

PILAR ALONSO SUELA

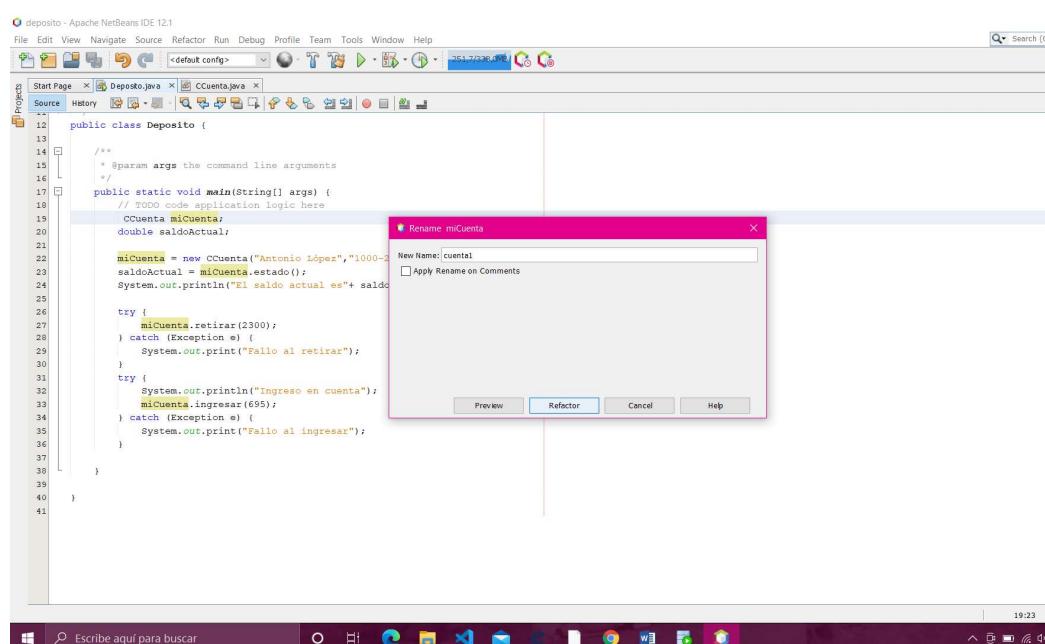
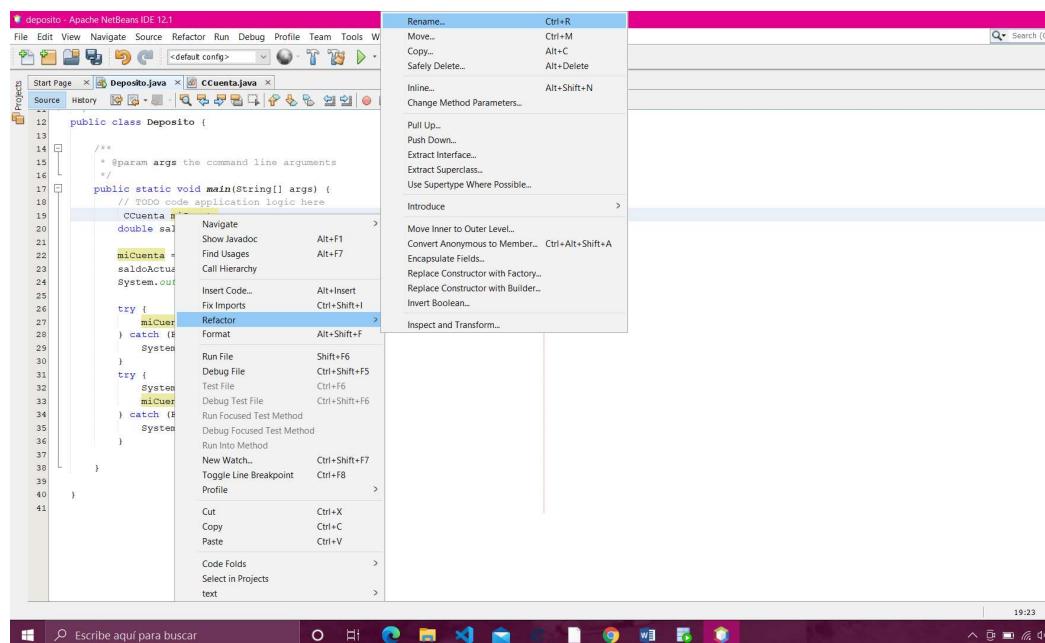
ENTORNOS DE DESARROLLO

TAREA 4

<https://github.com/PilarAlonsoSuela/entornosTarea/tree/master/nbproject/private>

REFACTORIZACIÓN

1. Las clases deberán formar parte del paquete cuentas.
2. Cambiar el nombre de la variable "miCuenta" por "cuenta1".



The screenshot shows the Apache NetBeans IDE interface. The title bar reads "deposito - Apache NetBeans IDE 12.1". The menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help. The toolbar has icons for New Project, Open Project, Save, Cut, Copy, Paste, Find, Replace, Run, Stop, and others. The source editor window displays the following Java code:

```
12  public class Deposito {
13
14      /**
15      * @param args the command line arguments
16      */
17      public static void main(String[] args) {
18          // TODO code application logic here
19          Cuenta cuenta1;
20          double saldoActual;
21
22          cuenta1 = new CCuenta("Antonio López", "1000-2365-85-1230456789", 2500, 0);
23          saldoActual = cuenta1.estado();
24          System.out.println("El saldo actual es" + saldoActual );
25
26          try {
27              cuenta1.retirar(2300);
28          } catch (Exception e) {
29              System.out.print("Fallo al retirar");
30          }
31          try {
32              System.out.println("Ingreso en cuenta");
33              cuenta1.ingresar(695);
34          } catch (Exception e) {
35              System.out.print("Fallo al ingresar");
36          }
37      }
38
39  }
```

