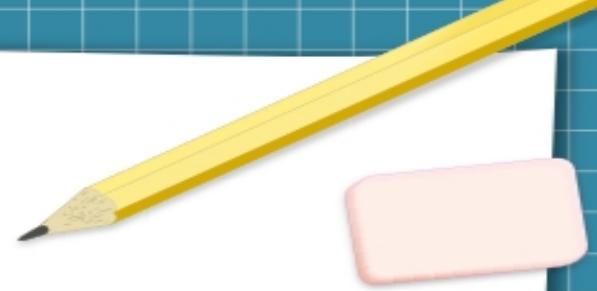


NUCLEO





<https://www.facebook.com/groups/nucleolinux.uagrm>



<https://github.com/nucleolinux-uagrm>



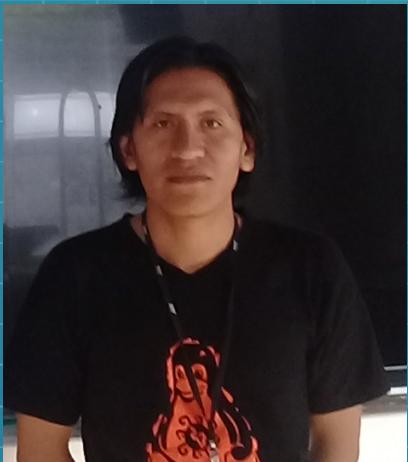
[https://t.me/nucleolinux\\_uagrm](https://t.me/nucleolinux_uagrm)



