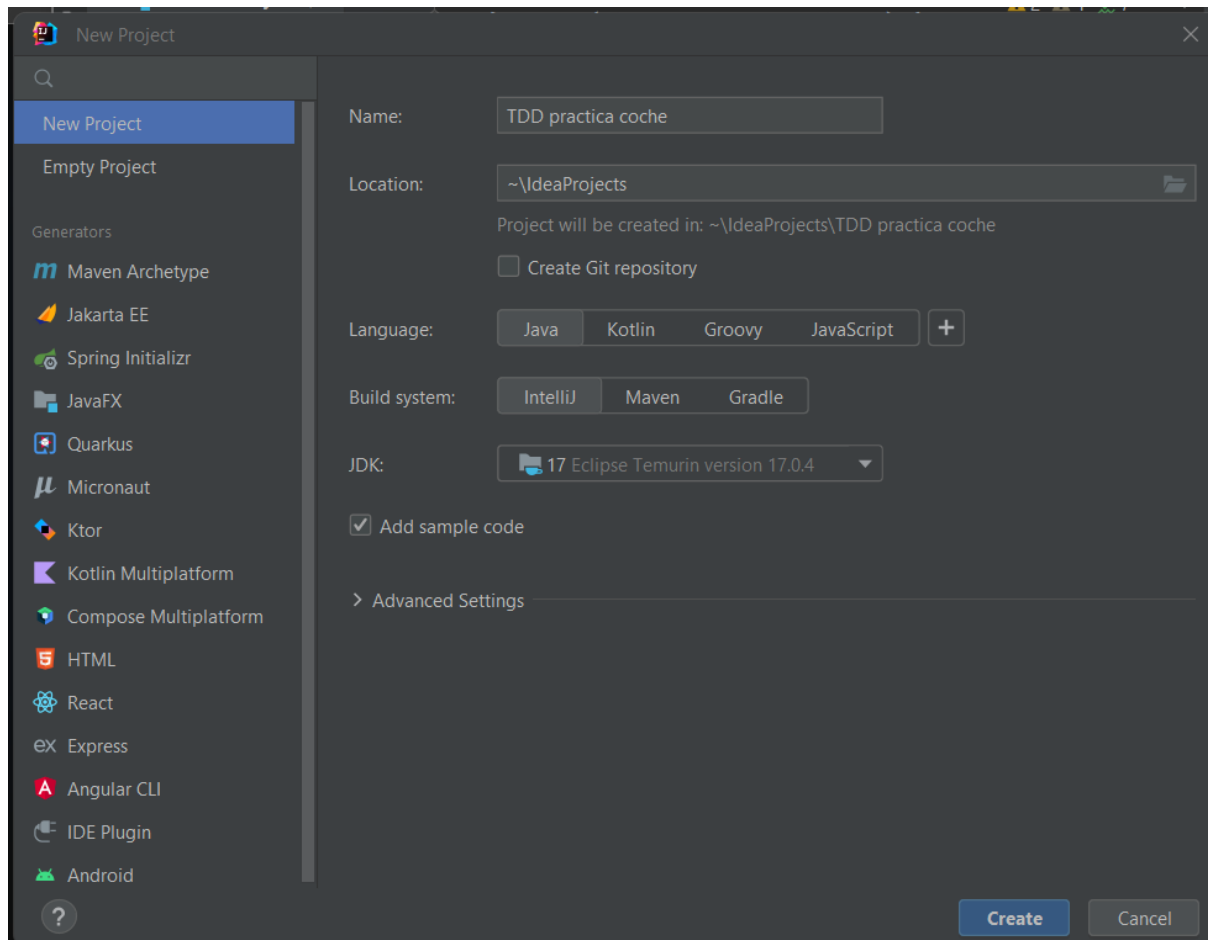
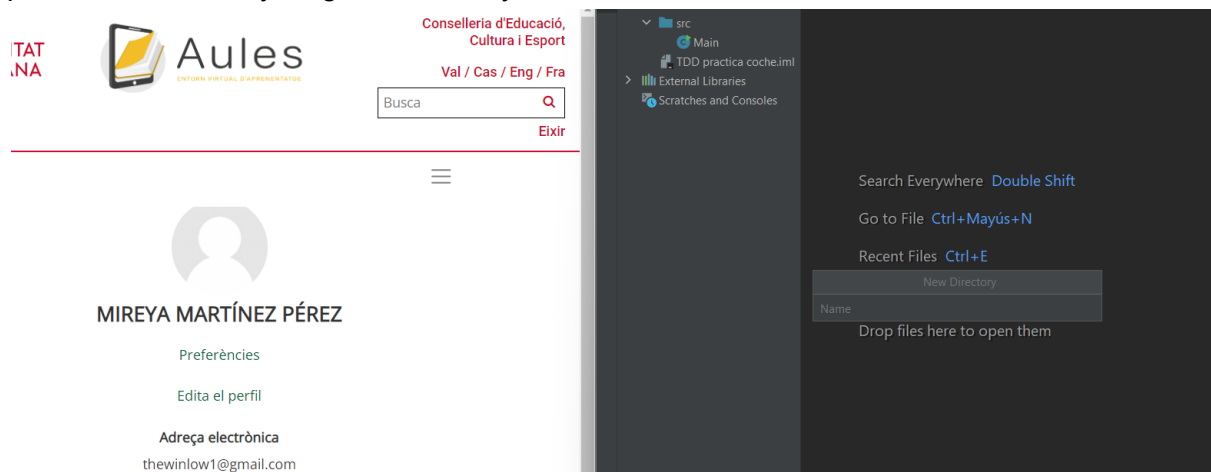


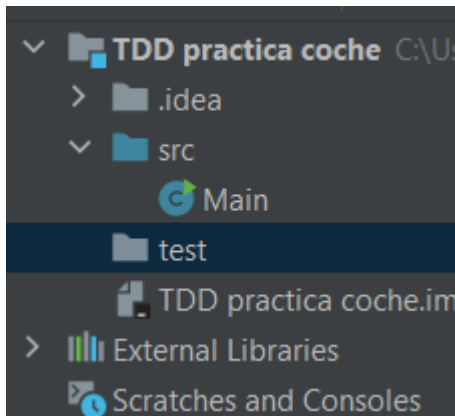
Primero creamos un nuevo proyecto java en IntelliJ pinchando en File, New Project. A continuación ponemos el nombre TDD practica coche y pinchamos en Create:



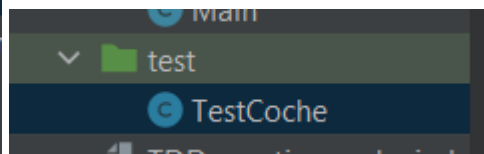
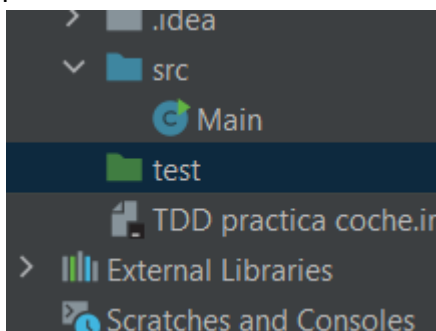
Una vez creado el proyecto hacemos click derecho sobre la carteta TDD practica coche, pinchamos en new y luego en directory



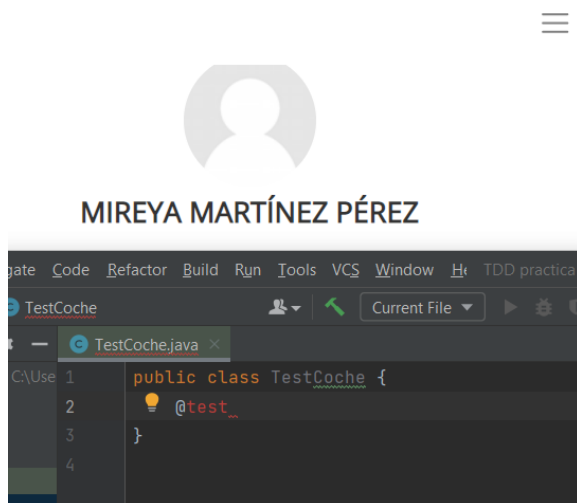
A continuación le ponemos el nombre al directorio en este caso test. Hacemos click derecho en test, después pinchamos en mark directory as y luego en test sources root.



