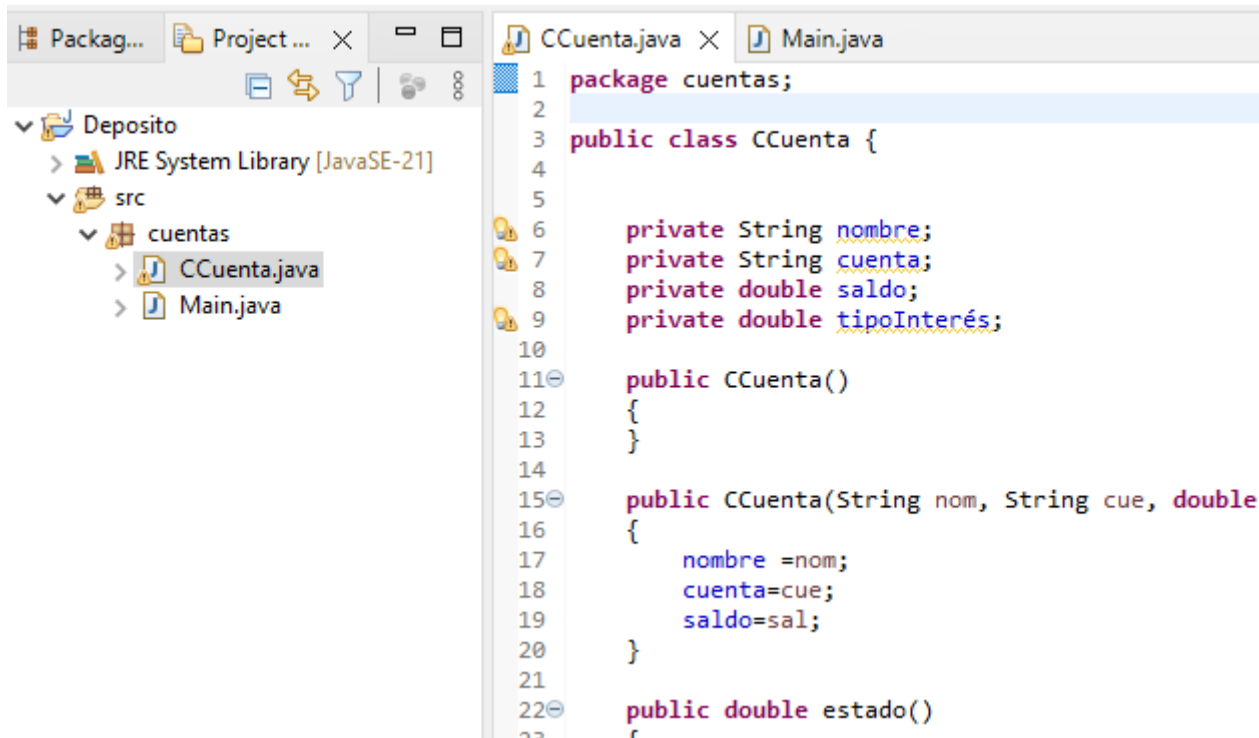


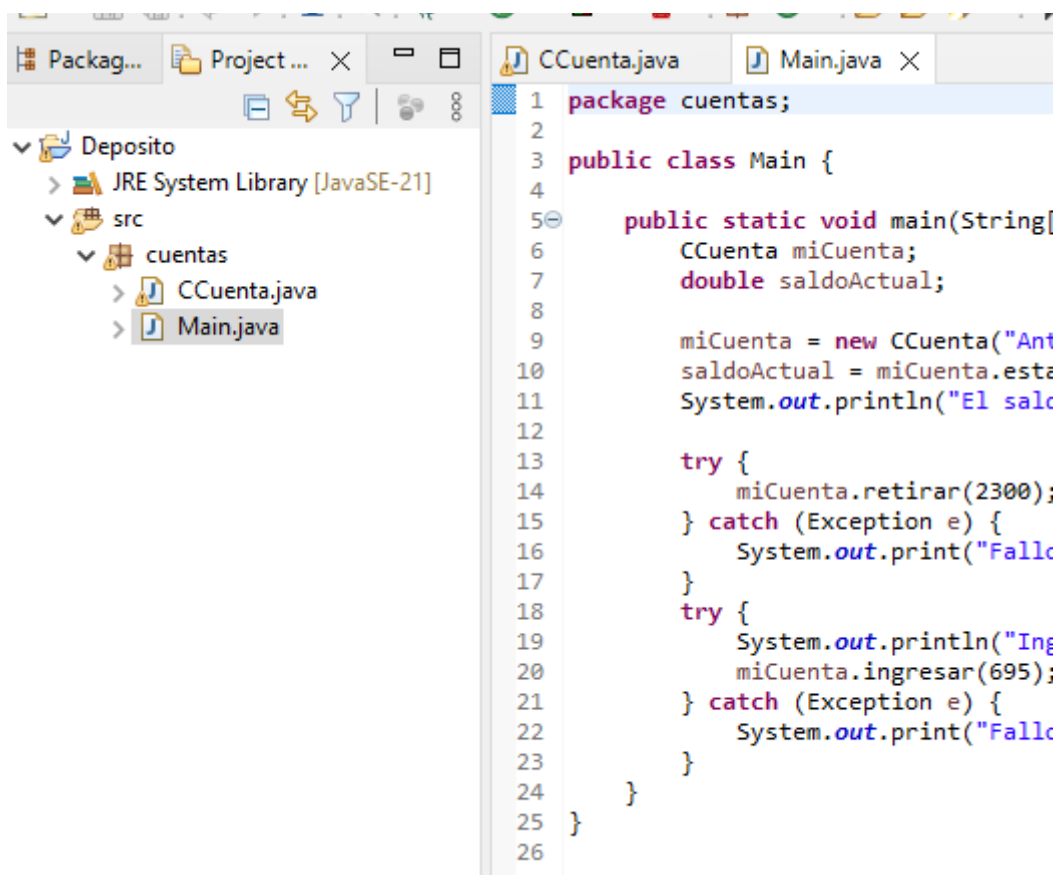
REFACTORIZACIÓN

1. Las clases deberán formar parte del paquete cuentas.

Click derecho proyecto → refactor → Move



```
1 package cuentas;
2
3 public class CCuenta {
4
5
6     private String nombre;
7     private String cuenta;
8     private double saldo;
9     private double tipoInterés;
10
11     public CCuenta()
12     {
13     }
14
15     public CCuenta(String nom, String cue, double
16     {
17         nombre = nom;
18         cuenta = cue;
19         saldo = sal;
20     }
21
22     public double estado()
23     {
```



```
1 package cuentas;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         CCuenta miCuenta;
