

Alvarez_Valencia_Gianfranco_ED04_Tarea

Índice

1.	Refactorización.....	3
1.1.	Refactorización 1.....	3
1.2.	Refactorización 2.....	4
1.3.	Refactorización 3.....	5
1.4.	Refactorización 4.....	8
2.	Git.....	9
2.5.	Git 5.....	9
2.6.	Git 6.....	11
2.7.	Git 7.....	13
3.	JAVADOC	14
3.8.	JAVADOC 8	14
3.9.	JAVADOC 9	16
3.10.	JAVADOC 10	18

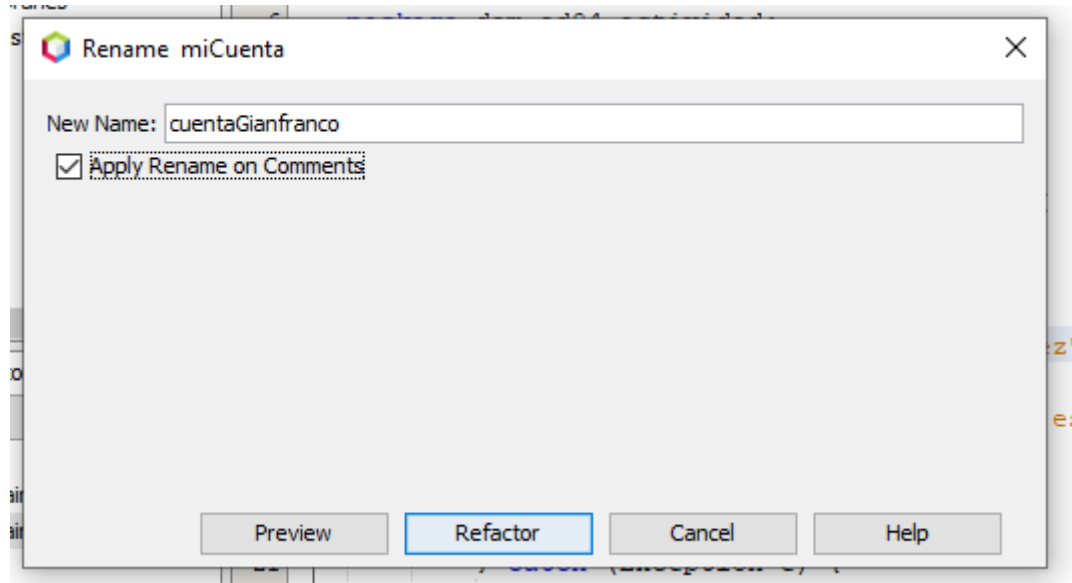
1. Refactorización

Debes refactorizar el código de la Clase **Main**, utilizando las herramientas de reestructuración de código de NetBeans:

1.1. Refactorización 1

Cambia el nombre de la variable "**miCuenta**" por "**cuentaTUNOMBRE**".

Seleccionar **miCuenta** en la clase y desde el menú **Refactor>Rename**:



Resultado:

```
~ and open the template in the editor.
 */

package dam_ed04_actividad;

public class Main {

    public static void main(String[] args) {
        CCuenta cuentaGianfranco;
        double saldoActual;

        cuentaGianfranco = new CCuenta("Antonio", 1000, 1, 0);
        saldoActual = cuentaGianfranco.estado();
        System.out.println("El saldo actual es " + saldoActual);

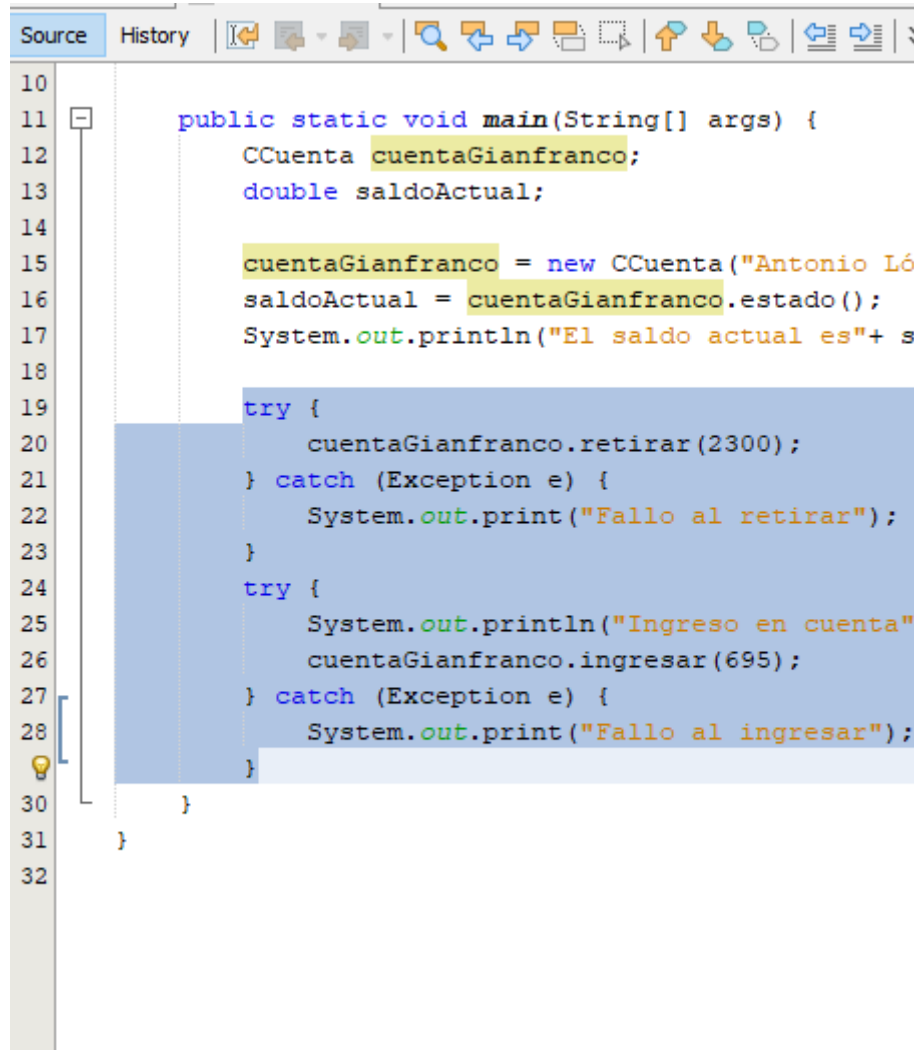
        try {
            cuentaGianfranco.retirar(2300);
        } catch (Exception e) {
            System.out.print("Fallo al retirar");
        }

        try {
            System.out.println("Ingreso en cuenta");
            cuentaGianfranco.ingresar(695);
        } catch (Exception e) {
            System.out.print("Fallo al ingresar");
        }
    }
}
```

