

---

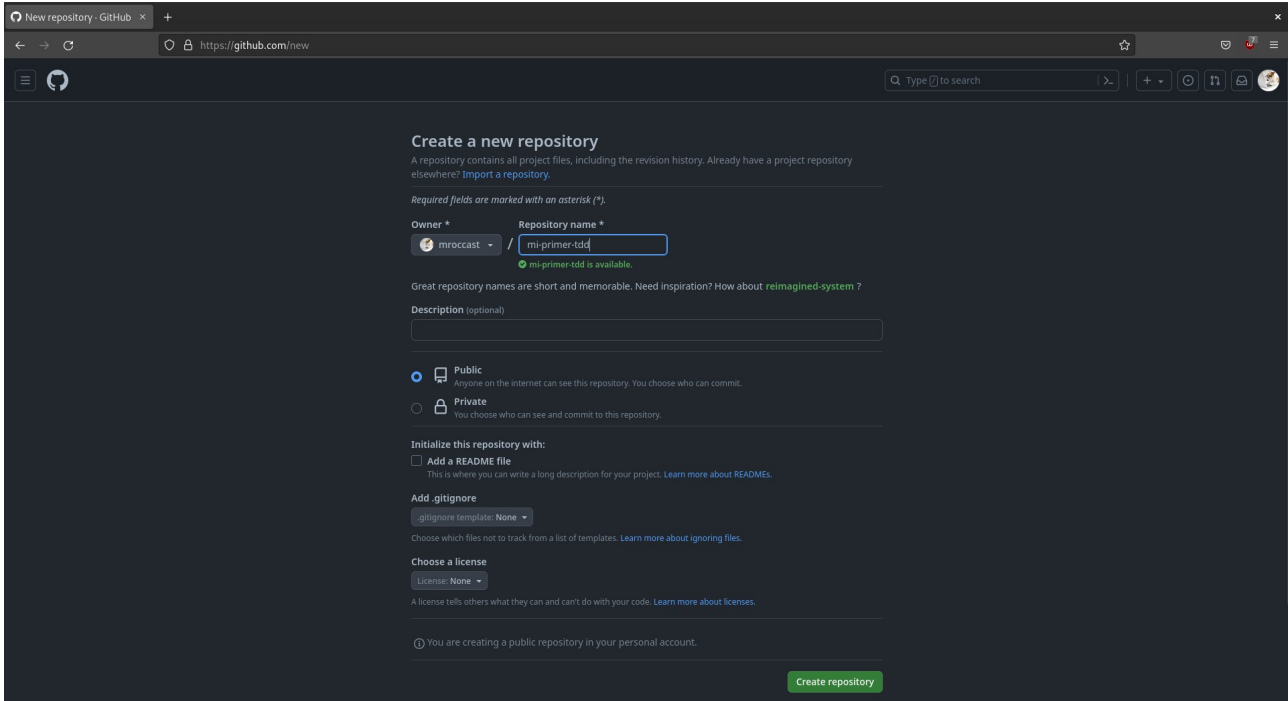
# **Mi primer TDD**

---

# Índice

1. Crear repositorio en GitHub.....	3
2. Crear proyecto en IntelliJ.....	4
3. Inicializamos el repositorio de Git.....	4
3. Primer commit del proyecto.....	6
4. Creación del directorio.....	7
5. Creación de la clase.....	10
6. Creación del test.....	13
7. Commit del test.....	15
8. Creación de la clase Coche.....	17
9. Mejoramos el test.....	19
10. Error del atributo velocidad.....	20
11. Nuevo test.....	22
11. Método decelerar.....	24
12. Test deceleración no puede ser menor que cero.....	26

## 1. Crear repositorio en GitHub



New repository - GitHub

https://github.com/new

### Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Required fields are marked with an asterisk (\*).

Owner \* / Repository name \*  
mroccast / mi-primer-tdd  
mi-primer-tdd is available.

Great repository names are short and memorable. Need inspiration? How about [reimagined-system](#)?

Description (optional)

☒ Public  
Anyone on the internet can see this repository. You choose who can commit.

☐ Private  
You choose who can see and commit to this repository.

Initialize this repository with:

☒ Add a README file  
This is where you can write a long description for your project. [Learn more about READMEs](#).

Add .gitignore  
gitignore template: None  
Choose which files not to track from a list of templates. [Learn more about ignoring files](#).

Choose a license  
License: None  
A license tells others what they can and can't do with your code. [Learn more about licenses](#).

① You are creating a public repository in your personal account.

Create repository

Figura 1: Creamos nuestro repositorio en GitHub

Para comenzar con el proyecto, crearemos un repositorio en GitHub, con el nombre “mi-primer-tdd”.

Entraremos en GitHub, seleccionamos la opción “New”, y aparecerá la opción “Create a new repository”, donde podremos poner el nombre a nuestro repositorio, y hacemos click sobre el icono verde “Create repository” para finalizar.

## 2. Crear proyecto en IntelliJ

Ahora vamos a crear un proyecto Java con IntelliJ.

Abrimos el IntelliJ, y seleccionamos “New project”, vamos a escoger el lenguaje Java, y el nombre de nuestro proyecto será “TDD Practica Coche” .

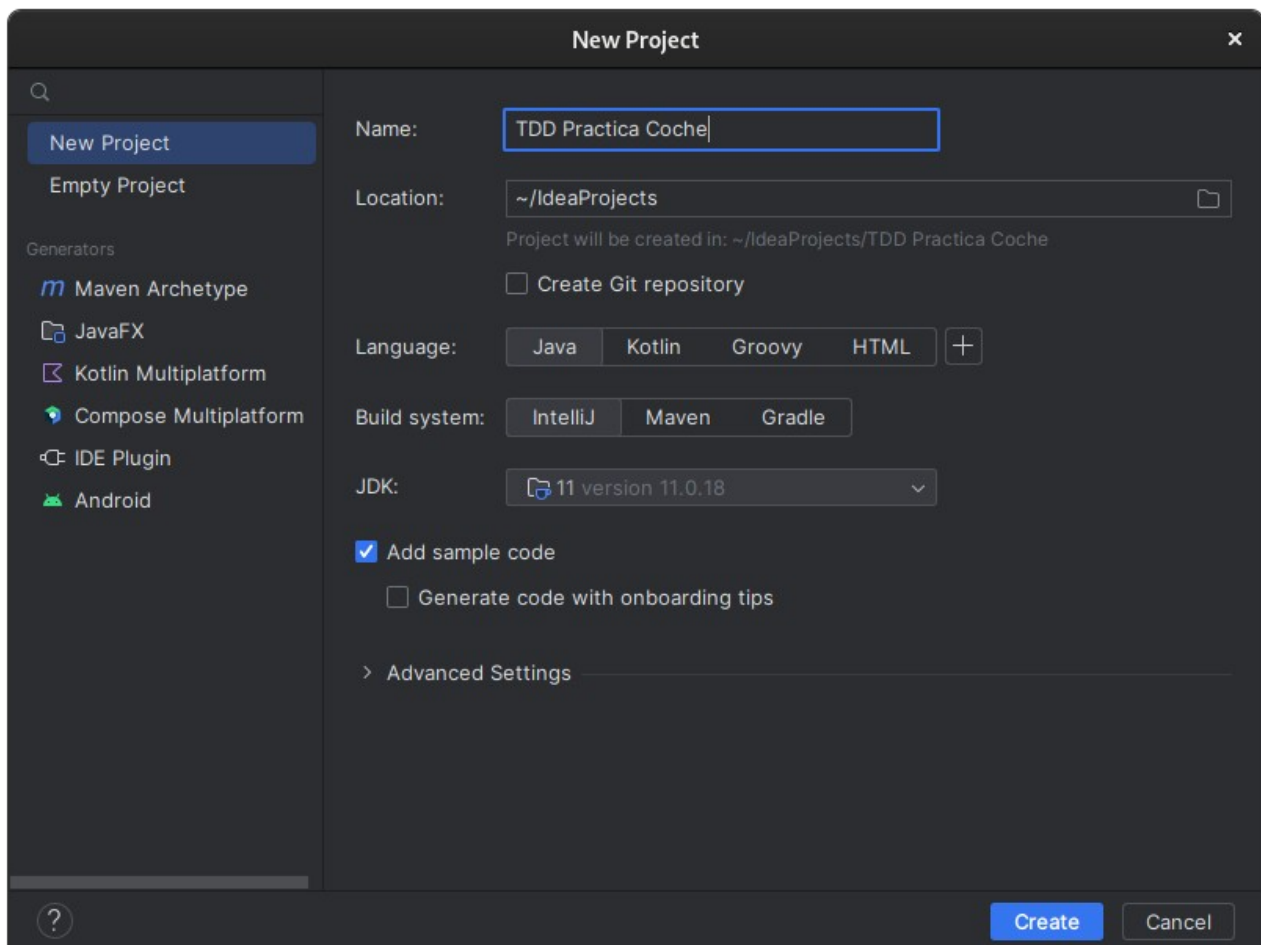
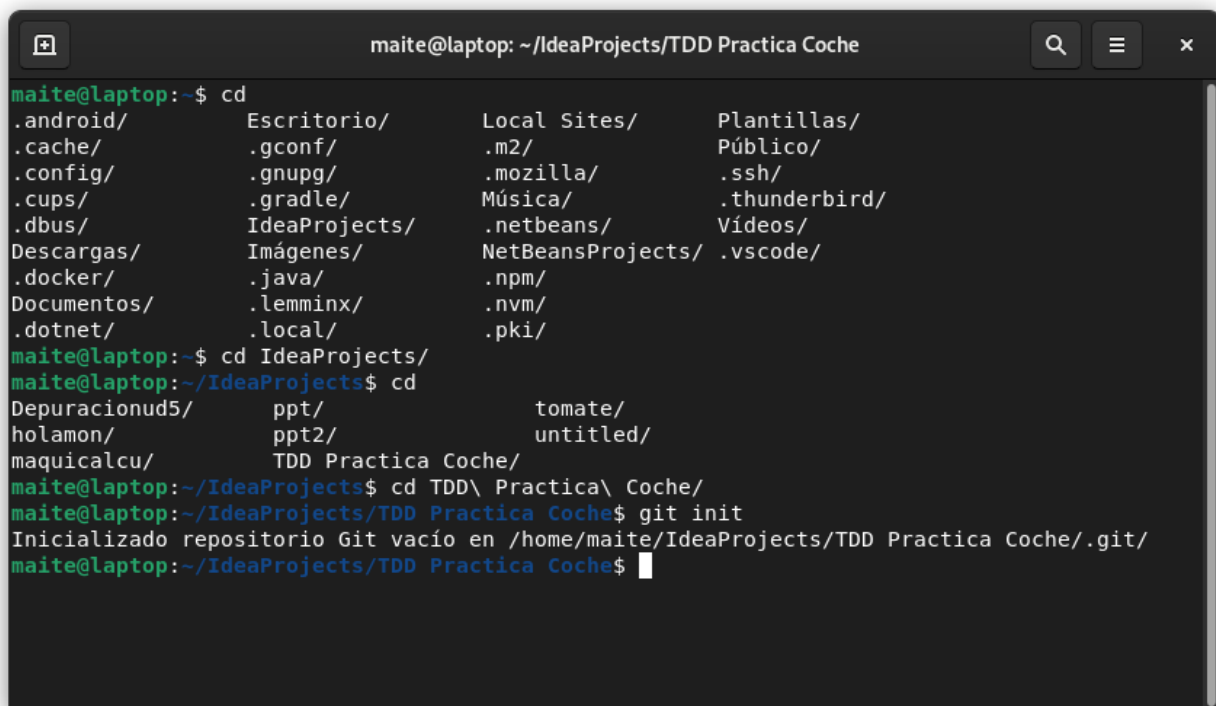


Figura 2: Creamos nuestro proyecto Java en el IDE IntelliJ

Una vez que tenemos todo puesto, seleccionamos “Create”.

## 3. Inicializamos el repositorio de Git

Vamos a inicializar el repositorio de git, mediante la terminal, utilizando el comando `<git init>`.



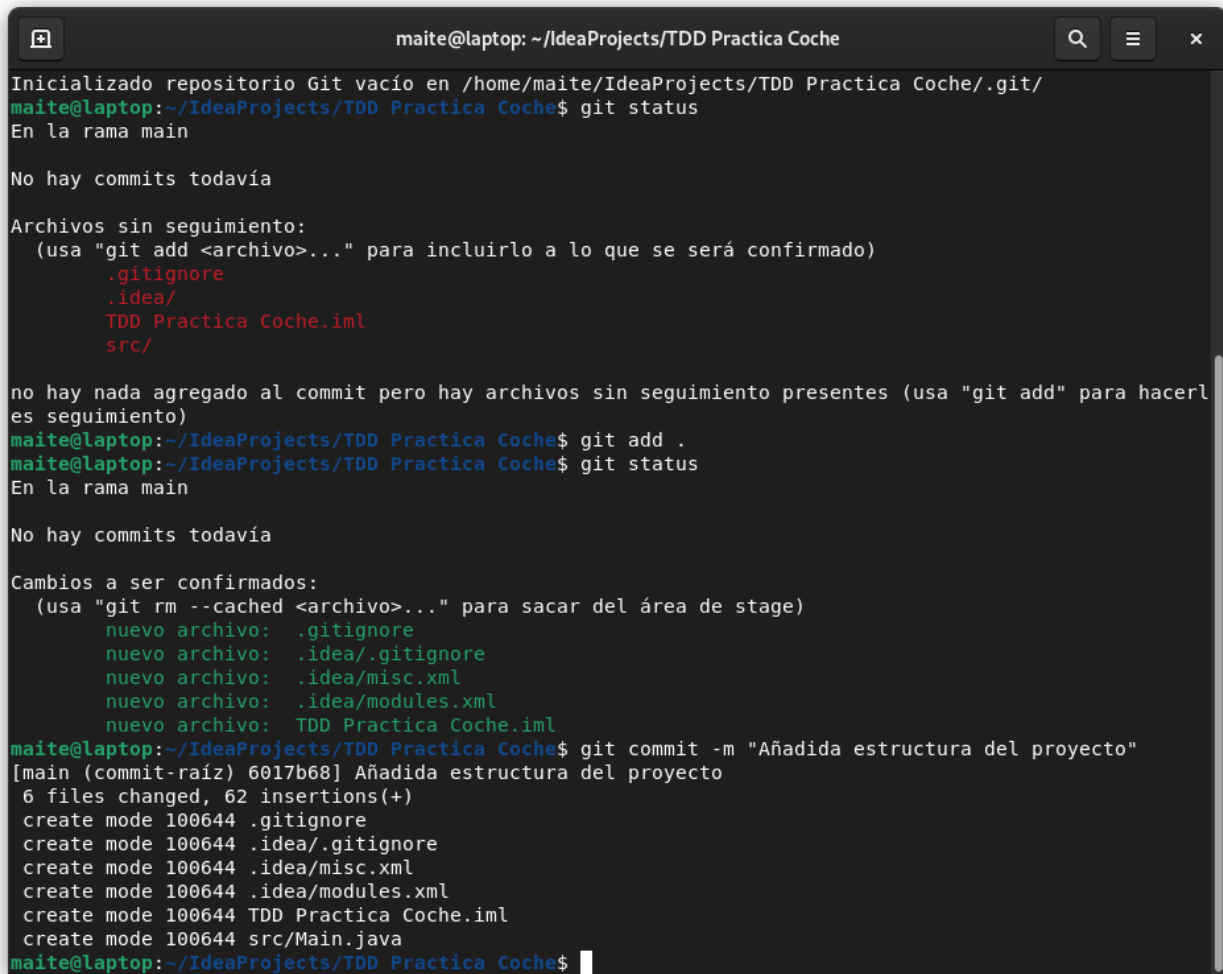
```
maite@laptop: ~/IdeaProjects/TDD Practica Coche
maite@laptop:~$ cd
. android/      Escritorio/      Local Sites/      Plantillas/
. cache/        .gconf/          .m2/              Público/
. config/       .gnupg/          .mozilla/         .ssh/
. cups/         .gradle/         Música/           .thunderbird/
. dbus/         IdeaProjects/    .netbeans/        Vídeos/
Descargas/      Imágenes/        NetBeansProjects/ .vscode/
. docker/       .java/           .npm/
Documentos/     .lemminx/        .nvm/
. dotnet/       .local/          .pki/
maite@laptop:~$ cd IdeaProjects/
maite@laptop:~/IdeaProjects$ cd
Depuracionud5/  ppt/             tomate/
holamon/       ppt2/            untitled/
maquicalcu/    TDD Practica Coche/
maite@laptop:~/IdeaProjects$ cd TDD\ Practica\ Coche/
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git init
Iniciado repositorio Git vacío en /home/maite/IdeaProjects/TDD Practica Coche/.git/
maite@laptop:~/IdeaProjects/TDD Practica Coche$
```