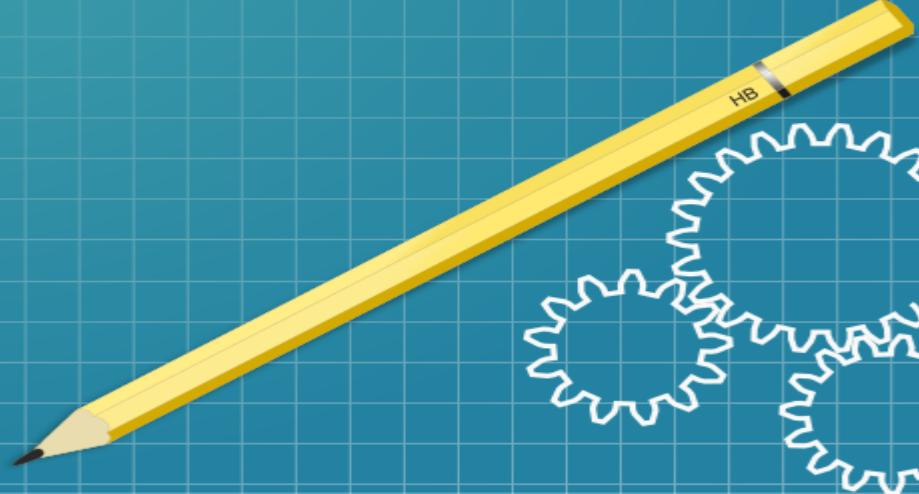


# Primeros pasos con git

## control de versiones de proyectos de software



Juan Vladimir  
@juanvladimir13



# Agenda

## 1. Sistema control de versiones (VCS)

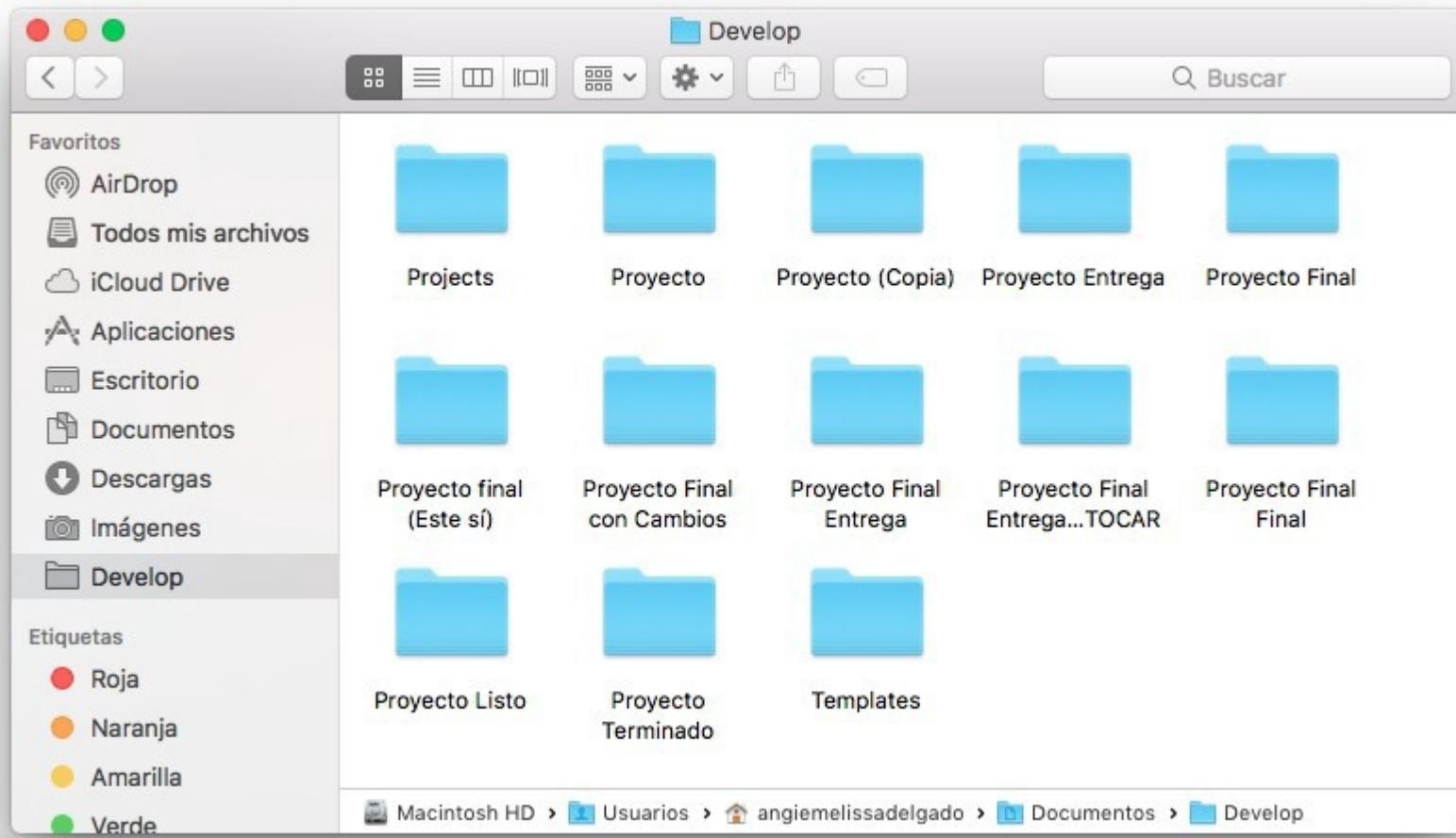
- Un desarrollador sin VCS
- ¿ Qué es ?
- Arquitecturas de almacenamiento
- Aplicaciones
- Historia de git

## 2. Primeros pasos con git

## 3. Caso de estudio



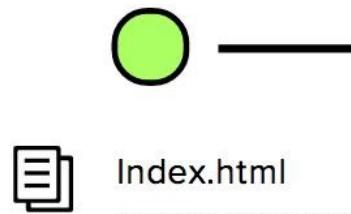
# Un desarrollador sin un VCS



# ¿Que es un VCS ?

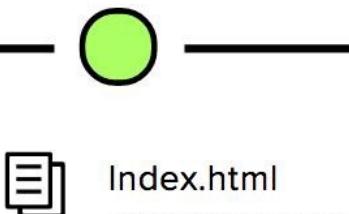
- ✓ Es un sistema que **registra los cambios realizados** sobre un archivo o conjunto de archivos a lo **largo del tiempo**, de modo que puedes recuperar versiones específicas más adelante.
- ✓ Se llama control de versiones a la gestión de los diversos cambios que se realizan sobre los elementos de algún producto o una configuración del mismo.
- ✓ Una versión, revisión o edición de un producto, es el estado en el que se encuentra el mismo en un momento dado de su desarrollo o modificación.

