Taller de GIT

Git Commands- Git Flow - Github

RONY HANCCO
GIRO PINTO

¿Qué es Git?

¿Por qué es necesario?

Git es un sistema de control de versiones distribuido.

- Git permite trabajar en paralelo y maneja un historial de cambios.
- Con Git puedes trabajar tanto con un repositorio remoto y un repositorio local.
- Git permite revertir y volver a una versión anterior.

Instalación Windosws

Link de instalación git-scm

Herramientas adicionales

- Git BASH
- Git CMD
- Git GUI

Instalación en Linux

Dependiendo de tu distribución

```
$sudo apt install git
```

```
$sudo dnf install git
```

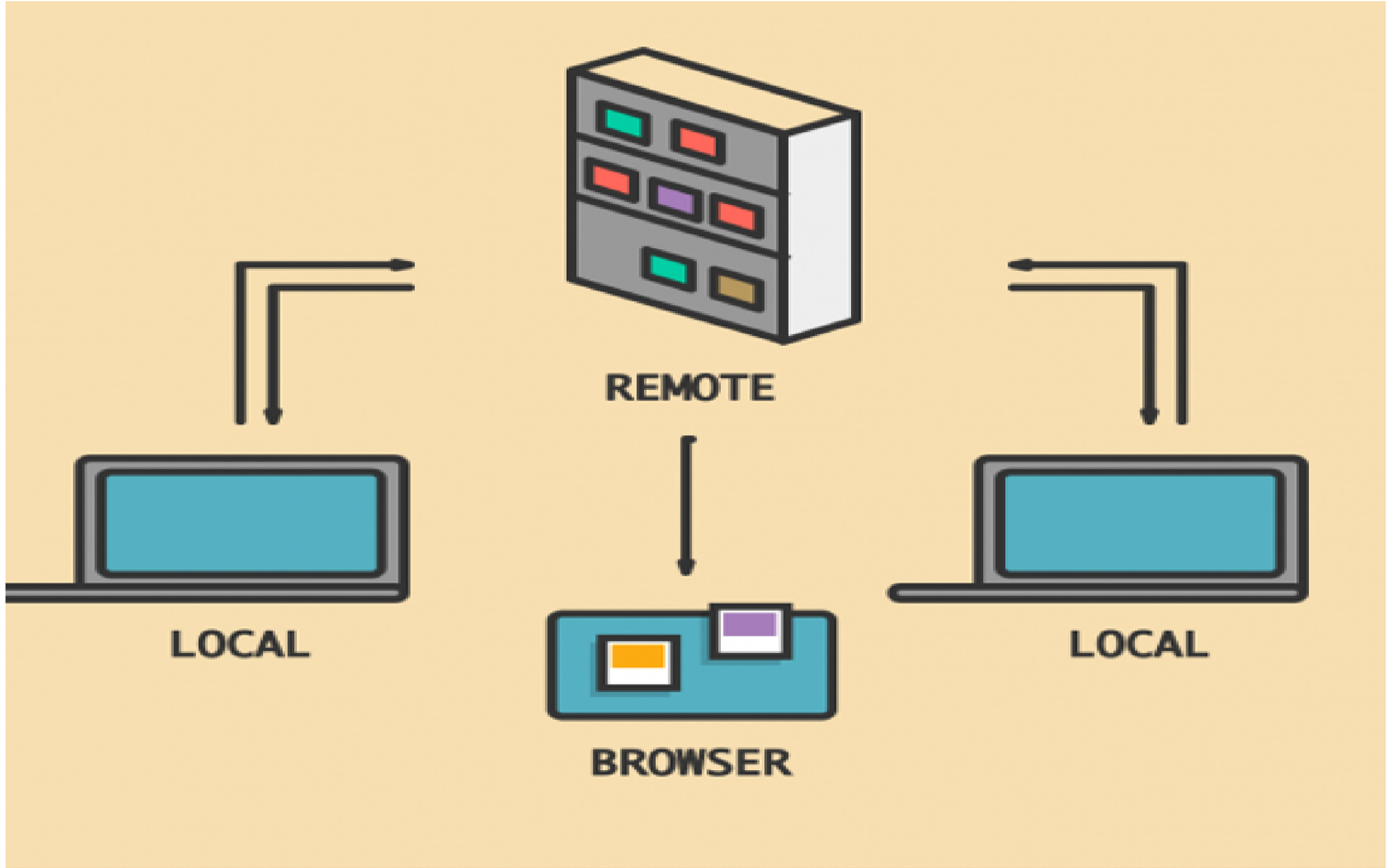
```
$sudo pacman -S install git
```

Configuración global de usuario

```
$git config --global user.name NOMBRE
```

```
$git config --global user.email CORREO
```