Figura 3: Inicializamos el proyecto usando el comando `<git init>`

Nos posicionamos en la carpeta donde tenemos el proyecto y una vez ahí ejecutamos el comando `<git init>`.

### 3. Primer commit del proyecto

Una vez hecho esto, haremos nuestro primer commit, donde empezaremos a estructurar el proyecto.

A terminal window titled 'maite@laptop: ~/IdeaProjects/TDD Practica Coche' showing the execution of git commands. The output indicates that the repository is initialized, files are tracked, and a commit is created with the message 'Añadida estructura del proyecto'.

```
maite@laptop: ~/IdeaProjects/TDD Practica Coche
Inicializado repositorio Git vacío en /home/maite/IdeaProjects/TDD Practica Coche/.git/
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git status
En la rama main

No hay commits todavía

Archivos sin seguimiento:
  (usa "git add <archivo>..." para incluirlo a lo que se será confirmado)
      .gitignore
      .idea/
      TDD Practica Coche.iml
      src/

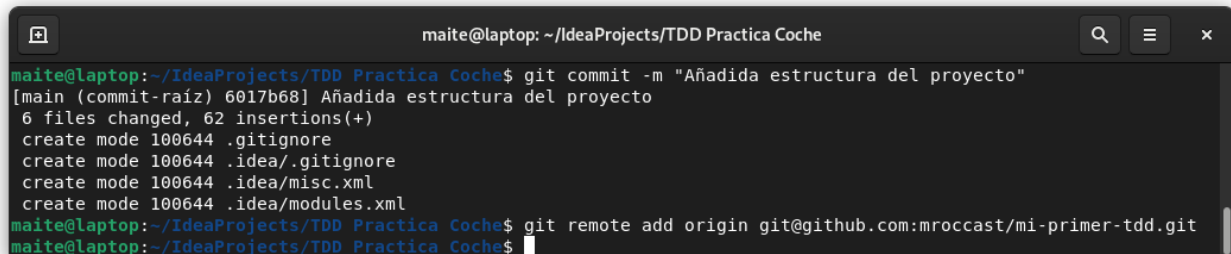
no hay nada agregado al commit pero hay archivos sin seguimiento presentes (usa "git add" para hacerle seguimiento)
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git add .
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git status
En la rama main

No hay commits todavía

Cambios a ser confirmados:
  (usa "git rm --cached <archivo>..." para sacar del área de stage)
      nuevo archivo: .gitignore
      nuevo archivo: .idea/.gitignore
      nuevo archivo: .idea/misc.xml
      nuevo archivo: .idea/modules.xml
      nuevo archivo: TDD Practica Coche.iml
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git commit -m "Añadida estructura del proyecto"
[main (commit-raíz) 6017b68] Añadida estructura del proyecto
6 files changed, 62 insertions(+)
create mode 100644 .gitignore
create mode 100644 .idea/.gitignore
create mode 100644 .idea/misc.xml
create mode 100644 .idea/modules.xml
create mode 100644 TDD Practica Coche.iml
create mode 100644 src/Main.java
maite@laptop:~/IdeaProjects/TDD Practica Coche$
```

Figura 4: Primer commit del proyecto con todos los comandos: `<git status>`, `<git add>` y `<git commit>`

Haremos nuestro primer commit el cual llamaremos “Añadida estructura del proyecto”, y usaremos el comando `<git remote>` para añadir la sincronización con GitHub.

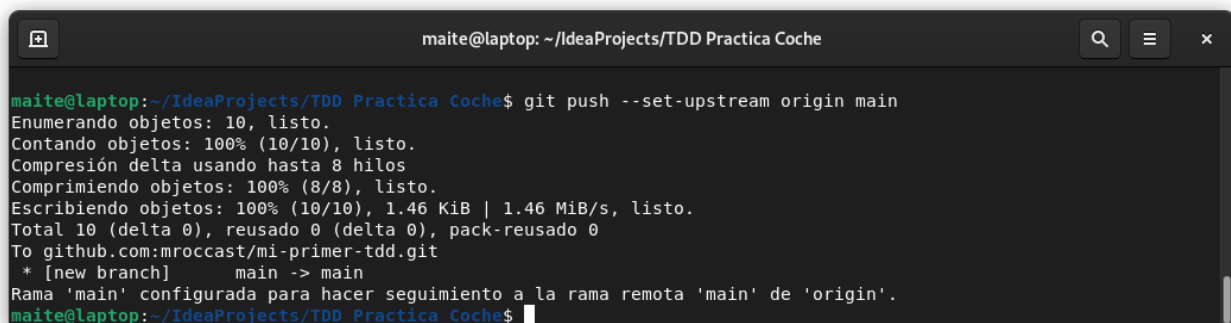


```
maite@laptop: ~/IdeaProjects/TDD Practica Coche
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git commit -m "Añadida estructura del proyecto"
[main (commit-raíz) 6017b68] Añadida estructura del proyecto
 6 files changed, 62 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 .idea/.gitignore
 create mode 100644 .idea/misc.xml
 create mode 100644 .idea/modules.xml
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git remote add origin git@github.com:mroccast/mi-primer-tdd.git
maite@laptop:~/IdeaProjects/TDD Practica Coche$
```

Figura 5: Ejecución del comando `<git remote>`

Usaremos el enlace que nos facilita GitHub, con el nombre de nuestro repositorio y lo ejecutamos.

Ahora usaremos el comando `<git push>` para sincronizar nuestros cambios locales al remoto, lo iremos haciendo en cada paso para evitar la pérdida de trabajo.



```
maite@laptop: ~/IdeaProjects/TDD Practica Coche
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git push --set-upstream origin main
Enumerando objetos: 10, listo.
Contando objetos: 100% (10/10), listo.
Compresión delta usando hasta 8 hilos
Comprimiendo objetos: 100% (8/8), listo.
Escribiendo objetos: 100% (10/10), 1.46 KiB | 1.46 MiB/s, listo.
Total 10 (delta 0), reusado 0 (delta 0), pack-reusado 0
To github.com:mroccast/mi-primer-tdd.git
 * [new branch]      main -> main
Rama 'main' configurada para hacer seguimiento a la rama remota 'main' de 'origin'.
maite@laptop:~/IdeaProjects/TDD Practica Coche$
```

Figura 6: Ejecución del comando `<git push>`

## 4. Creación del directorio

A continuación vamos a crear un directorio para los tests. Vamos al IntelliJ, en la columna de la izquierda donde aparece el nombre de nuestro proyecto hacemos click con el botón derecho y seleccionamos las siguientes opciones: **New > Directory**

Llamaremos a nuestro directorio **“tests”**

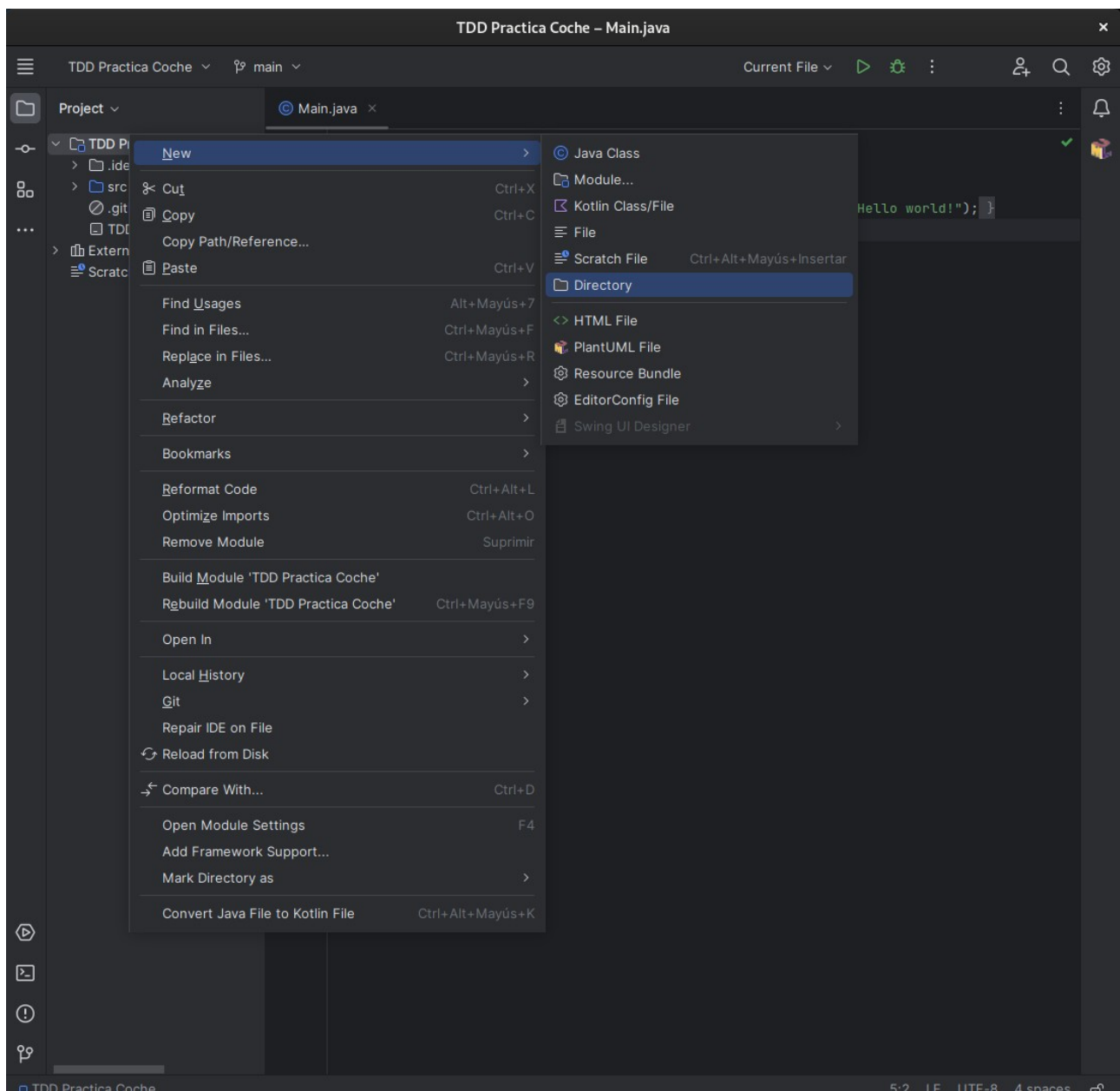


Figura 7: Primer paso, crear un directorio para los tests



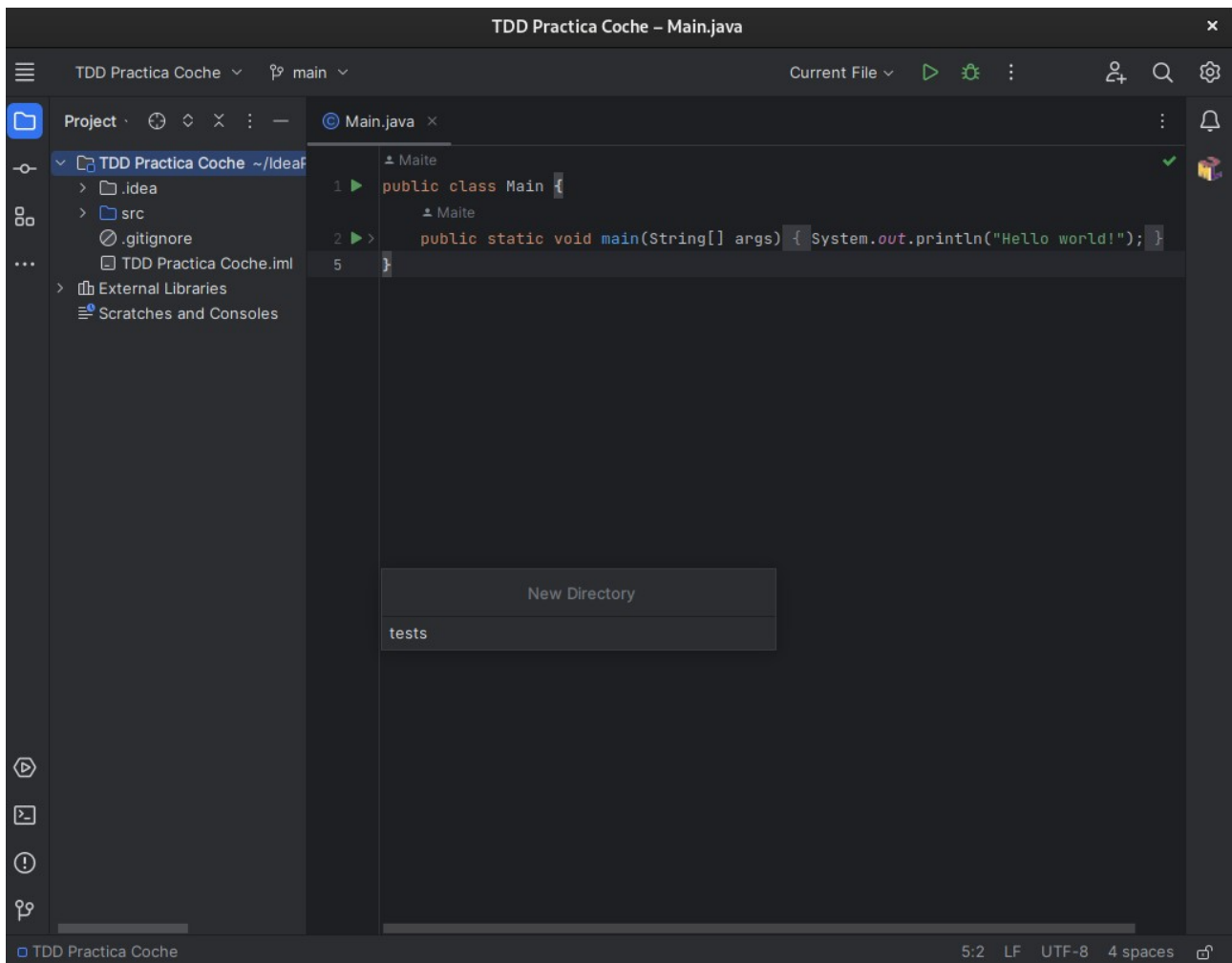


Figura 8: Segundo paso, nombrar el directorio para los tests

Y marcaremos el directorio como directorio de tests, hacemos click con el botón derecho sobre “tests” y seguimos las siguientes opciones: **Mark Directory As > Test Sources Root**