1.2. Refactorización 2

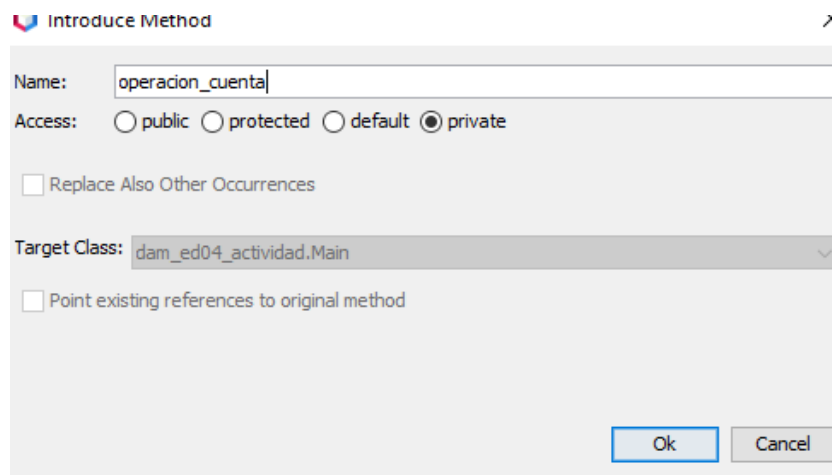
Introduce el método `operacion_cuenta`, que englobe las sentencias de la clase `Main` que operan con el objeto `cuentaTUNOMBRE`.

Seleccionar el código:



```
10
11 public static void main(String[] args) {
12     CCuenta cuentaGianfranco;
13     double saldoActual;
14
15     cuentaGianfranco = new CCuenta("Antonio López", 1000, 1, 0);
16     saldoActual = cuentaGianfranco.estado();
17     System.out.println("El saldo actual es"+ saldoActual);
18
19     try {
20         cuentaGianfranco.retirar(2300);
21     } catch (Exception e) {
22         System.out.print("Fallo al retirar");
23     }
24     try {
25         System.out.println("Ingreso en cuenta");
26         cuentaGianfranco.ingresar(695);
27     } catch (Exception e) {
28         System.out.print("Fallo al ingresar");
29     }
30 }
31 }
32
```

Desde el menú Refactor>Introduce>Method



Introduce Method

Name:

Access: ☐ public ☐ protected ☐ default ☒ private

☐ Replace Also Other Occurrences

Target Class:

☐ Point existing references to original method

Ok Cancel

Resultado:

```
private static void operacion_cuenta(CCuenta cuentaGianfranco) {  
    try {  
        cuentaGianfranco.retirar(2300);  
    } catch (Exception e) {  
        System.out.print("Fallo al retirar");  
    }  
    try {  
        System.out.println("Ingreso en cuenta");  
        cuentaGianfranco.ingresar(695);  
    } catch (Exception e) {  
        System.out.print("Fallo al ingresar");  
    }  
}
```