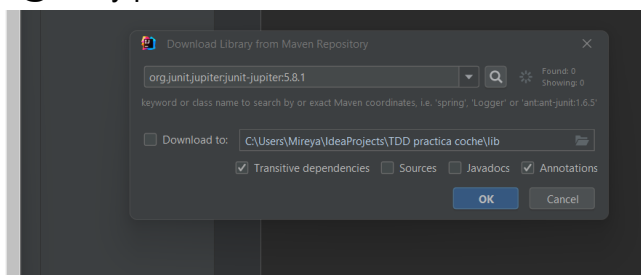
Vemos cómo ha cambiando el color del directorio, a continuación hacemos click derecho en tests (otra vez) y esta vez pinchamos en new y después en java class y de nombre le ponemos TestCoche



A continuación en la clase que acabamos de crear escribimos @Test y como vemos nos aparecerá en rojo

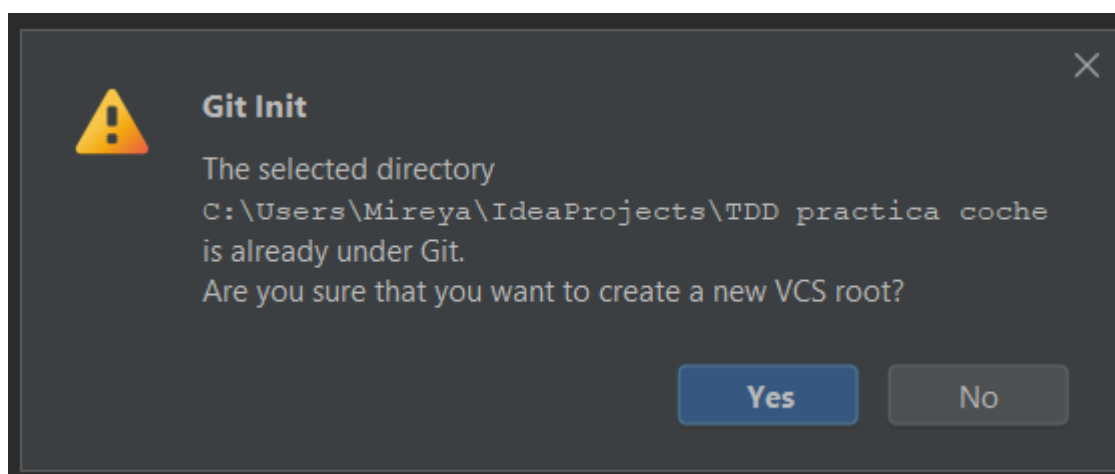
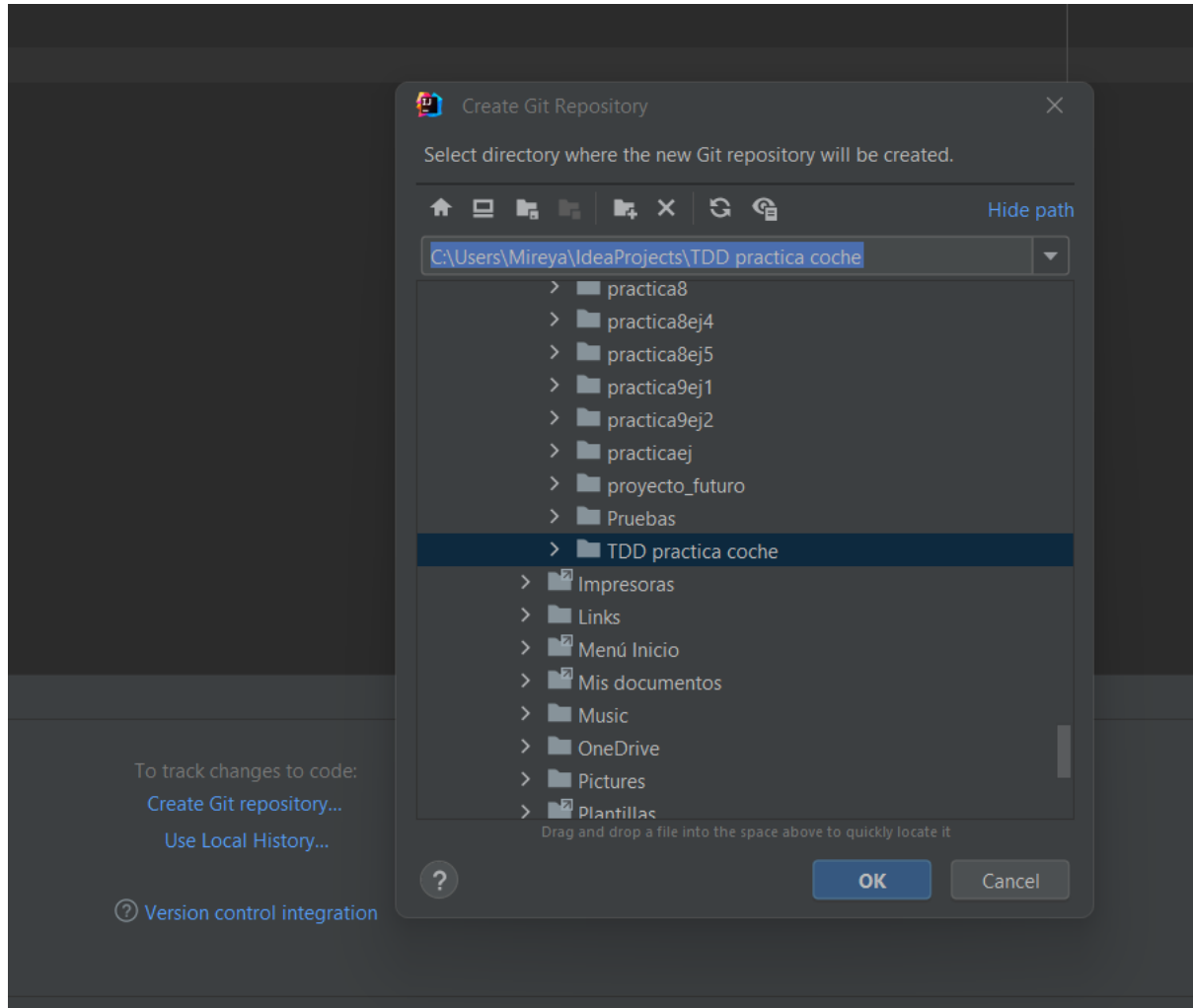


a continuación hacemos click derecho sobre @Test y pinchamos en add JUnit 5.8

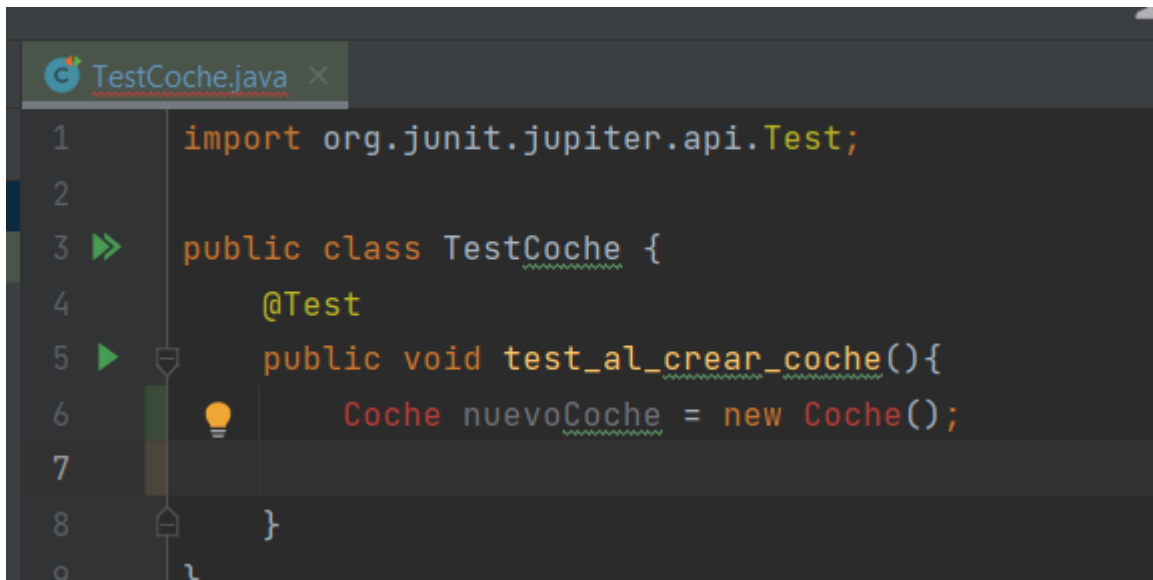


Nos aparecerá lo el mensaje anterior, marcamos la opción de la derecha “Annotations” y luego pinchamos en ok

