

Justificación y capturas

Entornos de desarrollo, DAM

Francisco Jesús Díaz Martínez
IES AGUADULCE , ALMERÍA

Apartado 1.1, refactorizaciones usando las herramientas del IDE NETBEANS:

This screenshot shows the NetBeans IDE interface. On the left, there is a web browser window displaying the 'Perfil de usuario' (User Profile) page of the 'Formación Profesional' (Professional Training) website. The profile belongs to Francisco Jesús Díaz Martínez. The main area of the IDE shows the 'Vehiculo' project with a file named 'VehiculoDiazMartinezFranciscoJesus2223.java'. A 'Rename' dialog box is open, showing the current name and a new name: 'miVehiculoDiazMartinezFranciscoJesus2223'. The 'Apply Rename on Comments' checkbox is checked. The 'Refactor' button is highlighted.

This screenshot shows the NetBeans IDE interface. On the left, there is a web browser window displaying the 'Perfil de usuario' (User Profile) page of the 'Formación Profesional' (Professional Training) website. The profile belongs to Francisco Jesús Díaz Martínez. The main area of the IDE shows the 'Vehiculo' project with a file named 'Vehiculo.java'. A 'Rename Class Vehiculo' dialog box is open, showing the current name and a new name: 'VehículoFranciscoJesusDiazMartinez2223'. The 'Apply Rename on Comments' checkbox is checked. The 'Refactor' button is highlighted.

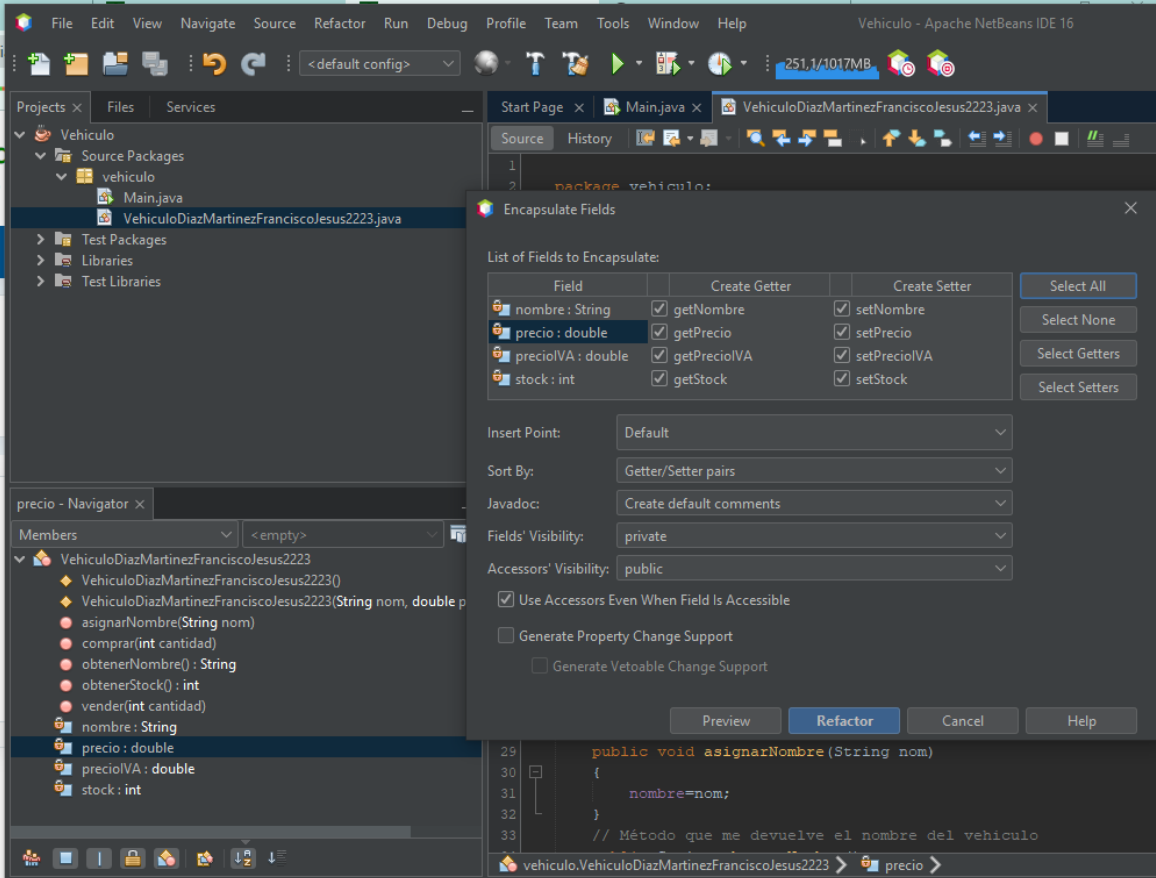
Apartado 1.2, introducción del método “operativaVehículos”

The image displays a web browser on the left and the Apache NetBeans IDE on the right. The browser shows a user profile page for Francisco Jesús Díaz Martínez. The IDE is open to a Java project named 'Vehiculo'. The 'Main.java' file is selected, and the 'Introduce Method' dialog is visible, showing the method name 'operativaVehiculosDiazMartinezFranciscoJesus2223' and the target class 'vehiculo.Main'. The code in 'Main.java' is as follows:

```
10 public static void main(String[] args) {
11     VehiculoDiazMartinezFranciscoJesus2223 miVehiculoDiazMartinezFranciscoJesus2223;
12     int stockActual;
13
14     miVehiculoDiazMartinezFranciscoJesus2223 = new VehiculoDiazMartinezFranciscoJesus2223();
15     try
16     {
17
18     }
19
20     stockActual = miVehiculoDiazMartinezFranciscoJesus2223.obtenerStock();
21     System.out.println("El stock actual es" + stockActual);
22 }
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
```

The 'operativaVehiculosDiazMartinezFranciscoJesus2223()' method is highlighted in red in the code editor. The 'operativaVehiculosDiazMartinezFranciscoJesus2223()' method is also visible in the 'Main' class member list.