7         double saldoActual;
8
9         miCuenta = new CCuenta("Antonio", "1234", 1000, 0.05);
10        saldoActual = miCuenta.estado();
11        System.out.println("El saldo actual es " + saldoActual);
12
13        try {
14            miCuenta.retirar(2300);
15        } catch (Exception e) {
16            System.out.print("Falló el retiro");
17        }
18        try {
19            System.out.println("Ingresando dinero...");
20            miCuenta.ingresar(695);
21        } catch (Exception e) {
22            System.out.print("Falló el ingreso");
23        }
24    }
25 }
26
```

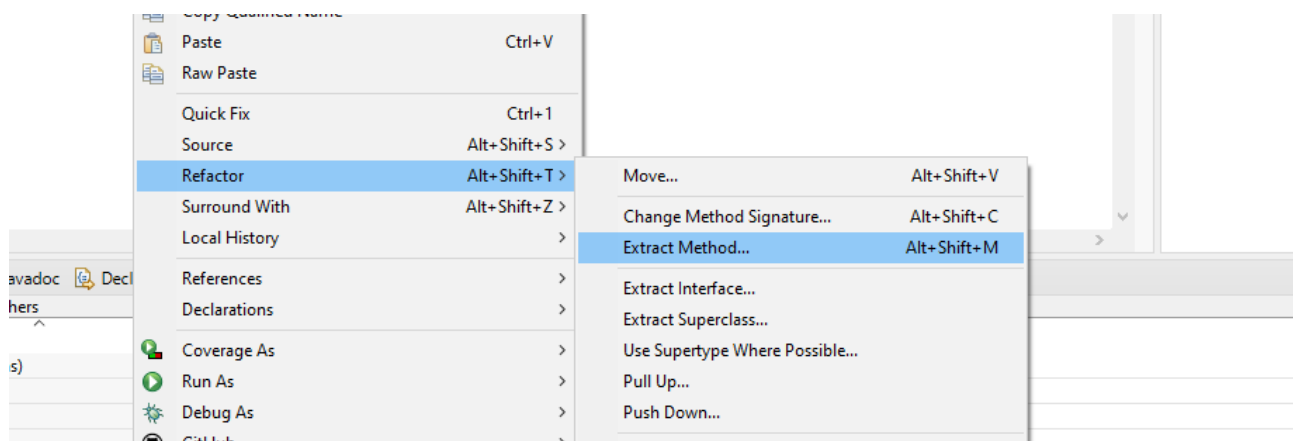
2. Cambiar el nombre de la variable "miCuenta" por "cuenta1".

