La mejor documentación que podemos tener es el libro oficial:

https://git-scm.com/book/en/v2

No obstante, este documento pretende listar los comandos principales para trabajar con git, conectarse con un repositorio remoto (github o gitlab), subir y/o descargar cambios, crear ramas y mezclar el trabajo realizado en diferentes ramas.

Listar configuración de git y configurar nombre de usuario y correo electrónico:

git config --list

git config --global user.name "Iván Gómez Conde"

git config --global user.email xxxxxxx@gmail.com

Buscar ayuda sobre un comando:

git config --help

Editar archivos:

touch index.html (crea archive)

nano index.html (opción 1 para editar)

vim index.html (opción 2 para editar)

Iniciar el repositorio dentro de una carpeta:

git init

Ver el estado de los archivos

git status

Si hay archivos que no me interesan (por ejemplo: trabajo con Symfony y todos los archivos propios de Symfony no me interesan; creo un archivo .gitignore y pongo en cada línea el directorio o archivo que no quiero que git le haga seguimiento)

touch .gitignore

Ahora añado los que me interesan para que queden listos para hacer commit:

git add * (para meter todos)

git add nombreArchivo (para meter uno)

Hacer un commit de nuestra primera versión:

git commit -m "version inicial"

Modificar un arhivo y volver a ver el estado y hacer commit

Edición del archivo

git status

git add archivo

git commit -m “segunda versión”

Eliminar un commit y borrar los cambios:

git reset --hard HEAD~1

Eliminar un commit sin borrar los cambios:

git reset HEAD~1

(De igual forma, si en lugar de poner al final 1, ponemos un 2 se borrará el penúltimo commit y desde ese todos los más recientes)

Para mirar el historial de commits:

git log

git log -2 (para ver los dos últimos commits)

Para añadir algún pequeño cambio que quería incluir en el último commit:

git add [archivoModificado]

git commit –amend

Clonado de github o gitlab

Vamos a nuestra cuenta de github o gitlab y creamos uno vacío (si lo que queremos es pasar un repositorio de git que tenemos en local a uno virtual)

Si queremos descargar uno, primero clonamos a un directorio nuestro local (nos creará una carpeta dentro de ese directorio con el nombre del proyecto de github o gitlab con todo el contenido del proyecto)

git clone https://gitlab.com/ivangconde/prueba.git

git remote (sirve para ver todos los repositorios virtuales que tenemos registrados, como mínimo aparecerá origin que es el nombre que git le da al servidor del que has clonado)

git remote -v (muestra las URLs que Git ha asociado al nombre y que serán usadas al leer y escribir en ese remoto)

git remote show origin (todos los enlaces entre local y origin)

```
ivan@DESKTOP-1V9TG1V MINGW64 ~/Documents/pruebaIvan/prueba (master)
$ git remote show origin
* remote origin
  Fetch URL: https://gitlab.com/ivangconde/prueba.git
  Push  URL: https://gitlab.com/ivangconde/prueba.git
  HEAD branch: master
  Remote branches:
    ivan   tracked
    master tracked
    pepe   tracked
  Local branches configured for 'git pull':
    ivan   merges with remote ivan
    master merges with remote master
    pepe   merges with remote pepe
  Local refs configured for 'git push':
    ivan   pushes to ivan   (up to date)
    master pushes to master (up to date)
    pepe   pushes to pepe   (up to date)
```

Creación de una nueva rama

git branch ivan

git checkout ivan (cambiar a la rama ivan)

Si hago algunos commits en la rama y esa rama no están en el repositorio virtual (origin) no puedo hacer un simple git push (aunque yo en local esté colocado en la rama ivan), debo hacer:

git push --set-upstream origin ivan

Ahora está en seguimiento en gitlab tanto la rama master como ivan.

Si quiero mezclar lo que haya hecho en la rama ivan con lo que había en master hacemos:

git checkout master

git merge ivan

Si tenemos conflictos, iremos a los archivos correspondientes, escogeremos con lo que nos queremos quedar (si la parte de master o la parte de ivan) y después:

git add [archivos]

git commit “versión mezclada”

Si lo queremos subir al repositorio virtual podemos hacer ahora:

git push

Si otro compañero estaba trabajando sobre una versión antigua del repositorio. Dicho compañero ***siempre trabajará en una rama independiente*** por ejemplo, pepe:

git branch pepe

git checkout pepe

Hace varios commits sobre Pepe y quiere hacer merge.

Antes de hacer merge deberá asegurarse de tener la última versión de master. Para ello hacemos:

git pull (que equivale a un git fetch + git merge)

Hacemos un merge con la rama pepe después y subimos al repositorio virtual (git push).

También podía haber subido la rama pepe a github o gitlab sin hacer merge.

git push --set-upstream origin pepe

Histórico de comandos en linux ($history)