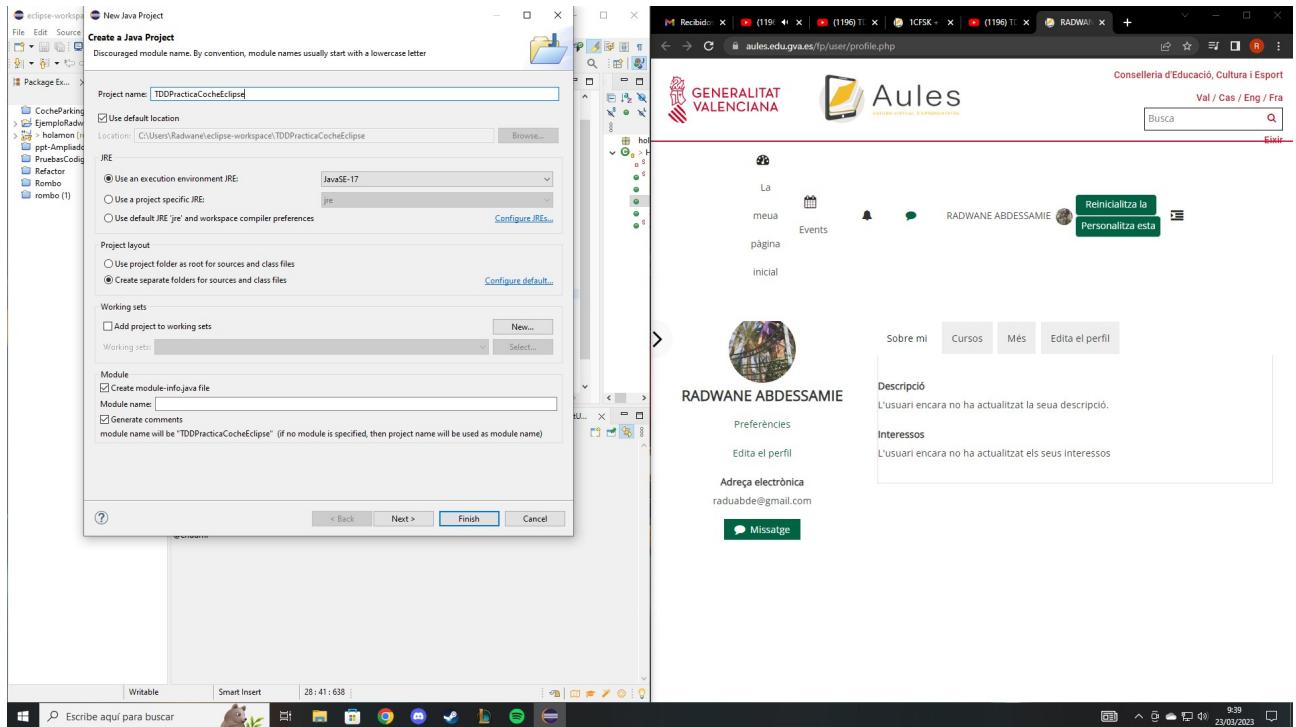
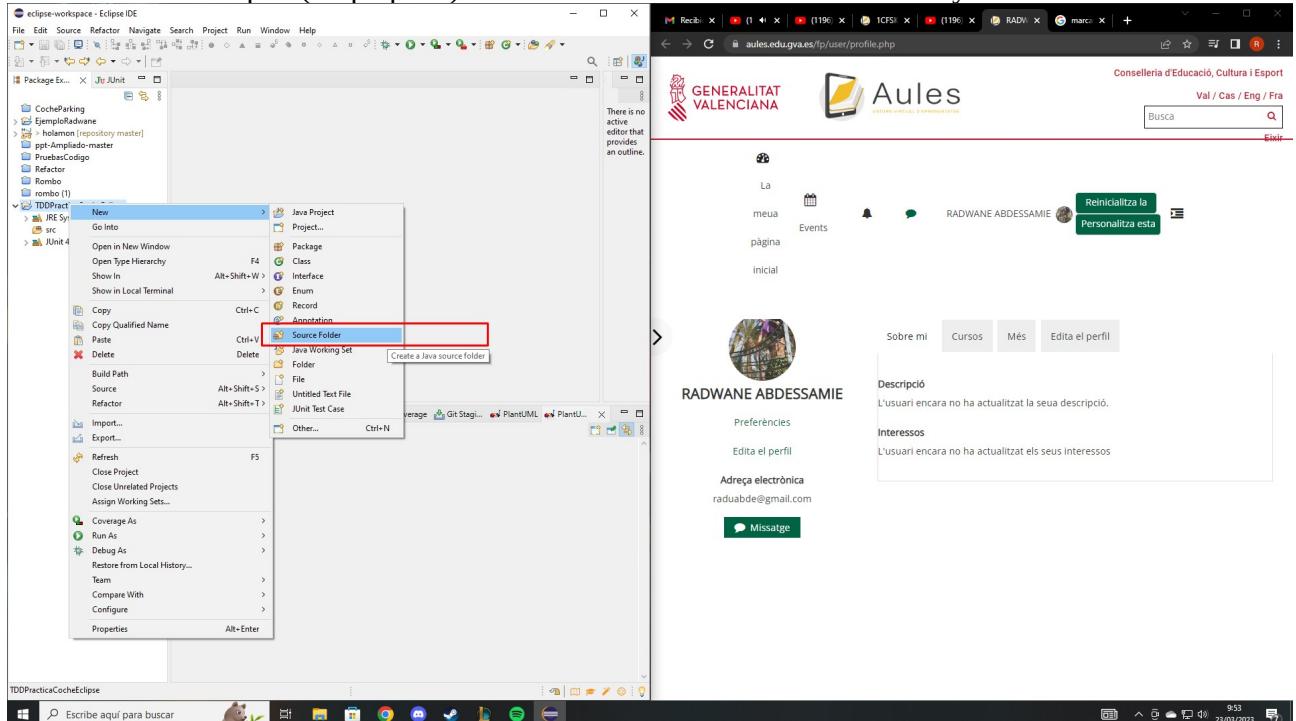


