

INTRO A TERMINAL + INTRO A GIT & GITHUB

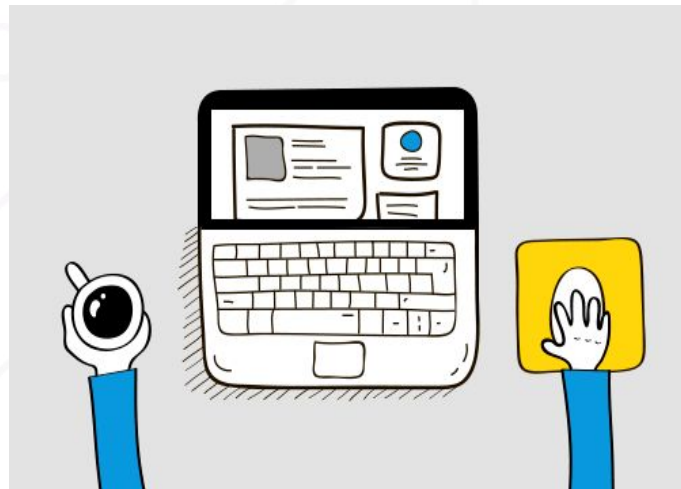
DEV.F.
DESARROLLAMOS(PERSONAS);

dev

Objetivos de la sesión

- ¿Qué es una Terminal? 🤔.
- Instalación de Git Bash para Windows 🧑.
- Uso de comandos básicos
- Instalación de Git 💻.
- Crear cuenta en Github 🧑.
- Comandos básicos de Git 😊.
- Clonar repositorios 🧐.
- Subir tus archivos y actualizar tus repositorios 😊.
- Reto subir tu propio repositorio a github 🐱.

`< eat, sleep, code, repeat />`



DEV.F

¿Que es una terminal?





Git Bash (Windows)



Terminal (MacOS)



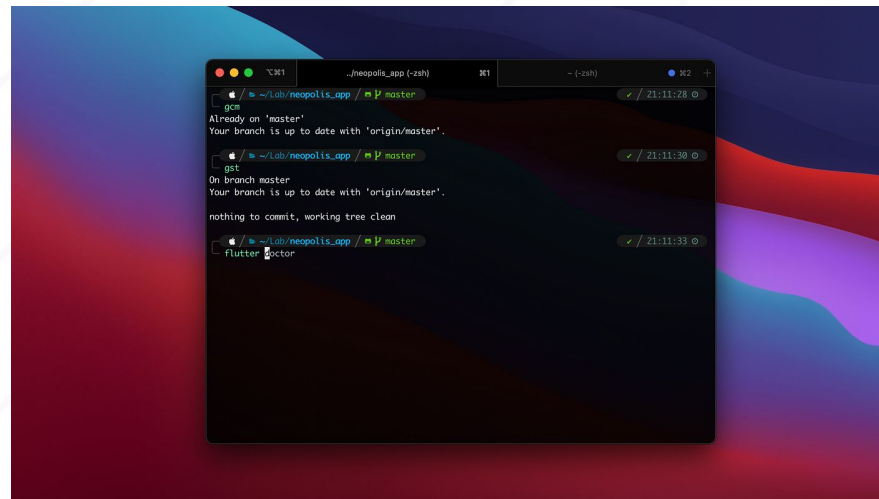
Terminal (Ubuntu Linux)

¿Que es la terminal?

Es una herramienta que te permite interactuar con una computadora utilizando comandos de texto en lugar de una interfaz gráfica.

Hablemos de la terminal

- Imagina que la terminal es como hablar con tu computadora en un lenguaje especial.
- En lugar de hacer clic en iconos o menús con el ratón, tú escribes palabras y comandos para decirle a la computadora qué hacer.
- Puedes usar la terminal para crear, copiar, mover o eliminar archivos y carpetas, instalar programas, compilar código fuente y realizar muchas otras tareas relacionadas con la programación y la administración de sistemas.