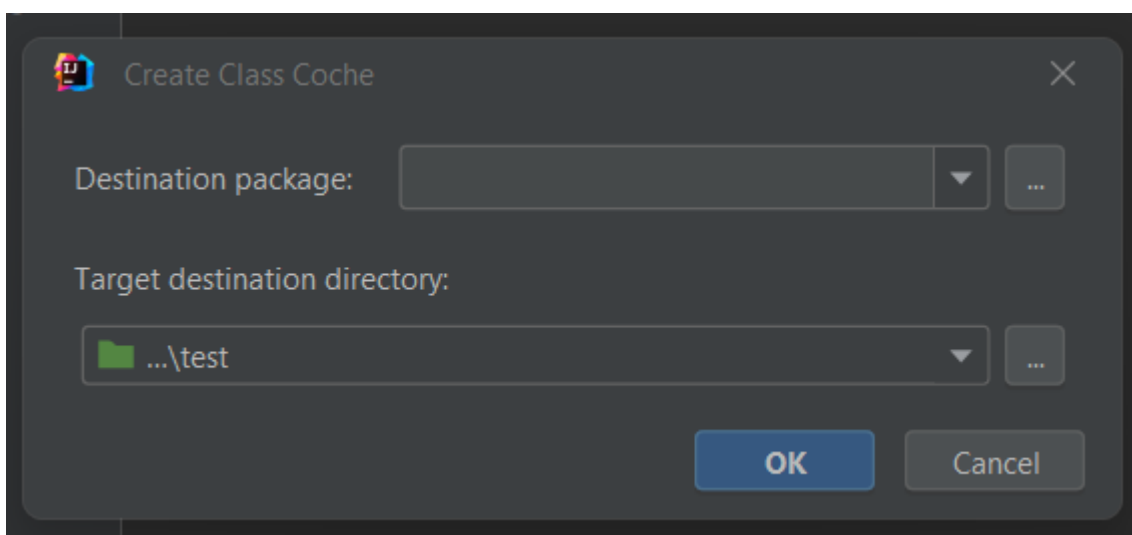
A continuación iniciaremos git para ir haciendo commits a lo largo de la práctica



Después del primer commit volvemos a la clase que hemos creado y hacemos el primer test, en el cual intentamos un objeto coche que como no tenemos su clase creada aparecerá en rojo, hacemos click derecho encima y nos dará una opción de que el propio programa te cree la clase



```
1 import org.junit.jupiter.api.Test;
2
3 public class TestCoche {
4     @Test
5     public void test_al_crear_coche(){
6         Coche nuevoCoche = new Coche();
7
8     }
9 }
```



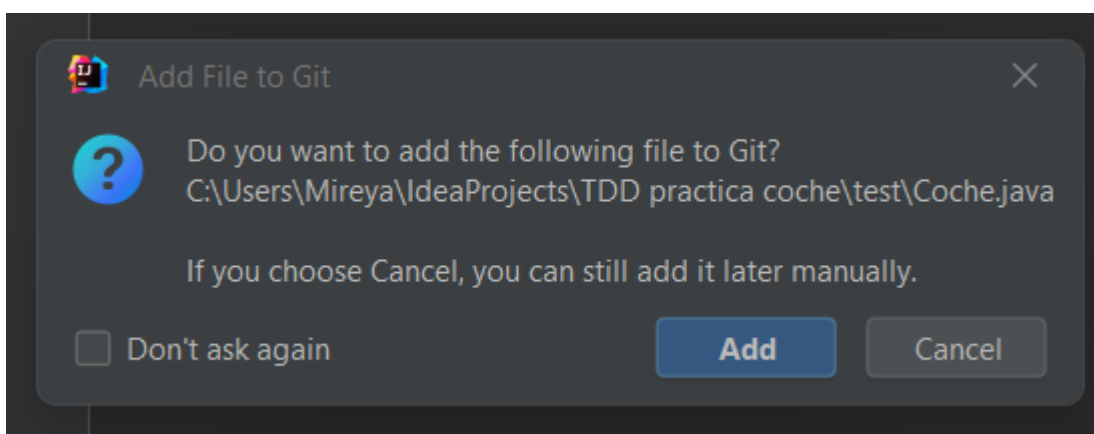
Create Class Coche

Destination package: ▼ ...

Target destination directory:

▼ ...

OK Cancel

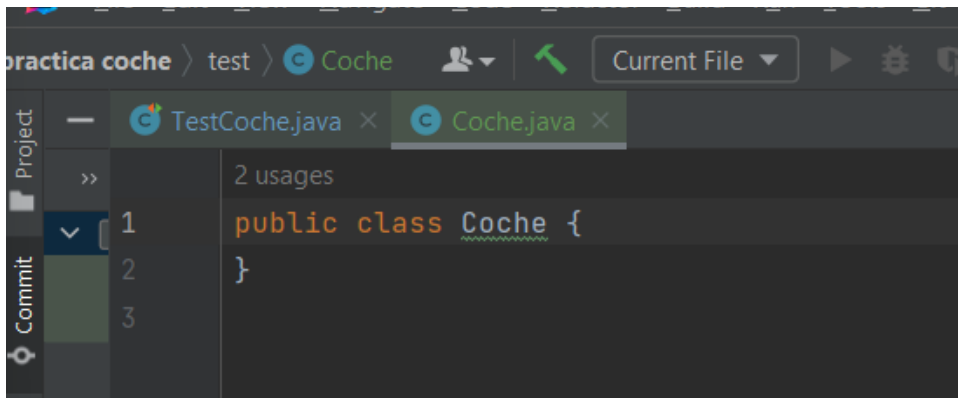
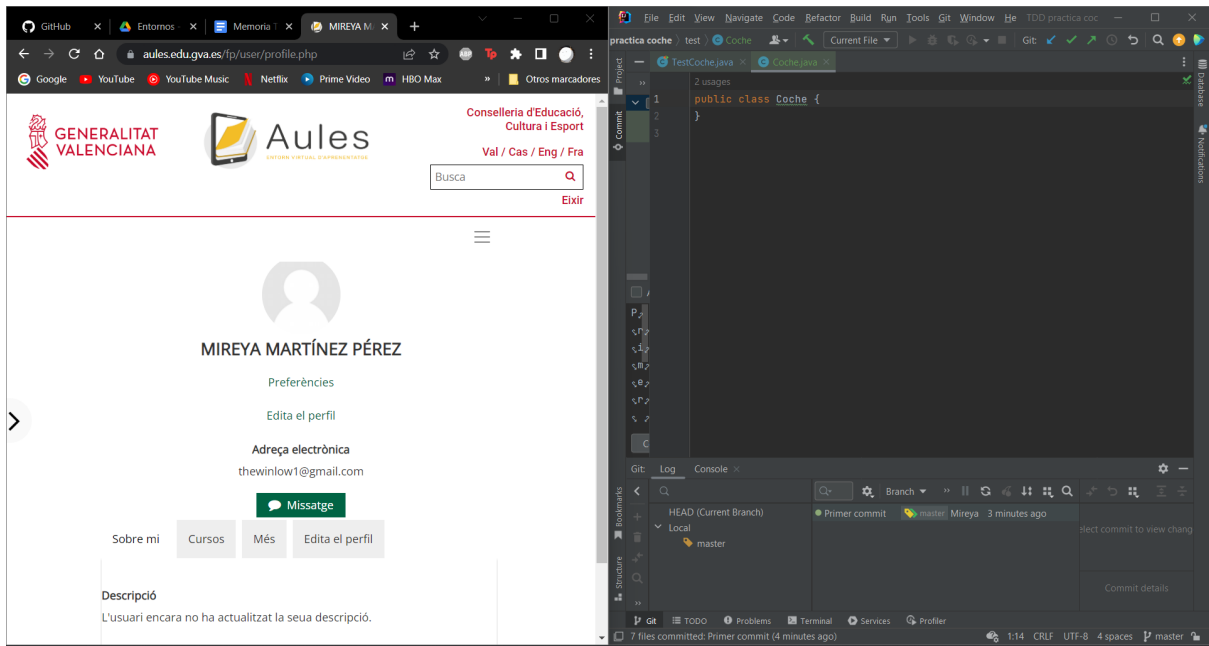


