

TAREA PARA ED04.

Detalles de la tarea de esta unidad.

Enunciado.

En el proyecto **Java "Deposito"**, hay definida una Clase llamada **CCUENTA**, que tiene una serie de atributos y métodos. El proyecto cuenta asimismo con una Clase **MAIN**, donde se hace uso de la clase descrita. Basándonos en ese proyecto, vamos a realizar las siguientes actividades.

Enlace Repositorio de la Actividad:

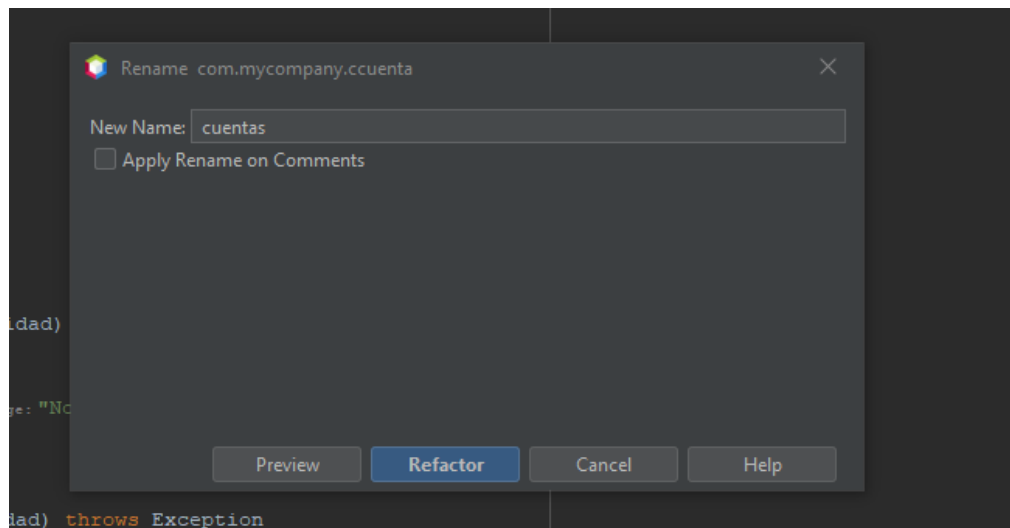
<https://github.com/TigXyz/TareaED04/tree/master/src/main/java/cuentas>

REFACTORIZACIÓN

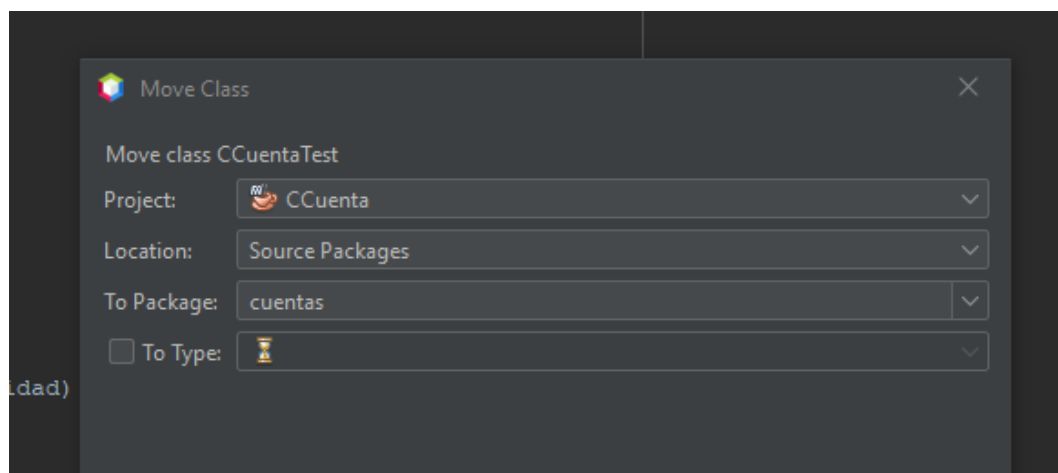
1. *Las clases deberán formar parte del paquete cuentas.*

Renombramos el paquete a cuentas haciendo clic derecho sobre el paquete

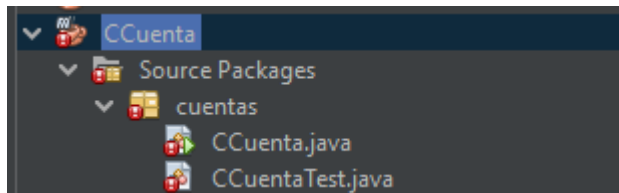
->Refactor->Rename y escribimos "Cuentas".