### Cambio 1



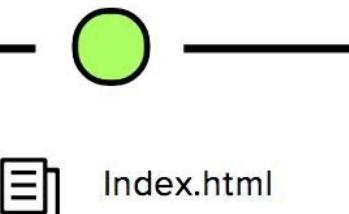
```
<DOCTYPE html>
```

### Cambio 2



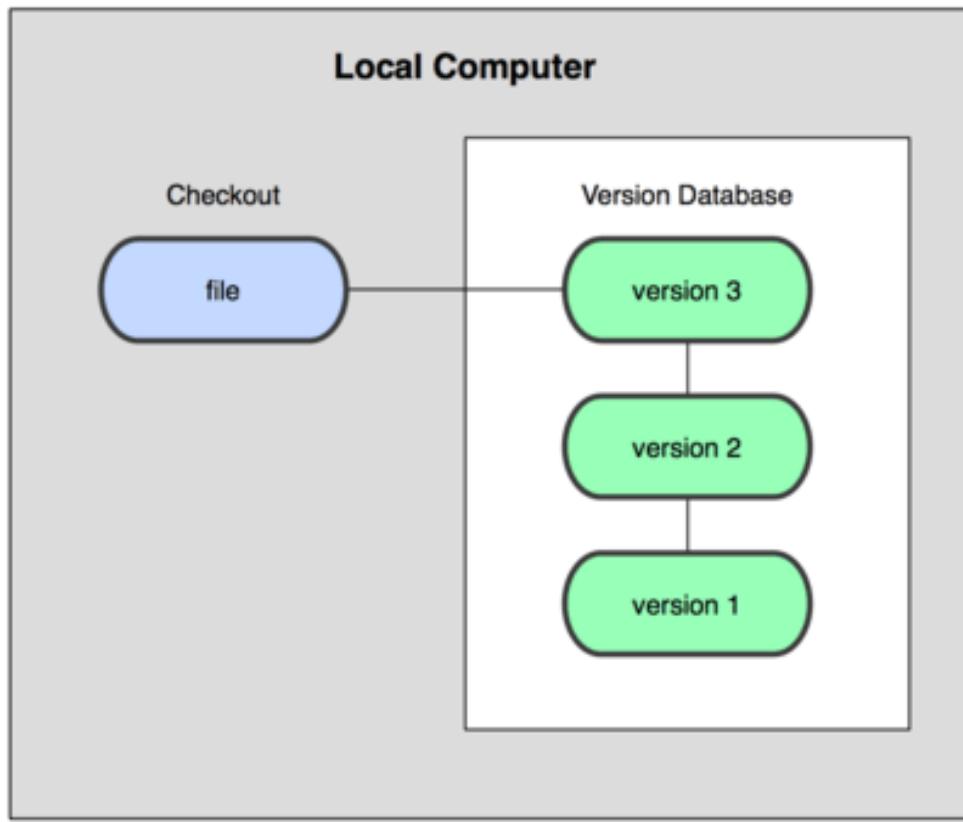
```
<DOCTYPE html>
<html>
<body>
</body>
</html>
```

### Cambio 3

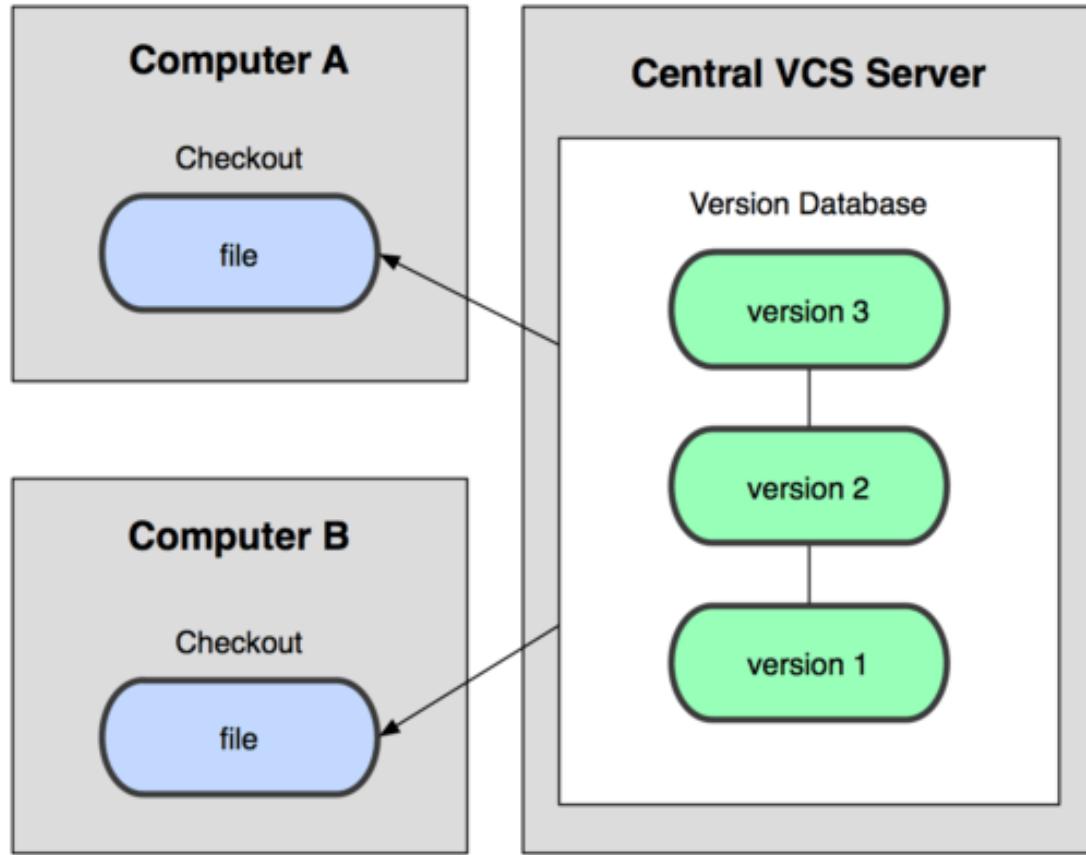


```
<DOCTYPE html>
<html>
<head>
...
</head>
<body>
...
</body>
</html>
```

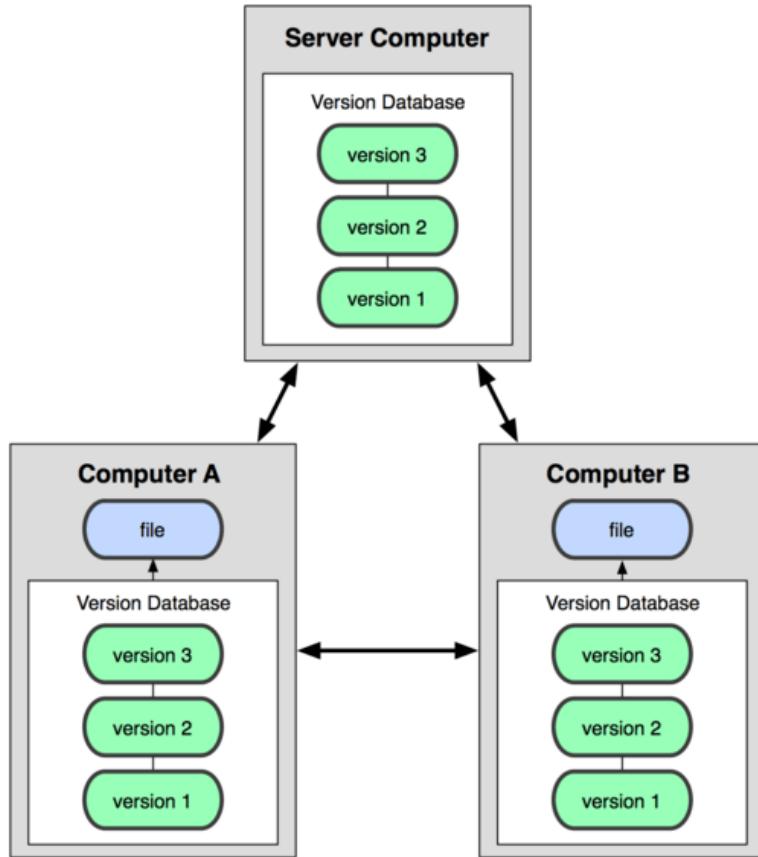
# Arquitectura de almacenamiento



# Arquitectura de almacenamiento



# Arquitectura de almacenamiento



# Aplicaciones



**PERFORCE**  
Version everything.



Bazaar



# Historia de git



El **kernel** de Linux es un proyecto de software de código abierto con un alcance bastante amplio.

En el 2002, el proyecto del **kernel** de Linux empezó a usar un **DVCS** propietario llamado ***BitKeeper***.

En el 2005, la herramienta dejó de ser ofrecida de manera gratuita.

