

TAREA UNIDAD 4 ED

1.- Refactorización

1.1.- Pasar las clases Main y Cuenta al paquete cuentas:

Para pasar las clases al paquete cuentas tenemos dos opciones. La primera es renombrar a **cuentas**, usando refactorización, el paquete en el que se encuentran actualmente, y la segunda es crear un nuevo paquete al que llamaremos **cuentas** y mover las clases a éste usando mediante refactorización. En las siguientes imágenes veremos como hacer las dos opciones.

1.1.1.- Renombrar paquete actual:

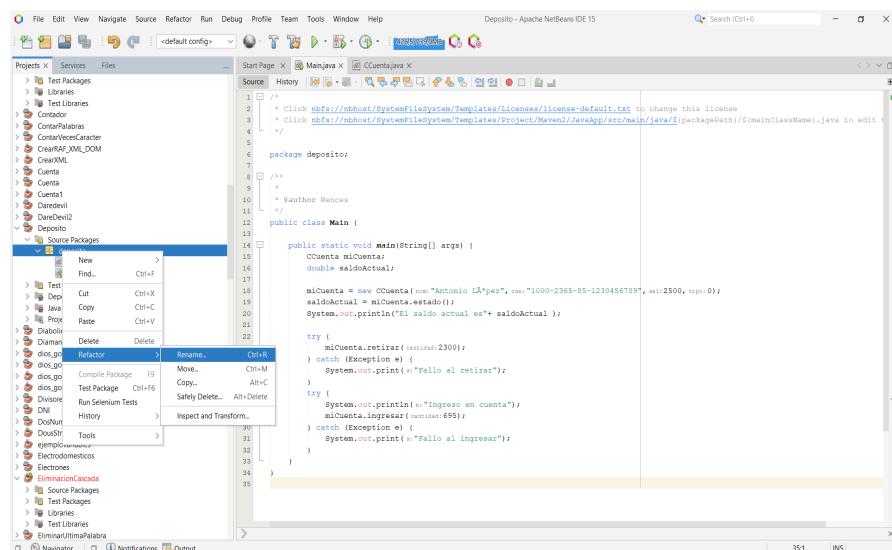


Imagen 1.- Escogemos la opcion **renombrar**

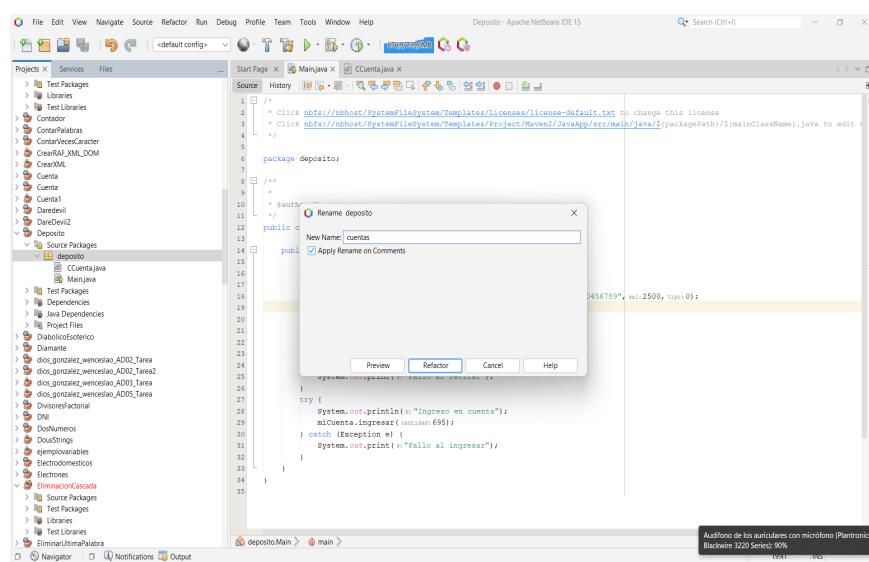


Imagen 2.- Escribimos el nuevo nombre **cuentas**

The screenshot shows the Apache NetBeans IDE interface. The left sidebar displays the project structure under 'Projects X Services Files'. A 'Source Packages' node is expanded, showing several packages like 'Cuenta', 'Cuenta1', 'Cuenta2', and 'Deposito'. Under 'Deposito', there is a 'cuentas' folder which contains 'Cuenta.java' and 'Manjava'. The right side shows the 'Start Page' with tabs for 'History', 'Source', and 'Output'. The 'Source' tab is active, displaying the code for 'Cuenta.java'. The code defines a package 'cuentas' and a main class 'Main' with a static void main method. It creates a 'CCuenta' object named 'miCuenta' with attributes 'nombre', 'cuenta', 'saldoActual', and 'estado'. It then attempts to withdraw 2300 units from the account. The code includes try-catch blocks for potential exceptions.

```

1 package cuentas;
2
3 /**
4  * 
5  * @author Wences
6 */
7
8 public class Main {
9
10    public static void main(String[] args) {
11        CCuenta miCuenta;
12        double saldoActual;
13
14        miCuenta = new CCuenta("Antonio Lopez", "1000-2345-85-1230456789", 2500, 0);
15        System.out.println("El saldo actual es "+ saldoActual);
16
17        try {
18            miCuenta.retirar(2300);
19        } catch (Exception e) {
20            System.out.print("Falló al retirar");
21        }
22        try {
23            System.out.println("Ingreso en cuenta");
24            miCuenta.ingresar(695);
25        } catch (Exception e) {
26            System.out.print("Falló al ingresar");
27        }
28    }
29
30}
31
32}
33
34}
35
36}
37
38}
39

```

Imagen 3.- Las clases ya están dentro del paquete **cuentas**

1.1.2.- Mover las clases a un nuevo paquete llamado **cuentas**:

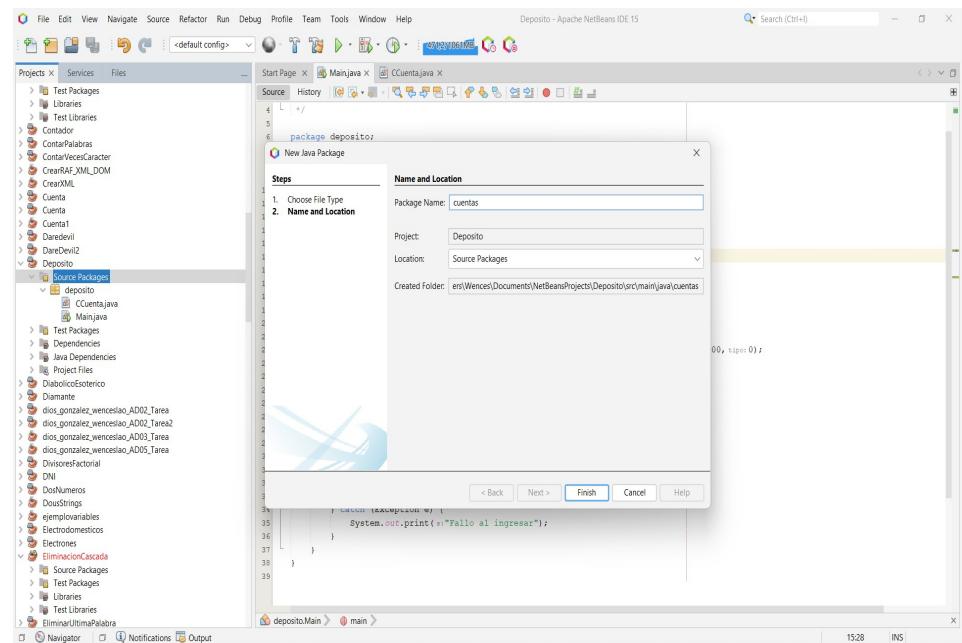


Imagen 4.- Creamos un nuevo paquete llamado **cuentas**

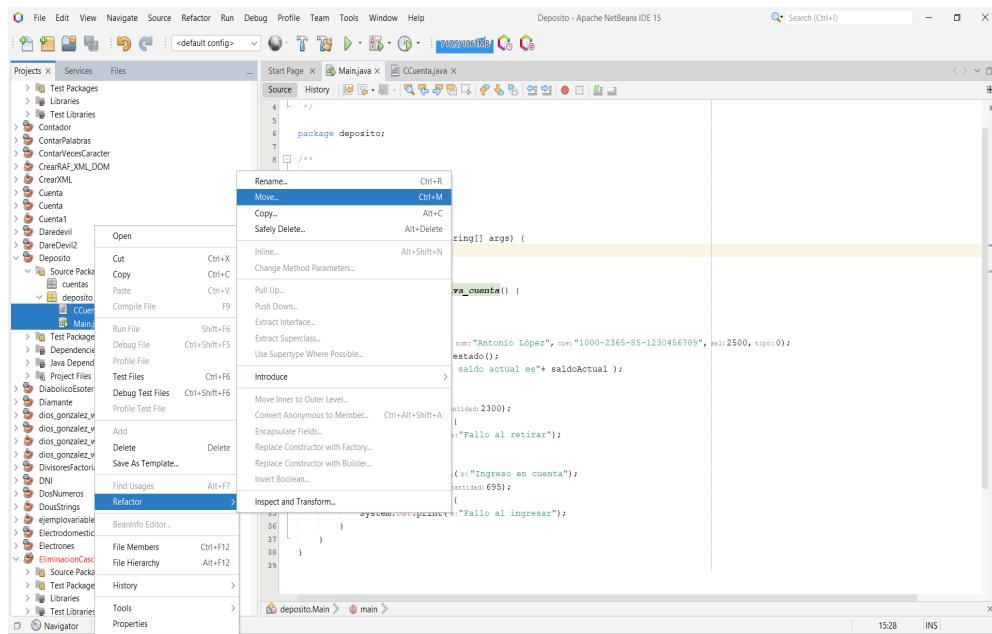


Imagen 5.- Elegimos la opción mover del menú de **refactorización**.

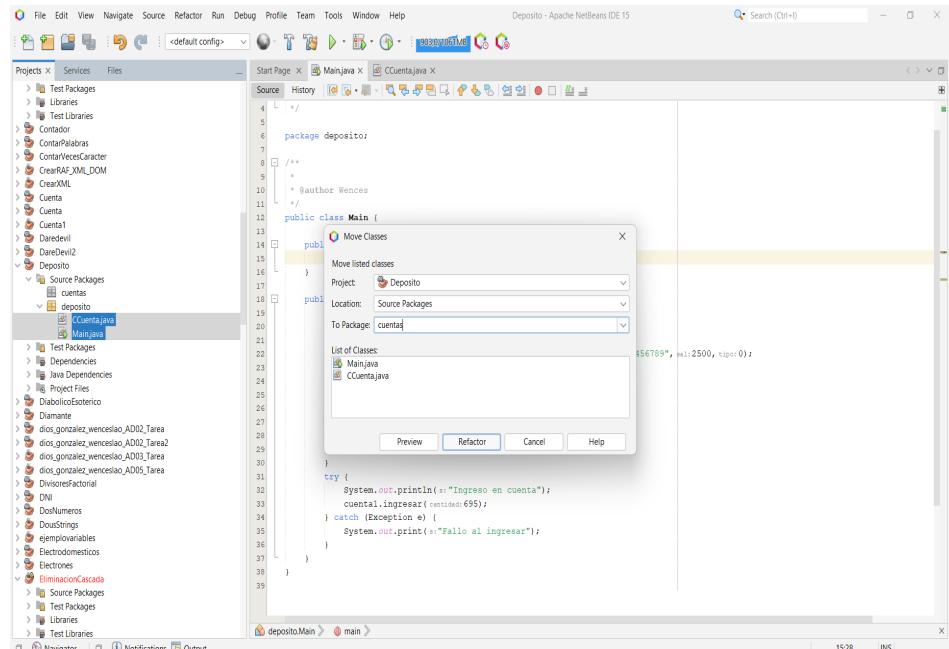


Imagen 6.- Elegimos el paquete al que queremos mover las clases y clicamos en **Refactor**

1.2.- Cambiar el nombre de la variable "miCuenta" por "cuenta1".

Para realizar el cambio de nombre de una variable deberemos seleccionar el nombre de la misma en una de las ocurrencias en el propio código, para luego clicar en el botón derecho del ratón e ir al menú de refactorización para escoger la opción renombrar. Luego escribiremos el nuevo nombre y clicamos en **Refactor**. Es importante tener seleccionado el check que nos aparece en la ventana emergente de que los cambios se apliquen también en los comentarios, así mantendremos la buena legibilidad del código. Todo este proceso se muestra en las imágenes siguientes:

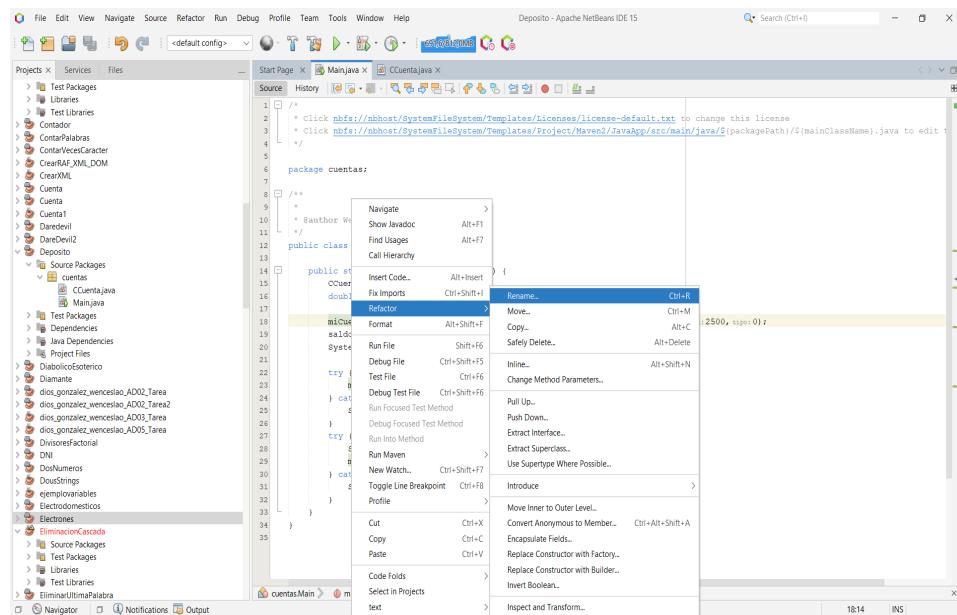


Imagen 7.- Con la variable seleccionada escogemos renombrar.