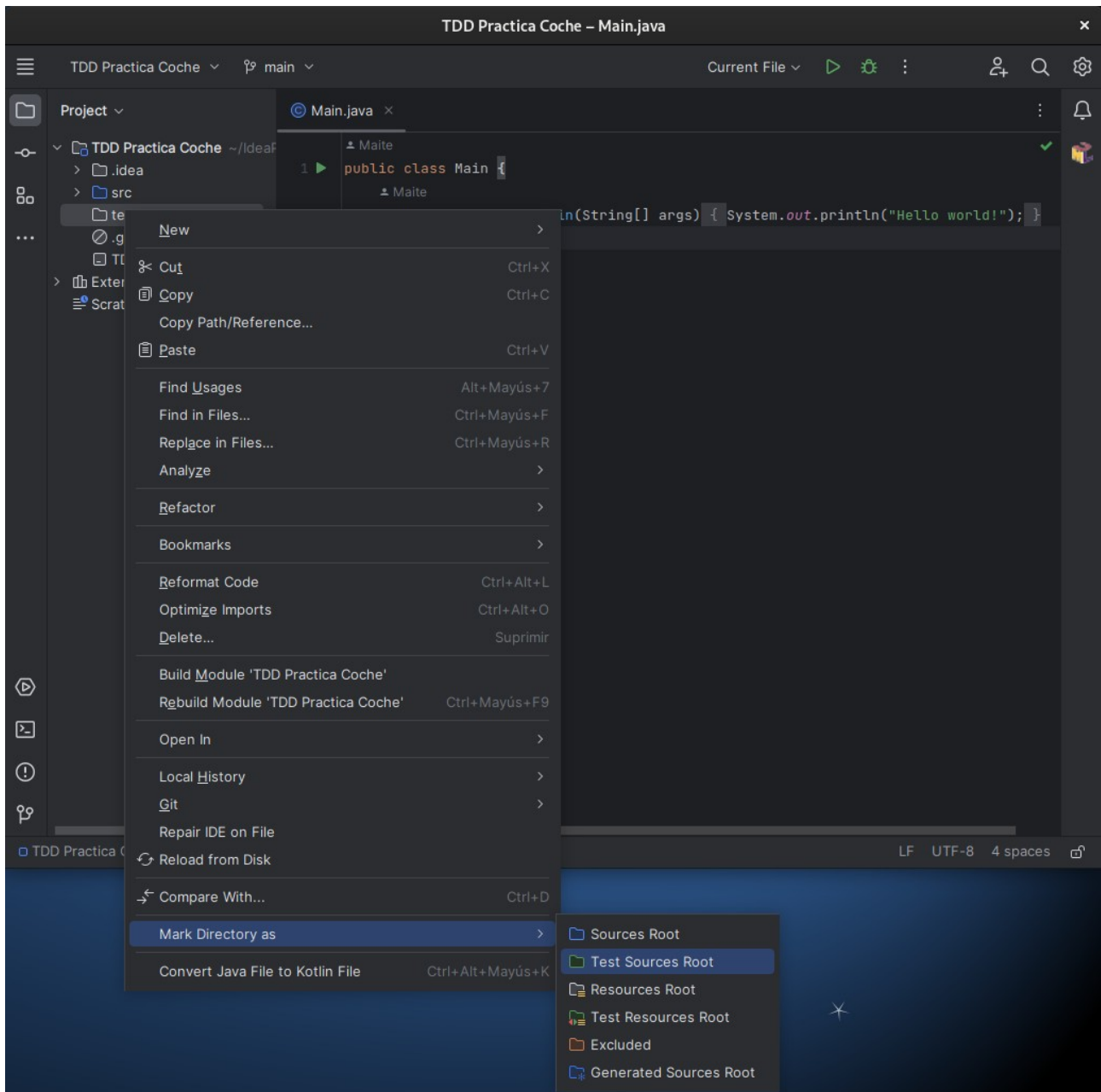
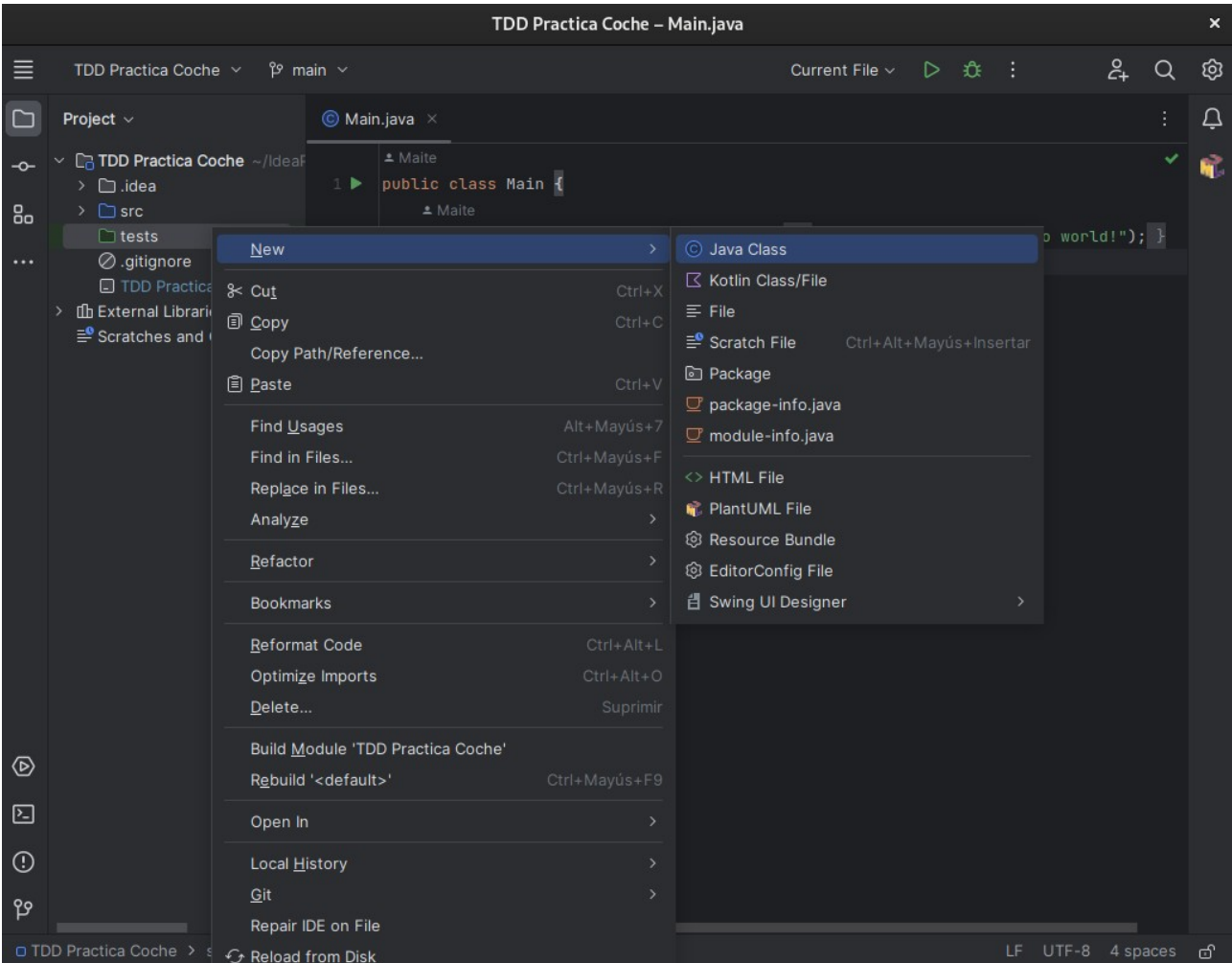


Figura 9: Tercer paso, marcamos nuestro directorio "tests" como directorio de tests

## 5. Creación de la clase

Vamos a crear dentro del directorio "tests" una clase Java.

Hacemos click derecho sobre nuestro directorio "tests": **tests > New > Java Class**



*Figura 10: Creamos dentro del directorio tests una clase Java*

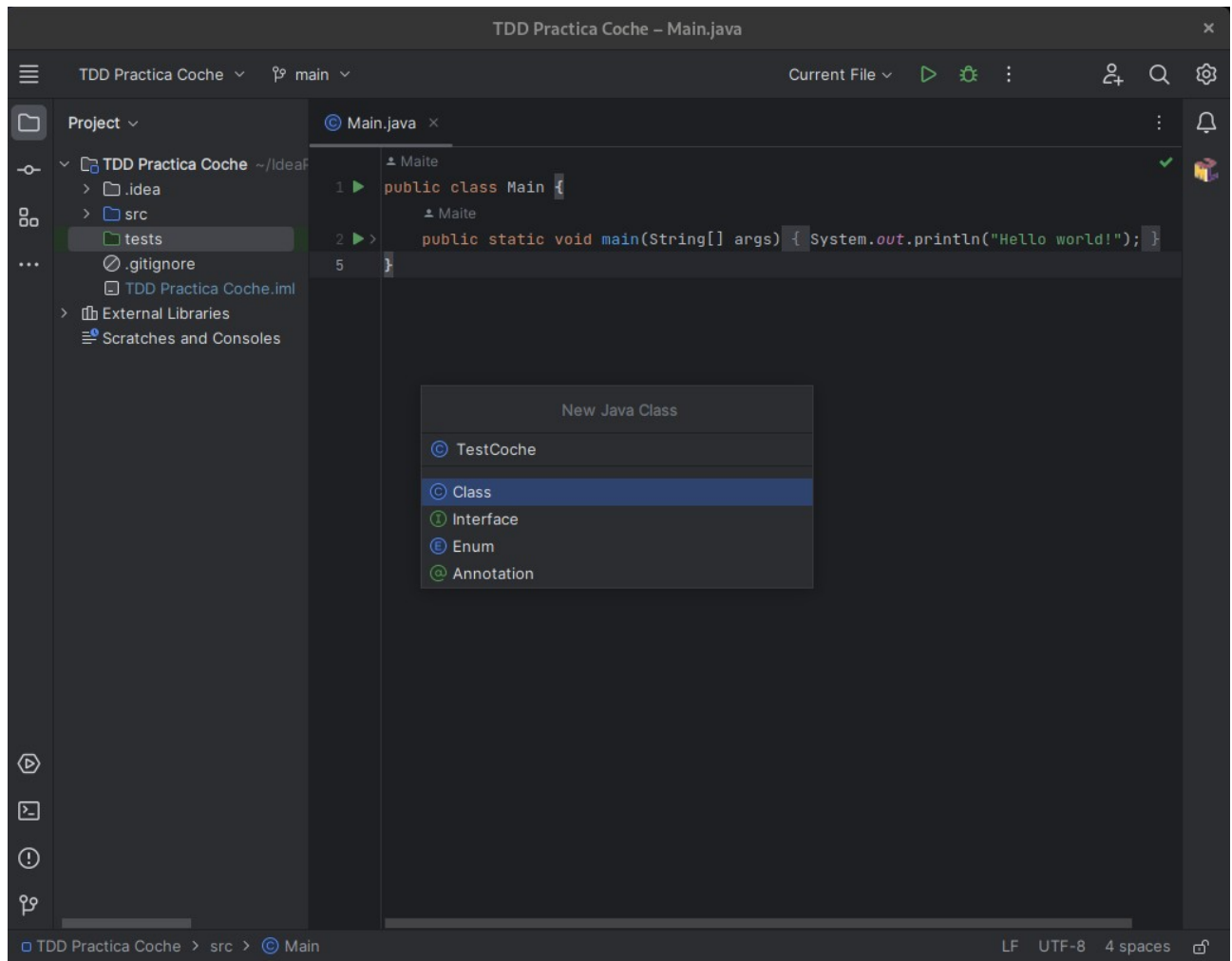


Figura 11: Nombramos la clase Java "TestCoche"

## 6. Creación del test

Escribiremos dentro de nuestra clase **TestCoche**, **@Test** para indicar que se trata de un test, y nos aparecerá un aviso para añadir la librería JUnit5 y lo seleccionamos.

