Amilcar Alexander Medina Méndez

Homework-1

Que es GIT

Git es un sistema de control de revisiones de código abierto que se mantiene activo y lo utilizan miles de desarrolladores en todo el mundo.

Sistema de control de versione, administrar mis cambios.

Control de versiones: antes, ahora, futuro

Multiusuario

Control de cambios por usuario

Regresar a versiones anteriores cuando quiera

Repositorios Locales y remotos

Yo decido cuando cambio

Control de versiones con GIT

Git es un Sistema de Control de Versiones Distribuido (DVCS) utilizado para guardar diferentes versiones de un archivo (o conjunto de archivos) para que cualquier versión sea recuperable cuando lo desee.

Git también facilita el registro y comparación de diferentes versiones de un archivo. Esto significa que los detalles sobre qué cambió, quién cambió qué, o quién ha iniciado una propuesta, se pueden revisar en cualquier momento.

Estados de un archivo en git

Git tiene tres estados principales en los que se pueden encontrar tus archivos: confirmado (committed), modificado (modified), y preparado (staged).

Estado modificado

Un archivo en el estado modificado es un archivo revisado – pero no acometido (sin registrar).

En otras palabras, archivos en el estado modificado son archivos que has modificado, pero no le has instruido explícitamente a Git que controle.

Estado preparado

Archivos en la etapa preparado son archivos modificados que han sido seleccionados – en su estado (versión) actual – y están siendo preparados para ser guardados (acometidos) al repositorio .git durante la próxima instantánea de confirmación.

Una vez que el archivo está preparado implica que has explícitamente autorizado a Git que controle la versión de ese archivo.

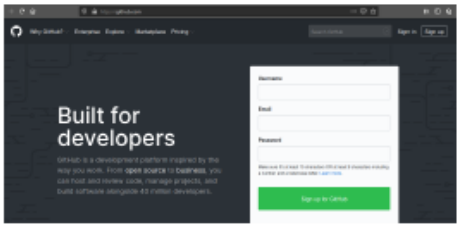
Estado confirmado

Archivos en el estado confirmado son archivos que se guardaron en el repositorio .git exitosamente.

Por lo tanto, un archivo confirmado es un archivo en el cual has registrado su versión preparada en el directorio (carpeta) Git.