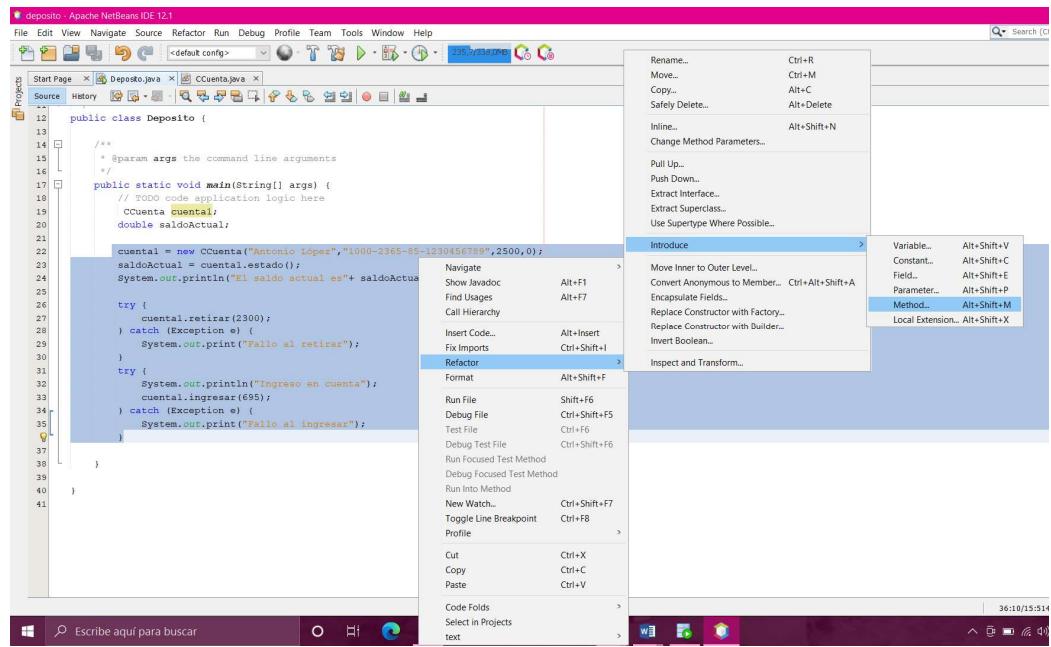
The status bar at the bottom shows "Save All finished." and the time "19:25".

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

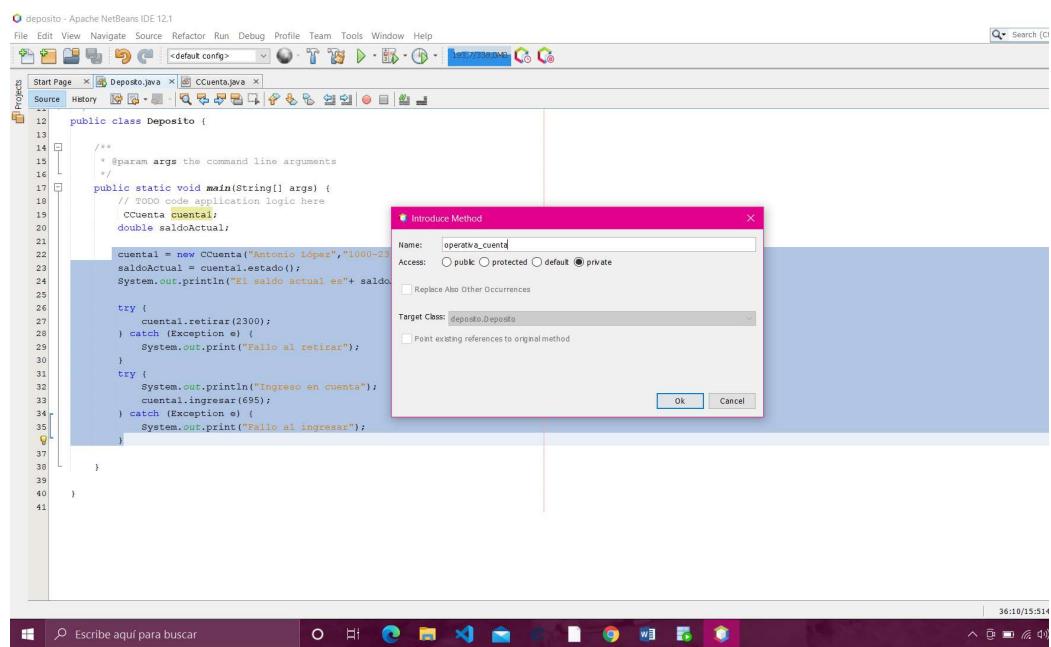
Selecciono el código

The screenshot shows the Apache NetBeans IDE interface with the same project and file structure as the previous screenshot. The code in the source editor is highlighted with a blue selection rectangle, indicating it is selected for refactoring. The status bar at the bottom shows "36:10/15:514".

Botón derecho → refactor → Introduce → method



Ponemos el nombre indicado en la tarea



```

12  public class Deposito {
13
14     /**
15      * @param args the command line arguments
16     */
17    public static void main(String[] args) {
18        // TODO code application logic here
19        Cuenta cuenta;
20        double saldoActual;
21
22        operativa_cuenta();
23    }
24
25    private static void operativa_cuenta() {
26        Cuenta cuenta;
27        double saldoActual;
28        cuenta = new CCuenta("Antonio López", "1000-2365-85-1230456789", 2500, 0);
29        saldoActual = cuenta.estado();
30        System.out.println("El saldo actual es "+ saldoActual );
31
32        try {
33            cuenta.retirar(2300);
34        } catch (Exception e) {
35            System.out.print("Fallo al retirar");
36        }
37
38        try {
39            System.out.println("Ingreso en cuenta");
40            cuenta.ingresar(695);
41        } catch (Exception e) {
42            System.out.print("Fallo al ingresar");
43        }
44    }
45
46}

