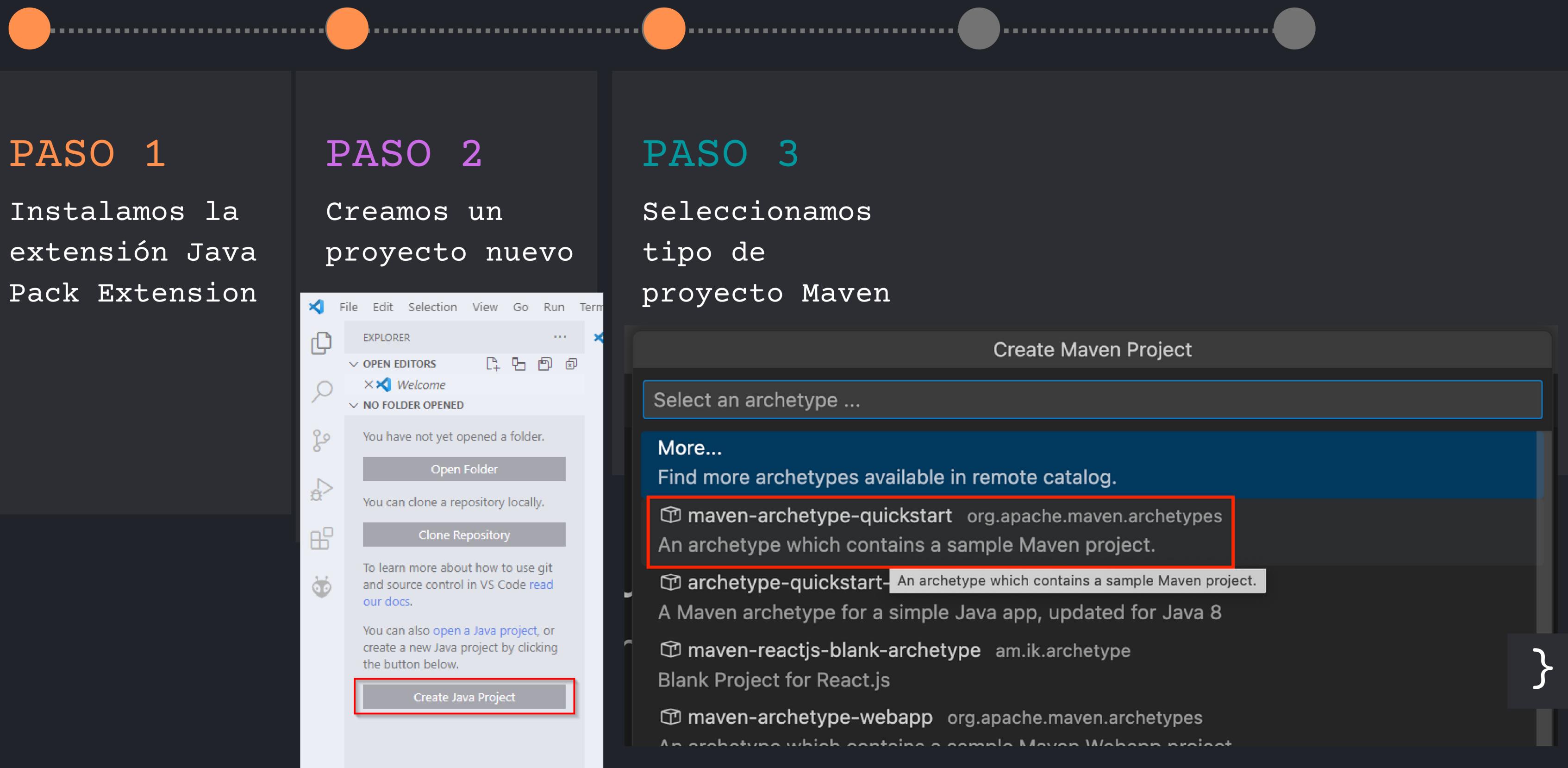
The screenshot shows the IntelliJ IDEA interface with the following details:

- Project View:** Shows the project structure under "junit-tutorial". It includes a "src" folder with "main" and "test" subfolders. "main/java/org/antonio/Main.java" is shown, and "test/java/org/antonio/MainTest.java" is selected. Other files like "pom.xml", ".gitignore", and "junit-tutorial.iml" are also listed.
- Code Editor:** Displays the content of "MainTest.java". The code defines a class "MainTest" with two test methods: "shouldAnswerWithTrue()" and "shouldAnswerWithFalse()". Both methods use the "assertTrue" and "assertFalse" assertions from the "org.junit.Assert" static import.
- Bottom Status Bar:** Shows "Ln 20, Col 6 Spaces: 4 UTF-8 LF {} Java".