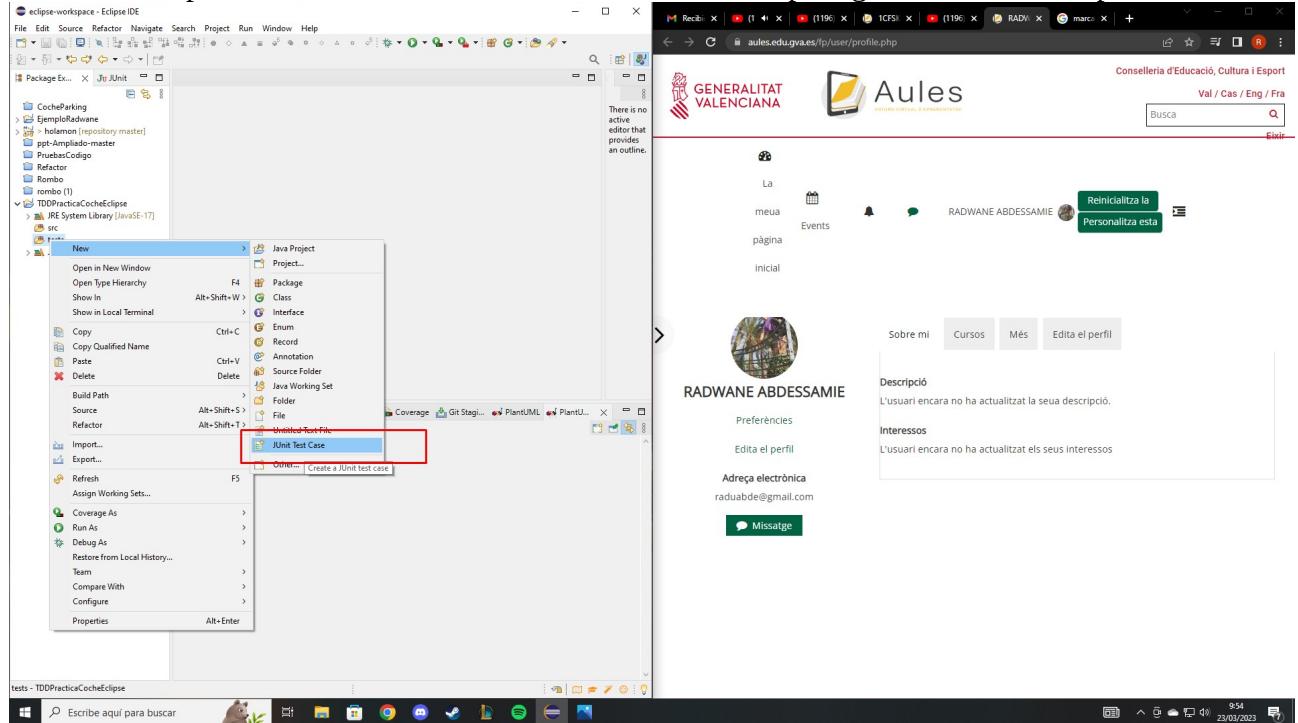
1.Creamos un proyecto en Eclipse con el nombre TDDPracticaCocheEclipse.



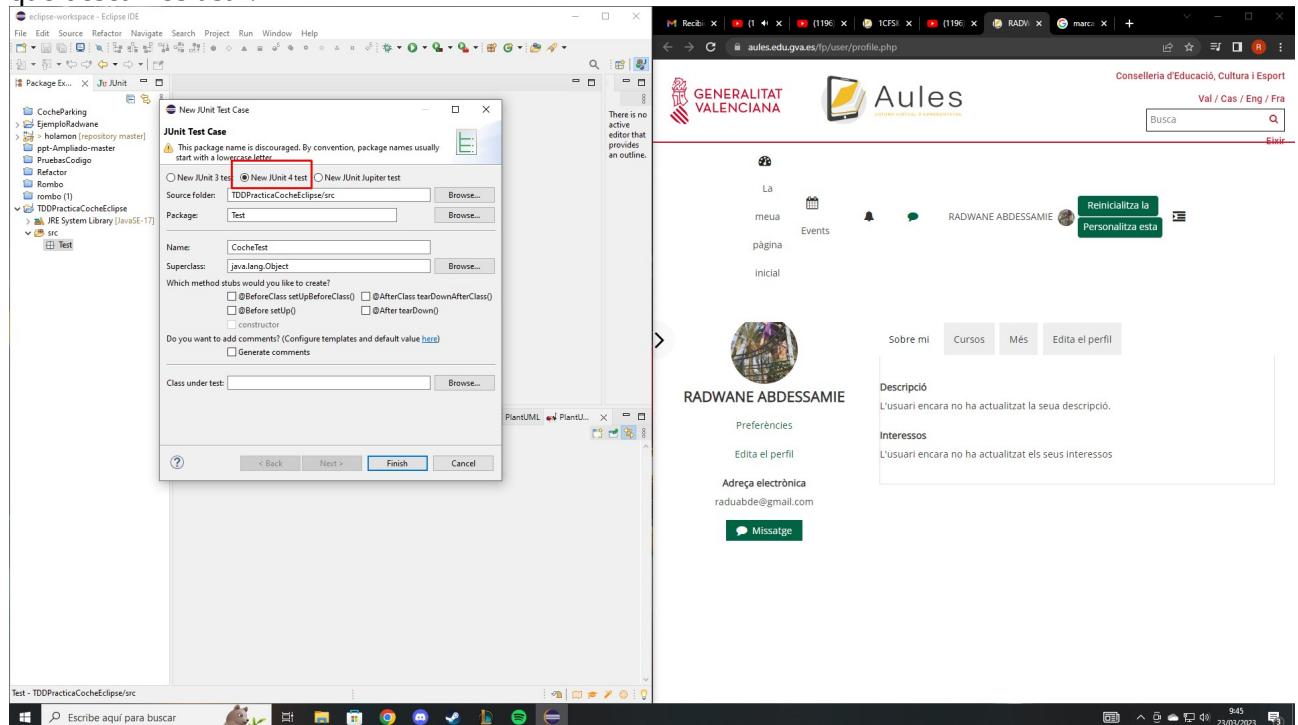
2.Creamos una carpeta(no paquete) con click derecho new Source Folder y lo llamamos tests



