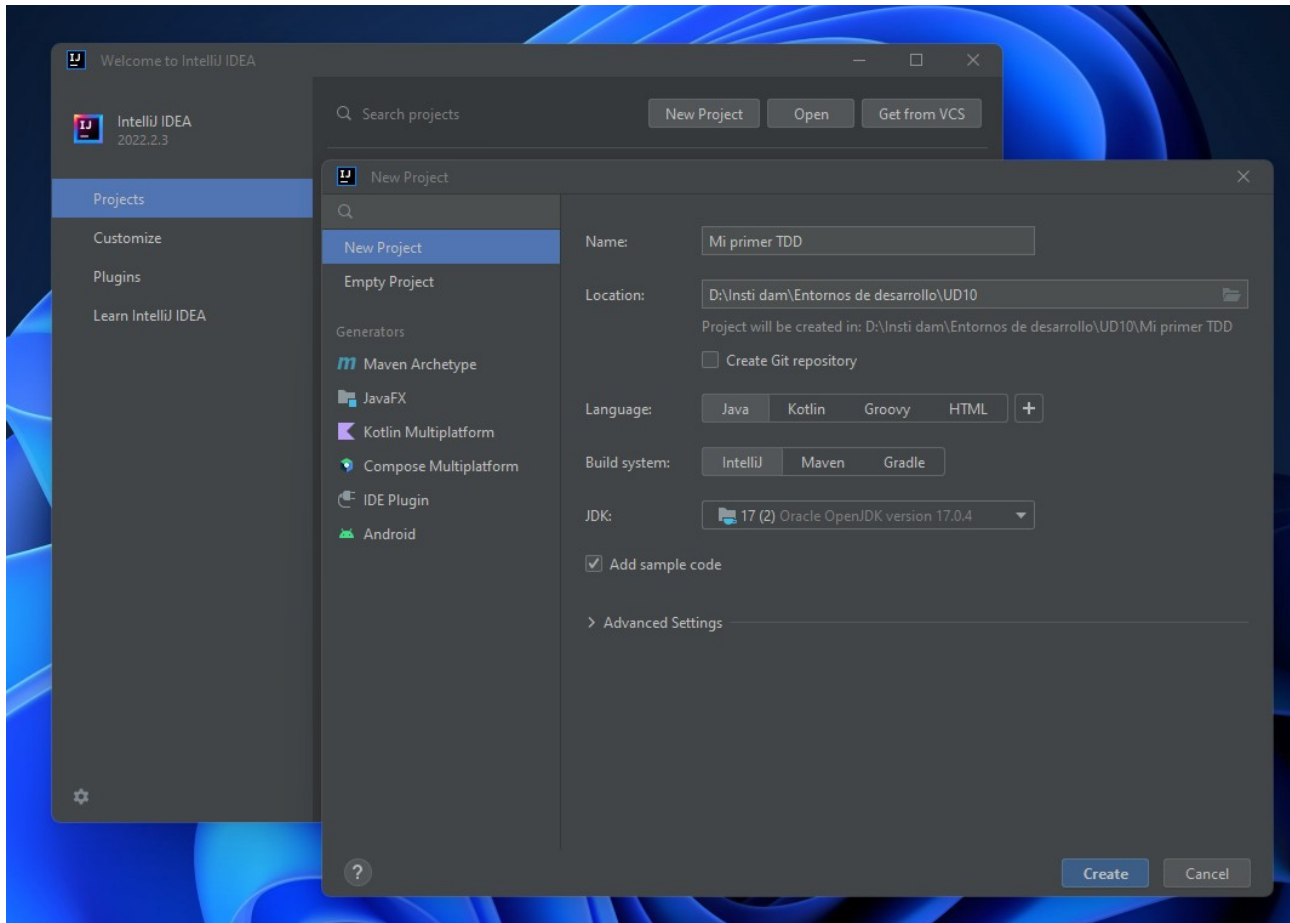


Memoria de Mi primer TDD

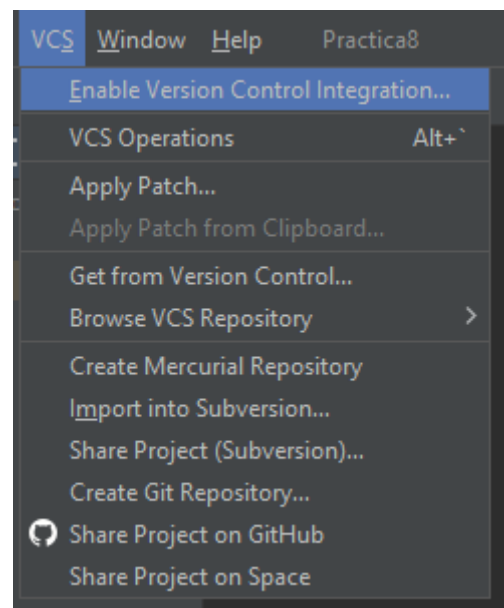
Alejandro Vicente Carpena

El primero paso es crear un proyecto en el entorno IntelliJ.

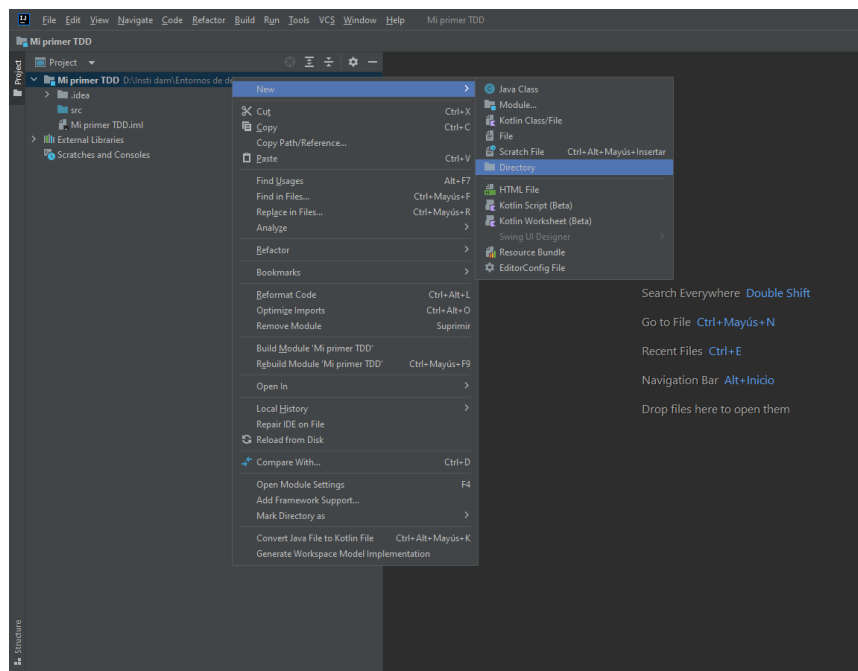
Al abrir el entorno de desarrollo, tendremos que darle a New Project, y se nos abra la segunda ventana, en este caso dejamos los parametros como estan, y ponemos el nombre que queramos, y la localizacion deseada



