

Comenzamos creando un nuevo proyecto de Java en Eclipse y, dentro de la carpeta src del proyecto, crearemos un nuevo paquete. Dentro de este paquete, crearemos un directorio de tests de JUnit con click derecho > New > JUnit Test Case.

here)' with a 'Generate comments' checkbox. At the bottom, there is a 'Class under test:' field with a 'Browse...' button. The footer contains a help icon, '< Back', 'Next >', 'Finish', and 'Cancel' buttons."/>

New JUnit Test Case

The use of the default package is discouraged.

☒ New JUnit 3 test ☐ New JUnit 4 test ☐ New JUnit Jupiter test

Source folder: MiPrimerTDD/src Browse...

Package: (default) Browse...

Name: tests

Superclass: java.lang.Object Browse...

Which method stubs would you like to create?

☐ @BeforeAll setUpBeforeClass() ☐ @AfterAll tearDownAfterClass()
☐ @BeforeEach setUp() ☐ @AfterEach tearDown()
☐ constructor

Do you want to add comments? (Configure templates and default value [here](#))

☐ Generate comments

Class under test: Browse...

? < Back Next > Finish Cancel

Aparecerá una ventana en la que indicaremos la versión de JUnit (Jupiter en este caso) y el nombre del directorio.

New JUnit Test Case

JUnit 5 is not on the build path. Do you want to add it?

☐ Not now
☐ Open the build path property page
☒ Perform the following action:

Add JUnit 5 library to the build path

OK Cancel

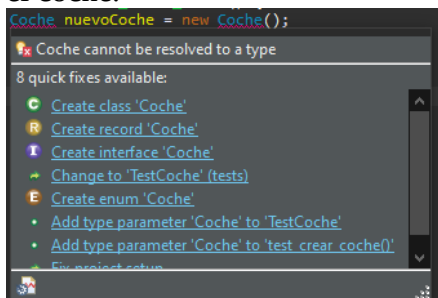
Indicaremos que queremos añadir la librería de JUnit 5 en la siguiente ventana.

```

1 package tests;
2
3 import static org.junit.jupiter.api.Assertions.*;
4
5
6
7 class TestCoche {
8
9     @Test
10    public void test_crear_coche() {
11        Coche nuevoCoche = new Coche();
12    }
13
14 }
15

```

Siguiendo los mismos pasos que en la versión de TDD de IntelliJ, creamos un primer test para crear el coche.



New

Java Class

Create a new Java class.

Source folder: MiPrimerTDD/src Browse...

Package: tests Browse...

☐ Enclosing type: tests.TestCoche Browse...

Name: Coche

Modifiers: ☐ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static
☐ none ☐ sealed ☐ non-sealed ☐ final

Superclass: java.lang.Object Browse...

Interfaces: Add...
Remove

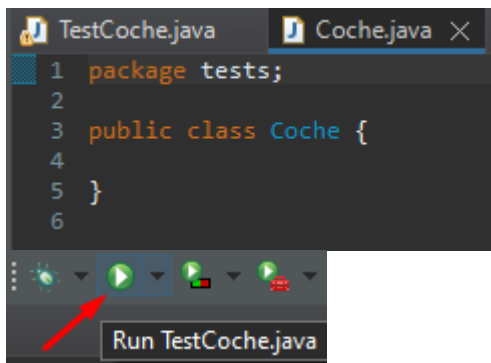
Which method stubs would you like to create?