Selecciona el nombre de la variable → refactor → Rename

```
public class Main {  
    public static void main(String[] args) {  
        CCuenta cuenta1;  
        double saldoActual;  
        cuenta1 = new CCuenta("Antonio López", "1000-2365-8"  
        saldoActual = cuenta1.estado();  
        System.out.println("El saldo actual es"+ saldoActual);  
  
        try {  
            cuenta1.retirar(2300);  
        } catch (Exception e) {  
            System.out.print("Fallo al retirar");  
        }  
        try {  
            System.out.println("Ingreso en cuenta");  
            cuenta1.ingresar(695);  
        } catch (Exception e) {  
            System.out.print("Fallo al ingresar");  
        }  
    }  
}
```

3. Introducir el método operativa_cuenta, que englobe las sentencias de la clase Main que operan con el objeto cuenta1.

Selecciona el código → refactor → extract method



```

package cuentas;

public class Main {

    public static void main(String[] args) {
        operativa_cuenta();
    }

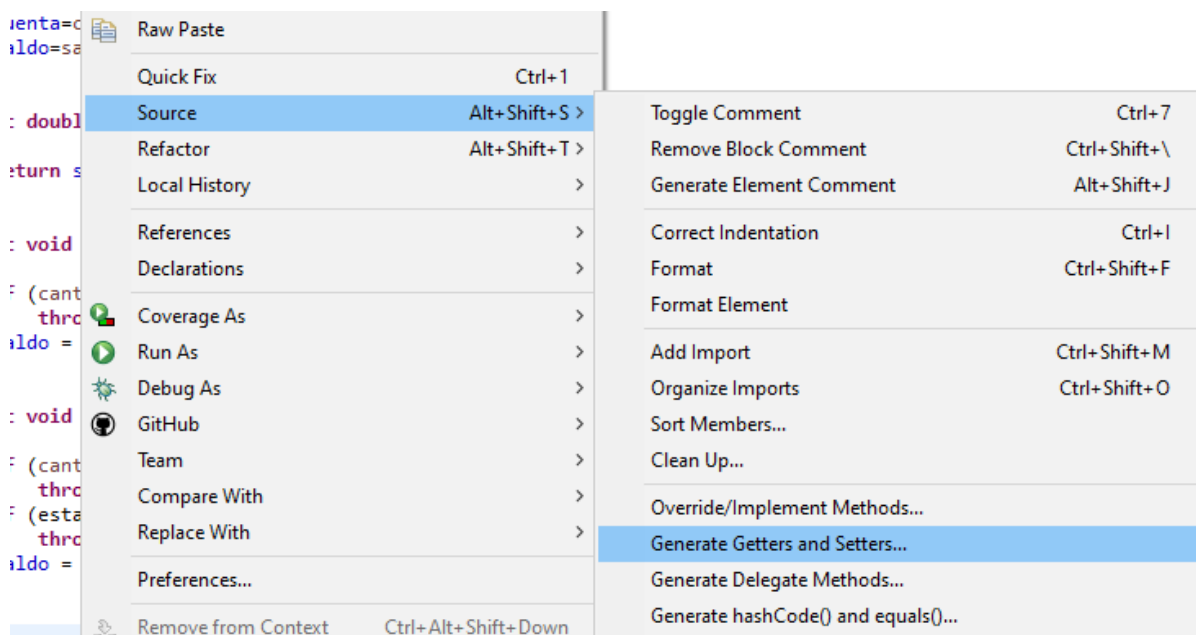
    public static void operativa_cuenta() {
        CCuenta cuenta1;
        double saldoActual;
        cuenta1 = new CCuenta("Antonio López", "1000-2365-85-1230456789", 2500, 0);
        saldoActual = cuenta1.estado();
        System.out.println("El saldo actual es"+ saldoActual );

        try {
            cuenta1.retirar(2300);
        } catch (Exception e) {
            System.out.print("Fallo al retirar");
        }
        try {
            System.out.println("Ingreso en cuenta");
            cuenta1.ingresar(695);
        } catch (Exception e) {
            System.out.print("Fallo al ingresar");
        }
    }
}