Y movemos la otra clase del proyecto al mismo paquete:

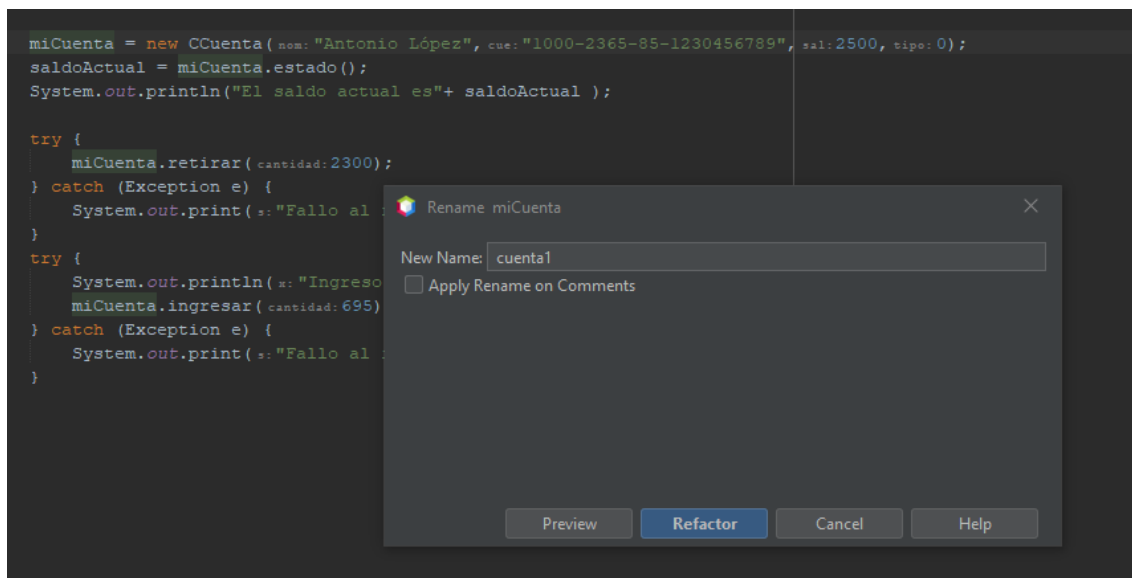


Ya están ambas clases en el paquete cuentas del proyecto CCuenta:



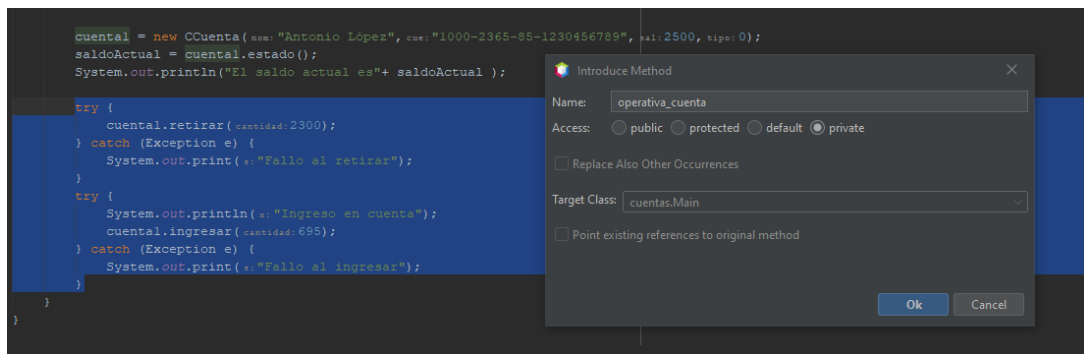
2. Cambiar el nombre de la variable "miCuenta" por "cuenta1".

Selecciono la variable a cambiar y pulsamos click derecho sobre ella. Refactor -> Rename y al pulsar Refactor cambiará el nombre de la variable en todas las clases.