☐ public static void main(String[] args)
☐ Constructors from superclass
☒ Inherited abstract methods

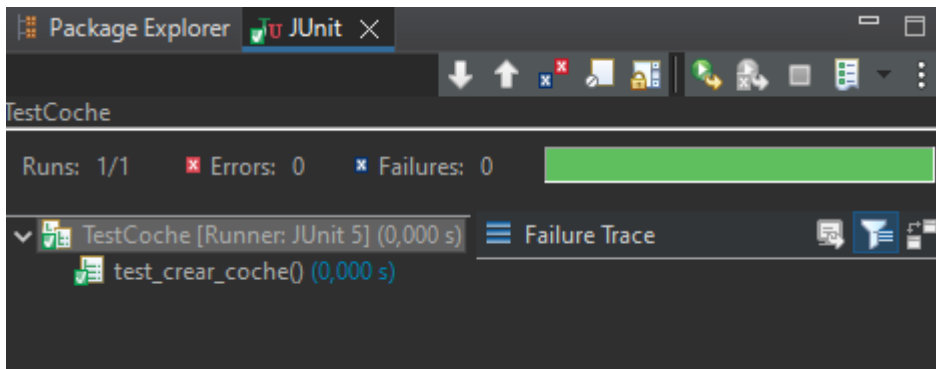
Do you want to add comments? (Configure templates and default value [here](#))
☐ Generate comments

? Finish Cancel

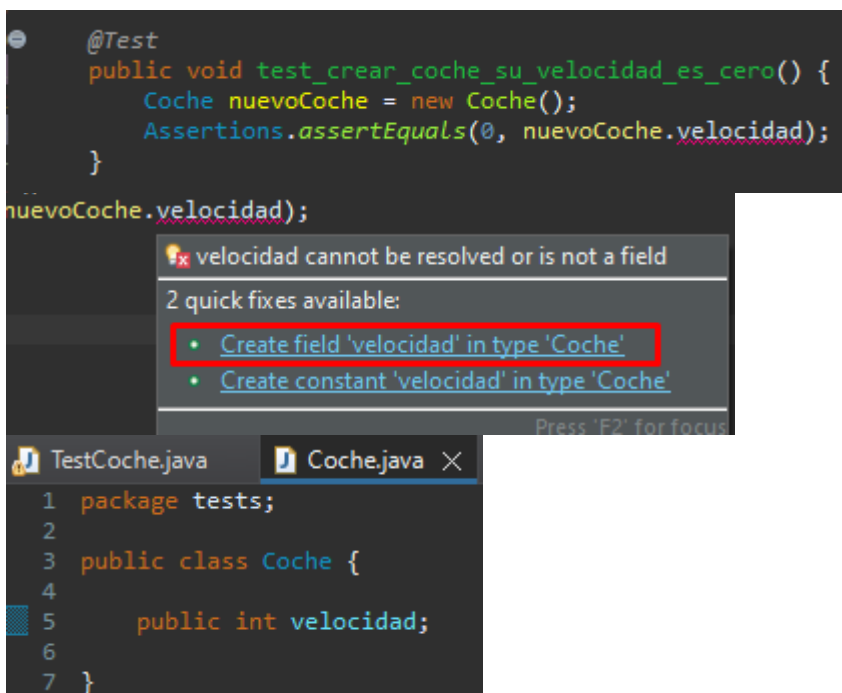
Aprovechamos y creamos la clase Coche al pasar el cursor por encima de la palabra subrayada en rojo y haciendo click en Create class 'Coche'. Aparecerá un recuadro de creación de nueva clase en el que indicaremos el destino de la clase.



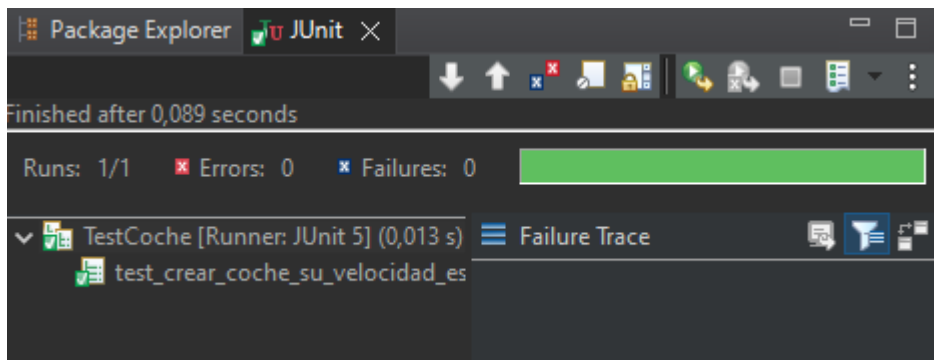
Una vez creada, podemos iniciar el primer test iniciando el programa.



