TAREA 2

Eres un desarrollador y quieres crear una aplicación de escritorio en Java llamada"MiAppJava". Para mantener un control adecuado de versiones, usarás Git. Seguirás unflujo de trabajo que incluirá ramas para nuevas funciones, commits organizados y gestión de versiones.

Control adecuado de versiones de Git

Índice

[1. Crea un nuevo directorio para tu proyecto. 2](#_Toc195258253)

[2. Inicializa un repositorio Git dentro del directorio. 3](#_Toc195258254)

[3. Crea la estructura básica del proyecto. 4](#_Toc195258255)

[4. Abre Main.java y agrega código. 6](#_Toc195258256)

[5. Agregalo al área de preparación y realiza el primer commit. 6](#_Toc195258257)

[6. Crea una nueva rama llamada modificacion1. 7](#_Toc195258258)

[7. Cambiate a la nueva rama. 7](#_Toc195258259)

[8. Crea una nueva clase que aporte una nueva funcionalidad. 8](#_Toc195258260)

[9. Modifica Main.java para usar la nueva funcionalidad. 10](#_Toc195258261)

[10. Agrega y confirma los cambios en la rama. 10](#_Toc195258262)

[11. Cambia de vuelta a la rama principal (master). 11](#_Toc195258263)

[12. Fusiona la rama modificación1 con master. 11](#_Toc195258264)

[13. Elimina la rama modificacion1, ya que sus cambios fueron fusionados. 11](#_Toc195258265)

[14. Crea una etiqueta para marcar la versión 1.0 de la aplicación. 12](#_Toc195258266)

[15. Verifica que la etiqueta ha sido creada. 12](#_Toc195258267)

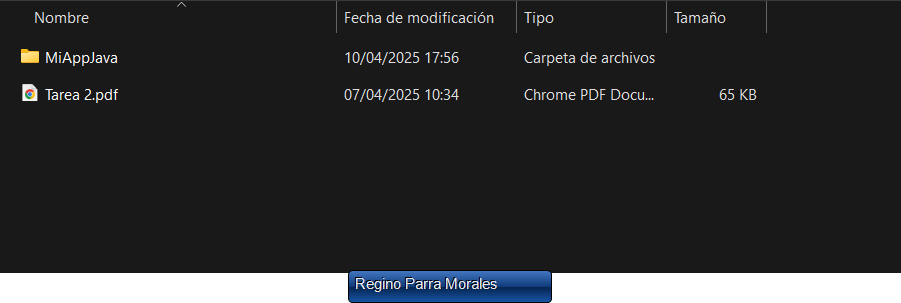
[16. Edita la última clase creada para hacer un commit con un error. 13](#_Toc195258268)