```

4. Encapsular los atributos de la clase CCuenta.

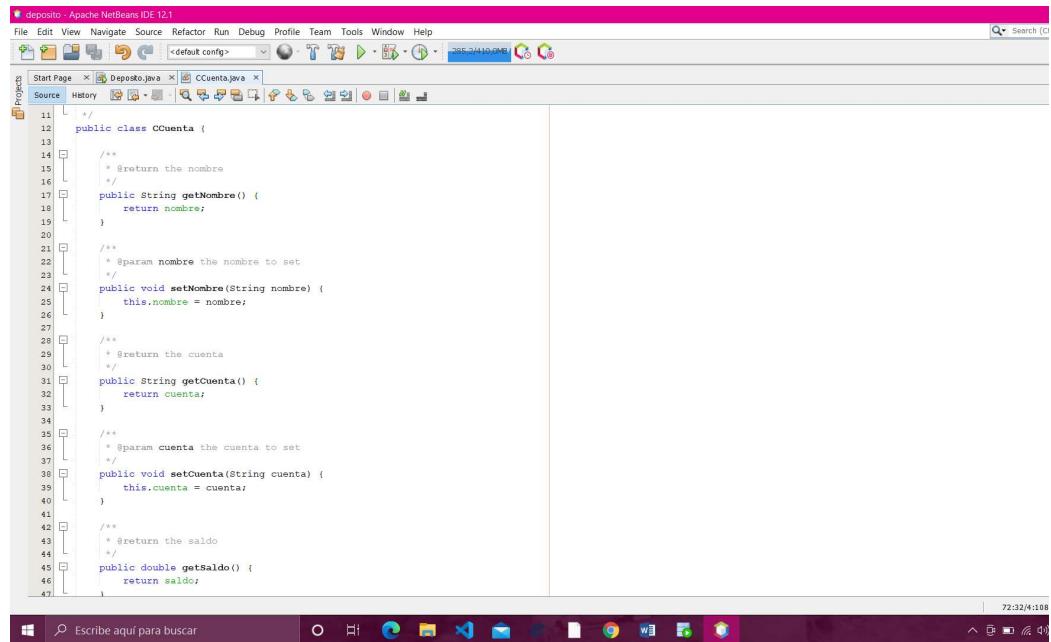
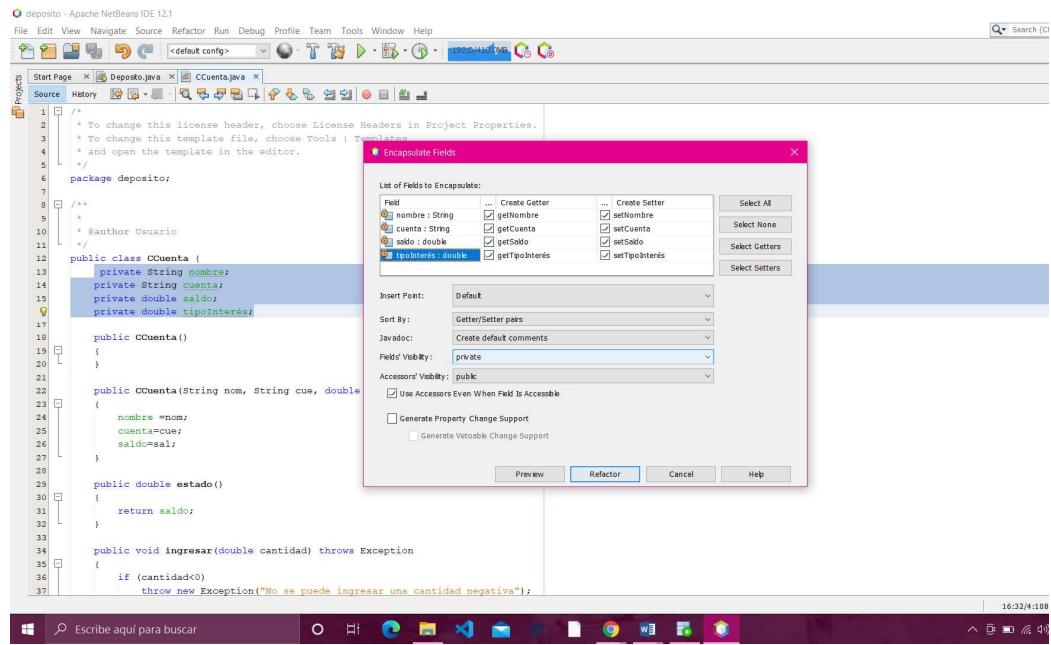
Selecciono los atributos → refactor → encapsulate fields

```

9 package deposito;
10
11 /**
12  * Author Usuario
13 */
14 public class CCuenta {
15     private String nombre;
16     private String cuenta;
17     private double saldo;
18     private double tipoInteres;
19
20     public CCuenta()
21     {
22
23     }
24
25     public CCuenta(String nom, St
26     {
27         nombre=nom;
28         cuenta=cue;
29         saldo=sal;
30     }
31
32     public double estado()
33     {
34         return saldo;
35     }
36
37     public void ingresar(double c
38     {
39         if (cantidad<0)
40             throw new Exception("nagativa");
41         saldo = saldo + cantidad;
42     }
43
44     public void retirar(double ca
45
46

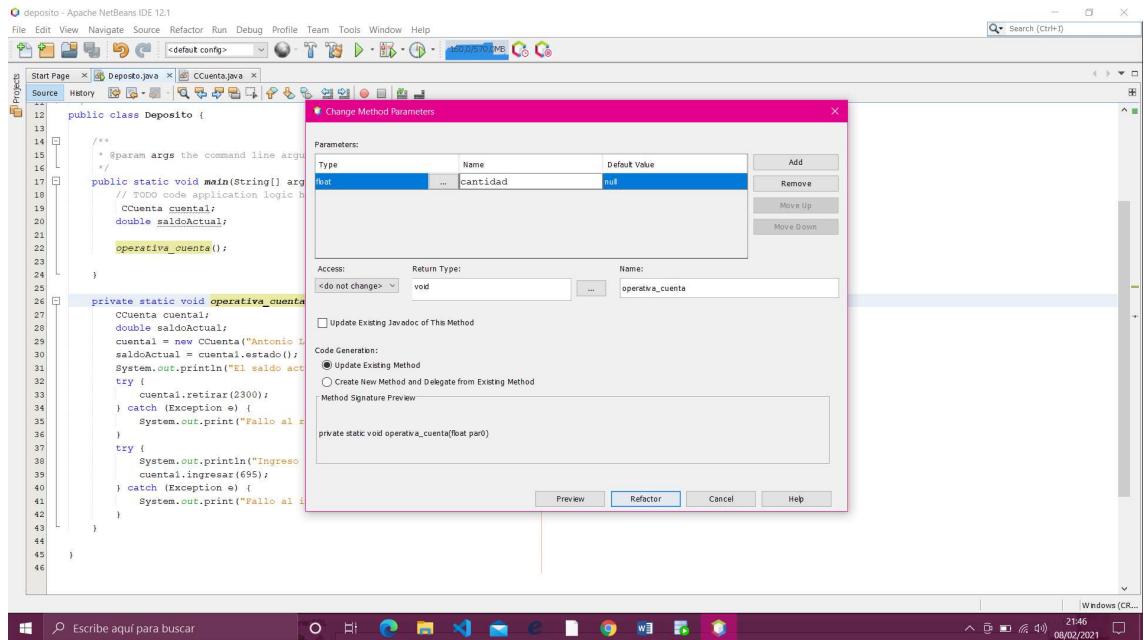
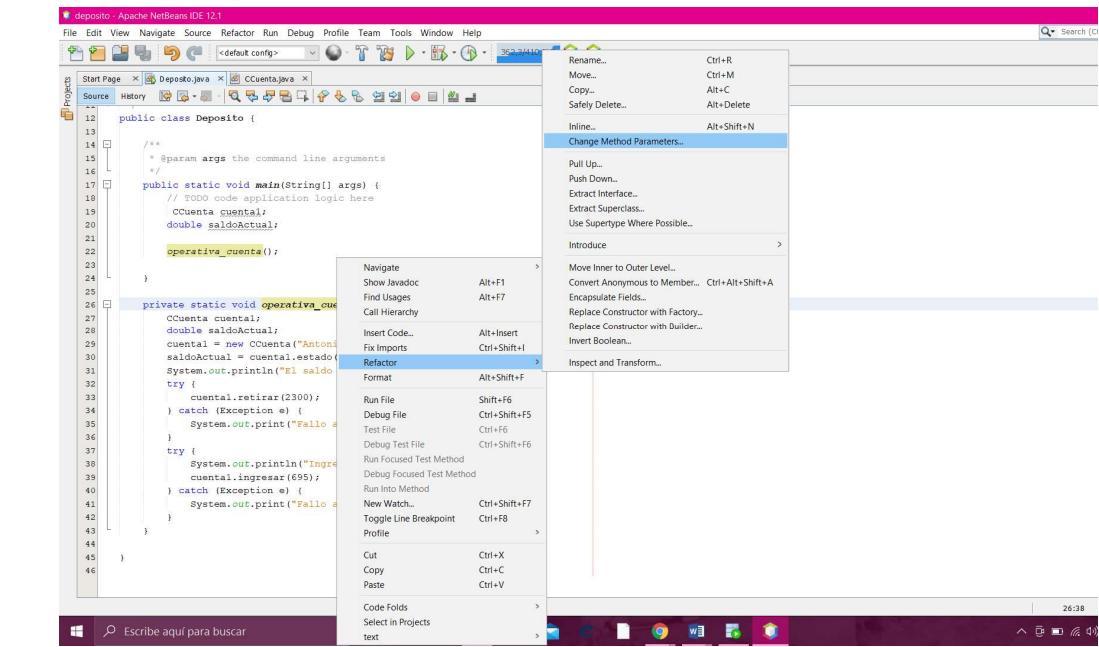
```

