Taller de GIT

Sesión 1

git

Nixvoid

¿Qué es Git?

¿Por qué es necesario?

Git es un sistema de control de versiones distribuido.

- Git permite el trabajo en paralelo y colaborativo, con una gran cantidad de usuarios.
- Con Git puedes trabajar tanto con un repositorio remoto y un repositorio local. Siendo los host de repositorios más usados: GitHub, GitLab y BitBucket
- 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
```

Verificando la versión instalada

```
$git --version
```

Recursos en línea

- Git-SCM-docs
- Git-SCM-book
- Learing git branching
- Oh my Git!
- GitKraken
- DevHints

Recomendación: usar VS Code y la extensión GitLens.

Para la consola de comandos usaremos Git Bash, pero pueden usar la de su preferencia.

Configuración global de usuario

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

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

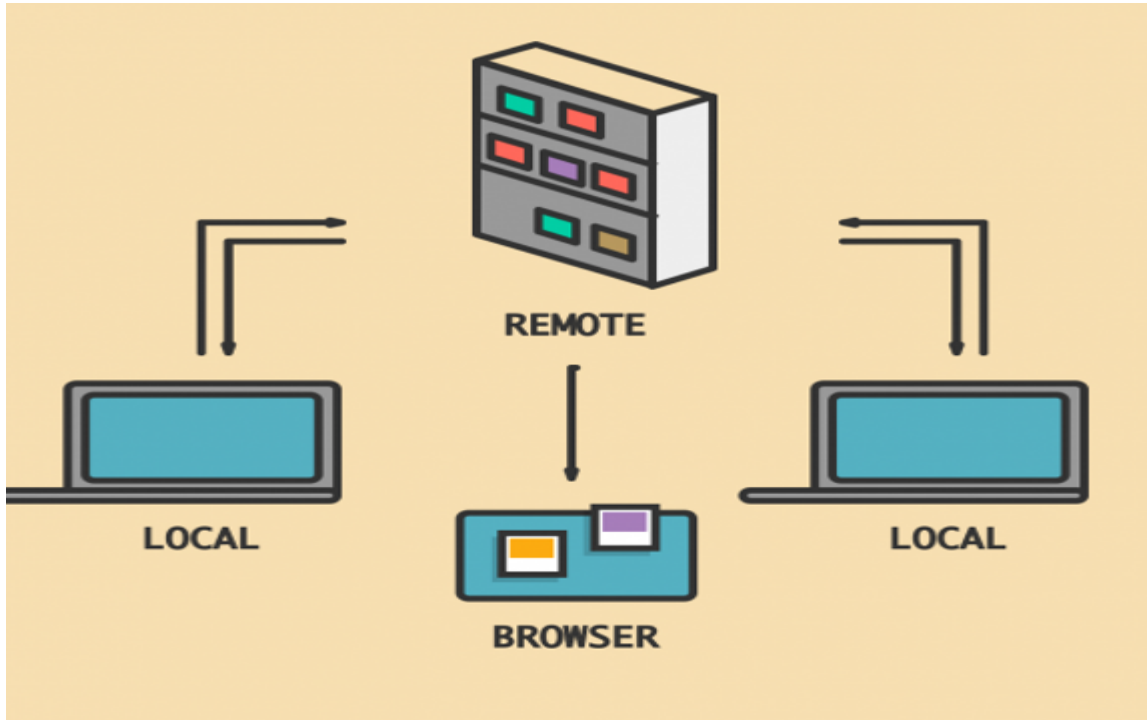
Comprobar la configuración aplicada

