|  |
| --- |
| GIT BASH (GITHUB) CMD (WINDOWS) |
| MANUAL - COMANDOS |
|  |
|  |
|  |

|  |
| --- |
|  |

**Git Bash (GitHub) - CMD (Windows)**

Contenido

[Git Bash 3](#_Toc512777053)

[Links: 3](#_Toc512777054)

[Configuracion: 3](#_Toc512777055)

[Crear un repositorio: 3](#_Toc512777056)

[Inicializar un repositorio vacio 3](#_Toc512777057)

[Clonar repositorio 3](#_Toc512777058)

[Staging index -> GIT ADD: 3](#_Toc512777059)

[*Deshacer cambios del Staging index (ANTES DE -> GIT COMMIT):* 4](#_Toc512777060)

[Confirmar cambios -> GIT COMMIT: 4](#_Toc512777061)

[Historial de cambios: 4](#_Toc512777062)

[*1.* *Ver GIT COMMIT realizados:* 4](#_Toc512777063)

[*2.* *Examinar/Levantar COMMIT previo:* 5](#_Toc512777064)

[*3.* *Crear Tag/Etiqueta/Alias a COMMIT previo:* 5](#_Toc512777065)

[Ramas: 5](#_Toc512777066)

[*1.* *Crear/Saltar/Eliminar:* 5](#_Toc512777067)

[*2.* *Fusionar:* 6](#_Toc512777068)

[Push a repositorios remotos: 6](#_Toc512777069)

[Actualizando repositorios remotos con comando Pull: 6](#_Toc512777070)

[Comandos varios: 6](#_Toc512777071)

[CMD (Windows) 7](#_Toc512777072)

[Links: 7](#_Toc512777073)

[Comandos: 7](#_Toc512777074)

[Comandos internos: 7](#_Toc512777075)

[Comandos externos: 8](#_Toc512777076)

# Git Bash

## Links:

https://gitforwindows.org/ (descarga Git

https://www.youtube.com/watch?v=QGKTdL7GG24 (Video)

https://git-scm.com/book/es/v1/Ramificaciones-en-Git-%C2%BFQu%C3%A9-es-una-rama%3F

## Configuracion:

git config --global user.name "*NicoDelRio*"

git config --global user.name (muestra el nombre del usuario activo)

git config --global user.email "<*nicolas-delrio@hotmail.com*>"

git config --global user.email (muestra el email del usuario activo)

## Crear un repositorio:

### Inicializar un repositorio vacio

1. Situarse en el ***directorio de trabajo***. (ver: CMD (Windows)-> Comandos-> Comandos internos) (el ***directorio de trabajo*** es el directorio donde se crea el repositorio)
2. git init (crea repositorio, carpeta ".git" que almacena los datos del repositorio)
3. git init --bare <*nombre*>.git (crea repositorio desnudo, solo se pueden enviar cambios desde otro repositorio).

### Clonar repositorio

1. Situarse en el ***directorio de trabajo***.
2. git clone <*direccion\_repositorio\_a\_clonar*>
   1. Puede ser carpeta Local
   2. Puede ser dirección HOST (URL https://github.com/NicoDelRio/BroccoliChat.git)

## Staging index -> GIT ADD:

*Es un área temporal en donde se guardan los cambios (archivos, carpetas) (comando-> git add) que se enviarán al repositorio en Git (comando-> git commit).*

1. *Añadir/modificar/eliminar archivos/carpetas del proyecto dentro del directorio de trabajo*
2. *Seleccionar archivos cuyos cambios se quieren enviar y agregarlos al staging index.*
3. *Confirmar ("commit") esos cambios, almacenándolos en el repositorio.*

git add . (agrega al staging index los archivos que hayan sufridos cambios)

git add *<nombre>.<extencion>*  (agrega al staging index el archivo - se debe respetar mayus/minusc).

git status (muestra los archivos que se están por añadir en el sig commit "en verde" , y los archivos que sufrieron cambios pero sin haberlos agregados en el staging index con git add "en amarillo").

git rm *<nombre>.<extencion>*  (eliminar del Git)

rm *<nombre>.<extencion>*  (eliminar del sistema)

### *Deshacer cambios del Staging index (ANTES DE -> GIT COMMIT):*

git reset HEAD *<nombre>.<extencion>*  (saca archivo del staging index)

git checkout -- *<nombre>.<extencion>*  (deshace cambios del archivo del staging index posteriores al anterior git commit)

## Confirmar cambios -> GIT COMMIT:

git commit (confirma cambios del staging index)

## Historial de cambios:

### *Ver GIT COMMIT realizados:*

git log (ver los commit realizados, con detalle)

git log --oneline (ver los commit realizados, resumen)

git log --oneline --decorate

### *Examinar/Levantar COMMIT previo:*

git checkout <*codigo\_commit o tag\_commit*> (***Restaura*** archivos y carpetas al momento de realizar el commit).

*Funciona como una rama.*

### *Crear Tag/Etiqueta/Alias a COMMIT previo:*

git tag <*alias\_nuevo*> (Agregar tag en el commit que se está parado).

git tag <*alias\_nuevo*> <*codigo\_commit*> (Agregar tag a un commit previo).

git tag (Ver tags creados)

## Ramas:

*Por defecto, se trabaja en la rama "master".*

*Para entender realmente cómo ramifica Git, previamente hemos de examinar la forma en que almacena sus datos. Recordando lo citado en el capítulo 1, Git no los almacena de forma incremental (guardando solo diferencias), sino que los almacena como una serie de instantáneas (copias puntuales de los archivos completos, tal y como se encuentran en ese momento).*

*En cada confirmación de cambios (commit), Git almacena un punto de control que conserva: un apuntador a la copia puntual de los contenidos preparados (staged), unos metadatos con el autor y el mensaje explicativo, y uno o varios apuntadores a las confirmaciones (commit) que sean padres directos de esta (un padre en los casos de confirmación normal, y múltiples padres en los casos de estar confirmando una fusión (merge) de dos o más ramas).*

### *Crear/Saltar/Eliminar:*

git branch <*nombre\_rama*> (crear rama)

git branch (muestra ramas del repositorio local. La rama actual está marcada con \*)

git branch --all (muestra ramas del repositorio local y remoto asociado)

git checkout <*nombre\_rama*> (salta a la rama)

git checkout -b <*nombre\_rama*> (crea y salta a la rama)

git branch -d <*nombre\_rama*> (borra rama)

### *Fusionar:*

*1. Ir a la rama de destino (rama padre)*

git merge <*rama\_hija*> (fusiona rama hija con rama padre)

*2. Revisar si crearon conflictos con* git status. "both modified: <archivo>"

*3. En los archivos con conflictos, Git crea marcas especiales dentro de dicho archivo para modificarlo y elegir la mejor version.*

*4. Git add / git commit/...*

## Push a repositorios remotos:

git push <*repositorio\_destino*> < *rama\_a\_enviar(master)*> (sube cambios al repositorio destino)

## Actualizando repositorios remotos con comando Pull:

git pull <*repositorio\_origen*> < *rama\_destino(master)*> (actualiza cambios del repositorio a la rama)

## Comandos varios:

popd (buscar que hace)

touch *<nombre>.<extencion*> (crea archivo vacio)

# CMD (Windows)

## Links:

https://www.taringa.net/posts/info/17523260/Todos-Los-Comandos-Para-CMD-WINDOWS.html

## Comandos:

### Comandos internos:

\* CD o CHDIR (Cambia de directorio)

\* CD <NOMBRE\_CARPETA>

\* CD < NOMBRE\_CARPETA/ NOMBRE\_SUBCARPETA>

\* CD .. (sube directorio)

\* DIR (ver por pantalla una lista con los archivos y directorios.(carpetas) que se encuentran en un directorio del disco.)

\* MD o MKDIR <nombre\_carpeta> (crea una nueva carpeta)

\* MV <nombre\_ archivo > (crea una nueva carpeta)

\* RM <nombre\_archivo> (borrar archivo)

\* LS (muestra subcarpetas y archivos)

\* VI <nombre\_archivo> (abre archivo)

\* SHIFT + INSERT (Pegar texto)

\* CLS (Borra todo de la pantalla y regresa el cursor en la esquina izquierda superior)

\* COPY <nombre\_archivo> (copiar archivo)

\* DATE (ver la fecha)

\* ECHO ON (Permite ver las lineas de comandos que son automáticamente ejecutadas.)

\* ECHO OFF (Quita la visulización de lineas de comando automáticamente ejecutadas)

\* PATH (Establece o declara el curso o trayectoria entre directorios)

\* PROMPT (Cambia la línea de visualización de la orden)

\* REN (Renombrar archivos)

\* RD (Para Remover un Directorio que esté totalmente vacío)

\* TIME (Visualiza la hora del reloj interno)

\* VER (Muestra la versión del sistema operativo)

\* VOL (Muestra la etiqueta del volumen y el número de serie del disco)

### Comandos externos:

\* CHKDSK Comprueba el disco y da un informe de su estado.

\* FORMAT Orden del sistema operativo MS-DOS cuya misión es formatear las unidades de almacenamiento (discos duros y disquetes).

\* DISKCOPY Nos permite hacer una copia idéntica de un disquete a otro, pertenece al grupo de las órdenes externas.

\* DOSKEY Nos permite mantener residente en memoria RAM las órdenes que han sido ejecutadas en el punto indicativo.

\* KEYB Configura un teclado para un idioma especifico.

\* DELTREE Usado para borrar un directorio raíz no importa que contenga subdirectorios con todos sus contenidos.

\* TREE Su función es presentar en forma gráfica la estructura de un directorio raíz.

\* SCANDISK Si se esta ejecutando en windows abre scandisk.

\* LABEL Crea cambia o elimina la etiqueta del volumen del disco.

\* MEM Presenta un informe sobre la memoria RAM.