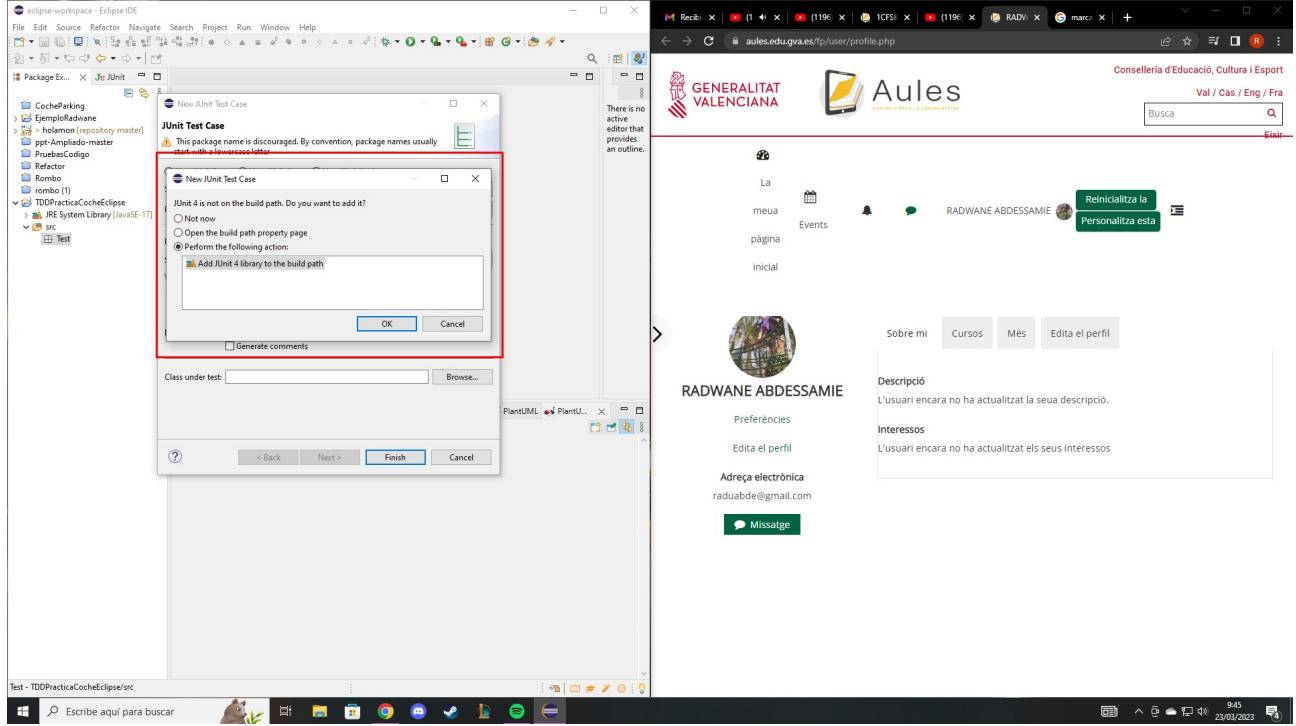
3.Sobre la carpeta tests, click derecho , new, Junit Test Case para generar la clase de pruebas.



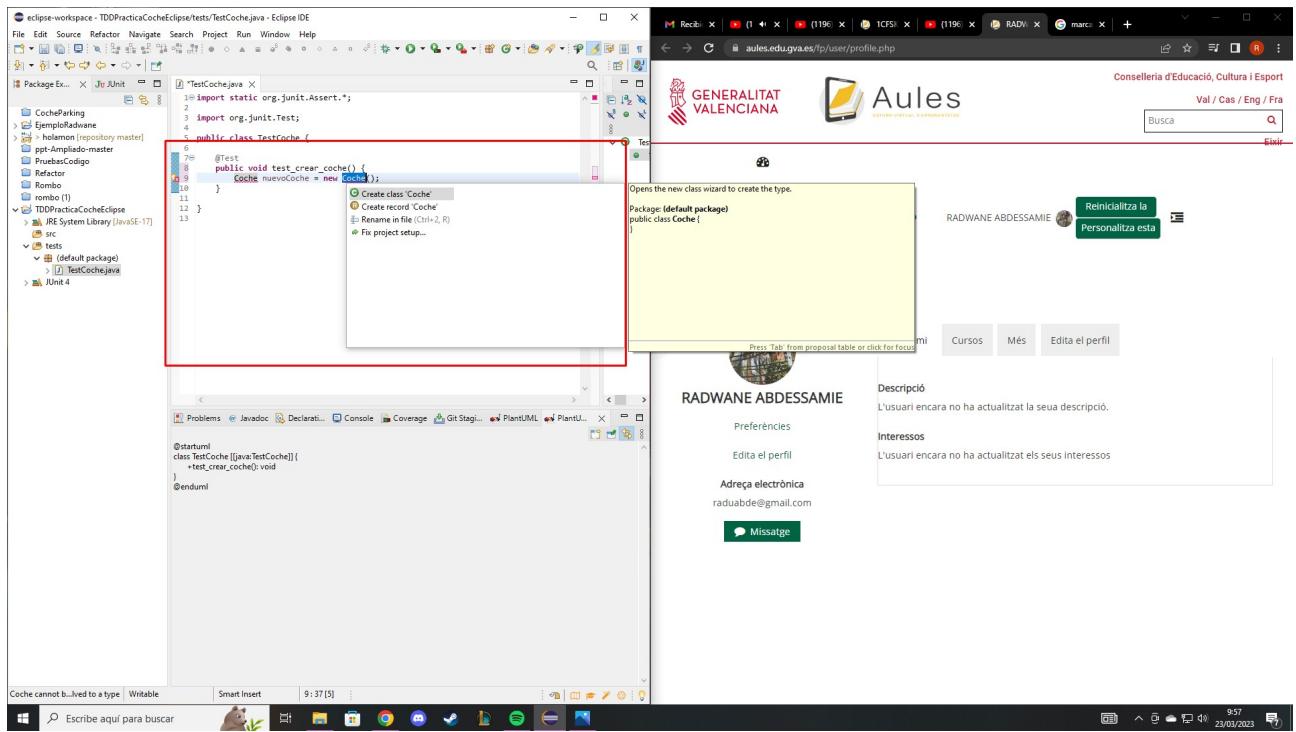
4.En la ventana de creación de la clase, seleccionamos el nombre de la clase y la versión de Junit que deseamos usar.