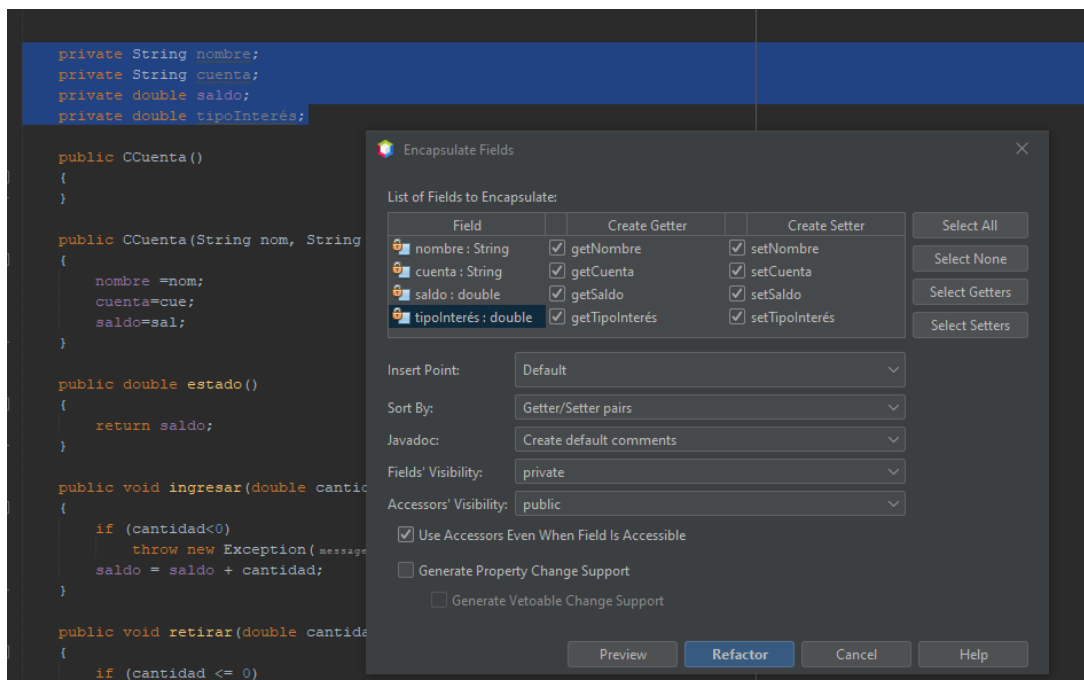
3. Introducir el método operativa_cuenta, que englobe las sentencias de la clase Main que operan con el objeto cuenta1.

Para introducir el método "operativa_cuenta", selecciono los dos bloques try-catch de main y en el menú Refactor->Introduce->Method e introducimos el nombre.



4. Encapsular los atributos de la clase CCuenta.

Seleccionamos los atributos, clic derecho -> Refactor -> Encapsulate Fields

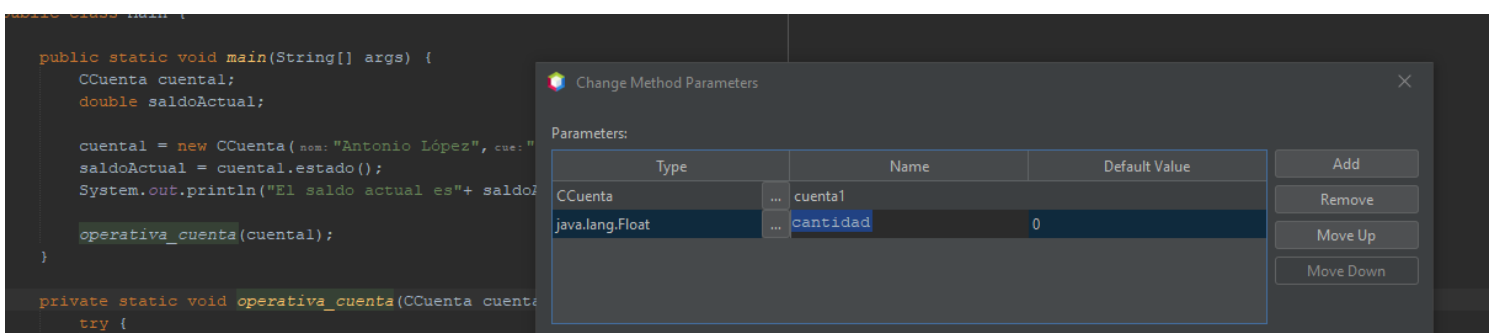


Marcamos las casillas correspondientes y esto nos añadirá los métodos setters y getters correspondientes para la encapsulación, así como el modificador de acceso private en el caso que no estuviera.

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

Clic derecho sobre operativa_cuenta y pulsamos en Refactor otra vez y seleccionamos Change Method Parameters.

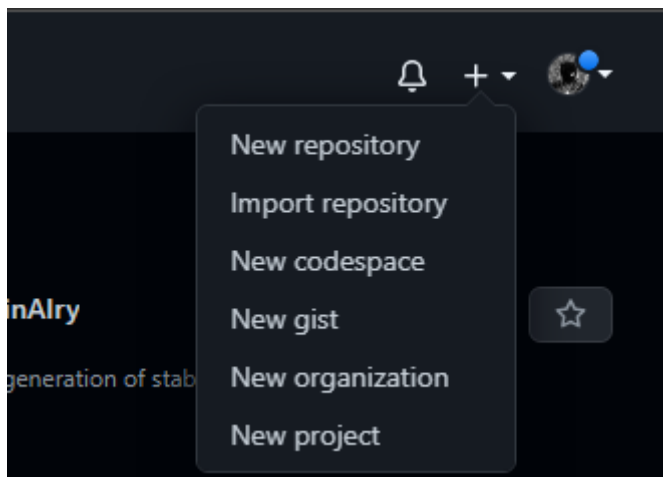
Pulsamos ADD y añadimos los parámetros requeridos. Pulsamos sobre Reestructurar y ya tendremos el nuevo parámetro añadido al método.