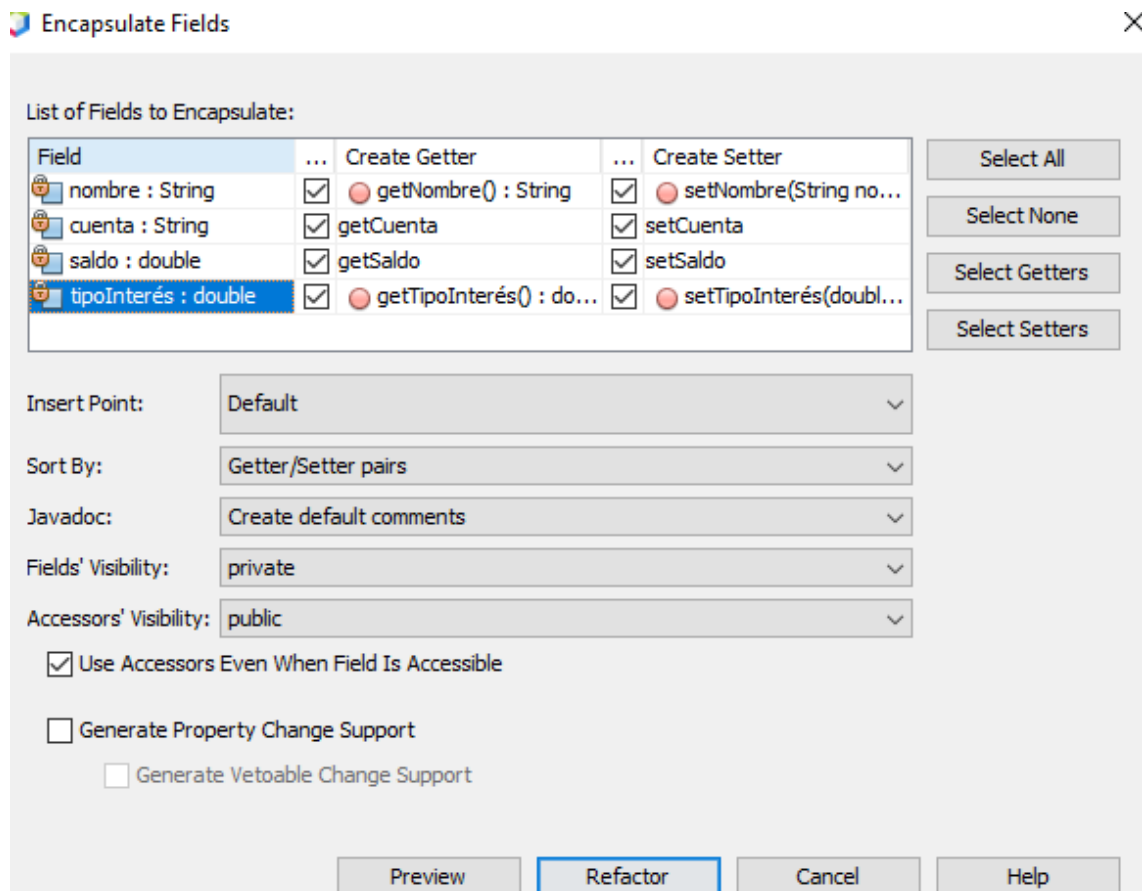
1.3. Refactorización 3

Encapsula los cuatro atributos de la clase **CCuenta**, creando pares de métodos getter/setter de acceso público.

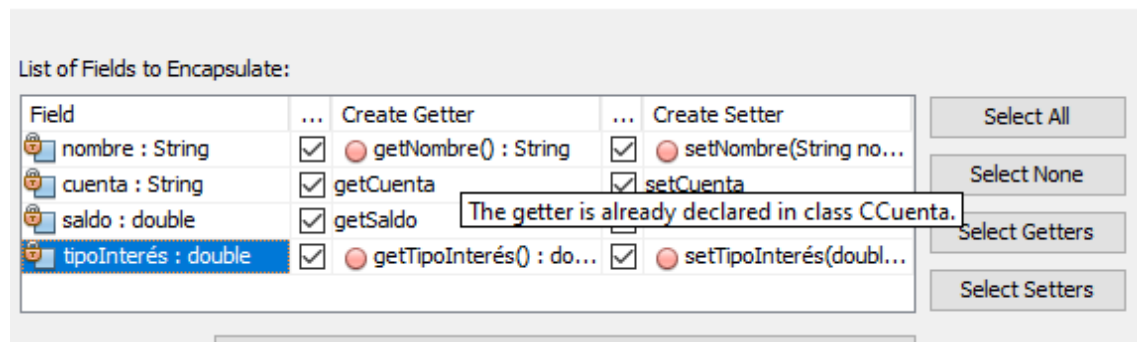
Los atributos a encapsular:

```
private String nombre;  
private String cuenta;  
private double saldo;  
private double tipoInterés;
```

Desde el menú Refactor>Encapsulate Fields:



El punto rojo nos dice que ya están definidos esos getters o setters:



Resultado, se crean los métodos:

```
/**
 * @return the cuenta
 */
public String getCuenta() {
    return cuenta;
}

/**
 * @param cuenta the cuenta to set
 */
public void setCuenta(String cuenta) {
    this.cuenta = cuenta;
}

/**
 * @return the saldo
 */
public double getSaldo() {
    return saldo;
}

/**
 * @param saldo the saldo to set
 */
public void setSaldo(double saldo) {
    this.saldo = saldo;
}
```

1.4. Refactorización 4

Añade un nuevo parámetro al método `operacion_cuenta`, de nombre **importe** y de tipo `float`.

Desde el menú Refactor>Change Method Parameters:

Change Method Parameters

Parameters:

Type	Name	Default Value
CCuenta	cuentaGianfranco	

Access: <do not change> Return Type: void Name: operacion_cuenta

☐ Update Existing Javadoc of This Method

Code Generation:

☒ Update Existing Method
☐ Create New Method and Delegate from Existing Method

Method Signature Preview

```
private static void operacion_cuenta(CCuenta cuentaGianfranco)
```

Buttons: Preview, Refactor, Cancel, Help

Add y rellenar con las especificaciones del parámetro:

Change Method Parameters

Parameters:

Type	Name	Default Value
CCuenta	cuentaGianfranco	
float	importe	null

Access: <do not change> Return Type: void Name: operacion_cuenta

☐ Update Existing Javadoc of This Method

Code Generation:

☒ Update Existing Method
☐ Create New Method and Delegate from Existing Method

Method Signature Preview

```
private static void operacion_cuenta(CCuenta cuentaGianfranco, float importe)
```

Buttons: Preview, Refactor, Cancel, Help

Resultado después de añadir el parámetro:

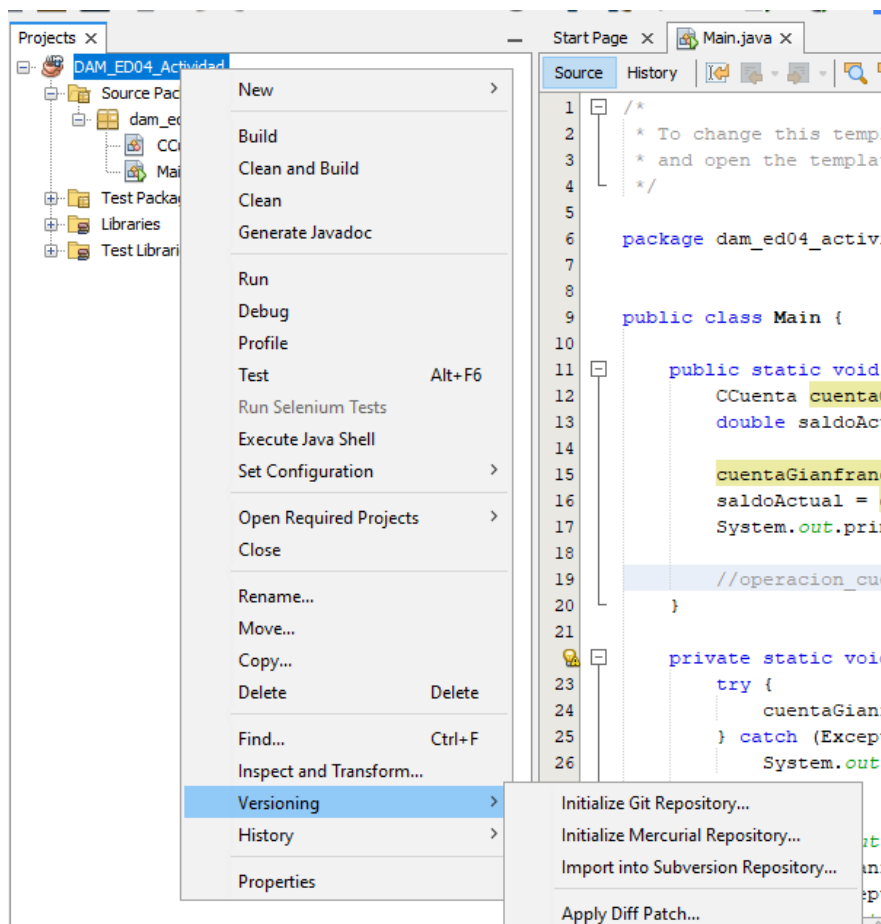
```
private static void operacion_cuenta(CCuenta cuentaGianfranco, float importe) {
    try {
        cuentaGianfranco.retirar(2300);
    } catch (Exception e) {
        System.out.print("Fallo al retirar");
    }
    try {
        System.out.println("Ingreso en cuenta");
        cuentaGianfranco.ingresar(695);
    } catch (Exception e) {
        System.out.print("Fallo al ingresar");
    }
}
```

2. Git

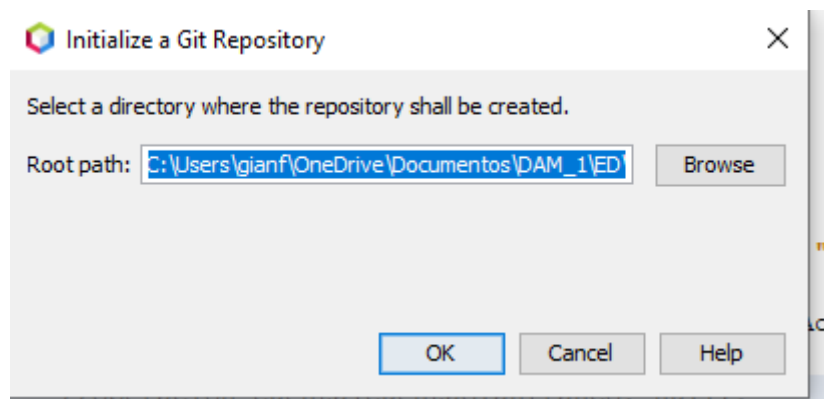
2.5. Git 5

Inicializa un repositorio de Git para el proyecto DAW_ED04_Actividad en la carpeta por defecto del proyecto.

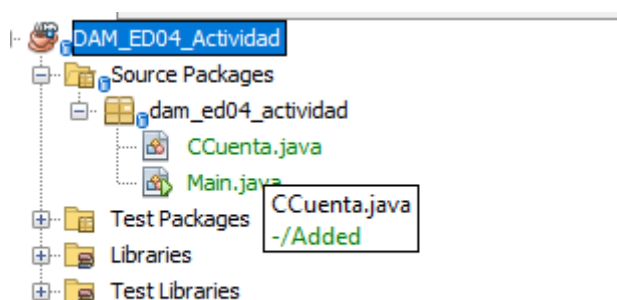
Inicializar un nuevo Git desde el proyecto click derecho>vesioning>Inizialize Git repository:



En la carpeta que está el proyecto:



Ahora aparece el estado en el Git del fichero en NetBeans:



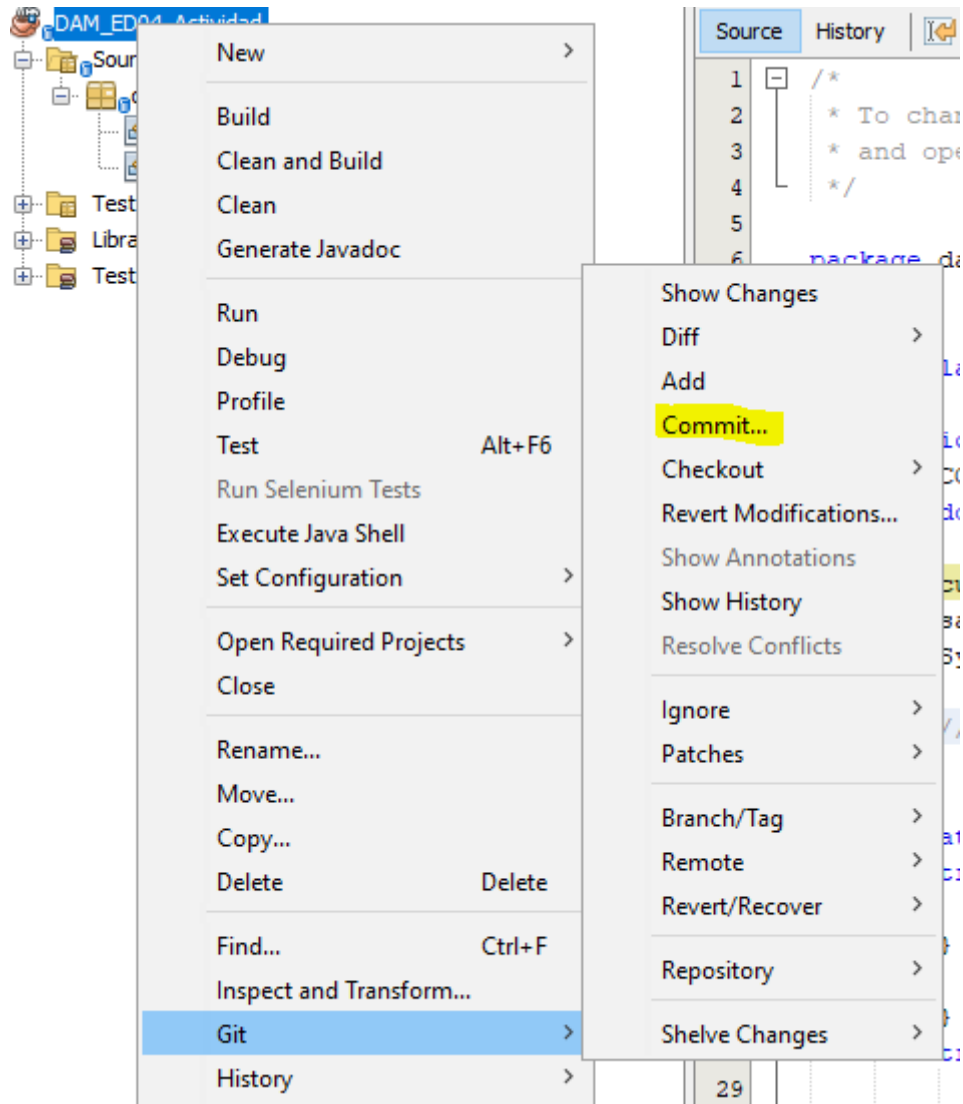
Y se habrá creado la carpeta (oculta) del Git:

OneDrive - Personal > Documentos > DAM_1 > ED > T4 > DAM_ED04_Actividad				
Nombre	Estado	Fecha de modificación	Tipo	
.git	✓	23/02/2022 19:11	Carpeta de archivos	
build	✓	23/02/2022 19:07	Carpeta de archivos	
dist	✓	23/02/2022 19:07	Carpeta de archivos	
nbproject	✓	22/02/2022 16:17	Carpeta de archivos	
src	✓	04/02/2019 9:45	Carpeta de archivos	
test	✓	12/06/2011 0:22	Carpeta de archivos	
build.xml	✓	22/02/2022 16:17	Documento XML	
manifest.mf	✓	12/06/2011 0:22	Archivo MF	


2.6. Git 6



Realiza, al menos, una operación **commit**, con el comentario "Commit inicial".

Desde el proyecto>click derecho>git>commit:



Commit inicial:



 DAM_ED04_Actividad - (no branch) ✕

Commit Message:  

Commit inicial

Author: v Committer: v

☐ Amend Last Commit


Files to Commit:  


...	File	Status	Commit Action	Repository Path ▲
<input checked="" type="checkbox"/>	build.xml	-/Added	Commit	build.xml
<input checked="" type="checkbox"/>	manifest.mf	-/Added	Commit	manifest.mf
<input checked="" type="checkbox"/>	build-impl.xml	-/Added	Commit	nbproject\build-impl.xml
<input checked="" type="checkbox"/>	genfiles.properties	-/Added	Commit	nbproject\genfiles.properties
<input checked="" type="checkbox"/>	project.properties	-/Added	Commit	nbproject\project.properties
<input checked="" type="checkbox"/>	project.xml	-/Added	Commit	nbproject\project.xml
<input checked="" type="checkbox"/>	CCuenta.java	-/Added	Commit	src\dam_ed04_actividad\CCuenta.java
<input checked="" type="checkbox"/>	Main.java	-/Added	Commit	src\dam_ed04_actividad\Main.java

By right-clicking on a row you may specify some additional Actions.

+ Update Task

Pregunta por el autor:

 Set Repository User ✕

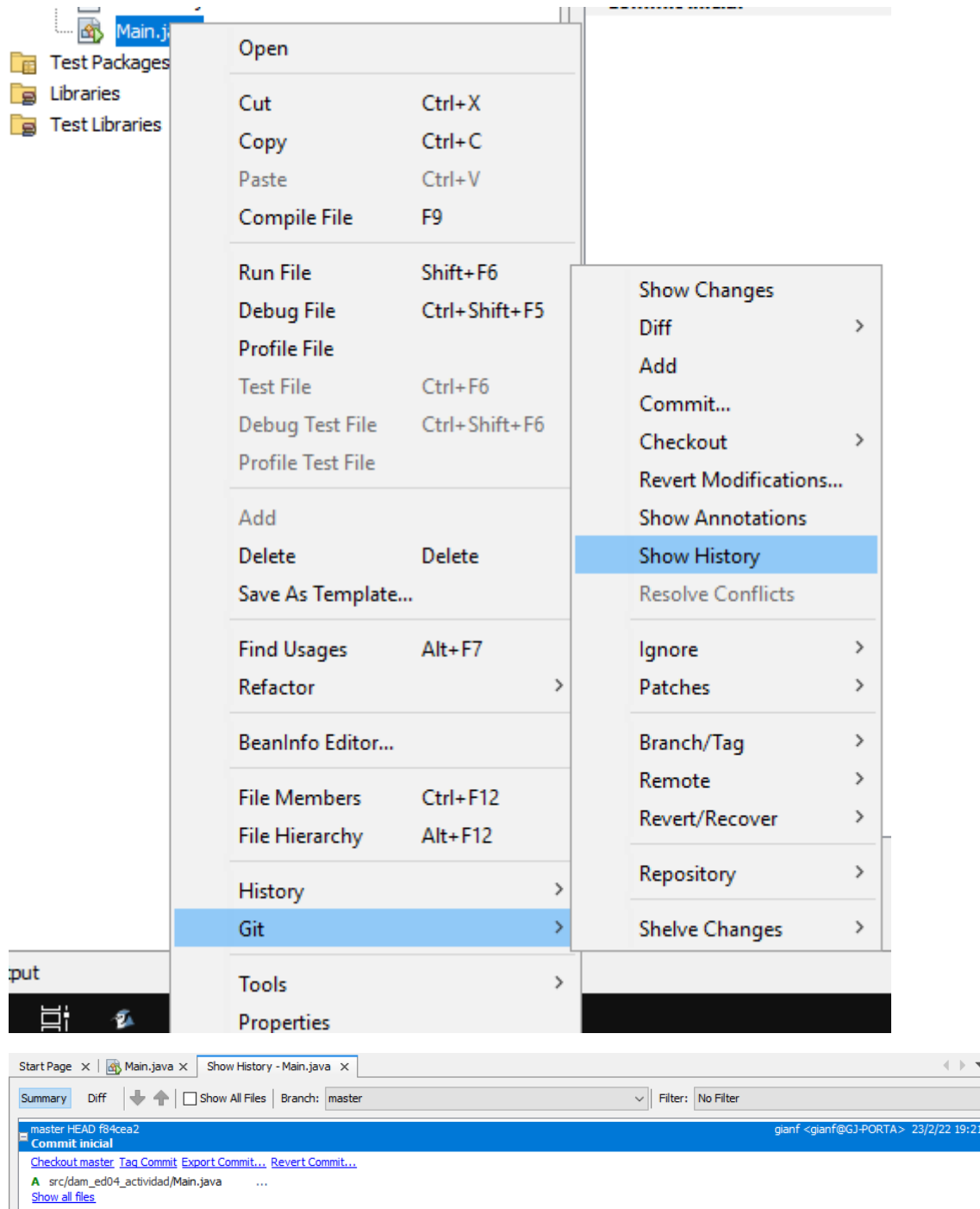
 Repository does not have fully specified user yet.