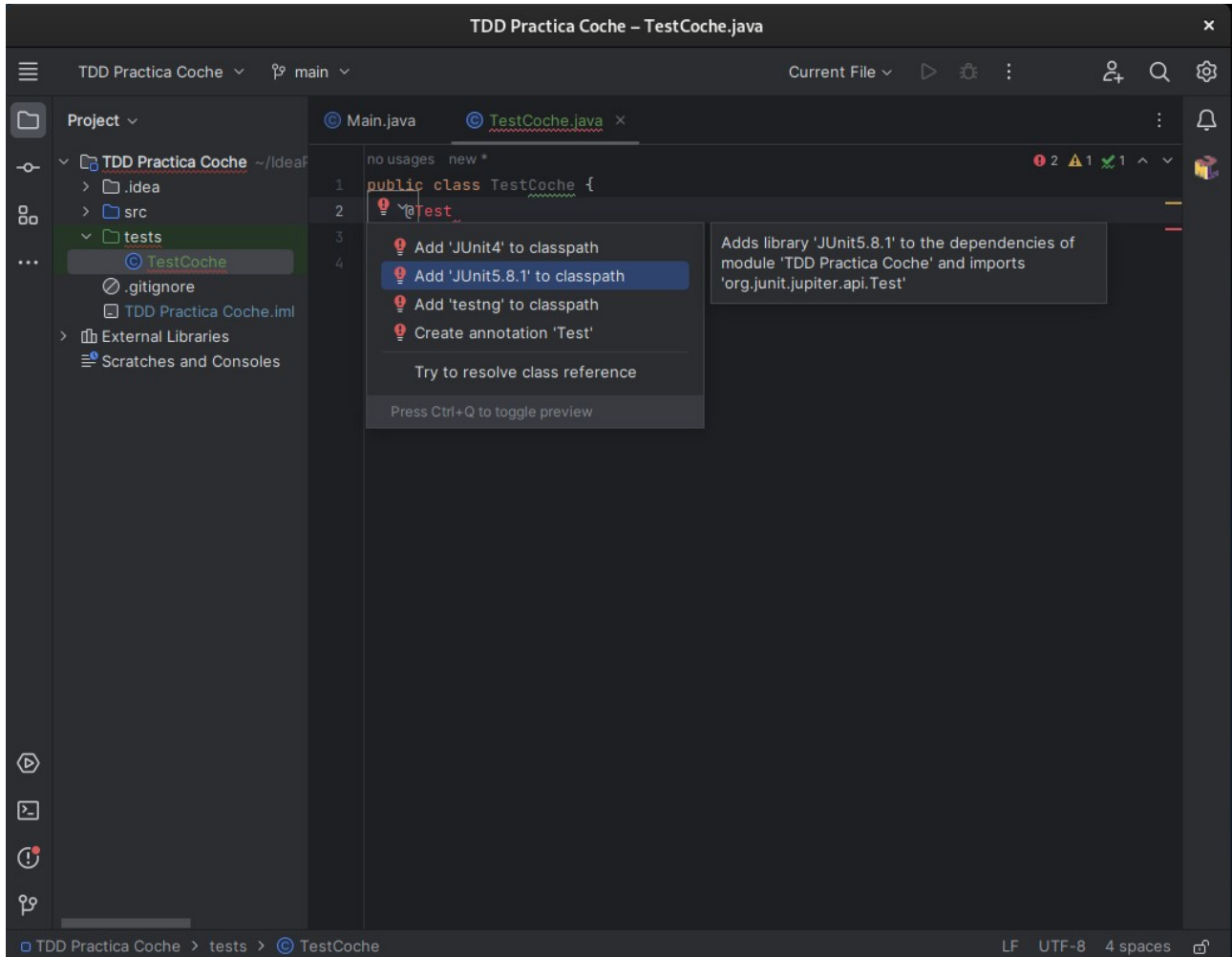


Figura 12: Aviso para indicar que se use la libreria JUnit5

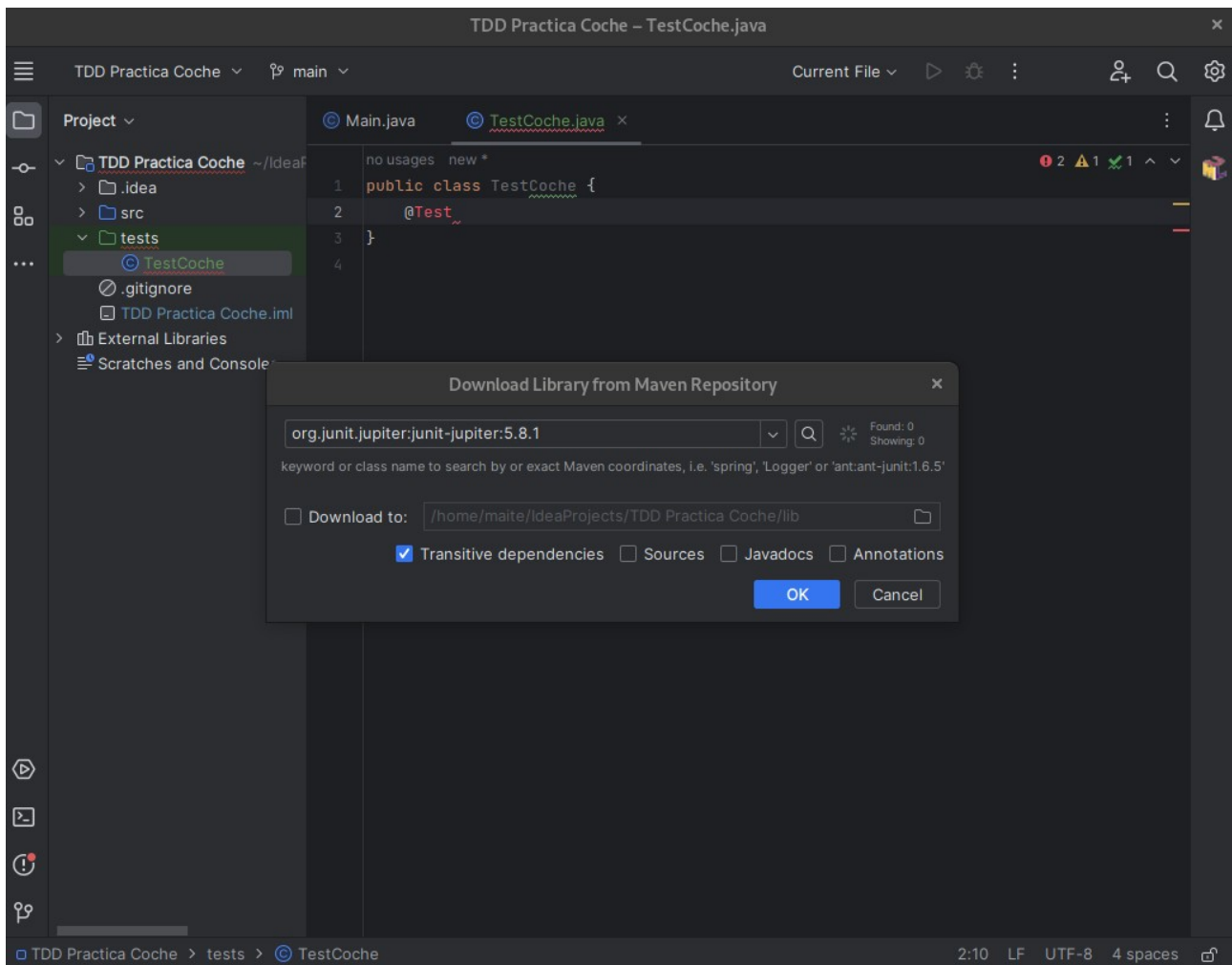


Figura 13: Confirmamos el uso de la librería `jUnit5`

Una vez hecho esto comenzamos a crear el test, al que llamaremos **`test_crear_coche`** por ejemplo, y ejecutamos el test.

Nos va a dar error porque no existe la clase `coche` y por tanto no compila.

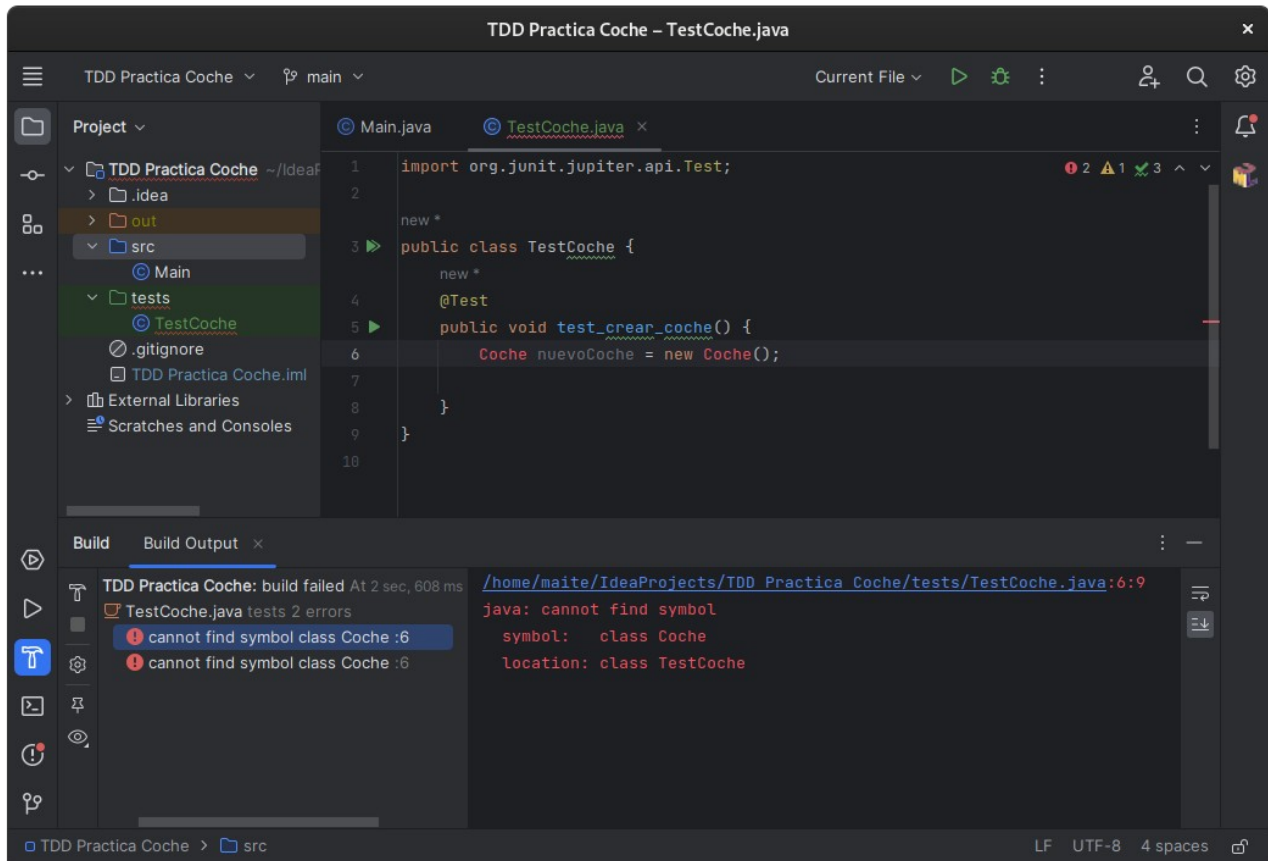
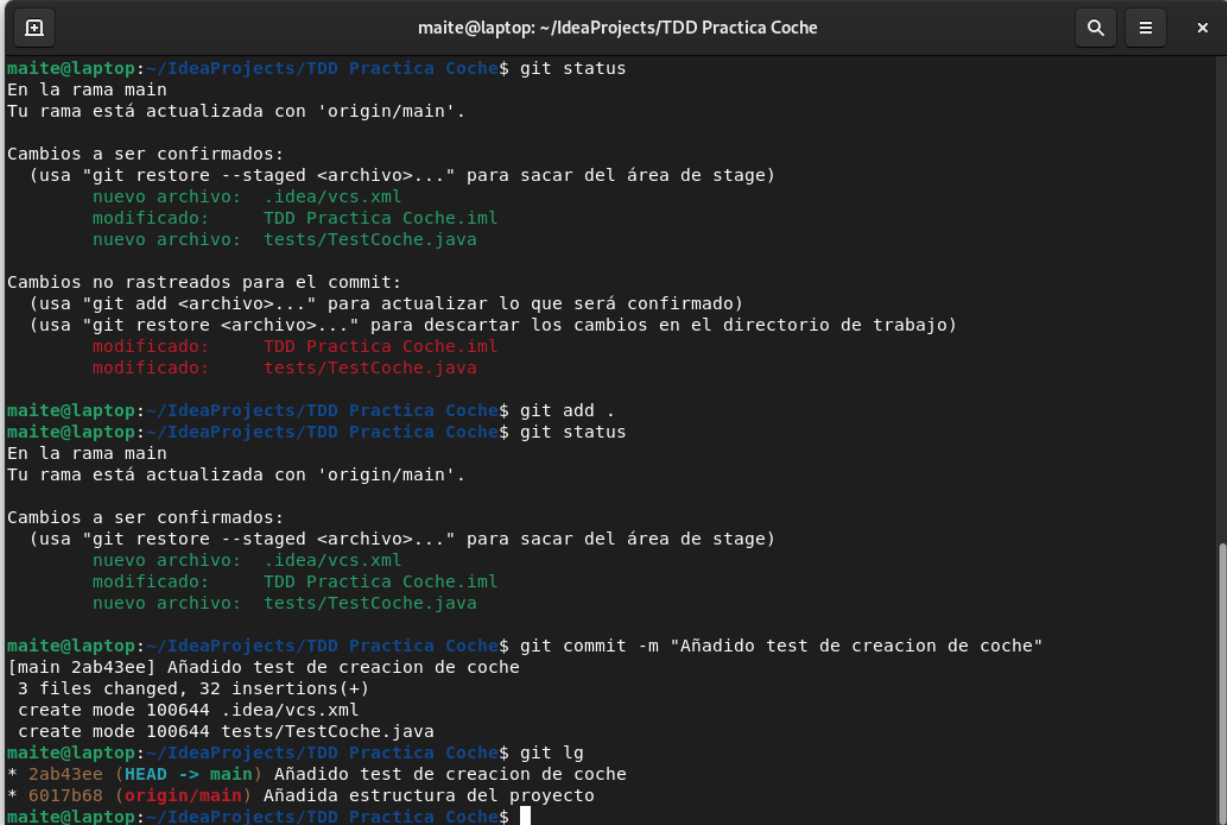


Figura 14: Error, el test no compila porque falta la clase Coche

## 7. Commit del test

A partir de aquí iremos paso a paso.

Vamos a hacer el commit de los cambios que tenemos en nuestro proyecto. Acabamos de crear nuestra clase Java, que da error y utilizaremos los comandos **<git status>**, **<git add>** y **<git commit>** para asegurarnos de commitear los cambios.