```
$git config --list
```

Modificar la configuración global

```
$git config --global --edit
```

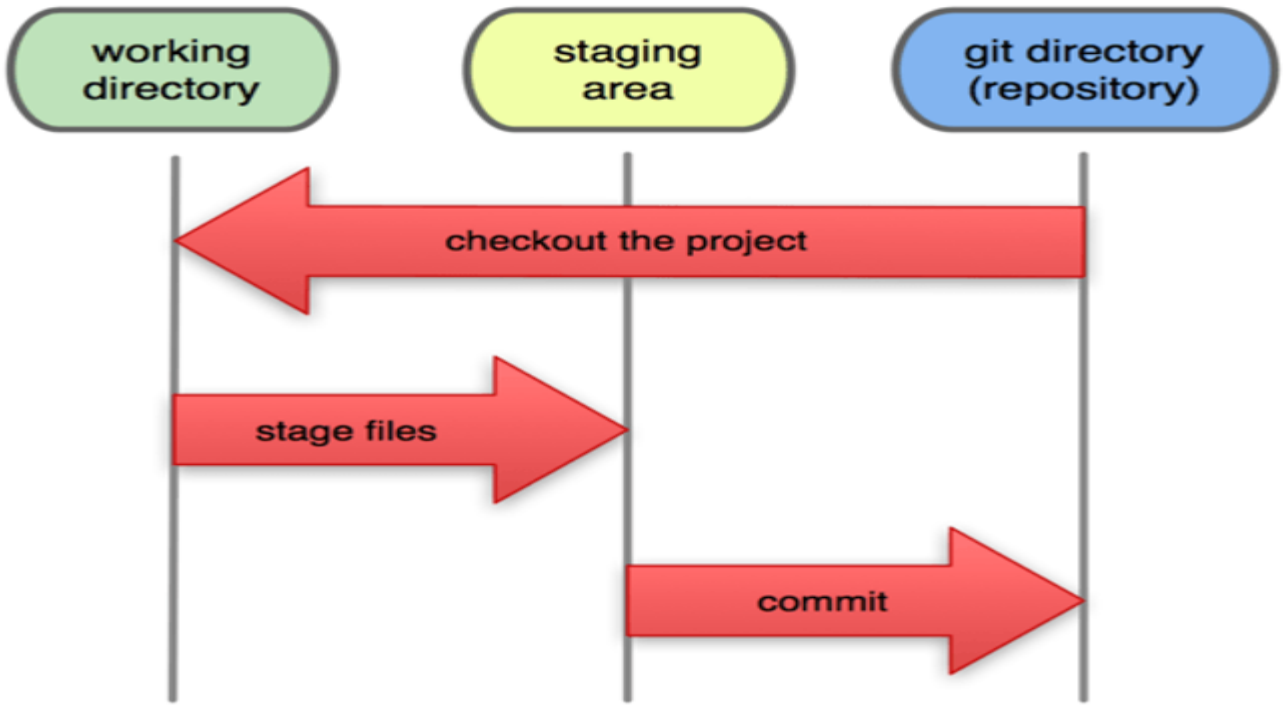
Repositorios locales y remotos



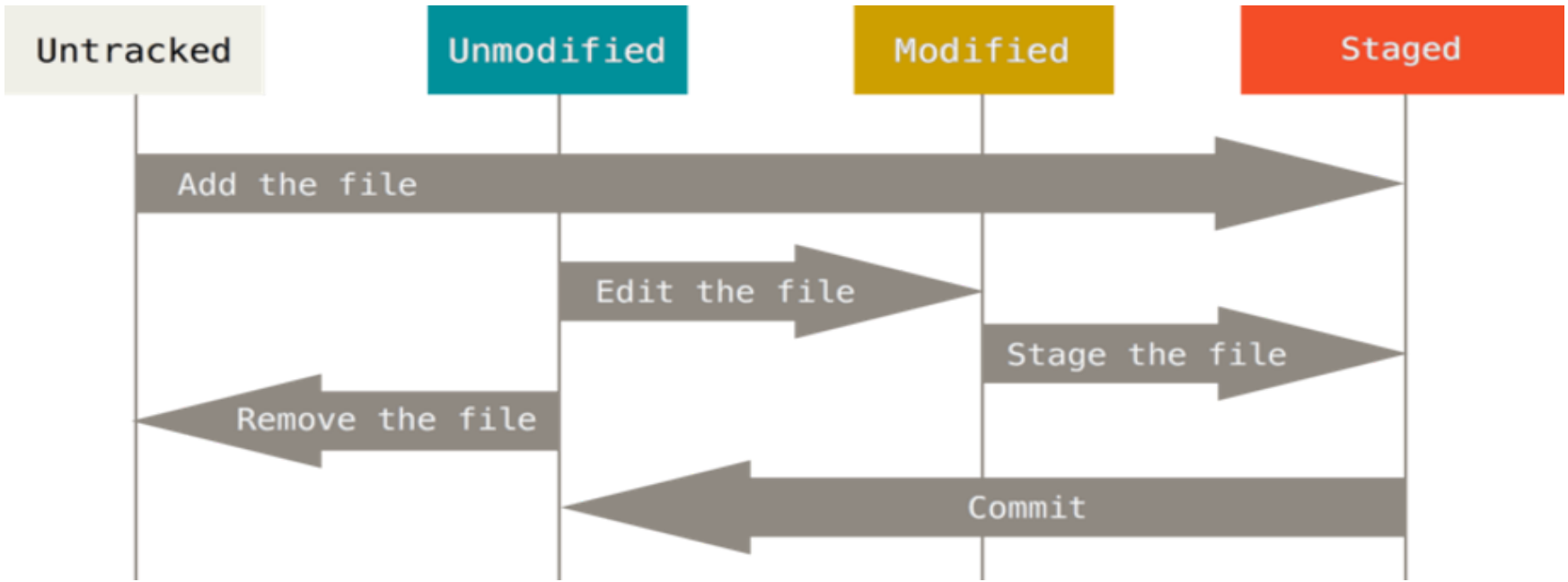
Comandos básicos

