¿Por qué usar un sistema de control de versiones como Git?

Git ayuda a mantener una visión de conjunto, a preservar las versiones antiguas y a integrar los cambios de manera coherente. Para ello, Git agrupa una serie de programas de línea de comandos y crea un efectivo entorno de trabajo. Mientras que Linux y macOS contienen utilidades de línea de comandos de Unix y, por lo tanto, son complementarias con Git, Microsoft utiliza su propio símbolo del sistema de Windows.

¿Qué es Git?

Git es un Sistema de Control de Versiones Distribuido (DVCS) utilizado para guardar diferentes versiones de un archivo (o conjunto de archivos) para que cualquier versión sea recuperable cuando lo desee.

Git también facilita el registro y comparación de diferentes versiones de un archivo. Esto significa que los detalles sobre qué cambió, quién cambió qué, o quién ha iniciado una propuesta, se pueden revisar en cualquier momento.

¿Cómo instalar GitBash en Windows?

Git Bash se puede encontrar en la web oficial Git Bash para Windows. Allí encontrarás siempre la versión más actual, con la seguridad de que el archivo ha sido analizado. Al hacer clic en el botón de Download, descargarás el archivo .exe. En principio, puedes abrirlo de la misma manera que otros archivos

¿Cuáles son los comandos principales para utilizar en GitBash? (Presenta una lista describiendo su función).

Comandos

* **pwd:**nos muestra la carpeta actual en la que nos encontramos.
* **mkdir:**nos permite crear carpetas, p. ej. **mkdir** *NuevaCarpeta*
* **touch:**nos permite crear archivos nuevos, p.ej.**touch** *NuevoArchivo.txt*
* **cat:**nos permite ver el contenido de un archivo, p.ej. **cat** *NuevoArchivo.txt*
* **cd:**nos permite cambiarnos de carpeta, p.ej. **cd***NuevaCarpeta.*
* **cd ..** : nos permite regresar al directorio o carpeta anterior.
* **cd o cd ~:**nos lleva a la ruta del usuario.
* **cd /c:**nos vamos al disco C:/.
* **cd -:**nos lleva directamente al ultimo directorio visitado.
* **ls:**nos permite ver los archivos de la carpeta donde estamos actualmente.
* **ls -l:**Ver todos los archivos como una lista en donde incluye el usuario, grupo, permisos sobre el archivo, tamaño, fecha y hora de creación.
* **ls -lh**: Muestra la misma información que ls-l pero con unidades de tamaño, es decir, kb o mb.
* **ls-R**: muestra el contenido de todos los sudirectorios de forma recursiva.
* **ls -S** Ordena los resultados por tamaño de archivo.
* **rm:**Nos permite borrar un archivo o carpeta ej: rm NuevoArchivo.txt
* **rmdir** **“nombre del directorio”/**: borrar un directorio: Solo funciona con directorios vacíos.
* **rm -r ‘nombre de la carpeta’** :me permite eliminar la carpeta y los archivos dentro de ella de forma recursiva.
* **cp** **“nombre del archivo que quremos copiar” “nombre del directorio a donde lo queremos copiar”**: nos permite copiar un archivo.
* **mv** **“el directorio de donde queremos mover/el nombre del archivo” “el directorio hacia donde lo queremos mover”**: nos permite mover un archivo.
* **clear:**nos permite limpiar la pantalla.
* **history:**ver los últimos comandos que ejecutamos y un número especial con el que podemos volver a repetir el comando.

Crea un repositorio de Git y haz tu primer commit.

¿Qué es el Staging?

¿Qué es Branch (rama) y cómo funciona un Merge en Git?

¿Para qué sirve reset y checkout?

¿Cómo fusionar Ramas con Git merge?

¿Qué es GitHub? Haz un video realizando un push y un pull.

¿Qué es un archivo .gitignore?

¿Qué es GitHub Pages?

¿Cómo usar Git Stash?

Comparte tu cuenta de GitHub.

Después de investigar toda esta información sobre Git y GitHub,

elabora un ensayo donde expreses tu opinión sobre el uso de un

repositorio en el desarrollo de proyectos.