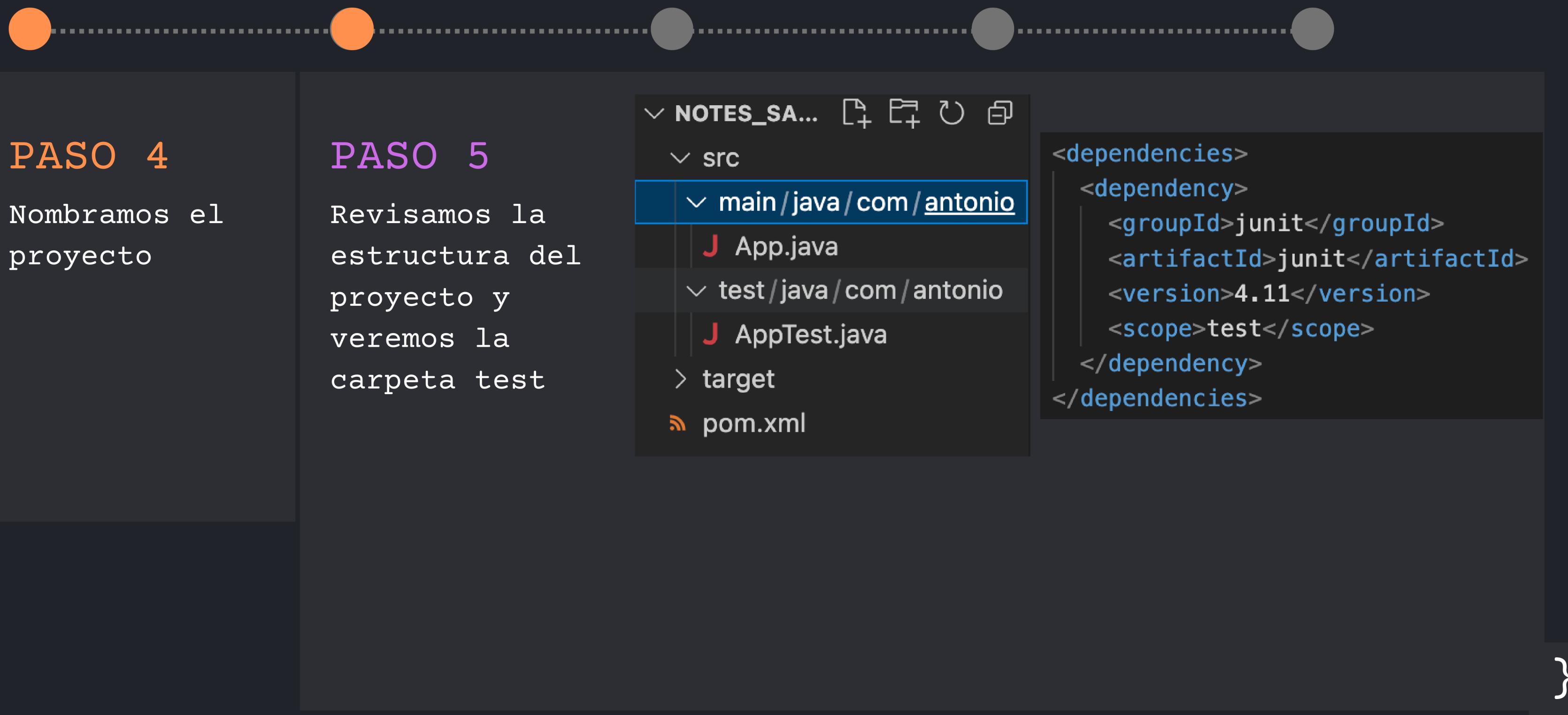
The screenshot shows the IntelliJ IDEA interface with the following details:

- Project View:** Shows the project structure under "PRUEBA". It includes a "src" folder with "lib" and "main/java/com/prueba" subfolders. "main/java/com/prueba/App.java" is shown, and "test/java/com/prueba/AppTest.java" is selected. Other files like "pom.xml" and "target" are also listed.
- Code Editor:** Displays the content of "AppTest.java". The code defines a class "AppTest" with a single test method "shouldAnswerWithTrue()", which uses the "assertTrue" assertion.
- Bottom Status Bar:** Shows "Ln 20, Col 6 Spaces: 4 UTF-8 LF {} Java".

Visual Studio Code {



Visual Studio Code {



Visual Studio Code {

PASO 6

Prueba de ejecución de los tests.

AppTest.java

```
1 package com.antonio;
2
3 import static org.junit.Assert.assertTrue;
4
5 import org.junit.Test;
6
7 /**
8 * Unit test for simple App.
9 */
10 public class AppTest
11 {
12     void com.antonio.AppTest.shouldAnswerWithTrue()
13     Rigorous Test :-)
14     shouldAnswerWithTrue() (Not yet run).
15
16     public void shouldAnswerWithTrue()
17     {
18         assertTrue( condition:true );
19     }
20 }
21
```

✓	✓	notes_sample	1.0ms
✓	✓	{ } com.antonio	1.0ms
✓	✓	AppTe	▶ ↗ ↖
✓	cube	shouldAnswerWith	