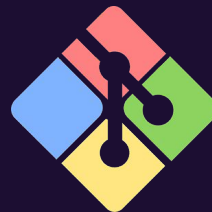
¿Porque es importante la terminal?



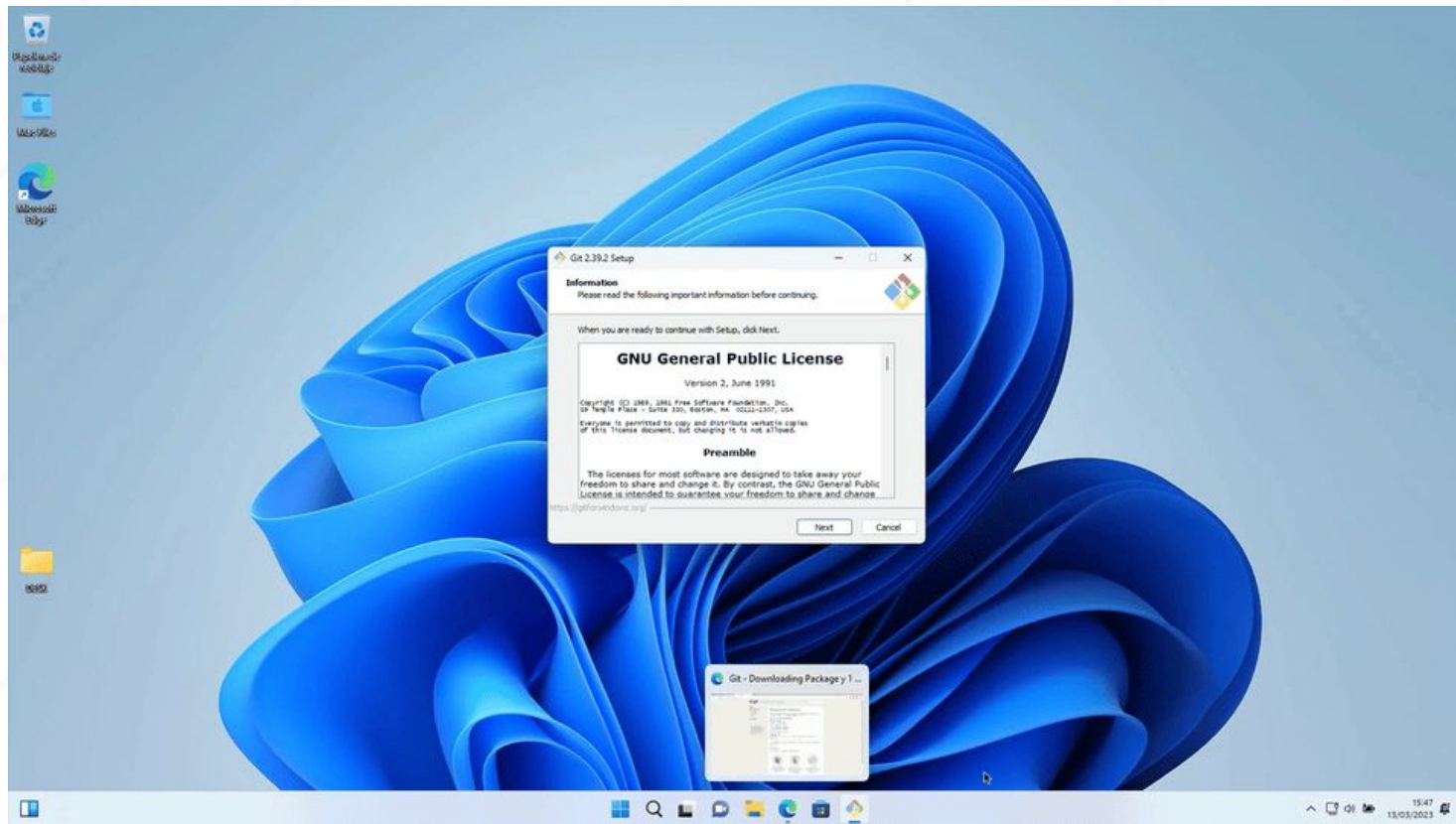
- Ejemplo 1: Aunque hay algunas interfaces gráficas para git, suele ser más cómodo y productivo hacerlo desde terminal.
- Ejemplo 2: Cuando contratas un servicio en la nube, **NO** te proporcionan una interfaz gráfica.