Apartado 1.3, encapsulamiento de todos los atributos de la clase “VehículoXXX2223”

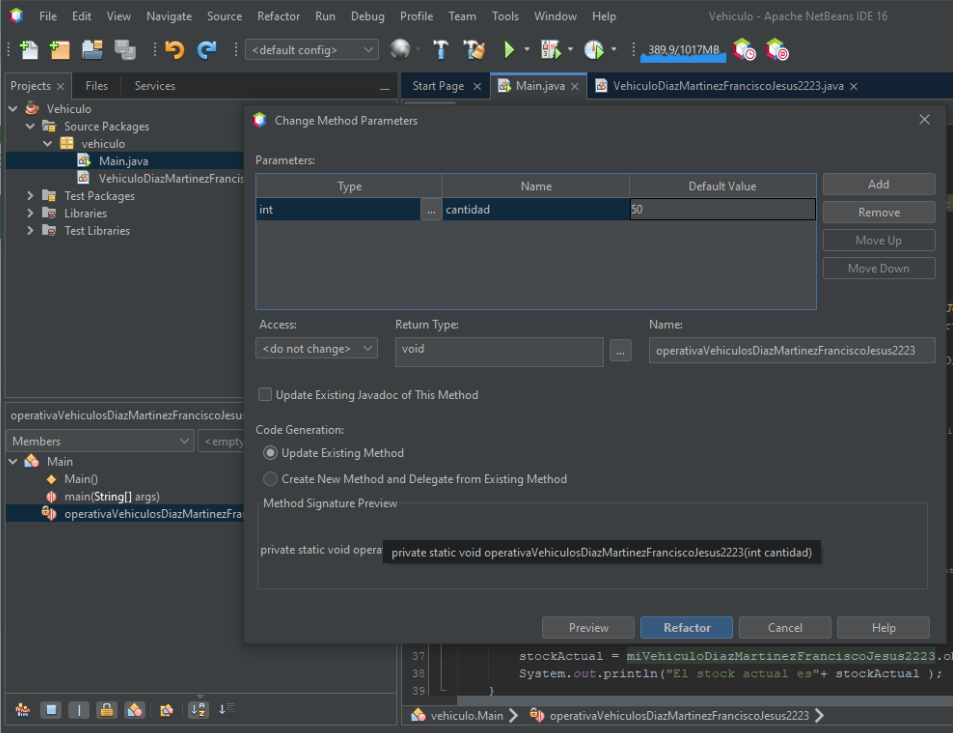


The screenshot shows the Apache NetBeans IDE interface. On the left, there's a sidebar with a user profile for Francisco Jesús Díaz Martínez. The main workspace displays the 'Encapsulate Fields' dialog box. The dialog lists the following fields to encapsulate:

Field	Create Getter	Create Setter
nombre : String	<input checked="" type="checkbox"/> getNombre	<input checked="" type="checkbox"/> setNombre
precio : double	<input checked="" type="checkbox"/> getPrecio	<input checked="" type="checkbox"/> setPrecio
precioIVA : double	<input checked="" type="checkbox"/> getPrecioIVA	<input checked="" type="checkbox"/> setPrecioIVA
stock : int	<input checked="" type="checkbox"/> getStock	<input checked="" type="checkbox"/> setStock

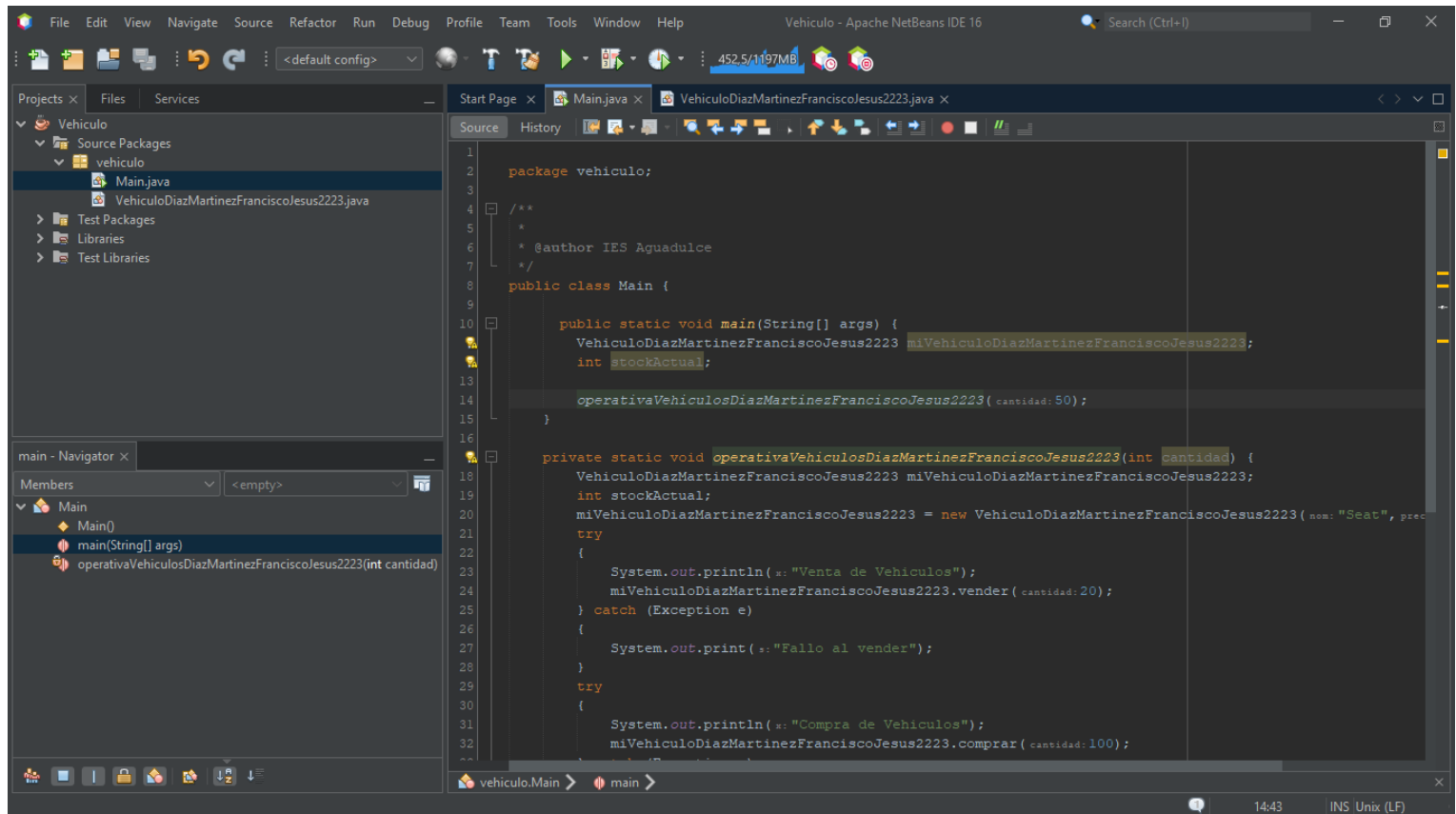
The dialog also includes options for 'Insert Point' (Default), 'Sort By' (Getter/Setter pairs), 'Javadoc' (Create default comments), 'Fields' Visibility' (private), and 'Accessors' Visibility' (public). The 'Refactor' button is highlighted.