IntelliJ IDEA {



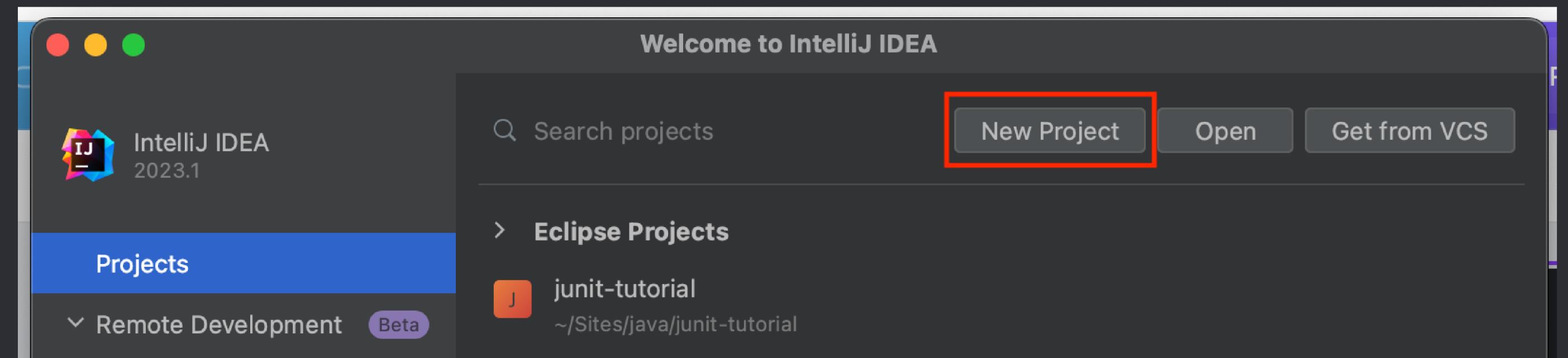
PASO 1

Instalamos
IntelliJ IDEA
Community,
producto de
Jetbrains.

Podemos
crearnos una
cuenta para
educación
usando nuestro
correo de
alu.edu.gva.es

