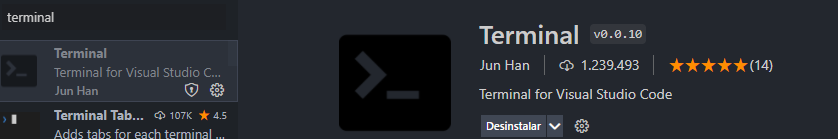
**Curso de GIT + GitHub**

Antes de empezar necesitaremos instalar lo siguiente:



Visual Studio Code: <https://code.visualstudio.com/>

Un navegador

GIT: <https://git-scm.com/>

Recomendable (Icono) <https://marketplace.visualstudio.com/items?itemName=PKief.material-icon-theme>

Comprobar que GIT está instalado:



**Comandos generales**

Ver la versión de GIT

git --version

Ayuda

git help

Configurar usuario y email

git config --global user.name "Jose"

git config --global user.email "jjoseaangel99@gmail.com"

Mirar o modificar que es correcto el email y usuario (con a podemos modificar y si presionamos esc + :wq! salimos)

git config --global -e

Iniciar un repositorio

git init

Información

git status

Prepara el archivo para el commit

git add index.html (Para un archivo)

git add . (Para todos los archivos)

Hacer un commit

git commit -m "Primer commit"

Si ya hemos hecho el primer commit podemos usar esta forma

git commit -am "commit"

Cambiar el mensaje del commit

git commit --amend -m "notas modificadas"

Ver log

git log

Ver historial más profundo que log

git reflog

Si a un archivo se le ha hecho un add con reset podemos ponerlo en el estado que estaba antes del add

git reset index.html

Volver al commit anterior

git checkout -- .

Ver en que rama estamos trabajando

git branch

Cambiar el nombre de la rama master

git branch -m master main

git config --global init.defaultBranch main (Hacerlo de forma global)

Es una extensión de archivo que pesa muy poco para que haya algo en el repositorio y se pueda hacer commit de el

.gitkeep

Si queremos que git ignore algún archivo, debemos crear el archivo .gitignore y dentro de él escribir los directorios o archivos que deseamos que ignore

.gitignore

Crear un alias para hacer los comandos más cortos

git config alias.s "status --short"

git s (Así quedaría el comando)

Crear un alias del log (RECOMENDADO)

git config --global alias.lg "log --graph --abbrev-commit --decorate --format=format:'%C(bold blue)%h%C(reset) - %C(bold green)(%ar)%C(reset) %C(white)%s%C(reset) %C(dim white)- %an%C(reset)%C(bold yellow)%d%C(reset)' --all"

git lg (Así quedaría el comando)

Cambiar el nombre de un archivo

git mv nombre1.txt nombre\_nuevo.txt

Eliminar un archivo

git rm nombre\_archivo-txt

Crear una rama

git branch nombre\_rama

Movernos de rama

git checkout nombre\_rama

Crear y movernos de rama

git checkout -b nombre\_rama

Unir ramas

git merge nombre\_rama

Borrar una rama / Borrar una rama de forma forzada

git branch -d nombre\_rama

git branch -d nombre\_rama -f

Ver los tags

git tag

Crear un tag (El tag se aplica al último commit)

git tag nombre\_tag

Crear un release tag (El tag se aplica al último commit)

git tag -a v1.0.0 -m "Version 1.0.0 lista"

Crear un release tag de un commit antiguo (hay que introducir el hast se puede ver en el log)

git tag -a v0.1.0 752331 -m "Version Alpha"

Eliminar un tag

git tag -d nombre\_tag

Ver información del tag

git show v0.1.0

Agregar algo al stash (El stash es como un contenedor donde almacenar información que luego podemos recuperar)

git stash

Crear un stash y ponerle una descripción

git stash save "este es un stash"

Recuperar el stash la información del stash (Se utiliza para cuando solo hay uno)

git stash pop

Recuperar la información de un stash en concreto (con el list podemos ver el id que hay que poner)

git stash apply stash@{2}

Ver los stash

git stash list

Borrar todos los stash

git stash clear

Borrar un stash en concreto

git stash drop stash@{2}

Ver la información de un stash

git stash show stash@{2}

Para unir las ramas cuando hay archivos que dan problemas (Desde la rama que no es la master), tras esto nos cambiamos a la rama master y hacemos el merge

git rebase master

Subir los tags a GitHub

git push --tags

**Solucionar warning de pull**

**git config --global pull.ff only**

Clonar un repositorio

git clone https://url/del/repositorio