Los campos field visibility deben estar en privado y los accesos visibility en publico

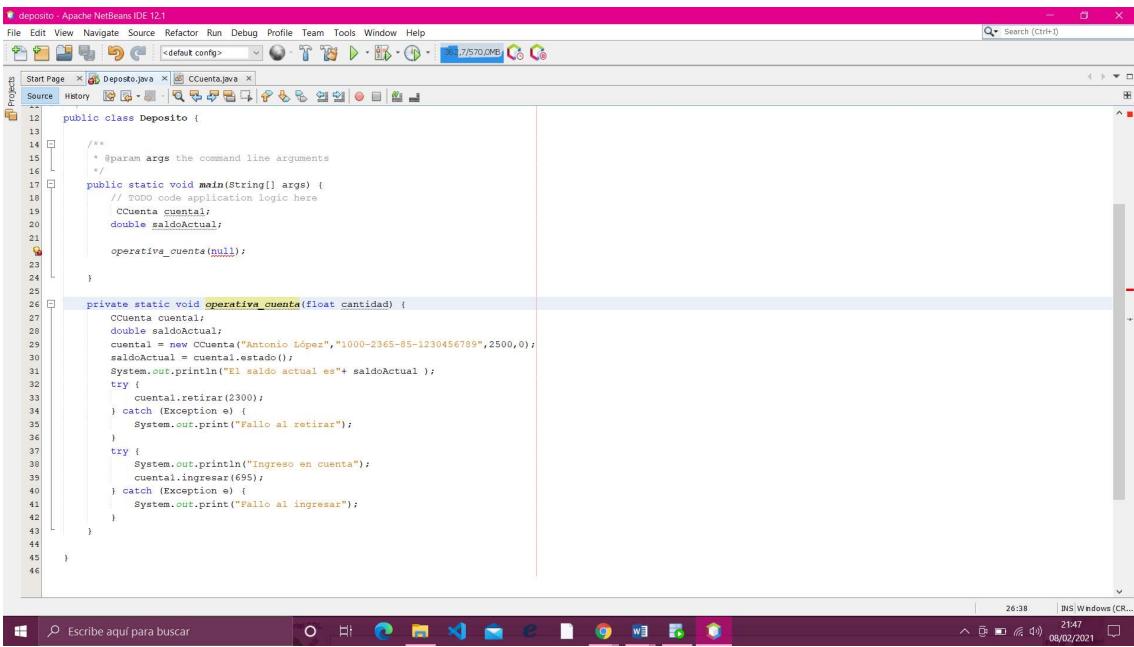


5. Añadir un nuevo parámetro al método operativa_cuenta, de nombre cantidad y de tipo float.

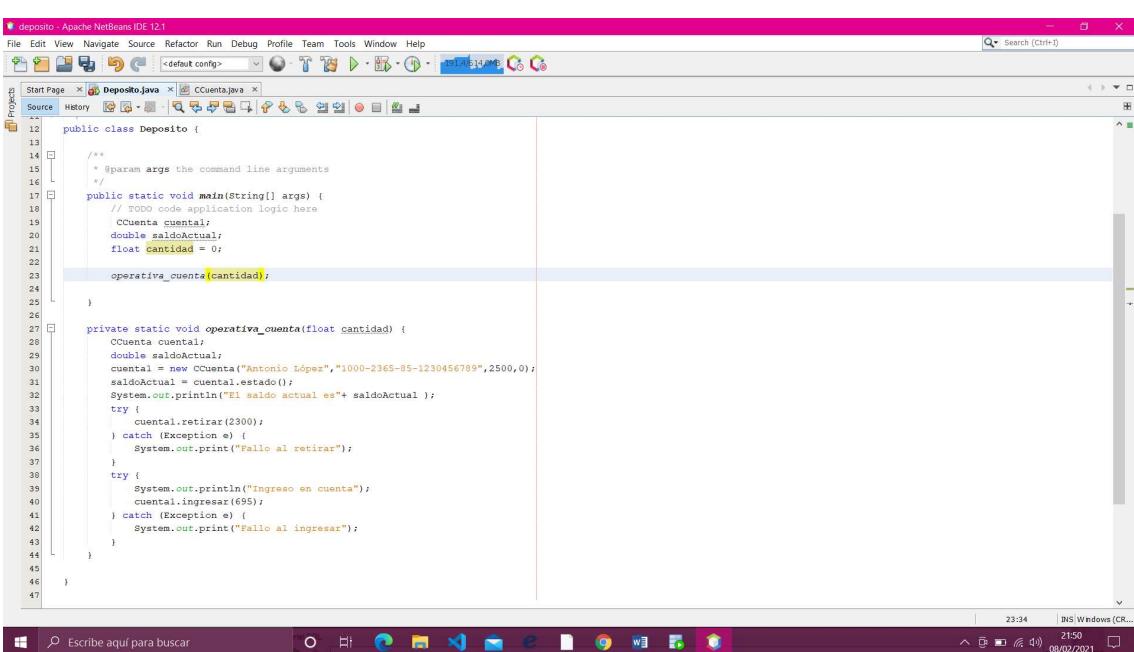
Refactor→change method parameters



Add→float y al parámetro lo nombramos cantidad



```
12 public class Deposito {
13
14     /**
15      * @param args the command line arguments
16     */
17     public static void main(String[] args) {
18         // TODO code application logic here
19         CCuenta cuenta;
20         double saldoActual;
21
22         operativa_cuenta(null);
23     }
24
25
26     private static void operativa_cuenta(float cantidad) {
27
28         CCuenta cuenta;
29         double saldoActual;
30         cuenta = new CCuenta("Antonio López", "1000-2365-85-1230456789", 2500, 0);
31         saldoActual = cuenta.estado();
32         System.out.println("El saldo actual es" + saldoActual );
33         try {
34             cuenta.retirar(2300);
35         } catch (Exception e) {
36             System.out.print("Fallo al retirar");
37         }
38         try {
39             System.out.println("Ingreso en cuenta");
40             cuenta.ingresar(695);