A terminal window titled 'maite@laptop: ~/IdeaProjects/TDD Practica Coche' showing the execution of git commands. The user runs 'git status' twice, which shows changes to '.idea/vcs.xml', 'TDD Practica Coche.iml', and 'tests/TestCoche.java'. The user then runs 'git add .' and 'git commit -m "Añadido test de creacion de coche"', which successfully creates a new commit on the 'main' branch.

```
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git status
En la rama main
Tu rama está actualizada con 'origin/main'.

Cambios a ser confirmados:
  (usa "git restore --staged <archivo>..." para sacar del área de stage)
    nuevo archivo:  .idea/vcs.xml
    modificado:     TDD Practica Coche.iml
    nuevo archivo:  tests/TestCoche.java

Cambios no rastreados para el commit:
  (usa "git add <archivo>..." para actualizar lo que será confirmado)
  (usa "git restore <archivo>..." para descartar los cambios en el directorio de trabajo)
    modificado:     TDD Practica Coche.iml
    modificado:     tests/TestCoche.java

maite@laptop:~/IdeaProjects/TDD Practica Coche$ git add .
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git status
En la rama main
Tu rama está actualizada con 'origin/main'.

Cambios a ser confirmados:
  (usa "git restore --staged <archivo>..." para sacar del área de stage)
    nuevo archivo:  .idea/vcs.xml
    modificado:     TDD Practica Coche.iml
    nuevo archivo:  tests/TestCoche.java

maite@laptop:~/IdeaProjects/TDD Practica Coche$ git commit -m "Añadido test de creacion de coche"
[main 2ab43ee] Añadido test de creacion de coche
3 files changed, 32 insertions(+)
create mode 100644 .idea/vcs.xml
create mode 100644 tests/TestCoche.java
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git lg
* 2ab43ee (HEAD -> main) Añadido test de creacion de coche
* 6017b68 (origin/main) Añadida estructura del proyecto
maite@laptop:~/IdeaProjects/TDD Practica Coche$
```

Figura 15: Ejecución del comando `<git commit>` y estado actual de nuestro proyecto



## 8. Creación de la clase Coche

Vamos a crear la clase Coche para que nuestro test funcione. Hacemos click con el boton derecho y seleccionamos: **New > Java Class**

Y la nombraremos “**Coche**”

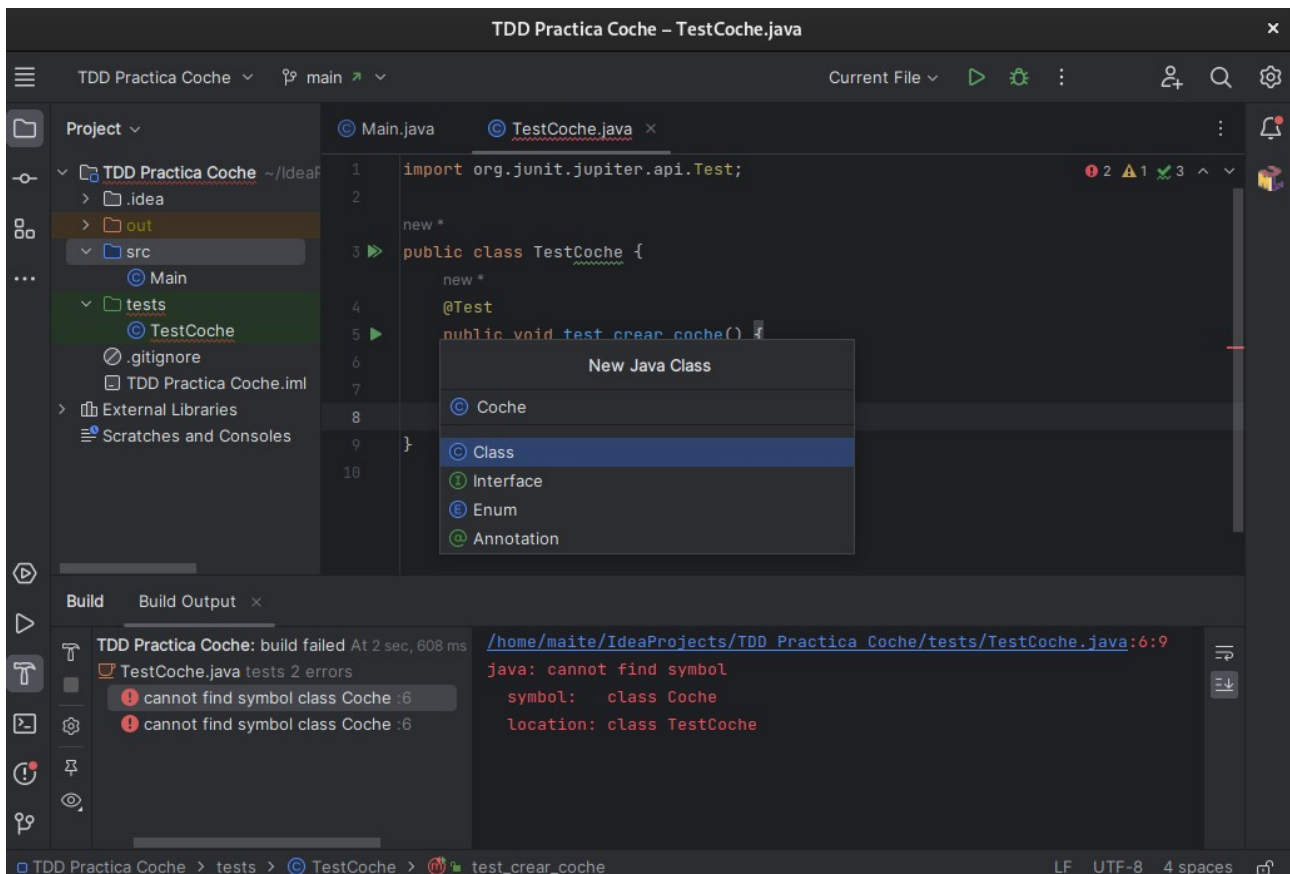


Figura 16: Creación de la clase Coche

Una vez creada la clase coche volveremos a ejecutar nuestro test.

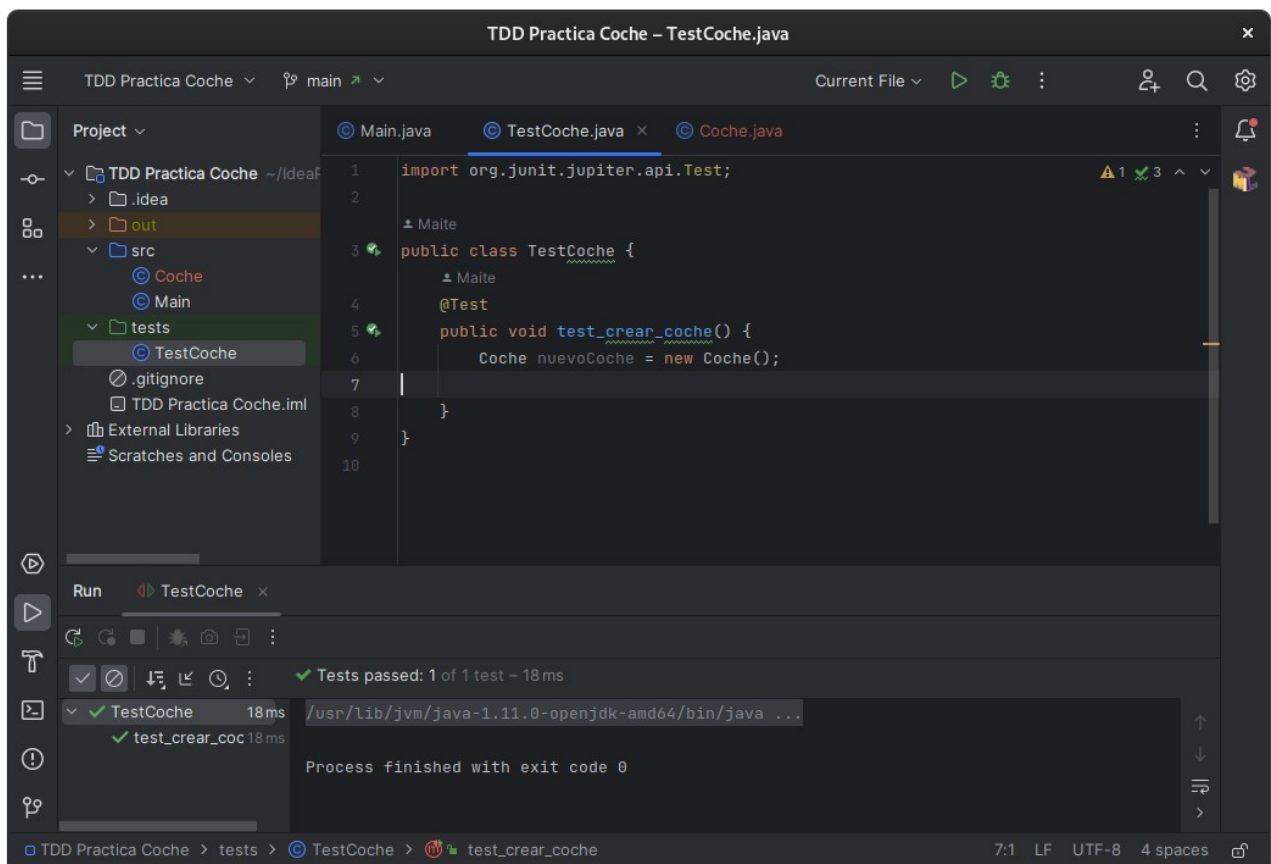


Figura 17: Ejecución del test correcta

Una vez que la ejecución del test es correcta, hacemos un commit “Añadida clase Coche”

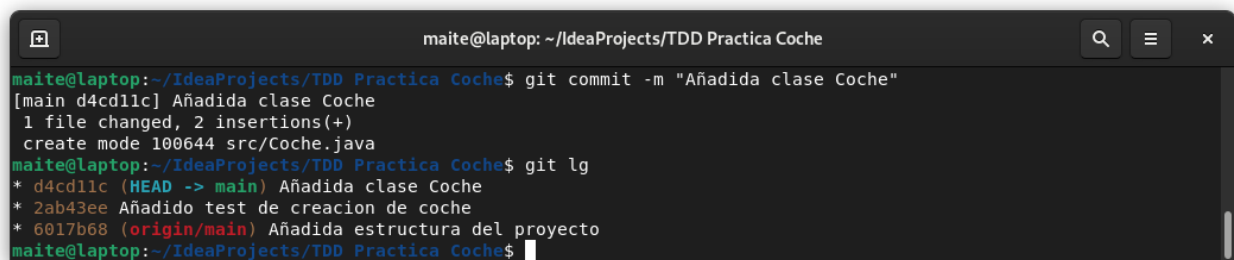


Figura 18: Commit de la clase Coche añadida

## 9. Mejoramos el test

Vamos a mejorar el test.

Comenzamos por cambiar el nombre del test y lo llamaremos

**“test\_al\_crear\_un\_coche\_su\_velocidad\_es\_cero”** y utilizaremos una aserción y vamos a afirmar que la velocidad del coche será igual a **0** y el valor actual será **velocidad**.

Lo ejecutamos y nos dará un error porque dentro de la clase coche no existe la variable velocidad.

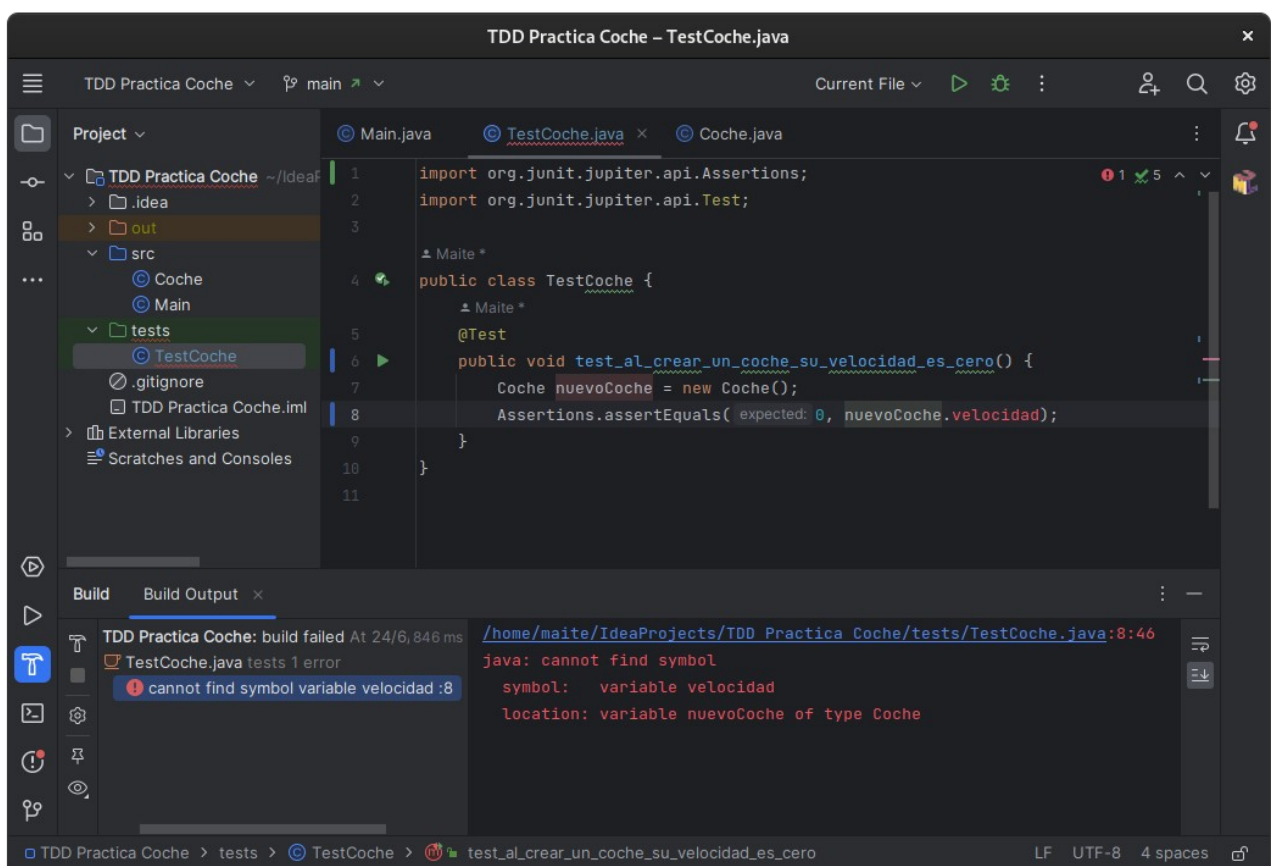
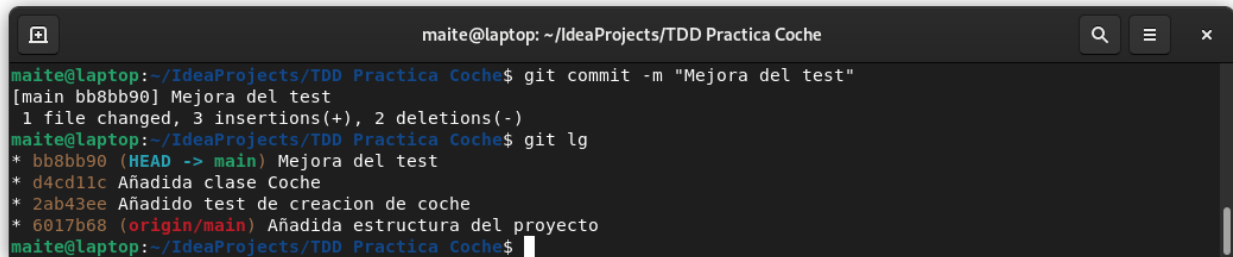


Figura 19: Error de la variable, no existe dentro de la clase.

