



## COMANDOS MAS UTILIZADOS

COMANDO	DESCRIPCION	APLICACIÓN o EJEMPLO
<b>cd</b>	Accesar/ ingresar a directorio ó carpeta/subir o entrar en el siguiente nivel de directorio	<b>USUARIO-MINGW64~/escritorio/</b> \$ cd dh RTA: <b>USUARIO-MINGW64~/escritorio/dh</b>
<b>cd ..</b>	Retornar al directorio ó carpeta anterior, bajar o salir, retornar al nivel anterior de directorio	<b>USUARIO-MINGW64~/escritorio/dh</b> \$ cd .. RTA: <b>USUARIO-MINGW64~/escritorio/</b>
<b>pwd</b>	Nos muestra la ubicación donde estamos posicionados o ubicados dentro de la terminal	<b>USUARIO-MINGW64~/escritorio/</b> \$ pwd RTA: <b>USUARIO-MINGW64~/escritorio</b>
<b>mkdir</b>	Crear un nuevo directorio o carpeta, debemos indicar el nombre que le vamos a asignar	<b>USUARIO-MINGW64~/escritorio/</b> \$ mkdir ejemplo RTA: <b>USUARIO-MINGW64~/escritorio/ejemplo</b>
<b>rmdir</b>	Elimina un directorio o carpeta que se encuentra vacía.	<b>USUARIO-MINGW64~/escritorio/ejemplo</b> \$ rmdir ejemplo RTA: <b>USUARIO-MINGW64~/escritorio/</b>
<b>rm -r</b>	Elimina / borrar un directorio o carpeta y su contenido, debemos indicar el nombre del directorio que deseamos eliminar	<b>USUARIO-MINGW64~/escritorio/</b> \$ rm -r ejemplo RTA:
<b>touch</b>	Crear un nuevo archivo, debemos indicar el nombre que le vamos a asignar al archivo. Lo más recomendable es colocarle la extensión (de lo contrario sería un archivo de bits)	<b>USUARIO-MINGW64~/escritorio/</b> \$ touch Archivo.js RTA: <b>Archivo.js</b>
<b>rm</b>	Eliminar / borrar un archivo, debemos indicar el nombre del archivo que deseamos eliminar	<b>USUARIO-MINGW64~/escritorio/</b> \$ rm Archivo.js RTA:
<b>nano</b>	Nos permite modificar un archivo accediendo al editor de texto de la terminal	<b>USUARIO-MINGW64~/escritorio/</b> \$ rm Archivo.js RTA:
<b>clear ó Ctrl + L</b>	Limpiar la pantalla de la terminal, ambos cumplen el mismo propósito	<b>USUARIO-MINGW64~/escritorio/</b> \$ clear Ctrl + L
<b>ls</b>	Lista los archivos, nos muestra los archivos que hay en la carpeta o directorio en el cual nos encontramos posicionados (pwd)	<b>USUARIO-MINGW64~/escritorio/</b> \$ ls RTA: <b>Archivo.js</b>
<b>ls -a</b>	También lista o nos muestra todos los archivos que hay en la carpeta o directorio en el cual estamos posicionado, pero esta vez sumándole "-a", nos mostrara adicionalmente los archivos ocultos.	<b>USUARIO-MINGW64~/escritorio/</b> \$ ls -a RTA: <b>Archivo.js</b> <b>..git</b>
<b>ctrl + insert</b>	Este atajo nos permite copiar un texto o selección en la terminal	 Copy
<b>Shift + insert</b>	Este atajo nos permite pegar un texto o selección en la terminal	 Paste

