**Comandos importantes de git:**

**git --version**

para poder ver que versión de git esta instalado

**git config --global user.name "Nombre Apellido"**

para configurar el user.name

**git config --global user.email email@email.com**

para configurar el user.email

git config --list

para poder revisar la configuración que realice anteriormente o la general git config user.name

para ver el user name que configure

git config user.email

para ver el user.email que configure

git help

para ver la ayuda que me brinda git poder escribir en el orden los posibles comandos de git

git config --help

la ayuda del config de git

**git init**

para crear un nuevo repositorio local git.

**git init mi\_carpeta**

Como alternativa, puedes crear un repositorio dentro de un nuevo directorio especificando el nombre del proyecto en este caso mi\_carpeta

**git status**

git status muestra la lista de los archivos que se han cambiado junto con los archivos que están por ser preparados o confirmados.

***git add nombre\_archivo***

git add se usa para agregar archivos

***git add .***

git add . se usa para agregar todos los archivos

**git commit -m "mensaje"**

git commit creará una instantánea de los cambios y la guardará git log --oneline

**git branch mi-rama**

git branch se usa para listar, crear o borrar ramas

**git branch -l**

lista ramas

**git checkout mi\_rama**

cambiar ramas

git branch -D mi\_rama

eliminar rama, solo si no la van a usar más

git branch merge mi\_rama

git rm --cached -r node\_modules/ => Elimina del trackeo (git add) la carpeta node\_modules

EJEMPLO DE LA CLASE

domih@DESKTOP-DF3I8VT MINGW64 ~/Documents (master)

$ git --version

git version 2.30.0.windows.2

domih@DESKTOP-DF3I8VT MINGW64 ~/Documents (master)

$ git cofig --global user.name "andrea profe"

git: 'cofig' is not a git command. See 'git --help'.

The most similar command is

config

domih@DESKTOP-DF3I8VT MINGW64 ~/Documents (master)

$ git config --global user.name "andrea profe"

domih@DESKTOP-DF3I8VT MINGW64 ~/Documents (master)

$ git config --global user.email paolaandreanoemi@gmail.com

domih@DESKTOP-DF3I8VT MINGW64 ~/Documents (master)

$ git config --list

http.sslcainfo=C:/Program Files/Git/mingw64/ssl/certs/ca-bundle.crt http.sslbackend=openssl

diff.astextplain.textconv=astextplain

filter.lfs.clean=git-lfs clean -- %f

filter.lfs.smudge=git-lfs smudge -- %f

filter.lfs.process=git-lfs filter-process

filter.lfs.required=true

credential.helper=manager-core

core.autocrlf=true

core.fscache=true

core.symlinks=true

pull.rebase=false

credential.https://dev.azure.com.usehttppath=true

init.defaultbranch=master

user.name=andrea profe

user.email=paolaandreanoemi@gmail.com

core.repositoryformatversion=0

core.filemode=false

core.bare=false

core.logallrefupdates=true

core.symlinks=false

core.ignorecase=true

domih@DESKTOP-DF3I8VT MINGW64 ~/Documents (master)

$ git config user.name

andrea profe

domih@DESKTOP-DF3I8VT MINGW64 ~/Documents (master)

$ git config user.email

paolaandreanoemi@gmail.com

domih@DESKTOP-DF3I8VT MINGW64 ~/Documents (master)

$ git help

usage: git [--version] [--help] [-C <path>] [-c <name>=<value>] [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path] [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare] [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>] <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)

clone Clone a repository into a new directory init Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)

add Add file contents to the index

mv Move or rename a file, a directory, or a symlink restore Restore working tree files

rm Remove files from the working tree and from the index sparse-checkout Initialize and modify the sparse-checkout

examine the history and state (see also: git help revisions)

bisect Use binary search to find the commit that introduced a bug diff Show changes between commits, commit and working tree, etc grep Print lines matching a pattern

log Show commit logs

show Show various types of objects

status Show the working tree status

grow, mark and tweak your common history

branch List, create, or delete branches

commit Record changes to the repository

merge Join two or more development histories together rebase Reapply commits on top of another base tip reset Reset current HEAD to the specified state switch Switch branches

tag Create, list, delete or verify a tag object signed with GPG

collaborate (see also: git help workflows)

fetch Download objects and refs from another repository pull Fetch from and integrate with another repository or a local branch

push Update remote refs along with associated objects

'git help -a' and 'git help -g' list available subcommands and some concept guides. See 'git help <command>' or 'git help <concept>' to read about a specific subcommand or concept.

See 'git help git' for an overview of the system.

domih@DESKTOP-DF3I8VT MINGW64 ~/Documents (master)

$ git init

Reinitialized existing Git repository in C:/Users/domih/Documents/.git/

domih@DESKTOP-DF3I8VT MINGW64 ~/Documents (master)

$ git init proyecto-arquitectura

Initialized empty Git repository in C:/Users/domih/Documents/proyecto arquitectura/.git/

domih@DESKTOP-DF3I8VT MINGW64 ~/Documents (master)

$ git status

warning: could not open directory 'Mi música/': Permission denied warning: could not open directory 'Mis imágenes/': Permission denied warning: could not open directory 'Mis vídeos/': Permission denied On branch master

Untracked files:

(use "git add <file>..." to include in what will be committed) Default.rdp

My Games/

Zoom/

cache/

desktop.ini

elproyecto/

mirepositorio/

proyecto-arquitectura/

nothing added to commit but untracked files present (use "git add" to track)

domih@DESKTOP-DF3I8VT MINGW64 ~/Documents (master)

$ git add index.html

domih@DESKTOP-DF3I8VT MINGW64 ~/Documents (master)

$ ls

Default.rdp 'My Games'/ elproyecto/

'Mi música'@ Zoom/ index.html

'Mis imágenes'@ cache/ mirepositorio/

'Mis vídeos'@ desktop.ini proyecto-arquitectura/

domih@DESKTOP-DF3I8VT MINGW64 ~/Documents (master)

$ git commit -m "subo index modificado title"

[master e41669c] subo index modificado title

1 file changed, 1 insertion(+), 1 deletion(-)

domih@DESKTOP-DF3I8VT MINGW64 ~/Documents (master)

$ git log --oneline

e41669c (HEAD -> master) subo index modificado title

a727d1d (ramarenueva, rama, laramanueva) borre h2

c4cad0d cambios en index

¿Que es **git ?**

**Git** es un sistema de control de versiones. Un sistema de control de versiones nos va a servir **para** trabajar en equipo de una manera mucho más simple y optima cuando estamos desarrollando software

¿Que es **git init?**

es un comando que se utiliza una sola vez durante la configuración inicial de un repositorio nuevo. Al ejecutar este comando, se creará un nuevo subdirectorio . git en tu directorio de trabajo actual. También se creará una nueva rama maestra.

¿Que es un repositorio?

Un **repositorio** o “repo” es un directorio donde se almacenan los archivos de tu proyecto. Puede estar ubicado en el almacenamiento de **GitHub** o en un **repositorio** local en tu computadora. Puedes almacenar archivos de código, imágenes, audios o todo lo relacionado con el proyecto en el **repositorio**

¿Que es commit?

cada cambio que se realiza por cada desarrollador

¿Qué es un Branch ?

Es una rama diferente a la rama master que tenemos y se utiliza para trabajar de forma colaborativa y eficaz con esto podremos asignarnos tareas y ordenar mejor nuestro proyecto.

¿Qué es un Merge?

Es una fusion entre ramas.

Los conceptos entre Branch y Merge suelen ser dificiles de aprender pero con los ejemplos podrás tener una mejor idea de que es y para que utiliza.