Repositorios locales y remotos



Comandos básicos

```
$git init
```

```
$git status
```

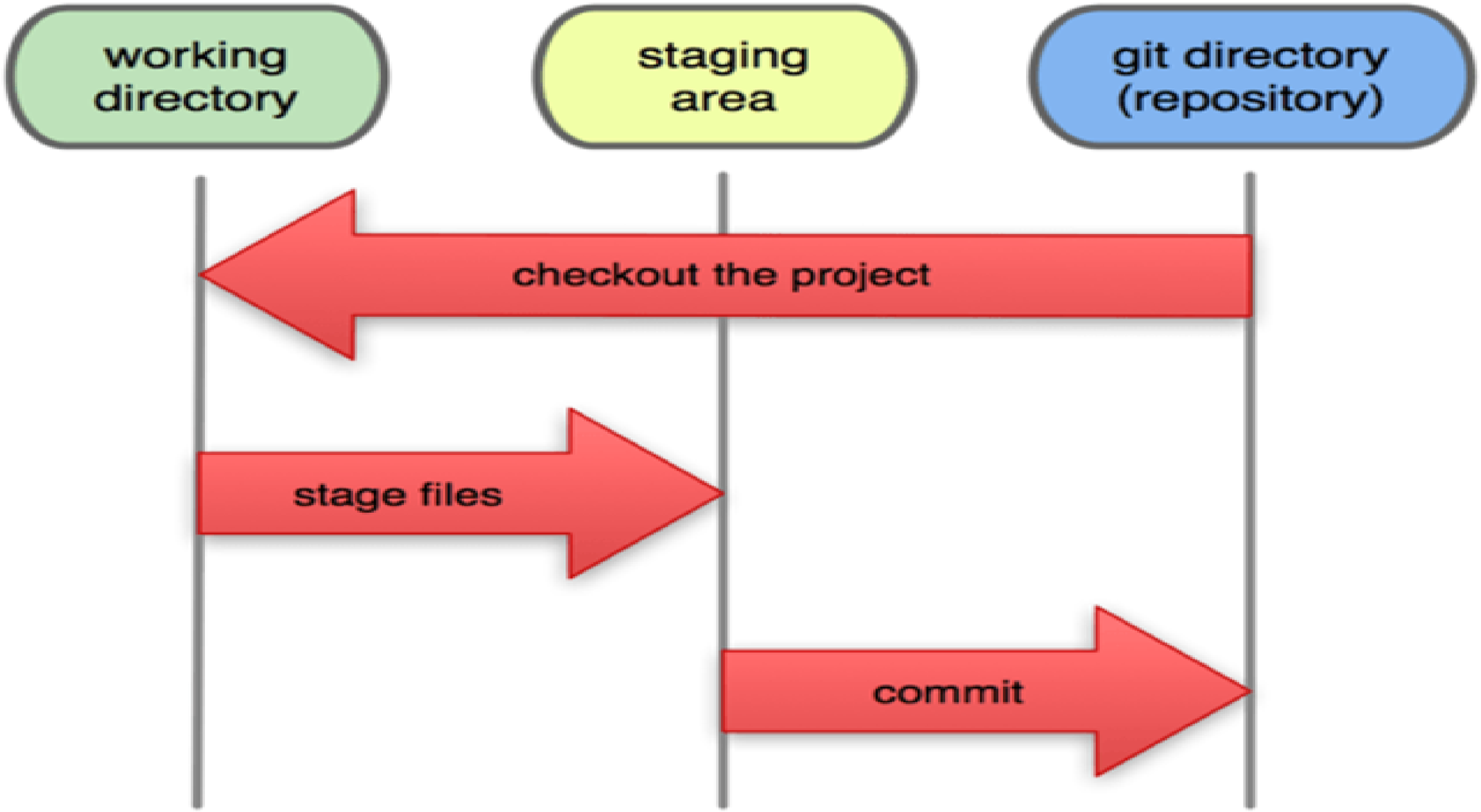
```
$git add
```

```
$git commit
```

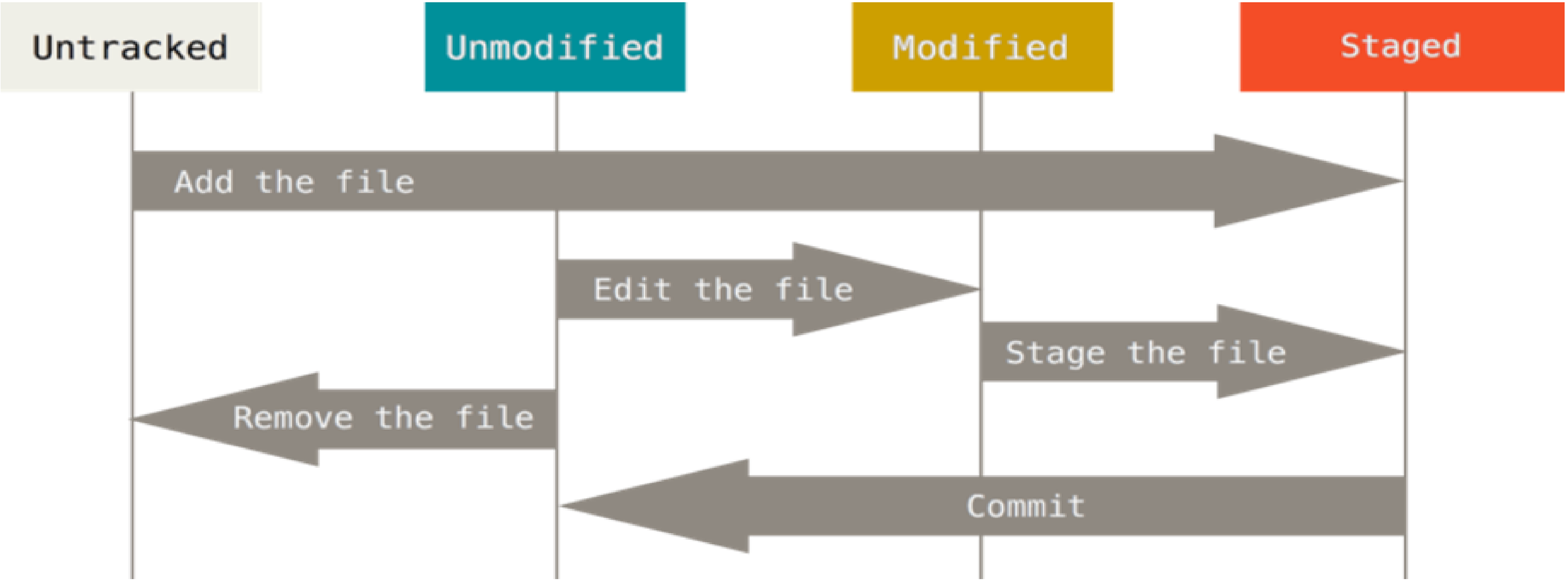
```
$git log
```

```
$git checkout
```

Repositorios locales y remotos



Seguimiento de archivos



Git version 2.23.0

```
$git status
On branch master
Changes to be committed:
 (use "git restore --stage <file>... " to unstage)
       new file:   fourth_file.txt
       deleted:    second_file.txt
       renamed:    README.md -> README

Changes not staged for commit:
 (use "git add <file>..." to update what will be committed)
 (use "git restore <file>..." to discard changes in working directory)
       modify: first_file.txt
```

Versiones anteriores

```
$git status
On branch master
Changes to be committed:
 (use "git reset HEAD <file>... " to unstage)
       new file:   fourth_file.txt
       deleted:    second_file.txt
       renamed:    README.md -> README

Changes not staged for commit:
 (use "git add <file>..." to update what will be committed)
 (use "git checkout -- <file>..." to discard changes in working directory)
       modify: first_file.txt
```

Recomendaciones

- Usar siempre que puedes el git status
- Usar las opciones que consideres necesarias para cada comando
- Usar de preferencia:

```
$git commit -m MENSAJE
```

- El mensaje del commit debe ser significativo, entendible y referirse a los cambios hechos
- Un ejemplo de un comando sobrecargado de opciones:

```
$git log --topo-order --all --graph --date=local --pretty=format:'%C(green)%h%C(reset) %><(55,trunc)%s%C(red)%d%C(rese
```