Hacemos un commit con la mejora del test.



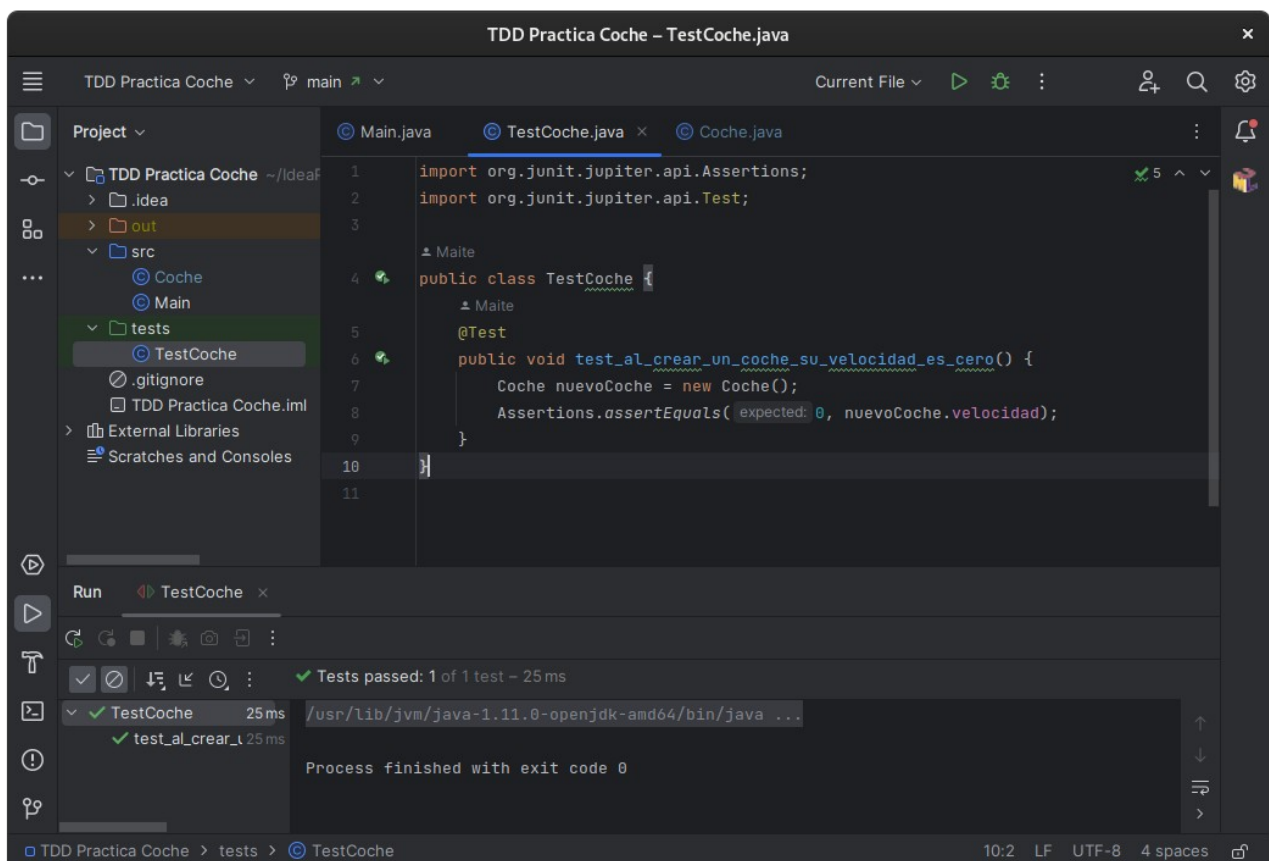
```
maite@laptop: ~/IdeaProjects/TDD Practica Coche
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git commit -m "Mejora del test"
[main bb8bb90] Mejora del test
1 file changed, 3 insertions(+), 2 deletions(-)
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git lg
* bb8bb90 (HEAD -> main) Mejora del test
* d4cd11c Añadida clase Coche
* 2ab43ee Añadido test de creacion de coche
* 6017b68 (origin/main) Añadida estructura del proyecto
maite@laptop:~/IdeaProjects/TDD Practica Coche$
```

Figura 20: Commit con la mejora del test realizado

## 10. Error del atributo velocidad

Ahora vamos a solucionar el error del atributo velocidad, y lo añadiremos dentro de la clase Coche.

Una vez hecho esto, ejecutaremos el test y ahora si podremos comprobar que funciona correctamente.



```
TDD Practica Coche - TestCoche.java
import org.junit.jupiter.api.Assertions;
import org.junit.jupiter.api.Test;

public class TestCoche {
    @Test
    public void test_al_crear_un_coche_su_velocidad_es_cero() {
        Coche nuevoCoche = new Coche();
        Assertions.assertEquals( expected: 0, nuevoCoche.velocidad);
    }
}
```

Run TestCoche

Tests passed: 1 of 1 test - 25 ms

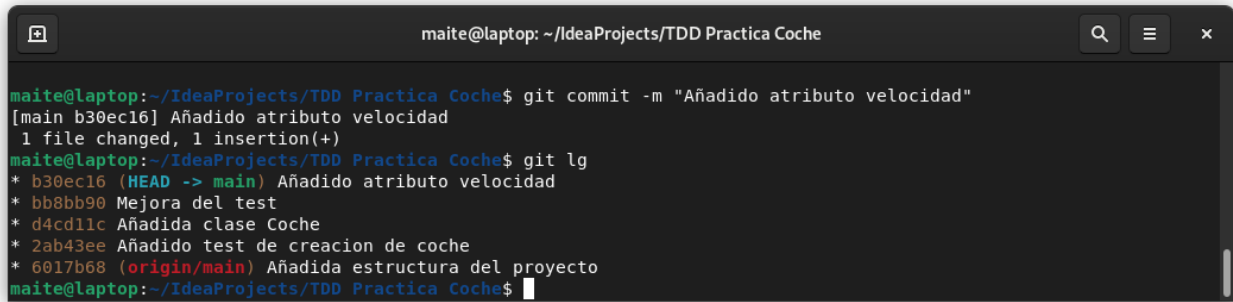
TestCoche 25 ms /usr/lib/jvm/java-1.11.0-openjdk-amd64/bin/java ...

test\_al\_crear\_u 25 ms

Process finished with exit code 0

Figura 21: Ejecución correcta del test, después de añadir el atributo

Ahora haremos un commit donde declararemos que el atributo velocidad ha sido añadido a la clase Coche.

A terminal window titled 'maite@laptop: ~/IdeaProjects/TDD Practica Coche'. The terminal shows the following commands and output:

```
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git commit -m "Añadido atributo velocidad"
[main b30ec16] Añadido atributo velocidad
1 file changed, 1 insertion(+)
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git lg
* b30ec16 (HEAD -> main) Añadido atributo velocidad
* bb8bb90 Mejora del test
* d4cd11c Añadida clase Coche
* 2ab43ee Añadido test de creacion de coche
* 6017b68 (origin/main) Añadida estructura del proyecto
maite@laptop:~/IdeaProjects/TDD Practica Coche$
```

## 11. Nuevo test

A continuación vamos a añadir un nuevo test que llamaremos

***“test\_al\_acelerar\_un\_coche\_su\_velocidad\_aumenta”*** y añadiremos un nuevo atributo que será **acelerar**.

Lo vamos a ejecutar y nuevamente vemos que da error porque el atributo no se encuentra dentro de la clase Coche.

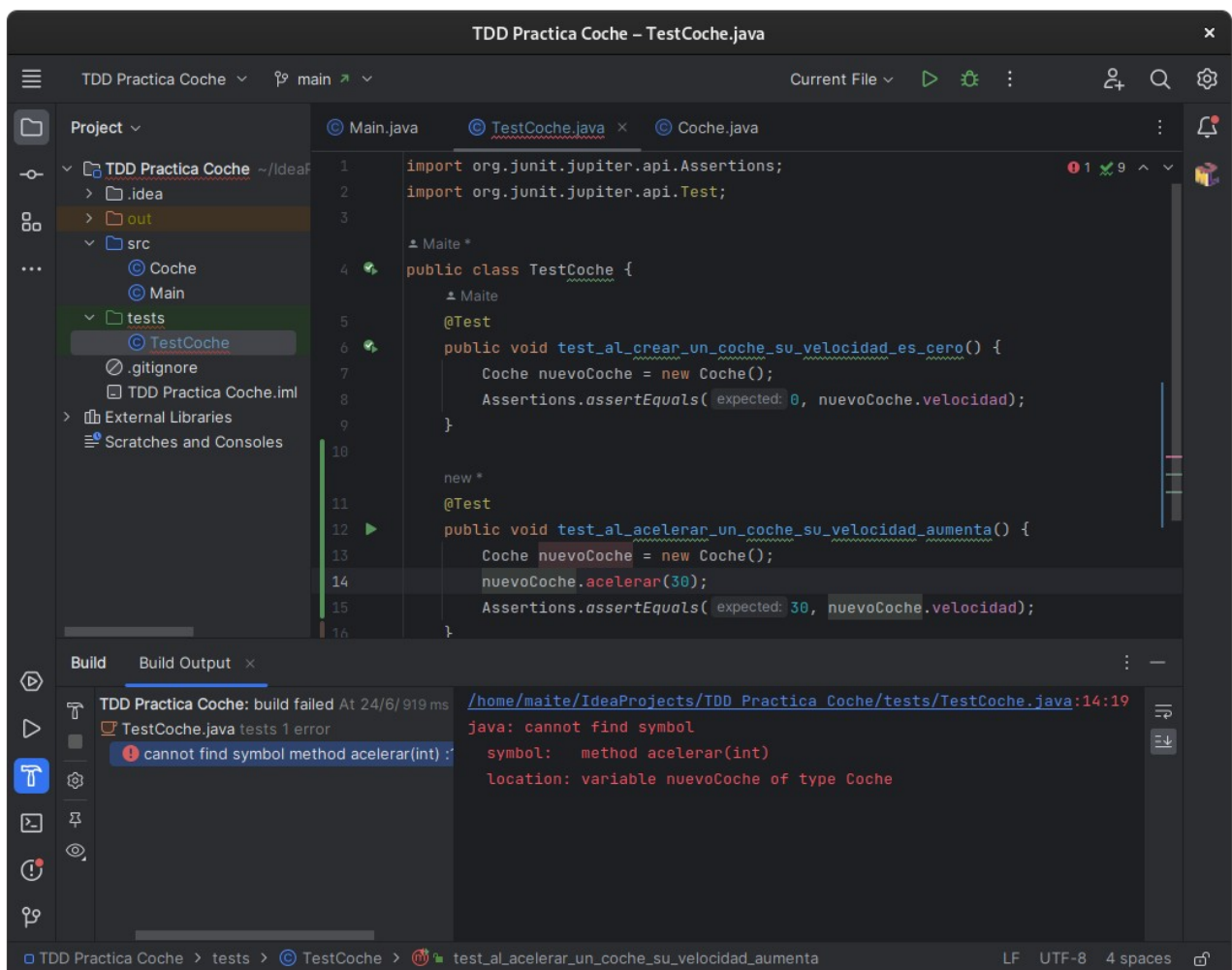
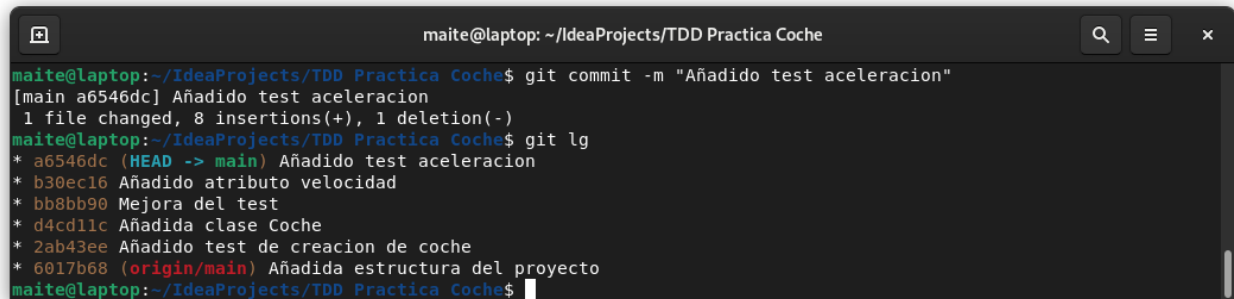


Figura 22: Fallo del test acelerar. Falta el atributo en la clase Coche.

Haremos un commit para guardar los cambios con el nuevo test.