```
$git init

$git status

$git add

$git commit

$git log

$git checkout
```

Repositorios locales y remotos



Seguimiento de archivos



Restaurar cambios

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
```

Restaurar cambios

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
```

Git Log opciones

```
$git log <NombreRama>

$git log --oneline

$git log -p

$git log --author=”<NombrePatron>”

$git log --grep=”<Patron>”

$git log --graph --decorate

$git log --pretty="format:%H"

$git log -<Limite>
```

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(reset) %C(blue
```

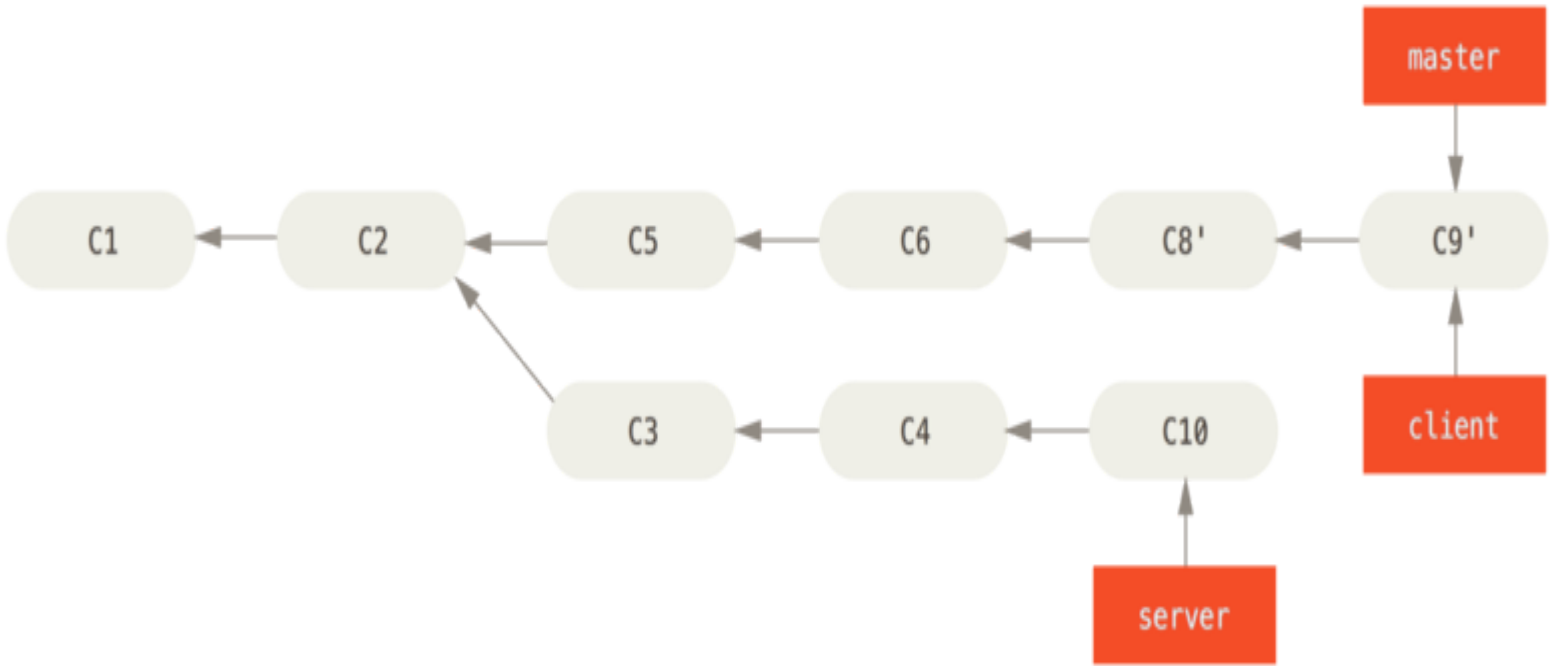