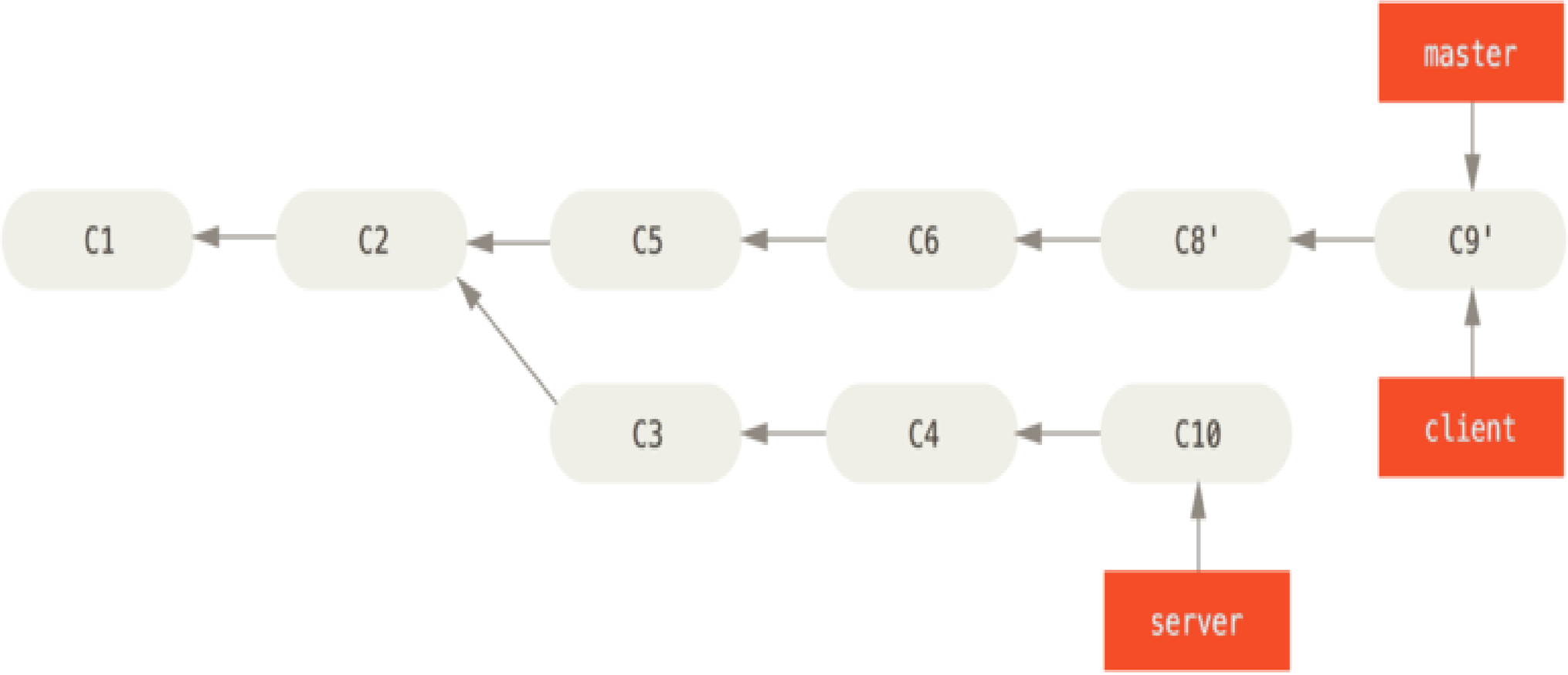
Alias

```
$git config --global alias.ALIASNAME COMANDO
```

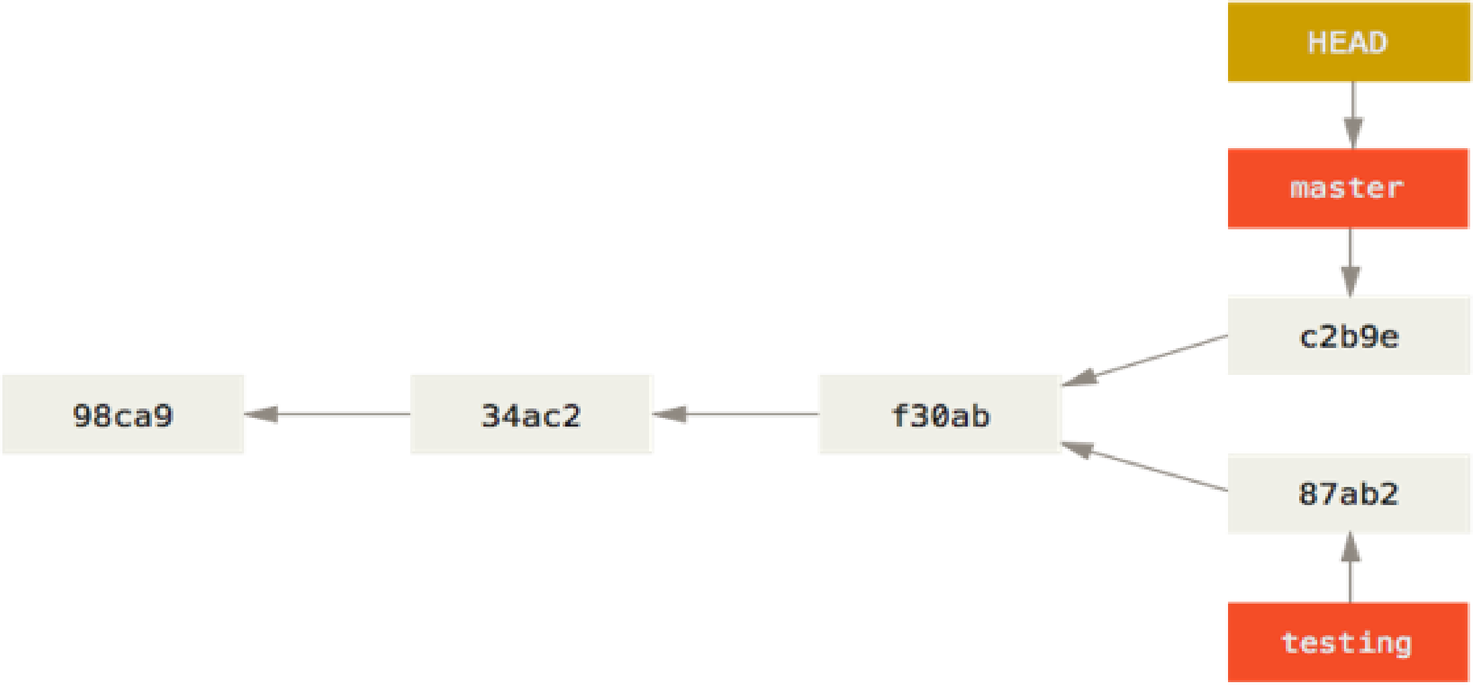
Ejemplo

```
$git config --global alias.last 'log -1 HEAD'
$git last

commit 45564654....
Author: Giro Pinto <email>
Date: Tue Aug 10 23:24:05 2020 +0800

   Current head test
```