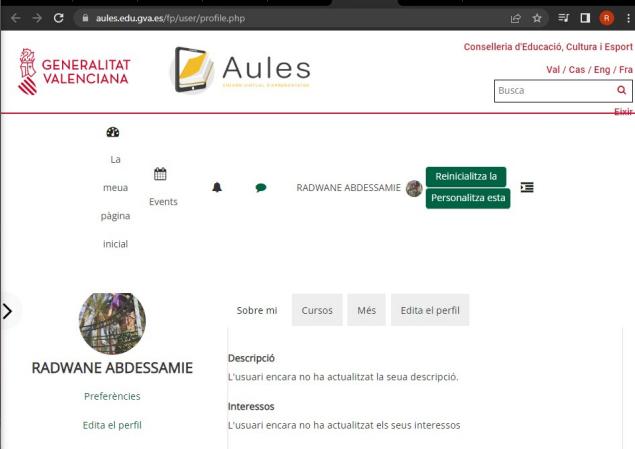
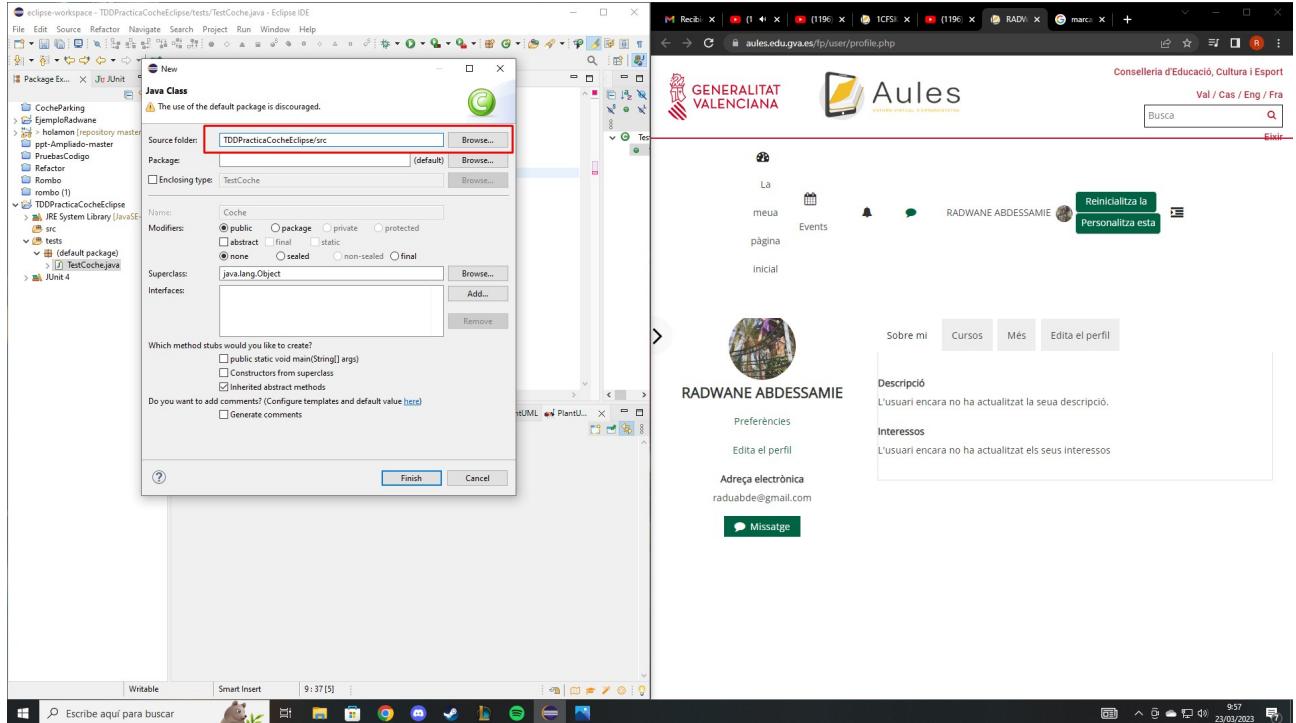
5.Nota: Si no tenemos la librería de Junit instalada, Eclipse nos mostrara dicho aviso y nos implementara la librería automáticamente si así lo deseamos.