Apartado 1.4, añadir un parámetro (por defecto 50) al método “operativaVehículos”

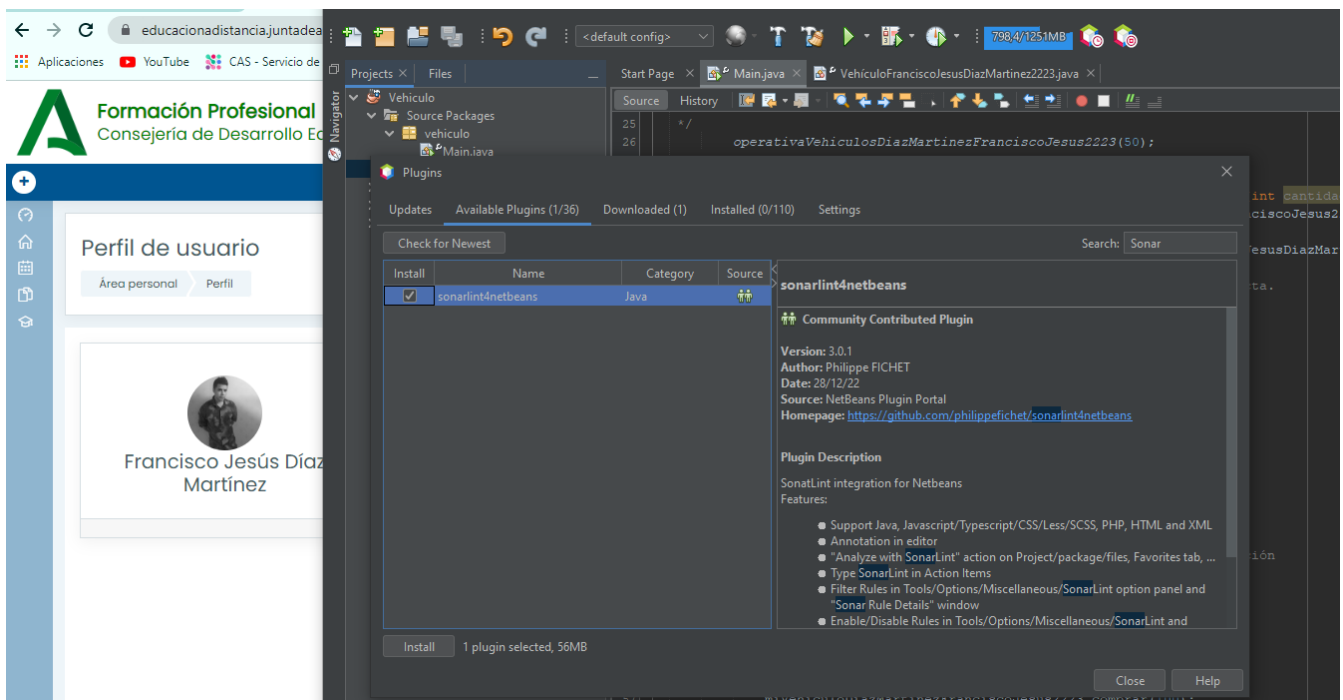


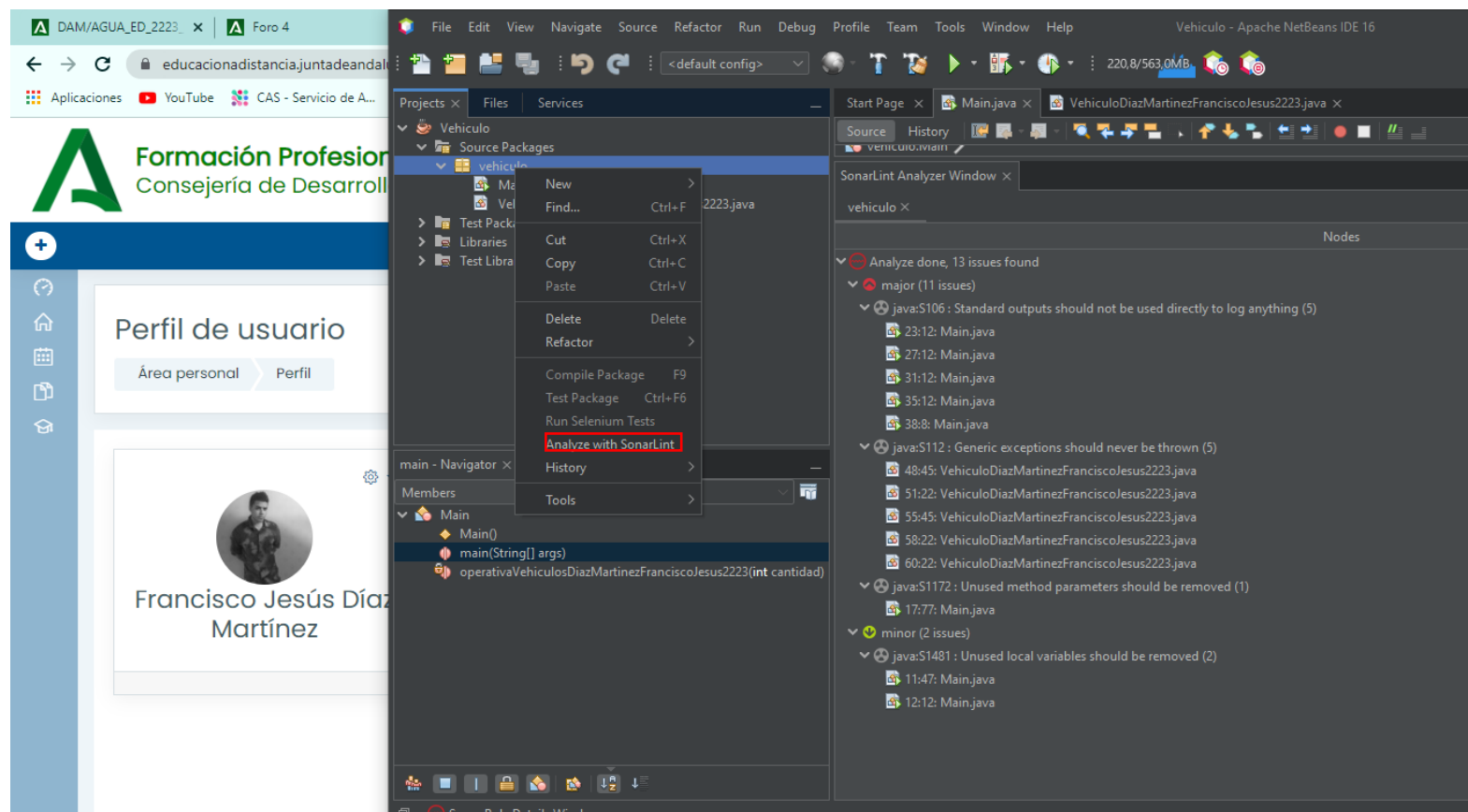
The screenshot shows the Apache NetBeans IDE interface. On the left, there's a sidebar with a user profile for Francisco Jesús Díaz Martínez. The main workspace displays the 'Change Method Parameters' dialog box. The dialog shows the method 'operativaVehiculosDiazMartinezFranciscoJesus2223' with a new parameter 'cantidad' of type 'int' and a default value of '50'. The 'Refactor' button is highlighted.

Muestra de la cantidad por defecto (50) y del parámetro “cantidad” al método



Apartado 2.1, instalación y análisis del plugin SonarLint en el IDE NetBeans





Después de pasar el código por el análisis de SonarLint, podemos observar que nos devuelve una tabla (a la derecha en la ventana de outputs) con los problemas que se han encontrado.

Son 13 en total, que se dividen según su importancia. A su vez, dentro de esta división, se definen cuáles son los errores, y en qué clase se encuentran.

Si comentamos brevemente el resultado del análisis, básicamente todos los errores que nos indica, van orientados a optimizar la sintaxis del código al máximo y evitar la redundancia.

En ningún caso provocan un fallo del código, pero si es interesante atender a sus instrucciones para limpiar nuestro programa para su máxima legibilidad.

Apartado 2.2, modificación de las reglas del plugin para eliminar una de las condiciones