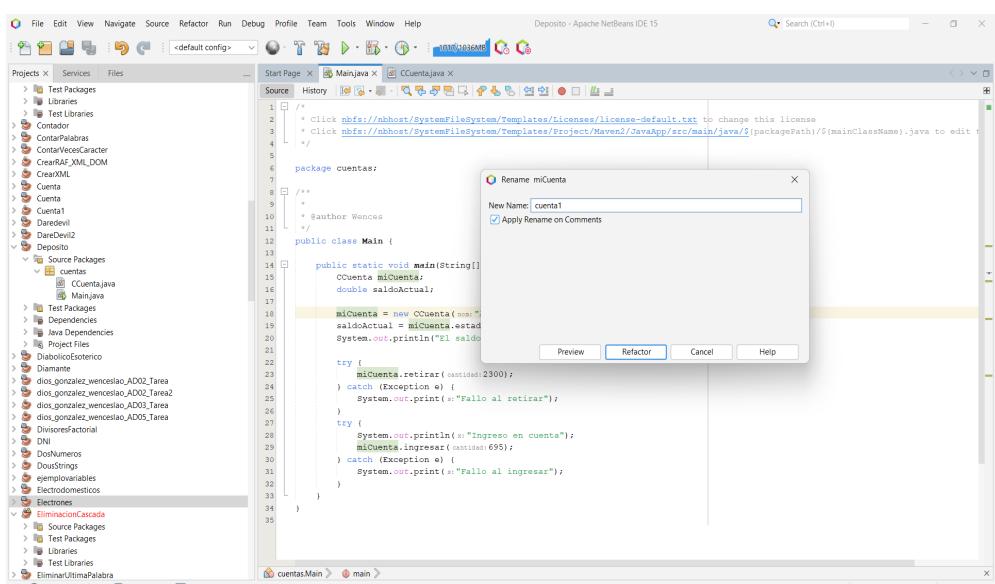


Imagen 8.- Escribimos el nuevo nombre y decimos que cambie también los comentarios

```

1 /*
2  * Click http://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click http://nbhost/SystemFileSystem/Templates/Project/Maven2/JavaApp/src/main/java/ to edit this file
4 */
5
6 package cuentas;
7
8 /**
9  * @author Wences
10 */
11 public class Main {
12
13     public static void main(String[] args) {
14         CCuenta cuenta;
15         double saldoActual;
16
17         cuenta = new CCuenta("Antonio López", "1000-2365-85-1230456789", 2500, 0);
18         saldoActual = cuenta.estado();
19         System.out.println("El saldo actual es " + saldoActual);
20
21         try {
22             cuenta.retirar(2300);
23         } catch (Exception e) {
24             System.out.print("Fallo al retirar");
25         }
26         try {
27             System.out.println("Ingreso en cuenta");
28             cuenta.ingresar(650);
29         } catch (Exception e) {
30             System.out.print("Fallo al ingresar");
31         }
32     }
33 }
34
35

```

Imagen 9.- El cambio se aplica a todas las ocurrencias de la variable.

1.3.-Introducir el método **operativa_cuenta()**, que englobe las sentencias de la clase **Main** que operan con el objeto **cuenta1**.

Para este apartado, tendremos que seleccionar todo el código que queramos introducir en el nuevo método y, de nuevo, clicar sobre la selección con el botón derecho para luego ir a **Refactor** → **Introduce** → **Method..**. Nos saldrá una ventana emergente que nos pedirá el nombre del método y el tipo de acceso que queremos darle, en mi caso será de acceso público. Damos a OK y ya tenemos nuestro nuevo método, tal como se muestra en las siguientes imágenes:

The screenshot shows the Java code editor with the same code as in Image 9. A context menu is open over the variable 'cuenta'. The 'Refactor' menu is expanded, showing the 'Introduce' submenu. The 'Method...' option is highlighted with a blue selection bar.

```

1 /*
2  * Click http://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click http://nbhost/SystemFileSystem/Templates/Project/Maven2/JavaApp/src/main/java/ to edit this file
4 */
5
6 package cuentas;
7
8 /**
9  * @author Wences
10 */
11 public class Main {
12
13     public static void main(String[] args) {
14         CCuenta cuenta;
15         double saldoActual;
16
17         cuenta = new CCuenta("Antonio López", "1000-2365-85-1230456789", 2500, 0);
18         saldoActual = cuenta.estado();
19         System.out.println("El saldo actual es " + saldoActual);
20
21         try {
22             cuenta.retirar(2300);
23         } catch (Exception e) {
24             System.out.print("Fallo al retirar");
25         }
26         try {
27             System.out.println("Ingreso en cuenta");
28             cuenta.ingresar(650);
29         } catch (Exception e) {
30             System.out.print("Fallo al ingresar");
31         }
32     }
33 }
34
35

```

Imagen 10.- Vamos a **Refactor** → **Introduce** → **Method..**

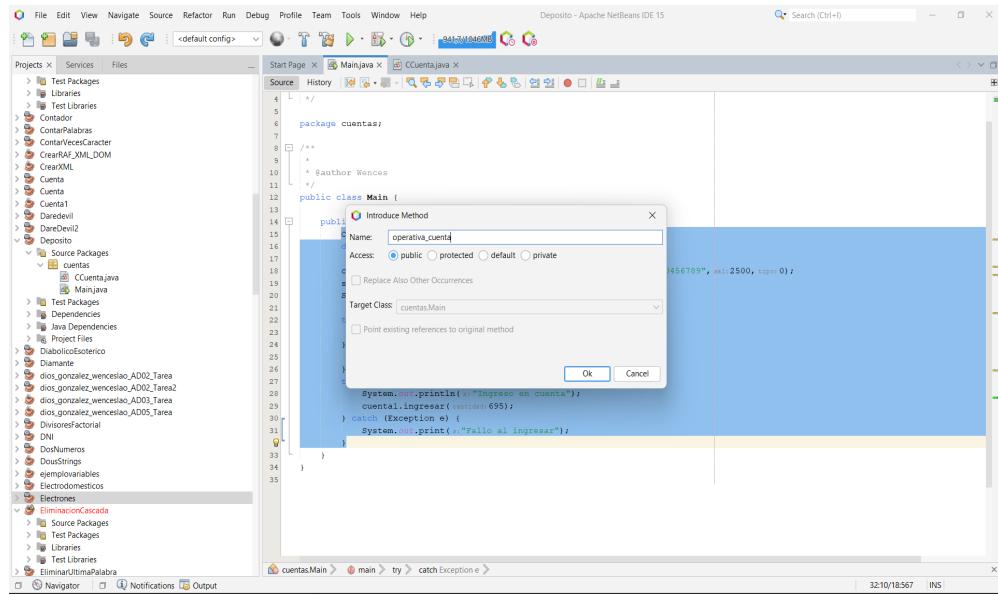


Imagen 11.- Damos nombre y tipo de acceso al nuevo método.

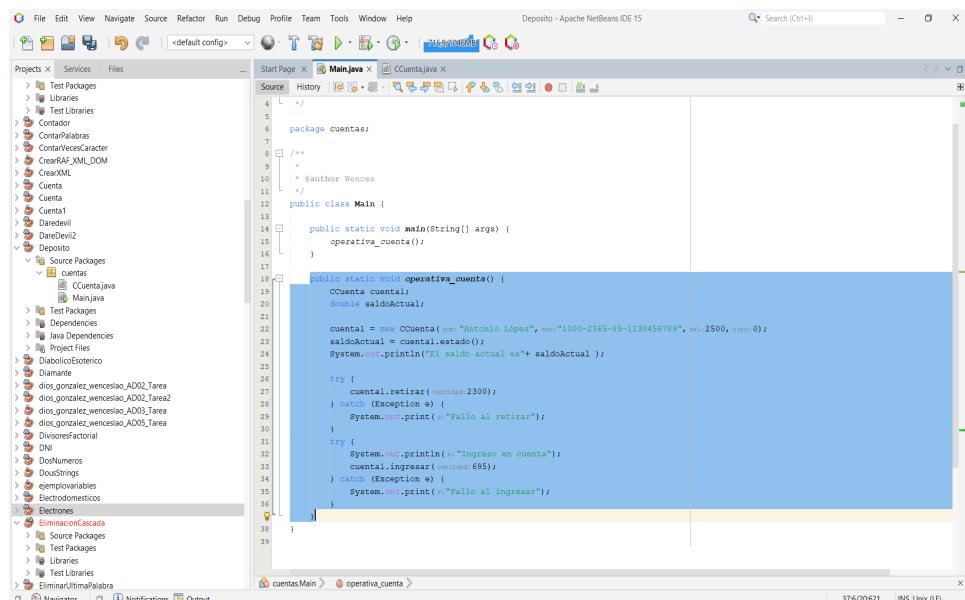


Imagen 12.- Ya tenemos el nuevo método en nuestro código.