Git Branching



Git HEAD



Comandos para realizar el branching

```
$git branch
```

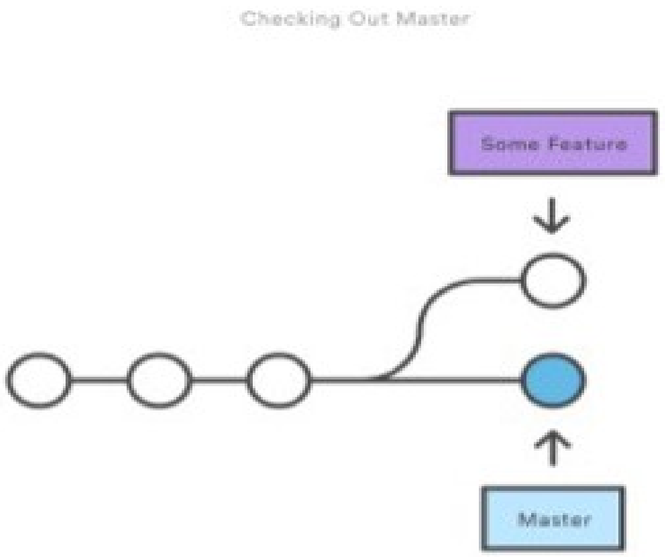
```
$git branch NombreRama
```

```
$git checkout NombreRama
```

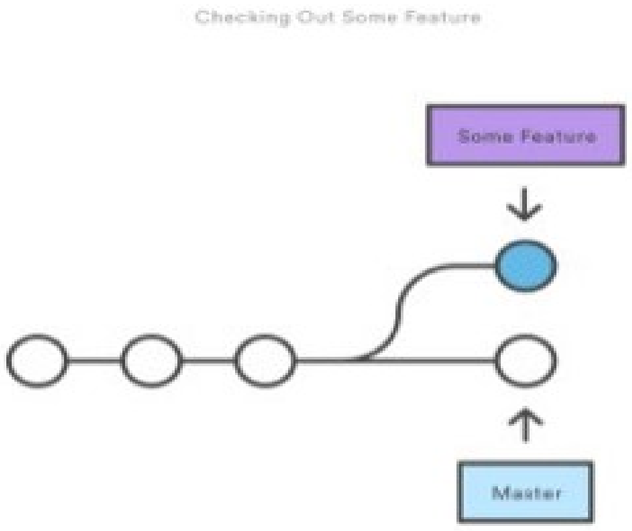
```
$git branch -d NombreRama
```

```
$git branch -m AntiguoNombre NuevoNombre
```

