



EJERCICIOS REFACTORIZACIÓN

Luis Luzuriaga Manero

- **Cambiar el nombre de la variable miCuenta por cuenta1.**

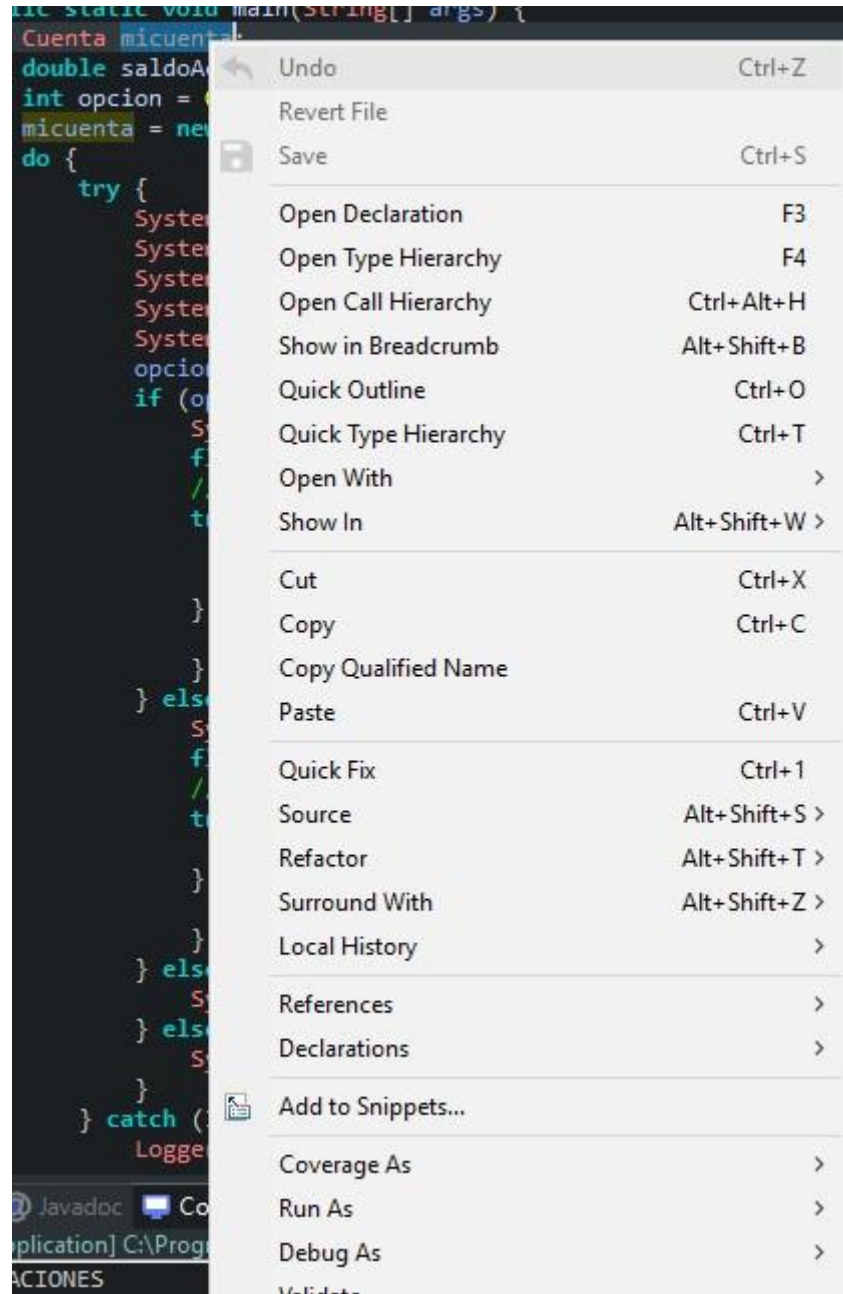
1. Localizamos la variable en la clase.

```
Cuenta micuenta;
```

