SISTEMAS DE GESTIÓN DE VERSIONES. GIT

git init

```
Inicializar un repositorio: crea un directorio dentro del cual crea un directorio que empieza de un .git en el que se almacenan todos los commits que hacemos; toda la información que, en definitiva relativa al repositorio.
```

git add

```
Le decimos al programa git que a partir de ahora, nos gestione el archivo que queremos.
```

git commit

```
Toma una fotografia de la situacion actual del repositorio y guardala.

git commit -a -m "Inicializo el repositorio"

Nos saldrá un error: crear usuario y email

Cuando hagamos git commit -m, la primera linea del archivo es muy importante, porque es la que salga el histórico.

Se puede modificar SOLO el último commit. Esto se hace con_ git commit -a --amend
```

git status

```
Estado de la copia del repositorio
```

En /home, en el archivo .gitconfig tenemos la información personal de git: usuario y mail. Se puede modificar si se requiere.

git log

```
Nos da la historia de todos los commits que hemos hecho
```

git show

```
Me dice información concreta de un commit en concreto (pasarle el
identificador)
```

git clone

```
Copiar todo el repositorio con toda su historia para traernoslo y trabajar
con él.
```

.gitignore

```
Evita subir al repositorio los archivos que especifiquemos en dicho
archivo. De esta forma, si modificaramos un archivo con extensión recogida
en el archivo .gitignore, cuando hagamos git status, no nos aparecerá como
commit pendiente.
```

git push

```
Cuando hacemos los commit, se hacen solo a nivel local. Si tengo que
hacerlo para que se guarde remotamente, tengo que hacer push
```

Crear llave publico-privada

```
ssh-keygen

cd .ssh

ls
more id_rsa.pub

copiar llave ssh en github
```

git pull

```
Para bajarnos los contenidos del repositorio; si algun colaborador ha
añadido cambios al repositorio, para que no haya interferencias,
```

git branch

```
Para crear una rama:

    git branch nombre_rama

Para ver las ramas:

    git branch
```

git checkout

```
Para cambiar de rama:

    git checkout rama_a_la_que_quiero_ir

También se puede crear una rama con:

    git checkout -b nombre_rama
```

git push -o origin nombre_rama

```
Para subir una rama (no es normal subir una rama a main; si hacemos solo
git push, nos dará una advertencia, y no se hará el push) hacemos

    git push -o origin nombre_rama

IMPORTANTE VERIFICAR SI HEMOS SUBIDO LA RAMA (ver en github si aparece
nuestra rama).
```

git merge

```
Para mezclar lo que he hecho en una rama con la rama principal del
proyecto, tengo que ir a la rama principal y traerme desde ahí la rama que
quiero:

git checkout main
git status (para comprobar que estamos en rama principal)
git pull (para bajar los cambios que haya habido)
git merge nombre_rama_que_quiero_mezclar

    Si hay conflicto en el merge, hay que resolverlo. ¿Cómo? Abro el
archivo, veo lo que hay, hablo con el tio que ha hecho los cambios y que me
confirme los cambios para que yo los aplique y modifico el archivo. Miro el
status y resuelvo el conflicto haciendo git add y ahora sí, una vez
resueltos los conflictos, hacemos commit y push.
```

git log --decorate --graph --oneline

```
Para mostrar los log de forma bonica.
```

git tag nombre_tag identificador_commit

```
Sirve para etiquetar un commit en concreto.
```