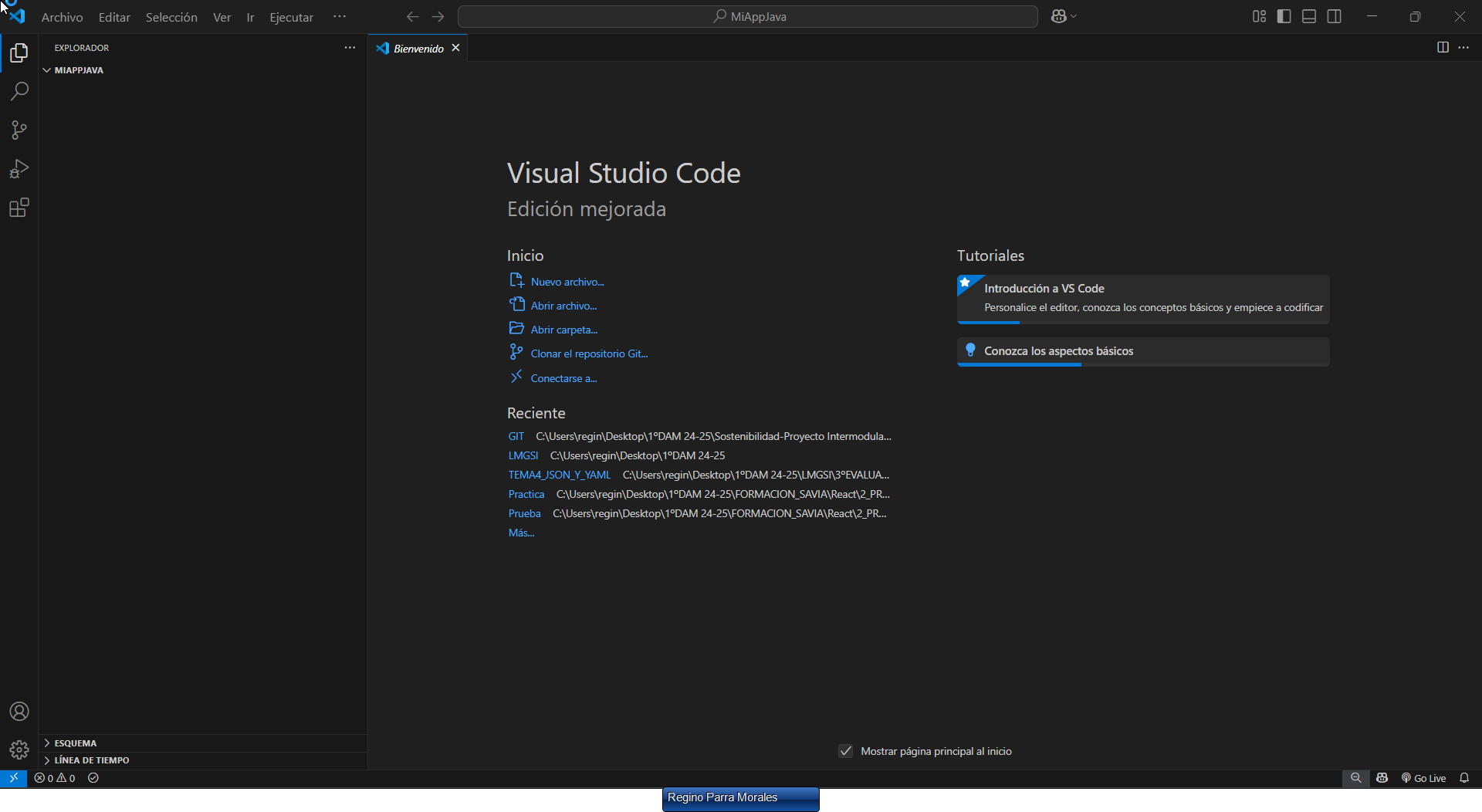
[17. Agrega y confirma el commit con el error. 14](#_Toc195258269)

[18. Revierte el último commit sin eliminarlo del historial. 14](#_Toc195258270)