41         } catch (Exception e) {
42             System.out.print("Fallo al ingresar");
43         }
44     }
45
46 }
```

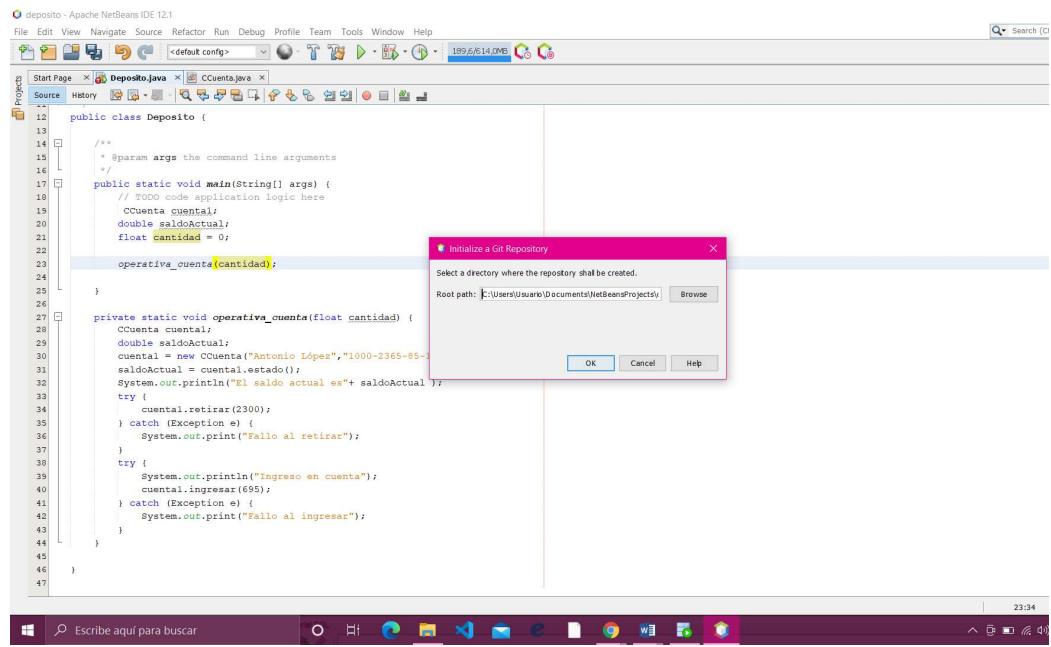
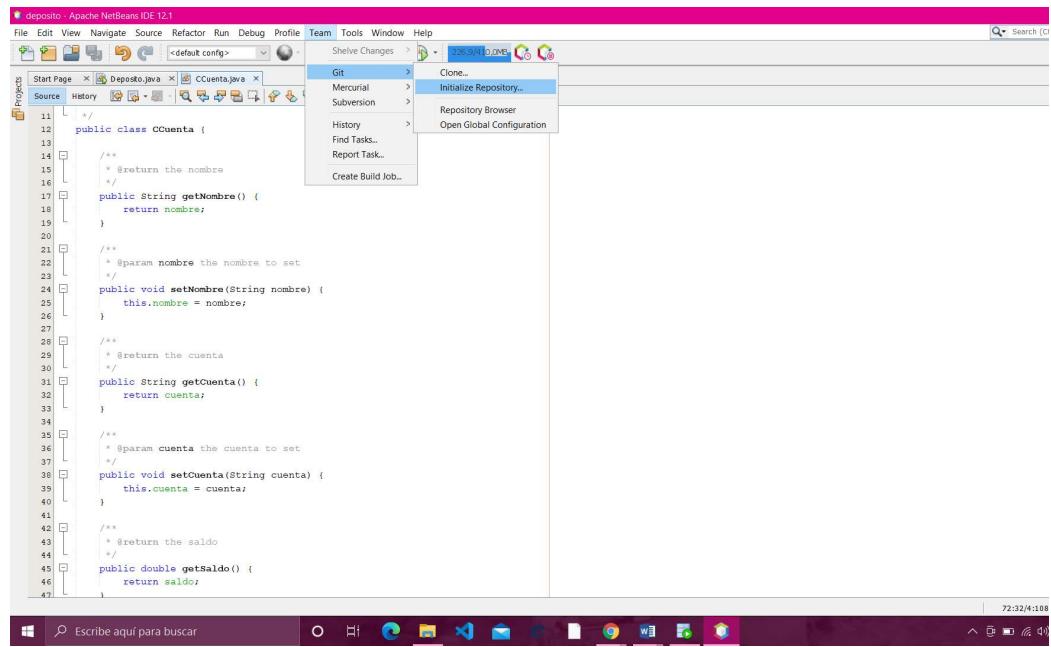


```
12 public class Deposito {
13
14     /**
15      * @param args the command line arguments
16     */
17     public static void main(String[] args) {
18         // TODO code application logic here
19         CCuenta cuenta;
20         double saldoActual;
21         float cantidad = 0;
22
23         operativa_cuenta(cantidad);
24     }
25
26
27     private static void operativa_cuenta(float cantidad) {
28
29         CCuenta cuenta;
30         double saldoActual;
31         cuenta = new CCuenta("Antonio López", "1000-2365-85-1230456789", 2500, 0);
32         saldoActual = cuenta.estado();
33         System.out.println("El saldo actual es" + saldoActual );
34         try {
35             cuenta.retirar(2300);
36         } catch (Exception e) {
37             System.out.print("Fallo al retirar");
38         }
39         try {
40             System.out.println("Ingreso en cuenta");
41             cuenta.ingresar(695);
42         } catch (Exception e) {
43             System.out.print("Fallo al ingresar");
44         }
45     }
46 }
```