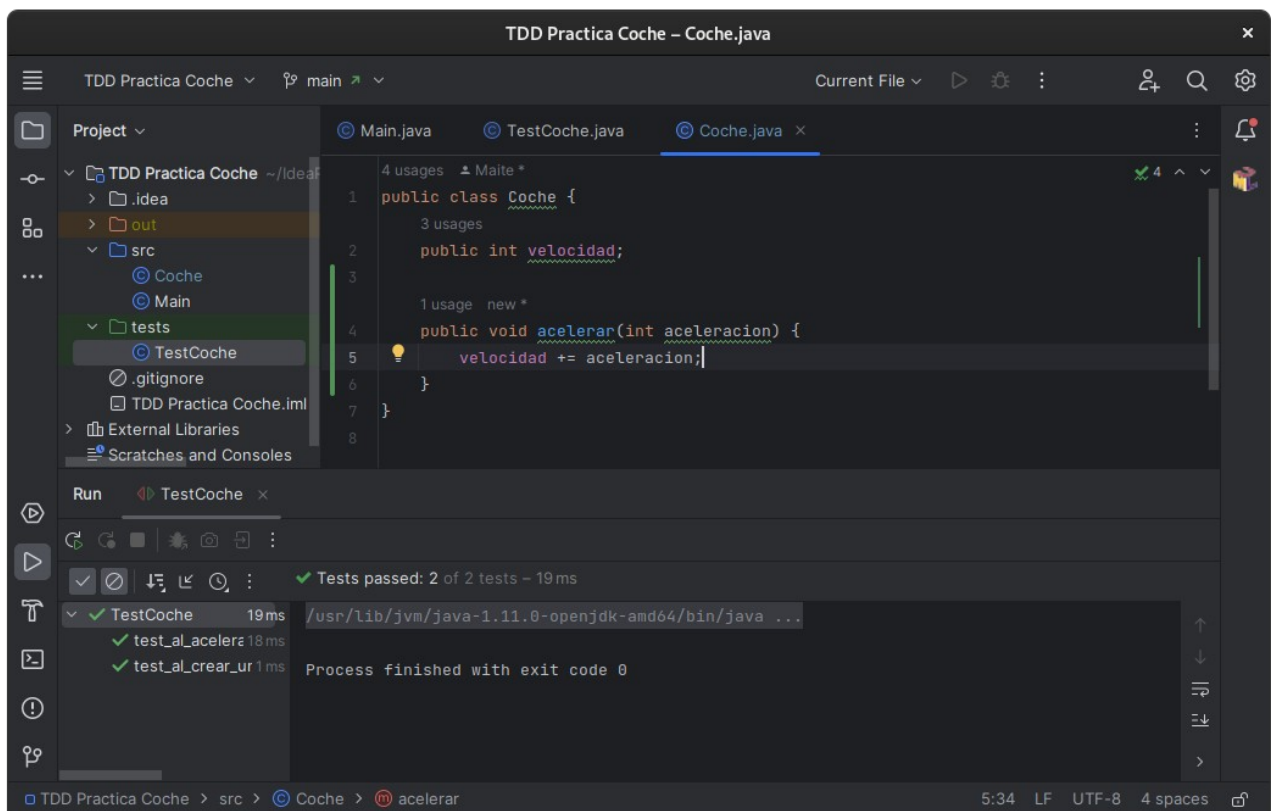


```
maite@laptop: ~/IdeaProjects/TDD Practica Coche
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git commit -m "Añadido test aceleracion"
[main a6546dc] Añadido test aceleracion
1 file changed, 8 insertions(+), 1 deletion(-)
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git lg
* a6546dc (HEAD -> main) Añadido test aceleracion
* b30ec16 Añadido atributo velocidad
* bb8bb90 Mejora del test
* d4cd11c Añadida clase Coche
* 2ab43ee Añadido test de creacion de coche
* 6017b68 (origin/main) Añadida estructura del proyecto
maite@laptop:~/IdeaProjects/TDD Practica Coche$
```

Figura 23: Commit con el nuevo test aceleración de coche

Añadimos el atributo acelerar a la clase Coche y volvemos a ejecutar el test.

Ahora si podemos ver que funciona correctamente y que ha pasado el test.



The screenshot shows the IntelliJ IDEA interface with the 'TDD Practica Coche' project. The 'Coche.java' file is open, showing the following code:

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

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

- Tests passed: 2 of 2 tests - 19ms
- test\_a\_acelera 18ms
- test\_a\_crear\_ur 1ms

The status bar at the bottom indicates the current file is 'Coche.java' and the method being executed is 'acelerar'.

Figura 24: Ejecución del test acelerar. Funciona correctamente.



## 11. Método decelerar

Vamos a crear ahora un método decelerar.

Crearemos un nuevo coche, con velocidad 50 y un atributo decelerar de 20.

Pero de nuevo el test nos da error porque no detecta el atributo decelerar que no se encuentra dentro de la clase coche.

De todos modos ejecutamos el test y vemos que da error.

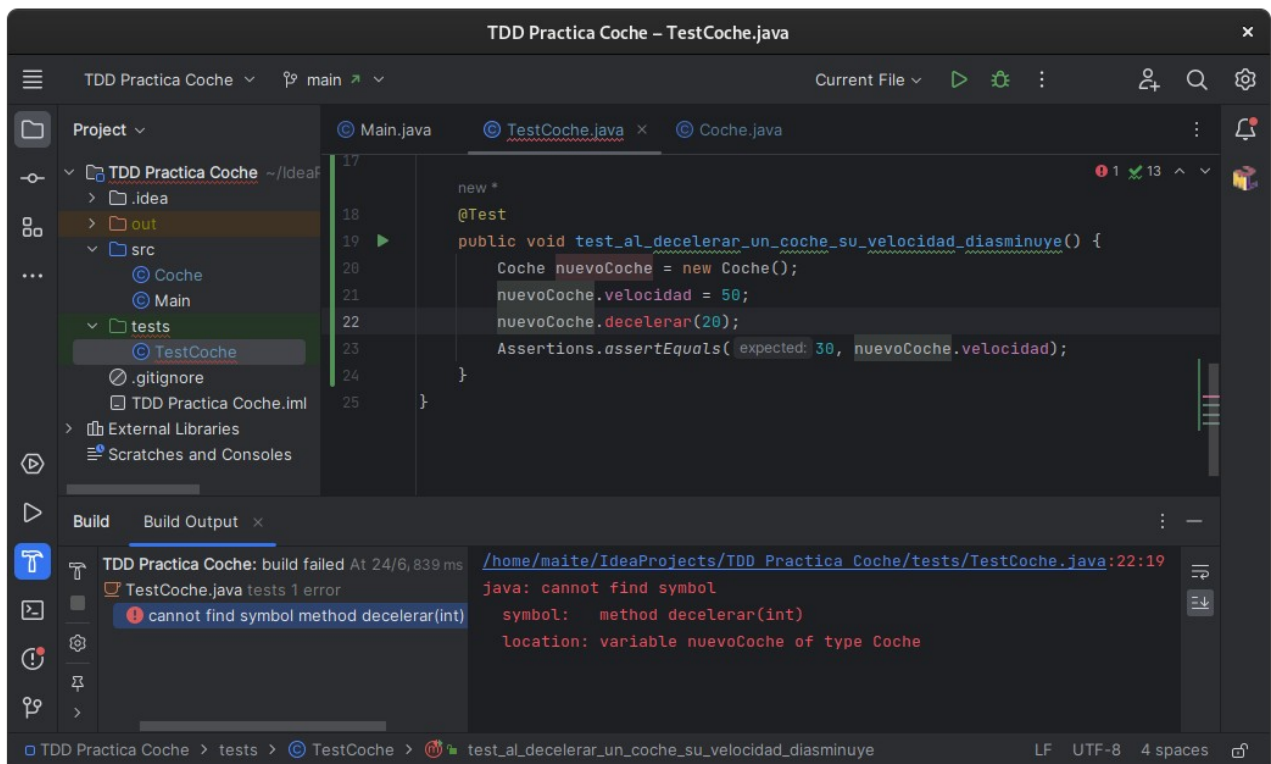
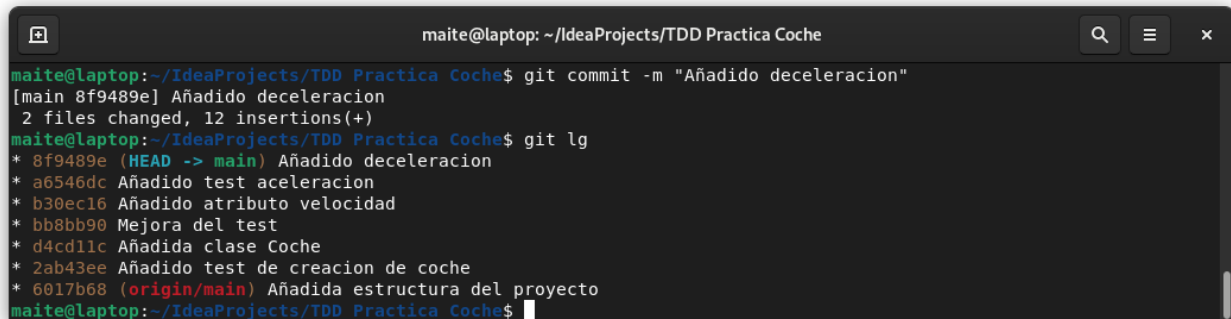


Figura 25: Añadido el test de deceleración y error de ejecución del test



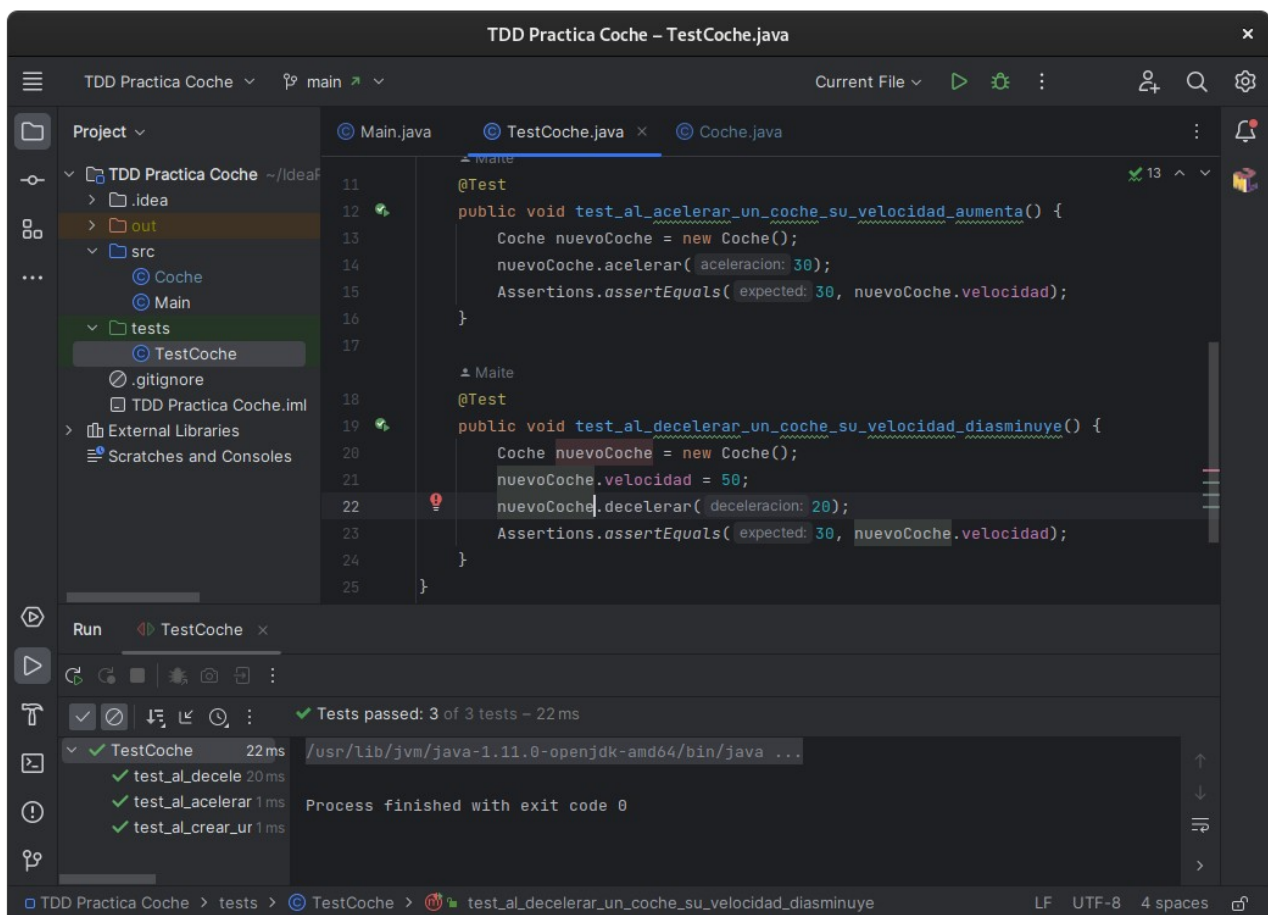
Hacemos el commit del test de deceleración.



```
maite@laptop: ~/IdeaProjects/TDD Practica Coche
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git commit -m "Añadido deceleracion"
[main 8f9489e] Añadido deceleracion
2 files changed, 12 insertions(+)
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git lg
* 8f9489e (HEAD -> main) Añadido deceleracion
* a6546dc Añadido test aceleracion
* b30ec16 Añadido atributo velocidad
* bb8bb90 Mejora del test
* d4cd11c Añadida clase Coche
* 2ab43ee Añadido test de creacion de coche
* 6017b68 (origin/main) Añadida estructura del proyecto
maite@laptop:~/IdeaProjects/TDD Practica Coche$
```

Figura 26: Commit del método de deceleración

Añadiremos el atributo deceleración dentro de la clase coche para que funcione el test.



```
TDD Practica Coche - TestCoche.java
Project: TDD Practica Coche
Main.java | TestCoche.java | Coche.java
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
18 Maite
19 @Test
20 public void test_al_decelerar_un_coche_su_velocidad_diasminuye() {
21     Coche nuevoCoche = new Coche();
22     nuevoCoche.velocidad = 50;
23     nuevoCoche.decelerar( deceleracion: 20);
24     Assertions.assertEquals( expected: 30, nuevoCoche.velocidad);
25 }

Run TestCoche
Tests passed: 3 of 3 tests - 22 ms
TestCoche 22ms
test_al_decele 20ms
test_al_acelerar 1ms
test_al_crear_ur 1ms
Process finished with exit code 0
```

Figura 27: Test de deceleración funciona correctamente.

## 12. Test deceleración no puede ser menor que cero.

Por último, vamos a crear un test de deceleración que no su valor no pueda ser menor a cero.

Vamos a crearlo y lo ejecutaremos, y esta vez vamos a ver que nos ha fallado un test.

Veremos que se esperaba un valor **0** pero el valor actual es **-30**, porque en nuestro test hemos puesto una deceleración de **80** y el valor de velocidad del coche es de **50**.