DEV.F

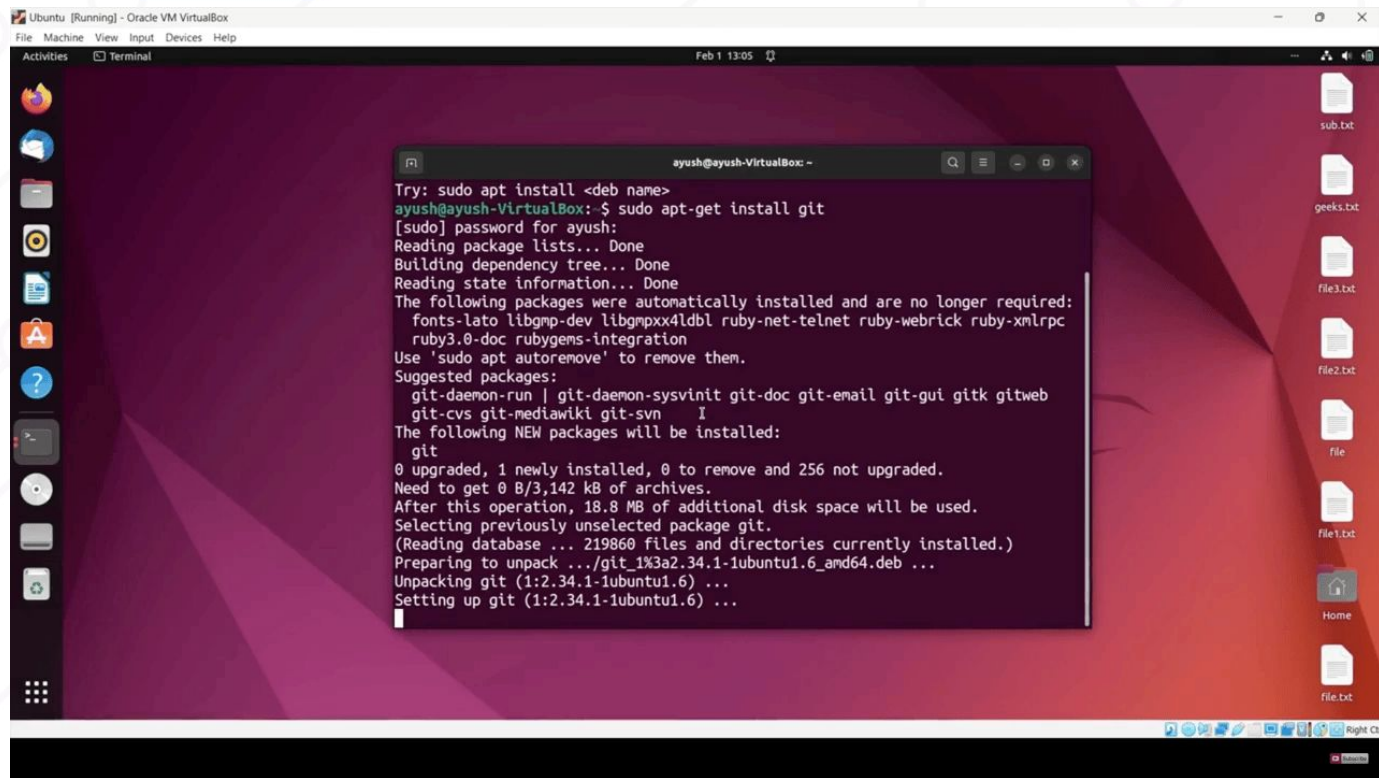
Instalación de git



Instalación de Git Bash para Windows



Pasos para instalar git en Linux



The screenshot shows a terminal window titled 'ayush@ayush-VirtualBox: ~' running the command 'sudo apt install git'. The output shows the package lists, dependency tree, and state information. It lists several packages that are no longer required and suggests other packages. The final output shows that git is being installed, and the disk space requirements are met.