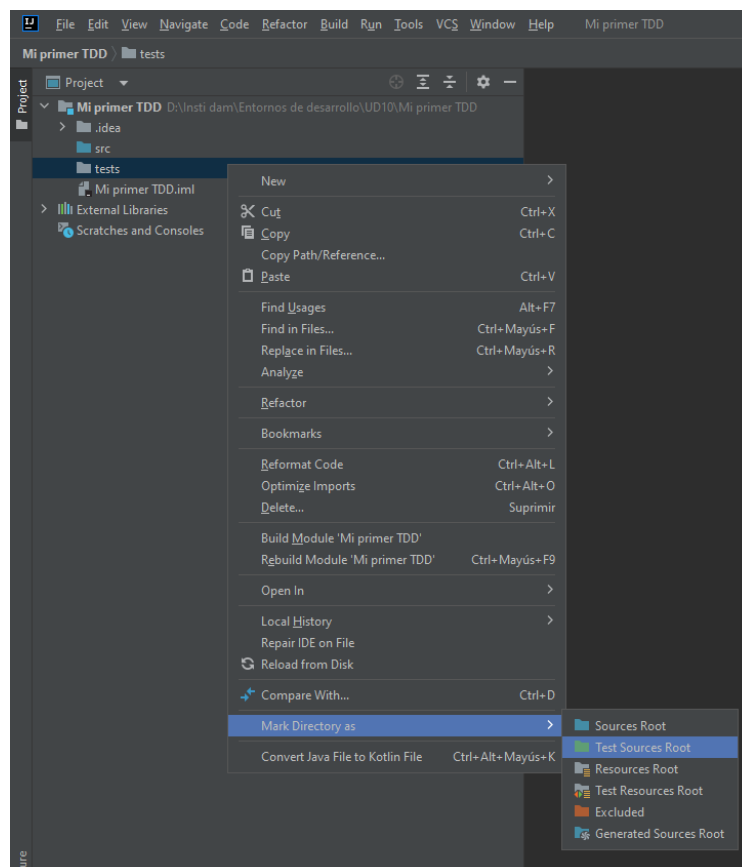
Una vez creado el proyecto, lo añadimos al repositorio git desde este apartado de las opciones de arriba de la pantalla



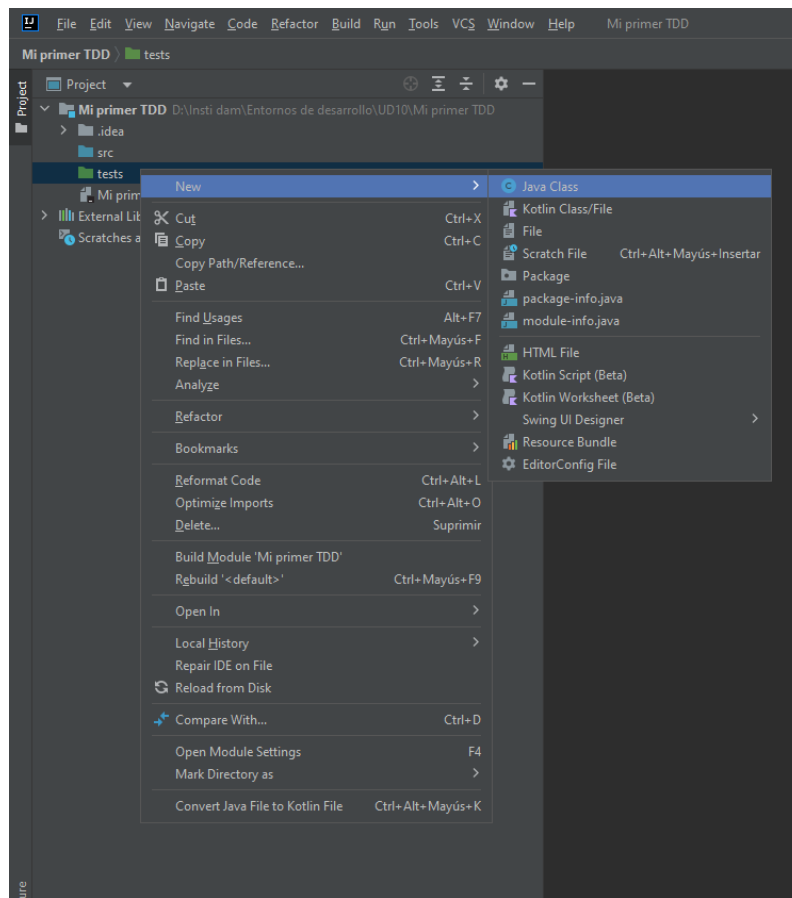
Dentro del proyecto, hacemos clic derecho sobre el proyecto, y en new, creamos un nuevo directorio llamado tests



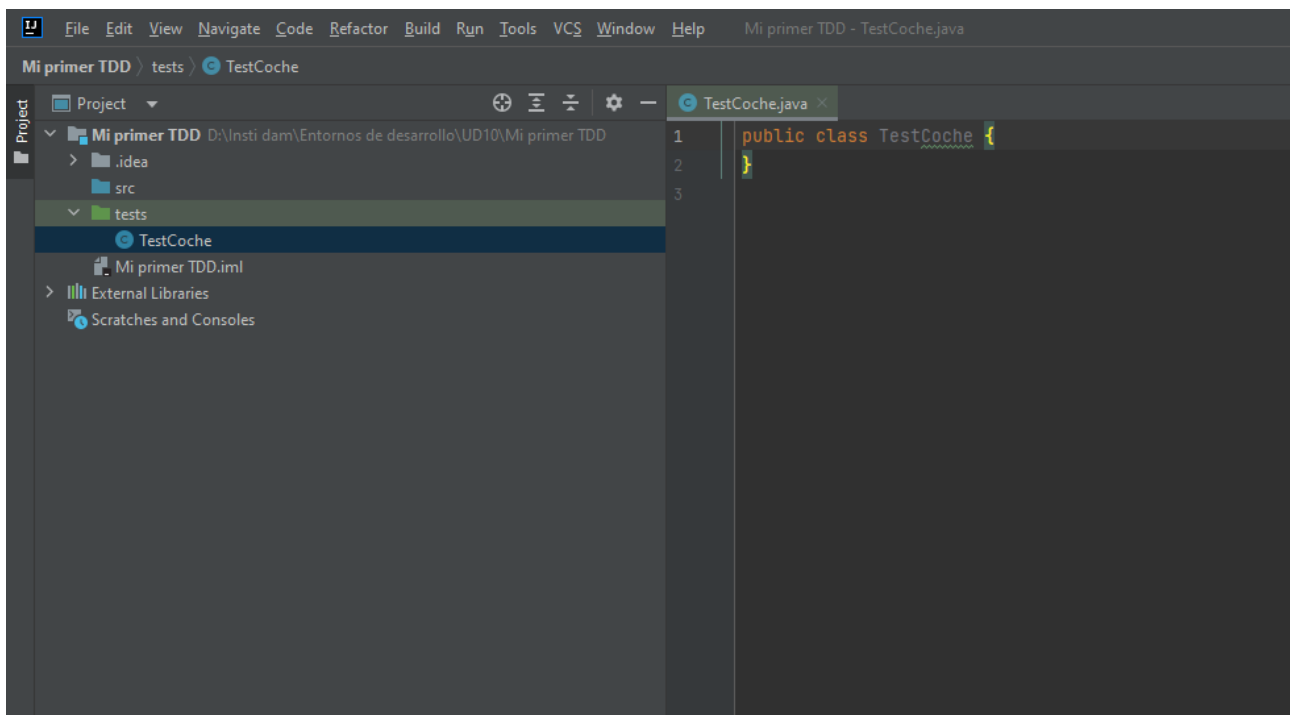
En este nuevo directorio, lo marcamos como la raíz de los tests, haciéndole clic derecho en el directorio, y dándole a esta opción