1.4.-Encapsular los atributos de la clase CCuenta.

Para este apartado, clicamos botón derecho encima de la clase Cuenta y vamos a **Refactor → Encapsulate fields..**. En la ventana emergente que nos sale escogemos las variables que queremos encapsular y si queremos crear **Getter**, **Setter** o ambos. En mi caso clicaré en **Select All** porque quiero crear **Getter** y **Setter** de todas los atributos de la clase. Se muestra el proceso en las imágenes que vienen a continuación.

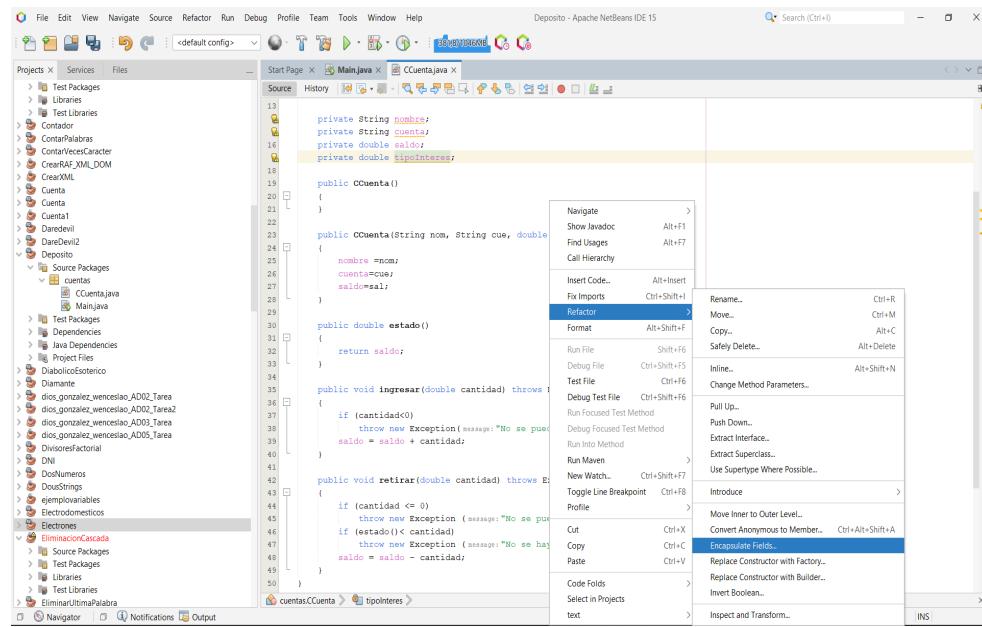


Imagen 13.- Elegimos **Refactor → Encapsulate fields..**

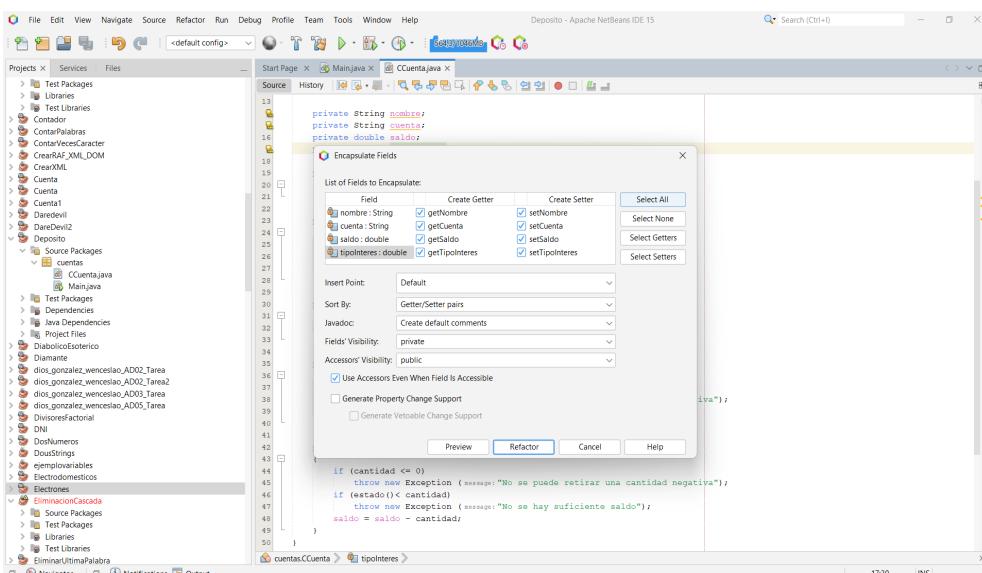


Imagen 14.- clicamos en **Select All** y luego en **Refactor**

1.5.-Añadir un nuevo parámetro al método `operativa_cuenta()`, de nombre `cantidad` y de tipo `float`.

Para introducir un nuevo parámetro en el método `operativa_cuenta()` seleccionamos el nombre de nuestro método y vamos a **Refactor** → **Change Method Parameters..** Nos sale una ventana donde clicaremos en Add para añadir un nuevo parámetro, donde escogeremos el tipo, el nombre y el valor por defecto del parámetro.Una vez hecho esto clicamos en **Refactor**. Las siguientes capturas muestran los pasos a seguir:

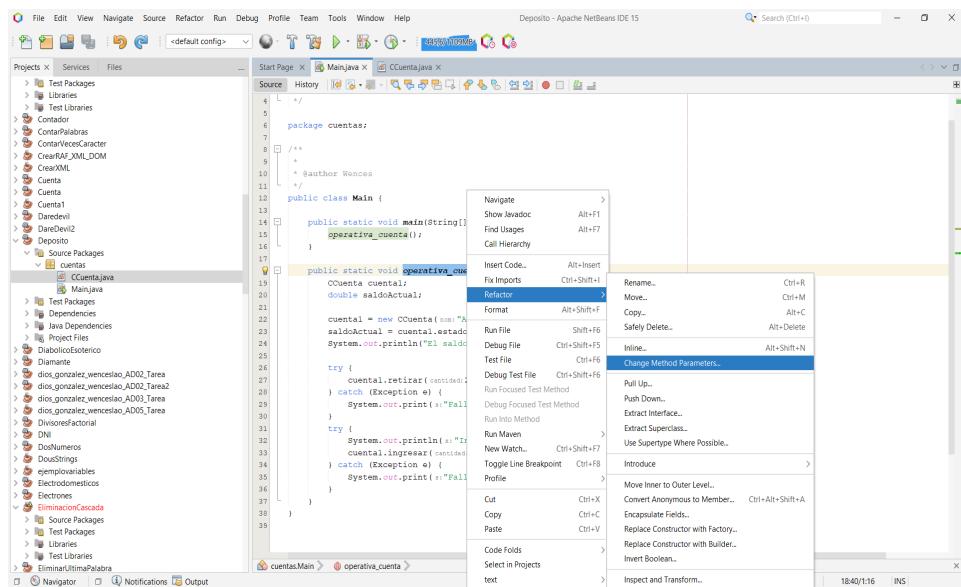


Imagen 15.- Vamos a Refactor → Change Method Parameters..