PASO 2

Creamos un
proyecto nuevo



}

IntelliJ IDEA {



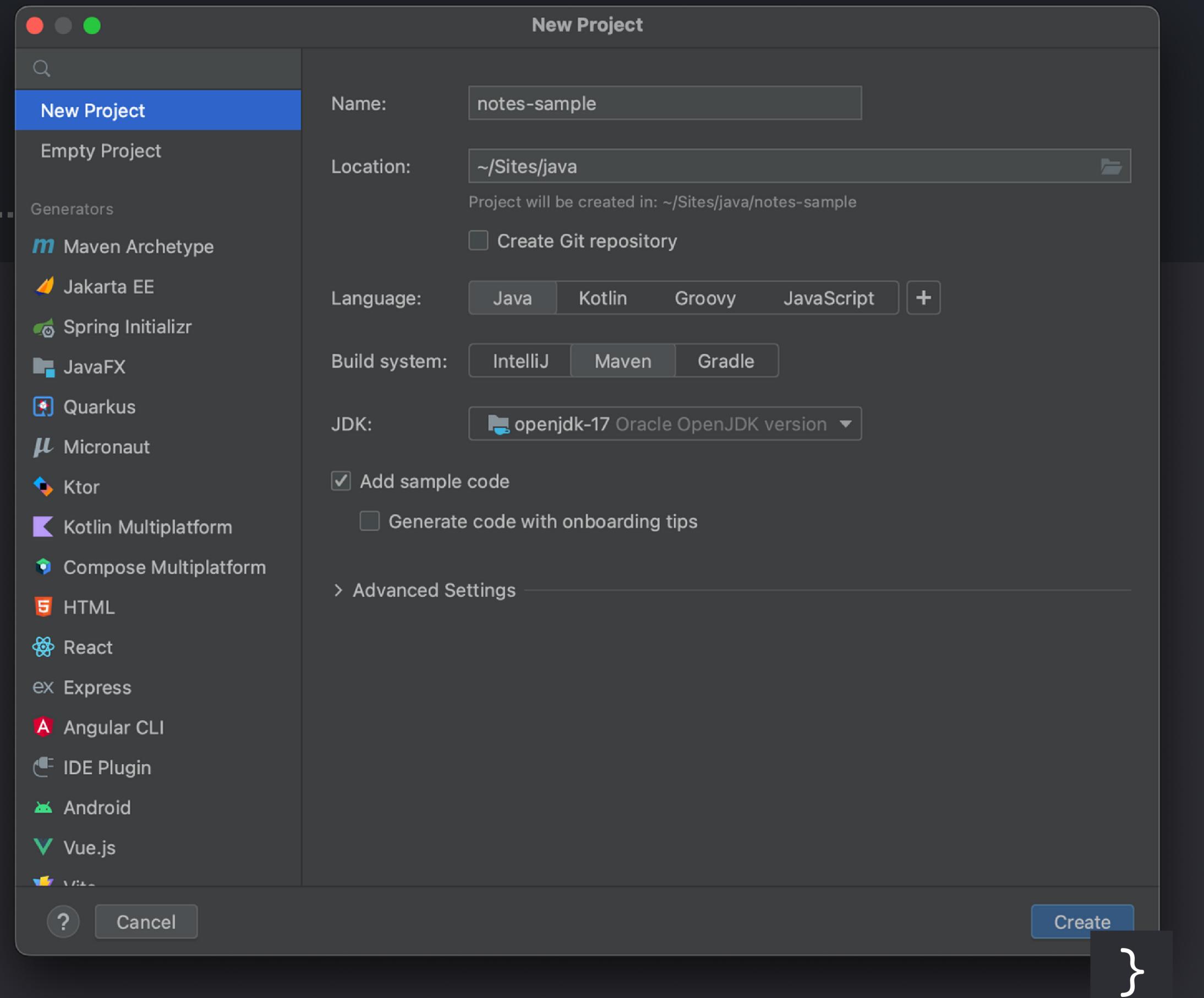
PASO 1

Instalamos
IntelliJ IDEA
Community,
producto de
Jetbrains.

Podemos
crearnos una