Git checkout en acción



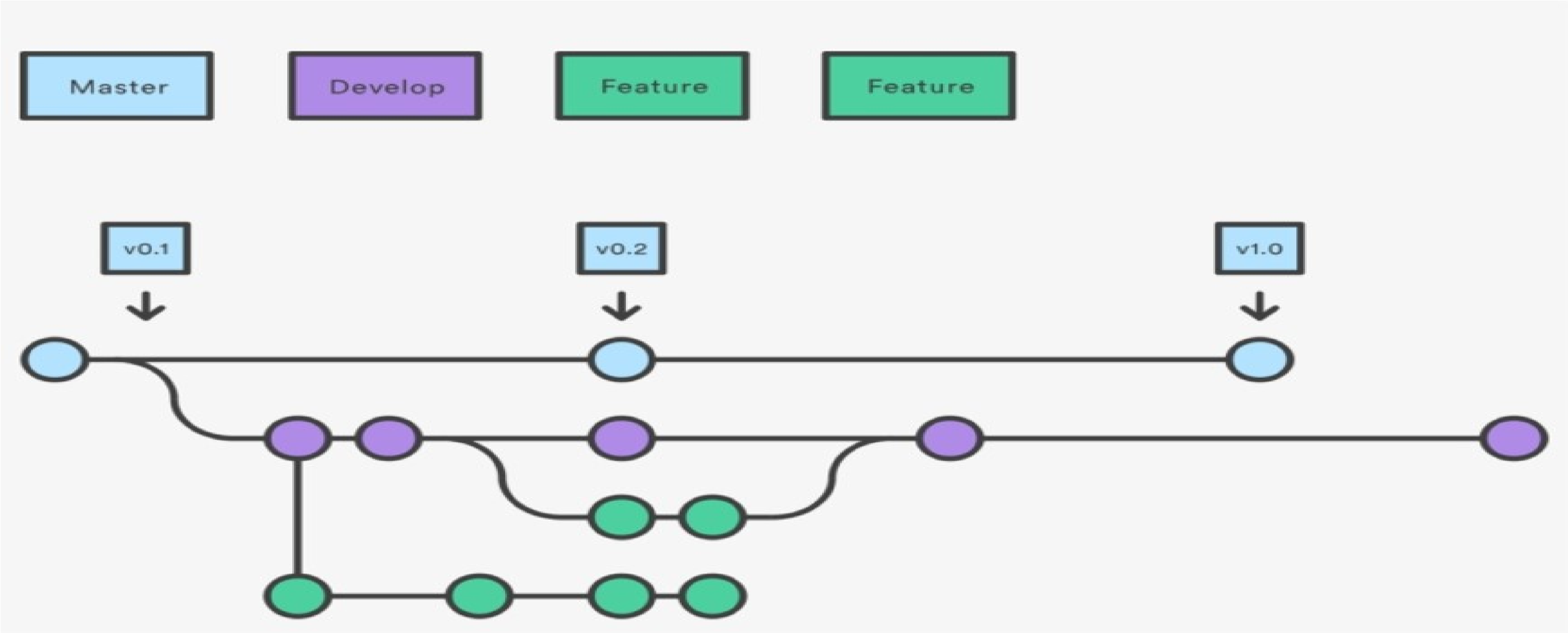
git checkout master



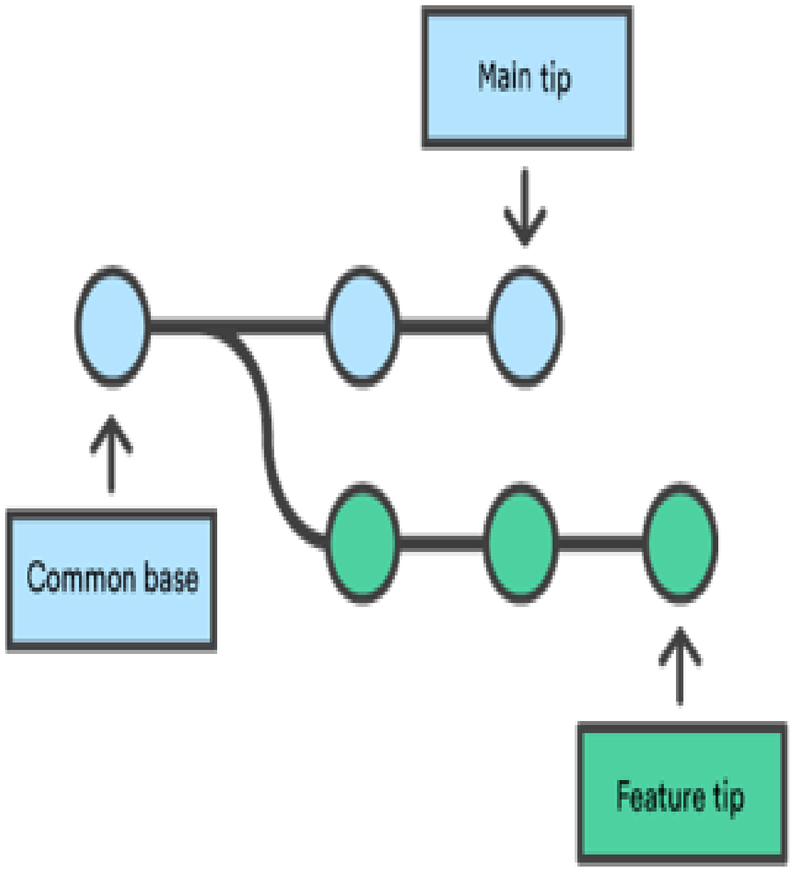
git checkout feature

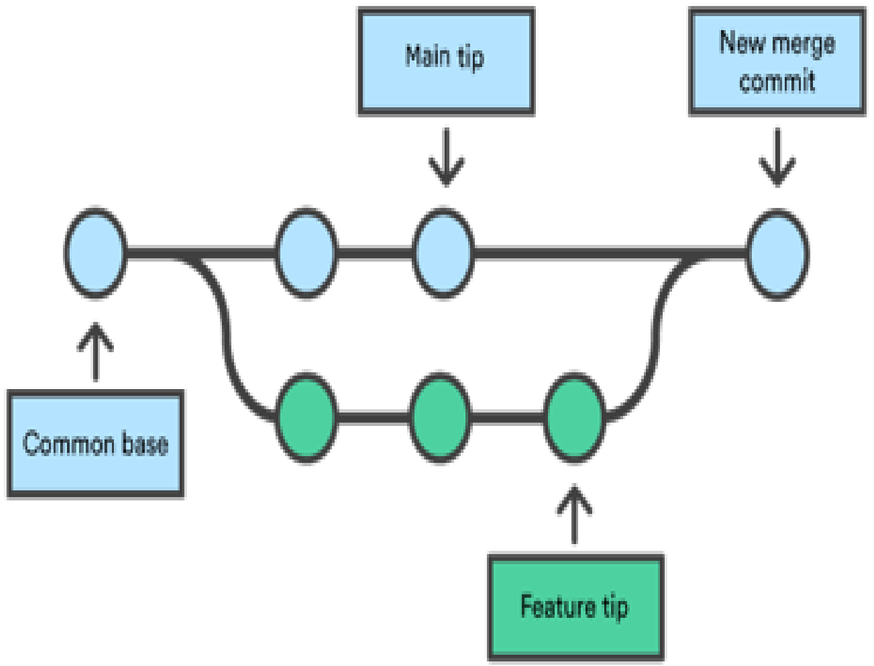
Pikicast

Git branching ejemplo



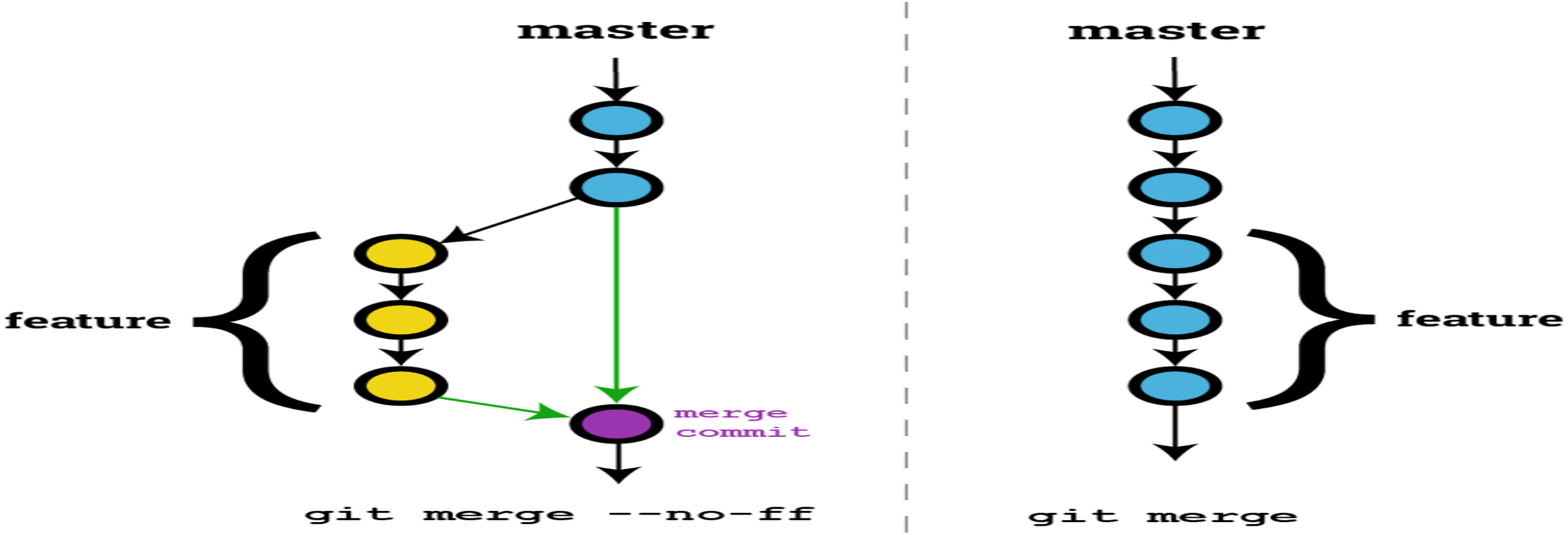
Git merge


Main tip
Common base
Feature tip


Main tip
New merge commit
Common base
Feature tip

Tipos de Merge

- Fast Forward Merge
- 3-Way-Merge



Comandos para hacer Merge

```
$git checkout master / ForMergeBranch
```

```
$git merge MergingBranch
```

```
$git branch -d MergingBranch
```

```
$git merge --abort
```

```
$git merge --no-ff
```