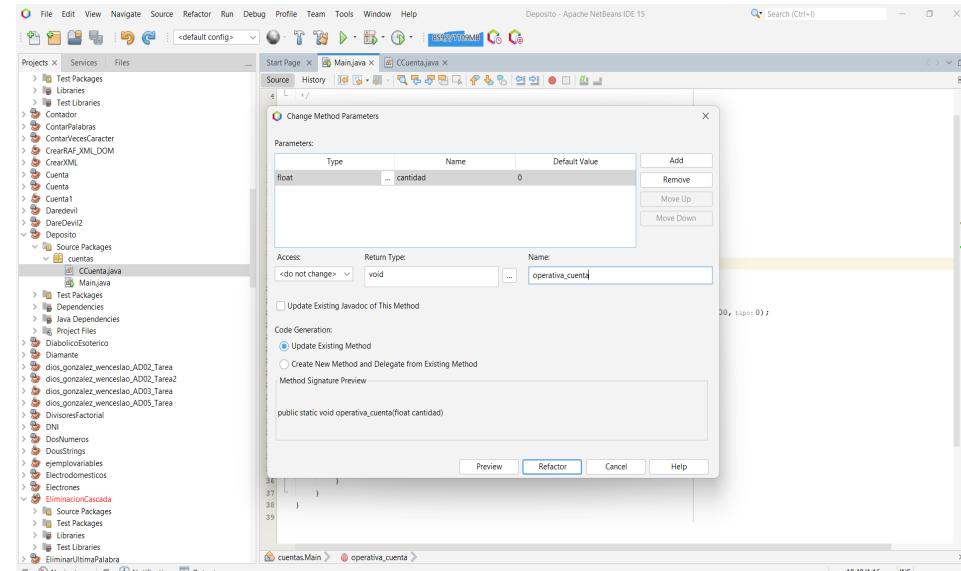


Imagen 16.- Configuramos el nuevo parámetro `cantidad`.

2.- GIT

2.1.-Configurar GIT para el proyecto. Crear un repositorio público en GitHub.

A continuación mostraremos como inicializar **GIT** para nuestro proyecto y como crear un repositorio público en **GitHub**. Para el primer ejercicio clicamos botón derecho en el proyecto y vamos a **Versioning → Initialize Git Repository..**. Escogemos la ruta donde queremos guardar el repositorio y damos a OK. Para el segundo ejercicio tenemos que ir a la página **Web de GitHub**, registrarnos y crear un nuevo repositorio público.

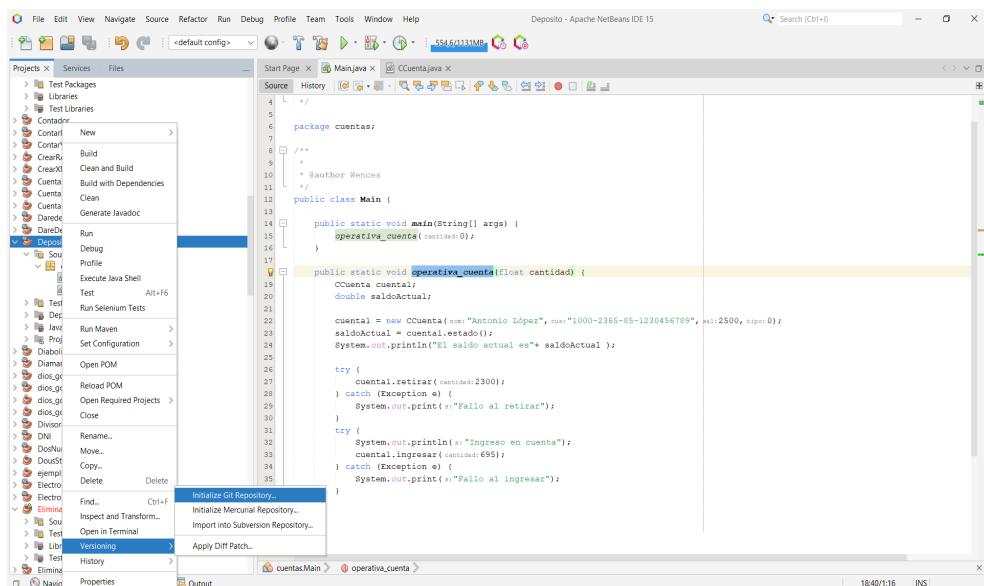


Imagen 17.- Vamos a **Versioning → Initialize Git Repository..**

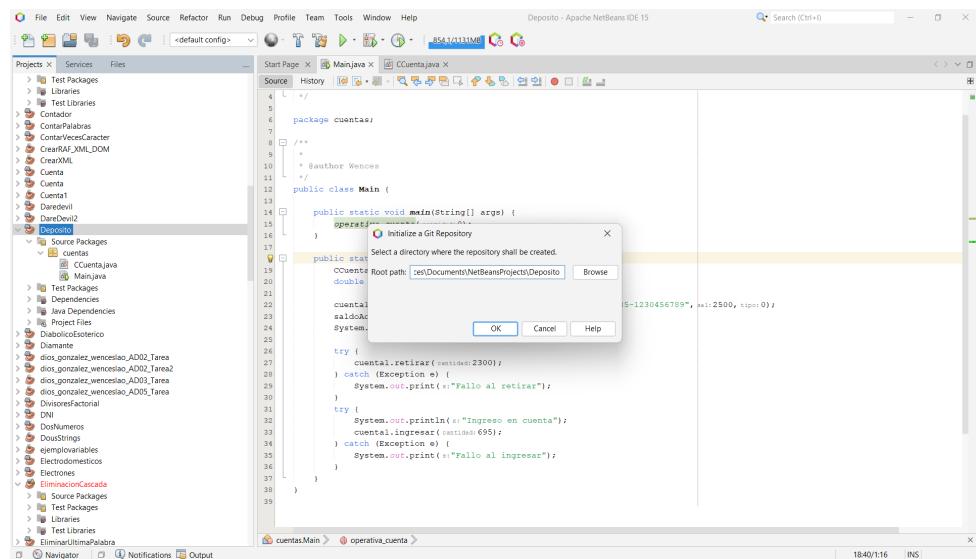


Imagen 18.- Seleccionamos la ruta y damos a OK.

```

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Deposito - Apache NetBeans IDE 15
Search (Ctrl+F) 1158/1171 files
Projects Services Files <default config>
Start Page X Main.java X CCuenta.java X
Source History ...
package cuentas;
/*
 * @author Wences
 */
public class Main {
    public static void main(String[] args) {
        operativa_cuentas();
    }
    public static void operativa_cuentas(float cantidad) {
        CCuenta cuenta;
        double saldoActual;
        cuenta = new CCuenta("Antonio López", "1000-2365-85-1230456789", 12500, tipo);
        saldoActual = cuenta.estado();
        System.out.println("El saldo actual es" + saldoActual);

        try {
            cuenta.retirar(cantidad);
        } catch (Exception e) {
            System.out.print("Falló al retirar");
        }
    }
}

```

Output - Deposito - C:\Users\Wences\Documents\NetBeansProjects\Deposito

```

==[IRB]== 27 feb. 2023 20:25:45 Initializing ...
Initializing repository
Creating git C:\Users\Wences\Documents\NetBeansProjects\Deposito/.git directory
git init C:\Users\Wences\Documents\NetBeansProjects\Deposito
==[IRB]== 27 feb. 2023 20:25:47 Initializing ... finished.

```

Imagen 19.- Repositorio inicializado...