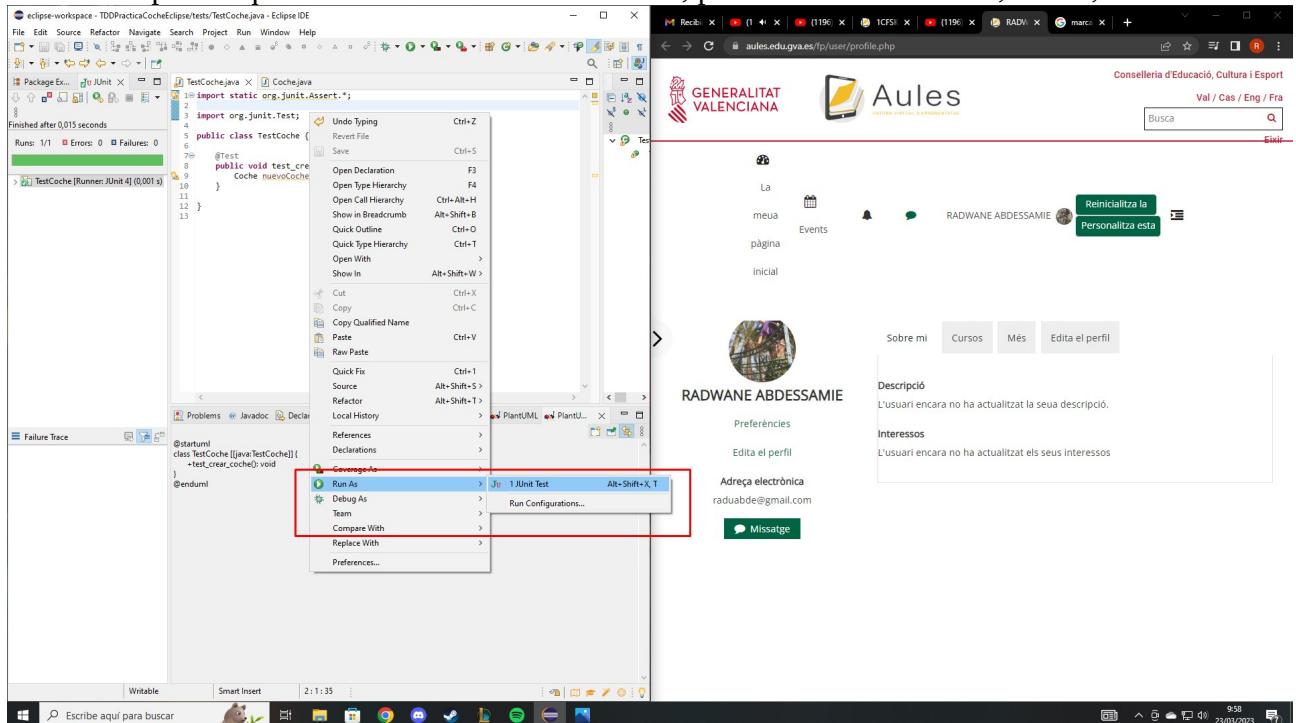
6.Creamos un metodo test en la clase recién creada. Nos marca en rojo Coche porque la clase no existe, así que nos situamos sobre el error y elegimos la opción Create class.



7.Al seleccionar la opción de crear clase, para que dicha clase no se cree en la carpeta test, cambiamos la ruta a /src

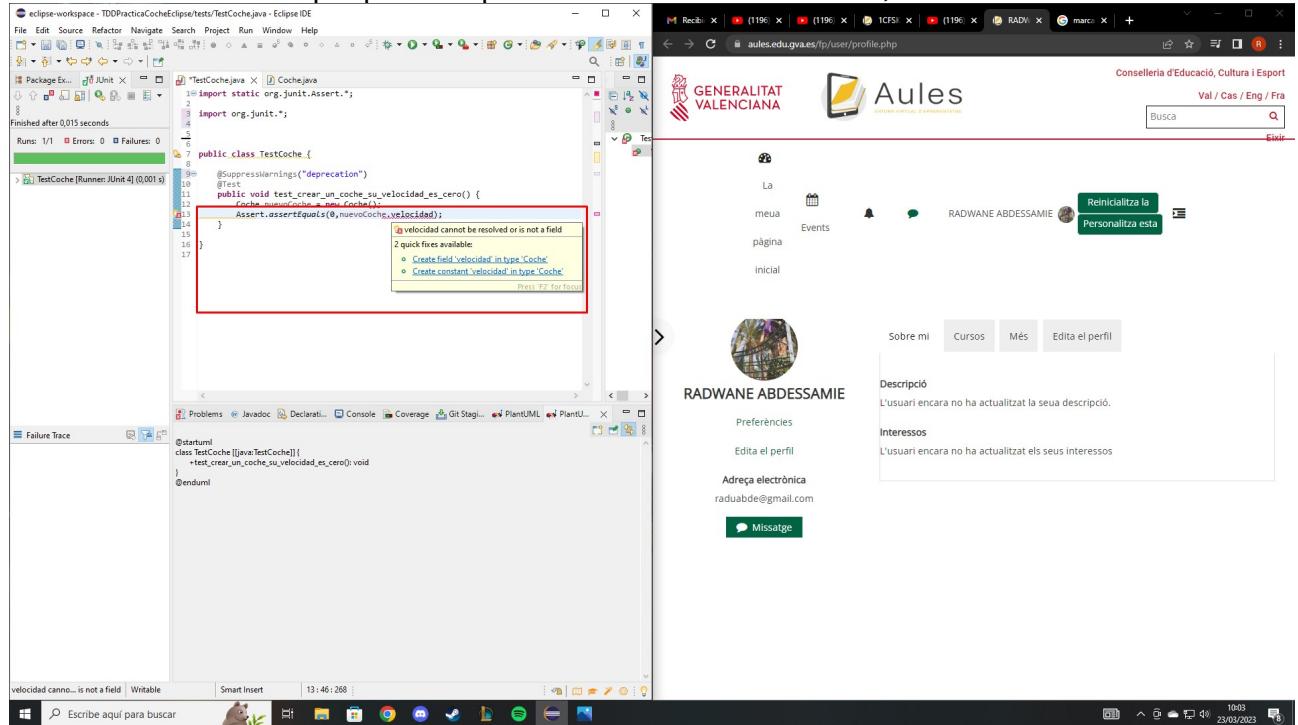


8.Para comprobar que el test funciona correctamente, pulsamos click derecho, run as, Junit Test