Add File to Git

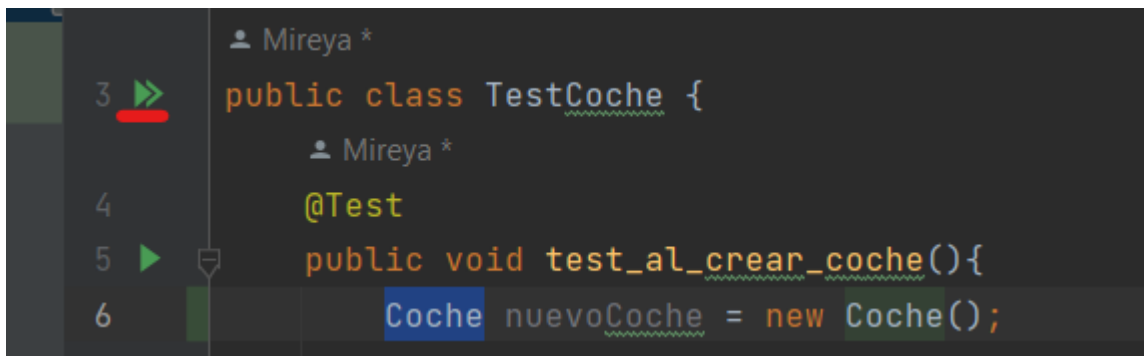
Do you want to add the following file to Git?
C:\\Users\\Mireya\\IdeaProjects\\TDD practica coche\\test\\Coche.java

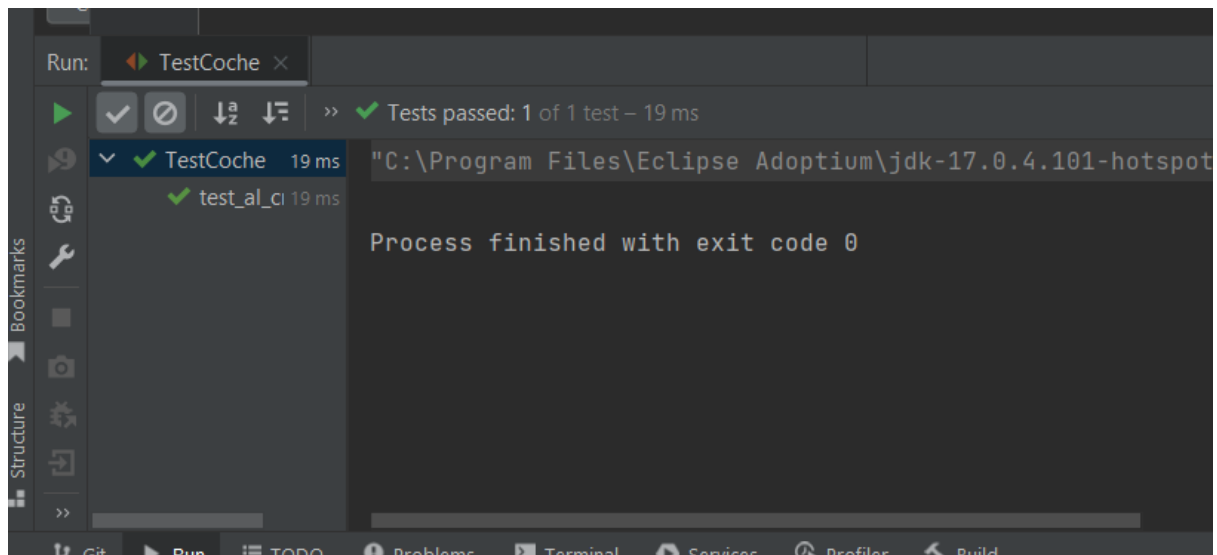
If you choose Cancel, you can still add it later manually.