Do you want to set gianf <gianf@GJ-PORTA> as the default author?

2.7. Git 7

Muestra el historial del versiones para el fichero `Main.java`.

Desde el menú del proyecto>clase `Main.java`>> Git>Show History se muestra el historial de versiones del fichero `Main.java`:

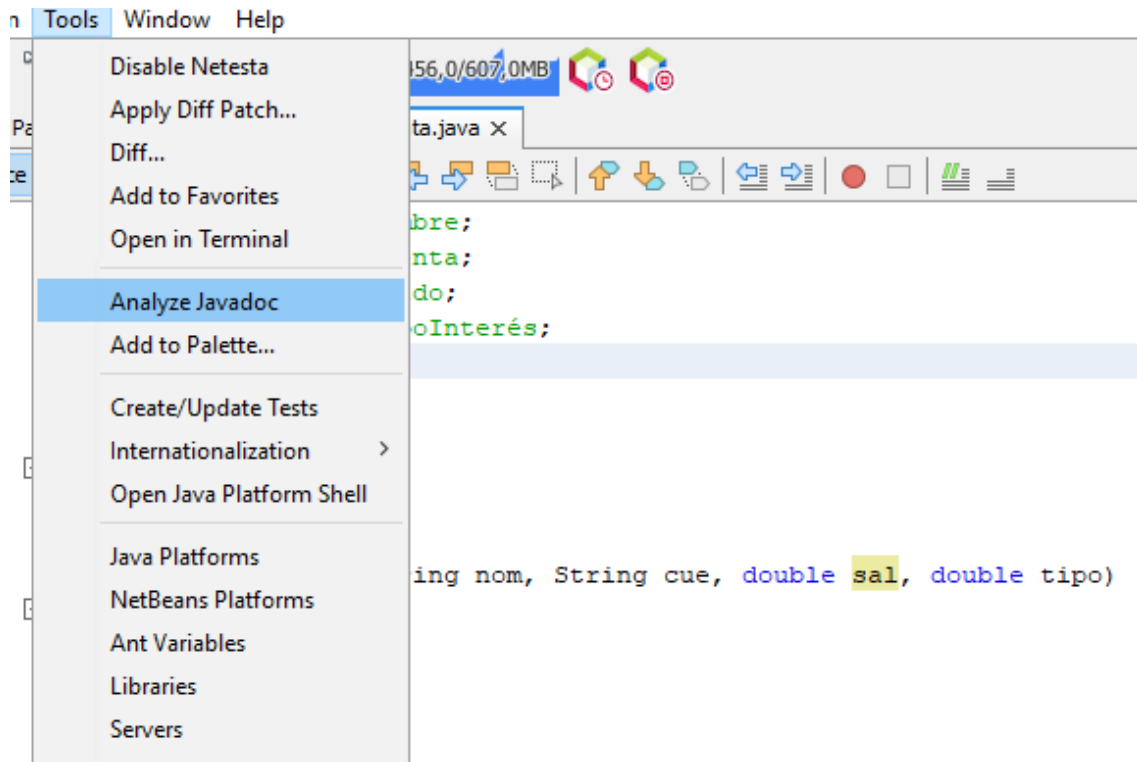


3. JAVADOC

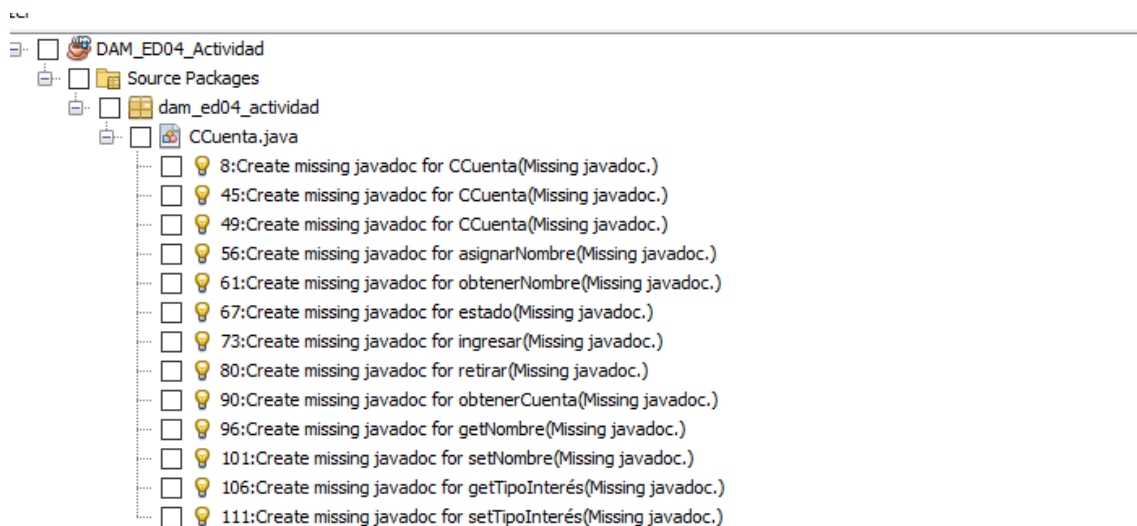
3.8. JAVADOC 8

Inserta comentarios Javadoc en la clase **CCuenta**.