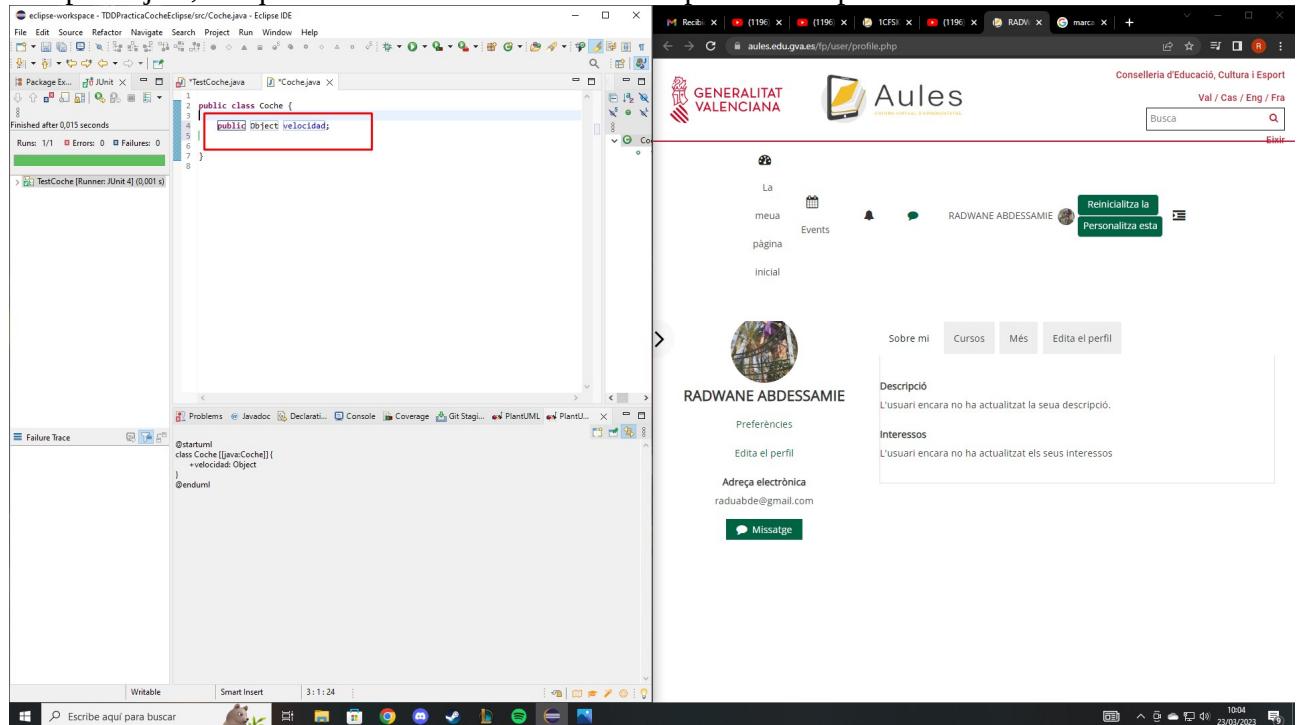


9.Creamos un método para comprobar que la velocidad de un coche al ser creado debe ser 0. Nos marca en rojo velocidad porque dicho atributo no existe en la clase.

Nota: En el caso de eclipse para comprobar valores usaremos Assert, no Assertions de IntelliJ.



10.Al seleccionar la opción que nos brinda Eclipse para crear el campo velocidad, este sera creado de tipo Object, así que nosotros debemos definir el tipo correcto para el caso.



11.en este caso realizamos lo mismo pero con un método llamado acelerar.

The screenshot shows the Eclipse IDE interface on the left and a user profile page on the right. In the Eclipse IDE, the code editor displays a Java test class named 'TestCoche.java'. A specific line of code is highlighted with a red box:

```
19+     @Test
20+     public void test_al_acelerar_un_coche_su_velocidad_aumenta() {
21+         Coche nuevoCoche = new Coche();
22+         nuevoCoche.acelerar(20);
23+         Assert.assertEquals(20, nuevoCoche.velocidad);
24+
25+     }
```

A tooltip appears over the 'acelerar' method call, indicating it is undefined for the type 'Coche'. Below the tooltip, two quick fixes are listed: 'Create method acelerar(int)' and 'Add cast to 'nuevoCoche''. The Eclipse status bar at the bottom shows the message 'The method ac...he type Coche'.

On the right, a user profile page for 'RADWANE ABDESSAMIE' is displayed. The profile includes basic information like a photo, a short bio ('L'usuari encara no ha actualitzat la seua descripció.'), and interest sections ('Interessos').

12.Creamos el método mediante Eclipse y le damos la lógica adecuada.

The screenshot shows the Eclipse IDE interface on the left and a user profile page on the right. In the Eclipse IDE, the code editor displays the completed Java code for the 'Coche' class and its test class 'TestCoche.java'. The 'acelerar' method is now defined in the 'Coche' class:

```
1  public class Coche {
2
3     public int velocidad;
4
5     public void acelerar(int aceleracion) {
6         velocidad += aceleracion;
7     }
8
9 }
```

The Eclipse status bar at the bottom shows the message 'Writable'.

On the right, the user profile page for 'RADWANE ABDESSAMIE' is displayed, showing the updated code in the 'Sobre mi' section: 'L'usuari encara no ha actualitzat la seua descripció.'.

13. Realizamos el mismo paso pero con un método que comprueba que un coche al frenar su velocidad disminuye.

The screenshot displays two windows side-by-side. On the left is the Eclipse IDE interface, specifically the Java editor. It shows a file named 'TestCoche.java' with the following code:

```
1 package com.example;
2 import org.junit.*;
3
4 public class TestCoche {
5     @Test
6     public void test_crear_un_coche_su_velocidad_es_cero() {
7         Coche nuevoCoche = new Coche();
8         Assert.assertEquals(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(50);
15         Assert.assertEquals(50,nuevoCoche.velocidad);
16     }
17
18     @Test
19     public void test_al_decelerar_un_coche_su_velocidad_disminuye() {
20         Coche nuevoCoche = new Coche();
21         nuevoCoche.acelerar(50);
22         nuevoCoche.decelerar(20);
23         Assert.assertEquals(30,nuevoCoche.velocidad);
24     }
25
26     @Test
27     public void test_al_decelerar_un_coche_su_velocidad_disminuye() {
28         Coche nuevoCoche = new Coche();
29         nuevoCoche.acelerar(50);
30         nuevoCoche.decelerar(20);
31         Assert.assertEquals(30,nuevoCoche.velocidad);
32     }
33 }
```

A red box highlights the line `nuevoCoche.decelerar(20);` because it is undefined for the type `Coche`. On the right is a web browser displaying a user profile page for 'RADWANE ABDESSAMIE' on the 'Aules' platform. The profile includes basic information like name, email, and a 'Personaliza esta' button.