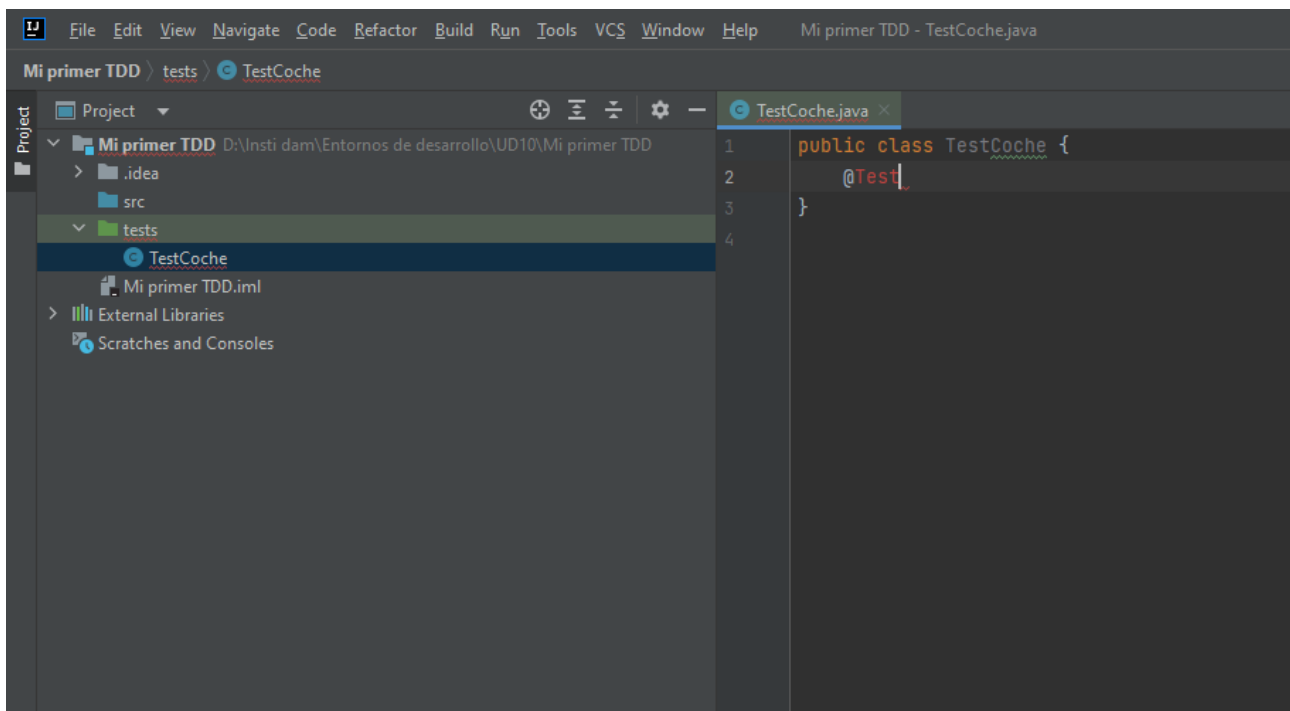
Ahora, dentro de tests, creamos una nueva clase, haciendo clic derecho, y dandole a new→ java class



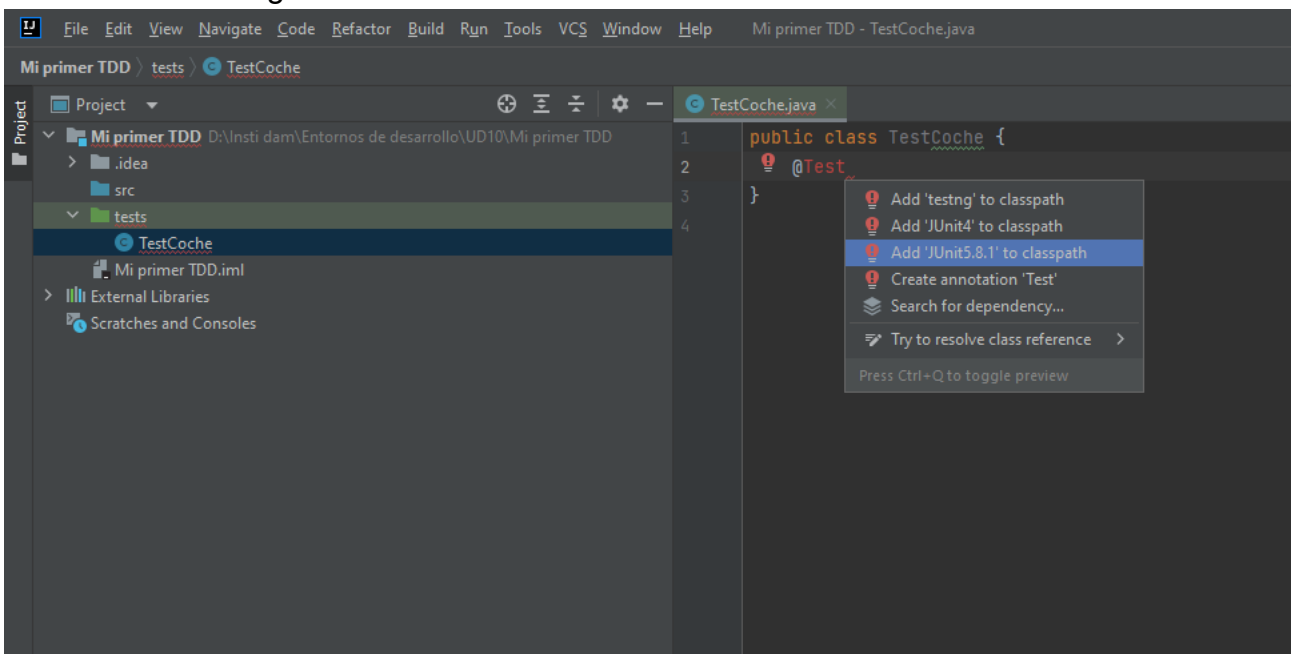
Asi se ve la clase una vez creada



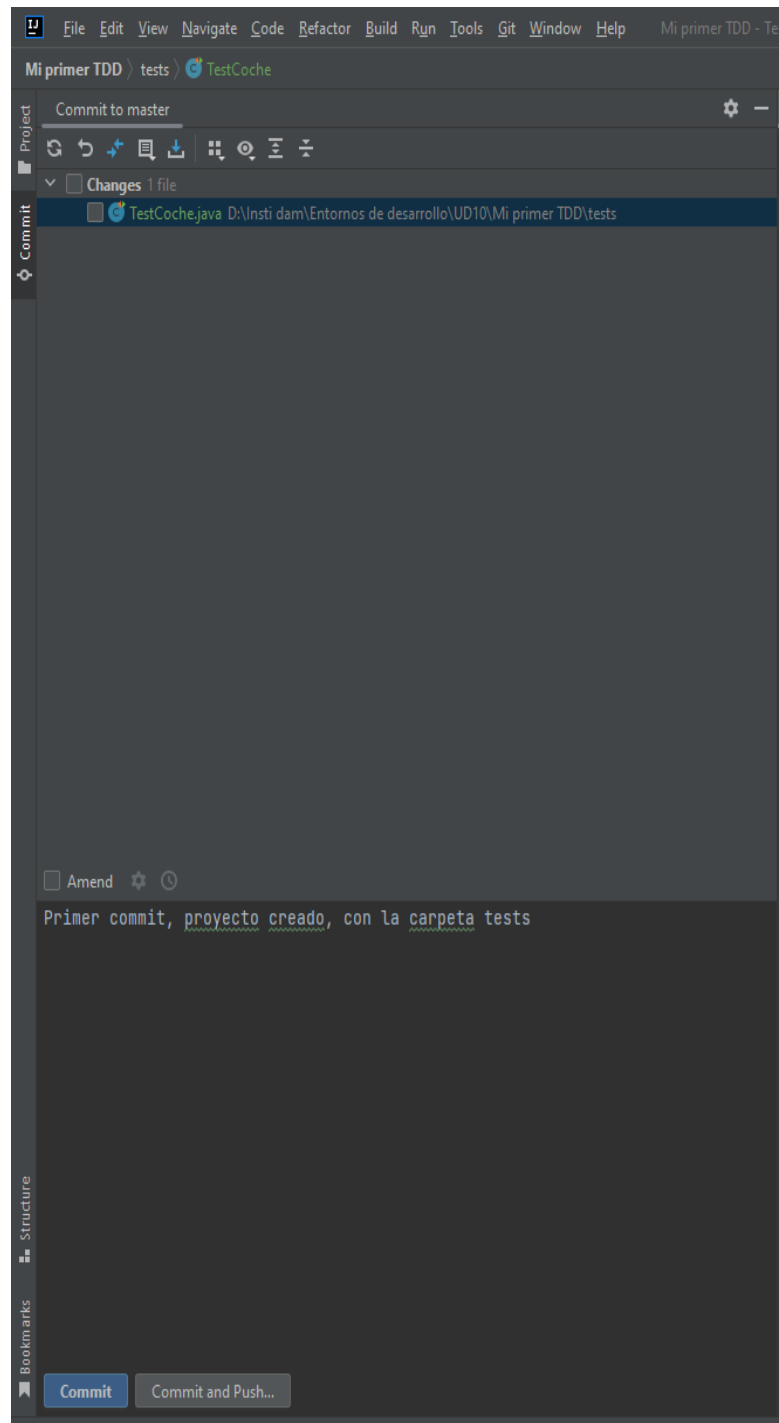
En el código escribimos `@Test`, y nos saldrá en rojo