GIT

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

Team→git→initialize repository



```

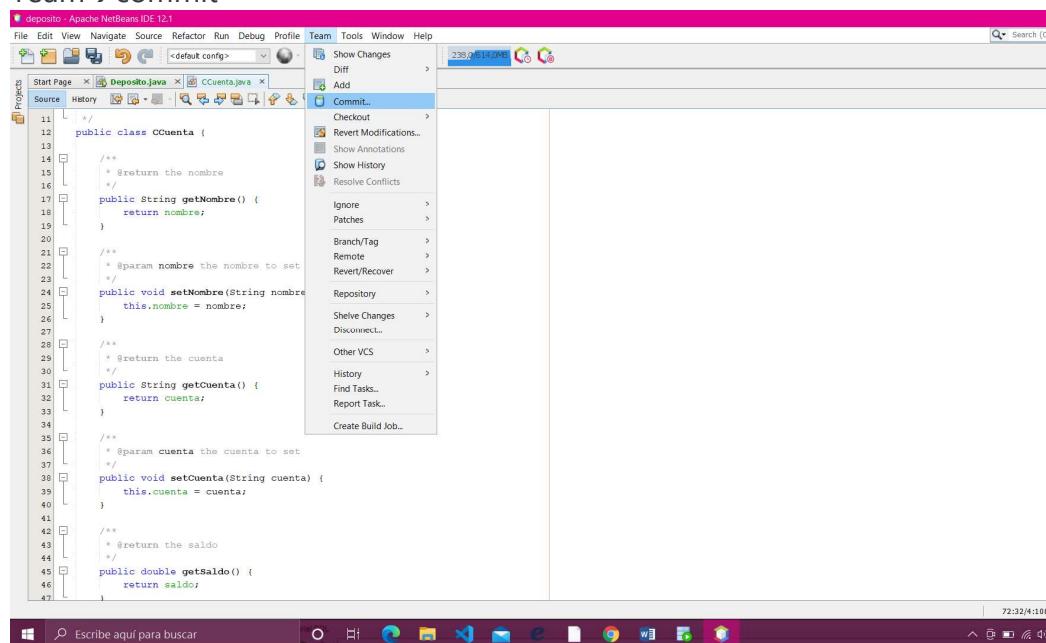
11  /*
12   * 
13   */
14  public class CCuenta {
15
16      /**
17      * @return the nombre
18      */
19      public String getNombre() {
20          return nombre;
21      }
22
23      /**
24      * @param nombre the nombre to set
25      */
26      public void setNombre(String nombre) {
27          this.nombre = nombre;
28      }
29
30      /**
31      * @return the cuenta
32      */
33      public String getCuenta() {
34          return cuenta;
35      }
36
37      /**
38      * @param cuenta the cuenta to set
39      */
40      public void setCuenta(String cuenta) {
41          this.cuenta = cuenta;
42      }
43
44      /**
45      * @return the saldo
46      */
47      public double getSaldo() {
48          return saldo;
49      }
50  }

```

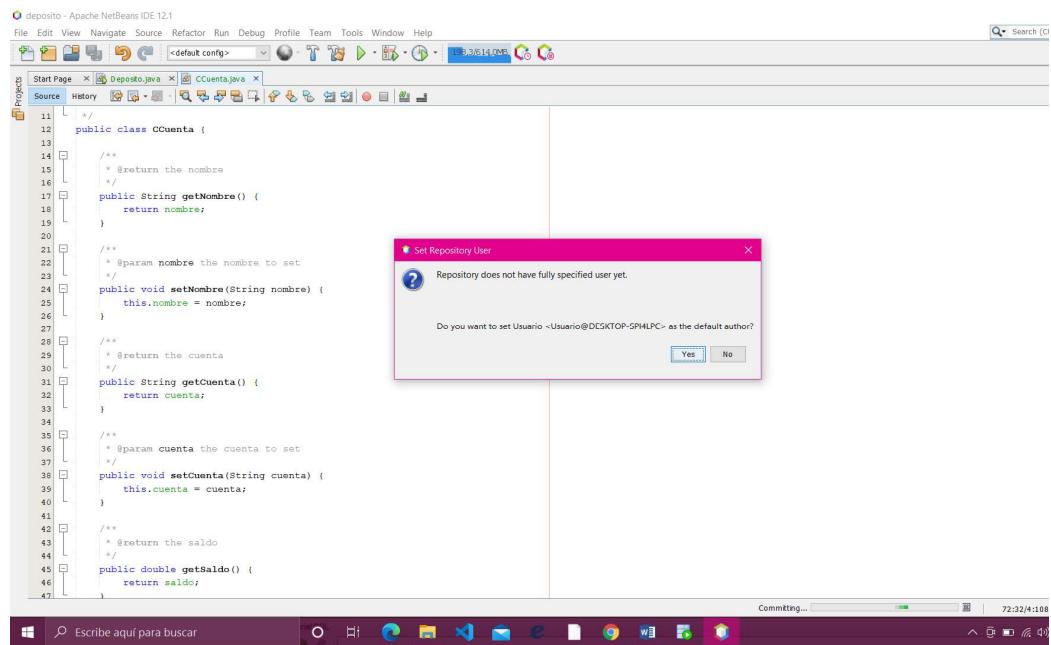
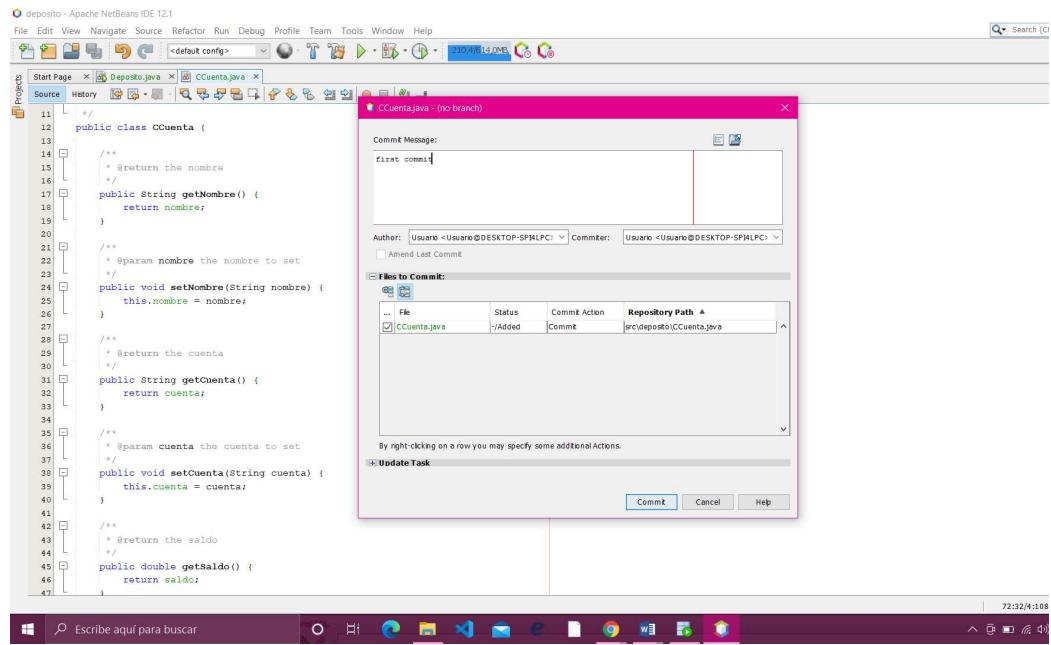
Podemos observar que el color de CCuenta y Main cambia a verde una vez inicializado