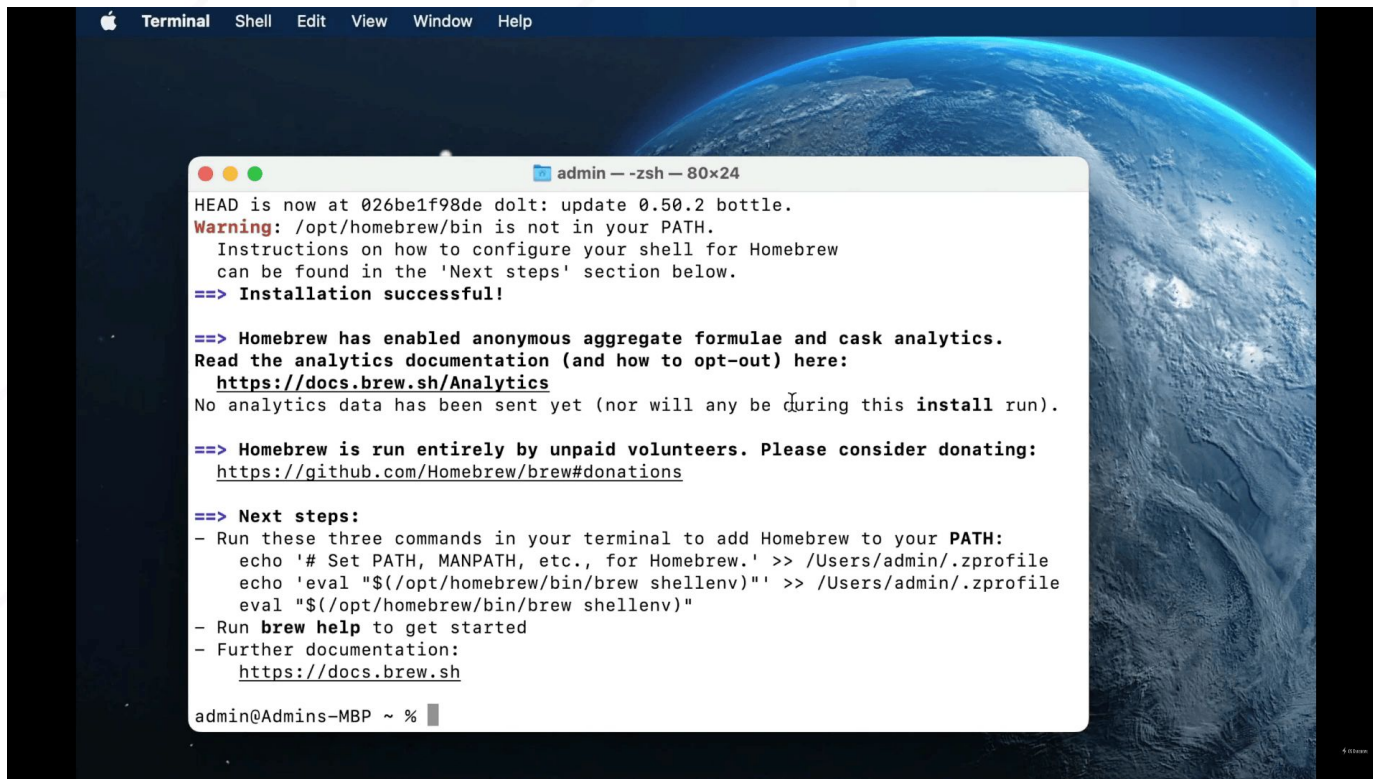
```
Try: sudo apt install <deb name>
ayush@ayush-VirtualBox:~$ sudo apt-get install git
[sudo] password for ayush:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  fonts-lato libgmp-dev libgmpxx4ldbl ruby-net-telnet ruby-webrick ruby-xmlrpc
  ruby3.0-doc rubygems-integration
Use 'sudo apt autoremove' to remove them.
Suggested packages:
  git-daemon-run | git-daemon-sysvinit git-doc git-email git-gui gitk gitweb
  git-cvs git-mediawiki git-svn
The following NEW packages will be installed:
  git
0 upgraded, 1 newly installed, 0 to remove and 256 not upgraded.
Need to get 0 B/3,142 kB of archives.
After this operation, 18.8 MB of additional disk space will be used.
Selecting previously unselected package git.
(Reading database ... 219860 files and directories currently installed.)
Preparing to unpack .../git_1%3a2.34.1-1ubuntu1.6_amd64.deb ...
Unpacking git (1:2.34.1-1ubuntu1.6) ...
Setting up git (1:2.34.1-1ubuntu1.6) ...
```