☐ Don't ask again Add Cancel

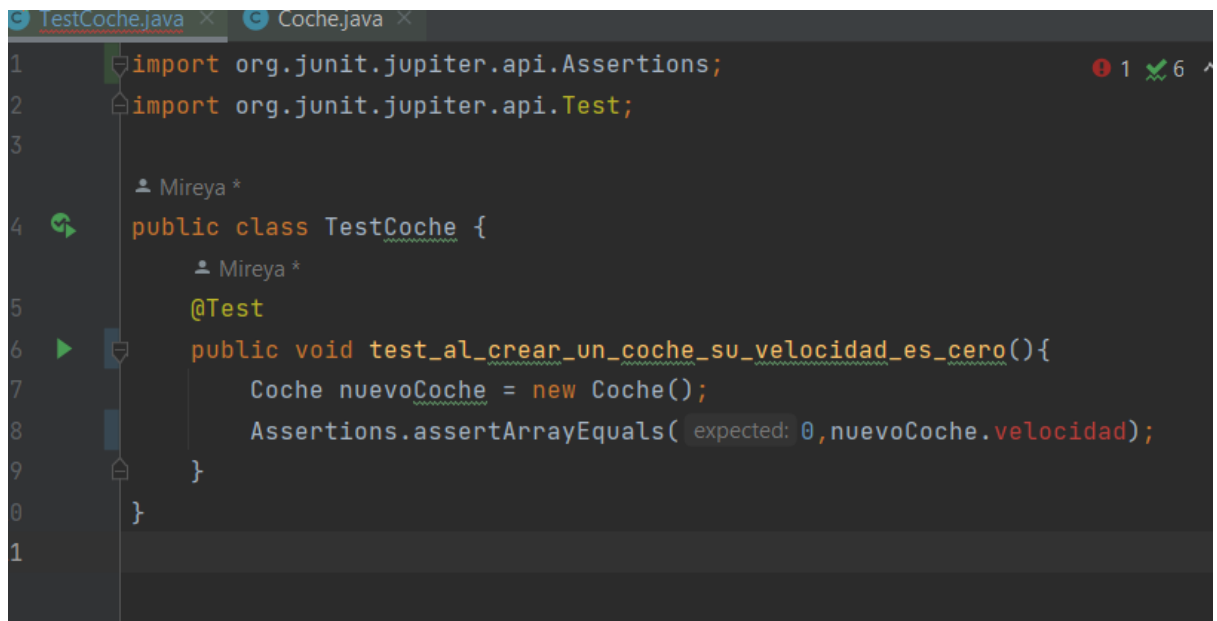


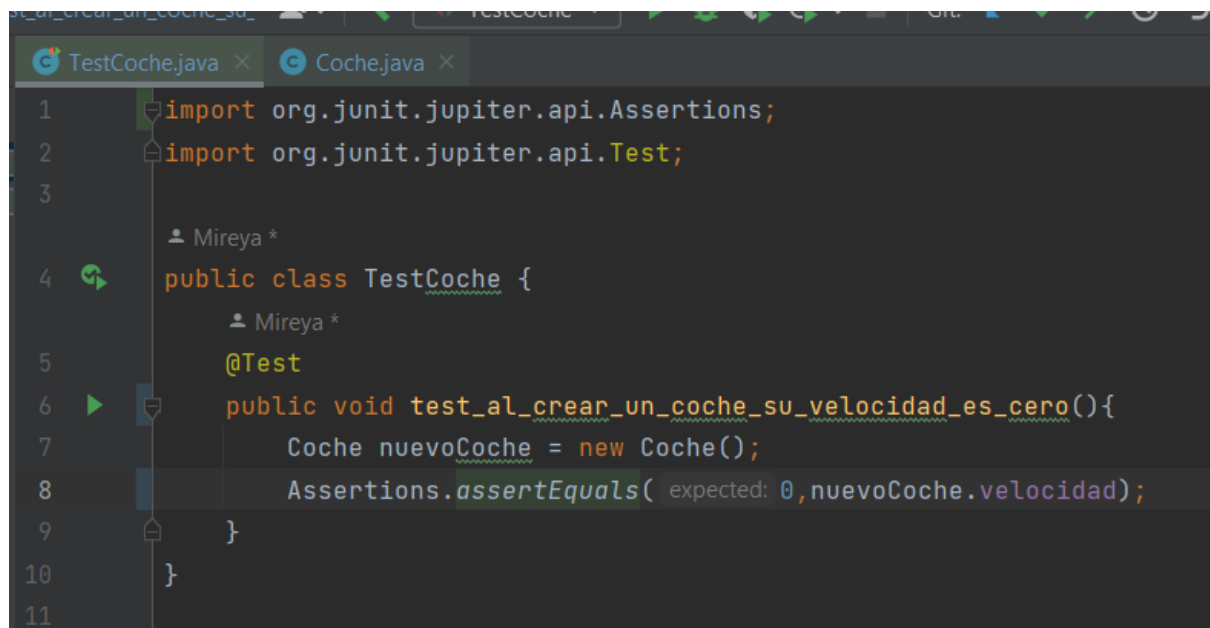
A continuación le damos a RUN y vemos si se ha hecho el tests bien





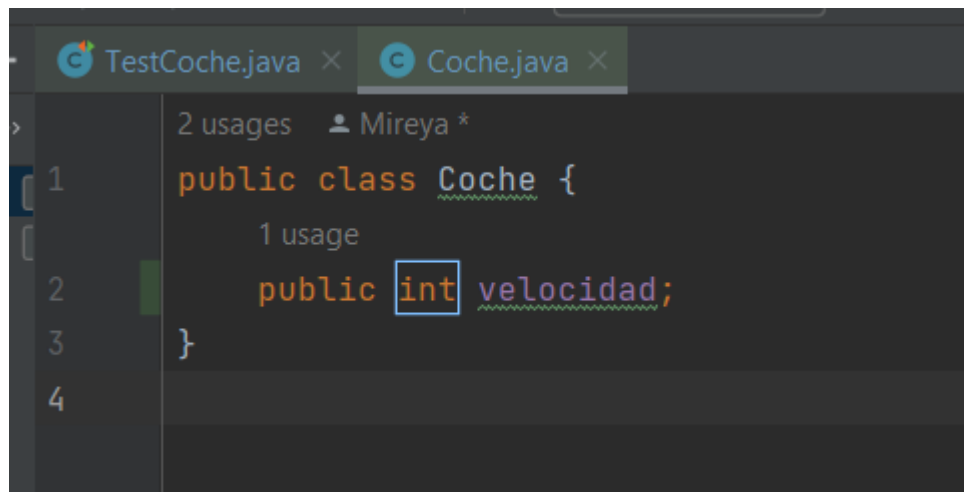
Cambiamos el test que hemos creado antes y añadimos lo que veis en la captura, como está intentado cambiar el valor de una variable que no existe en coche aparecerá en rojo, hacemos lo mismo que antes clicamos encima y nos dará la opción de crearlo automáticamente





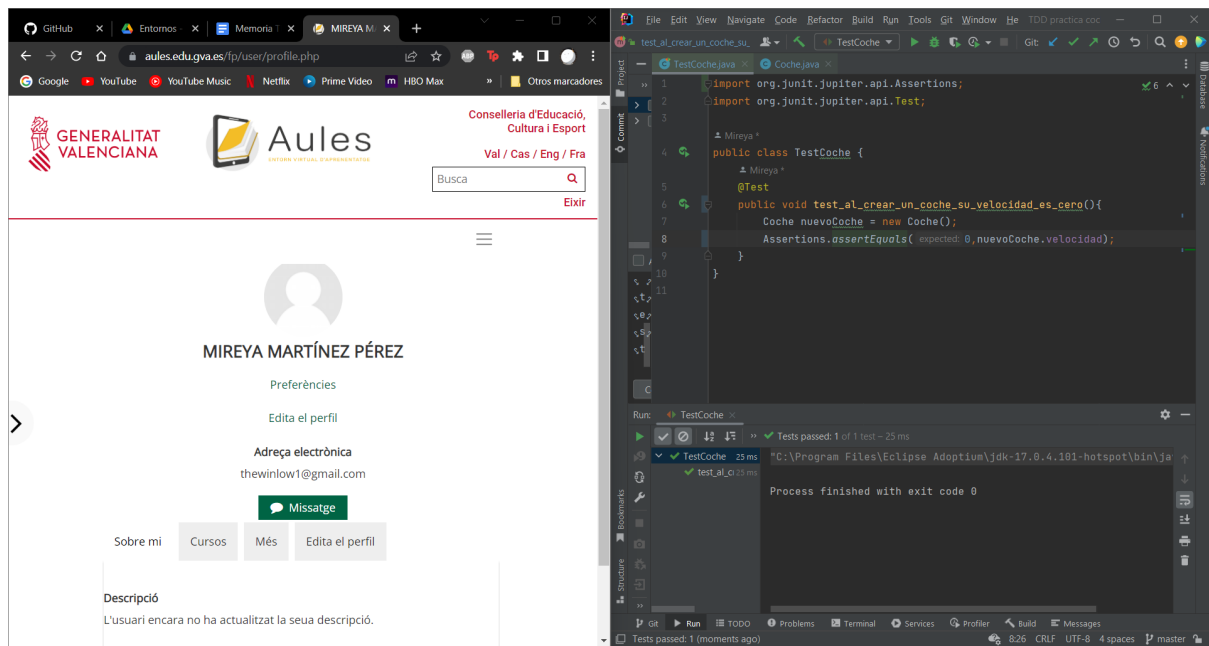
```
1 import org.junit.jupiter.api.Assertions;
2 import org.junit.jupiter.api.Test;
3
4 Mireya *
5 public class TestCoche {
6     Mireya *
7     @Test
8     public void test_al_crear_un_coche_su_velocidad_es_cero(){
9         Coche nuevoCoche = new Coche();
10        Assertions.assertEquals( expected: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

podemos ver como se ha creado la variable velocidad

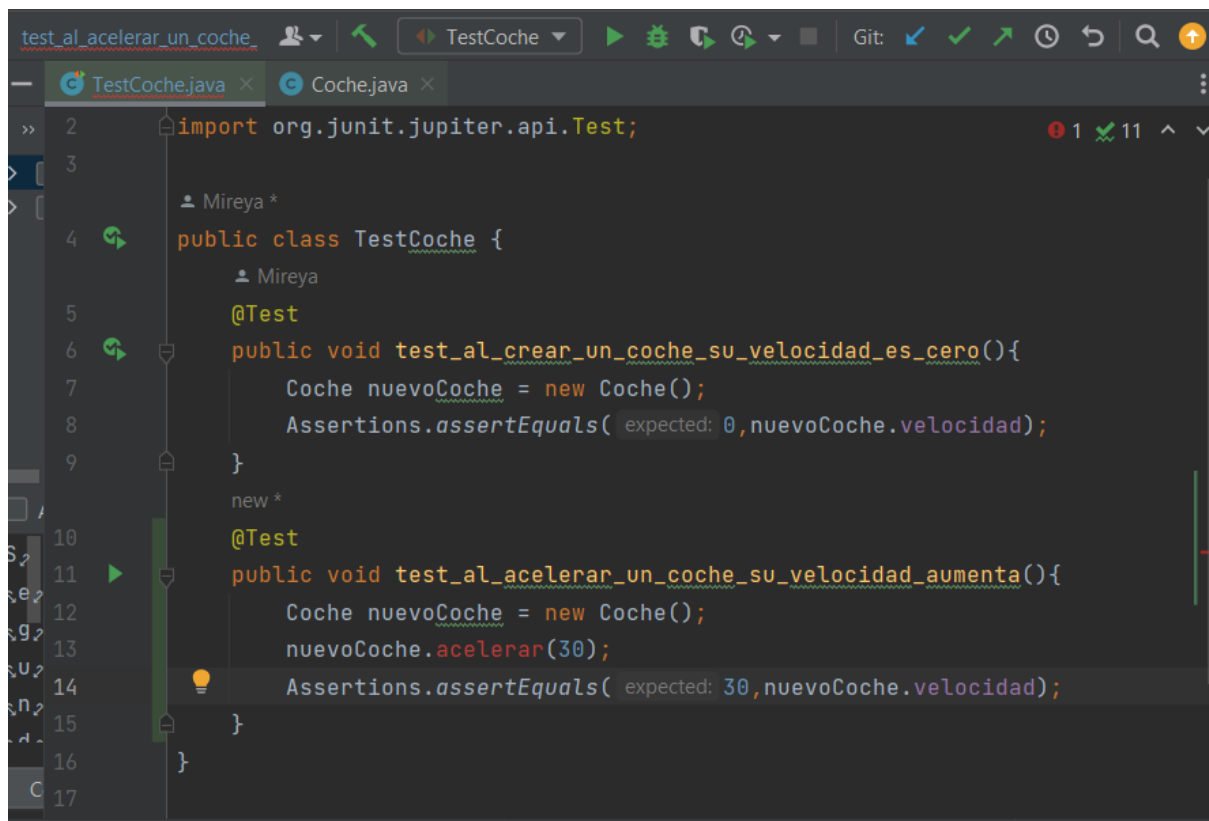


