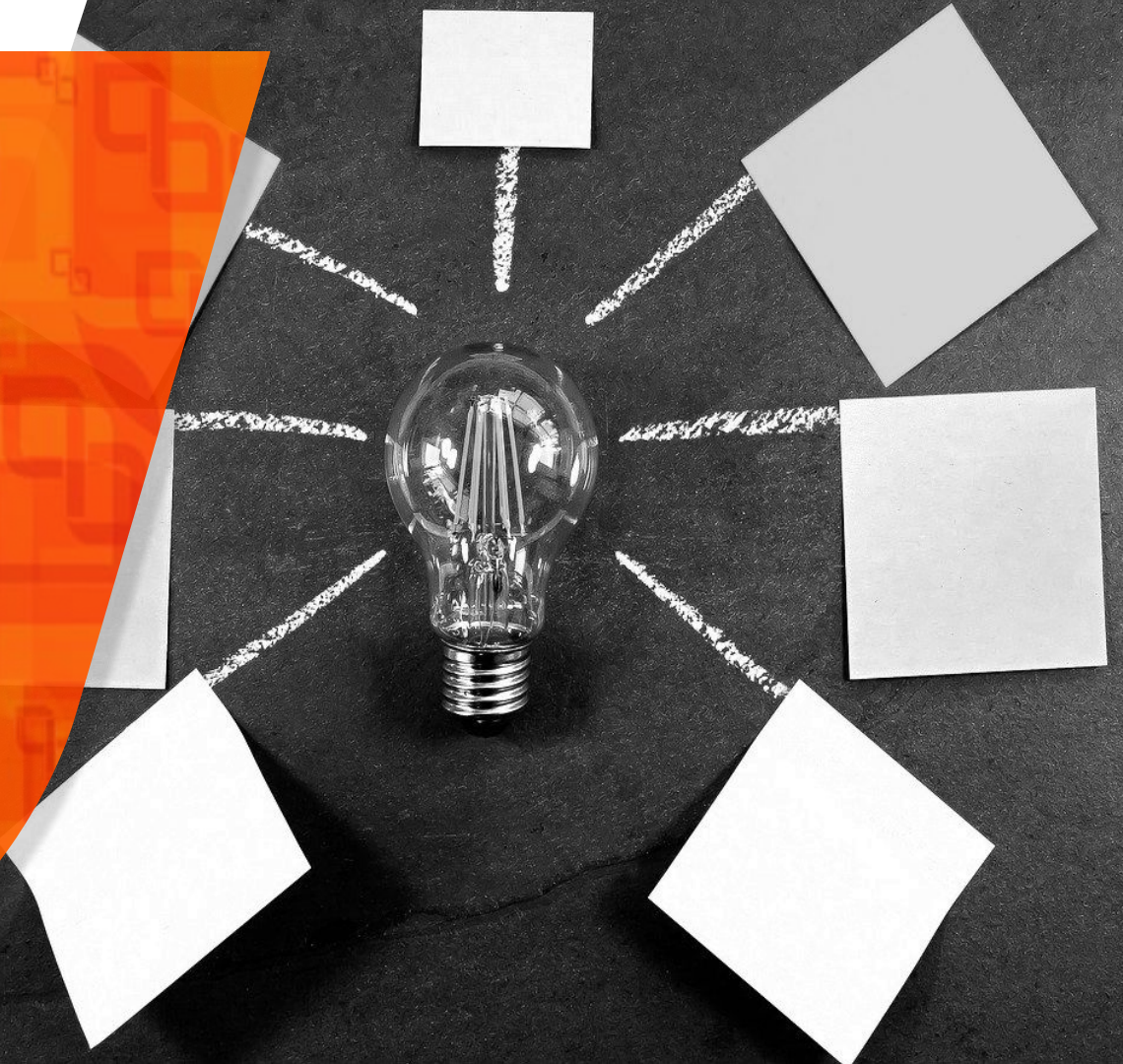


Git



Git es **sistema de control de versiones** (VCS) también conocidos como repositorios de código.