GitHub - GitLab - BitBucket

- GitHub Link
- GitLab Link
- BitBucket Link

Clonar un repositorio en GitHub

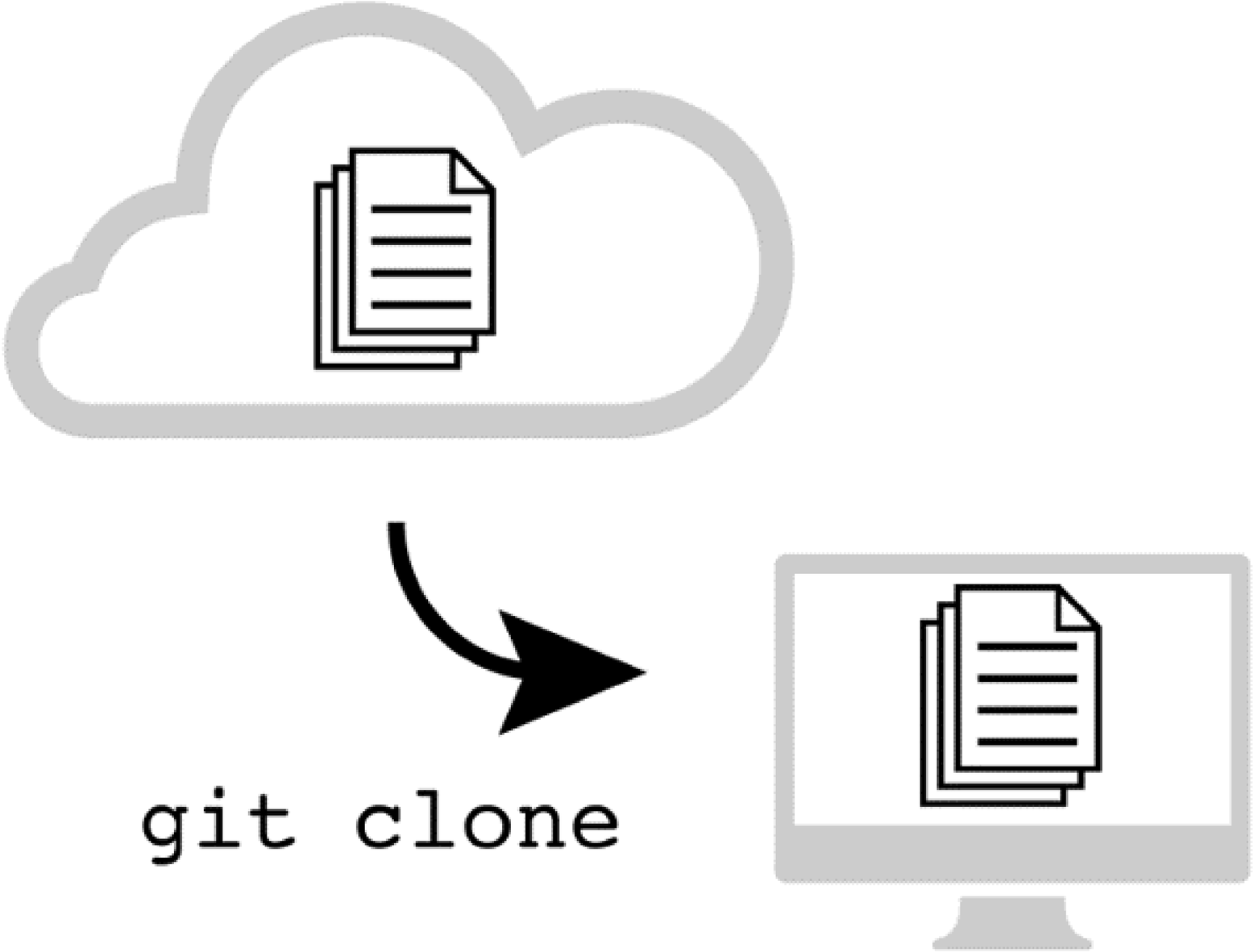
- Desde linea de comandos vía https

```
$git clone https://github.com/userName/repoName.git
```

- Desde linea de comandos vía SSH

```
$git clone https://github.com/userName/repoName.git
```

- Usando githubDesktop
- Descargar el repositorio (comprimido)
- Usando GitHubCLI


git clone

Comandos para trabajar con remotos

```
$git remote add remoteName remoteUrl
```

```
$git remote
```

```
$git remote remove remoteName
```

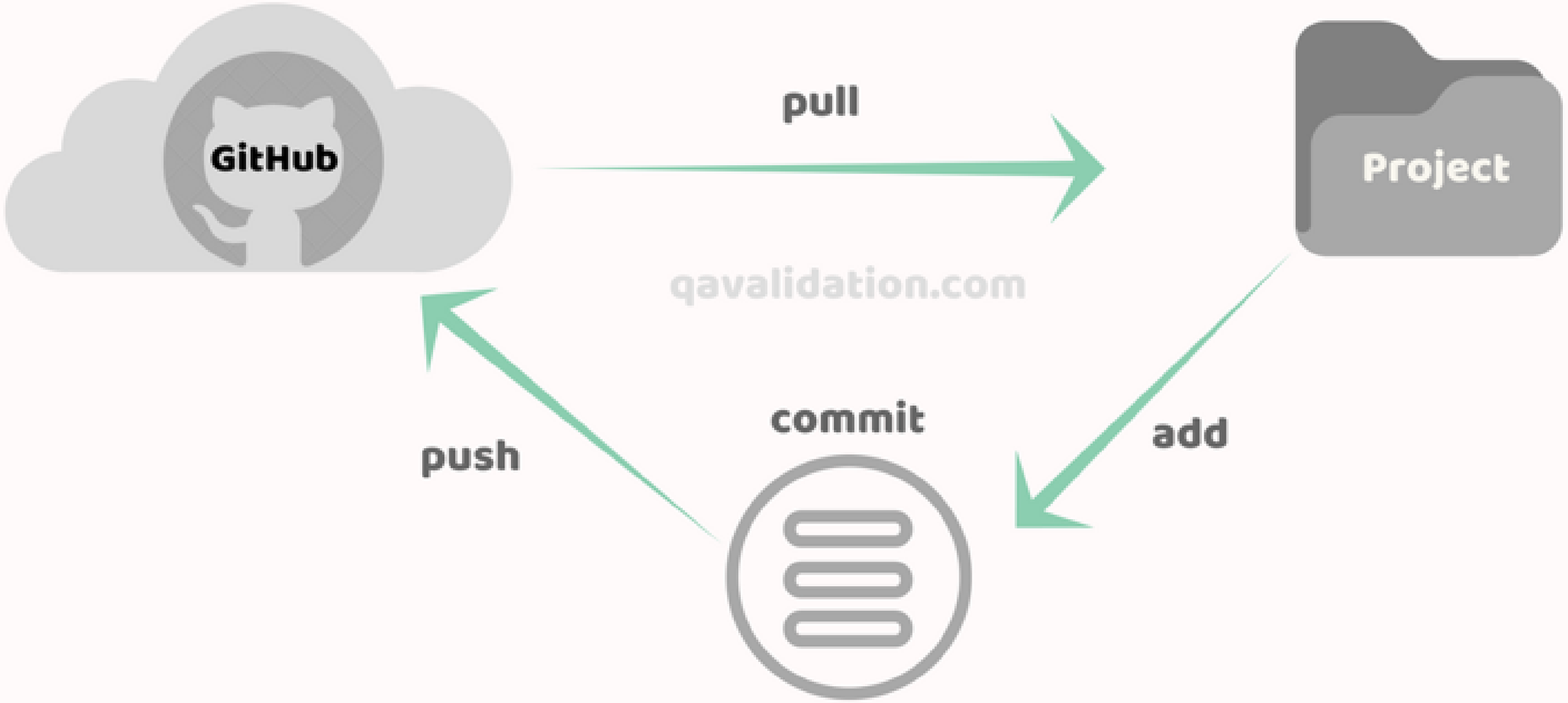
```
$git remote rename oldName newName
```

```
$git push remoteName branchName
```