```
1 2 usages Mireya *
2 public class Coche {
3     1 usage
4     public int velocidad;
5 }
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

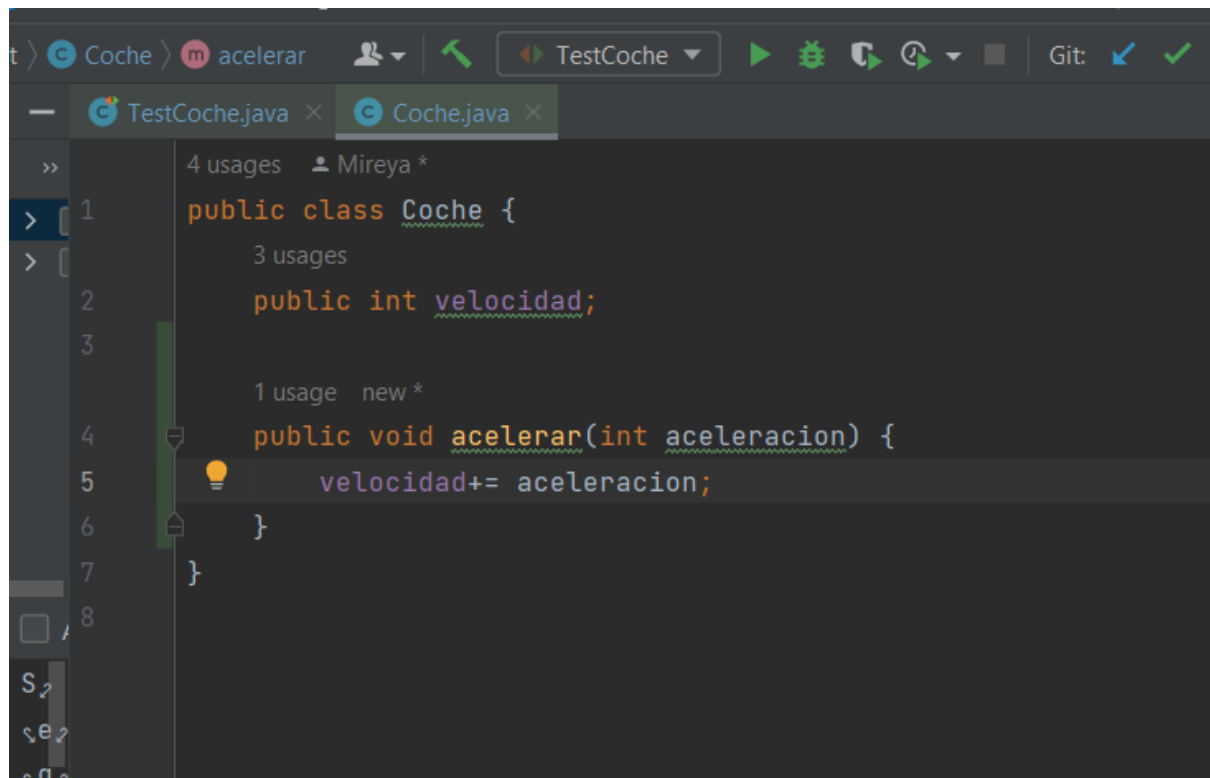
Hacemos Run y vemos que funciona bien



Añadimos otro test en el cual intentamos usar el método que aún no existe acelerar, repetimos los mismos pasos que antes

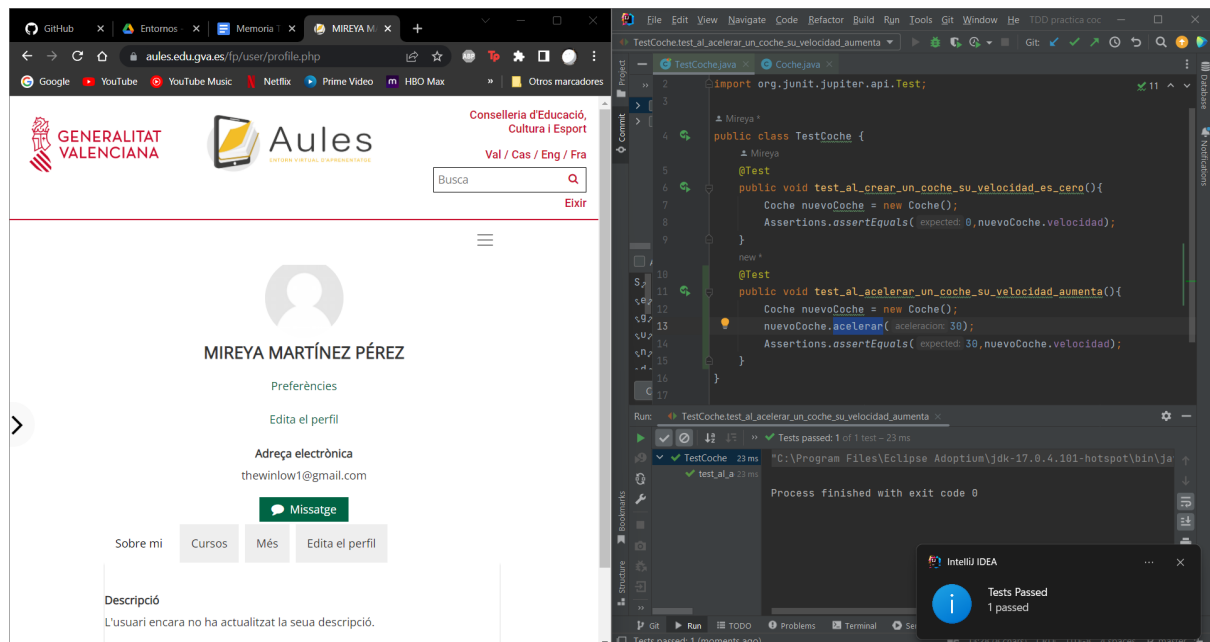


Vemos como se ha creado el método acelerar y ponemos dentro lo que veis que es para que cuando uses el metodo acelerar se suma la velocidad mas la aceleración



