Ejemplo de Pruebas Junit.

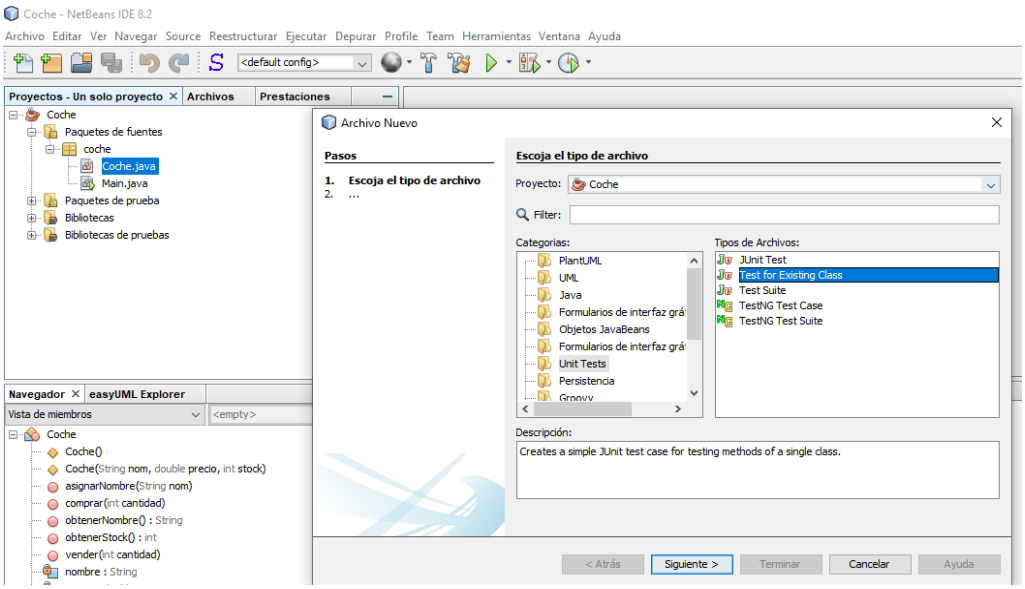
Vamos a realizar las pruebas Junit para la clase Coche que viene definida de la siguiente forma:

Coche.java

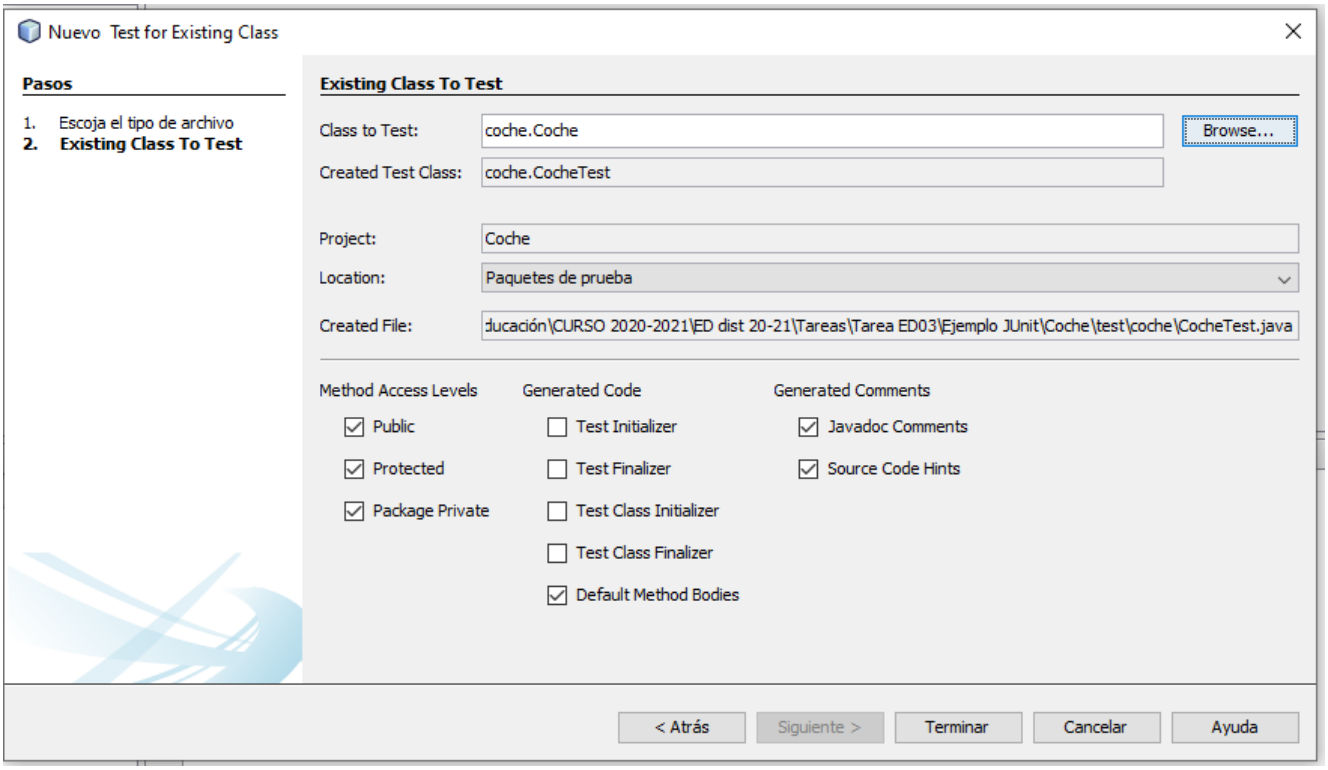
```
package coche;

public class Coche {

    private String nombre;
    private double precio;
    private double precioIVA;
    private int stock;

    /* Constructor sin argumentos */
    public Coche () {}

    // Constructor con parámetro para iniciar todas las propiedades de la clase
    // coche

    public Coche (String nom, double precio, int stock)
    {
        this.nombre =nom;
        this.precio=precio;
        this.stock=stock;
    }
   // Método para asignar el nombre del coche
    public void asignarNombre(String nom)
    {
        nombre=nom;
    }
    // Método que me devuelve el nombre del coche
    public String obtenerNombre()
    {
        return nombre;
    }

    // Método que me devuelve el stock de coches disponible en cada momento
     public int obtenerStock ()
    {
        return stock;
    }

    /* Método para comprar coches. Modifica el stock.
     * Este método va a ser probado con Junit
     */
    public void comprar(int cantidad) throws Exception
    {
        if (cantidad<0)
            throw new Exception("No se puede comprar un n° negativo de coches");
        stock = stock + cantidad;
    }

    public void vender (int cantidad) throws Exception
    {
        if (cantidad <= 0)
            throw new Exception ("No se puede vender una cantidad negativa de coches");
        if (obtenerStock()< cantidad)
            throw new Exception ("No se hay suficientes coches para vender");
        stock = stock - cantidad;
    }

}
```