Pasos para instalar git en Mac OS

```
1 # Install homebrew https://brew.sh/index_es
2 $ /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
3
4 # Install git with homebrew
5 $ brew install git
```

Para el caso de Mac OS, se recomienda hacer uso de la herramienta
Homebrew.

Pasos para instalar git en Mac OS



```
admin -- zsh -- 80x24
HEAD is now at 026be1f98de dolt: update 0.50.2 bottle.
Warning: /opt/homebrew/bin is not in your PATH.
Instructions on how to configure your shell for Homebrew
can be found in the 'Next steps' section below.
==> Installation successful!

==> Homebrew has enabled anonymous aggregate formulae and cask analytics.
Read the analytics documentation (and how to opt-out) here:
https://docs.brew.sh/Analytics
No analytics data has been sent yet (nor will any be during this install run).

==> Homebrew is run entirely by unpaid volunteers. Please consider donating:
https://github.com/Homebrew/brew#donations

==> Next steps:
- Run these three commands in your terminal to add Homebrew to your PATH:
  echo '# Set PATH, MANPATH, etc., for Homebrew.' >> /Users/admin/.zprofile
  echo 'eval "$(/opt/homebrew/bin/brew shellenv)"' >> /Users/admin/.zprofile
  eval "$(/opt/homebrew/bin/brew shellenv)"
- Run brew help to get started
- Further documentation:
  https://docs.brew.sh

admin@Admins-MBP ~ %
```

DEV.F

Comandos básicos de terminal



Comandos básicos de terminal

ls -> Muestra el contenido de un directorio (archivos y carpetas).

cd **directorio** -> Cambia el directorio actual, con “..” volvemos una carpeta atrás.

pwd -> Muestra la ruta completa del directorio actual.

mkdir **directorio** -> Crea un nuevo directorio.

touch **nombre_archivo** -> Crea un nuevo archivo vacío.

cp **directorio** -> Copia archivos o directorios (usar la bandera -R).

mv **dir_actual** **dir_nuevo** -> Mueve o cambia el nombre a archivos o directorios.

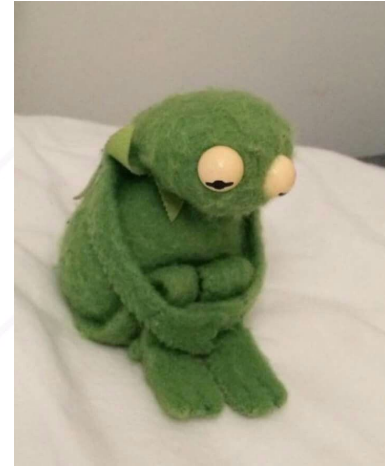
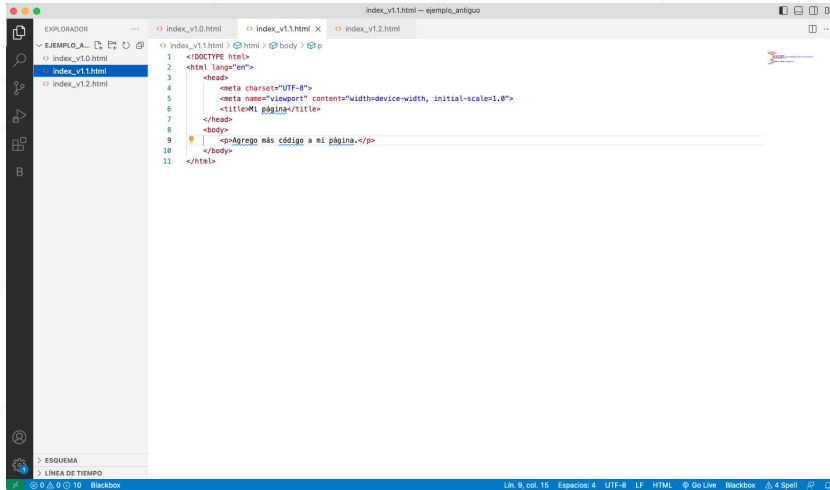
rm **directorio** -> Elimina archivos o directorios (usar la bandera -R).