Pasa el test sin problemas.



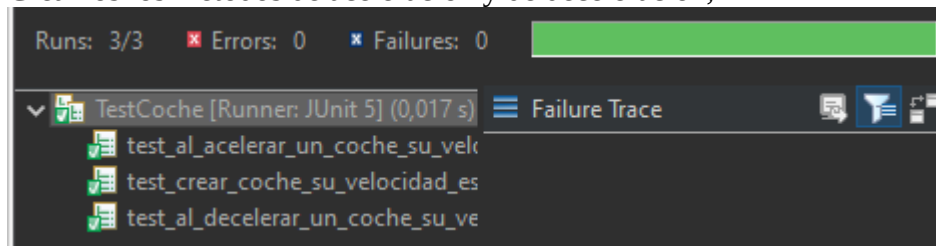
De la misma manera que en IntelliJ, creamos el método de la velocidad inicial, además de crear el atributo en la clase Coche.



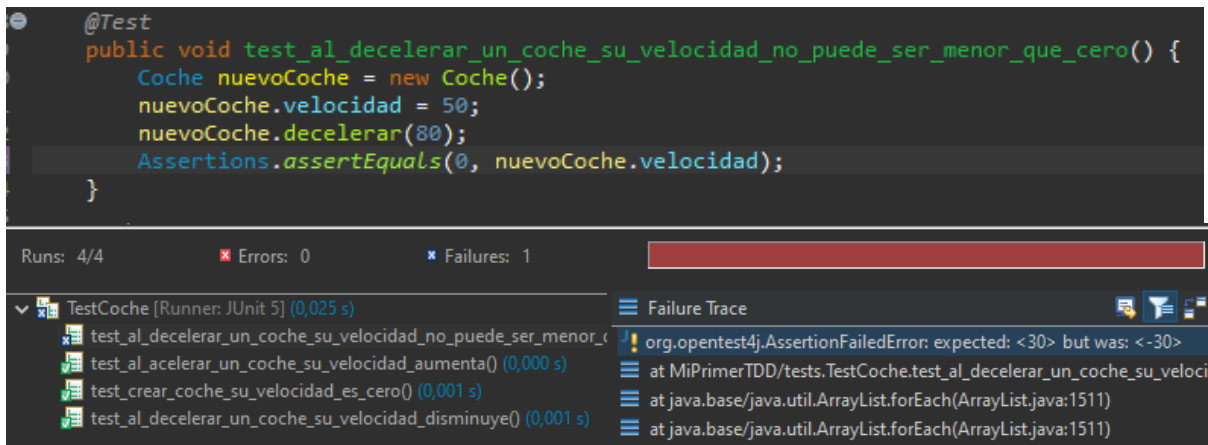
El test es correcto.

```
15 @Test
16 public void test_al_acelerar_un_coche_su_velocidad_aumenta() {
17     Coche nuevoCoche = new Coche();
18     nuevoCoche.acelerar(30);
19     Assertions.assertEquals(30, nuevoCoche.velocidad);
20 }
21 @Test
22 public void test_al_decelerar_un_coche_su_velocidad_disminuye() {
23     Coche nuevoCoche = new Coche();
24     nuevoCoche.velocidad = 50;
25     nuevoCoche.decelerar(20);
26     Assertions.assertEquals(30, nuevoCoche.velocidad);
27 }
28
29 }
30
7 @Test
8 public void acelerar(int aceleracion) {
9     velocidad += aceleracion;
10 }
11
12 @Test
13 public void decelerar(int deceleracion) {
14     velocidad -= deceleracion;
15 }
16 }
```

