**Comandos básicos de GitHub**

* git init: crear un repositorio.
* git add: agregar un archivo a staging.
* git commit -m “mensaje”: guardar el archivo en git con un mensaje.
* git branch: crear una nueva rama.
* git checkout: moverse entre ramas.
* git push: mandar cambios a un servidor remoto.
* git fetch: traer actualizaciones del servidor remoto y guardarlas en nuestro repositorio local.
* git merge: tiene dos usos. Uno es la fusión de ramas, funcionando como un *commit* en la rama actual, trayendo la rama indicada. Su otro uso es guardar los cambios de un servidor remoto en nuestro directorio.
* git pull: fetch y merge al mismo tiempo.

**Comandos para corrección en GitHub**

* git checkout “codigo de version” “nombre del archivo”: volver a la última versión de la que se ha hecho *commit*.
* git reset: vuelve al pasado sin posibilidad de volver al futuro, se debe usar con especificaciones.
* git reset --soft: vuelve a la versión en el repositorio, pero guarda los cambios en staging. Así, podemos aplicar actualizaciones a un nuevo *commit*.
* git reset --hard: todo vuelve a su versión anterior
* git reset HEAD: saca los cambios de staging, pero no los borra. Es lo opuesto a git add.
* git rm: elimina los archivos, pero no su historial. Si queremos recuperar algo, solo hay que regresar. se utiliza así:  
  git rm --cached elimina los archivos en staging pero los mantiene en el disco duro.  
  git rm --force elimina los archivos de git y del disco duro.

**Comandos para revisión y comparación en GitHub**

* git status: estado de archivos en el repositorio.
* git log: historia entera del archivo.
* git log --stat: cambios específicos en el archivo a partir de un commit.
* git show: cambios históricos y específicos hechos en un archivo.
* git diff “codigo de version 1” “codigo de version 2”: comparar cambios entre versiones.
* git diff: comparar directorio con *staging*.

Aporte creado por: Pedro Alejandro Silva.