14. Creamos el método mediante Eclipse y le damos la lógica adecuada.

This screenshot shows the Eclipse IDE and a web browser again. The Eclipse window now displays the 'Coche.java' file with the following code:

```
1 package com.example;
2
3 public class Coche {
4     public int velocidad;
5
6     public void acelerar(int aceleracion) {
7         velocidad += aceleracion;
8     }
9
10    public void decelerar(int deceleracion) {
11        velocidad -= deceleracion;
12    }
13
14 }
```

A red box highlights the newly added `decelerar(int)` method. The web browser window on the right shows the same user profile for 'RADWANE ABDESSAMIE'.

15.Comprobamos que los tres test funcionan correctamente.

The screenshot shows the Eclipse IDE interface with the TestCoche.java file open. The code contains three JUnit tests: `test_crear_un_coche_su_velocidad_es_cero()`, `test_al_acelerar_un_coche_su_velocidad_aumenta()`, and `test_al_decelerar_un_coche_su_velocidad_dismuye()`. The first test passes, while the second and third fail. The failure details are visible in the Failure Trace view. To the right, a browser window displays the user profile of RADWANE ABDESSAMIE on the Aules platform, showing basic information like name, email, and interests.

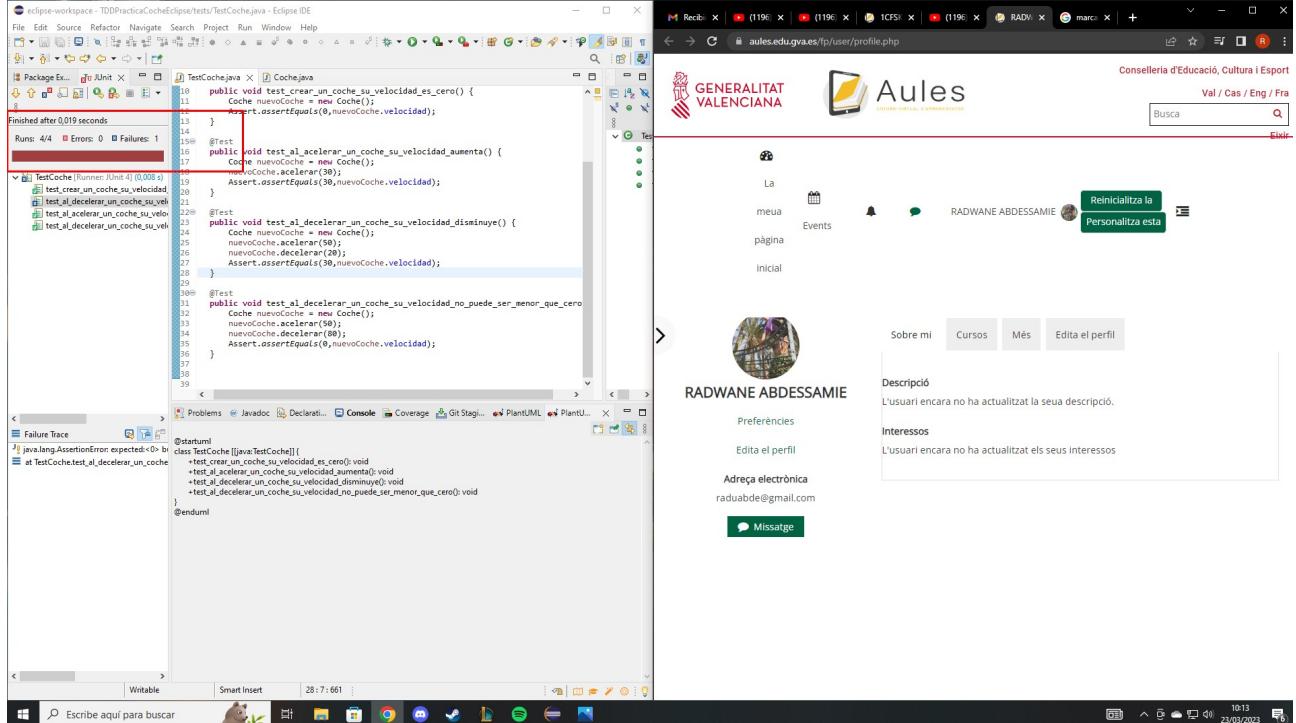
```
Finished after 0.012 seconds
Runs: 3/3 Errors: 0 Failures: 0
1 import org.junit.*;
2
3 public class TestCoche {
4
5     @Test
6     public void test_crear_un_coche_su_velocidad_es_cero() {
7         Coche nuevoCoche = new Coche();
8         nuevoCoche.acelerar(30);
9         Assert.assertEquals(0,nuevoCoche.velocidad);
10    }
11
12    @Test
13    public void test_al_acelerar_un_coche_su_velocidad_aumenta() {
14        Coche nuevoCoche = new Coche();
15        nuevoCoche.acelerar(30);
16        nuevoCoche.acelerar(20);
17        Assert.assertEquals(50,nuevoCoche.velocidad);
18    }
19
20    @Test
21    public void test_al_decelerar_un_coche_su_velocidad_dismuye() {
22        Coche nuevoCoche = new Coche();
23        nuevoCoche.acelerar(30);
24        nuevoCoche.decelerar(20);
25        Assert.assertEquals(10,nuevoCoche.velocidad);
26    }
27
28 }
29
30 }
```

16.Generamos un test para comprobar que la velocidad de un coche al frenar no debe ser menor a cero.