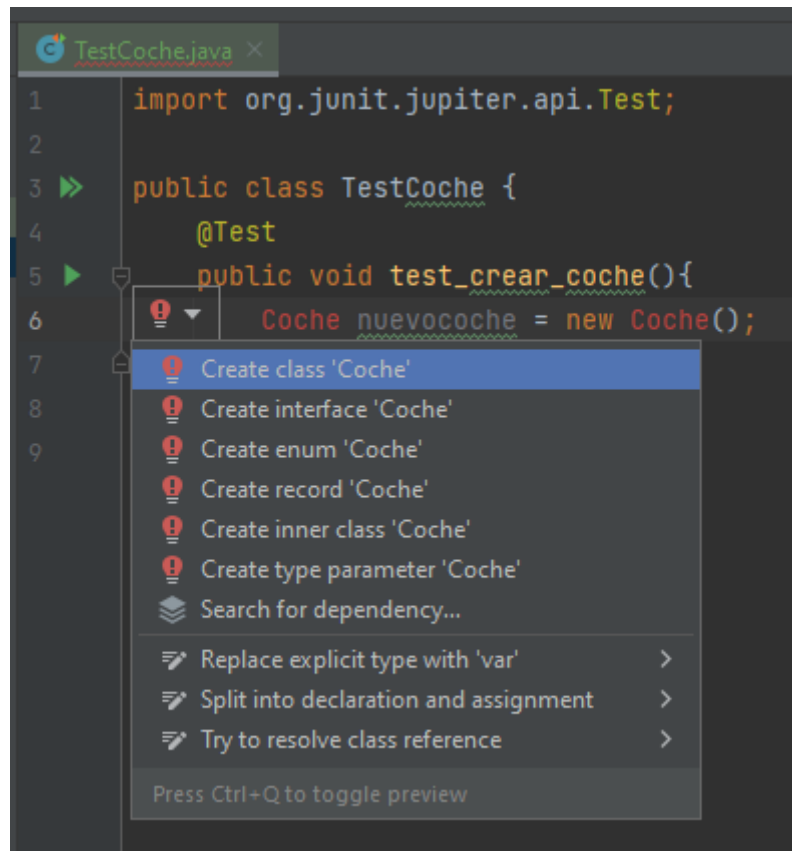
Pero dándole a las teclas `alt+intro` nos saldrá este desplegable para añadir Junit, y así nos saldrá bien el código



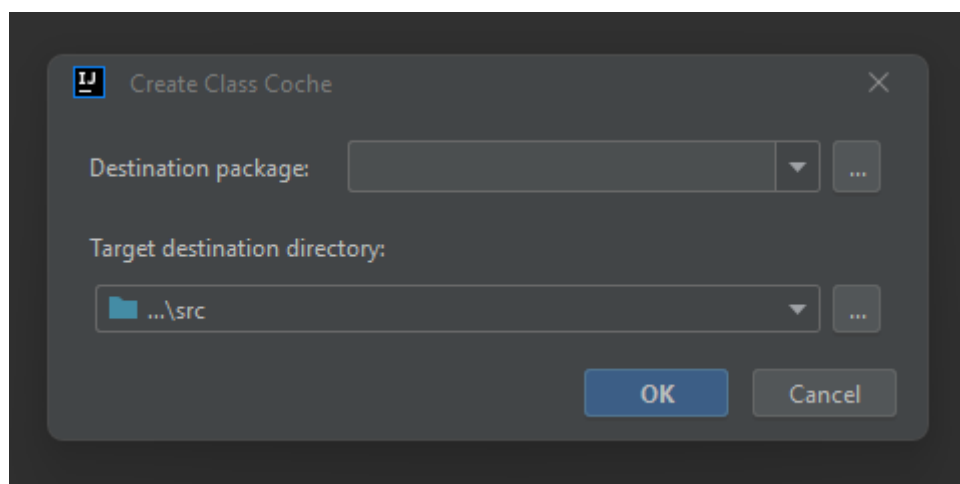
Una vez creado todo esto, he realizado el primer commit



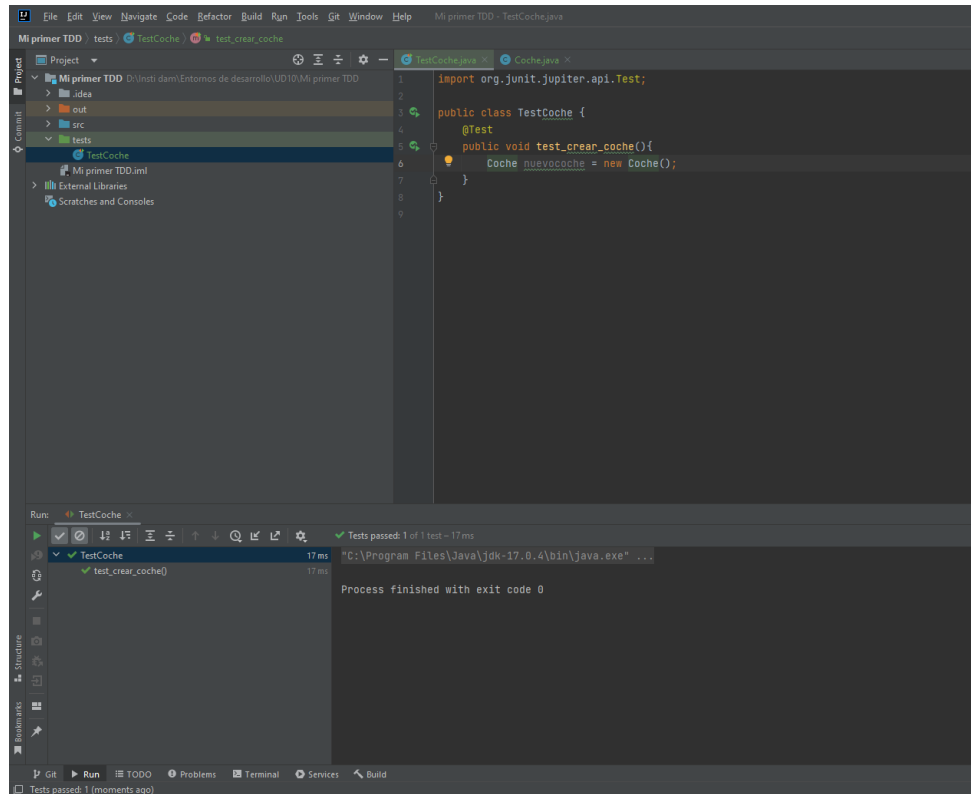
Ahora, dentro de la clase TestCoche, escribimos las siguientes sentencias. Como podemos ver la clase Coche nos sale en rojo porque no existe, por lo que usando las ayudas de IntelliJ, creamos la clase