# Crea un nuevo directorio para tu proyecto.

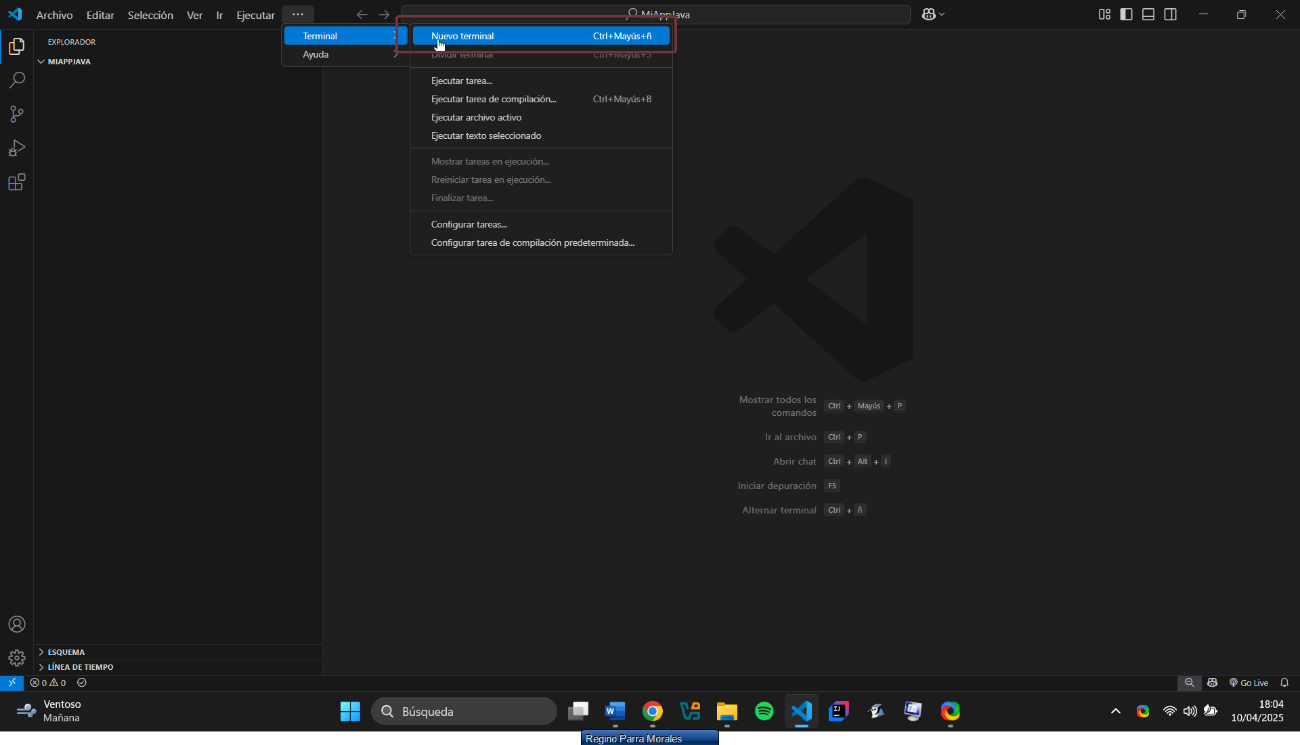
Creamos la carpeta de MiAppJava



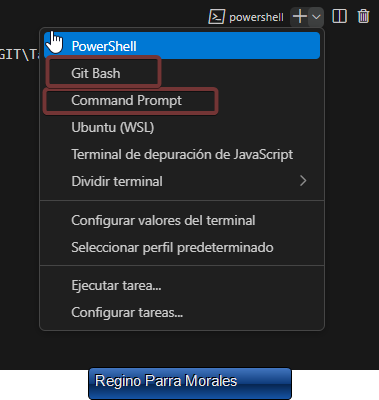
La metemos en Visual Studio Code

# Inicializa un repositorio Git dentro del directorio.

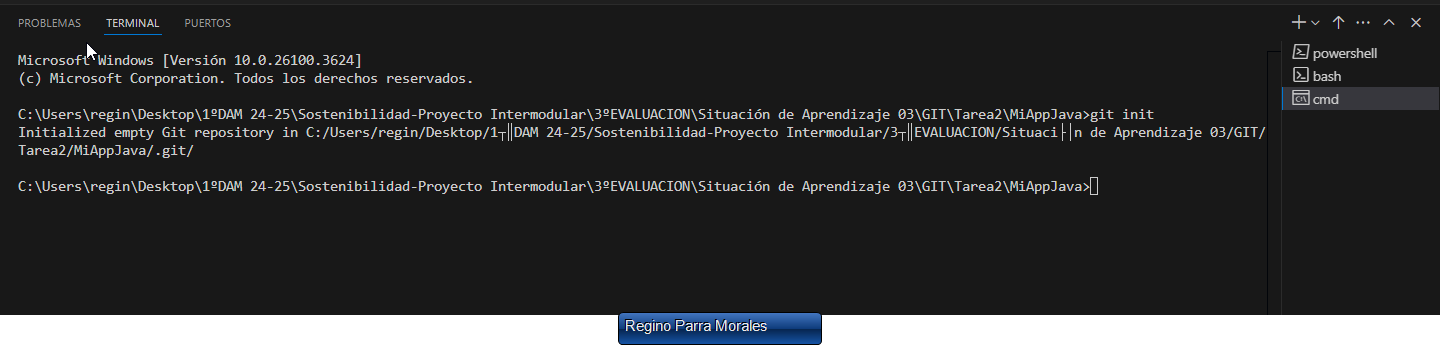
Nos vamos a la línea de comandos, en los 3 puntos arriba, terminal y nueva terminal