Lo veremos más claro en la captura:

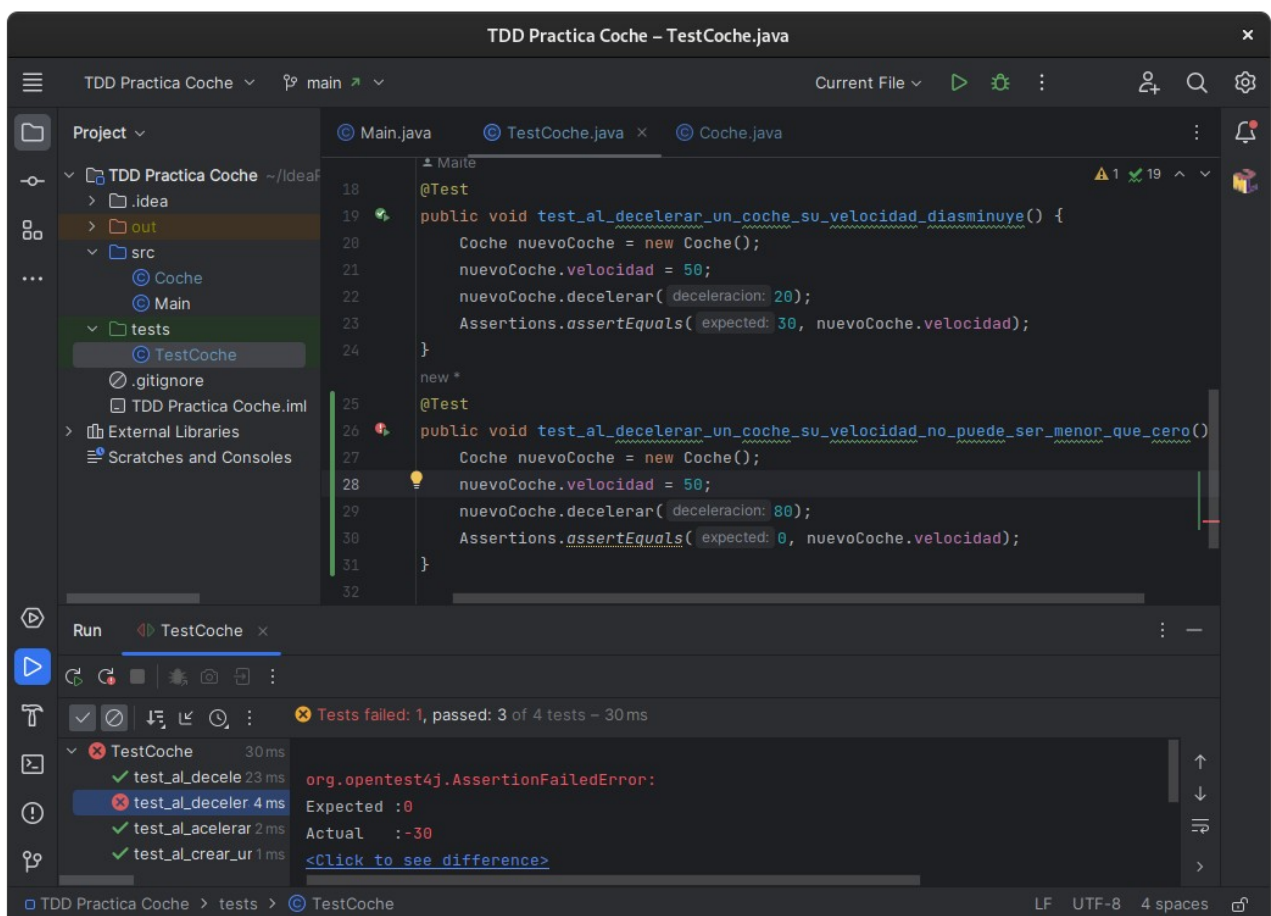
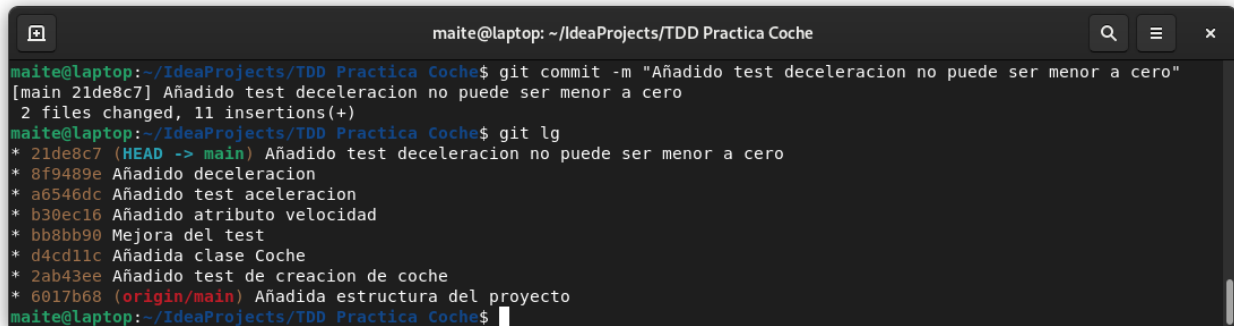


Figura 28: Error del test de deceleración, porque el valor actual es -30.

Podemos ver que nos falla el test porque su valor es inferior a cero.

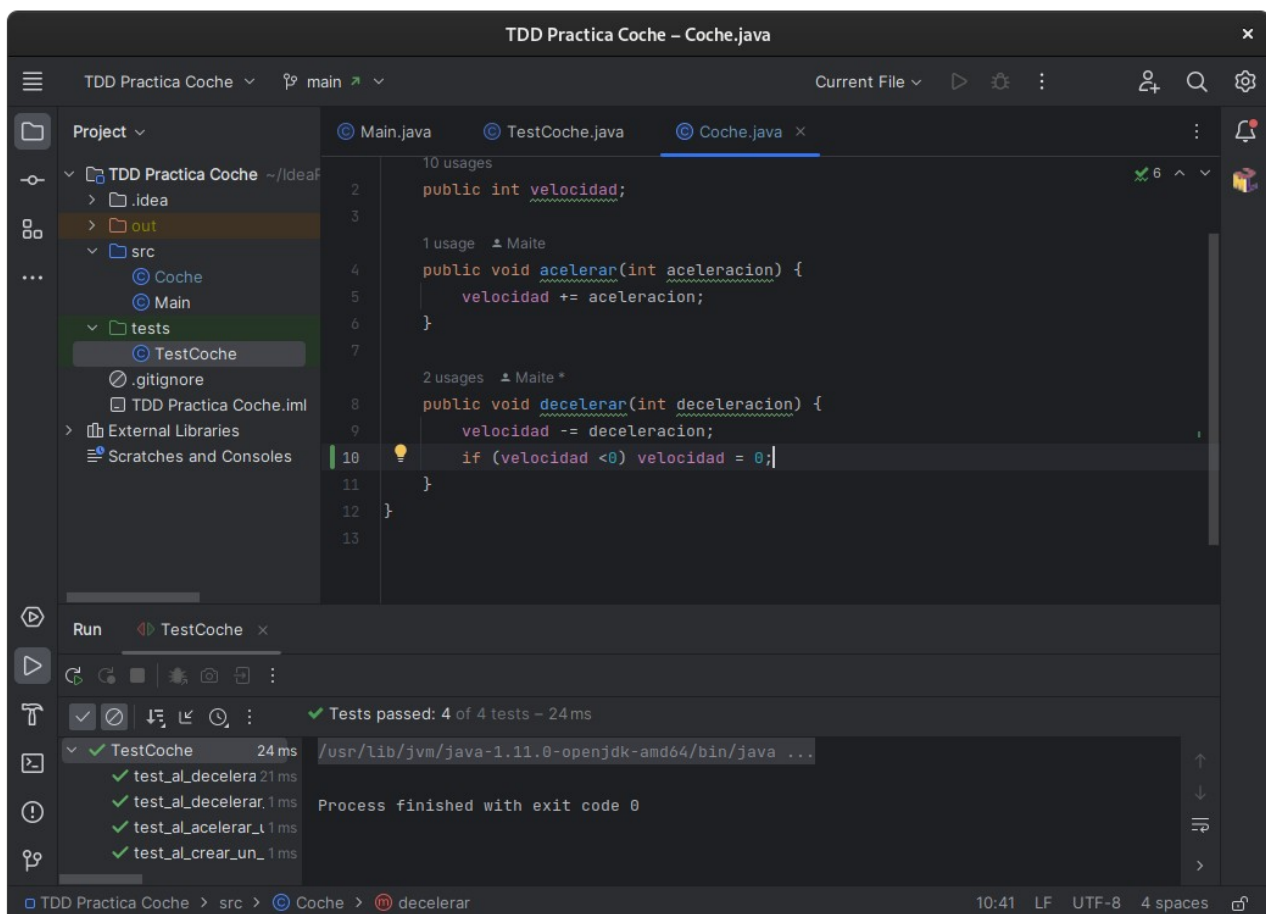
Primero haremos un commit para guardar nuestro nuevo test.



```
maite@laptop: ~/IdeaProjects/TDD Practica Coche
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git commit -m "Añadido test deceleracion no puede ser menor a cero"
[main 21de8c7] Añadido test deceleracion no puede ser menor a cero
2 files changed, 11 insertions(+)
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git lg
* 21de8c7 (HEAD -> main) Añadido test deceleracion no puede ser menor a cero
* 8f9489e Añadido deceleracion
* a6546dc Añadido test aceleracion
* b30ec16 Añadido atributo velocidad
* bb8bb90 Mejora del test
* d4cd11c Añadida clase Coche
* 2ab43ee Añadido test de creacion de coche
* 6017b68 (origin/main) Añadida estructura del proyecto
maite@laptop:~/IdeaProjects/TDD Practica Coche$
```

Figura 29: Commit del nuevo test deceleración no puede ser menor que cero

Vamos a la clase coche para arreglar el error y volveremos a ejecutar el test.



```
TDD Practica Coche - Coche.java
Project: TDD Practica Coche
src
  Coche
  Main
tests
  TestCoche
  .gitignore
  TDD Practica Coche.iml
External Libraries
Scratches and Consoles

Main.java
TestCoche.java
Coche.java

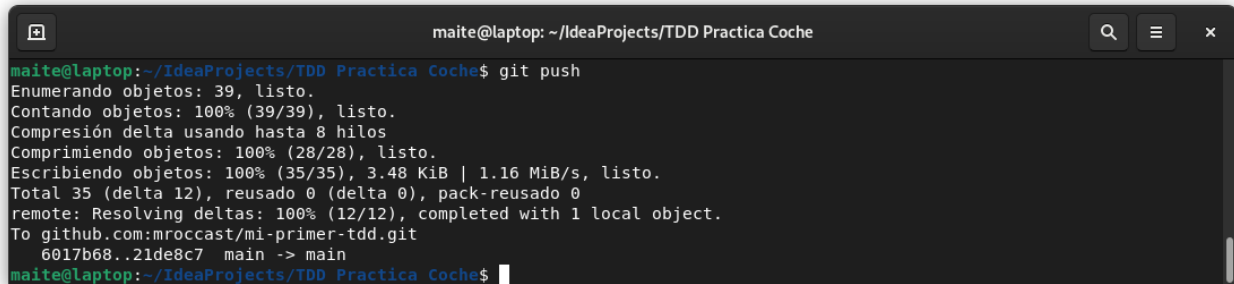
10 usages
2 public int velocidad;
3
4 1 usage Maite
5 public void acelerar(int aceleracion) {
6     velocidad += aceleracion;
7 }
8
9 2 usages Maite *
10 public void decelerar(int deceleracion) {
11     velocidad -= deceleracion;
12     if (velocidad < 0) velocidad = 0;
13 }

Run TestCoche x
Tests passed: 4 of 4 tests - 24 ms
TestCoche 24 ms
  test_al_decelera 21 ms
  test_al_decelerar 1 ms
  test_al_acelerar 1 ms
  test_al_crear_un_ 1 ms
Process finished with exit code 0
```

Figura 30: Resultado del test de deceleración es correcto.

## 13. Sincronizar el proyecto con GitHub

Para finalizar vamos a utilizar el comando `<git push>` para sincronizar el proyecto con GitHub.



```
maite@laptop: ~/IdeaProjects/TDD Practica Coche
maite@laptop:~/IdeaProjects/TDD Practica Coche$ git push
Enumerando objetos: 39, listo.
Contando objetos: 100% (39/39), listo.
Compresión delta usando hasta 8 hilos
Comprimiendo objetos: 100% (28/28), listo.
Escribiendo objetos: 100% (35/35), 3.48 KiB | 1.16 MiB/s, listo.
Total 35 (delta 12), reusado 0 (delta 0), pack-reusado 0
remote: Resolving deltas: 100% (12/12), completed with 1 local object.
To github.com:mroccast/mi-primer-tdd.git
  6017b68..21de8c7  main -> main
maite@laptop:~/IdeaProjects/TDD Practica Coche$
```

Figura 31: Ejecución del comando `<git push>`

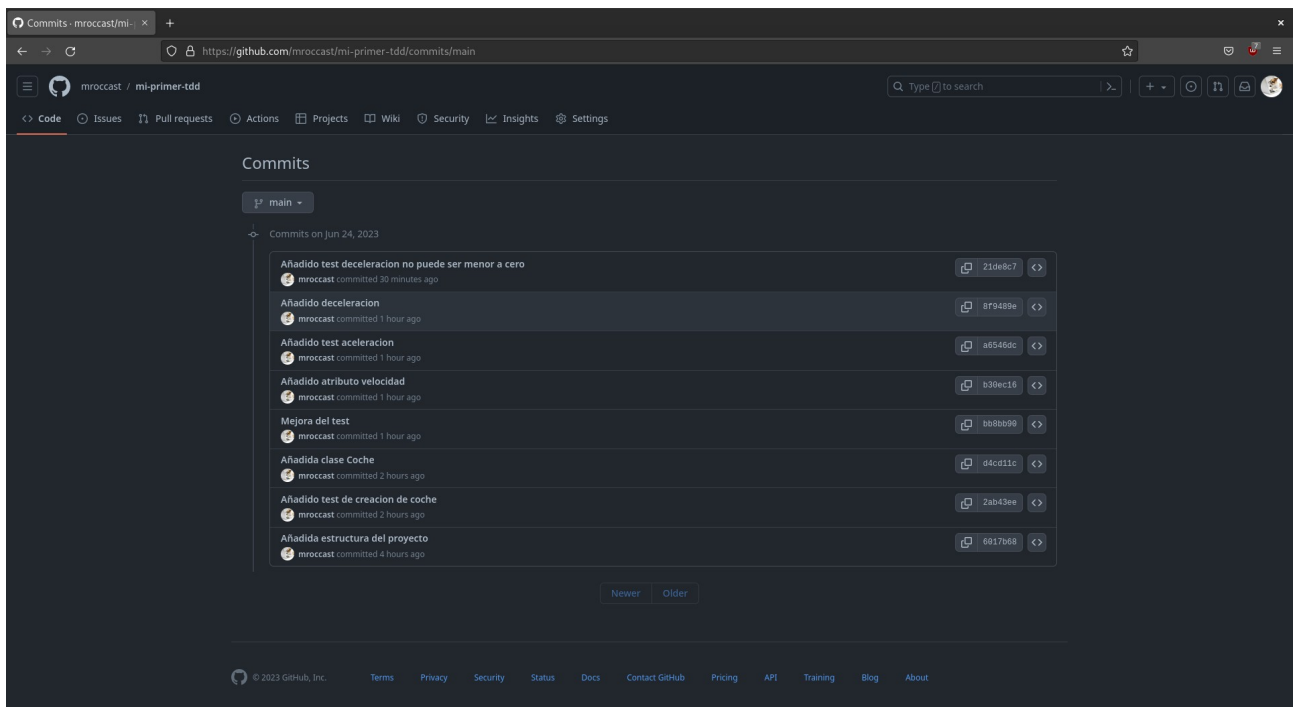


Figura 32: Commits sincronizados con GitHub