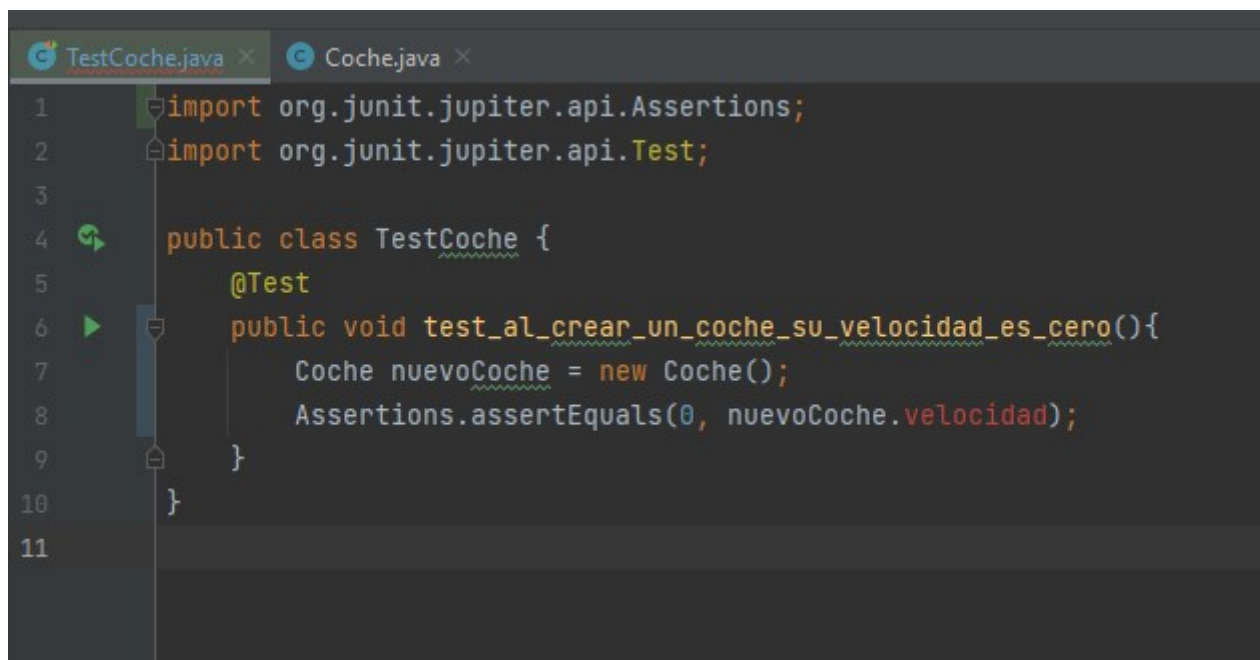
Seleccionamos la ruta donde queremos crearla, que en este caso seria en src



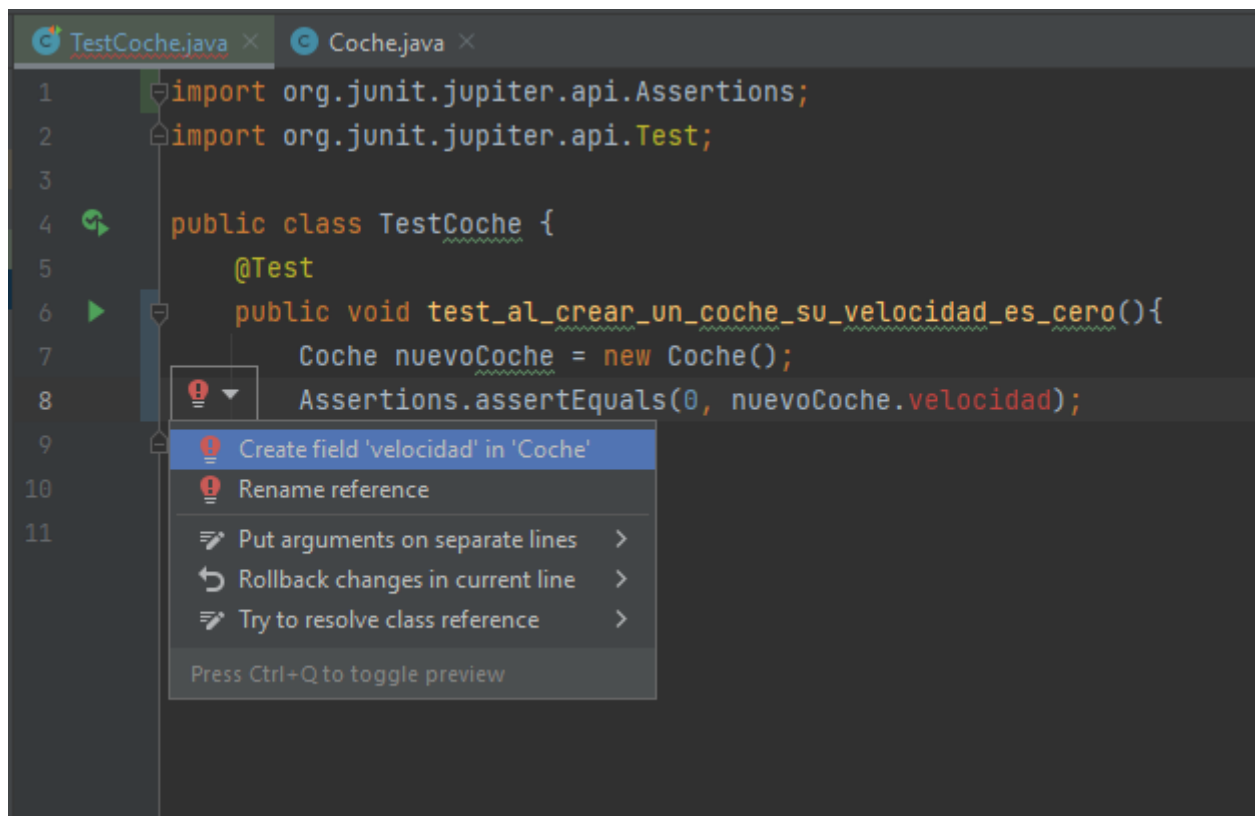
Realizamos el primer test, dándole a la flecha verde, y comprobamos que sale correcto



Añadimos un poco mas de complejidad al primer test, con un assertEquals. Esto comprueba que el parametro velocidad del objeto nuevoCoche sea 0.

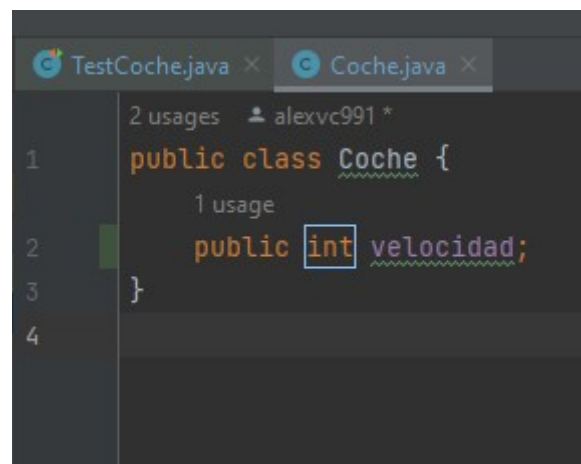


El parametro velocidad no esta creado, por lo que nos sale en rojo. Lo podemos crear dandole a create field 'velocidad' in 'Coche'