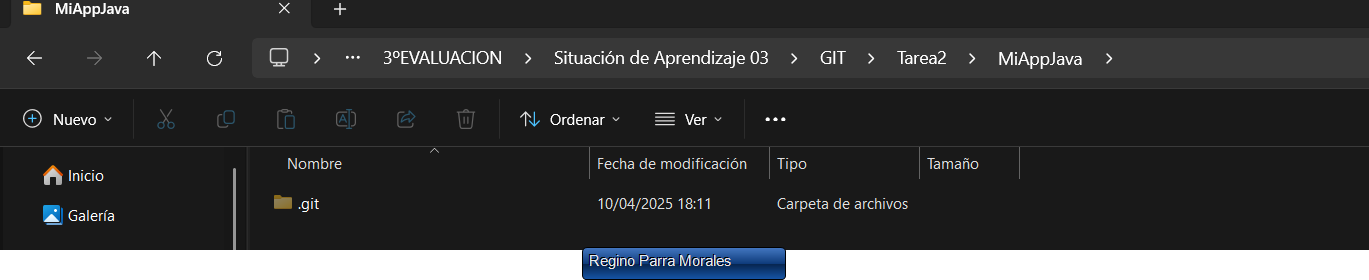
Podemos abrir la terminal o con git bash o con command prompt (CMD)en mi caso usare el CMD



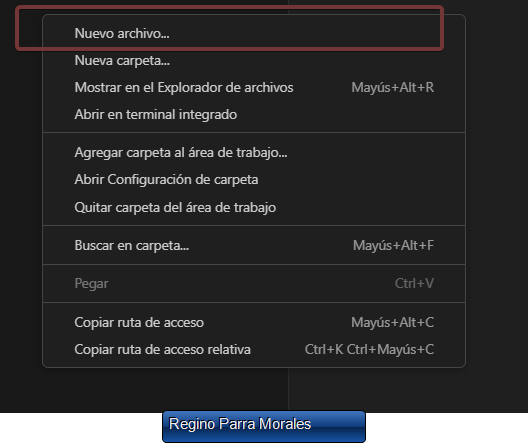
En la línea de comandos escribimos git init y se nos creara una carpeta oculta, con este comando inicializamos git



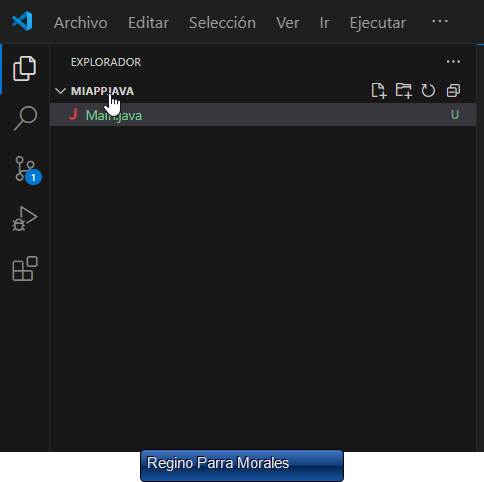
En la carpeta de MiAppJava, se podrá ver que se ha creado la carpeta



# Crea la estructura básica del proyecto.



Creamos el archivo Main.java



# Abre Main.java y agrega código.

Escribimos en el archivo Main.java, ese código :

