CLASE 3

**Commit**

El comando **git commit** captura una instantánea de los cambios preparados en ese momento del proyecto. Las instantáneas confirmadas pueden considerarse como versiones "seguras" de un proyecto: **Git** no las cambiará nunca a no ser que se lo pidas. El comando **COMMIT** define el final de una transacción ejecutada con éxito. Este comando asegura que todas las modificaciones efectuadas durante la transacción se vuelvan parte permanente de la base. También libera los recursos utilizados por la transacción.

**Push**

Git push se suele usar para publicar y cargar cambios locales a un repositorio central. Después de modificar el repositorio local, se ejecuta un comando push para compartir las modificaciones con miembros remotos del equipo. Git push es uno de los muchos componentes que se usan en el proceso general de sincronización de Git. Los comandos de sincronización funcionan en ramas remotas que se configuran mediante el comando git remote. Git push se puede considerar un comando de carga, mientras que git fetch y git pull se conciben como comandos de descarga.

**Pull**

El comando git pull ejecuta en primer lugar git fetch, que descarga el contenido del repositorio remoto especificado. Después, se ejecuta git merge para fusionar las referencias y los encabezados del contenido remoto en una nueva confirmación de fusión local. Supongamos que tenemos un repositorio con una rama maestra y un origen remoto**.** En este escenario, git pull descargará todos los cambios desde el punto de separación de la rama local y la rama maestra, git pull extraerá las confirmaciones remotas divergentes, el proceso de incorporación de cambios creará una nueva confirmación de fusión local que incluya el contenido de las nuevas confirmaciones remotas divergentes.

**Branch**

En **Git**, **es** simplemente el apuntador a la rama local en la que tú estés en ese momento, en este caso la rama master; pues el comando **git branch** solamente crea una nueva rama, pero no salta a dicha rama. Esto puedes verlo fácilmente al ejecutar el comando **git** log para que te muestre a dónde apunta cada rama. El comando git branch te permite crear, enumerar, cambiar el nombre y eliminar ramas. No te permite cambiar entre ramas o volver a unir un historial bifurcado. Por este motivo, git branch está estrechamente relacionado con los comandos git checkout y git merge.

**Fetch**

**Git fetch** lo que hace **es** obtener los metadatos del repositorio remoto u original y comprueba con tu repositorio local (sin descargar nada) y te muestra si en el repositorio original existen cambios, por decir algo seria comprobar cambios de una forma meramente informativa. git fetch baja los cambios de la rama determinada y la coloca en una rama espejo que simplemente es una clase de rama escondida en la cual tú puedes mirar los cambios de dicha rama, para posteriormente hacer merge con tu rama local. El git pull simplemente es un git fetch + git merge.

**Merge**

En **git** existen dos formas que nos permiten unir ramas, **git merge** y **git** rebase. La forma más conocida es **git merge**, la cual realiza una fusión a tres bandas entre las dos últimas instantáneas de cada rama y el ancestro común a ambas, creando un nuevo commit con los cambios mezclados. Git rebase básicamente lo que hace es recopilar uno a uno los cambios confirmados en una rama, y reaplicarlos sobre otra.