The screenshot shows the Apache NetBeans IDE interface. On the left, there is a sidebar with a navigation menu and a panel titled 'DAM/AGUA - Entornos de desarrollo - Grup'. The main workspace displays the 'Sonar Rule Details Window' for the rule 'Unused method parameters should be removed' (java:S1172). The rule is categorized as 'MAJOR' and 'CODE_SMELL'. The description states: 'Unused parameters are misleading. Whatever the values passed to such parameters, the behavior will be the same.' It provides a 'Noncompliant Code Example' and a 'Compliant Solution'. The 'Exceptions' section lists conditions where the rule will not raise issues: 'that are annotated with @javax.enterprise.event.Observes', 'in overrides and implementation methods', 'in interface default methods', and 'in non-private methods that only throw or that have empty bodies'.

The screenshot shows the Apache NetBeans IDE interface. On the left, there is a sidebar with a navigation menu and a panel titled 'DAM/AGUA - Entornos de desarrollo - Grup'. The main workspace displays the 'SonarLint Analyzer Window' for the project 'vehiculo'. The window shows a list of issues found during the analysis. The issues are categorized by severity: 'major (10 issues)' and 'minor (2 issues)'. The 'major' issues include 'Standard outputs should not be used directly to log anything (5)' and 'Generic exceptions should never be thrown (5)'. The 'minor' issues include 'Unused local variables should be removed (2)'. The 'main - Navigator' panel on the left shows the project structure, including 'Main' and 'operativaVehiculosDiazMartinezFranciscoJesús2223(int cantidad)'.