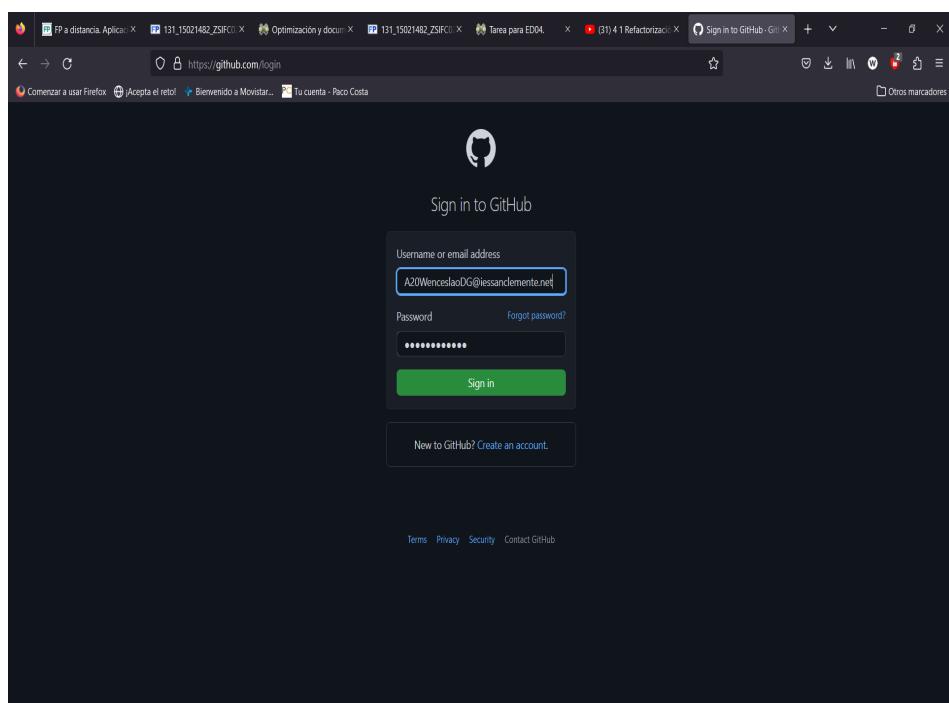


Imagen 20.- Iniciamos sesión o creamos cuenta en GitHub.

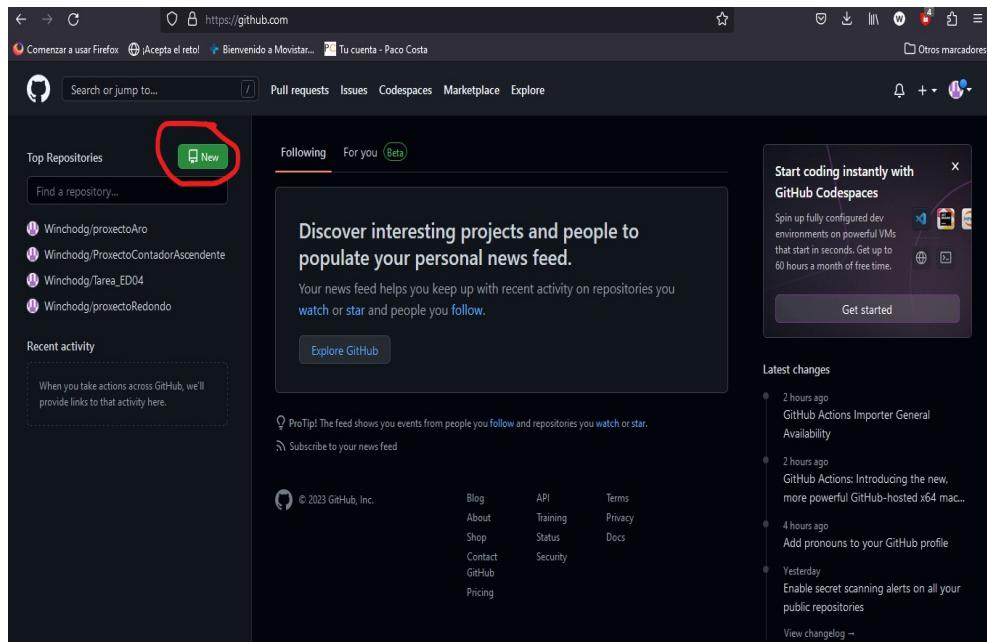


Imagen 21.- Damos clic en New

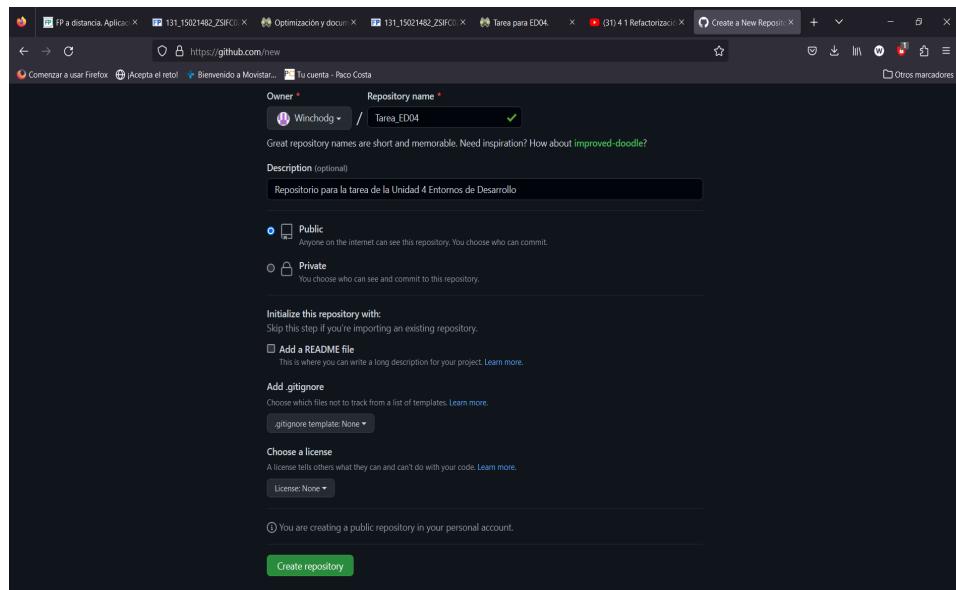


Imagen 22.- Le ponemos un nombre, seleccionamos Public y damos a Create Repository.

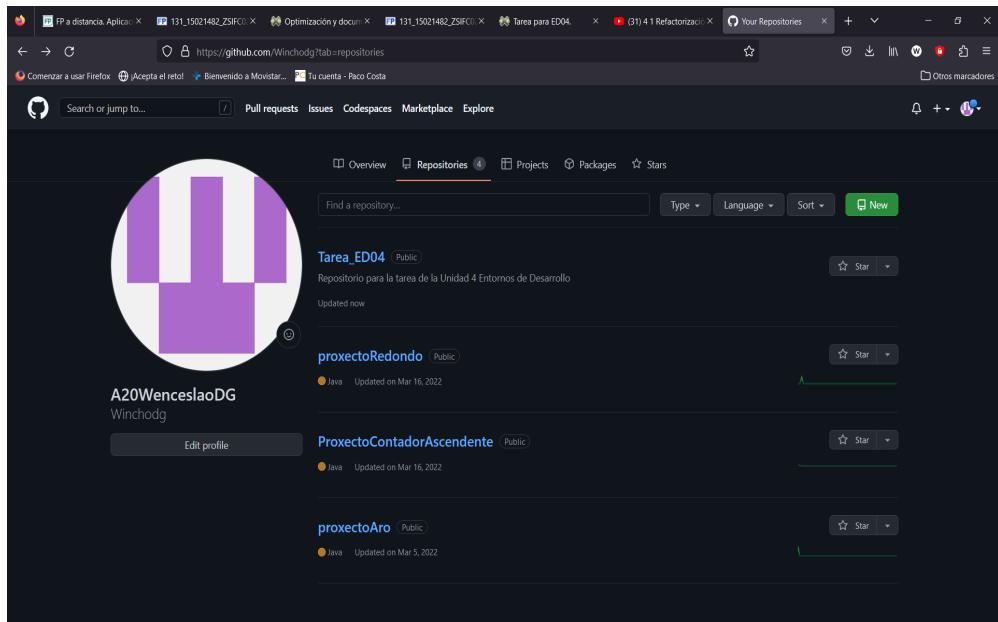


Imagen 23.- Repositorio creado.