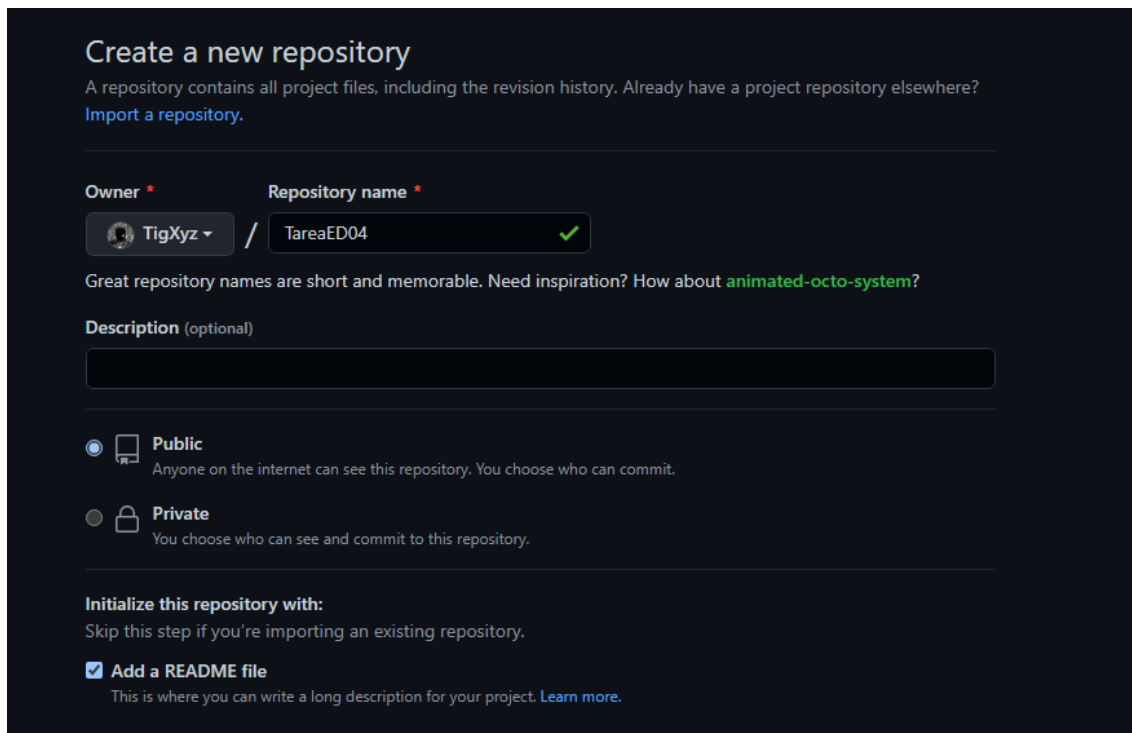
GIT

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

Hacemos click en "+" de nuestra cuenta de GutHub y pulsamos en New Repository




Añadimos el nombre, lo hacemos público y seleccionamos que nos añada un README para escribir una descripción del repositorio



Create a new repository


A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)


Owner * **Repository name ***

 TigXyz / TareaED04 ✓

Great repository names are short and memorable. Need inspiration? How about **animated-octo-system**?

Description (optional)

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.

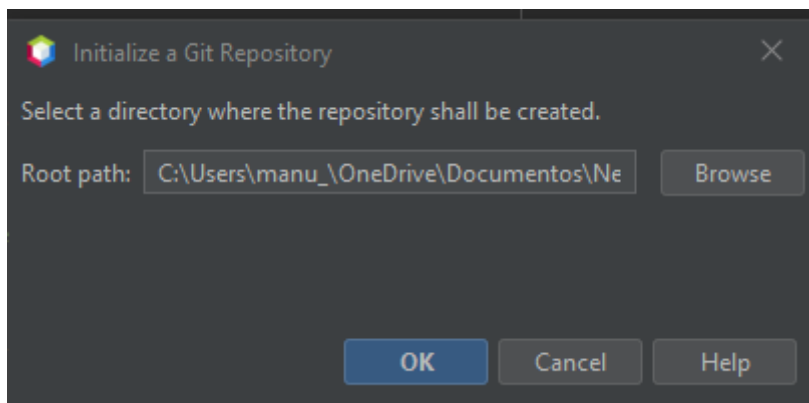
☐  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