- Un sistema que almacena los cambios en el código evitando que se pierdan
- Permite tener trazabilidad de los cambios y versionarlos
- Permite desarrollar de manera colaborativa
- Permite conocer quién hizo cada cambio y cuando
- Permite deshacer los cambios y volver a un estado previo

- Git es un repositorio distribuido
- Permite trabajar con multitud de ramas y facilita el trabajo colaborativo
- Altamente eficiente
- No es necesaria una conexión al servidor central de forma constante

## CENTRALIZED

Central Repository



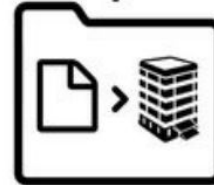
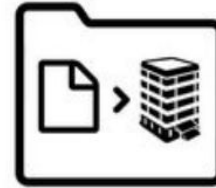
Working Copy



Working Copy

## DISTRIBUTED

Full Repository

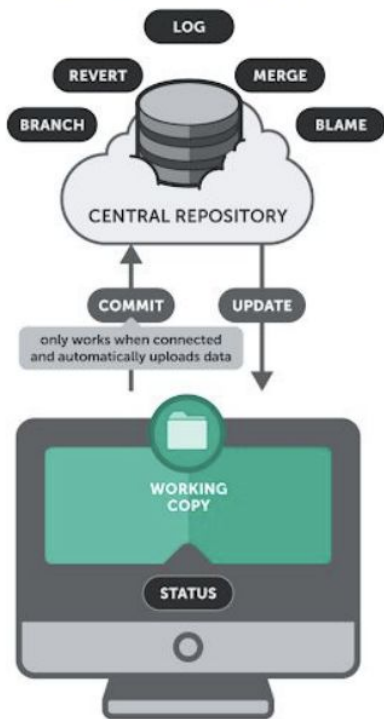


Full Repository



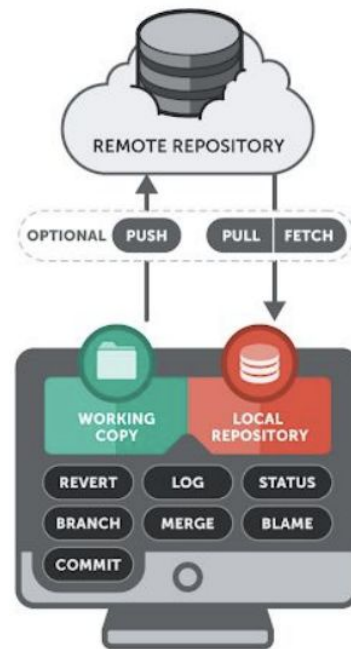
Full Repository

### SUBVERSION

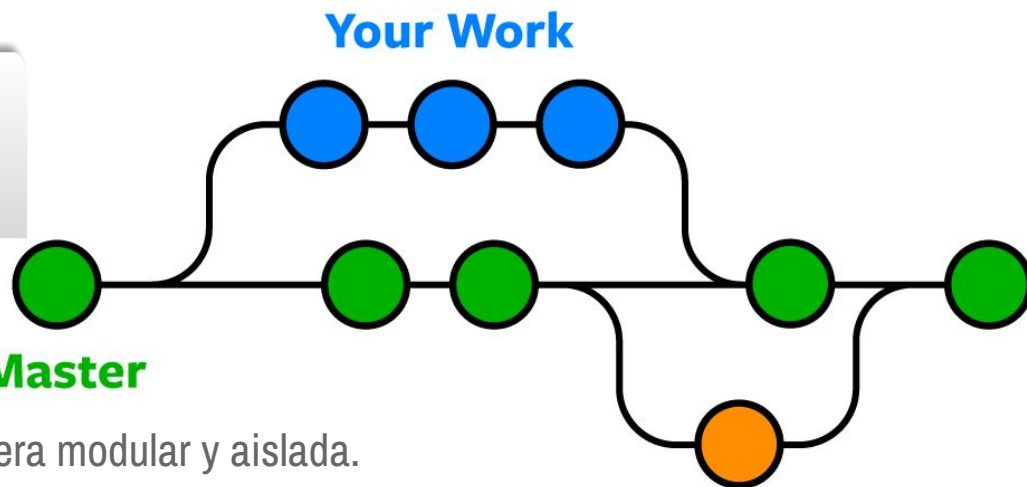


- En git no se hace "checkout" desde el repo central, se clona y se traen (pull) los cambios.