2.2.-Realizar, al menos, una operación commit.Comentando el resultado de la ejecución.

Para este apartado deberemos clicar con botón derecho encima de la clase o clases sobre las que queremos hacer el commit e ir a **Git → Commit..**. En la ventana emergente tendremos un espacio donde pondremos el comentario que queremos asociar al **commit**. A continuación se muestra en las capturas de pantalla.

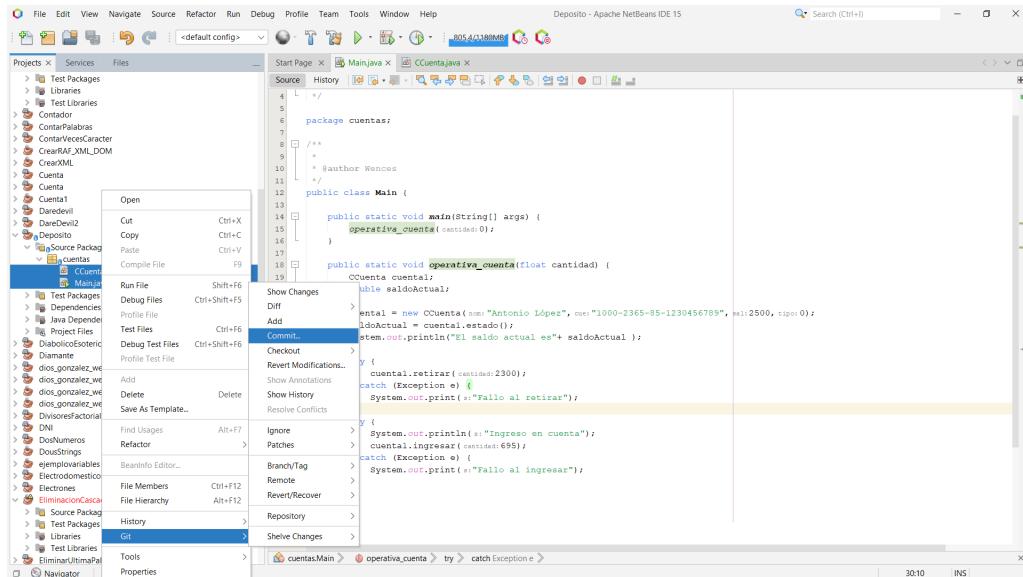


Imagen 24.- Vamos a **Git → Commit..**

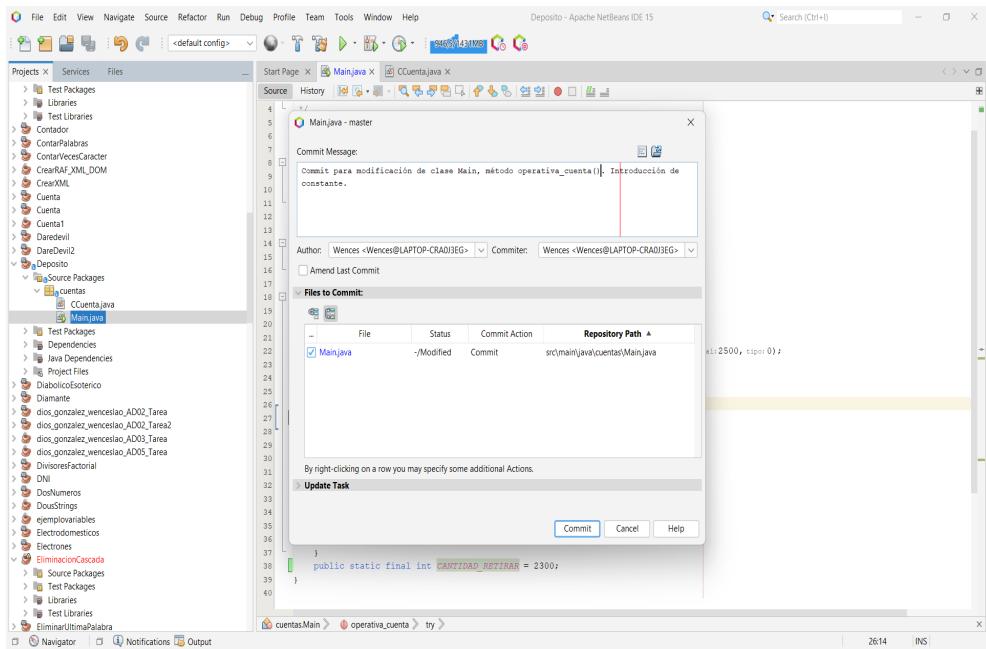


Imagen 25.- Introducimos el comentario y clicamos en **Commit**.

```

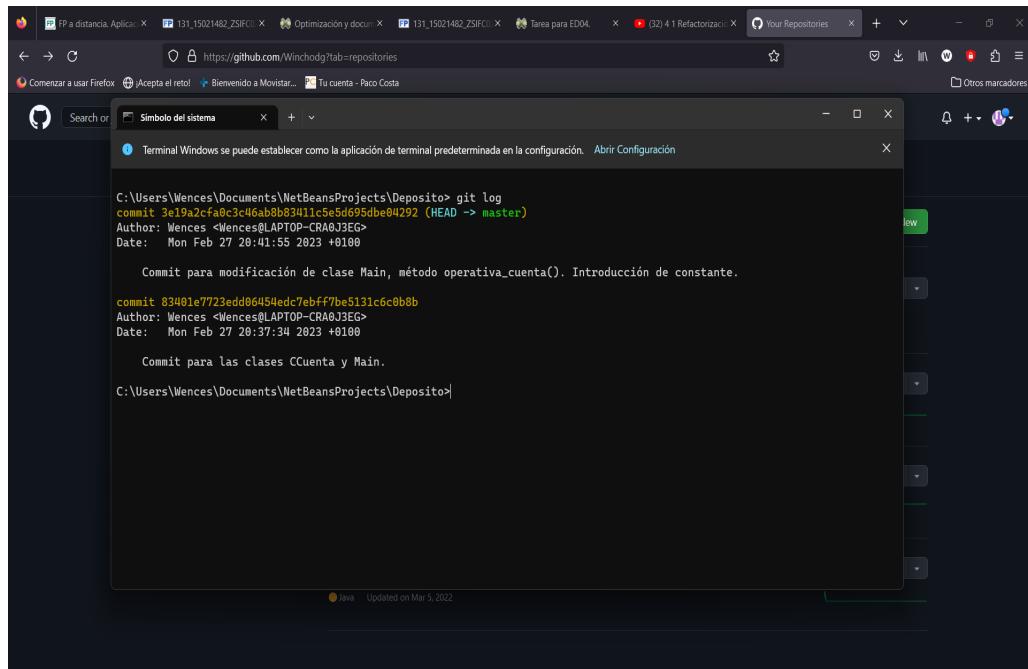
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help Deposito - Apache NetBeans IDE 15 Search (Ctrl+F) ...
Projects Services Files Start Page X Main.java X CCuenta.java X
Source History ...
Main.java - master
Commit Message:
Commit para modificación de clase Main, método operativa_cuenta(). Introducción de constante.
Author: Wences <Wences@LAPTOP-CRA0J3EG> Committer: Wences <Wences@LAPTOP-CRA0J3EG>
Amend Last Commit
File to Commit:
Main.java -/Modified Commit src/main/java/cuentas/Main.java
By right-clicking on a row you may specify some additional Actions.
Update Task
Commit Cancel Help
2500, tipo:0);
public static final int CANTIDAD_RETIRAR = 2300;
}
public static void main(String[] args) {
    operativa_cuenta(cantidad);
}
public static void operativa_cuenta(float cantidad) {
    CCuenta cuenta;
    double saldoActual;
    cuenta = new CCuenta("Antonio López", "1000-2365-85-1230456789", sal:2500, tipo:0);
    saldoActual = cuenta.estado();
    System.out.println("El saldo actual es "+ saldoActual);
}

```

Imagen 26.- Nos muestra por la salida que el commit se realizó con éxito.