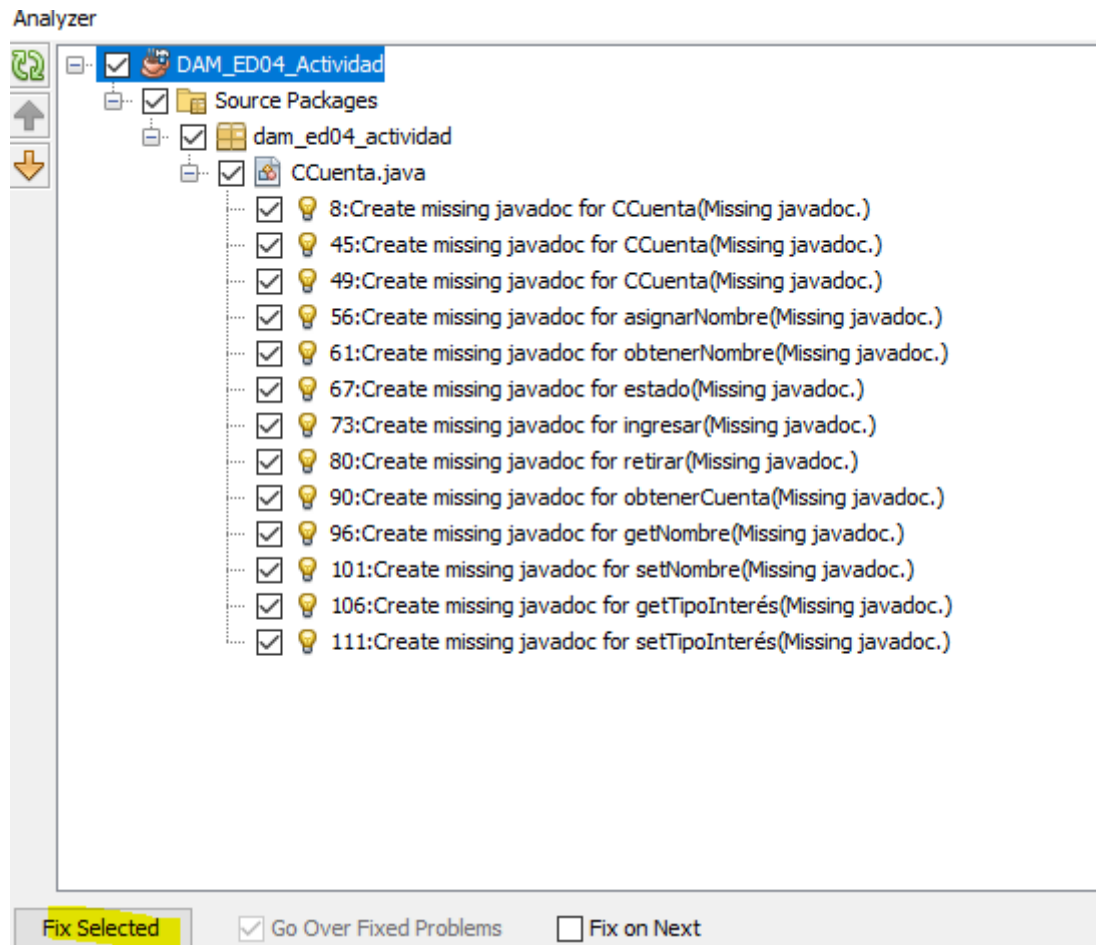
Desde Tools>Analyze Javadoc:



Después de analizar se muestran los elementos en los que puede generar JAVADOC:



Seleccionamos todos, después hacemos click en Fix Selected para generar todos los comentarios JAVADOC en la clase:



Luego podremos modificarlos con las especificaciones reales de cada elemento.

Por ejemplo, en el Constructor de la clase:

```
/**
 *
 * @param nom
 * @param cue
 * @param sal
 * @param tipo
 */
public CCuenta(String nom, String cue, double sal, double tipo)
{
    nombre =nom;
    cuenta=cue;
    saldo=sal;
}
```