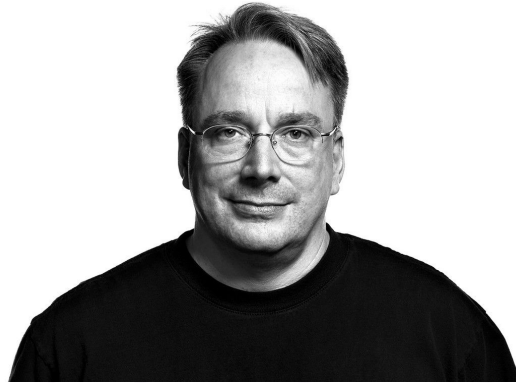
DEV.F

¿Que es git?



Imaginemos una vida sin git

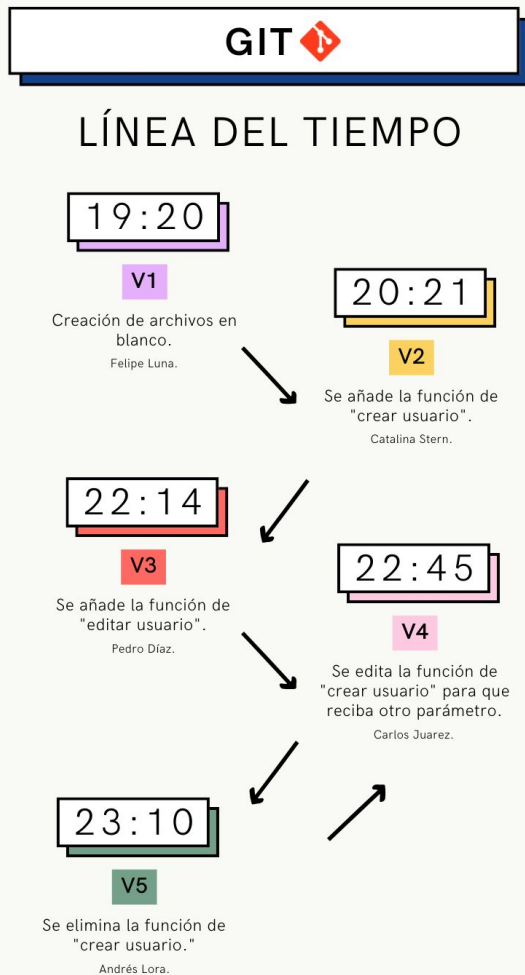




GIT es un sistema de control de versiones diseñado por Linus Torvalds,
pensando en la eficiencia y confiabilidad del mantenimiento de versiones.

Control de versiones (GIT)

Un sistema de control de versiones nos permite ver, guardar y organizar cambios en el código. En palabras sencillas **GIT** permite ver las partes del código que cambiaron de una versión a otra y llevar un **control** sobre los cambios en el tiempo.



