Aridany de la Fe Sarmiento

Git y GitHub Terminal

Instalar git
Git Instalación

Iniciar git. (Con Windows, con Linux directamente desde Visual Studio Code)
Click derecho en la carpeta abierta donde vamos a guardar documentos del proyecto.
Abrir Git Bash (Consola de Git)

Usar comandos:
git init
Iniciar Git

git status -s
Listado de archivos y directorios que tenemos en la carpeta del proyecto

git config --global user.name "Aridany"
Decirle a git de forma global tu nombre. Si no lo hacemos no nos dejará hacer el siguiente paso

git config --get user.name

Comprobar nombre de usuario

git config --global user.email aridany54@gmail.com Decirle a git de forma global tu nombre.

git config --get user.email

Comprobar email del usuario

git add index.html
Realiza seguimiento de un archivo (stage area o área de ensayo). Es el lugar en el que se encuentran los datos de un proyecto y sus cambios

git status -s
Para comprobar que el se le está haciendo el seguimiento al archivo

git commit -m "Comienzo"
Para realizar la instantánea. Desaparece del listado y se agrega al repositorio.
Si realizamos cambios y hacemos un **git status** veremos que ahora sí que sale en el listado y aparece en rojo y una M porque hay modificaciones.

git add index.html
De nuevo

git commit -m "Cambio realizado x"
De nuevo

git status -s
Para comprobación

Información extraída del canal de Youtube: Píldoras informáticas
Info comandos: Git
Lista con los Videos del curso Git & GitHub

git log --oneline
Listado de todas las copias con el código y la descripción (Ver las instantáneas realizadas)

git reset --hard 2bd6c5e
Restaura el archivo y quita cambios previos.
Ir hacia atrás hasta llegar a la versión 2bd6c5e
Quita el resto de versiones y deja **SOLO** esta. (Porq volvemos a la instantánea anterior)
git add .
(Después del add hay un espacio y un punto) Agregamos al área stage todos los archivos del proyecto. Seguimiento

git status -s
Comprobamos que está dentro de área stage los archivos del proyecto

git log --oneline

Ahora realizamos cambios en el proyecto

git status -s
Vemos que se han producido cambios

git commit -am "Cambios al H1 y color al mismo"
2 Operaciones a la vez (add y commit). Agregamos los cambios realizados al área stage y realizamos la instantánea de seguridad junto con el nombre que le hemos dado.
/* He añadido otros cambios que no son los del video, pero bien. */

git commit --amend
Abrir editor VIM (Editor dentro de la consola de git) para modificar algunas cosas, en este caso, el nombre del commit.
/* A mi se me abre en el Visual Studio Code por configuración al instalarlo, pero pude cambiar correctamente el nombre sin utilizar terminal. */

Para hacerlo en terminal:

:i
Para poder empezar a editar
Suprimir para borrar nombre
Apretar escape para salir de edición

:i
Para Poner el nombre

:wq
Para guardar y salir.

Información extraída del canal de Youtube: Píldoras informáticas
Info comandos: Git
Lista con los Videos del curso Git & GitHub

Aridany de la Fe Sarmiento

Subir proyecto a GitHub

Creamos la cuenta en github

Create new repository (Dentro de GitHub)

Elegir si crear un nuevo repositorio o subir un repositorio existente (En nuestro caso, subir un repositorio existente porque ya lo hemos creado nosotros a nivel local)

git remote add origin https://github.com/AridanyS/Primer-Proyecto-Git.git
Indica el repositorio al cual vas a subir los documentos (Solo por primera vez ya que es el primero)

git push origin master
Pedirá usuario y contraseña de GitHub si es la primera vez
Sube las modificaciones a GitHub **(Local a GitHub)**

git pull https://github.com/AridanyS/Primer-Proyecto-Git.git
Modificaciones del repositorio de **(GitHub a Local)**
(Link del repositorio de GitHub)

Añadir tags

// Tags: Especificar versiones de nuestro proyecto (Por ejemplo versión 1.0, 2.0, 3.0, ...)

git tag 22/06/2023v1 -m "Versión 1 del proyecto"
Esto es solo para decirle a Git que tenemos la primera versión del proyecto (Local)

git log --oneline
Comprobamos los cambios

git push --tags
Subir el tag a GitHub

Clonar proyecto desde el repositorio remoto (GitHub)

Entrar al repositorio

<> Code

Clone

Clonar with HTTPS, descargar el zip, SSH, ...
Depende lo que necesites, en este caso lo clonamos usando https