En concreto vamos a probar el correcto funcionamiento de los métodos comprar y vender.

Para ello generamos el fichero de pruebas JUnit : **CocheTest.java.** Para generar este fichero, posicionados sobre el fichero Coche.java nos vamos a Archivo→ Archivo Nuevo y nos situamos sobre la categoría Unit Tests, ahí escogeremos Test for Existing Class, ya que nuestro proyecto tiene ya el código desarrollado.



A continuación seleccionamos la clase sobre las que se van a generar los test de prueba (en este caso la clase Coche, que es donde se encuentran los métodos a probar) y no es necesario marcar los Test inicializadores ni finalizadores, ya que no vamos a usarlos.



NOTA: Hay que tener en cuenta que lo que se nos va a generar es un prototipo con código que habrá que modificar. Eliminaremos los test que genera para los métodos que no se solicitan (como por ej.asignarNombre, obtenerNombre u obtenerStock).

Con estas pruebas lo que se pretende es comprobar si los métodos Comprar y Vender son correctos.

Para ello vamos a comprobar si introduciendo **valores correctos** los métodos funcionan es decir vamos a comprobar que el método hace lo que debería hacer.

Por otro lado también vamos a comprobar que si introducimos **valores no válidos** como por ejemplo comprar un número negativo de coches o vender más coches de los que hay en stock los métodos comprar y vender no deberían permitirlo.

Para ello una vez generado el fichero CocheTest.java lo modificamos para que quede de la siguiente forma:

TEST DE PRUEBAS para el método COMPRAR

Pruebas para VALORES VÁLIDOS (comprar)

```
/**
* Test para el método Comprar. Si el método comprar es correcto la prueba debe ser exitosa.
* Vamos a comprobar si partiendo de un stock de 300 coches si compro 100 coches
* más el stock es 400. Si es asi el resultado es correcto.
* @throws java.lang.Exception
*/
    public void testComprarValido() throws Exception {
        System.out.println("Test de prueba para Comprar un número positivo de coches");
        int cantidad = 100;
        try{
            Coche cochel = new Coche("Ford",12000,300);
            cochel.comprar(cantidad);
            assertTrue(cochel.obtenerStock()==400); /* Como parto de un stock de 300 al comprar 100
             * coches más ahora el stock es de 400  */
        }catch (Exception e) { /* no debería saltar ninguna excepción en este caso,
            por lo que si lo hace es porque algo no está bien y el test debería fallar */
            fail ("Se ha producido una excepción no esperada: " +e);
        }

    }
```

Pruebas para VALORES NO VÁLIDOS (comprar)

```java
/**
* Test para el método Comprar. En esta prueba intento comprar una cantidad negativa de
* coches. Debe saltar la excepción. Con esta prueba comprobamos que el método comprar no
* acepta números negativos.
* @throws java.lang.Exception
*/

    public void testComprarNoValidoNegativo() throws Exception {
        System.out.println("Test de prueba para Comprar un número negativo de coches");
        int cantidad = -100;
        Coche cochel = new Coche("Ford",12000,300);
       try {
        cochel.comprar(cantidad);
        fail("Intento de comprar un número negativo de coches");
       }
       catch (Exception e){
         System.out.println(e);
         assertTrue(cochel.obtenerStock()==300);
       }
    }
```