- El repo local es una copia exacta de todo lo que hay en el repo remoto.
- Repositorio local:
  - check in/out
  - commit
  - Mantiene una versión del histórico
- Cuando los cambios están listos se suben (push) desde el server local al remoto.

### GIT



El uso de ramas (**branches**) es la característica más popular de Git.



- La rama por defecto de Git es **Master**
- Diseñado para trabajar de manera modular y aislada.
- Ligero y rápido para crear y mezclar cambios de otras ramas. **Someone Else's Work**
- Todo commit en git está alojado en una rama.
- Toda rama local o remota es un simple puntero a un commit concreto. El mismo concepto aplica a los TAGS y al HEAD.

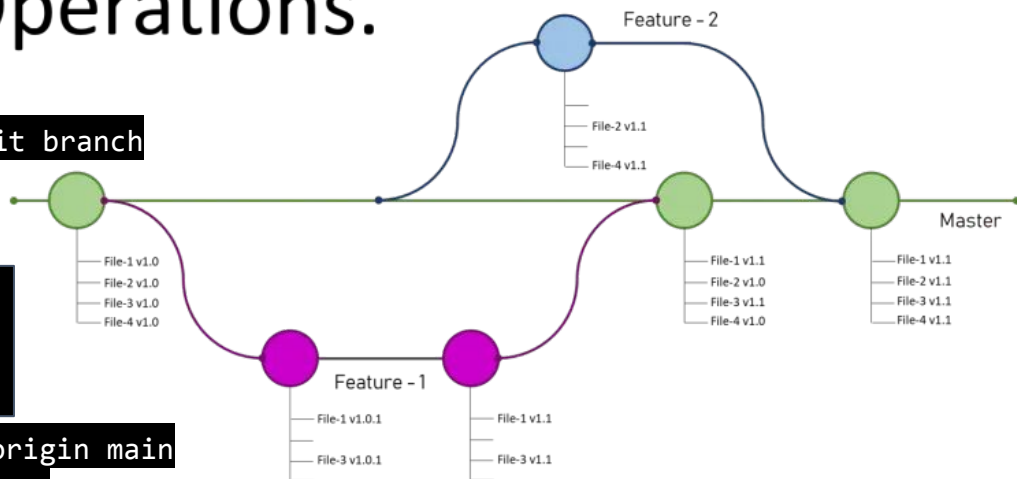


# GIT Branch and its Operations.

- Inicialización del repositorio: `git init`
- Clonar repositorio central: `git clone`
- Conocer ramas disponibles o crear ramas nuevas: `git branch`
- Navegar por ramas: `git checkout`
- Aplicación de cambios y confirmación:

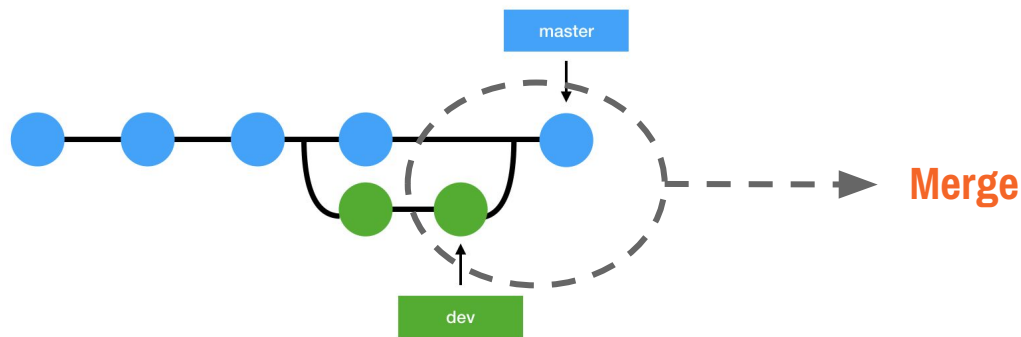
```
git status # View the state of the repo
git add <some-file> # Stage a file
git commit # Commit a file</some-file>
```

- Envío de cambios al repositorio remoto: `git push origin main`
- Traer cambios del repositorio remoto al local: `git pull`



Un **merge** es la «fusión» de los cambios de una rama con los de la otra.  
El resultado puede llegar a crear un nuevo commit junto a los de las otras dos ramas.

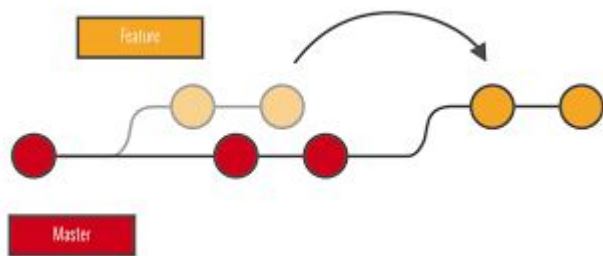
- Combina varias secuencias de commit en un historial unificado.
- La rama actual tendrá la capacidad de avanzar hacia el resultado.
- No permite la fusión hasta que se lleve a cabo el proceso de commit





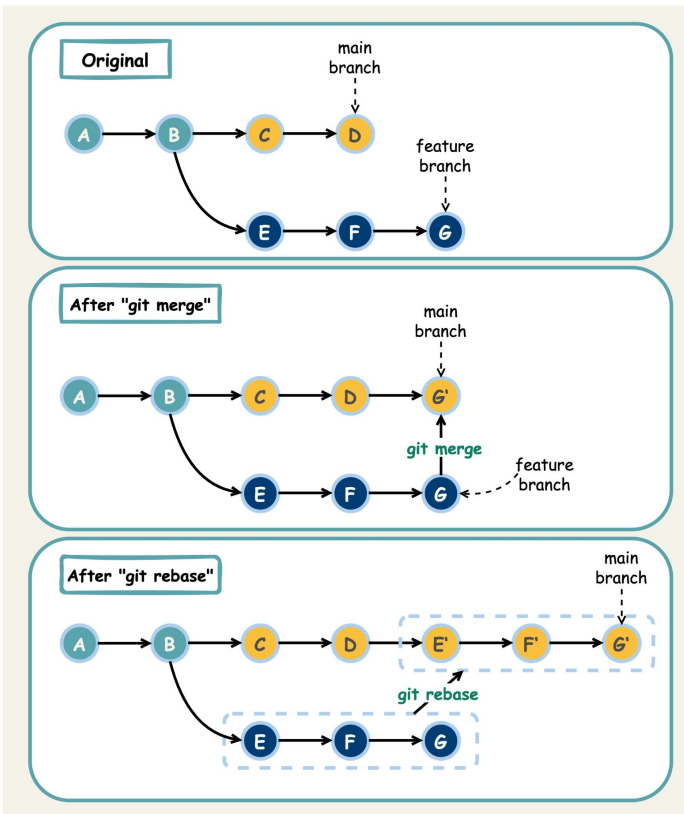
El comando **git rebase** permite mover la totalidad de una rama a otro punto del árbol

- Reordenar o editar commits
- Contribuye a tener un historial de proyecto más limpio
- No guarda la historia de la rama secundaria, sino que se encarga de reescribir la historia de la rama principal