Alias

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

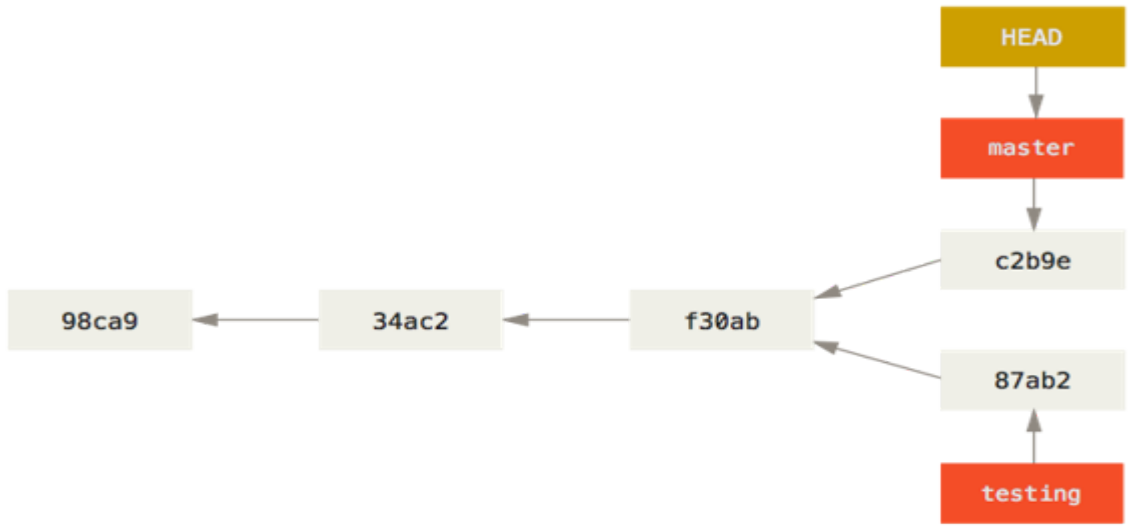
Ejemplo

```
$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



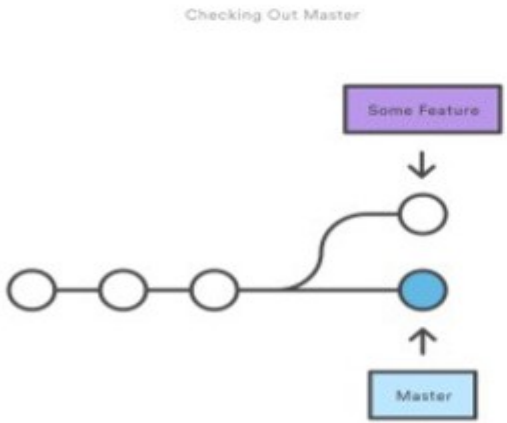
Git HEAD



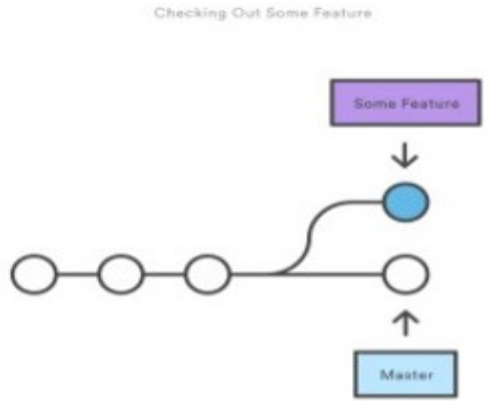
Comandos para realizar el branching

```
$git branch

$git branch -a

$git branch <NombreRama>

$git checkout <NombreRama>

$git checkout -b <NombreNuevaRama>

$git branch -d <NombreRama>

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

Git checkout en acción



git checkout master



git checkout feature

Pikicast

Git branching ejemplo