cuenta para
educación
usando nuestro
correo de
@alu.edu.gva.es

PASO 2

Creamos un
proyecto nuevo

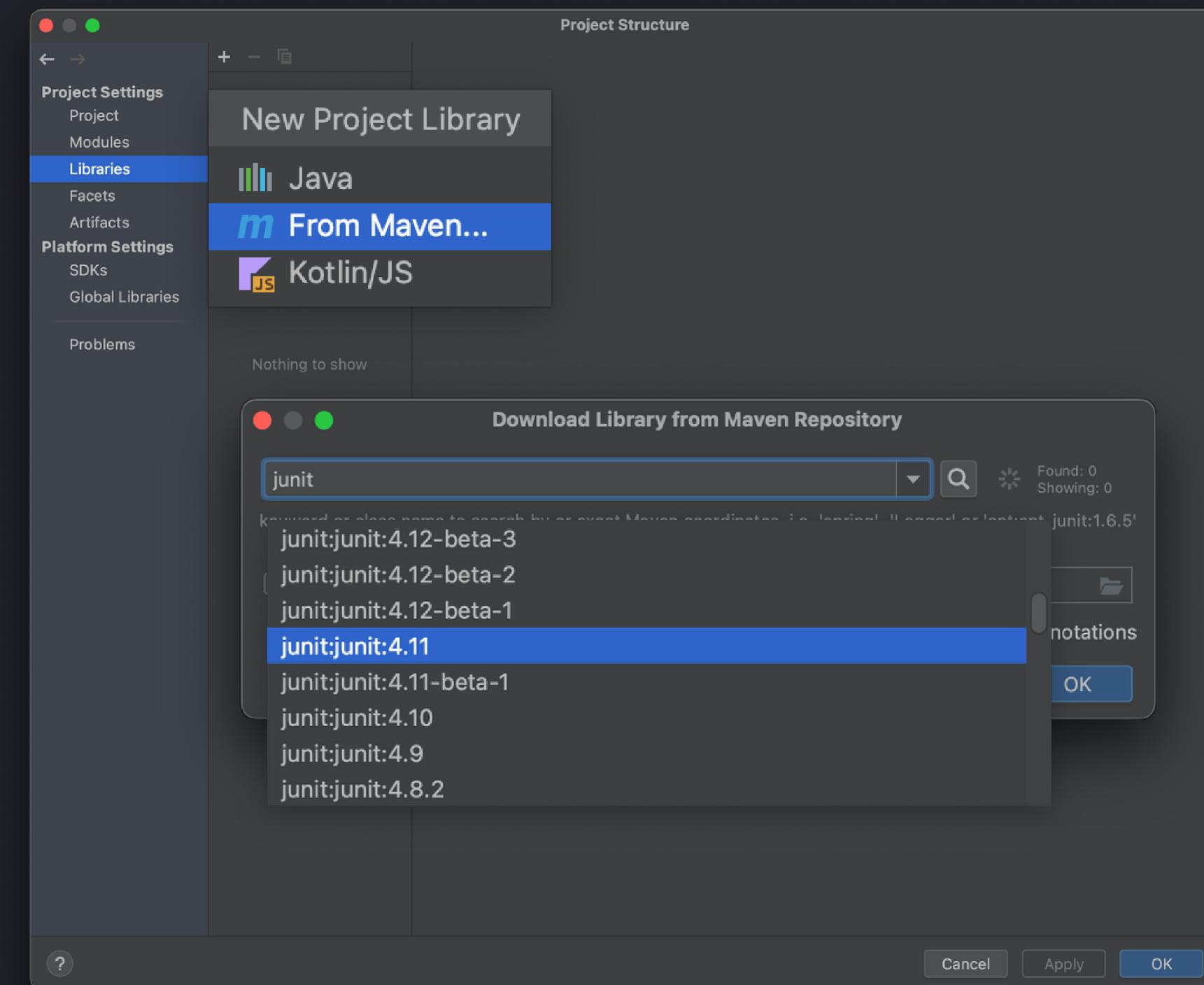
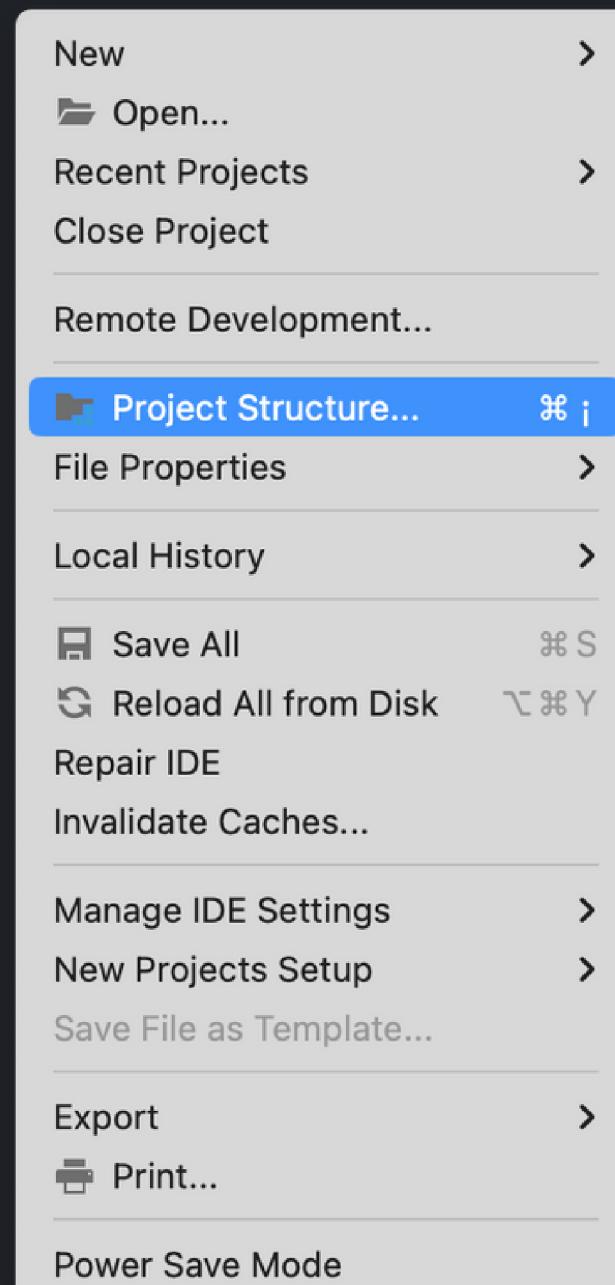