```

4. Encapsular los atributos de la clase Ccuenta.

Click derecho → source → generate getters and setters



```

package cuentas;

public class CCuenta {

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

    public CCuenta()
    {
    }

    public CCuenta(String nom, String cue, double sal, double tipo)
    {
        nombre =nom;
        cuenta=cue;
        saldo=sal;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public String getCuenta() {
        return cuenta;
    }

    public void setCuenta(String cuenta) {
        this.cuenta = cuenta;
    }

    public double getSaldo() {
        return saldo;
    }

    public void setSaldo(double saldo) {
        this.saldo = saldo;
    }

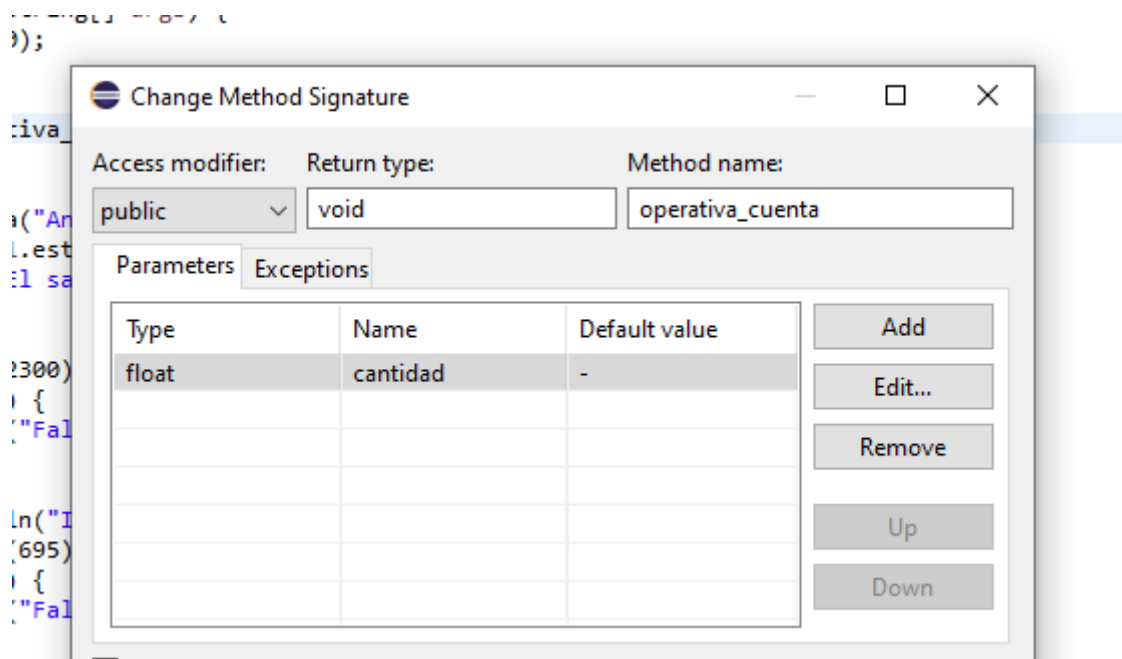
    public double getTipoInterés() {
        return tipoInterés;
    }

    public void setTipoInterés(double tipoInterés) {
        this.tipoInterés = tipoInterés;
    }
}

```

5. Añadir un nuevo parámetro al método operativa_cuenta, de nombre cantidad y de tipo float.

Selecciona el método → refactor → change method signature



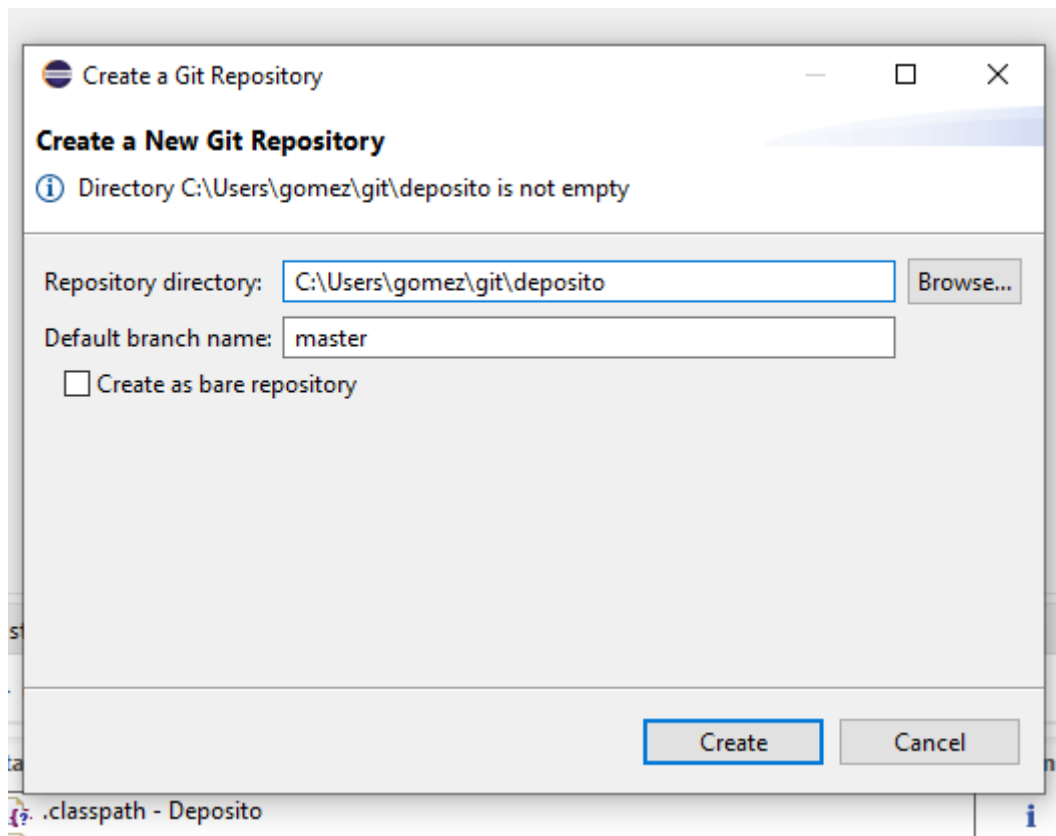
```

1 package cuentas;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         operativa_cuenta(2500);
7     }
8
9     public static void operativa_cuenta(float cantidad) {
10         CCuenta cuenta1;
11         double saldoActual;
12         cuenta1 = new CCuenta("Antonio López","1000-2365-85-1230456789",cantidad,0);
13         saldoActual = cuenta1.estado();
14         System.out.println("El saldo actual es"+ saldoActual );
15
16         try {
17             cuenta1.retirar(2300);
18         } catch (Exception e) {
19             System.out.print("Fallo al retirar");
20         }
21
22         try {
23             System.out.println("Ingreso en cuenta");
24             cuenta1.ingresar(695);
25         } catch (Exception e) {
26             System.out.print("Fallo al ingresar");
27         }
28     }
29 }

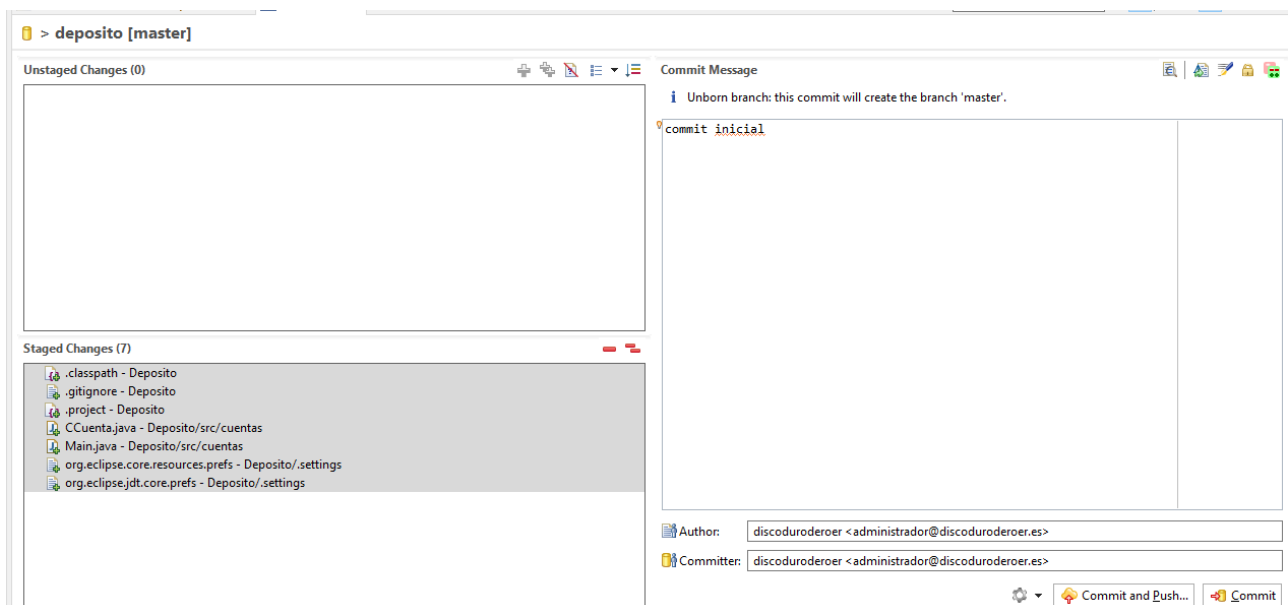
```

