María Sarrado Vázquez

**Practica 1.**

En esta práctica vamos a empezar probando una serie de comandos de git que se piden y comentando cuales son los resultados que obtenemos. Más tarde, verificaremos las instalaciones de Java 17, Maven, Docker e IntelliJ

Para empezar, nos hemos creado una cuenta de usuario con el nombre de “mariasarrado” donde vamos a trabajar con todos nuestros proyectos.

Imagen que contiene Gráfico

Descripción generada automáticamente

Hemos creado el repositorio que se ve en la imagen superior, este repositorio contiene nuestro primer proyecto de “hello-world”. El link para este repositorio es el siguiente:

<https://github.com/mariasarrado/hello-worldMARIA>

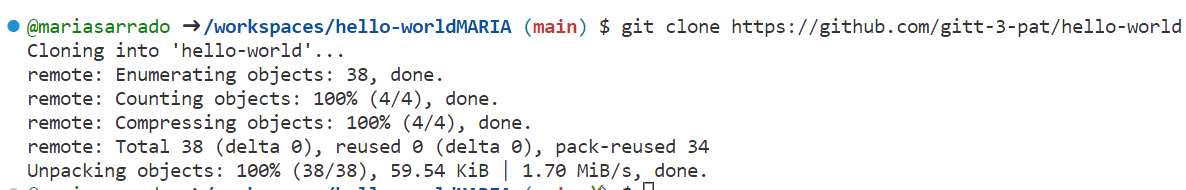
Pinchando en ese link entramos en el repositorio donde vamos a trabajar que tiene este aspecto:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

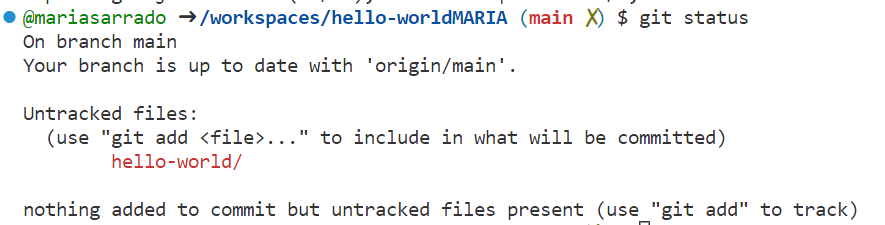
Descripción generada automáticamente

Después de realizar toda la configuración de inicio, probamos todos los comandos que se nos piden.

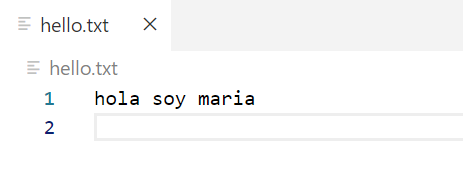
* Clone: como su propio nombre indica, este comando consiste en clonar lo que tenemos en el repositorio, es decir, una copia de él para modificarlo. Para utilizar este comando, simplemente basta con ejecutar “git clone link” y ese link será el link de nuestro repositorio. Todo esto se ejecuta en el terminal:

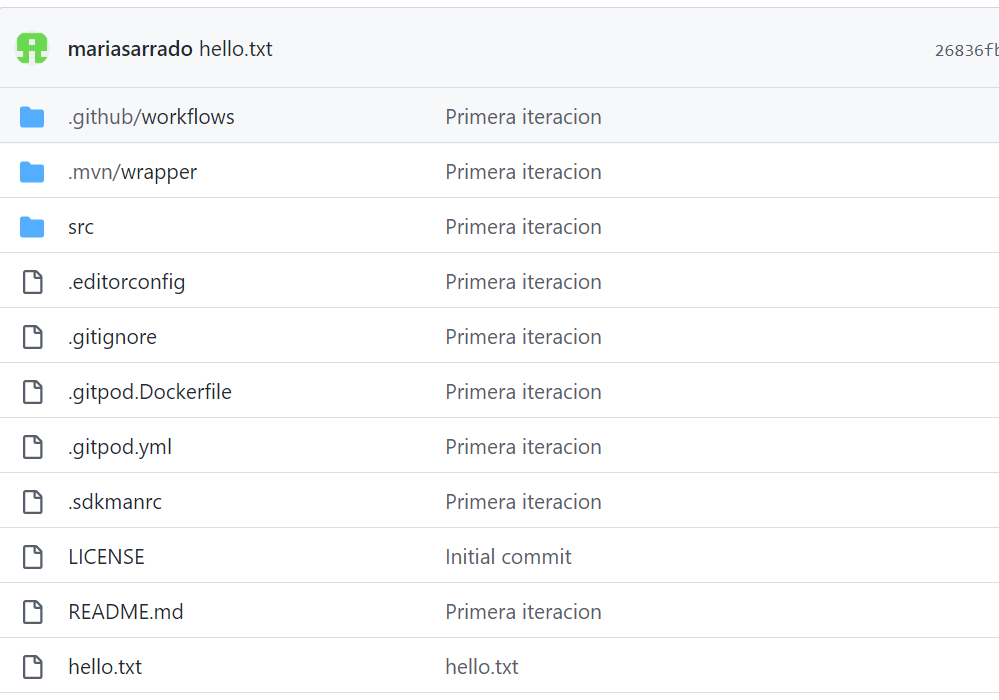


* Status: nos muestra el estado del directorio de trabajo. También nos dice lo que hace falta para actualiza. Este comando también se ejecuta desde el terminal usando el código “git status”. En la imagen de abajo se puede ver como nuestro directorio de trabajo está al dia de los cambios:

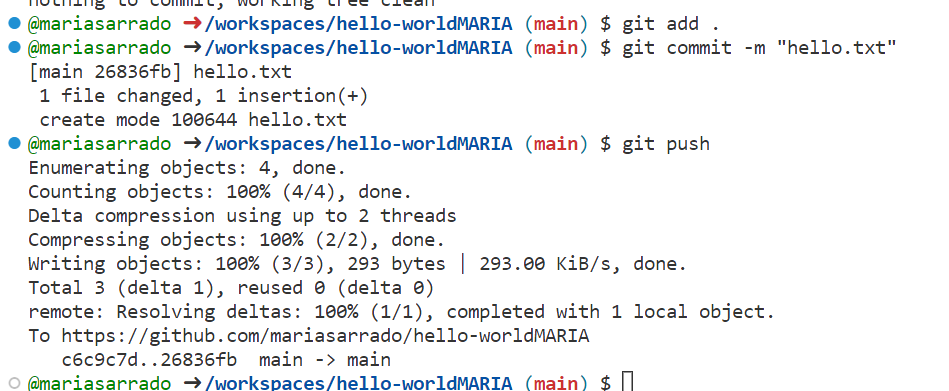


Para los siguientes comandos nos hemos creado un archivo que lo hemos llamado “hello.txt” y a partir de el vamos a implementar los comando que nos faltan. Este archivo, como se muestra en la segunda imagen se ha guardado en nuestro repositorio de hello\_world:

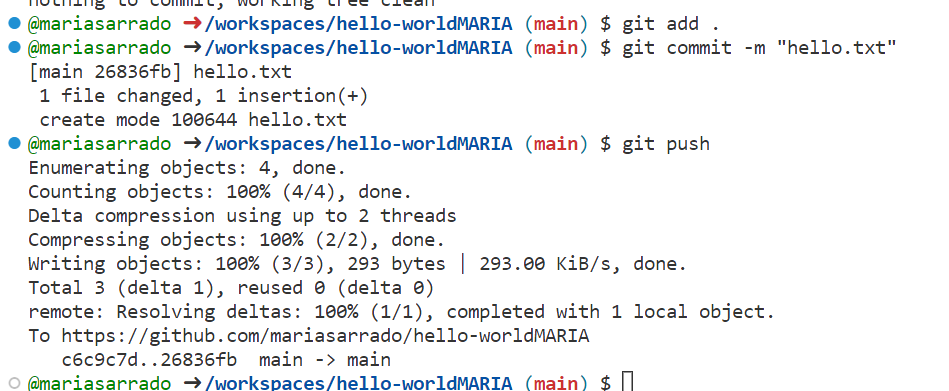




* Add: seleccionar los archivos que hemos modificado y cuyos cambios queremos que se guarden en la nube. En el caso que queramos añadir todos los archivos que se han visto modificados con distintos cambios, añadimos al final del comando git add un punto, es decir, quedaría escrito de la siguiente forma: “git add .”



* Commit: coge todos los archivos que han sido modificados y lo que hace es guardarlos en el repositorio local. Suele ir acompañado de un mensaje que se escribe después de un “-m”



* Push: este comando coge los commits y cambios modificados desde nuestro repositorio local al repositorio en remoto.

Texto

Descripción generada automáticamente

* CheckOut: este comando sirve para cambiar y navegar entre ramas de los árboles de trabajo. Si al código le añadimos un “-b” se crea una rama con el nombre que queramos indicar.

La segunda parte de la práctica consiste en comprobar la instalación de distintos entornos de trabajo. Primero comprobamos la versión que tenemos instalada en Java:

Texto

Descripción generada automáticamente

Luego nos hemos descargado MAVEN y hemos comprobado la version que esta instalada en el ordenador. Para ello, tanto en java como en maven, hemos tenido que crear variables de entorno para que nos funcionase el comando en la terminal. Hemos creado MAVEN\_HOME y JAVA\_HOME. En la imagen de debajo se muestra la version de Maven que nos hemos instalado:

Texto

Descripción generada automáticamente

Por último, tenemos que tener instalado un editor de código fuente. En nuestro caso tenemos IntelliJ:

