**STOS SON COMANDOS BASICO DE GIT HECHOS POR MI**

* ***CON ESTE COMANDO INICIAMOS NUESTRO REPOSITORIO O CREA EL REPOSITORIO EN GIT***

git init

* ***CON ESTE COMANDO AGREGAMOS O GRABAMOS LOS ARCHIVOS NUEVOS A NUESTRO REPOSITORIO***

git add .

* ***CON ESTE COMANDO AGREGAMOS O GRABAMOS LOS ARCHIVOS NUEVOS A NUESTRO REPOSITORIO, PERO CON EL NOMBRE DE ESE ARCHIVO***

git add index.html

* ***CON ESTE COMANDO AGREGAMOS O GRABAMOS LOS ARCHIVOS NUEVOS A NUESTRO REPOSITORIO, PERO CON LA EXTENCION DEL TIPO DE ARCHIVO***

git add \*.png

* ***¡¡CON ESTE COMANDO QUITAMOS UN ARCHIVO O CARPETA QUE NO QUEREMOS EN NUESTRO REPOSITORIO OJO!! CON LA EXTENSION O NOMBRE DEL ARCHIVO Y SU EXTENSION***

git reset \*.xml

git reser node/

* ***CON ESTE COMANDO AGREGAMOS O GRABAMOS LOS ARCHIVOS NUEVOS A NUESTRO REPOSITORIO, PERO DE UNA CARPETA***

git add css/

* **CON ESTE COMANDO AGREGAMOS O GRABAMOS LOS ARCHIVOS QUE FUERON MODIFICADO EN NUESTRO REPOSITORIO**

git add -A

* ***CON ESTE COMANDO MIRAMOS LOS ULTIMOS CAMBIOS QUE SEAN HAN HECHO DE NUESTRO PROYECTO***

git status

* ***CON ESTE COMANDO MIRAMOS LOS ULTIMOS CAMBIOS QUE SEAN HAN HECHO DE NUESTRO PROYECTO, PERO RESUMIDO***

git status -s

* ***CON ESTE COMANDO MIRAMOS LOS ULTIMOS CAMBIOS QUE SEAN HAN HECHO DE NUESTRO PROYECTO RESUMIDO Y EN QUE RAMA ESTAMOS TRABAJANDO***

git status -s -b

* ***¡¡CON ESTE COMANDO HACEMOS NUESTRAS CAPTURAS DE FOTOS O COMMIT OJO!! HAY QUE PONER UN NOMBRE***

git commit -m "Nombre del commit o de la foto"

* ***¡¡CON ESTE COMANDO ES COMO SI REALIZAREMOS CONTROL Z DE CUALQUIER MODIFICACION OJO!! SI YA HABIA HECHO UN COMMIT***

git checkout -- .

* ***ESTE COMANDO MIRAMOS TODOS LOS COMMIT REALIZADOS EN NUESTRO PROYECTO***

git log

* ***ALGUNOS EN LA PRÓXIMA CLASE HAN PRESENTADO ESTE PROBLEMA CON EL CRLF, NO ES NADA SERIO, ES BÁSICAMENTE UNA INTERPRETACIÓN DE UN CARACTER. SIMPLEMENTE EJECUTEN ESTE COMANDO SI PRESENTAN EL ERROR:***

git config core.autocrlf true

* ***ESTE COMANDO MIRAMOS TODOS LOS COMMIT REALIZADOS EN NUESTRO PROYECTO, PERO MAS SIMPLIFICADO***

git log --oneline

* ***ESTE COMANDO MIRAMOS TODOS LOS COMMIT REALIZADOS EN NUESTRO PROYECTO, PERO LA INFORMACION ES DECORADA***

got log --oneline -- decorate --all --graph

**CON ESTE COMANDO CREAMOS LOS ALIAS**

* ***ESTE ES UN ALIAS PARA LOG SIMPLIFICADO***

git config --global alias.lg "log --oneline --decorate --all --graph"

* ***ESTE ES UN ALIAS PARA STATUS SIMPLIFICADO***

git config --global alias-st "status -s -b"

* ***CON ESTE COMANDO MIRAMOS CAMBIOS HICIMOS***

git diff

* ***CON ESTE COMANDO MIRAMOS CAMBIOS HICIMOS MAS DETALLADO***

git diff --staged

* ***CON ESTE COMANDO QUITAMOS UN ARCHIVO CLARO CON LA EXTENSION DE NUESTRO REPOSITORIO***

git reset HEAD README.md

* ***CON ESTE COMANDO REESTABLECEMOS EL ARCHIVO QUE BORRAMOS CON EL COMANDO ANTERIOR DE NUESTRO REPOSITORIO CLARO CON EL NOMBRE DEL ARCHIVO Y EXTENSION RESPECTIVAMENTE***

git checkout -- README.md

**PARA TRABAJAR YA CON GITHUB**

* ***CON ESTE COMANDO MIRAMOS SI ESTA REMOTO NUESTRO PROYECTO***

git remote

* ***CON ESTE COMANDO MIRAMOS SI ESTA REMOTO CON VERSION NUESTRO PROYECTO***

git remote -v

* ***CON ESTE COMANDO SUBE LOS PUSH O RELEASES A GITHUB***

git push --tags

* ***CON ESTE COMANDO BAJAMOS LOS CAMBIOS HECHOS EN GITHUB***

git pull

* ***CON ESTE COMANDO SUBE LOS CAMBIO A GITHUB***

git push

**CON ESTE COMANDO CREO UN ARCHIVO PARA QUE GIT LO IGNORE**

Creo un archivo .gitignore en la principal de nuestro proyecto y allí agregamos los archivos o carpetas que no queremos que se carguen en nuestro repositorio

Ejemplo:

\*.mp4 = este ignora todos los archivos con esa extensión

presentacion/ = Este ignora todos los archivos de esa carpeta

index.js = Este ignora este archivo solamente.

* ***CON ESTE COMANDOS SUBIMOS NUESTRO REPOSITORIO A GITHUB***

git remote add origin https://a clonar

git push -u origin master