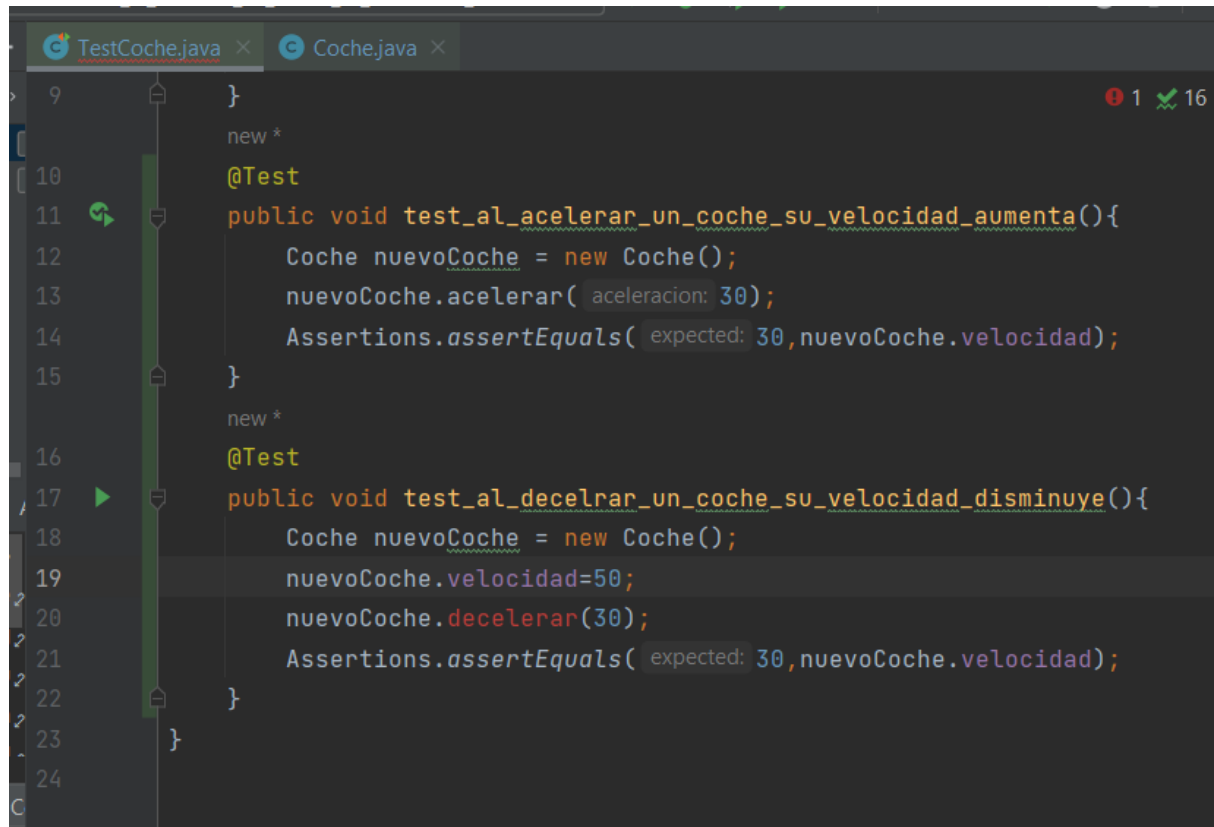
```
1 public class Coche {
2     public int velocidad;
3
4     public void acelerar(int aceleracion) {
5         velocidad+= aceleracion;
6     }
7 }
8
```

Hacemos la prueba del test y funciona perfectamente

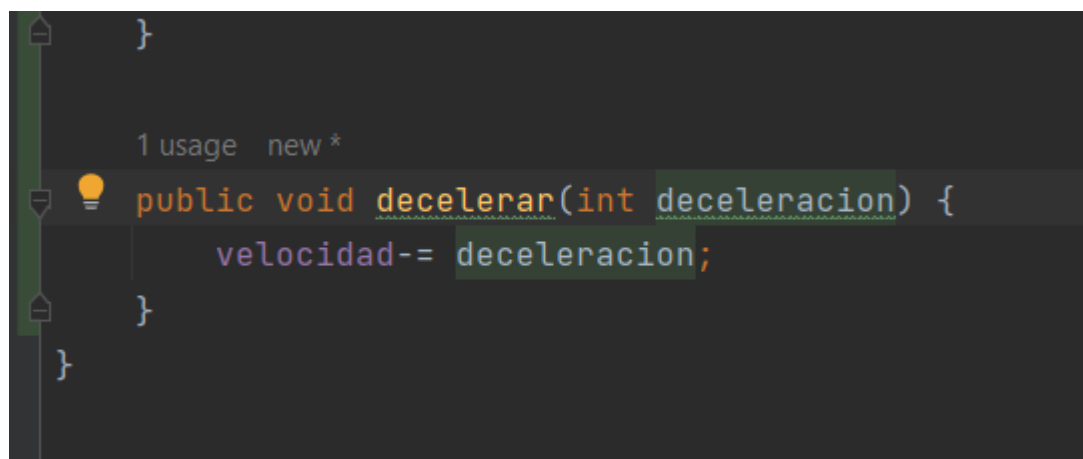


The image is a composite of two screenshots. The left screenshot shows a user profile on the Aules platform for MIREYA MARTÍNEZ PÉREZ, with options to edit the profile, view courses, and contact via email or messaging. The right screenshot shows an IDE with two unit tests for the Car class. The first test, `test_al_crear_un_coche_su_velocidad_es_cero`, verifies that a new car has a velocity of 0. The second test, `test_al_acelerar_un_coche_su_velocidad_aumenta`, verifies that calling `acelerar(30)` on a new car results in a velocity of 30. Both tests are marked as passed. A notification at the bottom right of the IDE states "Tests Passed 1 passed".

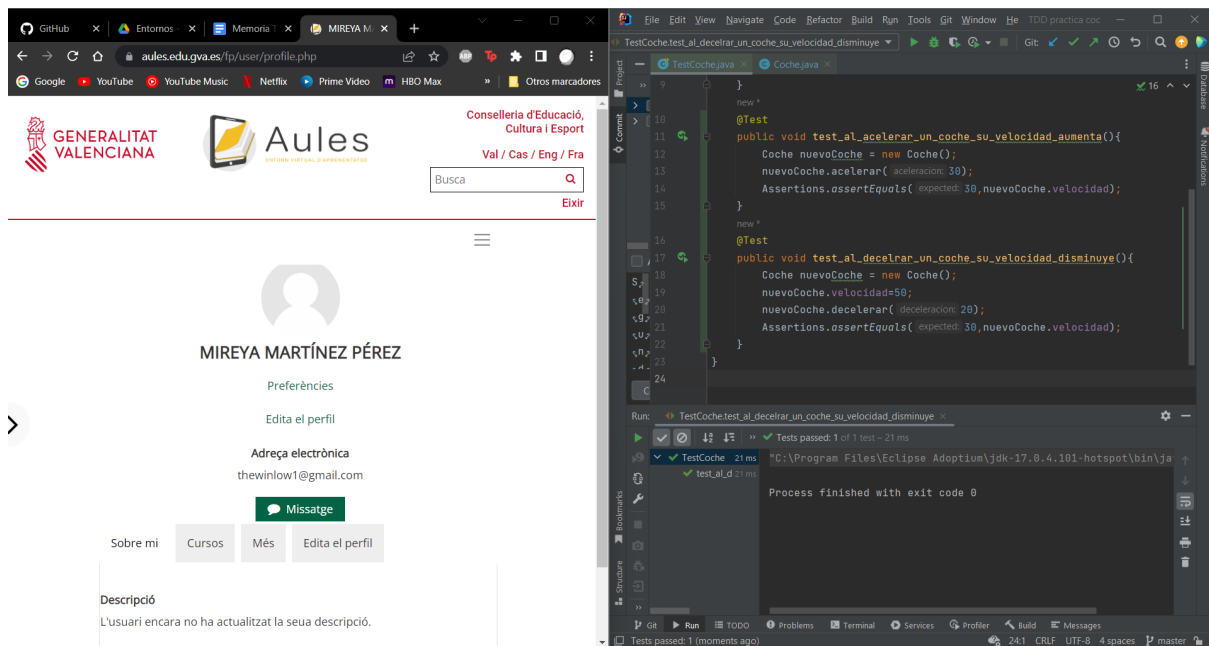
Hacemos lo mismo que antes pero con decelerar



```
9      }
10     new *
11     @Test
12     public void test_al_acelerar_un_coche_su_velocidad_aumenta(){
13         Coche nuevoCoche = new Coche();
14         nuevoCoche.acelerar( aceleracion: 30);
15         Assertions.assertEquals( expected: 30,nuevoCoche.velocidad);
16     }
17     new *
18     @Test
19     public void test_al_decelrar_un_coche_su_velocidad_disminuye(){
20         Coche nuevoCoche = new Coche();
21         nuevoCoche.velocidad=50;
22         nuevoCoche.decelerar(30);
23         Assertions.assertEquals( expected: 30,nuevoCoche.velocidad);
24     }
25 }
```