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

Team→commit



Escribimos el nombre del commit y seleccionamos commit



yes

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

Una vez descargado git del enlace facilitado en la unidad:

```

MINGW64:/c/Users/Usuario/documents/NetBeansProjects/deposito
.aliases
.dev_red
.favorites
Impresoras
Links
Mi Escritorio
Mis documentos
Mis estilos
Mi Escritorio.DAT
NTUSER.DAT{53b39e88-18c4-11ea-a811-000d3aa4692b}.TM.bif
NTUSER.DAT{53b39e88-18c4-11ea-a811-000d3aa4692b}.TMContainer000000000000000000000001
NTUSER.DAT{53b39e88-18c4-11ea-a811-000d3aa4692b}.TMContainer000000000000000000000002
regtrans-ms
derevive
drive
music
pictures
plantillas
recyclist
Sallyy Games
Searches
Sendto
videos
VirtualBox\ VMs
ansel
eject
ntuser.dat.LOG1
ntuser.dat.LOG2
ntuser.ini

UserarioDESKTOP-SP14LPC MINGW64 ~
$ cd documents
UserarioDESKTOP-SP14LPC MINGW64 ~/documents
$ dir
Documento.rtf           Mis música  Mis videos   Pilar( Alonso ) Suel1.docx  Plantillas\ personalizadas\ de Office  desktop.ini          ~wRL0422.tmp
Grabaciones\ de\ sonido  Mis\ imágenes NetBeansProjects  Pilar( Alonso ) Suel1.docx  VPPProjects  ~$tar\ Alonso\ Suel1.docx

UserarioDESKTOP-SP14LPC MINGW64 ~/documents/NetBeansProjects
$ dir
Examen Examen2 PROG04_Ejerc1 PROG04_Ejerc2 PROG04_Ejerc3 PROG04_Ejerc4 PROG04_Ejerc5 PROG05_Tarea PROG06_Tarea PilarAlonso TareaEntornosTest autoevaluacin deposito
UserarioDESKTOP-SP14LPC MINGW64 ~/documents/NetBeansProjects
$ cd deposito
UserarioDESKTOP-SP14LPC MINGW64 ~/documents/NetBeansProjects/deposito (master)
$ git Tag
* tag v29f3b06afab6af6ad59317c8881b3bd1 {HEAD -> master}
Author: Userario <userarioDESKTOP-SP14LPC>
Date:  Mon Feb 8 21:58:27 2021 +0100

  first commit
UserarioDESKTOP-SP14LPC MINGW64 ~/documents/NetBeansProjects/deposito (master)

