UNIDAD 3

GIT

Cristian López Kostiouk 1ºDAM

Contenido

[Primeros comandos 3](#_Toc128173109)

[Configuración 3](#_Toc128173110)

[Establecer nombre de usuario 3](#_Toc128173111)

[Configurar la dirección de correo 3](#_Toc128173112)

[Crear el primer repositorio 4](#_Toc128173113)

[Cambiar el nombre de la rama 6](#_Toc128173114)

[Configurar el valor por defecto de la rama 6](#_Toc128173115)

[Readme.md y log 7](#_Toc128173116)

[Commits realizados 7](#_Toc128173117)

[Adds y commits desde VSC 8](#_Toc128173118)

[Archivo .gitkeep 8](#_Toc128173119)

[Comparar cambios en los archivos 10](#_Toc128173120)

[Actualizar mensajes de commits y deshacer 10](#_Toc128173121)

[Actualizar el mensaje del último commit 10](#_Toc128173122)

[Eliminar un commit 10](#_Toc128173123)

[Ignorar archivos que no deseamos 11](#_Toc128173124)

[GitHub 12](#_Toc128173125)

[Push a GitHub 12](#_Toc128173126)

# Primeros comandos

Para visualizar la versión del Git se usa:

Git --version

Si queremos mostrar la ayuda sobre los comandos:

Git help

## Configuración

### Establecer nombre de usuario

Git config –global user.name “CristianLK”

### Configurar la dirección de correo

Git config –global user.email cristianlk52@edcastur.es

# Crear el primer repositorio

Un repositorio de Git es un tipo de almacenamiento virtual de algún proyecto. Permite guardar versiones del código a las que se puede acceder cuando se necesite.

En primer lugar, se crea una carpeta para guardar todo el repositorio. En mi caso, es un proyecto de prueba.

Para inicializarlo usamos el comando:

Git init

Texto

Descripción generada automáticamente

Dentro, de la carpeta se creará un .git (una carpeta oculta).

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Para mostrar el estado de nuestro repositorio:

Git status

Texto

Descripción generada automáticamente

Si queremos añadir todo el contenido de la carpeta actual, usamos:

Git add .

Texto

Descripción generada automáticamente

Si queremos guardar el estado actual de los archivos, usamos el comando:

Git commit -m “Mi primer commit”

Texto

Descripción generada automáticamente

En caso que queramos reconstruir el proyecto, usamos:

Git checkout -- .

# Cambiar el nombre de la rama

Si queremos saber en que rama estamos actualmente trabajando, usamos:

Git branch

Para cambiar el nombre de la rama de desarrollo:

Git branch -m master main

## Configurar el valor por defecto de la rama

Git config –global init.defaultBranch main

# Readme.md y log

El README es un archivo que contiene información acerca de otros archivos en un directorio. Es una forma de documentación de software.

Creamos un archivo llamado readme.md y escribimos algo de contenido:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Hacemos seguimiento del archivo:

Texto

Descripción generada automáticamente

## Commits realizados

Para visualizar los commits usamos el comando:

Git log

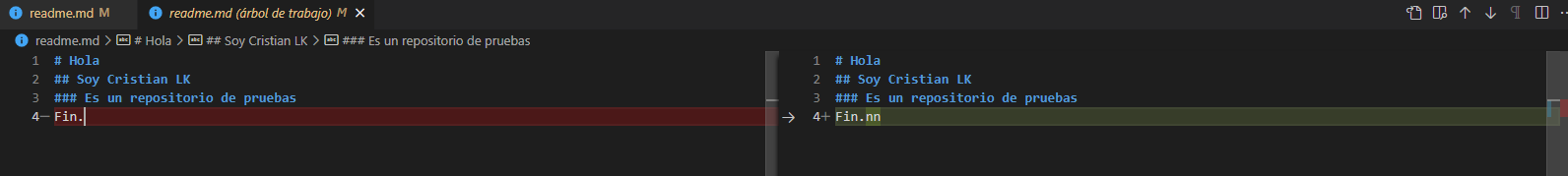
Texto

Descripción generada automáticamente

# Adds y commits desde VSC

También se puede realizar estas funciones, desde el apartado gráfico.

En este caso edité el mensaje de readme, y se muestra de esta manera:



Y si queremos revertir los cambios, usamos la flecha de deshacer cambios.



# Archivo .gitkeep

Cuando queramos añadir al menos un archivo a una carpeta vacía para que git la tenga en cuenta, en vez de crear cualquier archivo existe un archivo con un nombre especial que podemos crear: el archivo .gitkeep

Para añadir este archivo se usa:

Git add gitkeep/\*.gitkeep

Ahora usamos el commit.

Texto

Descripción generada automáticamente

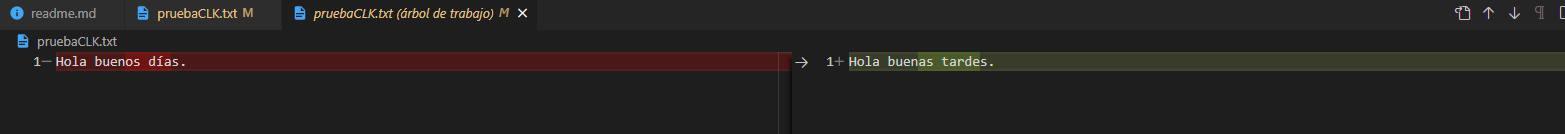
Se verá de esta manera:

Texto

Descripción generada automáticamente

# Comparar cambios en los archivos

Si queremos mostrar ´las modificaciones realizadas, hay una manera visual:



# Actualizar mensajes de commits y deshacer

## Actualizar el mensaje del último commit

Para cambiar el mensaje del último commit realizado usamos:

Texto

Descripción generada automáticamente con confianza baja

## Eliminar un commit

Esto funciona si queremos eliminar el último commit realizado usamos:

Texto

Descripción generada automáticamente

# Ignorar archivos que no deseamos

Para esta prueba usaremos los siguientes archivos y carpetas:

Pantalla de computadora con letras

Descripción generada automáticamente con confianza media

Después creamos el archivo .gitignore, y escribimos el nombre de los archivos y directorios que queremos que git ignore.

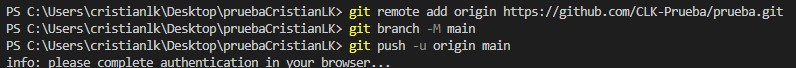
Texto

Descripción generada automáticamente

# GitHub

## Push a GitHub

Usamos los comandos para añadir el repositorio local a nuestro repositorio remoto:



Aquí no se muestra la captura de los archivos mostrados hasta ahora, debido a que fue un cambio de directorio.

Se vería de esta manera en Github:

Captura de pantalla con la imagen de una pantalla

Descripción generada automáticamente con confianza media

Si queremos traer los cambios de nuestro Github a local, ejecutamos:

Git pull

Y si queremos hace rla reversa, de local a remoto usamos:

Git push