2. La seleccionamos dándole doble clic.

```
Cuenta micuenta;
```

3. Le damos clic derecho.



4. En el panel que aparece, buscamos “**Refactor**”.

Refactor Alt+Shift+T >

5. En el subpanel hacemos clic encima de la opción “**Rename**”

Refactor Alt+Shift+T > Rename... Alt+Shift+R

6. Ahora nos aparecerá coloreadas todas las llamadas a esta variable. Sólo tenemos que escribir el nuevo nombre que le queremos dar y listo.

```

Cuenta cuenta1;
double saldoActual;
int opcion = 0;
cuenta1 = new Cuenta("Juan Magán", "1000-2365-85-123456789", 2500, 0);
do {
    try {
        System.out.println("MENÚ DE OPERACIONES");
        System.out.println("-----");
        System.out.println("1 - Ingresar");
        System.out.println("2 - Retirar");
        System.out.println("3 - Finalizar");
        opcion = Integer.parseInt(dato.readLine());
        if (opcion == 1) {
            System.out.println("¿Cuánto desea ingresar?: ");
            float ingresar = Integer.parseInt(dato.readLine());
            // Operativa con la cuenta para la opción 1
            try {
                System.out.println("Ingreso en cuenta");
                cuenta1.ingresar(ingresar);
            } catch (Exception e) {
                System.out.print("Fallo al ingresar");
            }
        } else if (opcion == 2) {
            System.out.println("¿Cuánto desea retirar?: ");
            float retirar = Integer.parseInt(dato.readLine());
            // Operativa con la cuenta para la opción 2
            try {
                cuenta1.retirar(retirar);
            } catch (Exception e) {
                System.out.print("Fallo al retirar");
            }
        } else if (opcion == 3) {
            System.out.println("Finalizamos la ejecución");
        } else {
            System.err.println("Opción errónea");
        }
    } catch (IOException ex) {
        Logger.getLogger(Main.class.getName()).log(Level.SEVERE, null,

```



- Introduce el método `operativaCuenta` en la clase `Main`, que englobe las sentencias de la misma que operan con el objeto `cuenta1`.
 1. Primero, seleccionamos el fragmento de código que queremos extraer.

```

if (opcion == 1) {
    System.out.println("¿Cuánto desea ingresar?: ");
    float ingresar = Integer.parseInt(data.readLine());
    // Operativa con la cuenta para la opción 1
    try {
        System.out.println("Ingreso en cuenta");
        cuenta1.ingresar(ingresar);
    } catch (Exception e) {
        System.out.print("Fallo al ingresar");
    }
} else if (opcion == 2) {
    System.out.println("¿Cuánto desea retirar?: ");
    float retirar = Integer.parseInt(data.readLine());
    // Operativa con la cuenta para la opción 2
    try {
        cuenta1.retirar(retirar);
    } catch (Exception e) {
        System.out.print("Fallo al retirar");
    }
} else if (opcion == 3) {
    System.out.println("Finalizamos la ejecución");
} else {
    System.err.println("Opción errónea");
}
} catch (IOException ex) {
    Logger.getLogger(Main.class.getName()).log(Level.SEVERE, null,
        ex);
}
}

```

2. Una vez seleccionado, hacemos clic derecho.

	Undo Typing	Ctrl+Z
	Revert File	
	Save	Ctrl+S
	Open Declaration	F3
	Open Type Hierarchy	F4
	Open Call Hierarchy	Ctrl+Alt+H
	Show in Breadcrumb	Alt+Shift+B
	Quick Outline	Ctrl+O
	Quick Type Hierarchy	Ctrl+T
	Open With	>
	Show In	Alt+Shift+W >
	Cut	Ctrl+X
	Copy	Ctrl+C
	Copy Qualified Name	
	Paste	Ctrl+V
	Quick Fix	Ctrl+1
	Source	Alt+Shift+S >
	Refactor	Alt+Shift+T >