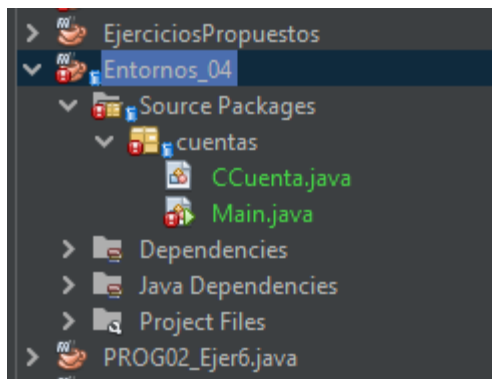
☒ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)

En NetBeans vamos al proyecto -> Team -> Git -> Inicializar repositorio y

Seleccionamos la ruta donde vamos a crear el repositorio:

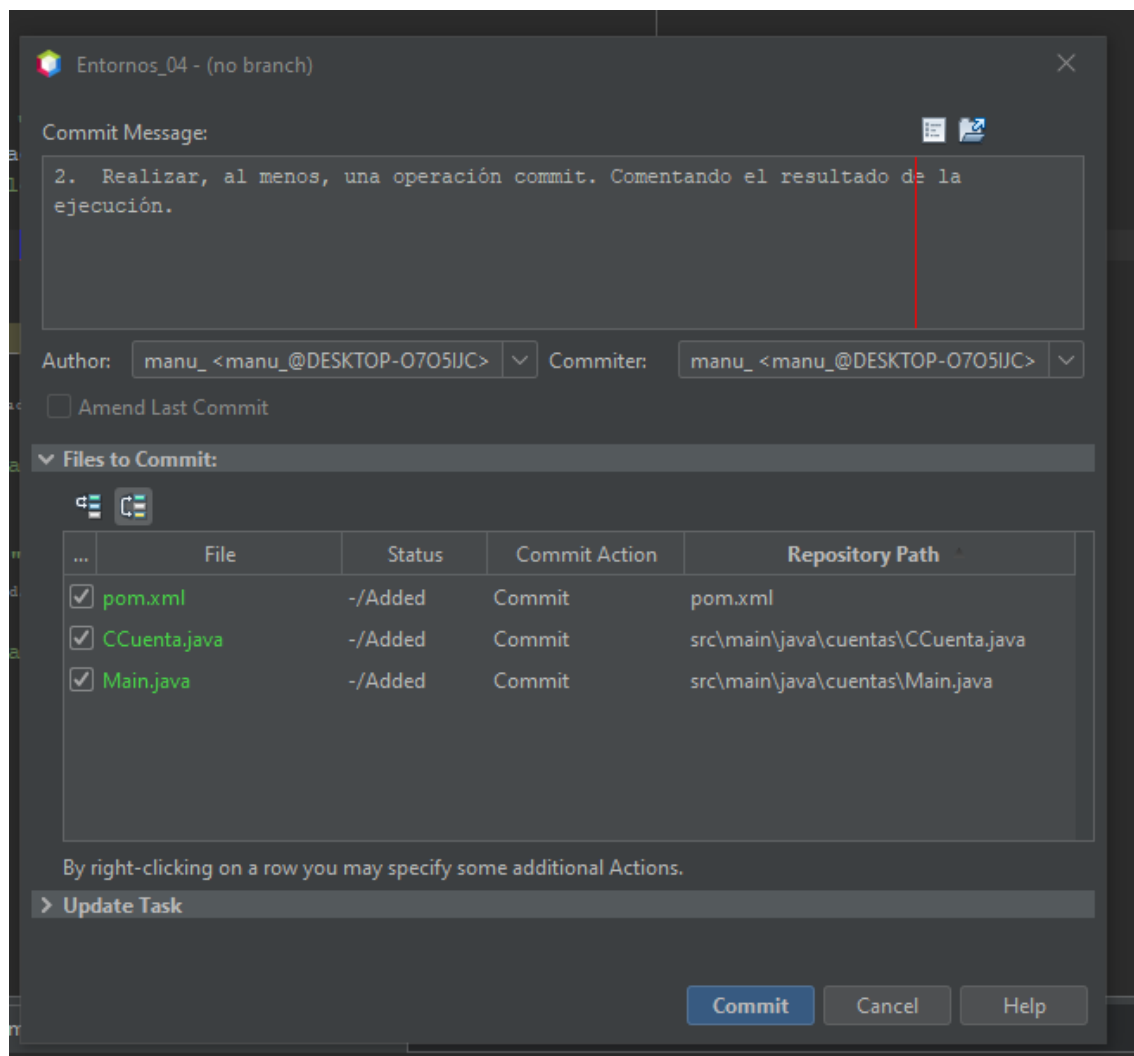


Observamos que el estado de nuestro proyecto ha cambiado:

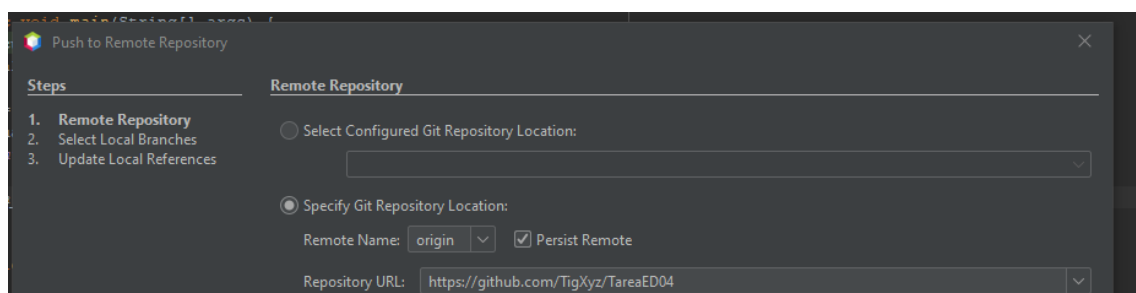


2. Realizar, al menos, una operación commit. Comentando el resultado de la ejecución.

Hacemos un git Commit con clic derecho sobre el proyecto->Git->Commit

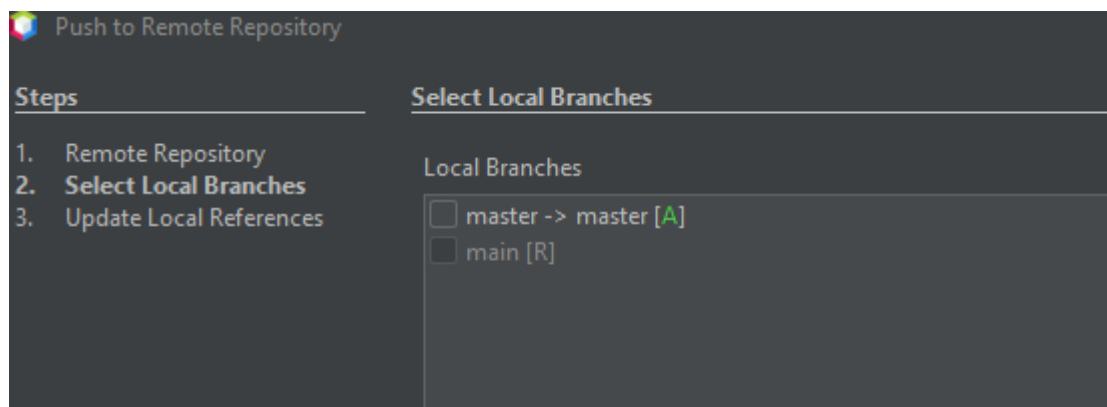


Y hacemos un push para mandarlo **con clic derecho sobre el proyecto -> Git -> Remote -> Push**

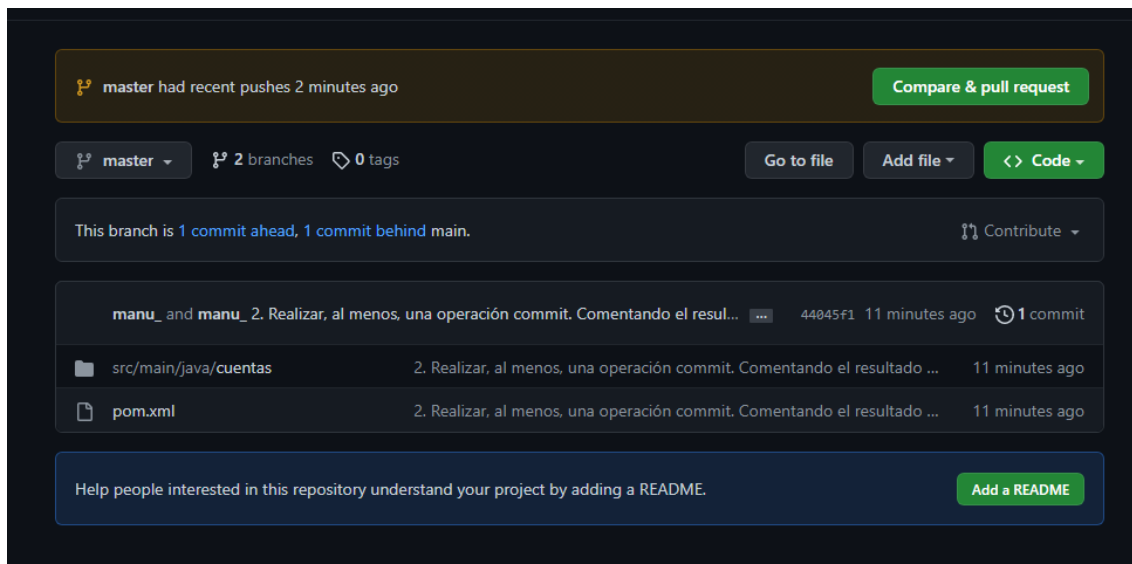


Insertamos la URL del repositorio que hemos creado en GitHub con las credenciales de nuestra cuenta. Es importante que en "Password" pongamos un token generado y no la contraseña de nuestra cuenta GitHub.

Como no hemos creado una rama diferente, tenemos la Master por defecto:



Resultado:



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


```
Commit Log
revision : 44045f126cdacee84774a08e0436dda9fcl49088
author   : manu_ <manu_@DESKTOP-0705IJC>
date     : 24 ene 2023, 17:32:08
summary  : 2. Realizar, al menos, una operación commit. Comentando el resultado de la ejecución.

INFO: End of Commit

==[IDE]== 24 ene 2023, 17:32:08 Committing... finished.
==[IDE]== 24 ene 2023, 17:36:39 Pushing - Entornos_04
git branch
git remote -v
setting up remote: origin
git submodule status
git push https://github.com/TigXyz/TareaED04 refs/heads/master:refs/heads/master
git push https://github.com/TigXyz/TareaED04 refs/heads/master:refs/heads/master
git push https://github.com/TigXyz/TareaED04 refs/heads/master:refs/heads/master
Create a pull request for 'master' on GitHub by visiting:
  https://github.com/TigXyz/TareaED04/pull/new/master

Remote Repository Updates
Branch Add : master
Id         : 44045f126cdacee84774a08e0436dda9fcl49088
Result     : OK

Local Repository Updates
Branch Add : origin/master
Id         : 44045f126cdacee84774a08e0436dda9fcl49088
Result     : NEW

==[IDE]== 24 ene 2023, 17:41:11 Setting Tracked Branch
git branch --set-upstream-to origin/master master
Branch master set to track origin/master

==[IDE]== 24 ene 2023, 17:41:11 Pushing - Entornos_04 finished.
git branch --set-upstream-to origin/master master
Branch "master" marked to track branch "origin/master"
==[IDE]== 24 ene 2023, 17:41:11 Setting Tracked Branch finished.
```