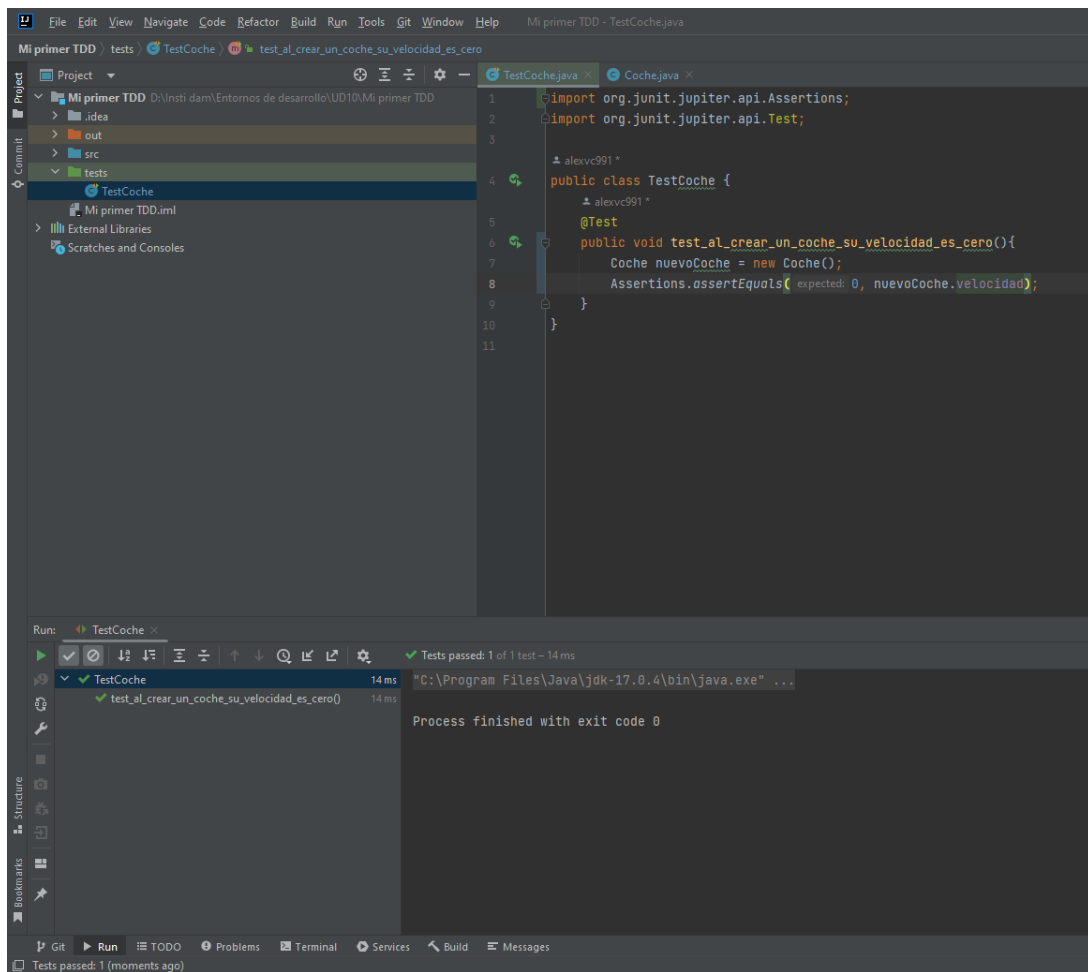
```
1 import org.junit.jupiter.api.Assertions;
2 import org.junit.jupiter.api.Test;
3
4 public class TestCoche {
5     @Test
6     public void test_al_crear_un_coche_su_velocidad_es_cero(){
7         Coche nuevoCoche = new Coche();
8         Assertions.assertEquals(0, nuevoCoche.velocidad);
9
10
11
```

Asi se quedaria el parametro una vez creado

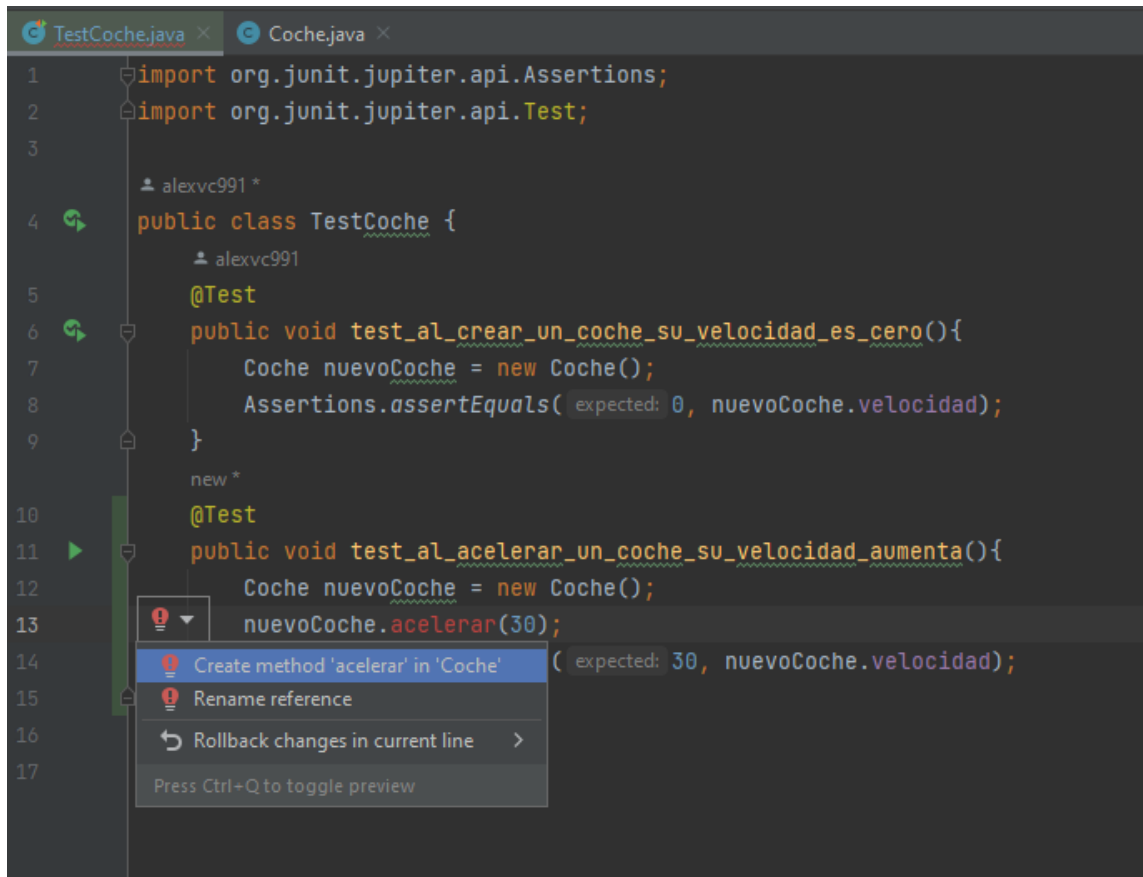


```
2 usages alexvc991 *
1 public class Coche {
2     1 usage
3     public int velocidad;
4 }
```


Corremos el test de nuevo, y comprobamos que lo pasa correctamente



Ahora creamos un segundo test, en el que el coche acelera. Por lo que el assertEquals comprueba que la velocidad final sea 30.
Como no tiene creado el metodo acelerar, debemos crearlo en la clase Coche. Lo creamos como en los casos anteriores