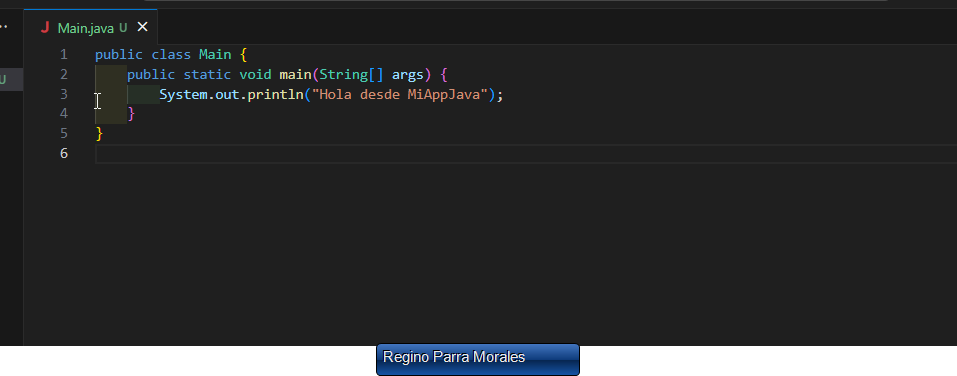
**public class Main {**

**public static void main(String[] args) {**

**System.out.println("Hola desde MiAppJava");**

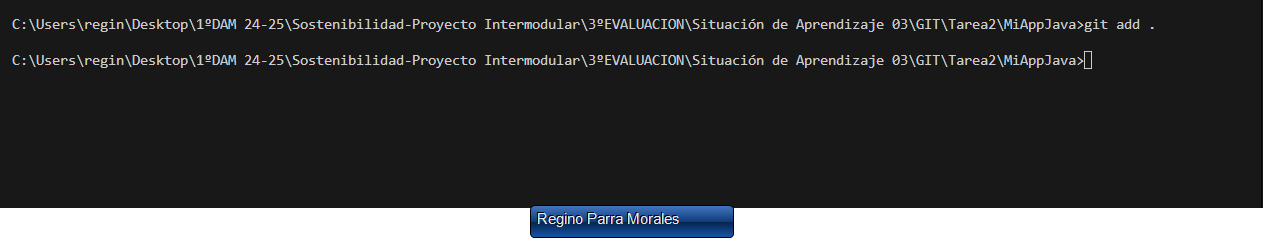
**}**

**}**



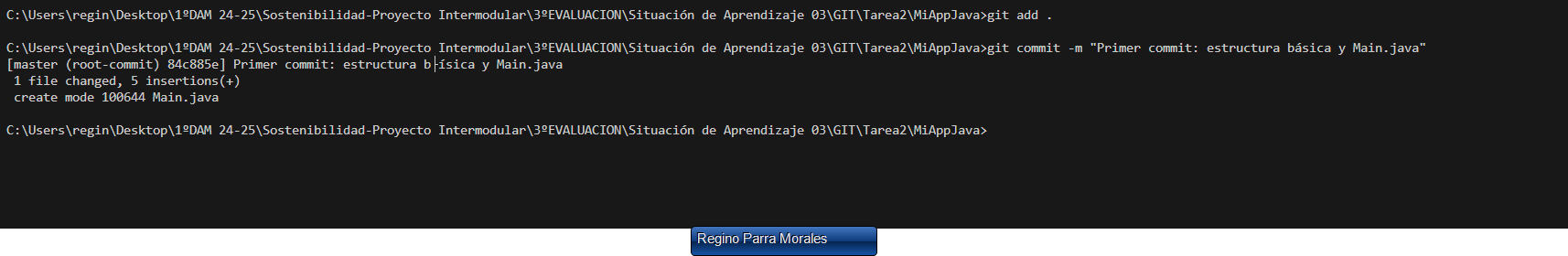
# Agregalo al área de preparación y realiza el primer commit.

Hacemos un : git add . , para agregar todo lo que este en la carpeta,en este caso el Main.java



Hacemos un :

**git commit -m "Primer commit: estructura básica y Main.java"**



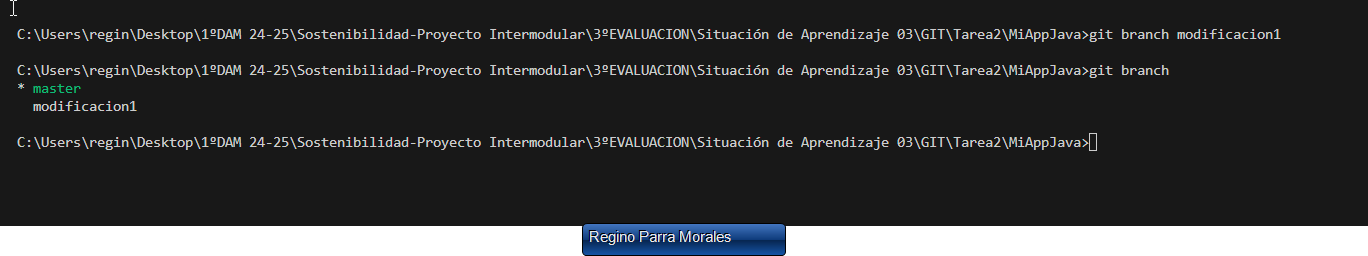
# Crea una nueva rama llamada modificacion1.

Creamos una rama:

git branch modificacion1

Y para ver la lista de ramas:

git branch



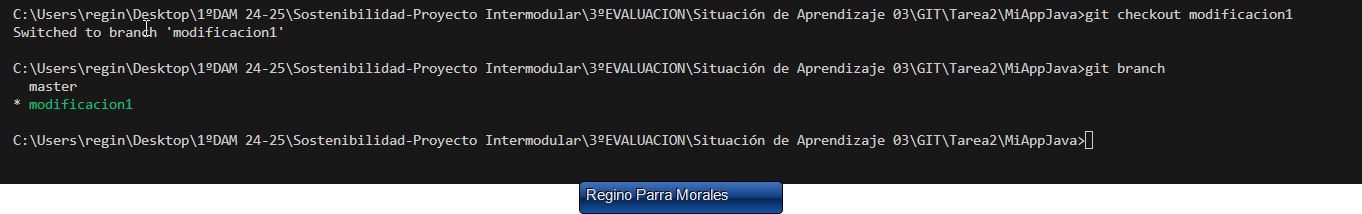
# Cambiate a la nueva rama.