TEST DE PRUEBAS para el método VENDER

Pruebas para VALORES VÁLIDOS (vender)

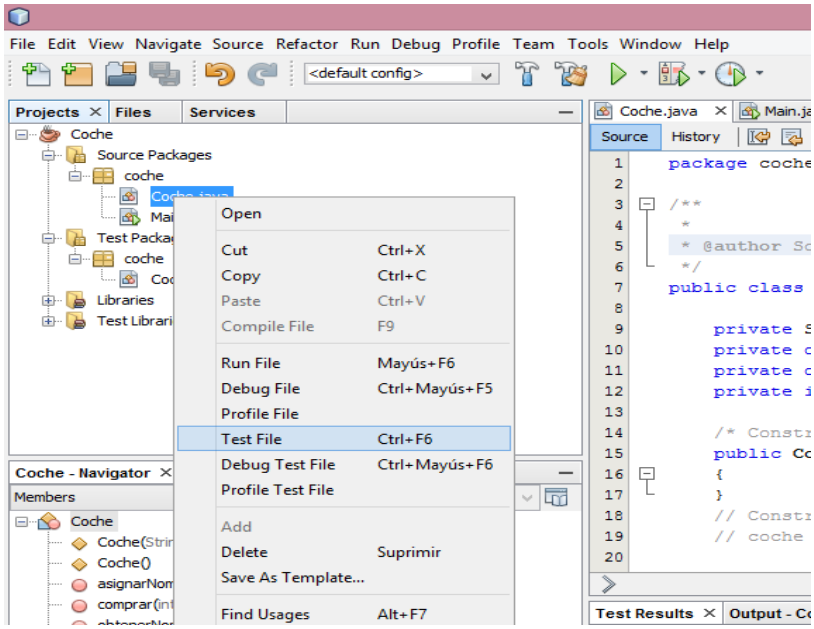
```java
/**
* Test para el método Vender.Si el método vender es correcto la prueba debe
* ser exitosa. En esta prueba vamos a comprobar que si partiendo de un stock
* de 300 coches vendo 200 el stock que me queda es 100
* @throws java.lang.Exception
*/
   public void testVenderValido() throws Exception {
       System.out.println("Test de prueba para vender un número positivo y menor que"
               + " el stock de coches");
       int cantidad = 200;
       Coche cochel = new Coche("Ford",12000,300);
       try {
       cochel.vender(cantidad);
       assertTrue(cochel.obtenerStock()==100); /* Como parto de un stock de 300 al comprar 100
        * coches más ahora el stock es de 400  */
       }catch (Exception e) { /* no debería saltar ninguna excepción en este caso,
           por lo que si lo hace es porque algo no está bien y el test debería fallar */
           fail ("Se ha producido una excepción no esperada: " +e);
       }
   }
```

Pruebas para VALORES NO VÁLIDOS (vender)

```java
/**
* Test para el método Vender. En esta prueba intento vender un número negativo de coches.
* El método vender no debe permitirlo y debe saltar
* la excepción.
* @throws java.lang.Exception */

    public void testVenderNoValidoNegativo() throws Exception {
        System.out.println("Test de prueba para vender un número negativo de coches");
        int cantidad = -200;
        Coche cochel = new Coche("Ford",12000,300);
         try {
             cochel.vender(cantidad);
             fail("Intento de vender un número negativo de coches");
       }
       catch (Exception e){
         System.out.println(e);
         assertTrue(cochel.obtenerStock()==300);
       }

    }
```

```java
/**
 * Test para el método Vender.En esta prueba intento vender más coches
 * que hay en stock. El método vender no debe permitirlo y debe saltar la excepción.
 * @throws java.lang.Exception
 */
    public void testVenderNoValidoMayor() throws Exception {
        System.out.println("Test de prueba para vender un número positivo pero mayor que"
                + " el stock de coches ");
        int cantidad = 400;
        Coche cochel = new Coche("Ford",12000,300);
        try {
             cochel.vender(cantidad);
             fail("Intento de vender más coches que hay en el stock");
       }
       catch (Exception e){
         System.out.println(e);
         assertTrue(cochel.obtenerStock()==300);
       }

    }
```

Ahora nos queda ejecutar las pruebas para ver los resultados: Para ello pinchamos con el botón derecho sobre CocheTest.java y seleccionamos Ejecutar archivo o directamente sobre el menú contextual del proyecto Coche seleccionando Probar.



Efectivamente los resultados son los esperados. Todas las pruebas son exitosas, lo que significa que, si las pruebas están correctamente diseñadas, no se ha encontrado ningún error en el código.