IntelliJ IDEA {

PASO 3

Añadimos JUnit
al proyecto que
hemos creado.

Varios pasos a
seguir.



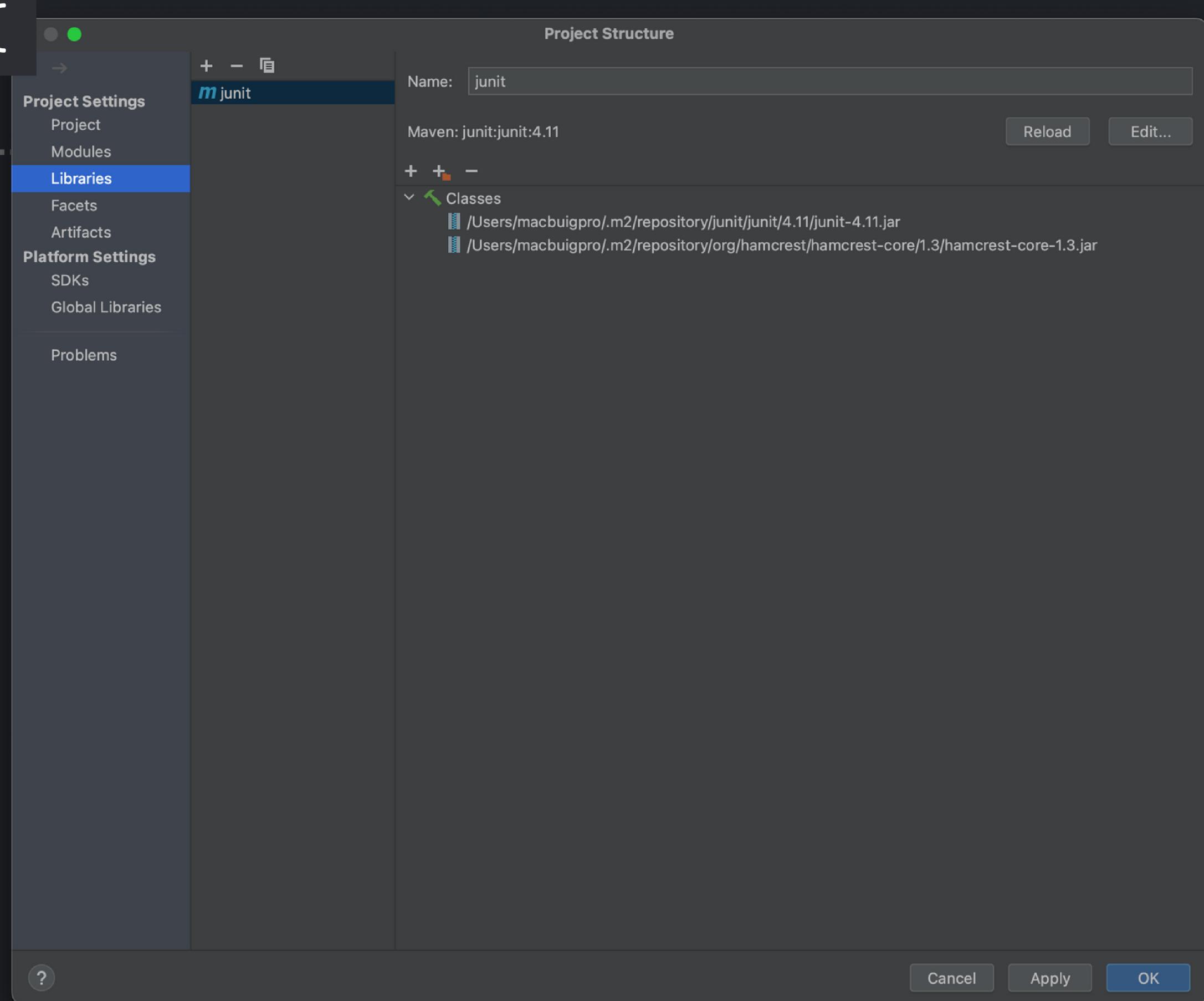
}

IntelliJ IDEA {

PASO 3

Añadimos JUnit
al proyecto que
hemos creado.

Varios pasos a
seguir.

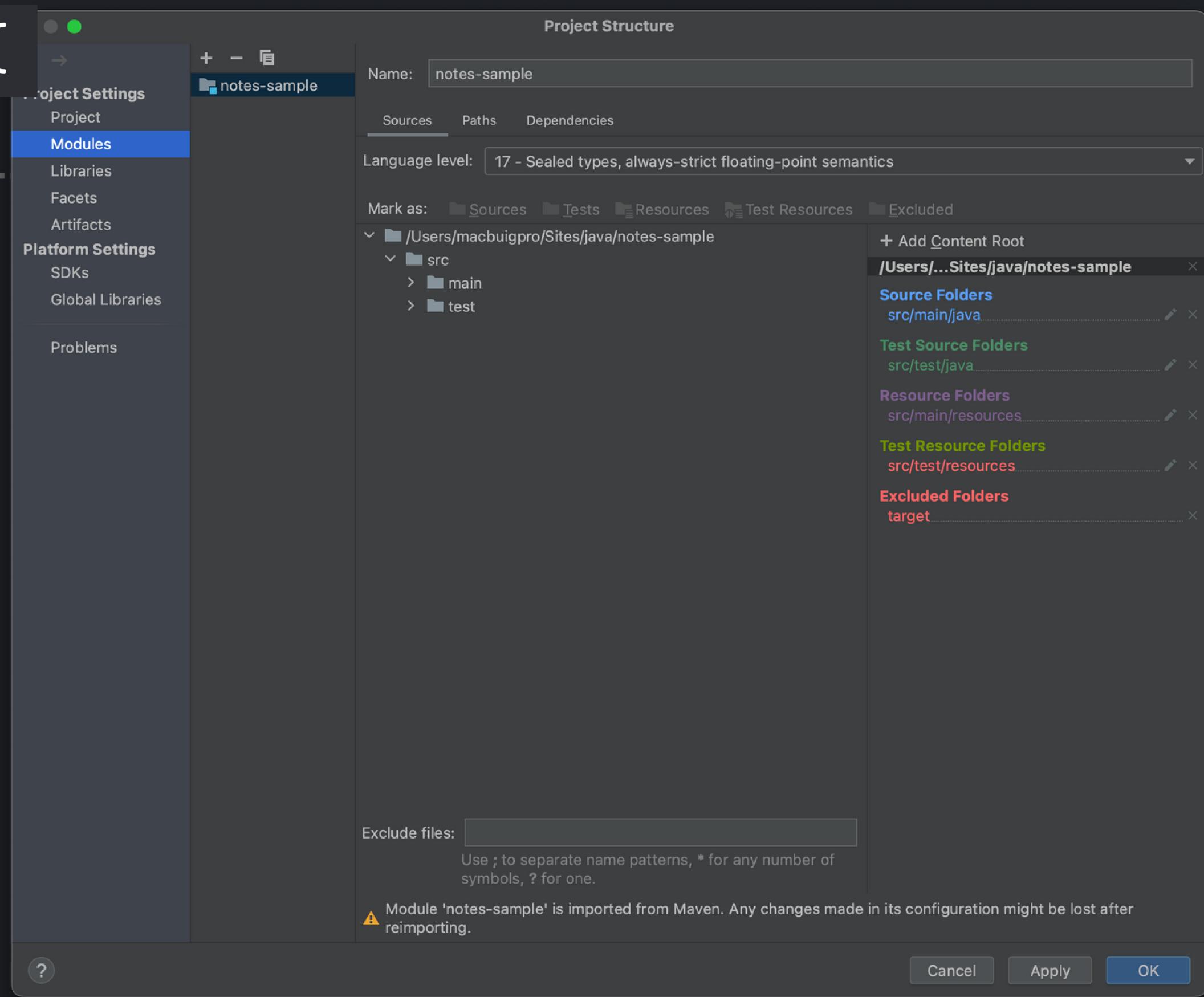


IntelliJ IDEA {

PASO 3

Añadimos JUnit
al proyecto que
hemos creado.