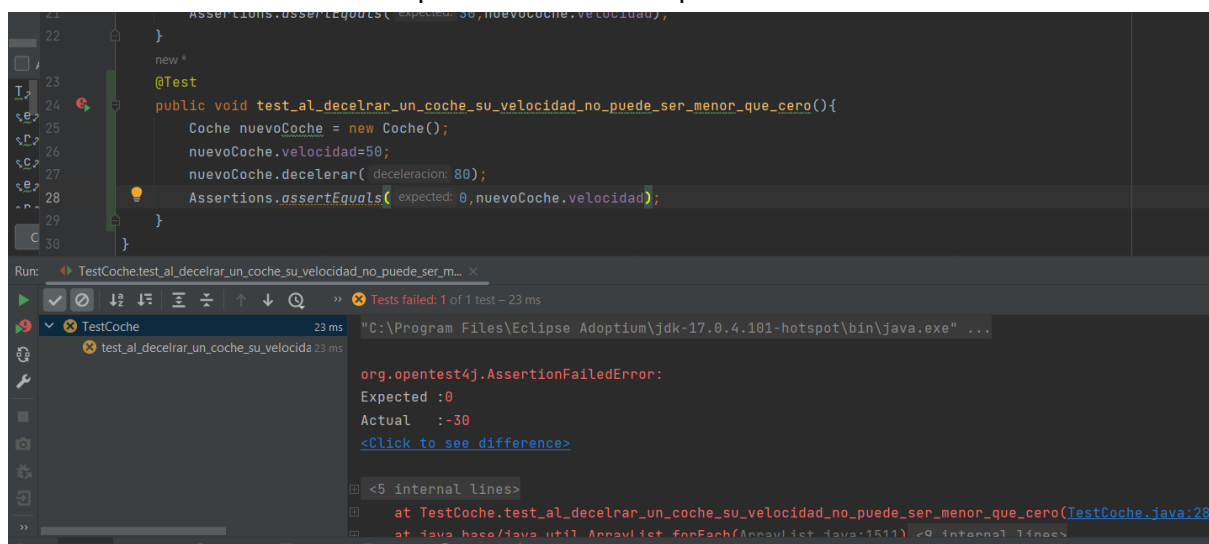
```
1 usage new *
2 public void decelerar(int deceleracion) {
3     velocidad-= deceleracion;
4 }
5 }
```



Añadimos otro test en el cual indicaremos que la velocidad nunca puede ser menor que cero



Al intentar hacer Run veremos que esta vez no compila



Nos vamos al método decelerar y añadimos el siguiente if

```
2 usages Mireya *  
public void decelerar(int deceleracion) {  
    velocidad -= deceleracion;  
    if (velocidad < 0) velocidad = 0;  
}
```

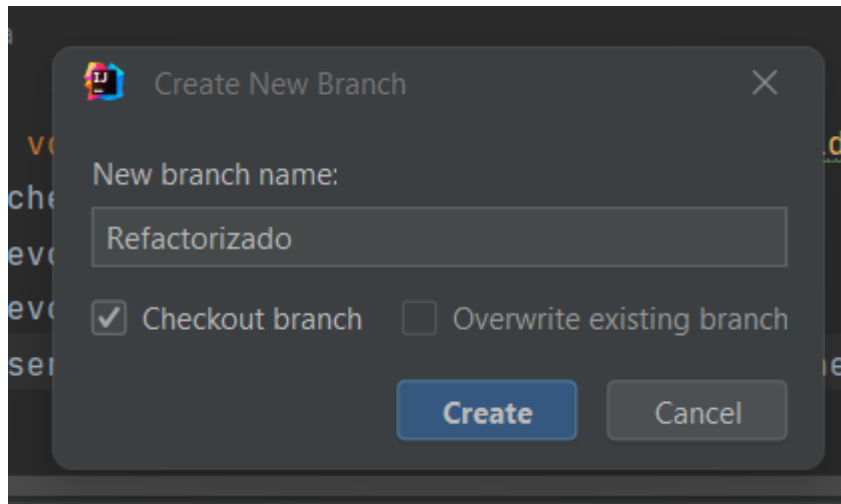
Y ahora funcionará perfecto

The screenshot is split into two parts. The left part shows a web browser with the URL `aules.edu.gva.es/fp/user/profile.php`. It displays the profile of MIREYA MARTÍNEZ PÉREZ, with options to edit the profile, view courses, and contact via email or message. The right part shows an IDE window with a Java file named `TestCoche.java`. It contains two JUnit tests: `test_al_decelnar_un_coche_su_velocidad_disminuye()` and `test_al_decelnar_un_coche_su_velocidad_no_puede_ser_menor_que_cero()`. Both tests are passing, as indicated by green checkmarks and the message "Tests passed: 1 of 1 test - 21 ms". The IDE also shows the output of the tests, confirming that the velocity is correctly updated and never drops below zero.

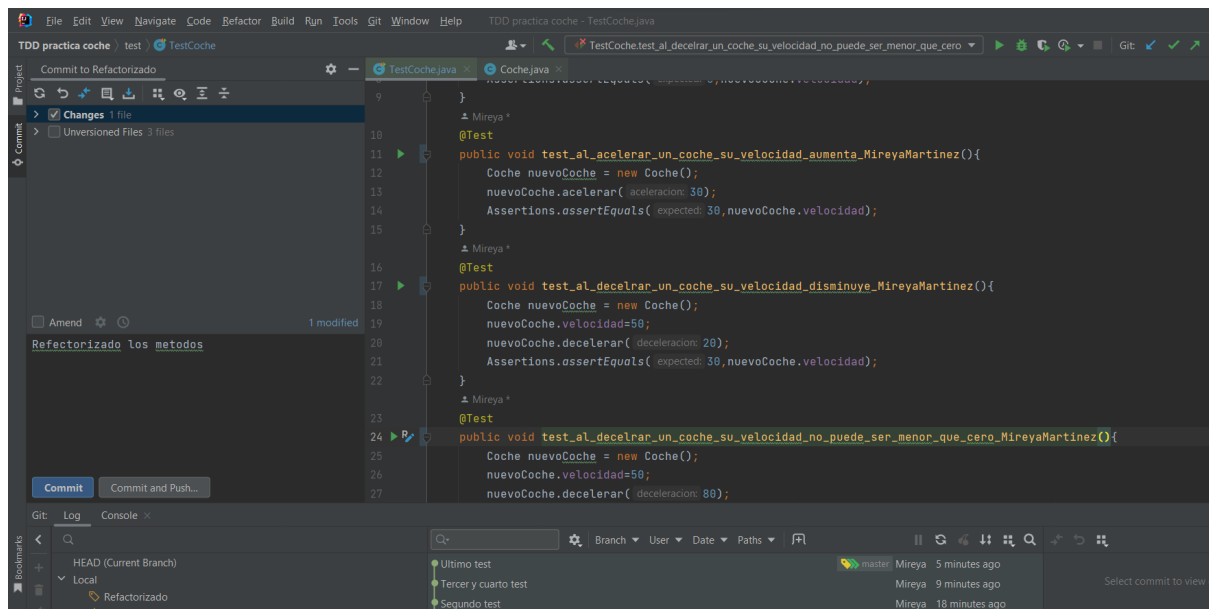
Creamos otra rama de git

The screenshot shows the IDE interface. The top part displays the test results: "Tests passed: 1 of 1 test - 21 ms" and "Process finished with exit code 0". The bottom part shows the Git Branches panel, which lists the current branch as "master". A dropdown menu is open, showing options to "New Branch", "Checkout Tag or Revision...", and "Local Branches". The "New Branch" option is highlighted. The IDE's status bar at the bottom shows the time as 28:33, the encoding as CRLF, the file encoding as UTF-8, and the indentation as 4 spaces.

Le ponemos Refactorizar de nombre a la rama que estamos creando



Y refactorizar el nombre de todos los test añadiendo mi nombre al final



Por último haremos un commit y push de todas las ramas, incluido una que se llame memoria donde introduciremos este mismo pdf