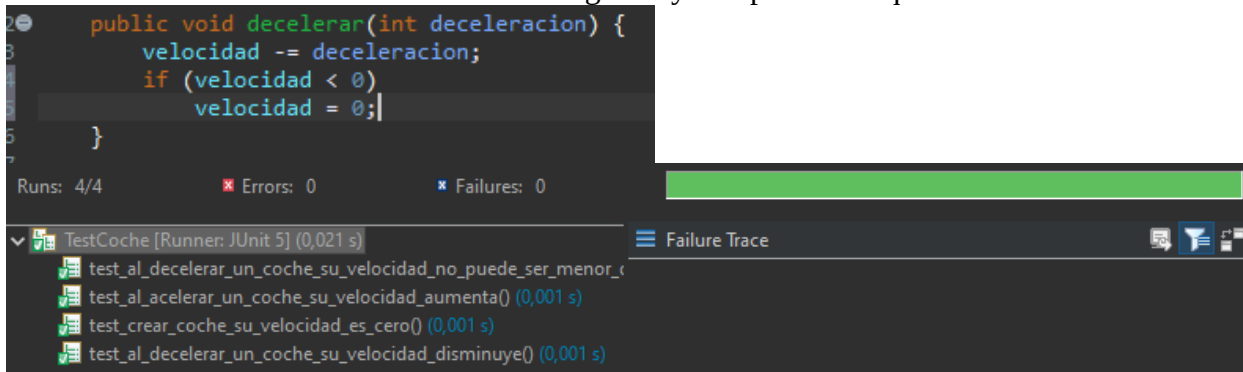
Creamos los métodos de aceleración y de deceleración,



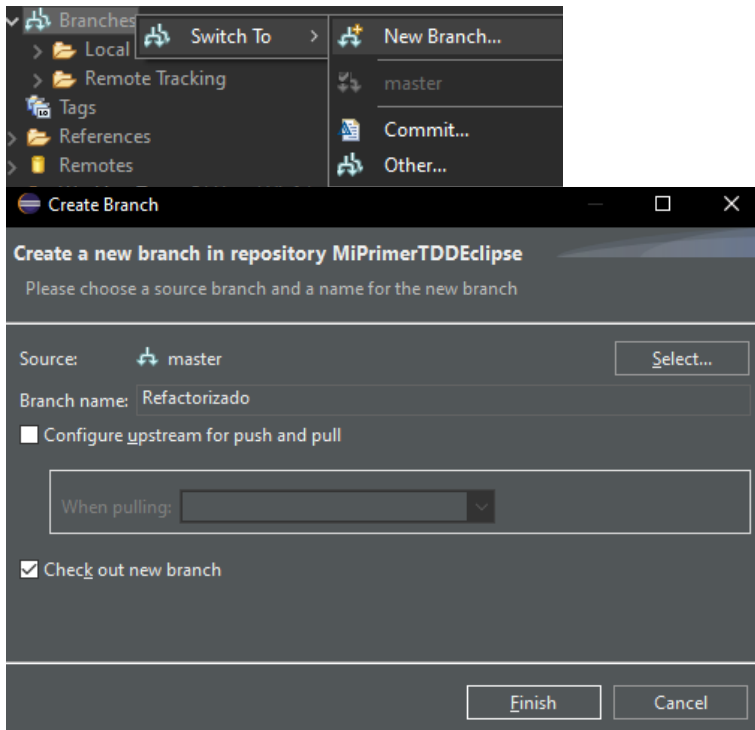
Comprobamos que el test sea correcto.