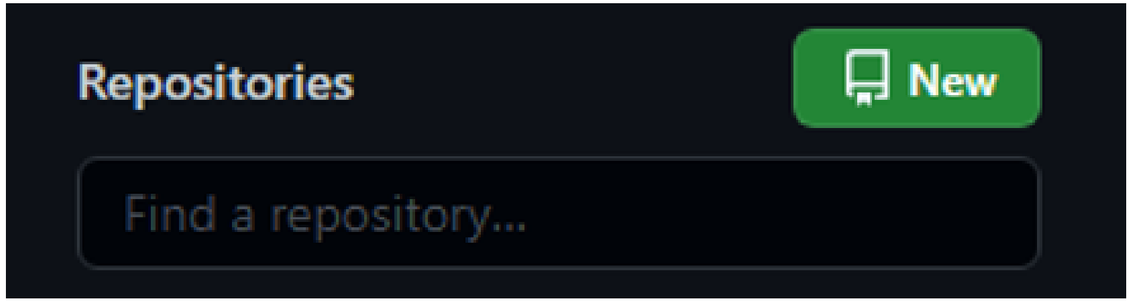
```
$git pull
```

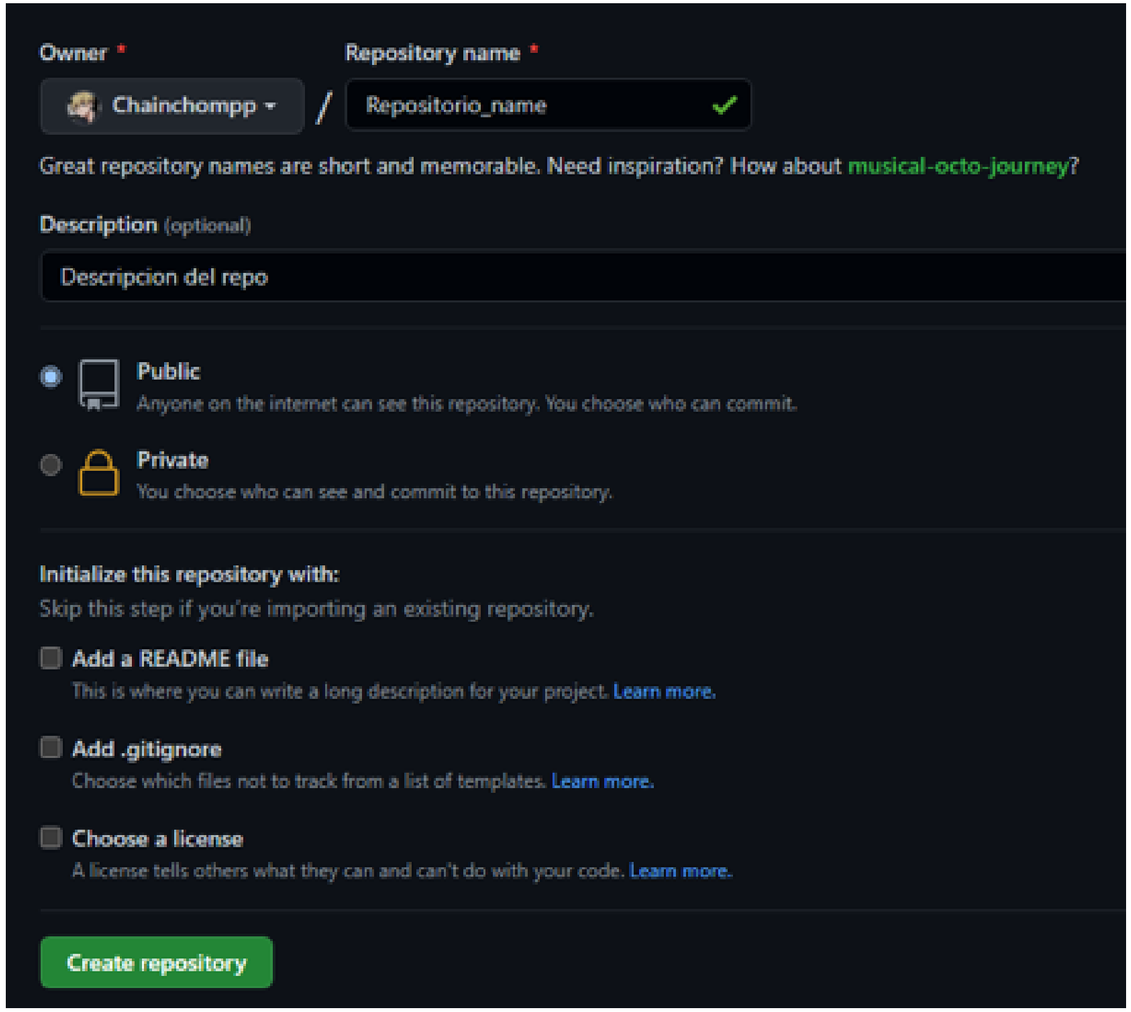
```
$git fetch
```

Git PUSH PULL



Crear repositorios en GitHub





Cambiar ramas remotas

```
$git branch -m oldName newName
```

```
$git push --set-upstream remoteName newName
```

```
$git push origin --delete oldName
```

Arreglar Commits

```
$git commit --amend
```

Deshacer Commits

```
$git reset --hard
$git reset --soft
```

Git Diff

```
$git diff
```

Ejemplo

```
$git diff
diff --git a/first_file.txt b/first_file.txt
 index 0e3da6d..fe563b0 100644
 --- a/first_file.txt
 +++ b/first_file.txt
 @@ -1,4 +1,4 @@
 Este es mi primer mensaje:
 "Hola"
 Este es mi segundo mensaje:
 -"Versionar es muy divertido"
 +"Versionar es muy muy divertido"
```