Desactivamos la regla “Unused method parameters should be removed” y como vemos ya no aparece en la lista del análisis el S1172, a diferencia de la primera captura.

Apartado 3.1, creación del repositorio de GitHub

The screenshot shows the GitHub repository creation page for 'FranJDM/VehiculoFJDM2223'. The page is titled 'Quick setup — if you've done this kind of thing before'. It offers three options: 'Set up in Desktop', 'HTTPS', and 'SSH'. The 'HTTPS' option is selected, and the repository URL 'https://github.com/FranJDM/VehiculoFJDM2223.git' is displayed. Below this, there is a section titled '...or create a new repository on the command line' with a code block containing the following commands:

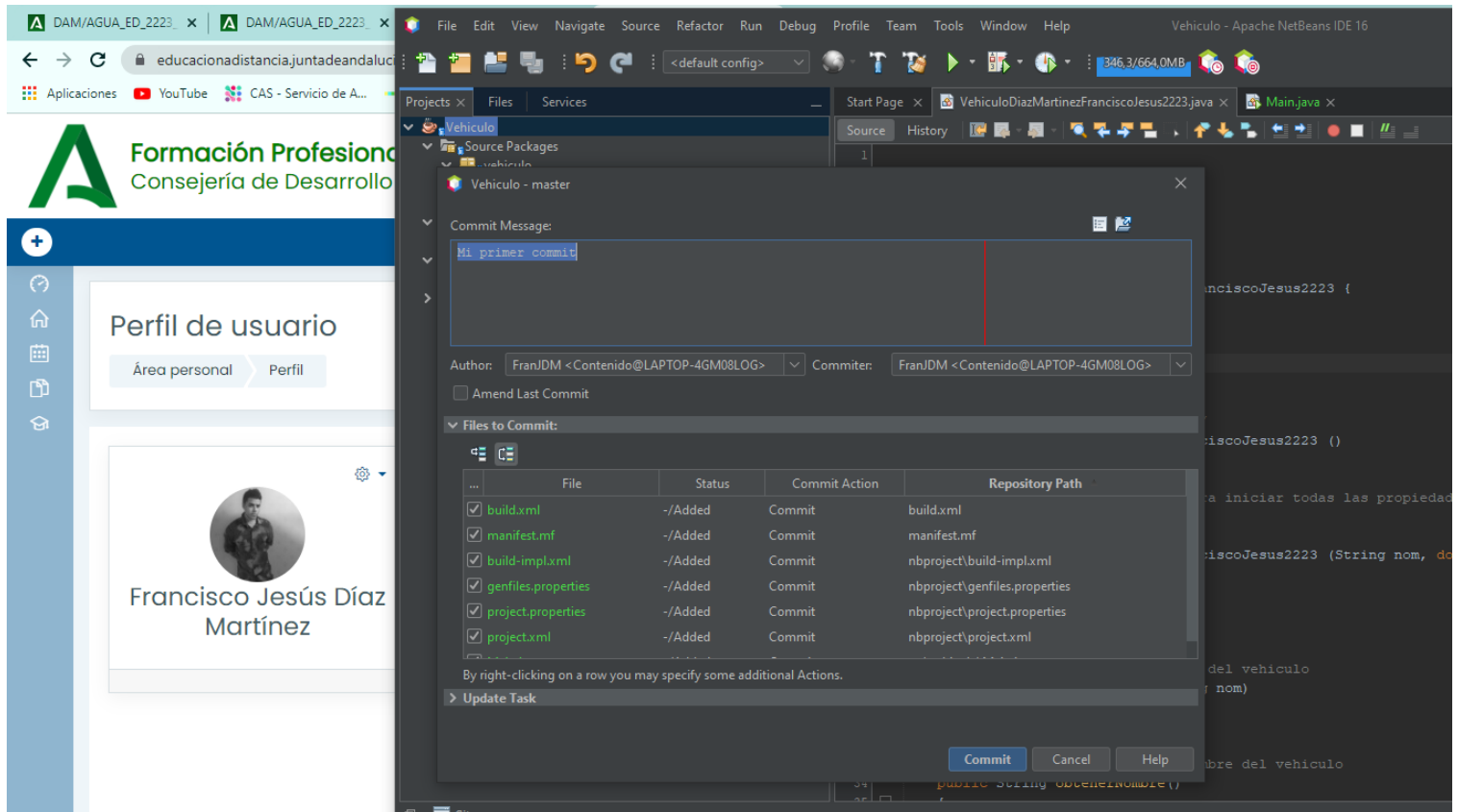
```
echo "# VehiculoFJDM2223" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/FranJDM/VehiculoFJDM2223.git
git push -u origin main
```

Below the command line section, there is a section titled '...or push an existing repository from the command line' with the following command:

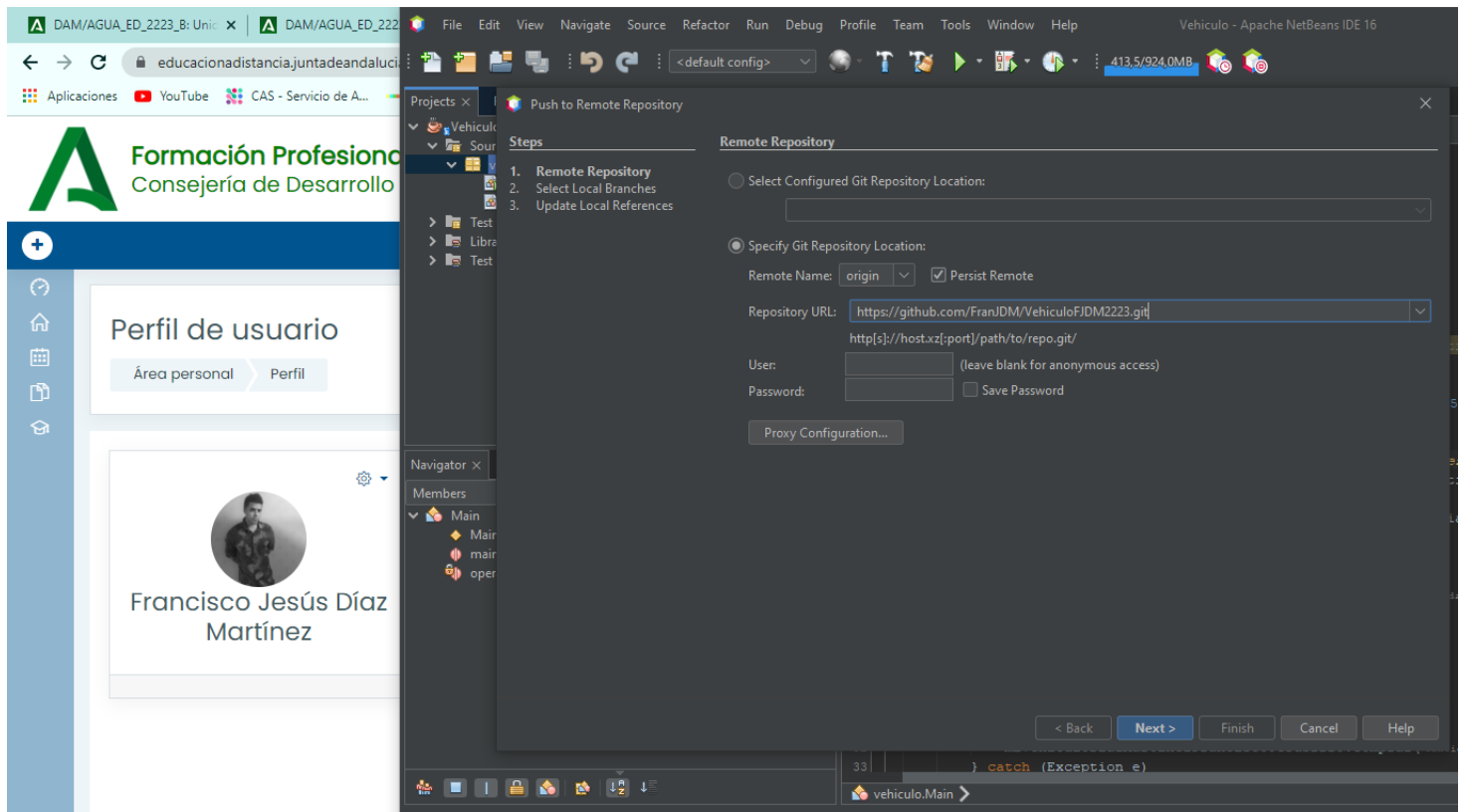
```
git remote add origin https://github.com/FranJDM/VehiculoFJDM2223.git
```

Apartado 3.2 inicializamos el repositorio y hacemos el commit con el mensaje indicado

The screenshot shows the Apache NetBeans IDE interface. The 'Projects' view on the left shows a project named 'Vehiculo' with a sub-project 'vehiculo'. The 'Files' view shows the 'Main.java' file. The 'Run' view on the right shows the output of the 'git init' command, indicating that the repository has been initialized successfully. The 'Run' view also shows the output of the 'git commit' command, indicating that the commit has been made successfully with the message 'first commit'.



Apartado 3.3 hacemos el push para sincronizar el repositorio mediante el enlace de GitHub



The screenshot shows a GitHub repository page for 'FranJDM / -VehiculoFJDM2223'. The repository is public and has 1 branch (master) and 0 tags. The commit history shows a single commit by 'Fran and Fran' with the message 'Mi primer Commit FranciscoJesúsDíazMartínez 2223' 11 minutes ago. The commit includes files: nbproject, src/vehiculo, build.xml, and manifest.mf. The repository description is 'Repositorio del proyecto Vehículo'. The page also shows sections for Releases and Packages, both of which are currently empty.

File	Commit Message	Time
nbproject	Mi primer Commit FranciscoJesúsDíazMartínez 2223	11 minutes ago
src/vehiculo	Mi primer Commit FranciscoJesúsDíazMartínez 2223	11 minutes ago
build.xml	Mi primer Commit FranciscoJesúsDíazMartínez 2223	11 minutes ago
manifest.mf	Mi primer Commit FranciscoJesúsDíazMartínez 2223	11 minutes ago

Podemos observar como el contenido del proyecto está disponible en el repositorio con la descripción indicado en el commit.

[Enlace del repositorio disponible aquí.](#)

Cabe destacar, que a la hora de hacer el push desde el IDE, hay que indicar un usuario y una contraseña. Desde agosto del 2021, GitHub no valida las contraseñas; solo usa tokens generados aleatoriamente para cada usuario.

Es decir, para hacer el push, necesitamos generar un token desde nuestra cuenta, y usarlo a modo de contraseña para que se pueda sincronizar el repositorio.