```

JAVADOC

1. Insertar comentarios JavaDoc en la clase CCuenta.

Para insertar comentarios se añade `/** +enter`

```

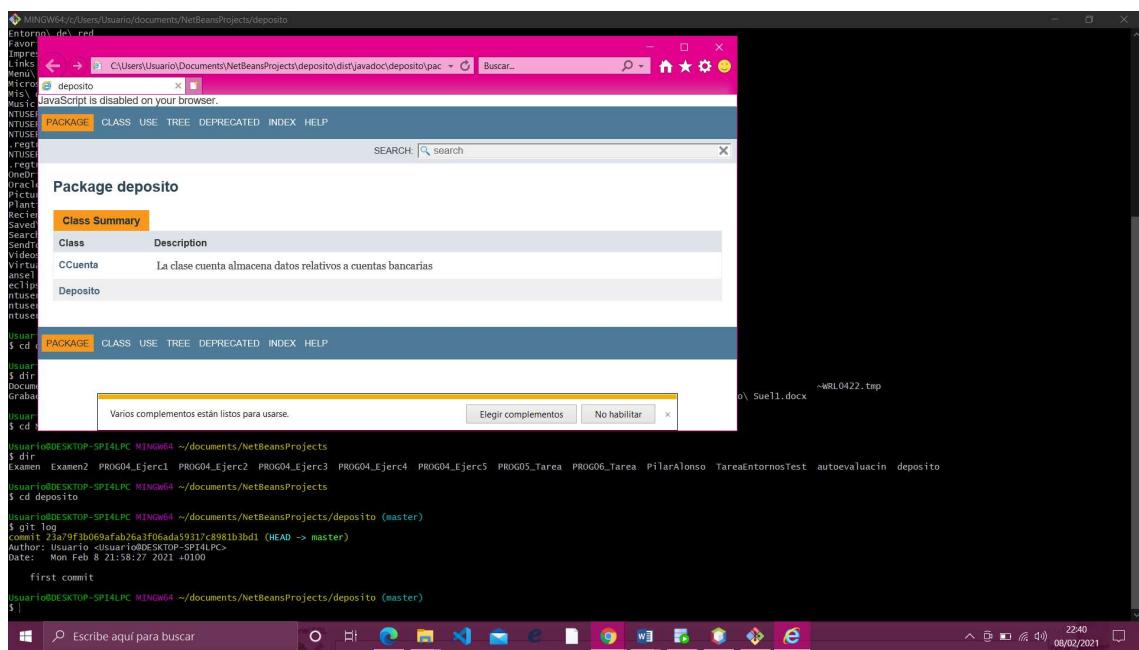
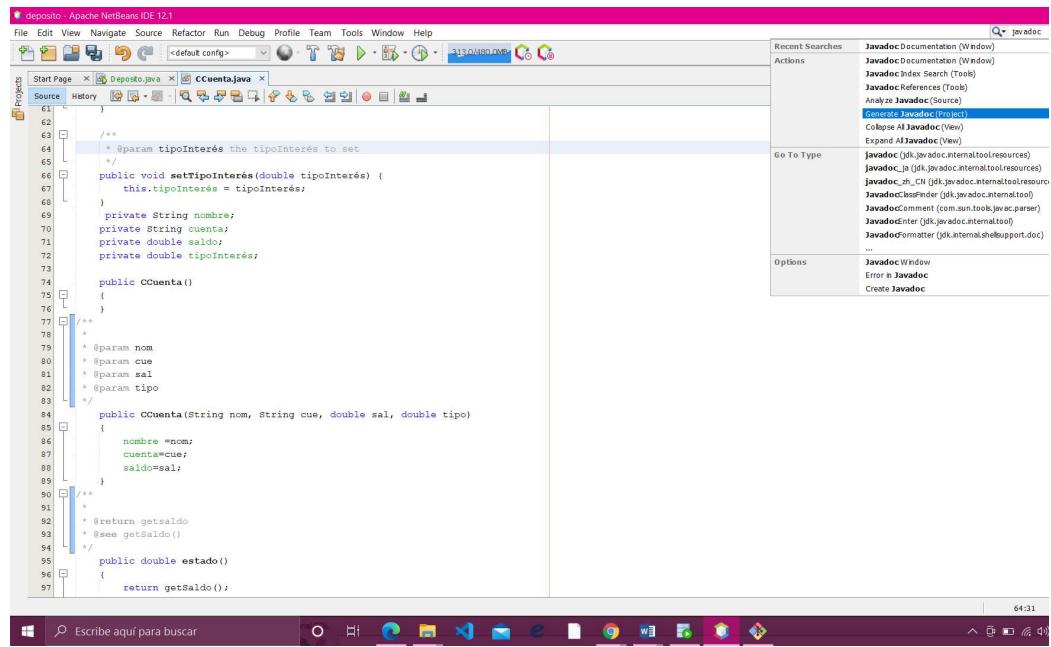
deposito - Apache NetBeans IDE 12.1
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Start Page Deposito.java CCuenta.java
Source History
2 /**
3 * To change this license header, choose License Headers in Project Properties.
4 * To change this template file, choose Tools | Templates
5 * and open the template in the editor.
6
7 package deposito;
8
9 /**
10 * La clase cuenta almacena datos relativos a cuentas bancarias
11 * @author Pilar Alonso Suela
12
13 public class CCuenta {
14
15     /**
16     * @return the nombre
17     */
18     public String getNombre() {
19         return nombre;
20     }
21
22     /**
23     * @param nombre the nombre to set
24     */
25     public void setNombre(String nombre) {
26         this.nombre = nombre;
27     }
28
29     /**
30     * @return the cuenta
31     */
32     public String getCuenta() {
33         return cuenta;
34     }
35
36     /**
37     * @param cuenta the cuenta to set
38     */
39     public void setCuenta(String cuenta) {
40
41     }
42 }

```

```
90     /**
91      * 
92      * @return getSaldo
93      * @see getSaldo()
94      */
95     public double estado()
96     {
97         return getSaldo();
98     }
99 
100 /**
101  * metodo que controla que no se ingrese una cantidad negativa
102  * @param cantidad
103  * @throws Exception
104  */
105 public void ingresar(double cantidad) throws Exception
106 {
107     if (cantidad<0)
108         throw new Exception("No se puede ingresar una cantidad negativa");
109     setSaldo(getSaldo() + cantidad);
110 }
111 /**
112  * metodo que controla que se retire cantidad siempre que no sea negativa o que haya suficiente saldo
113  * @param cantidad
114  * @throws Exception
115  */
116 public void retirar(double cantidad) throws Exception
117 {
118     if (cantidad <= 0)
119         throw new Exception ("No se puede retirar una cantidad negativa");
120     if (estado()< cantidad)
121         throw new Exception ("No se hay suficiente saldo");
122     setSaldo(getSaldo() - cantidad);
123 }
124 }
125 
```

```
61     )
62 
63 /**
64  * @param tipoInterés the tipoInterés to set
65  */
66 public void setTipoInterés(double tipoInterés) {
67     this.tipoInterés = tipoInterés;
68 }
69 private String nombre;
70 private String cuenta;
71 private double saldo;
72 private double tipoInterés;
73 
74 public CCuenta()
75 {
76 }
77 /**
78  * 
79  * @param nom
80  * @param cue
81  * @param sal
82  * @param tipo
83  */
84 public CCuenta(String nom, String cue, double sal, double tipo)
85 {
86     nombre =nom;
87     cuenta=cue;
88     saldo=sal;
89 }
90 /**
91  * 
92  * @return getSaldo
93  * @see getSaldo()
94  */
95 public double estado()
96 {
97     return getSaldo();
98 }
```

2. Generar documentación JavaDoc para todo el proyecto y comprueba que abarca todos los métodos y atributos de la clase CCuenta.



The screenshot shows a Java class documentation page for `CCuenta`. The page is titled "CCuenta" and includes tabs for PACKAGE, CLASS, USE, TREE, DEPRECATED, INDEX, and HELP. The CLASS tab is selected. A search bar at the top right contains the placeholder "Buscar...". Below the tabs, there are buttons for All Methods, Instance Methods, and Concrete Methods. The "All Methods" button is highlighted.

All Methods

Modifier and Type	Method	Description
double	<code>estado()</code>	
java.lang.String	<code>getCuenta()</code>	
java.lang.String	<code>getNombre()</code>	
double	<code>getSaldo()</code>	
double	<code>getTipoInterés()</code>	
void	<code>ingresar(double cantidad)</code>	metodo que controla que no se ingrese una cantidad negativa
void	<code>retirar(double cantidad)</code>	metodo que controla que se retire cantidad siempre que no sea negativa o que haya suficiente saldo
void	<code>setCuenta(java.lang.String cuenta)</code>	
void	<code>setNombre(java.lang.String nombre)</code>	
void	<code>setSaldo(double saldo)</code>	
void	<code>setTipoInterés(double tipoInterés)</code>	

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 Details

CCuenta

```
public CCuenta()
```

At the bottom of the window, there is a taskbar with icons for various applications like File Explorer, Task View, Mail, Edge, and others. The system tray shows the date and time as 08/02/2021 22:44.

1. .