Esto impulsó a la comunidad de desarrollo de Linux (y en particular a **Linus Torvalds**, el creador de Linux) a desarrollar su propia herramienta basada en algunas de las lecciones que aprendieron mientras usaban ***BitKeeper***

# Primeros pasos en git

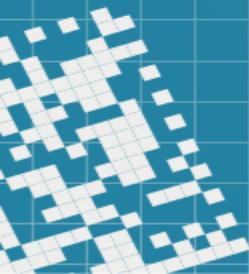
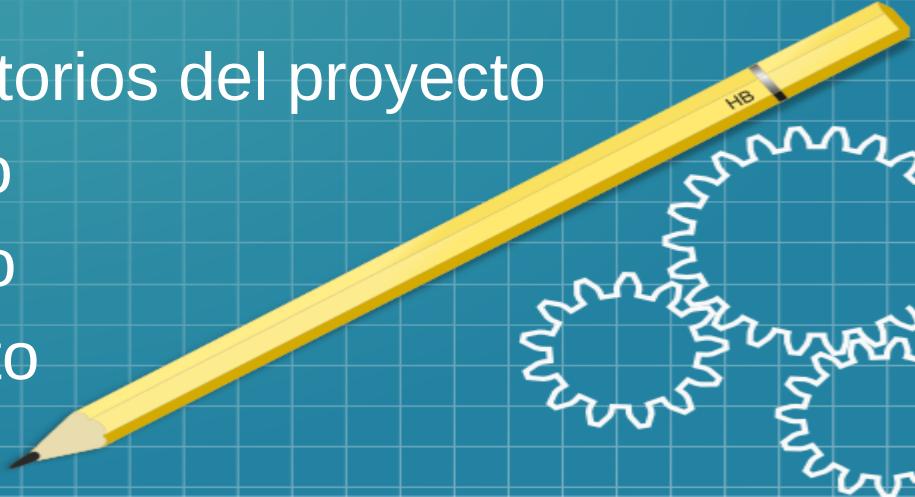


## Configurando git

- ✓ Configurando un usuario git
- ✓ Creando un proyecto

## Trabajando con un proyecto

- ✓ Excluyendo archivos y directorios del proyecto
- ✓ Ver información del proyecto
- ✓ Agregar archivos al proyecto
- ✓ Los tres estados del proyecto



# Configurando usuario git

## Verificando la instalación de git

```
git --version
```

## Configurando usuario

```
git config --global user.name "name address"
```

```
git config --global user.email email@example.com
```

```
git config --global core.editor vim
```

```
git config --global merge.tool vimdiff
```

```
git config --global color.ui true
```

## Visualizar configuracion de usuario

```
git config --list
```

# Sistema control de versiones git en un proyecto

## Proyecto vacío

```
git init project_name
```

```
cd project_name
```

## Proyecto clonado

```
git clone url | path
```

## Agregando a un proyecto existente

```
cd project_name
```

```
git init
```

# Excluyendo archivos del proyecto

Crear o modificar el archivo **.gitignore** en la raíz del proyecto

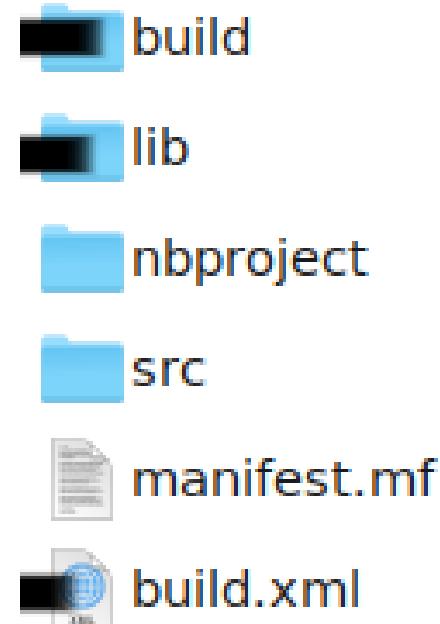
**Ignora los archivos terminados en .a**