## Git Merge vs. Git Rebase

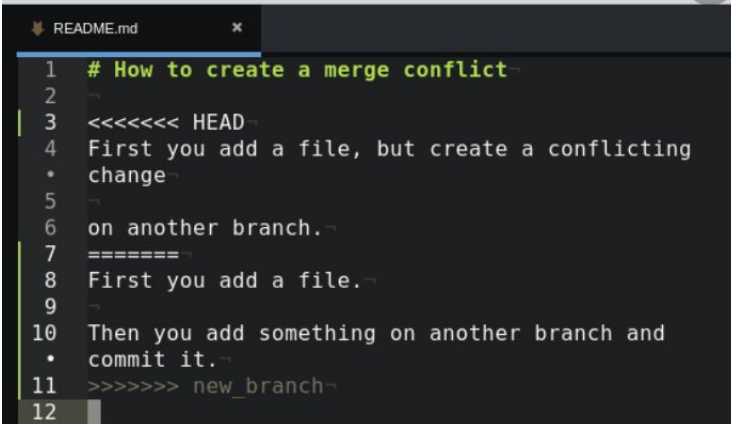
[blog.bytebytego.com](https://blog.bytebytego.com)



Cuando se intentan mezclar ramas de git, pueden producirse **conflictos**.

- Los conflictos ocurren cuando se han realizado cambios diferentes sobre la misma línea de un fichero.
- O cuando una persona edita un fichero y otra lo elimina.

Git no sabe determinar cuáles de los cambios son con los que debe quedarse. El fichero que presente conflictos contendrá unas secciones determinadas con <<< y >>> para indicar dónde están.



```
1 # How to create a merge conflict
2
3 <<<<<< HEAD
4 First you add a file, but create a conflicting
5 • change
6 on another branch.
7 =====
8 First you add a file.
9
10 Then you add something on another branch and
11 • commit it.
12 >>>>>> new_branch
13
```

- Restablecer el estado actual a un estado específico (a nivel de commit o de rama). Reescribe el historial de commits: `git reset`
- Deshacer los cambios realizados por un commit anterior creando un commit completamente nuevo. No altera el historial de commits: `git revert`
- Elegir un commit de una rama y llevarlo a otra: `git cherry-pick`
- Almacenar temporalmente los cambios que efectuado en el código en el que se está trabajando para trabajar en otra cosa: `git stash`
- Recuperar la última información de los metadatos del original (aunque no hace ninguna transferencia de archivos. Es más bien como comprobar si hay algún cambio disponible): `git fetch`
- Petición de incorporación de un commit a una rama: `git request-pull`

Una **clave SSH** es una credencial de acceso para el protocolo de red SSH (Secure Shell)

- SSH usa un par de claves para iniciar un protocolo de enlace seguro entre partes remotas, este par de claves se denominan clave pública y clave privada.
- La clave pública se instala en cualquier servidor y luego se desbloquea mediante la conexión con un cliente SSH que hace uso de la clave privada
- Se generan a través de un algoritmo criptográfico de clave pública. Los más comunes son RSA y DSA
- La mayoría de los proveedores de alojamiento Git ofrecen guías sobre cómo crear claves SSH.

- Generar clave ssh usando el correo electrónico como etiqueta:

```
ssh-keygen -t rsa -b 4096 -C "your_email@example.com"
```

```
ssh-keygen -t ed25519 -C "your_email@example.com"
```

- Almacenar la clave: A continuación, se pide introducir un archivo en el que se guardará la clave. Se puede especificar la ubicación del archivo o pulsar Intro y aceptar la ubicación de archivo predeterminada.

```
> Enter a file in which to save the key (/Users/you/.ssh/id_rsa): [Press enter]
```

- Generar una contraseña para la clave privada: Tras indicar la ruta en la que se almacenará la clave, lo siguiente que tendremos que hacer es indicar una contraseña:

```
Enter passphrase (empty for no passphrase):
```