3. En el panel, seleccionamos la opción Refactor.

Refactor	Alt+Shift+T >
----------	---------------

4. En el subpanel seleccionamos "Extract Method"

Refactor	Alt+Shift+T >	Move...	Alt+Shift+V
Surround With	Alt+Shift+Z >	Change Method Signature...	Alt+Shift+C
Local History	>	Extract Method...	Alt+Shift+M

5. En la ventana que te sale, tenemos que escribir el nombre del método **“operativaCuenta”**

Extract Method

Method name:

Access modifier: ☐ public ☐ protected ☐ package ☐ private

Parameters:

Type	Name
Cuenta	cuenta1
int	opcion

☐ Declare thrown runtime exceptions
☐ Generate method comment
☐ Replace additional occurrences of statements with method

Method signature preview:
`private static void operativaCuenta(Cuenta cuenta1, int opcion)
throws IOException`

6. Le damos a **“OK”** y se nos habrá extraído el método.


```

cuenta1 = new Cuenta("Main.java", "1000-2365-85-123456789", 2500, 0);
do {
    try {
        System.out.println("MENÚ DE OPERACIONES");
        System.out.println("-----");
        System.out.println("1 - Ingresar");
        System.out.println("2 - Retirar");
        System.out.println("3 - Finalizar");
        opcion = Integer.parseInt(dato.readLine());
        operativaCuenta(cuenta1, opcion);
    } catch (IOException ex) {
        Logger.getLogger(Main.class.getName()).log(Level.SEVERE, null,
            ex);
    }
} while (opcion != 3);
saldoActual = cuenta1.saldo;
System.out.println("El saldo actual es" + saldoActual);
}

private static void operativaCuenta(Cuenta cuenta1, int opcion) throws IOException {
    if (opcion == 1) {
        System.out.println("¿Cuánto desea ingresar?: ");
        float ingresar = Integer.parseInt(dato.readLine());
        // Operativa con la cuenta para la opción 1
        try {
            System.out.println("Ingreso en cuenta");
            cuenta1.ingresar(ingresar);
        } catch (Exception e) {
            System.out.print("Fallo al ingresar");
        }
    }
    else if (opcion == 2) {
        System.out.println("¿Cuánto desea retirar?: ");
        float retirar = Integer.parseInt(dato.readLine());
        // Operativa con la cuenta para la opción 2
        try {
            cuenta1.retirar(retirar);
        } catch (Exception e) {
            System.out.print("Fallo al retirar");
        }
    }
    else if (opcion == 3) {
        System.out.println("Finalizamos la ejecución");
    }
    else {
        System.err.println("Opción errónea");
    }
}

```

- **Encapsula los cuatro atributos de la clase Cuenta.**



1. Primero tenemos que seleccionar los atributos de la clase.

```

public String nombre;
public String cuenta;
public double saldo;
public double tipoInteres;

```

2. Ahora hacemos clic derecho.

	Undo Typing	Ctrl+Z
	Revert File	
	Save	Ctrl+S
	Open Declaration	F3
	Open Type Hierarchy	F4
	Open Call Hierarchy	Ctrl+Alt+H
	Show in Breadcrumb	Alt+Shift+B
	Quick Outline	Ctrl+O
	Quick Type Hierarchy	Ctrl+T
	Open With	>
	Show In	Alt+Shift+W >
	Cut	Ctrl+X
	Copy	Ctrl+C
	Copy Qualified Name	
	Paste	Ctrl+V
	Quick Fix	Ctrl+1
	Source	Alt+Shift+S >

3. Seleccionamos la opción de “**Source**”.

Source Alt+Shift+S >

4. Ahora la opción de “**Generate Getters and Setters**”.

Generate Getters and Setters...