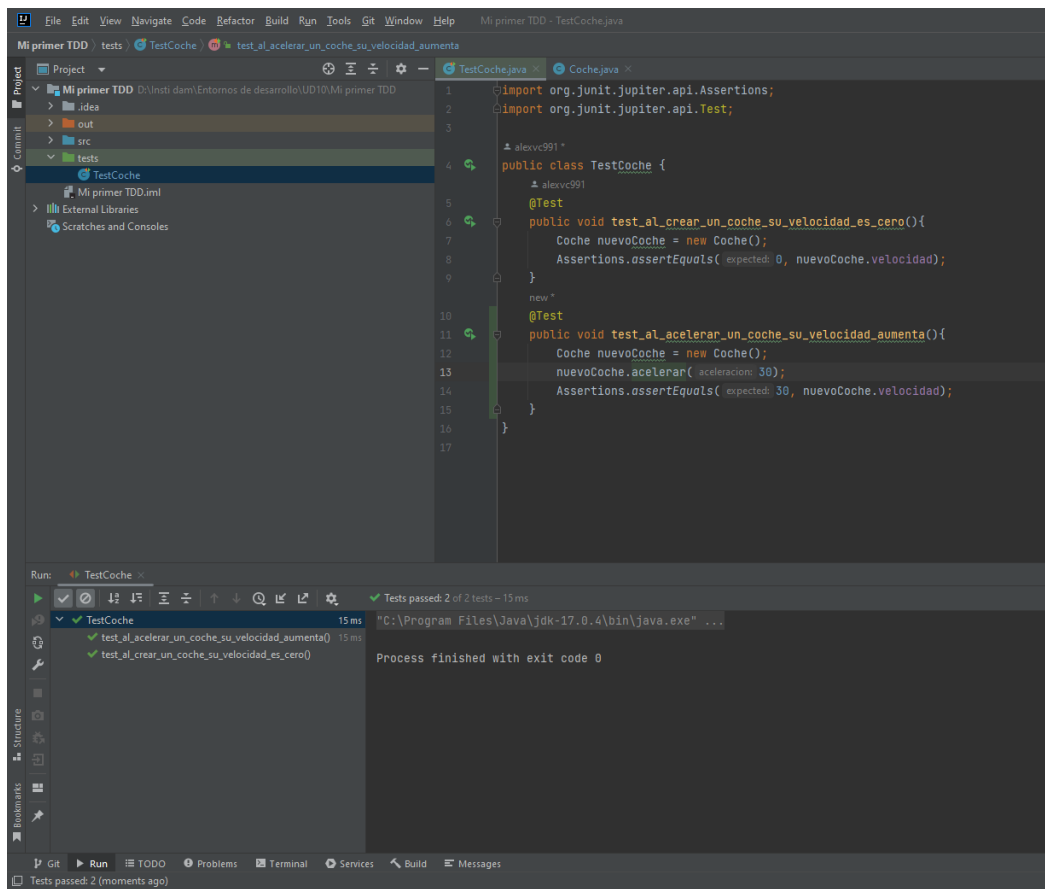
```
1 import org.junit.jupiter.api.Assertions;
2 import org.junit.jupiter.api.Test;
3
4 public class TestCoche {
5     @Test
6     public void test_al_crear_un_coche_su_velocidad_es_cero(){
7         Coche nuevoCoche = new Coche();
8         Assertions.assertEquals( expected: 0, nuevoCoche.velocidad);
9     }
10
11     @Test
12     public void test_al_acelerar_un_coche_su_velocidad_aumenta(){
13         Coche nuevoCoche = new Coche();
14         nuevoCoche.acelerar(30);
15         Assertions.assertEquals( expected: 30, nuevoCoche.velocidad);
16     }
17 }
```

Asi queda la clase Coche, con el metodo acelerar, el cual le suma la aceleracion a la velocidad



```
1 public class Coche {
2     public int velocidad;
3
4     public void acelerar(int aceleracion) {
5         velocidad += aceleracion;
6     }
7 }
8
```

Corremos el test, y comprobamos que lo pasa correctamente



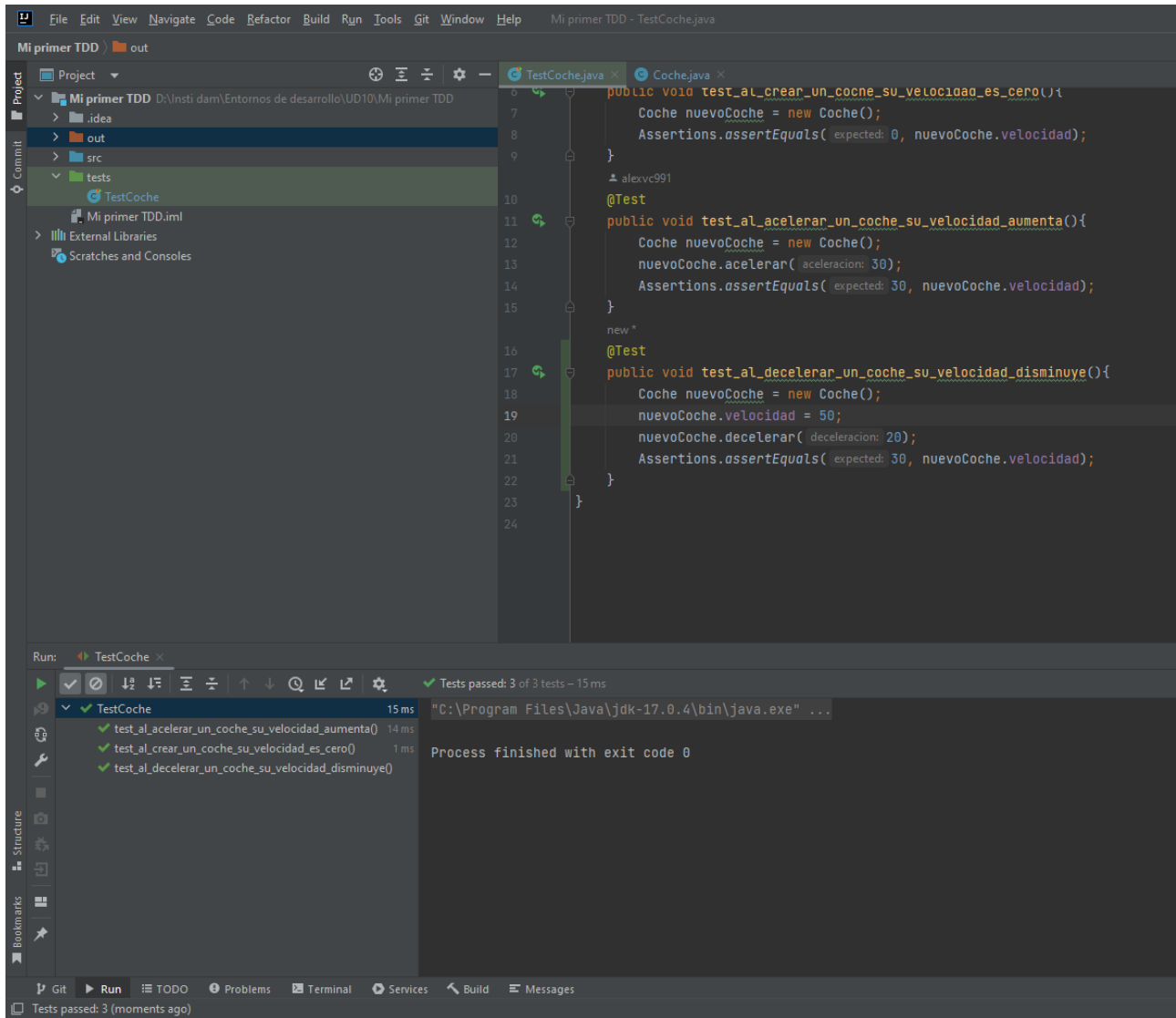
Añadimos el nuevo test decelerar, en el cual ponemos una velocidad inicial de 50, y deceleramos 20, por lo cual el assertEquals comprueba que la velocidad final sea 30. En este caso tampoco tenemos el método decelerar creado, por lo que tenemos que crearlo en la clase Coche.

```
TestCoche.java x Coche.java x
6 public void test_al_crear_un_cocne_su_velocidad_es_cero(){
7     Coche nuevoCoche = new Coche();
8     Assertions.assertEquals( expected: 0, nuevoCoche.velocidad);
9 }
alexvc991
10 @Test
11 public void test_al_acelerar_un_cocne_su_velocidad_aumenta(){
12     Coche nuevoCoche = new Coche();
13     nuevoCoche.acelerar( aceleracion: 30);
14     Assertions.assertEquals( expected: 30, nuevoCoche.velocidad);
15 }
new *
16 @Test
17 public void test_al_decelerar_un_cocne_su_velocidad_disminuye(){
18     Coche nuevoCoche = new Coche();
19     nuevoCoche.velocidad = 50;
20     nuevoCoche.decelerar( deceleracion: 20);
21     Assertions.assertEquals( expected: 30, nuevoCoche.velocidad);
22 }
23 }
24 }
```