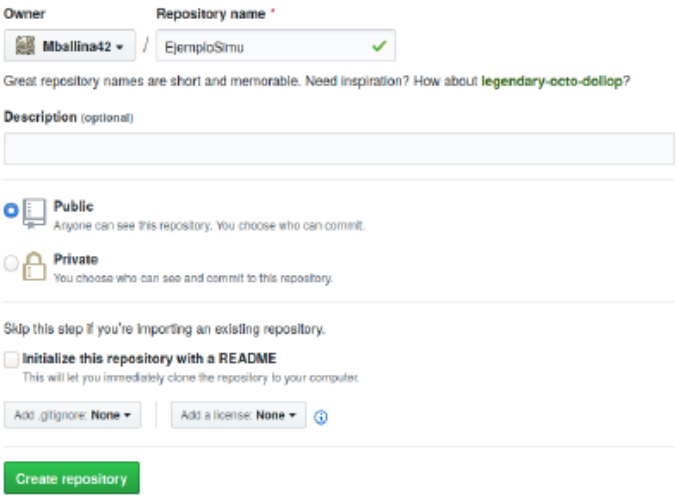
Como se configura un repositorio

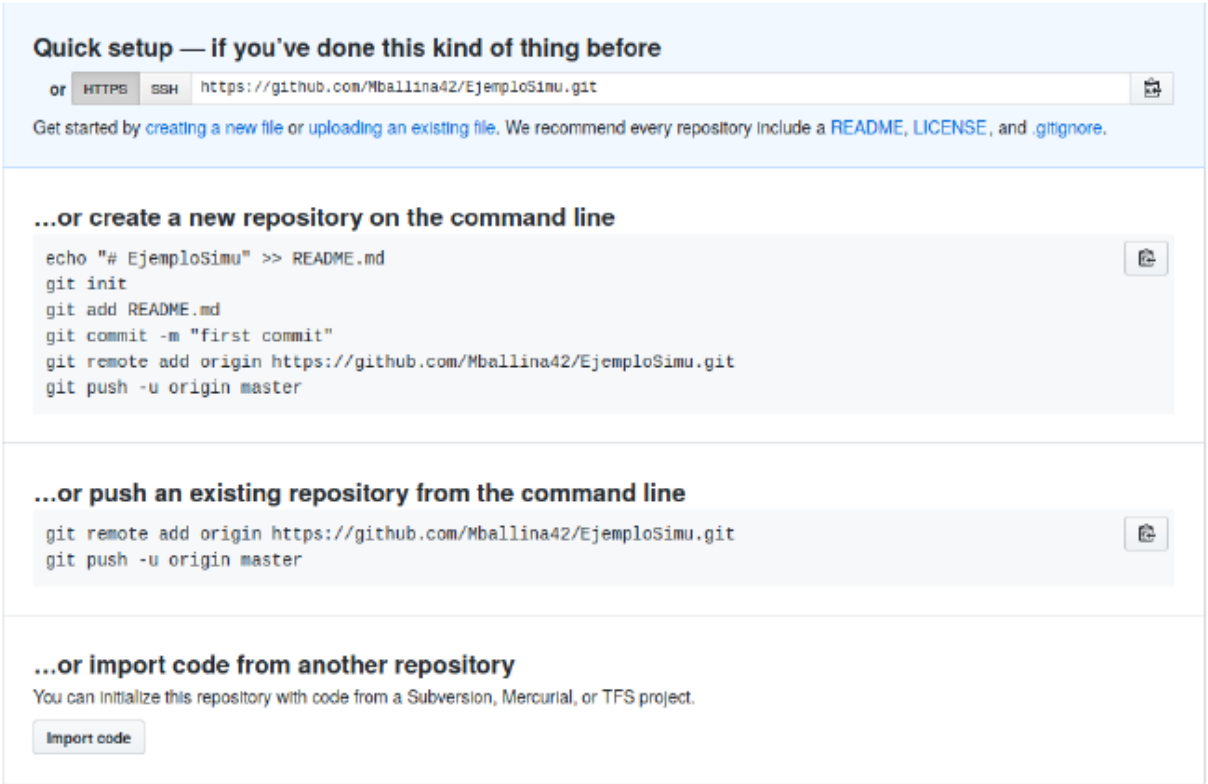
Creamos una cuenta.



Creamos un nuevo repositorio.

Configuramos repositorio.





Comandos en GIT

- $git log - vemos los elementos de nuestro commit junto a su hash
- git checkout master - cambiamos a la rama master

- git init - iniciamos git, carpeta se convierte en Working directory
- git status - revisamos el contenido
- git add file1 - agregamos file1 a staging area
- git status - muestra estados de los archivos
- git add file2 - agregamos file2 a staging area
- git status - muestra estados de los archivos
- git commit - abre su editor de texto, siempre agregar un comentario. Se mueve todo al repositorio

- git status - muestra que no hay nada a que hacer commit
- echo nueva linea >> file1 - escribimos en file1
- git status - muestra que hay un archivo en el WD
- git checkout −− file1 - Deshace la modificación, remplazando con lo del ultimo commit
- git status - muestra que no hay modificaciones
- echo 42 u infidel >> file1 - modificamos file1
- cat file1 - muestra diferencia entre file1 del WD y el Rep
- git diff file1 - agregamos file1 a staging area
- git add file1 - abre su editor de texto, agregar un comentario. Se mueve todo al repositorio
- git commit

• mkdir nover	- creamos una carpeta
• touch nover.txt	- creamos un archivo
• touch .gitignore	- necesitamos un archivo de nombre .gitignore
• echo nover >> .gitignore	- escribimos el nombre de la carpeta y del archivo dentro del archivo .gitignore
• echo nover.c >> .gitignore	
• git status	- muestra que solo existe .gitignore
• touch nover/otro.txt	- creo un archivo dentro de la carpeta ignorada
• git status	- muestra que solo existe .gitignore
• git add .gitignore	- agregamos .gitignore a staging area
• git commit -m "comentario"	- se mueve todo al repositorio con la instrucción (-m) podemos agregar un comentario en linea

• git branch	- muestra que solo existe la rama master
• git branch OTRAR	- creamos otra rama en nuestro proyecto
• git branch	- muestra las ramas existentes
• git checkout OTRAR	- cambiamos a la rama OTRAR
• git branch	- muestra que estamos en rama OTRAR
• touch f1 f2 f3 f4 f5	- creamos varios archivos
• git add .	- agregar los archivos del proyecto usamos (.)
• git status	- muestra los files en staging area
• git commit -m "comentar"	- se mueve todo al repositorio.
• git log	- vemos los elementos de nuestros commits junto a cada hash