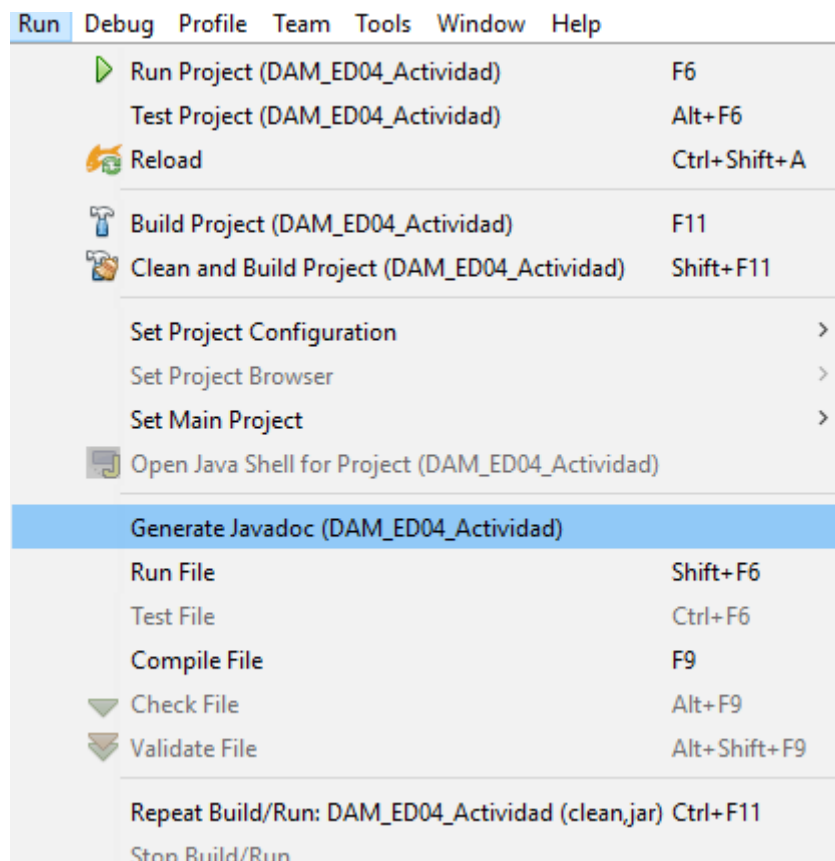
En realidad, sería algo así ya que no inicializa el atributo tipoInterés, no hace falta el parámetro:

```
/**
 *Constructor de la clase CCuenta, inicializa los atributos a excepción del tipo interés.
 * @param nom Parámetro que inicializa el nombre de la persona.
 * @param cue Parámetro que inicializa el identificador de la cuenta.
 * @param sal Parámetro que inicializa el saldo de la cuenta.
 */
public CCuenta(String nom, String cue, double sal, double tipo)
{
    nombre =nom;
    cuenta=cue;
    saldo=sal;
}
```

3.9. JAVADOC 9

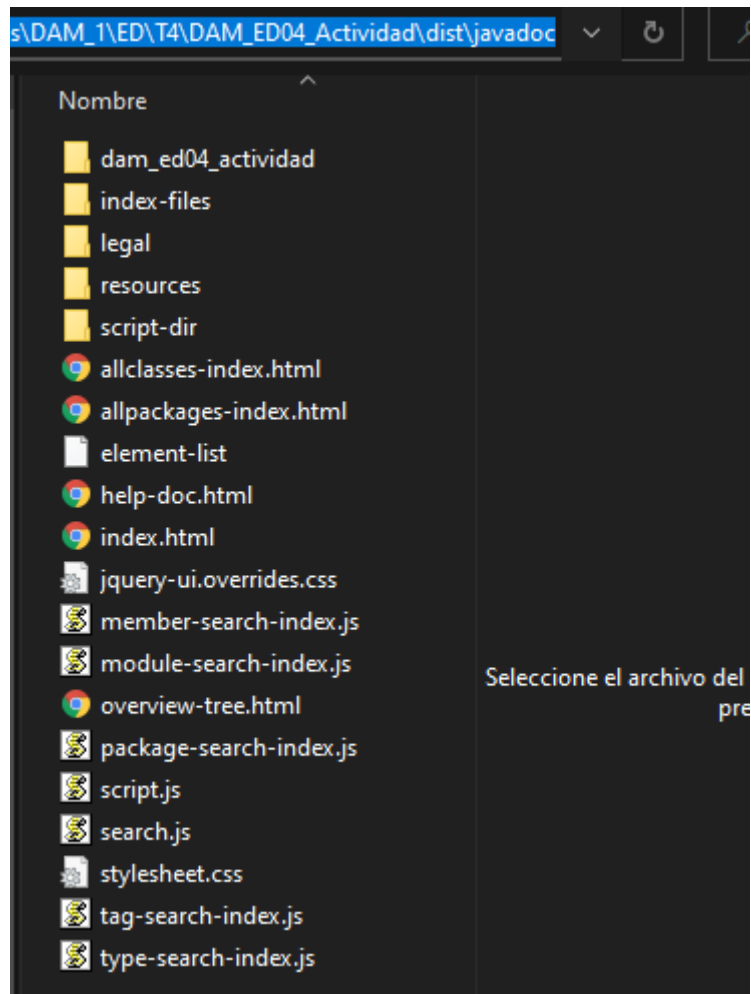
Genera documentación Javadoc para todo el proyecto.

Desde Run>Generate Javadoc:



Se genera la documentación en la carpeta del proyecto:

..DAM_ED04_Actividad\dist\javadoc



3.10. JAVADOC 10

Comprueba que la documentación generada por Javadoc, abarca todos los métodos y atributos de la clase **CCuenta**.

Ejemplo visionado de la documentación en navegador web:

All Classes and Interfaces

Classes

Class	Description
CCuenta	
Main	

← → ↻ ⓘ Archivo | file:///C:/Users/gianf/OneDrive/ ☆ ☆

PACKAGE CLASS USE TREE INDEX HELP

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

SEARCH: 🔍 Search

Constructor Details

CCuenta

```
public CCuenta()
```

CCuenta

```
public CCuenta(String nom,
                String cue,
                double sal,
                double tipo)
```

Constructor de la clase CCuenta, inicializa los atributos a excepción del tipo interés.

Parameters:

- nom - Parámetro que inicializa el nombre de la persona.
- cue - Parámetro que inicializa el identificador de la cuenta.
- sal - Parámetro que inicializa el saldo de la cuenta.

Method Summary

All Methods	Instance Methods	Concrete Methods
Modifier and Type	Method	Description
void	asignarNombre(String ¹ nom)	
double	estado()	
String ¹	getCuenta()	
String ¹	getNombre()	
double	getSaldo()	
double	getTipoInterés()	
void	ingresar(double cantidad)	
String ¹	obtenerCuenta()	
String ¹	obtenerNombre()	
void	retirar(double cantidad)	
void	setCuenta(String ¹ cuenta)	
void	setNombre(String ¹ nombre)	
void	setSaldo(double saldo)	
void	setTipoInterés(double tipoInterés)	