Para cambiarse a la rama nueva creada ,escribimos el siguiente comando:

git checkout modificacion1

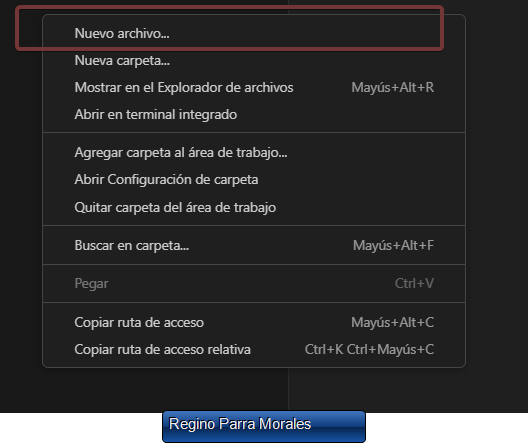
Y volvemos a escribir:

git branch

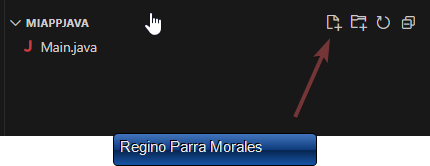


# Crea una nueva clase que aporte una nueva funcionalidad.

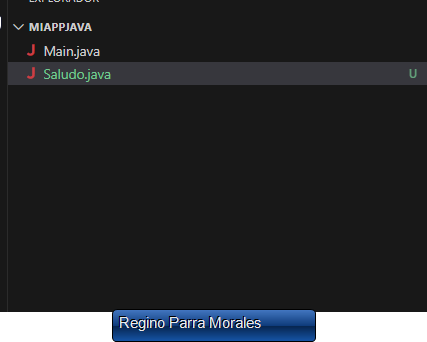
Volvemos a crear un archivo



O también a ese icono



Archivo creado



Escribimos contenido en Saludo.java:

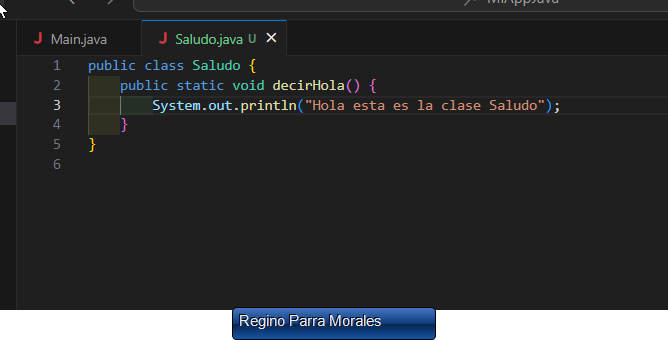
public class Saludo {

public static void decirHola() {

System.out.println("Hola desde la clase Saludo");

}

}



# Modifica Main.java para usar la nueva funcionalidad.

Modificamos con el siguiente código:

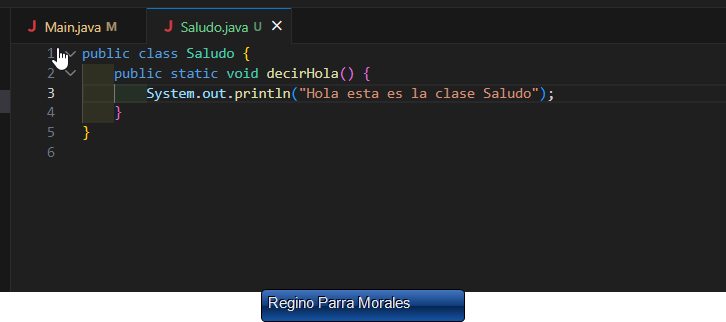
public class Main {

public static void main(String[] args) {

Saludo.decirHola();