Varios pasos a
seguir.



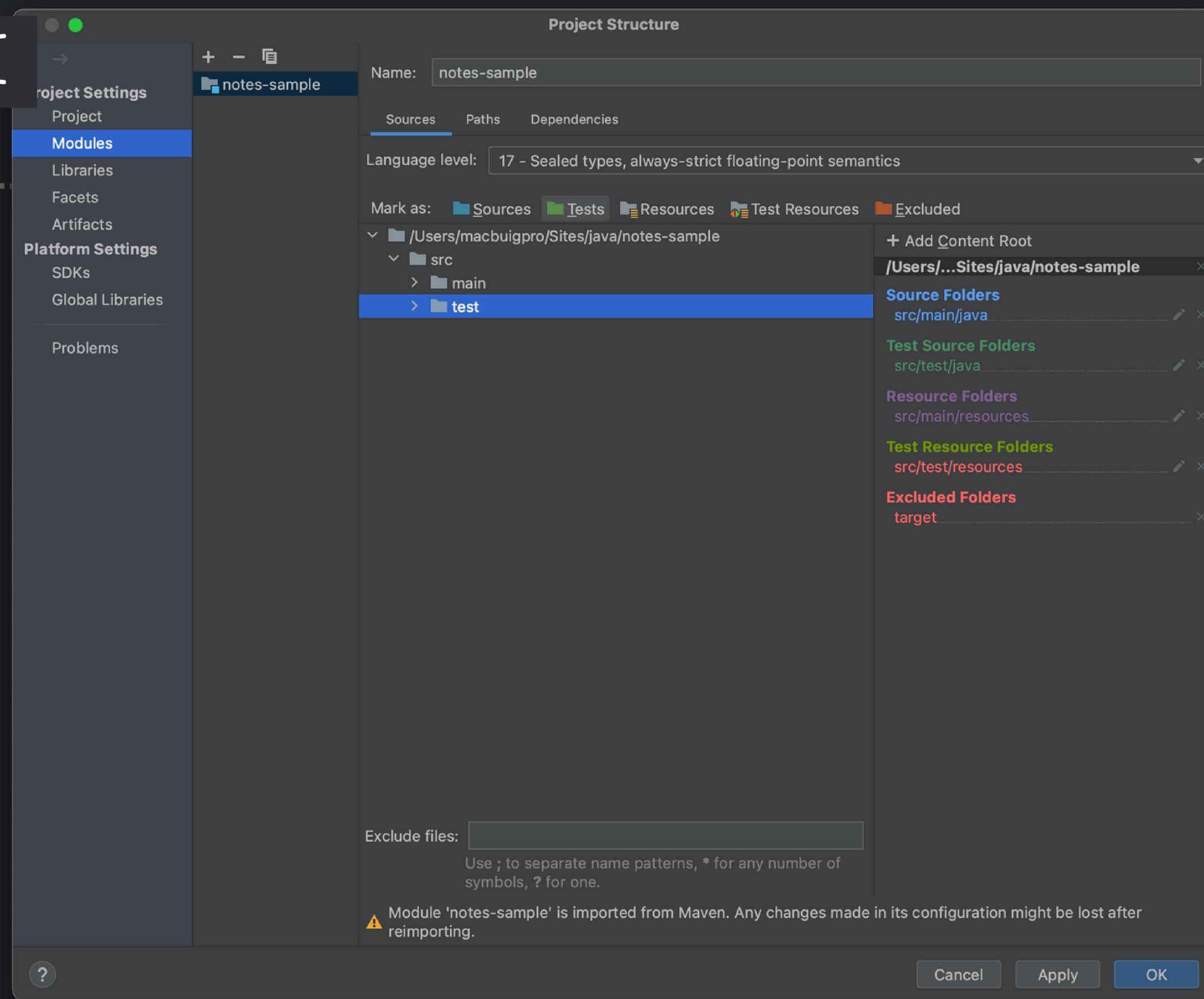
}

IntelliJ IDEA {

PASO 3

Añadimos JUnit
al proyecto que
hemos creado.

Varios pasos a
seguir.