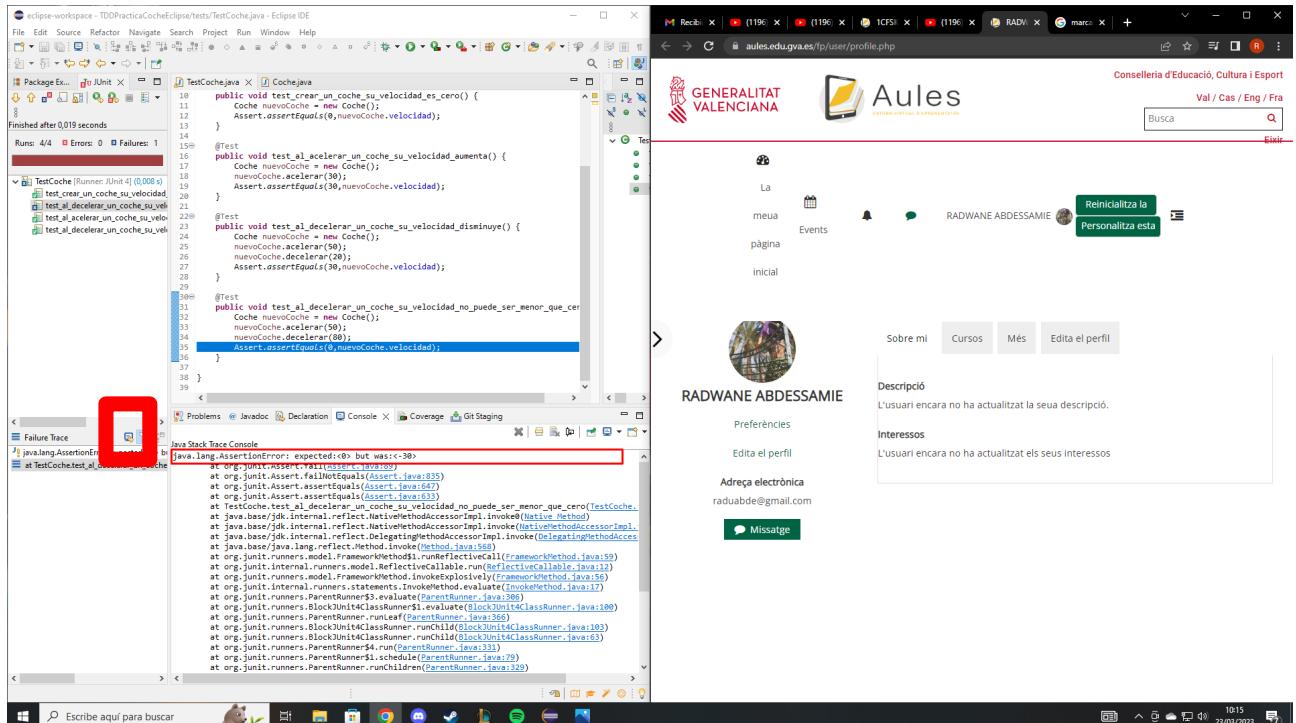
The screenshot shows the Eclipse IDE interface with the TestCoche.java file open. The code now includes a fourth test, `test_al_decelerar_un_coche_su_velocidad_no_puede_ser_menor_que_cero()`, which fails because the current implementation of the `decelerar` method does not handle negative speeds. The failure details are visible in the Failure Trace view. To the right, a browser window displays the user profile of RADWANE ABDESSAMIE on the Aules platform, showing basic information like name, email, and interests.

```
Finished after 0.012 seconds
Runs: 3/3 Errors: 0 Failures: 0
1 import org.junit.*;
2
3 public class TestCoche {
4
5     @Test
6     public void test_crear_un_coche_su_velocidad_es_cero() {
7         Coche nuevoCoche = new Coche();
8         Assert.assertEquals(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        Assert.assertEquals(30,nuevoCoche.velocidad);
16    }
17
18    @Test
19    public void test_al_decelerar_un_coche_su_velocidad_dismuye() {
20        Coche nuevoCoche = new Coche();
21        nuevoCoche.acelerar(50);
22        nuevoCoche.decelerar(30);
23        Assert.assertEquals(20,nuevoCoche.velocidad);
24    }
25
26    @Test
27    public void test_al_decelerar_un_coche_su_velocidad_no_puede_ser_menor_que_cero() {
28        Coche nuevoCoche = new Coche();
29        nuevoCoche.acelerar(50);
30        nuevoCoche.decelerar(90);
31        Assert.assertEquals(0,nuevoCoche.velocidad);
32    }
33
34 }
35
36 }
```

17.Dicho método compilara pero dará error ya que la lógica del método desacelerar no es del todo correcta.



18.En la ventana de la izquierda de Junit, si seleccionamos el método que da error y abajo, hacemos click sobre el ícono marcado en la imagen, podremos ver la pila de errores detallada en la consola como se muestra a continuación



19. Solucionamos el error de lógica en el método desacelerar

The screenshot displays the Eclipse IDE interface on the left and a web browser window on the right. In the IDE, the code for the `descelerar` method is shown:

```
public void descelerar(int desceleracion) {
    velocidad -= desceleracion;
    if (velocidad < 0) velocidad = 0;
}
```

The browser window shows a user profile for "RADWANE ABDESSAMIE" on the "Aules" platform. The profile includes basic information like a profile picture, a link to "Sobre mi", and sections for "Descripció" and "Interessos".

20. Comprobamos que ahora no hay ningún error.

The screenshot shows the Eclipse IDE and a web browser again. This time, the test results in the IDE indicate that all tests have passed (0 errors, 0 failures). The browser shows the same user profile for "RADWANE ABDESSAMIE".