5. En la ventana que nos aparece, seleccionamos todos los parámetros y le damos a “**OK**”.

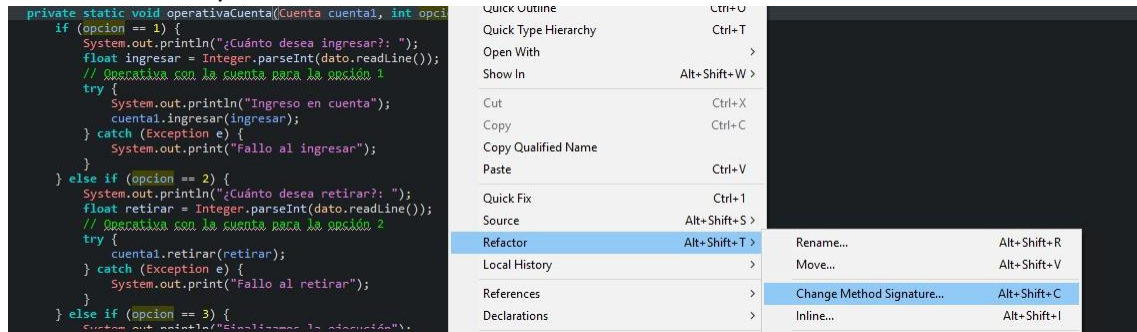


6. Ya tenemos los métodos generados.

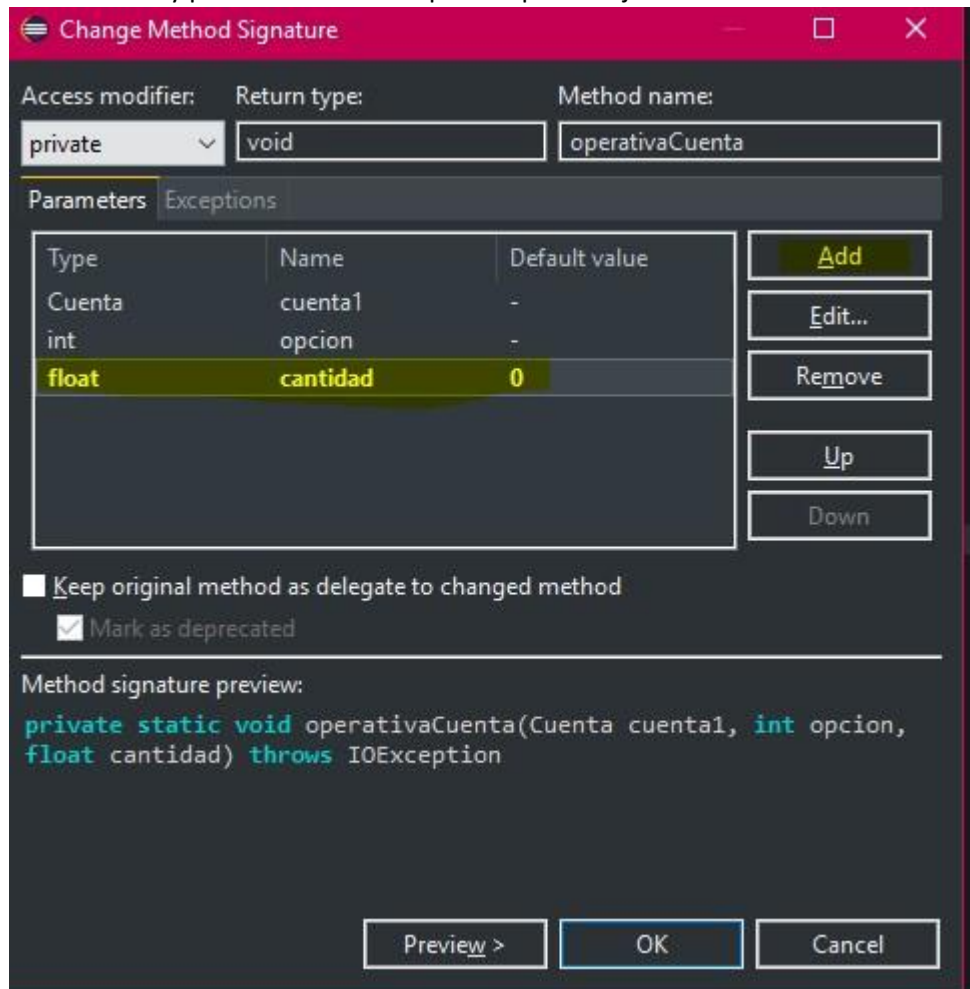
```
public String getNombre() {  
    return nombre;  
}  
  
public String getCuenta() {  
    return cuenta;  
}  
  
public double getSaldo() {  
    return saldo;  
}  
  
public double getTipoInteres() {  
    return tipoInteres;  
}  
  
public void setNombre(String nombre) {  
    this.nombre = nombre;  
}  
  
public void setCuenta(String cuenta) {  
    this.cuenta = cuenta;  
}  
  
public void setSaldo(double saldo) {  
    this.saldo = saldo;  
}  
  
public void setTipoInteres(double tipoInteres) {  
    this.tipoInteres = tipoInteres;  
}
```