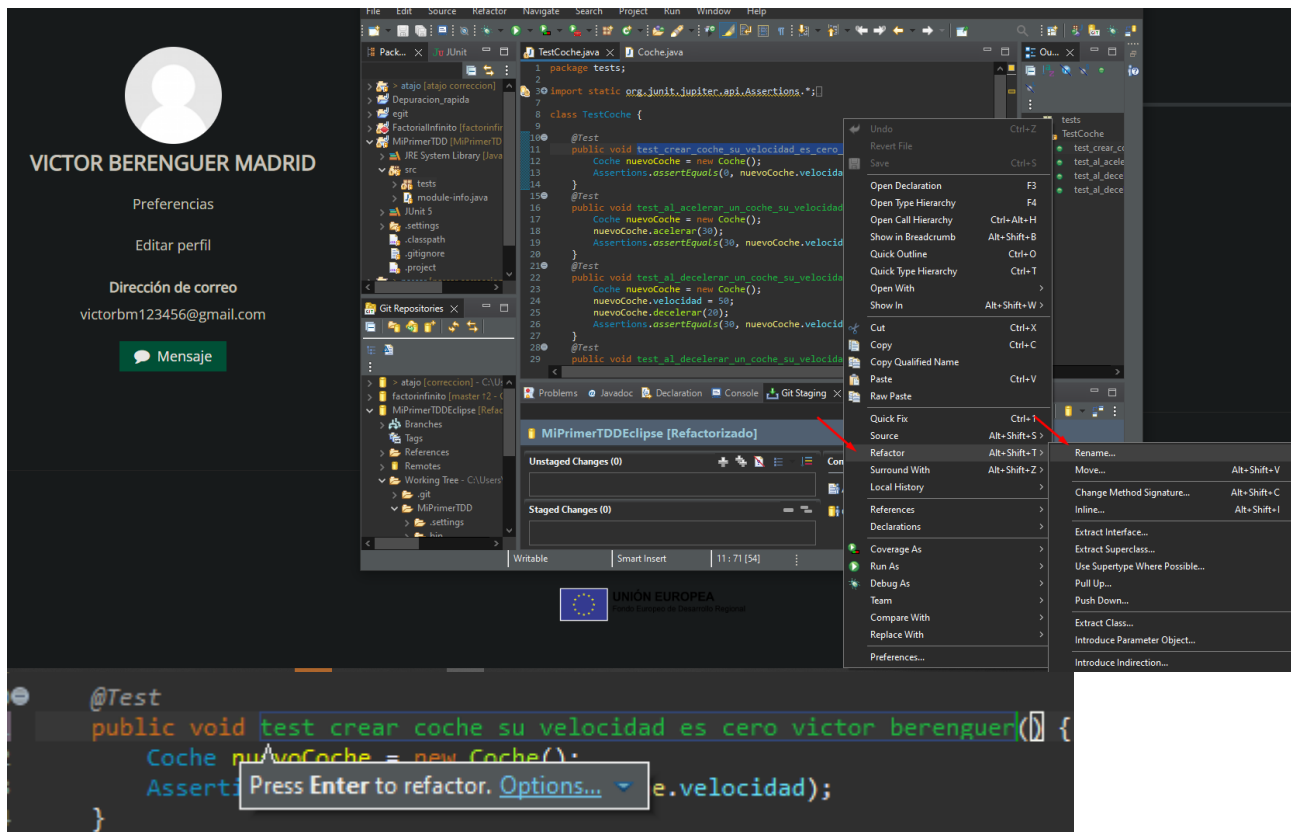
Forzamos el mismo error con la velocidad negativa y comprobamos que el test falle.



Por último, arreglamos el error con un condicional en el método decelerar y comprobamos que el test sea correcto.



Creamos la rama Refactorizado haciendo click derecho en el apartado de ramas de la interfaz de Git en Eclipse > Switch To > New Branch. Confirmamos en la siguiente ventana marcando la casilla de Check out new branch para cambiarnos a la rama recién creada.



Para refactorizar, hacemos click derecho sobre el método > Refactor > Rename.