}

}

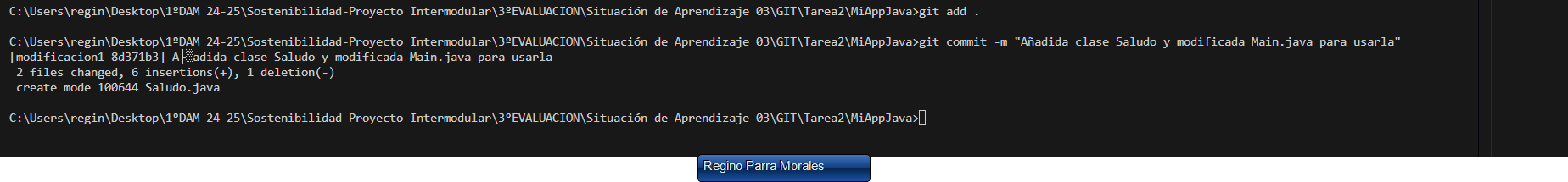


# Agrega y confirma los cambios en la rama.

Agregamos con este código otro commit:

git add .

git commit -m "Añadida clase Saludo y modificada Main.java para usarla"



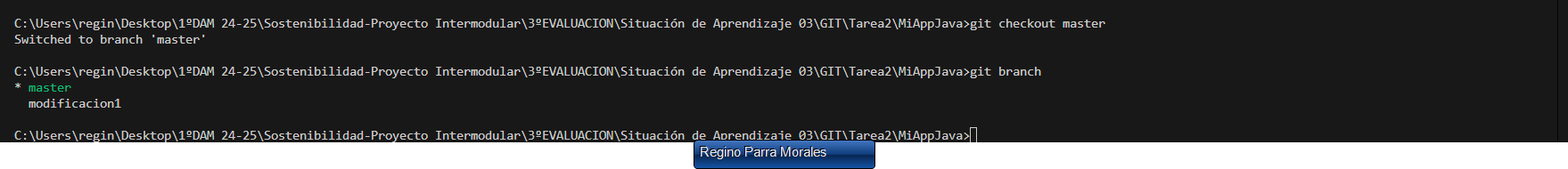
# Cambia de vuelta a la rama principal (master).

Cambiamos de vuelta a la rama principal

git checkout master

Y verificamos que lo hicimos con :

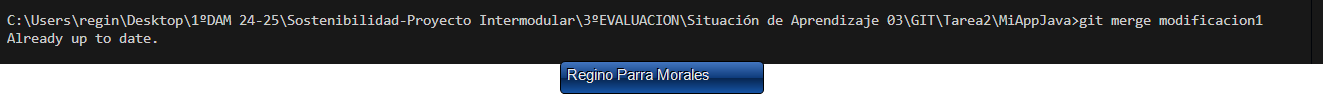
git branch



# Fusiona la rama modificación1 con master.

Fusionamos las dos ramas , la creada y la principal con el siguiente comando:

**git merge modificacion1**



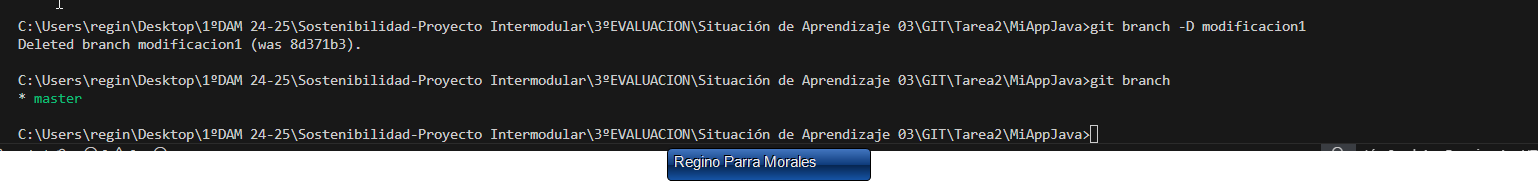
# Elimina la rama modificacion1, ya que sus cambios fueron fusionados.

