**Git init** Este comando inicializa un **nuevo** repositorio en el **repositorio local**.

Git init

**Git clone** Este comando inicializa un nuevo repositorio en el **repositorio local** **clonando** íntegramente el contenido de un**repositorio remoto** que le indiquemos mediante una **URL**.

Git clone”url”

**Comandos GIT manejo de archivos y cambios**

**Git add** Una vez que hemos realizado los cambios necesarios en nuestra área de trabajo (**working area**), para comenzar la confirmación de dichos cambios, es necesario pasar todos los archivos que queramos confirmar al área de preparación (**staging area**).

Git add “nombre del archivo”

Git add . (agrega todo)

**Git commit** Una vez que tenemos archivos preparados en el área de preparación, para **confirmar** dichos archivos y crear una confirmación de cambios la sentencia utilizada es **git commit**.

Git commit -m “nombre del commit muy obvio”

**Git log** Cada vez que confirmamos cambios (Hacemos un git commit), se crea una confirmación con un **número hash identificativo**. Dicho número hash determina cada confirmación realizada y será importante para futuras operaciones.El comando para **ver todas las confirmaciones** realizadas en nuestro repositorio es **Git log**.

**Comandos GIT para manejo de repositorios remotos**

**Git remote** Para conectar al **repositorio remoto** con el cual nos comunicaremos entre nuestro**repositorio local** y dicho repositorio remoto el comando utilizado es**git remote**.

Git remote add origin “url”

Git remote -v (muestra la lista de repo)

Git remote rm “nombre/origin” (borra repo)

**Git Branch** Las **ramas** es uno de los conceptos más interesantes dentro de GIT. Permite el **trabajo en paralelo** sobre distintas partes del código base para añadir funcionalidades, corregir errores o realizar versiones de nuestros proyectos.La rama principal de un repositorio es la **rama master**, la cual sirve para, cuando se realicen todos los cambios necesarios en el área de trabajo, confirmar una nueva versión y poder sincronizarla con el repositorio remoto.Para **listar** una rama, el comando utilizado es **git branch**. Si se desea crear una nueva rama hay que especificar un nombre para dicha rama nueva.

Git Branch “nombre de la rama” (crea la rama)

**Git checkout** Para **moverse** entre las ramas creadas y realizar los cambios necesarios sobre dichas ramas, el comando utilizado es **git checkout**.Para cambiar de la rama actual a la rama que indiquemos, el comando sería:

Git checkout “nombre de la rama”

**Git status** Podemos comprobar en cualquier momento el **estado de la rama** en la que nos encontramos para comprobar si existen archivos (o directorios) que tienen cambios que deben de ser confirmados o rechazados.

Git status

**Git merge** Una vez que se han realizado los cambios necesarios en la rama creada, se debe de **combinar** con la rama “superior” para que se unan los cambios entre ambas ramas y la rama superior “apunte” a la confirmación más actualizada del proyecto.Existen dos comandos en Git para realizar estas **combinaciones**, **git merge y git rebase**.Si se desea **conservar todos los cambios e historiales de la rama combinada**, el comando a utilizar sería **git merge** que realiza una **fusión** entre las ramas seleccionadas.

Git merge “nombre de la rama”

**Git push** Una vez que ya hemos realizado todos los cambios necesarios, hemos combinado todas nuestras ramas (o las deseadas) y tenemos dichos cambios confirmados en la rama maestra (master) se puede llevar a cabo la **sincronización** del contenido de nuestro **repositorio local** con el **repositorio remoto**.Para enviar todos los cambios que tenemos en nuestro repositorio local al repositorio remoto, el comando a utilizar sería **git push**.

Git push origin main / git push “nombre de la rama a la cual quiero subir”

**Git pull** Para **extraer todos los cambios del repositorio remoto en nuestro repositorio local** y poder tener así la última confirmación de dicho repositorio, el comando a utilizar es**git pull**. (Actualiza)

Git pull “nombre del repo o rama”

**Configurar usuario** podemos hacer un global o simplemente configura usuario y mail cada vez que nos pida la terminal.

Git config --global user.name “nombre del usuario”

Git config –global user.mail “mail del usuario de github”

Sino sin el global se configura una vez para ser utilizado.