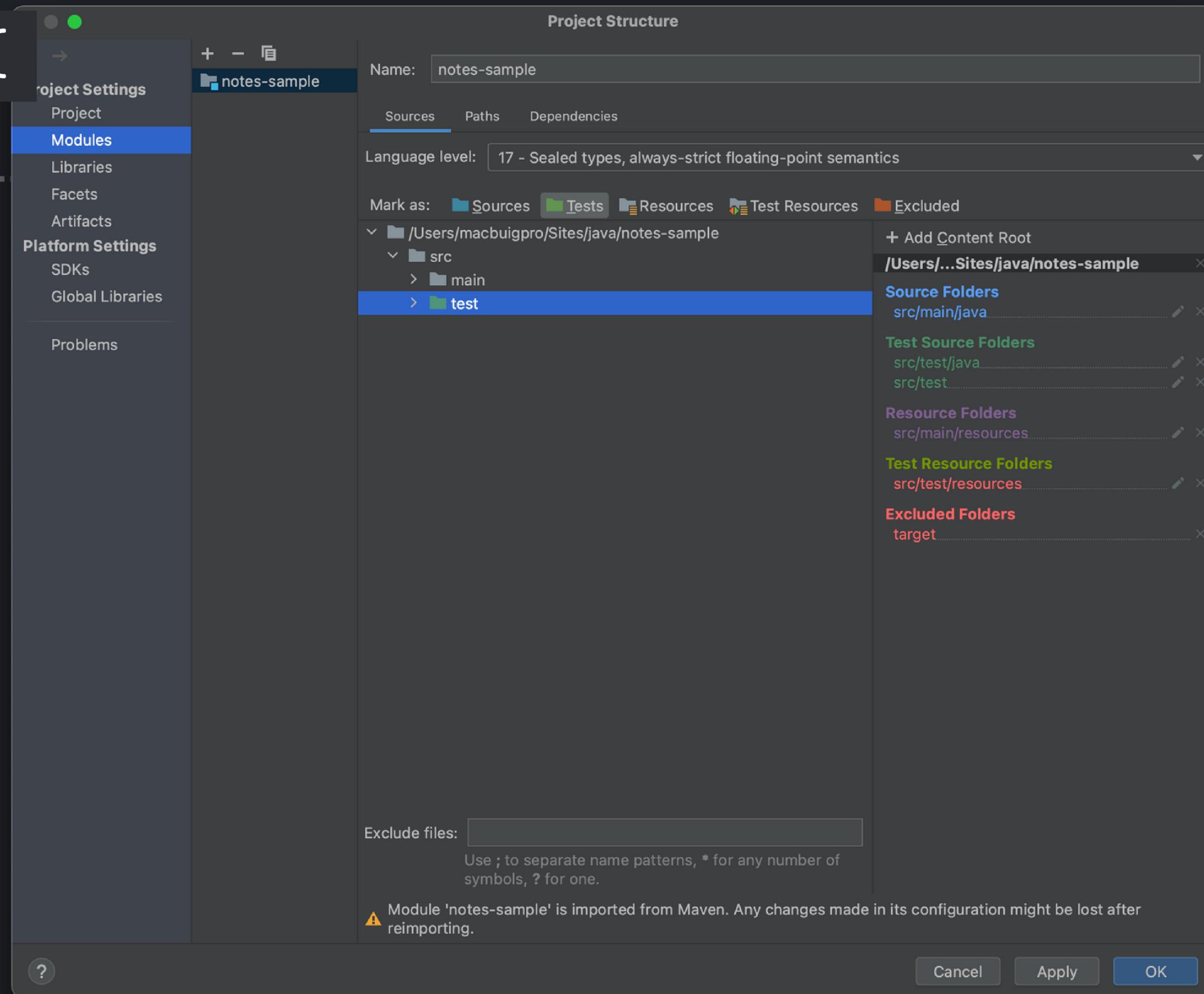
IntelliJ IDEA {

PASO 3

Añadimos JUnit
al proyecto que
hemos creado.

Varios pasos a
seguir.

Apply -> OK



}

Ejemplos {

Muestra fragmento de código de la Api de proveedores.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut a enim nec nisl ullamcorper eleifend.

Muestra fragmento de código de la exportación de datos.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut a enim nec nisl ullamcorper eleifend.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut a enim neDonec et sapien sit amet nisl pretium efficitur. Morbi faucibus felis mauris, sit amet finibus ipsum finibus at. Praesent maximus tincidunt fermentum. Fusce eget justo a sem auctor dapibus.c nisl ullamcorper eleifend.

}

Conclusiones {

Con lo que hemos visto, ya tendríamos disponibles nuestros entornos de trabajo para poder realizar tests sobre el código que nosotros programamos.

Ahora toca empezar a ver ejemplos de tipos de test y empezar a familiarizarnos con las funciones que nos permitirán validar nuestro código.

```
1 package org.antonio;
2
3 import org.junit.Test;
4
5 import static org.junit.Assert.assertFalse;
6 import static org.junit.Assert.assertTrue;
7
8 public class MainTest {
9     @Test
10    public void shouldAnswerWithTrue() { assertTrue( condition: true ); }
11
12
13
14
15    @Test
16    public void shouldAnswerWithFalse() { assertFalse( condition: false ); }
17
18
19
20 }
```

}