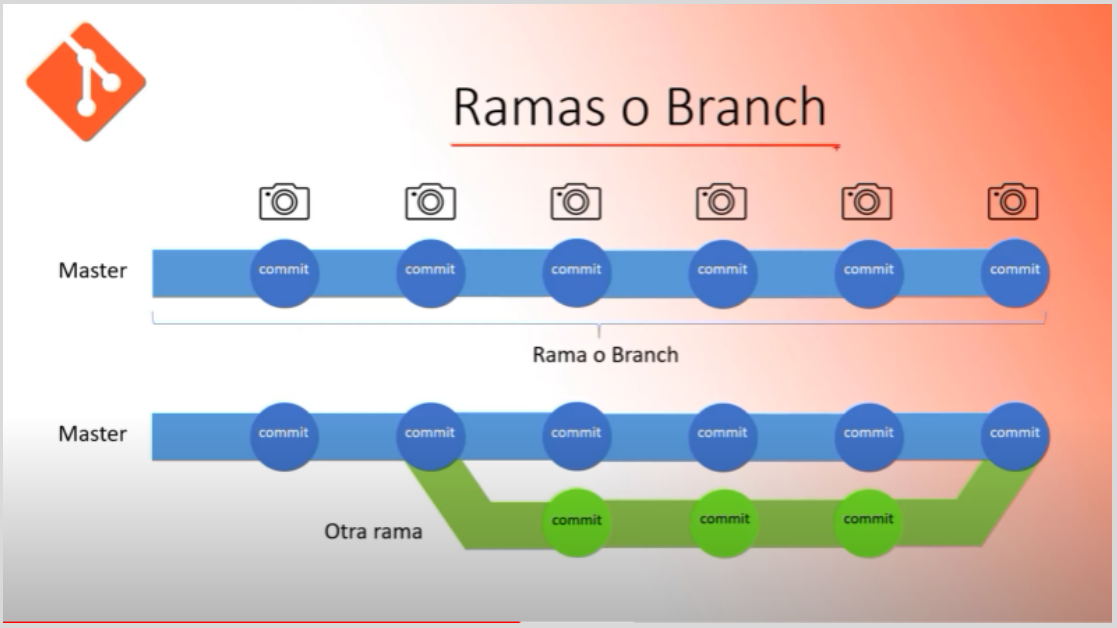
Copiar url del portapapeles

Abrir GitBash desde donde queremos clonar el proyecto

git clone https://github.com/AridanyS/Primer-Proyecto-Git.git
¡Proyecto clonado! 🙂

Información extraída del canal de Youtube: Píldoras informáticas
Info comandos: Git
Lista con los Videos del curso Git & GitHub

Ramas o Branch



// **Ramas o Branch:** Conjunto de commits que va desde el comienzo del proyecto hasta el final. Se suelen usar también los branch secundarios o paralelos que son una copia del primero y se usan para realizar pruebas o trabajar en él sin realizar cambios en el branch original.
// **Master:** Nombre por defecto del branch principal de Git

git branch javascript
Creamos el branch (javascript es el nombre que le quieres dar a la rama)

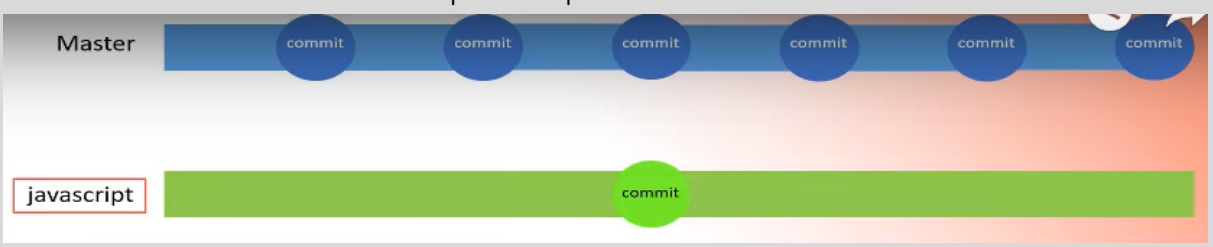
git log --oneline
Para comprobar si se ha creado correctamente

git branch
Para ver los branch del proyecto. También en cual estamos, nos lo indica con un asterisco (*) y en color verde

```
arida@DESKTOP-RQ44179 MINGW64 ~/Documents/Apuntes/Git/Proyectos/Airbnb (master)
$ git branch
  javascript
* master
```

El branch javascript está vacío y no está en uso. Si realizamos commits o modificaciones se añadirían al branch master.

git checkout javascript
Movernos al branch secundario para empezar a realizar cambios en el mismo.



Información extraída del canal de Youtube: Píldoras informáticas
Info comandos: Git
Lista con los Videos del curso Git & GitHub

git branch
Para comprobar cambios en los branchs.

Realizamos cambios en el archivo de javascript

git add .
Para llevarlo al area stage

git status -s
Para comprobar cambios

git commit -m "Añadimos modificación al archivo de javascript"
Añadimos instantánea al branch nuevo

git log --online
Comprobamos cambios realizados

git checkout master
Movernos del branch paralelo (javascript) al branch principal (Master).
Ahora, los cambios que realizamos en el branch secundario no los veremos reflejados en el fichero principal, eso es porque se encuentran en el branch secundario y no en el principal que es donde nos hemos movido.
¡De esta forma estamos trabajando con 2 copias del proyecto en paralelo!

git merge javascript
Fusiona el branch master con lo hecho en el branch javascript.
Es **obligatorio** moverse al branch principal (Master) (En este caso ya estábamos ahí).
/* Si hacemos cambios a un mismo archivo en 2 ramas distintas ocurrirán errores o conflictos..
Esto es normal porque hay cambios en el mismo archivo (mismo archivo porque se ha fusionado) y hay que recordar que lo que realmente queremos es fusionar una rama que contiene cambios en una que no los tiene. */

git push origin master
Para añadir los cambios a GitHub.

git branch -d Second-Branch
Eliminamos el branch (la -d es de delete) si no la queremos usar más.

git log --oneline
git branch
Para comprobar cambios
Cambiar nombre de la rama actual
git branch -m master main
"master" es el nombre de la rama actual, "main" el nombre que le quieres adjudicar.
Esto lo hacemos porque en GitHub la rama cuando le añadimos el README.md cambia a main

Información extraída del canal de Youtube: Píldoras informáticas
Info comandos: Git
Lista con los Videos del curso Git & GitHub