Escribimos el siguiente comando para eliminar la rama modificacion1:

git branch -D modificacion1

Y verificamos que lo hemos creado con:

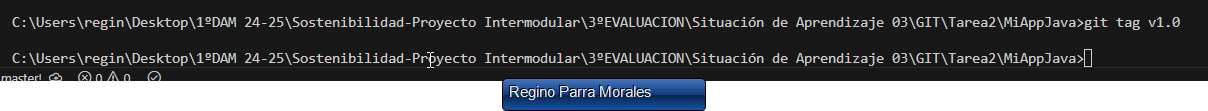
git branch



# Crea una etiqueta para marcar la versión 1.0 de la aplicación.

Para escribir una etiqueta escribimos:

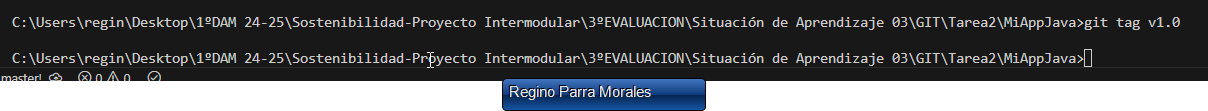
git tag v1.0

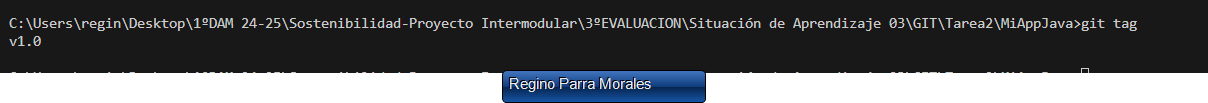


# Verifica que la etiqueta ha sido creada.

Para verificar la etiqueta creada escribimos:

git tag





# Edita la última clase creada para hacer un commit con un error.

Introducimos un error en la clase creada ultima de Saludo.java quitamos el paréntesis de decirHola() quedaría asi:

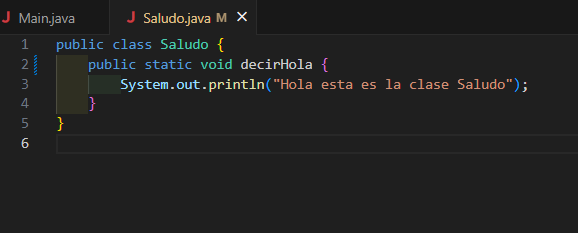
public class Saludo {

    public static void decirHola {

        System.out.println("Hola esta es la clase Saludo");

    }

}

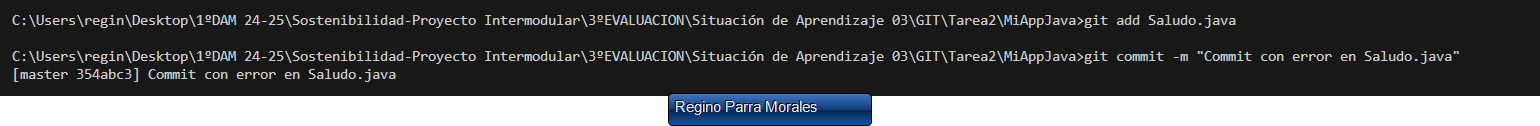


# Agrega y confirma el commit con el error.

Hacemos un commit con el error cometido:

git add Saludo.java

git commit -m "Commit con error en Saludo.java"



# Revierte el último commit sin eliminarlo del historial.

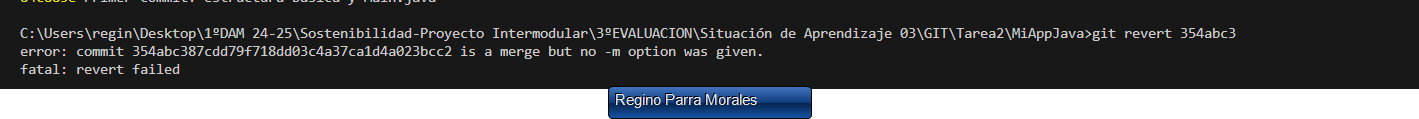
git revert HEAD

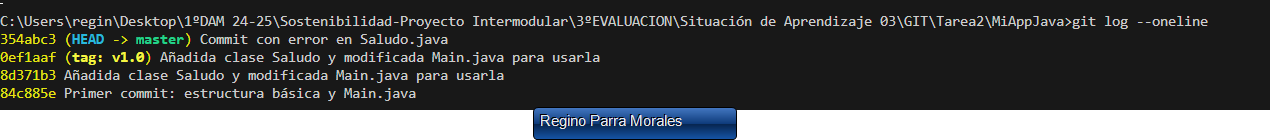
Para revertir al ultimo commit me sale asi con los siguiente comandos de la captura:

git revert 354abc3

Y luego:

git log --oneline





Como se ve me da un pequeño error pero sale la línea de comandos, asi que ya está