DEV.F

¿Que es github?



dev

GIT != GITHUB



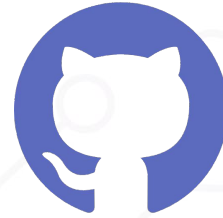
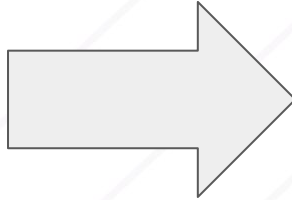
Es importante recalcar que git & github no son lo mismo, pero si **podemos usar ambas herramientas** para implementarlas en un proyecto de software.



GitHub es una **plataforma de desarrollo colaborativo** para alojar proyectos utilizando el sistema de control de versiones GIT.



Mientras que git se encarga de realizar el manejo de los archivos en “**local**”.



GitHub se encarga de realizar el manejo de los archivos en “**la nube**”.

DEV.F

Conceptos básicos de git y github



Repositorios de GitHub

Top Repositories

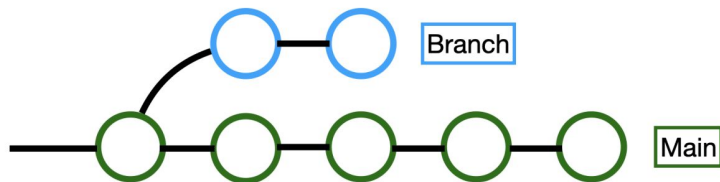


New

Find a repository...

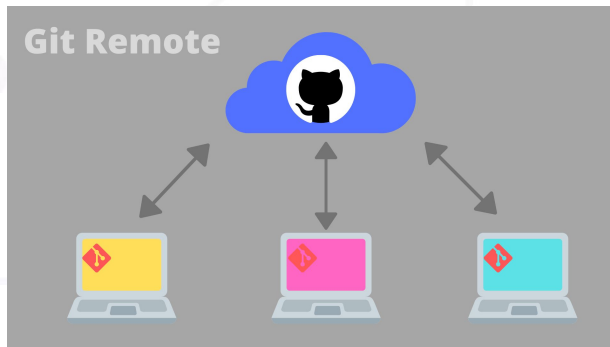
Un **repositorio** en GIT no es más que un “nuevo proyecto”.

Ramas



- Una rama la podemos ver como una **versión** de los archivos que se encuentran en nuestro repositorio, podemos crear tantas como queramos.
- Al crear un nuevo repositorio siempre se crea una rama por default llamada “**main**”.

Remotes



- Un remote no es más que una versión de tu proyecto alojada en un **servidor** que no está en tu máquina local
- Es la **conexión** entre tu local y la nube, entre git y github.
- Al crear un nuevo repositorio siempre se crea un remote default llamado "**origin**".

Configuración de Git

- Usando el flag “`--global`” podemos establecer la configuración de forma global y realizarla una sola vez.

```
git config --global user.name "César Guerra"  
git config --global user.email "cesar@devf.mx"
```



Nota: Podemos verificar la configuración actual con:

```
git config --list
```

Comandos básicos de Git

git clone url_repo -> Clonar un repo de github en local.

git status -> Nos muestra el estado actual de los archivos del repo, cuando están en rojo git no considera los cambios, cuando están en verde los considera.

git add nom_archivo -> *Le decimos a git que considere los cambios de un archivo (". para agregar todos).*

git commit -m "comentario" -> Le decimos a git que nuestros cambios están listos y agrega ese punto de historia al historial.

git diff nom_archivo -> *Muestra los cambios realizados en el archivo.*

git push nom_remote nom_rama -> Subir nuestro último commit a github.

git pull nom_remote nom_rama -> Bajar los últimos cambios de github a local.

DEV.F

Reto, subir código a github



Receta de cocina para subir a github

1. Crear nuevo repo en github.
2. `git clone url_repo` -> Clonar un repo de github en local.
3. Realizar cambios al código en local.
4. `git add .` -> Le decimos a git que considere los cambios de todos los archivos.
5. `git commit -m "comentario"` -> Le decimos a git que nuestros cambios están listos.
6. `git push nom_remote nom_rama` -> Subir nuestro último commit a github.
7. Repetir desde el paso 3.