- Cantidad introducida sea un parámetro más.

1. Vamos al método y le hacemos clic derecho.



2. Seleccionamos la opción “**Change method signature**”. En la ventana hacemos clic en “**Add**” y ponemos los datos que nos pide el ejercicio.



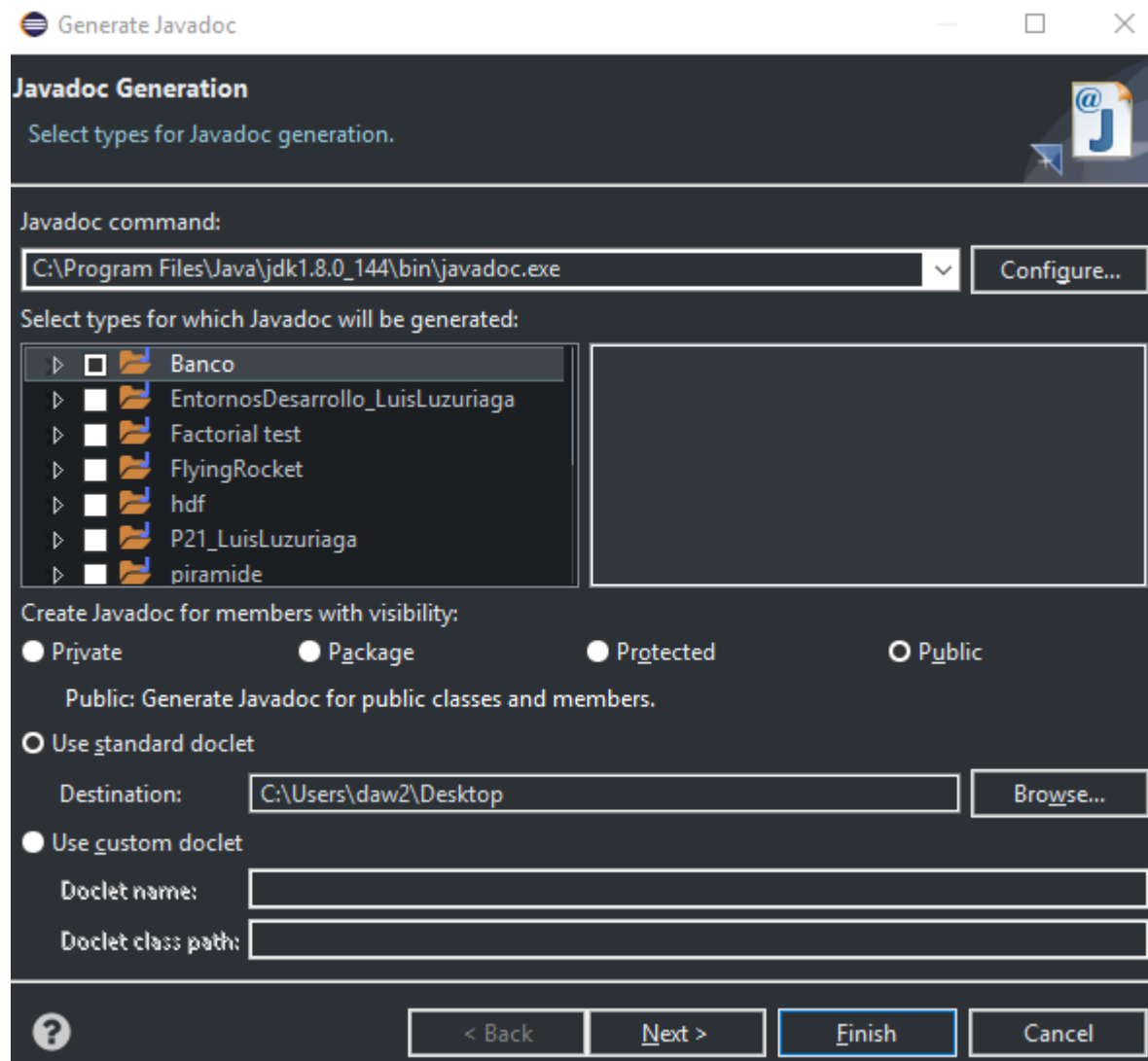
3. Le daremos a “**OK**” y se nos habrá añadido un nuevo parámetro.

```
private static void operativaCuenta(Cuenta cuenta1, int opcion, float cantidad) throws IOException {
    // ...
    operativaCuenta(cuenta1, opcion, 0);
}
```