- Una vez se finaliza el proceso, la clave pública y privada quedarán almacenadas con un formato similar a este:

- Clave pública: /home/demo/.ssh/id\_rsa.pub
- Clave privada: /home/demo/.ssh/id\_rsa

- Añadir la nueva clave SSH a ssh-agent (herramienta para el guardado de claves):

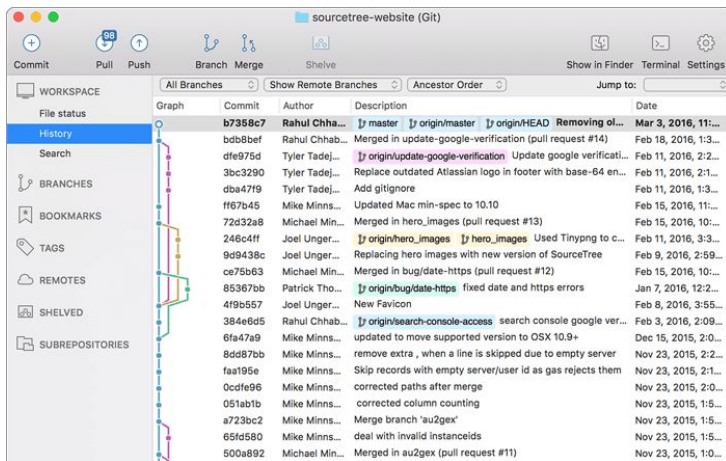
```
eval "$(ssh-agent -s)" #comprobar que se está ejecutando  
ssh-add -K /Users/you/.ssh/id_rsa
```

- Añadir la clave pública al servidor de git utilizado.

- Enlaces que os pueden ayudar en la generación de claves ssh:
  - <https://git-scm.com/book/en/v2/Git-on-the-Server-Generating-Your-SSH-Public-Key>
  - <https://www.atlassian.com/es/git/tutorials/git-ssh>
  - <https://confluence.atlassian.com/bitbucketserver/creating-ssh-keys-776639788.html>
  - <https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent>
  - <https://thenewstack.io/ssh-made-easy-with-ssh-agent-and-ssh-config/>

Existen multitud de clientes git. Los que usan la UI son unos parguelas.

CLI. Además del propio cliente git, dependiendo de la shell se pueden instalar otros plugins que ofrecen ayuda visual y para los comandos



```

$ mkdir zshhh
$ cd zshhh
~/zshhh$ git init
Initialized empty Git repository in /home/michiel/zshhh/.git/
~/zshhh$ git touch zshhh.txt
~/zshhh$ git add .
~/zshhh$ git commit -m "first commit"
[master (root-commit) 40ef851] first commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 zshhh.txt
~/zshhh$ git status
Zsh is great!
zsh: command not found: Zsh
~/zshhh$

```

Aplicaciones gráficas:

SourceTree

Kraken

Otros



**Bitbucket.** Es de Atlassian, por tanto, ofrece integración nativa con Jira, Confluence ...



GitLab

**Gitlab.** Es una solución completísima. Su versión gratuita ofrece las funcionalidades necesarias para trabajar perfectamente con el.



GitHub

**Github.** Estándar para el desarrollo de open source. Si el repositorio es abierto tienes acceso a toda su funcionalidad. Muy completo también.



- Hacerse un fork de este repo o de otro repo que prefirais:  
<https://github.com/sarapatoQualtio/demo-spring-boot>
- Hacer un clone del repo
- Crear una rama feature-demo (o el nombre que prefiráis)
- Hacer algún cambio en algún fichero (Si se eligió el proyecto anterior, añadir los cambios del Step2 del readme)
- Ver el estado local
- Commit de los cambios
- Push a la rama feature-demo
- Hacer merge a la rama main

Hacer estos ejercicios

<https://github.com/sarapatoQualtio/curso-de-git>