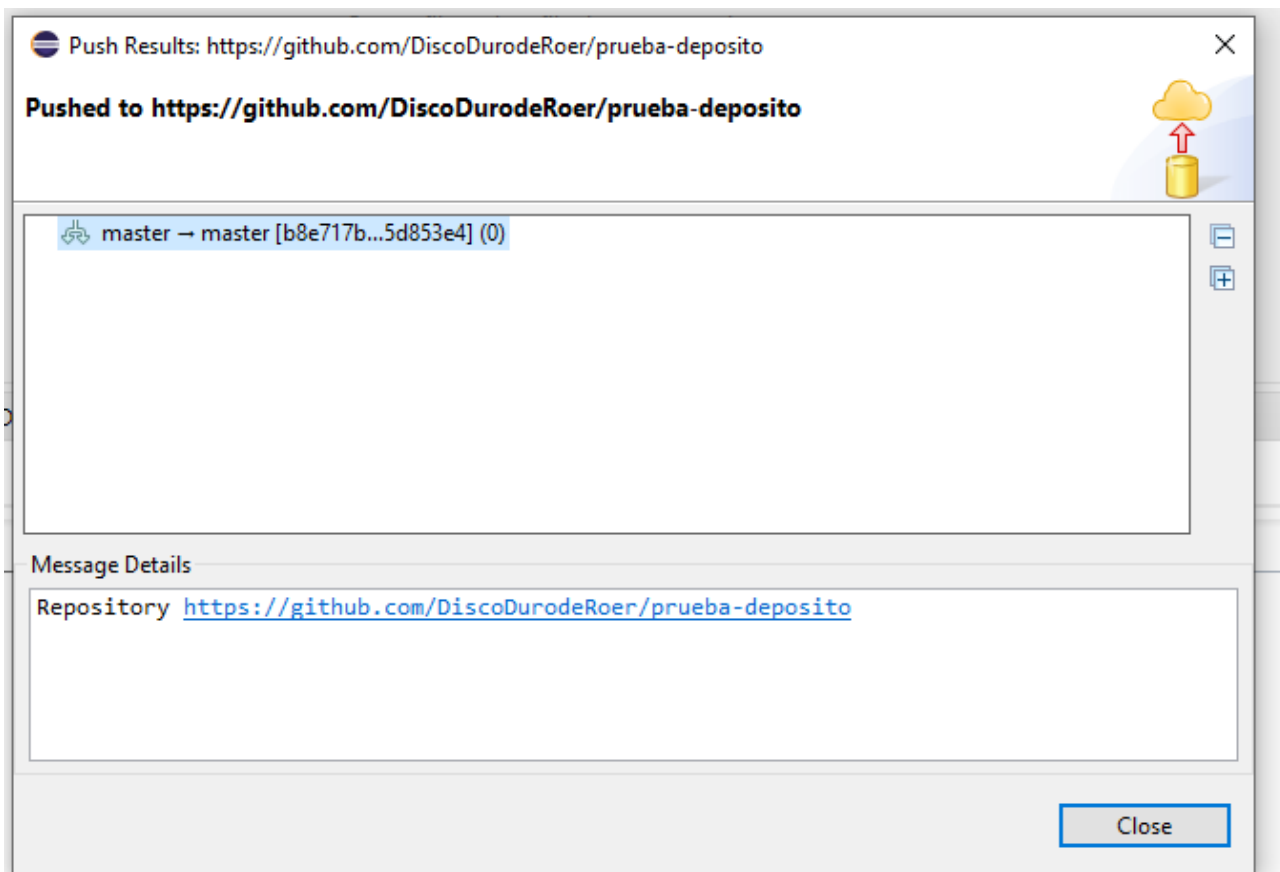
GIT

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



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





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

```
fernando@DESKTOP-T29D3Q4 MINGW64 ~/git/deposito (master)
$ git log
commit b8e717b5e75b9ea580aa73f65b12b3a0f8cc26d6 (HEAD -> master)
Author: discoduroderoer <administrador@discoduroderoer.es>
Date: Thu Feb 15 22:22:10 2024 +0100

    commit inicial

fernando@DESKTOP-T29D3Q4 MINGW64 ~/git/deposito (master)
$ |
```

JAVADOC

1. Insertar comentarios JavaDoc en la clase Ccuenta.

2. Generar documentación JavaDoc para todo el proyecto y comprueba que abarcatodos los métodos y atributos de la clase Ccuenta.

```

}

/**
 * Devuelve el estado de la cuenta
 * @return saldo de la cuenta
 */
public double estado()
{
    return saldo;
}

/**
 * Ingresa una cantidad en la cuenta
 * @param cantidad cantidad a ingresar
 * @throws Exception si la cantidad es negativa
 */
public void ingresar(double cantidad) throws Exception
{
    if (cantidad<0)
        throw new Exception("No se puede ingresar una cantidad negativa");
    saldo = saldo + cantidad;
}

/**
 * Retira una cantidad
 * @param cantidad cantidad a retirar
 * @throws Exception si la cantidad es 0 o negativas
 */
public void retirar(double cantidad) throws Exception
{
    if (cantidad <= 0)
        throw new Exception ("No se puede retirar una cantidad negativa");
    if (estado()< cantidad)
        throw new Exception ("No se hay suficiente saldo");
    saldo = saldo - cantidad;
}