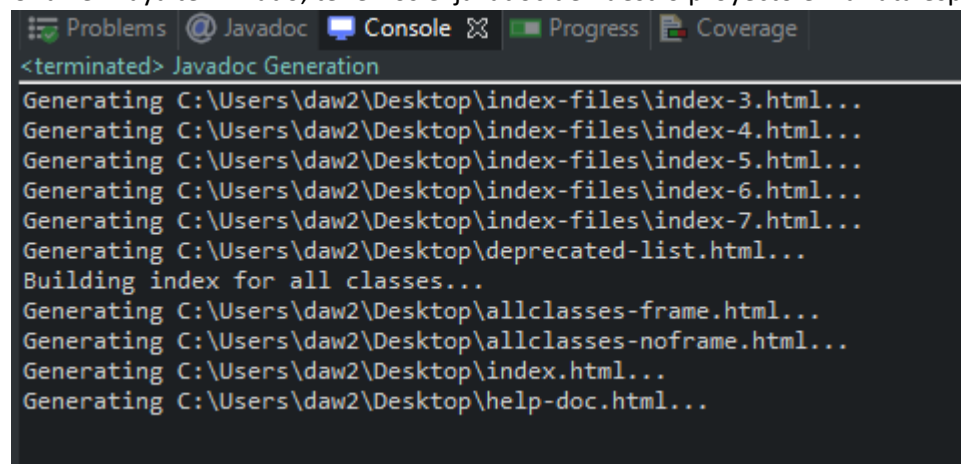
4. Para que sea introducido por el usuario, bastaría con añadir:

```
opcion = Integer.parseInt(dato.readLine());
int cantidad = Integer.parseInt(dato.readLine());
operativaCuenta(cuenta1, opcion, cantidad);
```


- Generar el javadoc.
 1. Rellenas los comentarios.
 2. Vas a la ventana "Proyecto".
 3. Seleccionas la opción "Generate Javadoc". Rellenas la ventana seleccionando la ruta del javadoc.exe



4. Una vez haya terminado, tenemos el javadoc de nuestro proyecto en la ruta especificada.



Class Cuenta

java.lang.Object
Cuenta

public class Cuenta
extends java.lang.Object

Field Summary

Fields

Modifier and Type	Field and Description
java.lang.String	cuenta
java.lang.String	nombre
double	saldo
double	tipoInteres

Constructor Summary

Constructors

Constructor and Description
Cuenta()
Cuenta(java.lang.String nom, java.lang.String cue, double sal, double tipo)

Method Summary

All Methods Instance Methods Concrete Methods

Modifier and Type	Method and Description
java.lang.String	getCuenta()
java.lang.String	getNombre()
double	getSaldo()