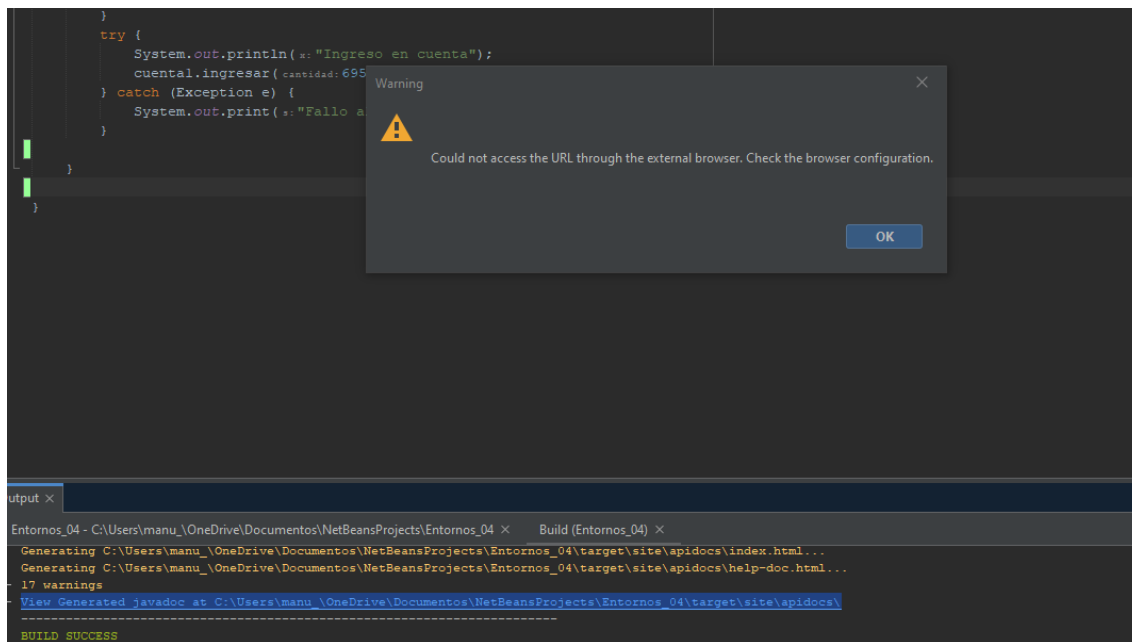
JAVADOC

1. Insertar comentarios JavaDoc en la clase CCuenta.

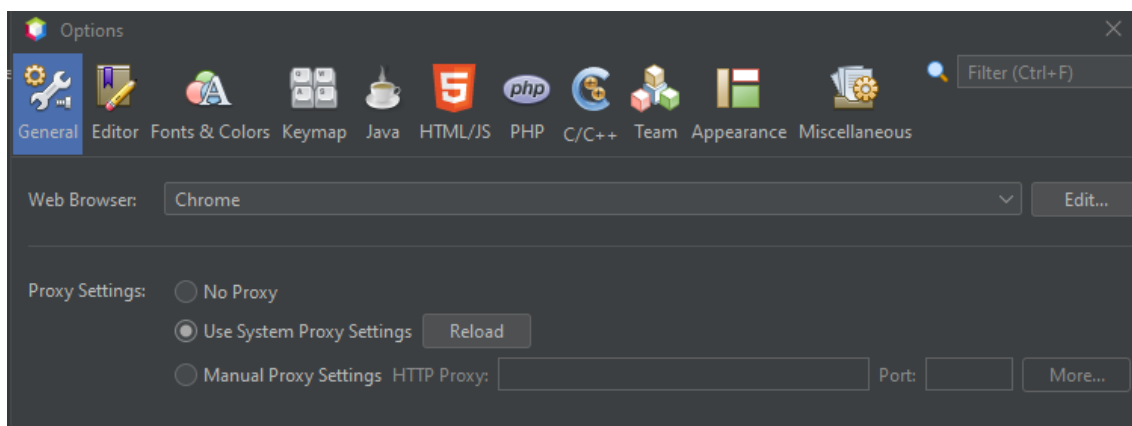
Seleccionamos la clase, clic derecho en la clase y selecciona "Run" -> "Generate JavaDoc Comment". Se abrirá una ventana de comentario con una estructura básica de JavaDoc.

Podemos utilizar "@param" y "@return" para documentar los parámetros y el valor de retorno de los métodos.

Si al hacer clic para visualizar el JavaDoc nos da el siguiente error:



Tools -> Options -> Seleccionamos nuestro buscador ya que NetBeans no es capaz de saber cuál es y abrimos el enlace de nuevo:



PACKAGE

CLASS

USE

TREE

INDEX

HELP

SUMMARY: NESTED | FIELD | CONSTR | METHODDETAIL: FIELD | CONSTR | METHOD

Package

cuentas

Class CCuenta

java.lang.Object[Ⓜ]
cuentas.CCuenta

public class CCuenta
extends Object[Ⓜ]

Constructor Summary

Constructors

Constructor	Description
CCuenta()	
CCuenta(String [Ⓜ] nom, String [Ⓜ] cue, double sal, double tipo)	

Method Summary

All Methods

Instance Methods

Concrete Methods

Modifier and Type	Method	Description
double	estado()	
String [Ⓜ]	getCuenta()	
String [Ⓜ]	getNombre()	
double	getSaldo()	
double	getTipoInterés()	
void	ingresar(double cantidad)	
void	retirar(double cantidad)	
void	setCuenta(String [Ⓜ] cuenta)	
void	setNombre(String [Ⓜ] nombre)	
void	setSaldo(double saldo)	
void	setTipoInterés(double tipoInterés)	

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

All Methods	Instance Methods	Concrete Methods
Modifier and Type	Method	Description
double	estado()	
String [Ⓔ]	getCuenta()	
String [Ⓔ]	getNombre()	
double	getSaldo()	
double	getTipoInterés()	
void	ingresar(double cantidad)	
void	retirar(double cantidad)	
void	setCuenta(String [Ⓔ] cuenta)	
void	setNombre(String [Ⓔ] nombre)	
void	setSaldo(double saldo)	
void	setTipoInterés(double tipoInterés)	
Methods inherited from class java.lang.Object [Ⓔ]		
clone [Ⓔ] , equals [Ⓔ] , finalize [Ⓔ] , getClass [Ⓔ] , hashCode [Ⓔ] , notify [Ⓔ] , notifyAll [Ⓔ] , toString [Ⓔ] , wait [Ⓔ] , wait [Ⓔ] , wait [Ⓔ]		

Constructor Details
CCuenta
public CCuenta()