Así sería el método, restandole a velocidad la deceleración

```
TestCoche.java x Coche.java x
6 usages alexvc991 *
1 public class Coche {
2     6 usages
3     public int velocidad;
4
5     1 usage alexvc991
6     public void acelerar(int aceleracion) {
7         velocidad += aceleracion;
8     }
9
10    1 usage new *
11    public void decelerar(int deceleracion) {
12        velocidad -= deceleracion;
13    }
14 }
```

Comprobamos que pasa el test



The screenshot shows an IDE window titled "Mi primer TDD - TestCoche.java". The editor displays the following Java code:

```
6 public void test_al_crear_un_cocne_su_velocidad_es_cero() {
7     Coche nuevoCoche = new Coche();
8     Assertions.assertEquals( expected: 0, nuevoCoche.velocidad);
9 }
10
11 @Test
12 public void test_al_acelerar_un_cocne_su_velocidad_aumenta(){
13     Coche nuevoCoche = new Coche();
14     nuevoCoche.acelerar( aceleracion: 30);
15     Assertions.assertEquals( expected: 30, nuevoCoche.velocidad);
16 }
17
18 @Test
19 public void test_al_decelerar_un_cocne_su_velocidad_disminuye(){
20     Coche nuevoCoche = new Coche();
21     nuevoCoche.velocidad = 50;
22     nuevoCoche.decelerar( deceleracion: 20);
23     Assertions.assertEquals( expected: 30, nuevoCoche.velocidad);
24 }
```

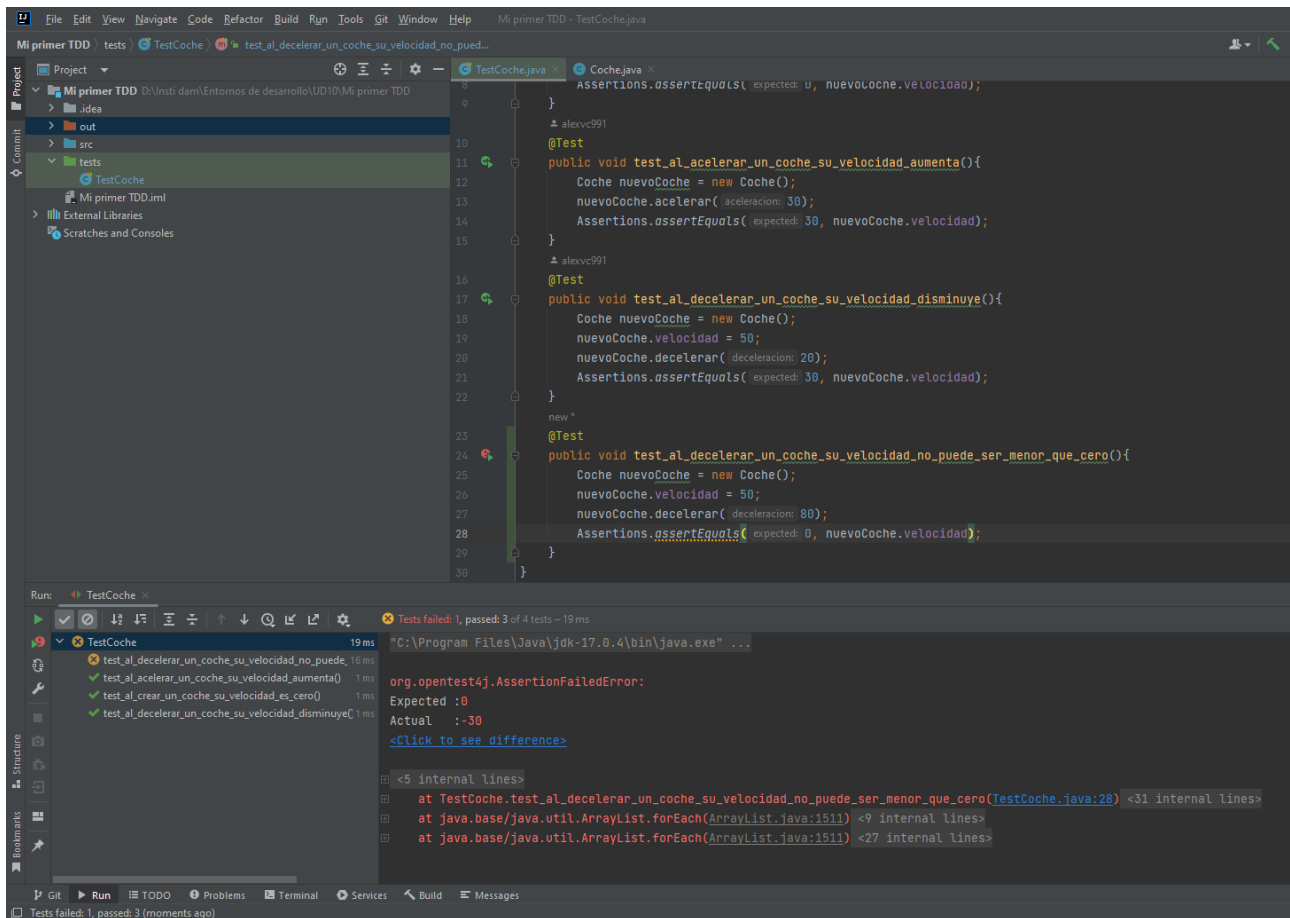
The "Run" tab at the bottom shows the test results for "TestCoche":

- TestCoche: 15 ms
- test_al_acelerar_un_cocne_su_velocidad_aumenta(): 14 ms
- test_al_crear_un_cocne_su_velocidad_es_cero(): 1 ms
- test_al_decelerar_un_cocne_su_velocidad_disminuye(): 1 ms

The output text indicates: "Process finished with exit code 0". The status bar at the bottom shows "Tests passed: 3 (moments ago)".

Y en el ultimo test, queremos que compruebe que la velocidad no sea menor que 0, por lo que añadimos una deceleracion mayor a la velocidad inicial y en el assertEquals ponemos que la velocidad esperada sea 0.

Una vez corremos el test, comprobamos que da error, por lo que tendremos que modificar el metodo decelerar en la clase Coche



```
File Edit View Navigate Code Refactor Build Run Tools Git Window Help Mi primer TDD - TestCoche.java
Mi primer TDD tests TestCoche test_al_decelerar_un_coche_su_velocidad_no_pued...
Project
  Mi primer TDD
    src
    tests
      TestCoche
        Mi primer TDD.iml
        External Libraries
        Scratches and Consoles
TestCoche.java
  8
  9
  10
  11
  12
  13
  14
  15
  16
  17
  18
  19
  20
  21
  22
  23
  24
  25
  26
  27
  28
  29
  30
  Assertions.assertEquals( expected: 0, nuevoCoche.velocidad);
  }
  }
  @Test
  public void test_al_decelerar_un_coche_su_velocidad_disminuye(){
    Coche nuevoCoche = new Coche();
    nuevoCoche.velocidad = 50;
    nuevoCoche.decelerar( deceleracion: 20);
    Assertions.assertEquals( expected: 30, nuevoCoche.velocidad);
  }
  @Test
  public void test_al_decelerar_un_coche_su_velocidad_no_puede_ser_menor_que_cero(){
    Coche nuevoCoche = new Coche();
    nuevoCoche.velocidad = 50;
    nuevoCoche.decelerar( deceleracion: 80);
    Assertions.assertEquals( expected: 0, nuevoCoche.velocidad);
  }
}

Run: TestCoche
Tests failed: 1, passed: 3 of 4 tests - 19 ms
C:\Program Files\Java\jdk-17.0.4\bin\java.exe ...
org.opentest4j.AssertionFailedError:
Expected :0
Actual   :-30
<Click to see difference>
at TestCoche.test_al_decelerar_un_coche_su_velocidad_no_puede_ser_menor_que_cero(TestCoche.java:28) <31 internal lines>
at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) <9 internal lines>
at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) <27 internal lines>
```

Para que la velocidad no pueda ser negativa, añadimos un if en el metodo, que haga que si la velocidad es menor a 0, velocidad sea 0.

Corremos el test de nuevo y lo pasa correctamente

