Sintaxis y Semántica de los Lenguajes
Comandos elementales de git

- ***git init***

Este comando permite inicializar una carpeta de trabajo asociándose a git, lo que nos permitirá generar un repositorio. Dentro de la carpeta que apliquemos el comando init se generará una carpeta oculta con el nombre .git donde se guardarán archivos referidos a nuestro sistema de control de versionado de archivos (CVS)

- ***git config***

Este comando establece diversos tipos de parámetros relacionados con toda la actividad que realicemos dentro de git. Principalmente lo que nos interesa conocer en este punto son la configuración de un nombre de usuario y un correo electrónico. Esto será necesario a la hora de hacer ***commit*** en el repositorio.

A modo de ejemplo, estos dos comandos siempre los debemos configurar previo a hacer cualquier confirmación de cambios (commit):

git config --global user.email “santiago.ferreiros@gmail.com”
git config --global user.name “Santiago Ferreiros Cabrera”

- ***git add***

Este comando nos permite agregar nuevos archivos a nuestra área de trabajo, como así también confirmar las modificaciones a nuestra área de trabajo. Importante: El área de trabajo, será el espacio intermedio entre las modificaciones que vamos realizando en nuestra carpeta y el repositorio.

El comando git add . permite agregar todos los archivos dentro del entorno de trabajo para no tenerlos que nombrar uno por uno

- ***git status***

Este comando nos informa sobre el estado de la carpeta asociada a git. ¿Qué archivos han sido agregados? ¿Qué están modificados? ¿Qué archivos ya están en el área de stage? Es importante revisar esto antes de realizar un commit. Veremos los siguientes estados:

- Untracked files → Se refiere a archivos nuevos
- Change no stages → Modificaciones aún no agregadas a la staging area
- Change to be committed → Se refiere a las modificaciones ya agregadas a la staging area que están listas para el próximo commit

➢ ***git commit***

Este comando realizará la confirmación en el repositorio de todo lo que se encuentra en la staging area. Previamente a utilizar este comando necesitamos haber configurando un usuario y correo electrónico. Lo necesita git para saber quién está haciendo el commit en el repositorio.

Por ejemplo: git commit -m “El mensaje que queremos dejar acompañando al commit”
En general, es muy importante dejar un mensaje descriptivo a la hora de realizar un commit, esto nos servirá luego para llevar un mejor control del repositorio. En caso de no poner un mensaje por consulta, lo solicitará. Luego de escribir el mensaje, desde Git Bash, apretar ESC, y escribir :wq en la línea de entrada de la parte inferior.

➢ ***git log***

Nos muestra un registro de cada commit con un comentario, fecha, usuario, correo que se ha realizando en el repositorio. Es un elemento importante para conocer quien ha realizado cada uno de los cambios, aquí es donde se visualizan los mensajes agregando cuando utilizamos el comando commit.

➢ ***git checkout***

Retrocede un commit que hayamos realizado. Más adelante veremos más detalles respecto a estos comandos.

➢ ***git diff***

Para ver las diferencias para un archivo entre los cambios que realice y la versión original. Es para comparar un archivo del working directory que fue modificado con respecto a su versión anterior que se encuentra en el repositorio del último commit (confirmación) realizado. Git reconoce cualquier tipo de archivo que es agregado, pero este comando tendrá sentido para aquellos archivos de texto en los cuales pueda indicarnos que líneas fueron modificadas.

➢ ***git remote***

Este comando permite agregar una dirección en la cual se va a compartir el repositorio para que deje de ser local y otros puedan acceder desde internet. Aquí se indica que se va a compartir pero luego hay que utilizar el comando push

Por ejemplo: git add origin https://github.com/santiagoferreiros/SSL_Ejemplos_Codigo_C

➢ ***git push***

Este comando suba a la dirección indicada con el comando git remote todo el repositorio. En nuestro caso utilizaremos una dirección de gitHub por lo que necesitamos previamente tener creado un usuario de gitHub. Al ejecutar este comando nos pedirá iniciar sesión para poder realizar la subida de nuestro repositorio a la nube.

Comando: git push -u origin master

La palabra master hace referencia a la versión en la que estamos trabajando. Luego veremos más detalles al respecto.

- ***git clone***

Este comando es el inversio a los anteriores. Nos permite copiar desde u remoto a local dirección del repositorio remoto

git clone <dirección del repositorio remoto>

- **.gitignore (archivo)**

Es un archivo dentro de nuestra carpeta de trabajo que permite indicar un listado de carpetas y archivos a ignorar para que cuando ejecutemos el comando git status no figuren dentro del listado. .gitignore no es un comando. Puede resultar cómodo cuando hay un conjunto de archivos que por el momento no nos interesa realizar ningún tipo de commit en el repositorio, dado que puede ser un poco tedioso que siempre aparezcan en el git status y nos evitamos realizar commit erróneos por subir algún archivo que no corresponda.

Para mayor detalle y comprensión pueden buscar en youtube, a mi me fue de utilidad ver el siguiente video, pero hay muchos más: https://www.youtube.com/watch?v=HiXLkL42tMU

Y por supuesto, la página oficial de Git: https://git-scm.com/book/en/v2 donde encontrarán muchos más detalles