4.1 Comentarios del código, de las dos clases del proyecto

- **Clase Main**

```
package vehiculo;

/**
 * Clase main, donde se describen las interacciones de los atributos
 * de la clase
 * "VehiculoDiazMartinezFranciscoJesus2223"
 *
 * @author IES Aguadulce
 */
public class Main {

    /**
     * Método main; principal para la ejecución del código
     * @param args
     */

    public static void main(String[] args) {
        VehiculoFranciscoJesusDiazMartinez2223
        miVehiculoDiazMartinezFranciscoJesus2223;
        int stockActual;

        /**
         * Introducimos el método que engloba las sentencias que operan con el
         * objeto
         * "miVehiculoDiazMartinezFranciscoJesus"
         */
        operativaVehiculosDiazMartinezFranciscoJesus2223(50);
    }

    private static void
    operativaVehiculosDiazMartinezFranciscoJesus2223(int cantidad) {
        VehiculoFranciscoJesusDiazMartinez2223
        miVehiculoDiazMartinezFranciscoJesus2223;
        int stockActual;
        miVehiculoDiazMartinezFranciscoJesus2223 = new
        VehiculoFranciscoJesusDiazMartinez2223("Seat",18000,100);
        /**
         * Se van a vender 20 vehículos por lo tanto, la ejecución debe ser
         * correcta.
         * Ya que tenemos 100 unidades en stock
         */

        try
        {
            System.out.println("Venta de Vehiculos");
            miVehiculoDiazMartinezFranciscoJesus2223.vender(20);
        } catch (Exception e)
        {
            System.out.print("Fallo al vender");
        }
    }
}
```

```
/**
 *Se van a intentar comprar 100 vehículos. No debería de haber
problema
 * en el método comprar no se establece cantidad máxima, solo la
restricción
 * para que no se compren cantidades iguales o menores que "0"
 *
 * Además se modificará el recuento de stock actual
 */
    try
    {
        System.out.println("Compra de Vehiculos");
        miVehiculoDiazMartinezFranciscoJesus2223.comprar(100);
    } catch (Exception e)
    {
        System.out.print("Fallo al comprar");
    }
    stockActual =
miVehiculoDiazMartinezFranciscoJesus2223.obtenerStock();
    System.out.println("El stock actual es "+ stockActual );
}
}
```

Como se puede observar, los comentarios correspondientes están en color azul.

- **Clase Vehículo**

```
package vehiculo;
```

```
/**
 * Clase enfocada a todas las operaciones posibles que se han
determinado para
 * la interacción con los coches. Venta, Compra y Control de Stock
 *
 * @author IES Aguadulce
 */
public class VehiculoFranciscoJesusDiazMartinez2223 {

    private String nombre; //Nombre del vehículo
    private double precio; //Precio del vehículo
    private double precioIVA; //Importe proporcional del precio por
el IVA del vehículo
    private int stock; //Disponibilidad en existencias del vehículo

    /* Constructor sin argumentos */
    public VehiculoFranciscoJesusDiazMartinez2223 ()
    {
    }

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

```
public VehiculoFranciscoJesusDiazMartinez2223 (String nom, double
precio, int stock)
{
    this.nombre =nom;
    this.precio=precio;
    this.stock=stock;
}
/** Método para asignar el nombre del vehiculo
 *
 * @param nom
 */
public void asignarNombre (String nom)
{
    setNombre (nom) ;
}
/** Método que me devuelve el nombre del vehiculo
 *
 * @return obtiene el nombre del vehículo
 */
public String obtenerNombre ()
{
    return getNombre () ;
}

/** Método que me devuelve el stock de vehiculos disponible en
cada momento
 *
 * @return obtiene el stock de vehículos
 */
public int obtenerStock ()
{
    return getStock () ;
}

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

/** Método para vender vehiculos.Modifica el stock.Este método
va a ser probado con Junit
 * @param cantidad
 * @throws java.lang.Exception
 */
public void vender (int cantidad) throws Exception
{
    if (cantidad <= 0)
```

```
        throw new Exception ("No se puede vender una cantidad
negativa de vehiculos");
        if (obtenerStock() < cantidad)
            throw new Exception ("No se hay suficientes vehiculos para
vender");
        setStock(getStock() - cantidad);
    }

    /**
     * Devuelve el nombre del producto
     * @return nombre del producto
     */
    public String getNombre() {
        return nombre;
    }

    /**
     * Establece el nuevo nombre asignado al producto
     * @param nombre "nuevo nombre de producto"
     */
    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    /**
     * Devuelve el precio del producto
     * @return el precio del producto
     */
    public double getPrecio() {
        return precio;
    }

    /**
     * Establece el nuevo precio del producto
     * @param precio "nuevo precio del producto"
     */
    public void setPrecio(double precio) {
        this.precio = precio;
    }

    /**
     * Devuelve el precio+IVA del producto
     * @return el precioIVA
     */
    public double getPrecioIVA() {
        return precioIVA;
    }

