## GitHub y Git

GitHub es en la red. Git es en local.

### Estructura GitHub/Git

WorkingDirectory → Index → Local Repository → Repositorio Remoto

### Comandos

**git init** → crea un repositorio, es decir crea un fichero .git que es donde se guarda toda la información relativa al repositorio.

Antes de subir archivos o manejar los commit, etc… se necesita ejecutar estos dos comandos:

$ git config --global user.name "Douglas Warner Jurado Peña"

$ git config --global user.email "douglaswarner.jp@gmail.com"

Para que la plataforma git sepa quien eres.

**git add [archivos]** → manda los archivos al stage.

**git commit -m ‘ ‘** → guarda los procesos incluyendo un mensaje.

**git commit -a -m ‘ ‘** → hace el add y el commit a la vez.

**git status** → te muestra los archivos que están enlazados a git.

**git log (--oneline)** → un historial de los commit realizados, cada commit tiene una clave asignada para identificarlos cómo únicos.

**git log --Graph --all --oneline** → muestra el historial de repositorios en forma de grafo, con sus ramas.

## Remoto

**git remote add + (alias Remoto) + dirección https del repositorio GitHub** → vincular el repositorio remoto local con el de GitHub. Es decir preparar los archivos para hacer el push al repositorio remoto GitHub

PD: la dirección https se encuentra en el repositorio en GitHub.

**git remote (-v)** → muestra todos los remotos vinculados que se han hecho del repositorio en el que estoy.

A partir de este momento, es cuando ya me puedo comunicarme con mi repositorio remoto.

Es decir que hasta que no haya dado de alta mi repositorio local, no podré comunicarme.

**git remote remove + (alias remoto)** → quita la conexión con el remoto GitHub.

**git push + (alias Remoto) + (rama,** por ejemplo master**)**  → pasa los archivos de local a la red, es decir al remoto dado de alta con git remote add.

**git checkout (código commit)** → podemos volver atrás eliminando todos los cambios hechos. Con el podemos volver a cualquier rama. Para volver al actual, podemos utilizar el comando git checkout master.

**git reset --(soft|mixed|hard) + (código commit)** → igual que el checkout pero elimina los commit.

**--soft** → sólo borra el repositorio local. No toca nuestro código. Mantiene el working directory.

**--mixed** → igual que --soft pero borra también el stage index.

**--hard** → borra todo lo que hay en el commit. Es el más peligroso.

## Ramas

**git branch** → te muestra todas la ramas que hay.

**git branch (nombre rama)** → crea una rama.

**git branch -a** → también muestra la rama del repositorio github. También te muestra la rama oculta.

**git branch -av** → lo mismo que -a pero también te muestra en el commit que este actualmente.

**git branch -d** → elimina la rama, y las modificaciones hechas en esas ramas desaparecen si no se fusionan.

**git merge + (nombre rama)** → fusiona las ramas, con las modificaciones hechas en esa rama. Es decir trae todos los cambios a la rama en la que estoy donde utilizó el comando.

**git checkout -b + (nombre rama)** → te crea la rama y te mueve a ella.

**Fast-Forward** → Sólo hace la fusión.

**Manual Merge** → Hay que reparar los conflictos.

## WorkFlows

En caso de querer crear una organización de un repositorio en github, y así poder trabajar un mismo grupo de usuarios sobre el mismo repositorio, ya sea por motivos de empresa, etc… Se debe de crear una nueva organización, luego importamos el repositorio a la organización.

Ahora debemos invitar a los usuarios que vayan a pertenecer al proyecto, y mediante un email ellos aceptaran y ya estarán dentro del proyecto.

En caso de no pertenecer a un proyecto, podemos traer una copia del repositorio de un proyecto público de github, a nuestra cuenta de github y trabajar con ella, sin modificar el original. Con ello podemos mandar nuestras modificaciones al original, y luego el autor del proyecto puede aceptar esas modificaciones o no.

## Issues

Prácticamente es reportar un bug o cosas pendientes para hacer.