}

```


Package cuentas

Class CCuenta

java.lang.Object[Ⓔ]
cuentas.CCuenta

```
public class CCuenta  
extends ObjectⒺ
```

Author:

DDR Clase Cuenta representa a una cuenta bancaria

Constructor Summary

Constructors

Constructor	Description
CCuenta()	Constructor por defecto
CCuenta(String [Ⓔ] nom, String [Ⓔ] cue, double sal, double tipo)	Constructro con todos los valores

Method Summary

All Methods

Instance Methods

Concrete Methods

Modifier and Type	Method	Description
double	estado()	Obtiene el saldo de la cuenta
String [Ⓔ]	getCuenta()	Obtiene el numero de cuenta
String [Ⓔ]	getNombre()	Obtiene el titular de la cuenta
double	getSaldo()	Obtiene el saldo de la cuenta
double	getTipoInterés()	Obtiene el tipo de interes
void	ingresar(double cantidad)	Ingresa cantidad a la cuenta, debe ser positivo
void	retirar(double cantidad)	Retira cantidad a la cuenta, debe ser positivo Debe haber sufiecinte sa
void	setCuenta(String [Ⓔ] cuenta)	Modifica el numero de cuenta
void	setNombre(String [Ⓔ] nombre)	Modifica el titular de la cuenta
void	setSaldo(double saldo)	Modifica el saldo de la cuenta

Javadoc command:

C:\Program Files\Java\jdk-21\bin\javadoc.exe

Configure...

Select types for which Javadoc will be generated:

> ☒ Deposito

Create Javadoc for members with visibility:

☐ Private

☐ Package

☐ Protected

☒ Public

Public: Generate Javadoc for public classes and members.

☒ Use standard doclet

Destination: C:\Users\gomez\git\deposito\Deposito\doc

Browse...

☐ Use custom doclet

Doclet name:

Doclet class path:



< Back

Next >

Finish

Cancel