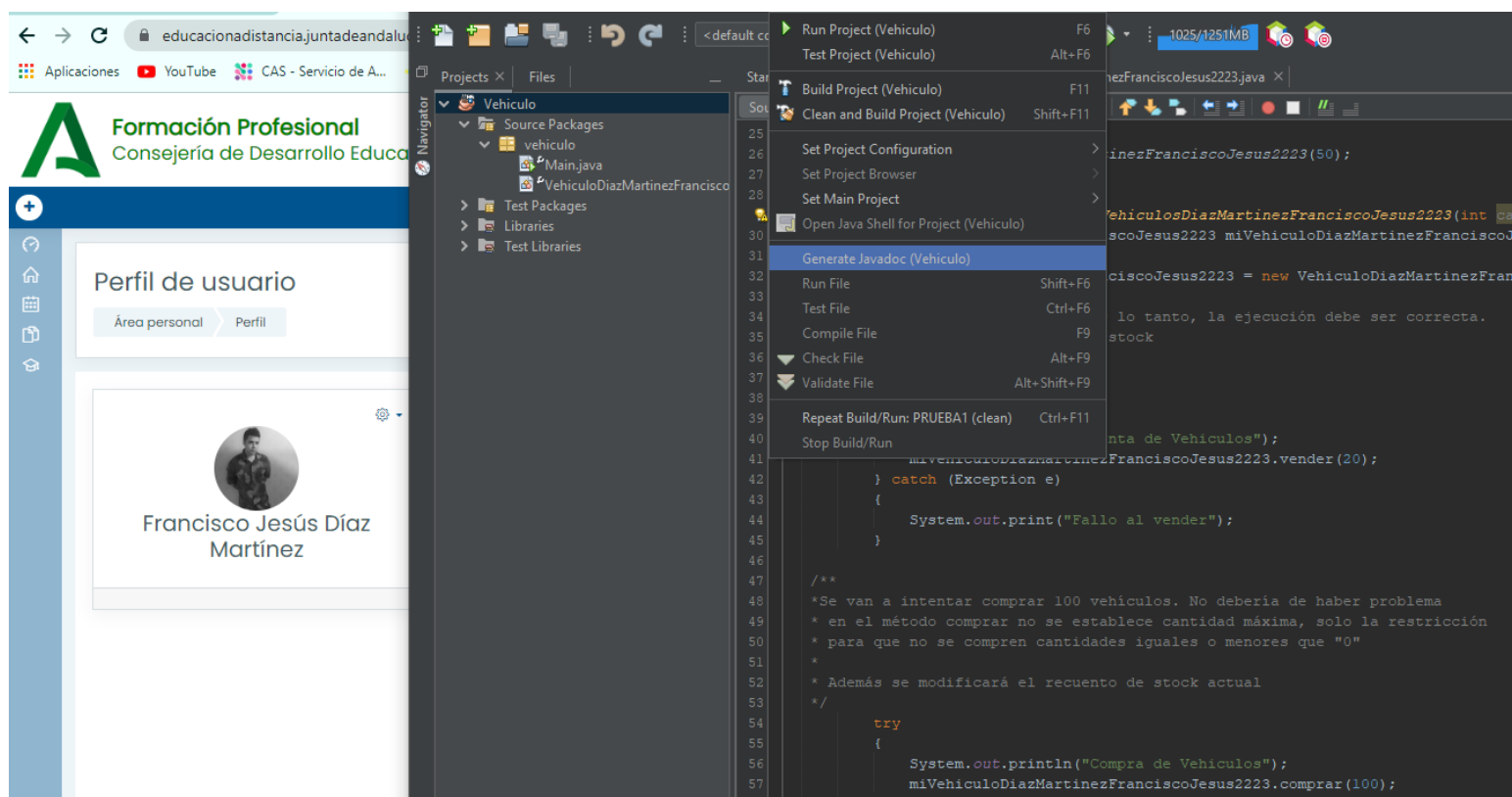
    /**
     * Establece el nuevo precio+IVA del producto
     * @param precioIVA "nuevo precio+IVA del coche"
     */
    public void setPrecioIVA(double precioIVA) {
        this.precioIVA = precioIVA;
    }

    /**
```

```
    * Devuelve el stock actual de coches
    * @return "el stock de vehículos"
    */
    public int getStock() {
        return stock;
    }

    /**
     * Establece el nuevo valor del stock de coches
     * @param stock "el nuevo stock de vehículos"
     */
    public void setStock(int stock) {
        this.stock = stock;
    }
}
```

4.2 Generando el JAVADOC de todo el proyecto



C:\Users\Fran\Desktop\Vehiculo\dist\javadoc\vehiculo\package-summary.html

Buscar...

vehiculo

PACKAGE

CLASS

USE

TREE

DEPRECATED

INDEX

HELP

ALL CLASSES

SEARCH:

Package vehiculo

Class Summary

Class	Description
Main	Clase main, donde se describen las interacciones de los atributos de la clase "VehiculoDiazMartinezFranciscoJesus2223"
VehiculoDiazMartinezFranciscoJesus2223	Clase enfocada a todas las operaciones posibles que se han determinado para la interacción con los coches.

C:\Users\Fran\Desktop\Vehiculo\dist\javadoc\vehiculo\Main.html

Buscar...

Main

PACKAGE

CLASS

USE

TREE

DEPRECATED

INDEX

HELP

ALL CLASSES

SEARCH:

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Constructor Summary

Constructors

Constructor	Description
<code>Main()</code>	

Method Summary

All MethodsStatic MethodsConcrete Methods

Modifier and Type	Method	Description
static void	<code>main(java.lang.String[] args)</code>	Método main; principal para la ejecución del código

Methods inherited from class java.lang.Object

<code>clone</code> , <code>equals</code> , <code>finalize</code> , <code>getClass</code> , <code>hashCode</code> , <code>notify</code> , <code>notifyAll</code> , <code>toString</code> , <code>wait</code> , <code>wait</code> , <code>wait</code>

Constructor Detail

All Methods	Instance Methods	Concrete Methods
Modifier and Type	Method	Description
void	asignarNombre (java.lang.String nom)	Método para asignar el nombre del vehiculo
void	comprar (int cantidad)	Método para comprar vehiculos.Modifica el stock.Este método va a ser probado con Junit
java.lang.String	getNombre ()	Devuelve el nombre del producto
double	getPrecio ()	Devuelve el precio del producto
double	getPrecioIVA ()	Devuelve el precio+IVA del producto
int	getStock ()	Devuelve el stock actual de coches
java.lang.String	obtenerNombre ()	Método que me devuelve el nombre del vehiculo
int	obtenerStock ()	Método que me devuelve el stock de vehiculos disponible en cada momento
void	setNombre (java.lang.String nombre)	Establece el nuevo nombre asignado al producto
void	setPrecio (double precio)	Establece el nuevo precio del producto
void	setPrecioIVA (double precioIVA)	Establece el nuevo precio+IVA del producto
void	setStock (int stock)	Establece el nuevo valor del stock de coches
void	vender (int cantidad)	Método para vender vehiculos.Modifica el stock.Este método va a ser probado con Junit

Este es el JAVADOC generado para el proyecto, todos los métodos están definidos e identificados según su tipo de variable.