2.3.-Mostrar el historial de versiones para el proyecto mediante un comando desde consola.

En este apartado mostremos por consola el historial de versiones de nuestro proyecto. Se haría de la siguiente manera:



The screenshot shows a terminal window titled "Símbolo del sistema" (Symbol of the system) running on a Windows operating system. The window displays the command "git log" being run in the directory "C:\Users\Wences\Documents\NetBeansProjects\Deposito". The terminal output shows two commits:

```
C:\Users\Wences\Documents\NetBeansProjects\Deposito> git log
commit 3e19a2cfa0c3c46a0b083411c5e5d695dbe04292 (HEAD -> master)
Author: Wences <Wences@LAPTOP-CRA0A3JEG>
Date:   Mon Feb 27 20:41:55 2023 +0100

    Commit para modificación de clase Main, método operativa_cuenta(). Introducción de constante.

commit 83401e7723ed0d064545edc7ebff7be5131c6c0b8b
Author: Wences <Wences@LAPTOP-CRA0A3JEG>
Date:   Mon Feb 27 20:37:34 2023 +0100

    Commit para las clases CCuenta y Main.

C:\Users\Wences\Documents\NetBeansProjects\Deposito>
```

Imagen 27.- Mostramos historial por la consola de comandos.

3- JAVADOC

3.1.-Insertar comentarios JavaDoc en la clase CCuenta.

Antes de generar la documentación **JavaDoc** insertaremos diversos comentarios para hacer el código más fácil de interpretar. Comentarios en métodos, etiquetando parámetros y valores de retorno y explicando para qué sirven. Etiquetas de autor, de versión y distintos comentarios aclarativos en distintas partes del código.

3.2.-Generar documentación JavaDoc para todo el proyecto y comprobar que abarque todos los métodos y atributos de la clase CCuenta.

Para generar JavaDoc desde Eclipse, tenemos que ir a **Project → Generate JavaDoc..**. En la ventana emergente seleccionamos nuestro proyecto y clicamos en **Private** para que genere documentación de todas las classes y miembros de del mismo. También podemos seleccionar la ruta donde nos guardará la documentación, que será en forma de archivos **HTML**, para visualizar como páginas web. Las siguientes imágenes muestran el proceso:

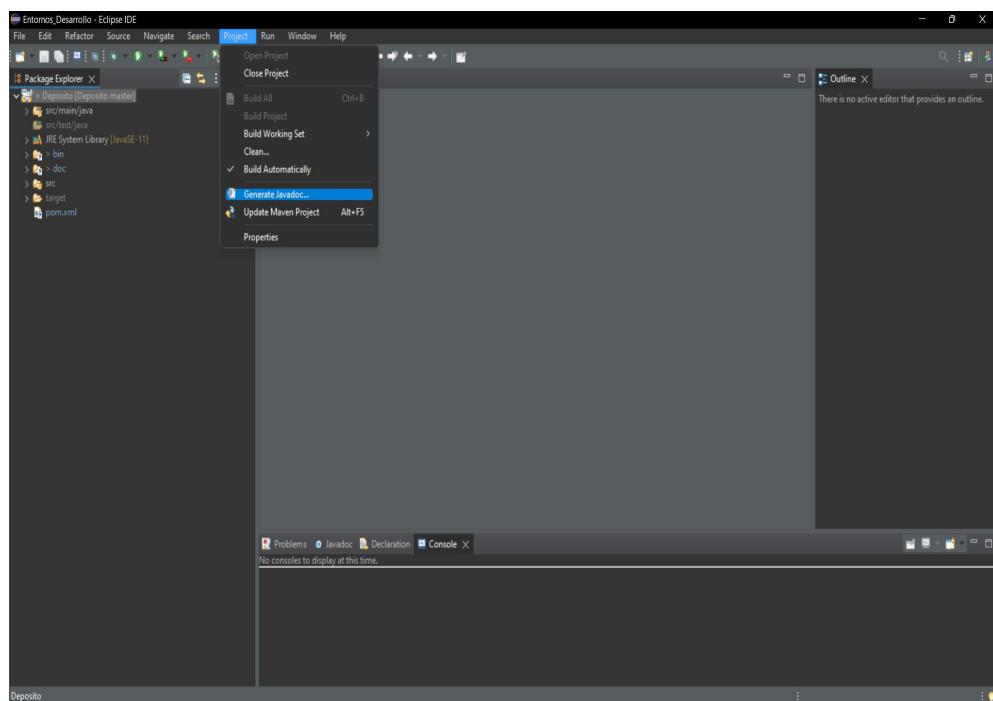


Imagen 28.- Vamos a **Project → Generate JavaDoc..**

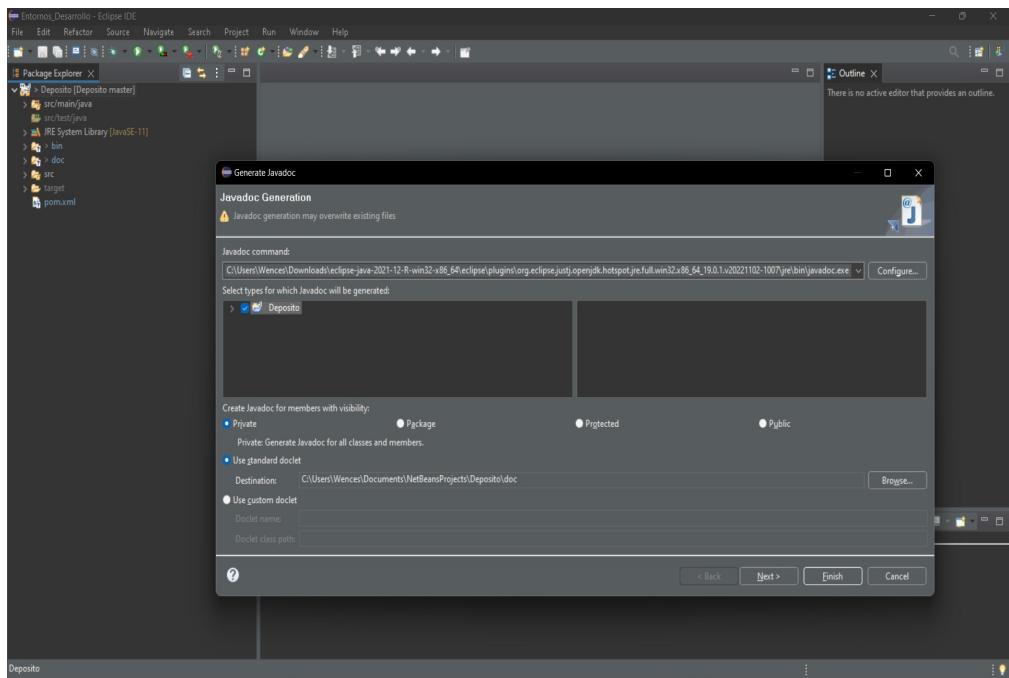


Imagen 29.- Seleccionamos el proyecto y clicamos en **Private**.

NOTA: Para este apartado utilicé el IDE Eclipse ya que Netbeans no me incluía los atributos de la clase Cuenta en la documentación.

Por último y para finalizar la práctica, dejo el enlace de mi repositorio de **GitHub**, al que llamé **Tarea_ED04**, donde está, tanto el proyecto Java **Deposito**, así como este documento explicativo.

https://github.com/Winchodg/Tarea_ED04