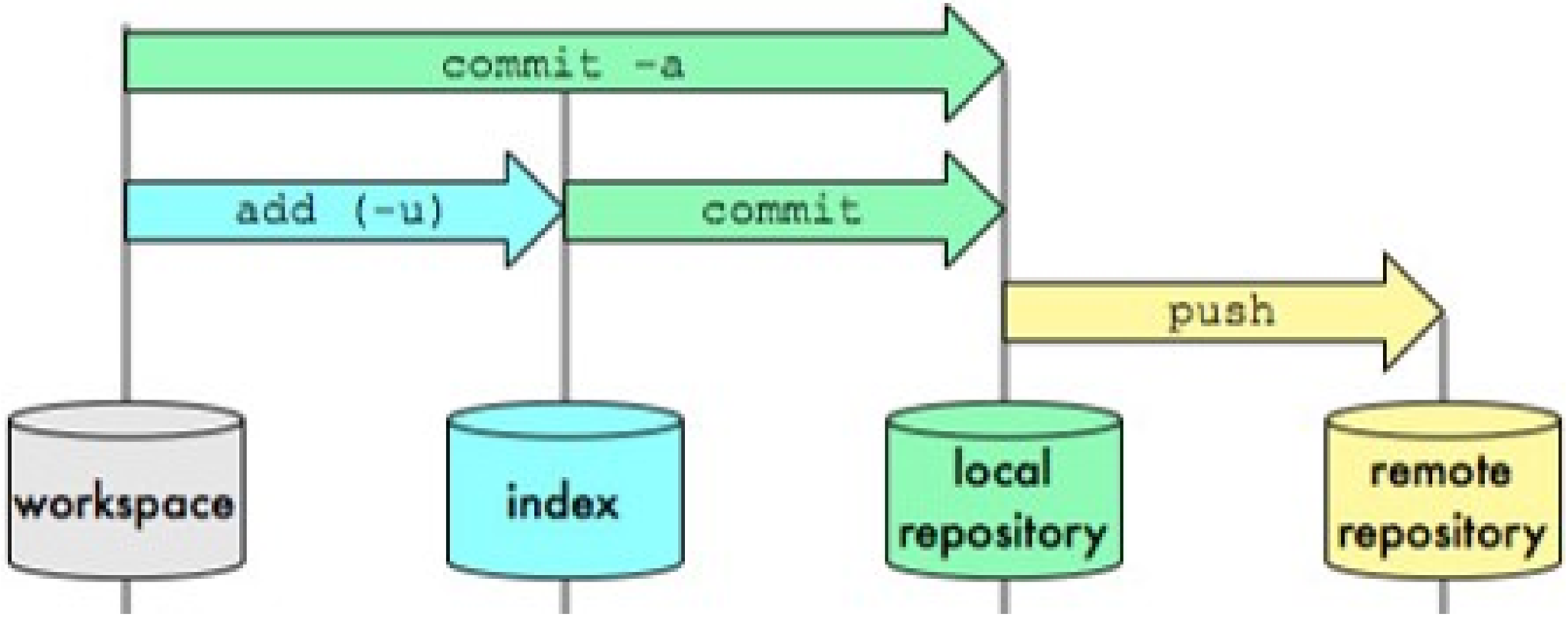
Git Merge Conflicts

colors.txt

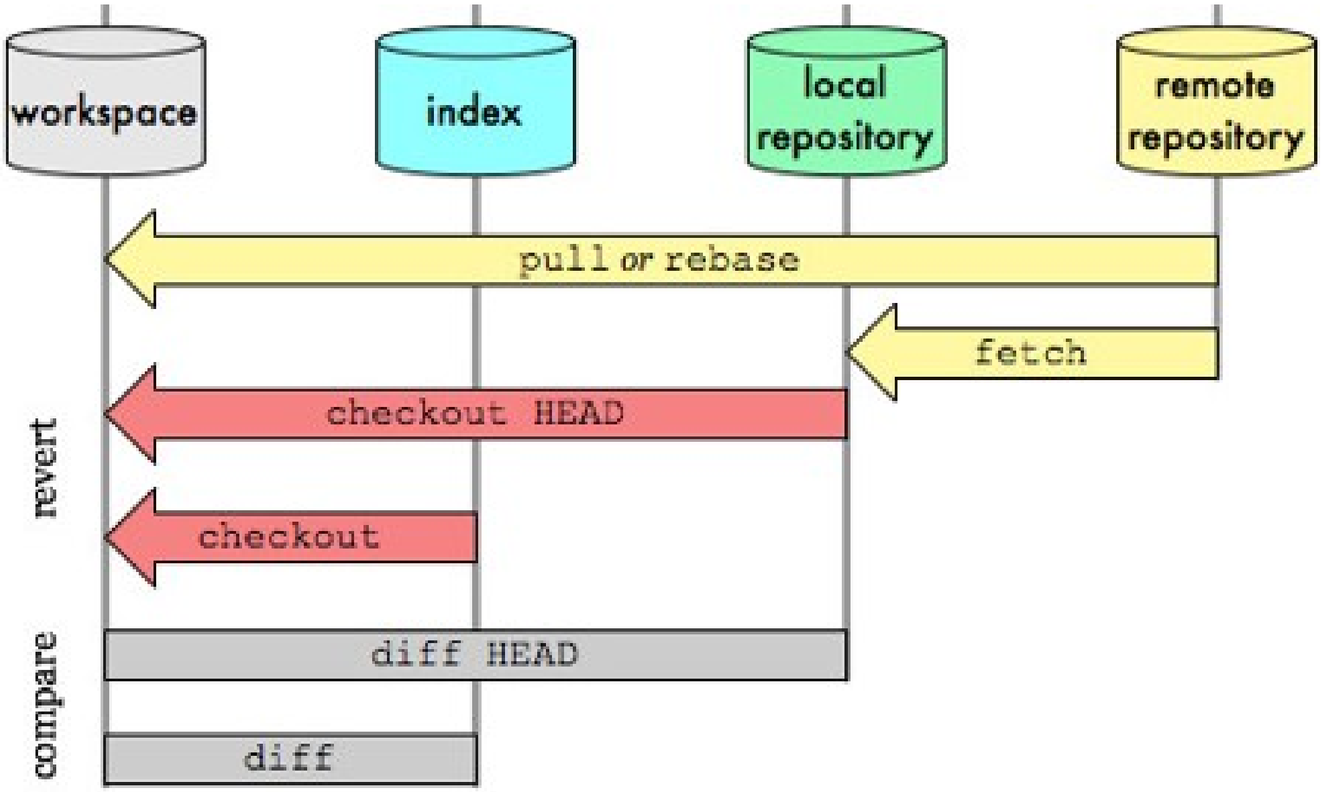
src > colors.txt

```
1  red
   Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
2  <<<<<<< HEAD (Current Change)
3  green
4  ||||||| merged common ancestors
5  yellow
6  =======
7  white
8  >>>>>>> his-branch (Incoming Change)
9  blue
```

Resumen


commit -a
add (-u)
commit
push
workspace
index
local repository
remote repository

Resumen


workspace
index
local repository
remote repository
pull or rebase
fetch
checkout HEAD
checkout
revert
diff HEAD
diff
compare

Visual Studio Code

Link de instalación