<b>git init</b>	Nos permite crear un directorio oculto / repositorio , en la ubicación donde nos encontremos posicionados	<pre>USUARIO-MINGW64~/escritorio/dh \$ git init RTA:USUARIO-MINGW64~/escritorio/dh (main)</pre>
<b>git add</b>	Nos permite agregar o hacerle seguimiento a un archivo al repositorio, indicando el nombre del archivo que queremos agregar o seguir	<pre>USUARIO-MINGW64~/escritorio/dh \$ git add archivo.js RTA: archivo.js ("en seguimiento o stage área")</pre>
<b>git add .</b>	Nos permite agregar o hacerle seguimiento a todos los archivos a la vez	<pre>USUARIO-MINGW64~/escritorio/dh (main) \$ git add . RTA: Archivo.js Archivo2.js</pre>
<b>git status</b>	Nos permite ver el estado de seguimiento de los archivos – cuales hay y cuales se les esta dando seguimiento ó no.	<pre>USUARIO-MINGW64~/escritorio/(main) \$ git status RTA: Archivo.js Archivo2.js</pre>
<b>git config user.name</b>	El solo comando nos muestra el nombre de usuario establecido en el repositorio y acompañado nos permite incluir el nombre del usuario que le asignaremos al repositorio	<pre>USUARIO-MINGW64~/escritorio/ (main) \$ git config user.name "usuario" RTA: Archivo.js Archivo2.js</pre>
<b>git config user.email</b>	El solo comando nos muestra el email de usuario establecido en el repositorio y acompañado nos permite incluir el nombre del email que le asignaremos al repositorio	<pre>USUARIO-MINGW64~/escritorio/ (main) \$ git config user.email "usuario@gmail.com" RTA: Archivo.js Archivo2.js</pre>
<b>-- global</b>	si adicionamos- -global a los 2 comando s anteriores instruiremos al pc para plantar por defecto registro de usuario y correo o email	<pre>USUARIO-MINGW64~/escritorio/ (main) \$ git config --global user.name "usuario" \$ git config --global user.email "usuario@gmail.com"</pre>
<b>git commit – m</b>	Nos permite hacer un commit – se genera un punto histórico, hacer un registro o foto de los archivos en el estado actual (nos arroja un numero de identificación denominado “hash”)	<pre>USUARIO-MINGW64~/escritorio/(main) \$ git commit -m "nombre del commit"</pre>
<b>git push –u origin</b>	Nos permite enviar (empujar) los commits con los archivos que se siguieron, al repositorio remoto en la nube	<pre>USUARIO-MINGW64~/escritorio/ (main) \$ git push -u origin "nombre repo remoto"</pre>
<b>git pull</b>	Nos permite traer la información de los archivos que se encuentran en el repositorio remoto para actualizar la información nueva o modificada frente a los que se encuentran en el repo local	<pre>USUARIO-MINGW64~/escritorio/ (main) \$ git pull -u origin "main"</pre>
<b>git clone</b>	Nos permite crear en el repo local (pc) una copia de los archivos existentes y/o modificados en que se encuentran remoto	<pre>USUARIO-MINGW64~/escritorio/ (main) \$ git clone + "link repo"</pre>
<b>git branch</b>	Nos permite Crear una rama, es decir un repositorio dentro de otro repositorio indicando el nombre que le deseamos dar a esa nueva rama	<pre>USUARIO-MINGW64~/escritorio/ (main) \$ git branch nuevarama</pre>
<b>git checkout</b>	Nos permite cambiarnos de una rama, a otra rama indicando el nombre a la deseamos dar a esa nueva rama	<pre>USUARIO-MINGW64~/escritorio/ (main) \$ git branch main</pre>

- a.** ¿Quién inventó el sistema de control de versión Git y por qué?

Inventado por Linus Torvalds

Frente a la gran cantidad de archivos de código de fuente y las distintas versiones de los mismos, considero regularizar y estandarizar un mantenimiento continuo y apropiado que repercutiera en una compatibilidad adecuada, que fuera confiable y eficiente.

- b.** ¿A quién pertenece actualmente Github y por qué?

Actualmente git hub pertenece a Microsoft, de acuerdo a la compra de qué hizo esta empresa, en virtud del auge colaborativo que representa esta plataforma, encaminada especialmente a enfocarse en el desarrollo, también atendiendo a el aporte que representa para los desarrolladores.

- c.** ¿Hay otra forma que no sea la terminal para trabajar con Github?

Se puede trabajar desde una interfaz grafica, también proveida por github, la cual nos permite realizar el seguimiento de archivos y efectuar los commits, cumpliendo las mismas actividades o parámetros de la terminal, pero desde la interfaz grafica