\*.a

**Ignora únicamente el archivo TODO de la raíz**

/TODO

**Ignora todos los archivos del directorio build/  
build/**



# Ver información del proyecto

Revisando el estado de tus archivos

*git status*

Ver historial de confirmaciones

*git log*

*git log --oneline*

Ver informacion modo gráfico servidor web ligero

*git instaweb -b google-chrome*

*git instaweb --stop*

# Agregar archivos al proyecto

**Agregar un archivo o un directorio**

**git add *filename***

**Agregar todo el contenido del directorio raíz del proyecto**

**git add .**

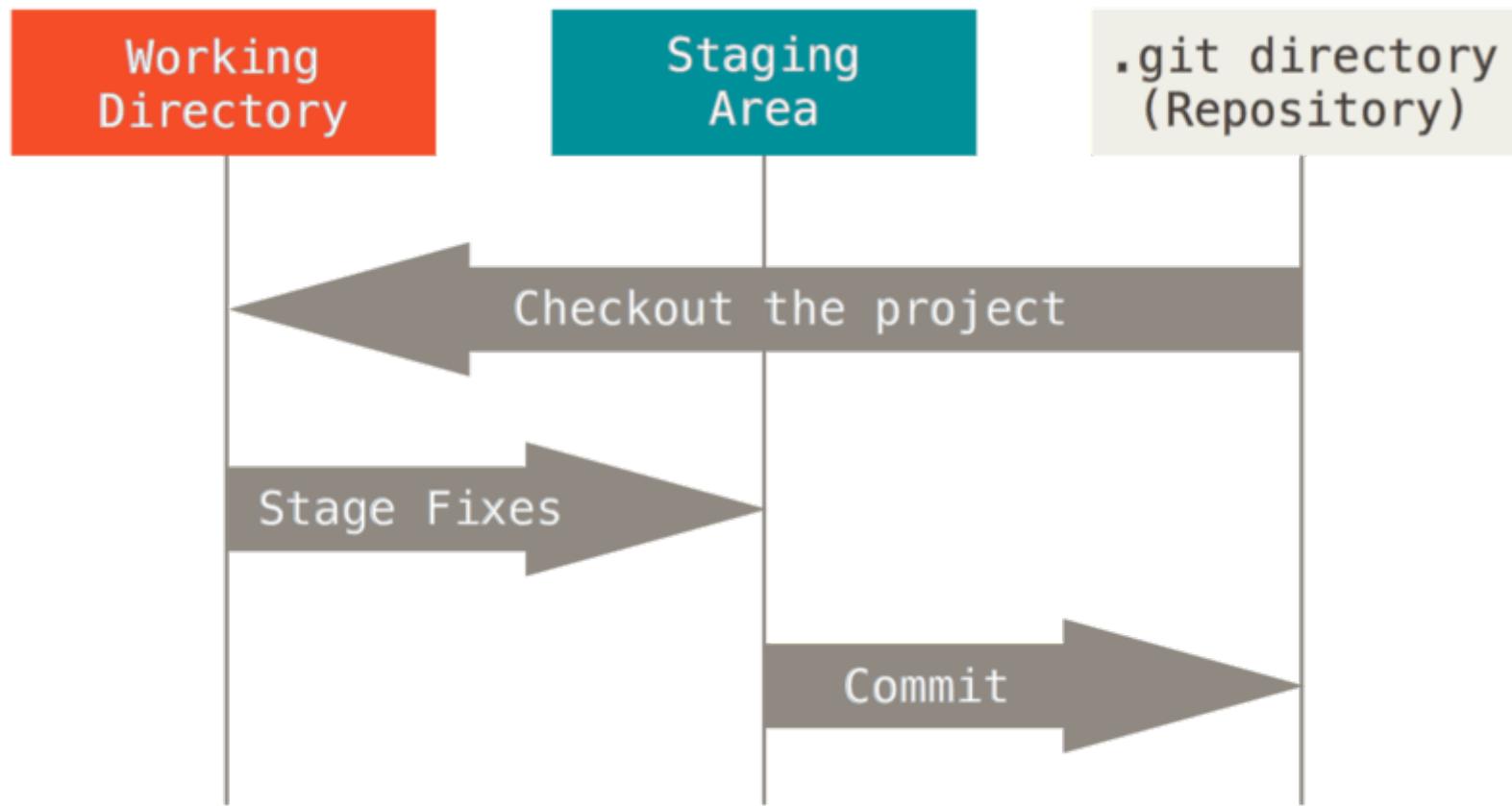
**Guardar los cambios**

**git commit -m “Nota que identifica un estado del project”**

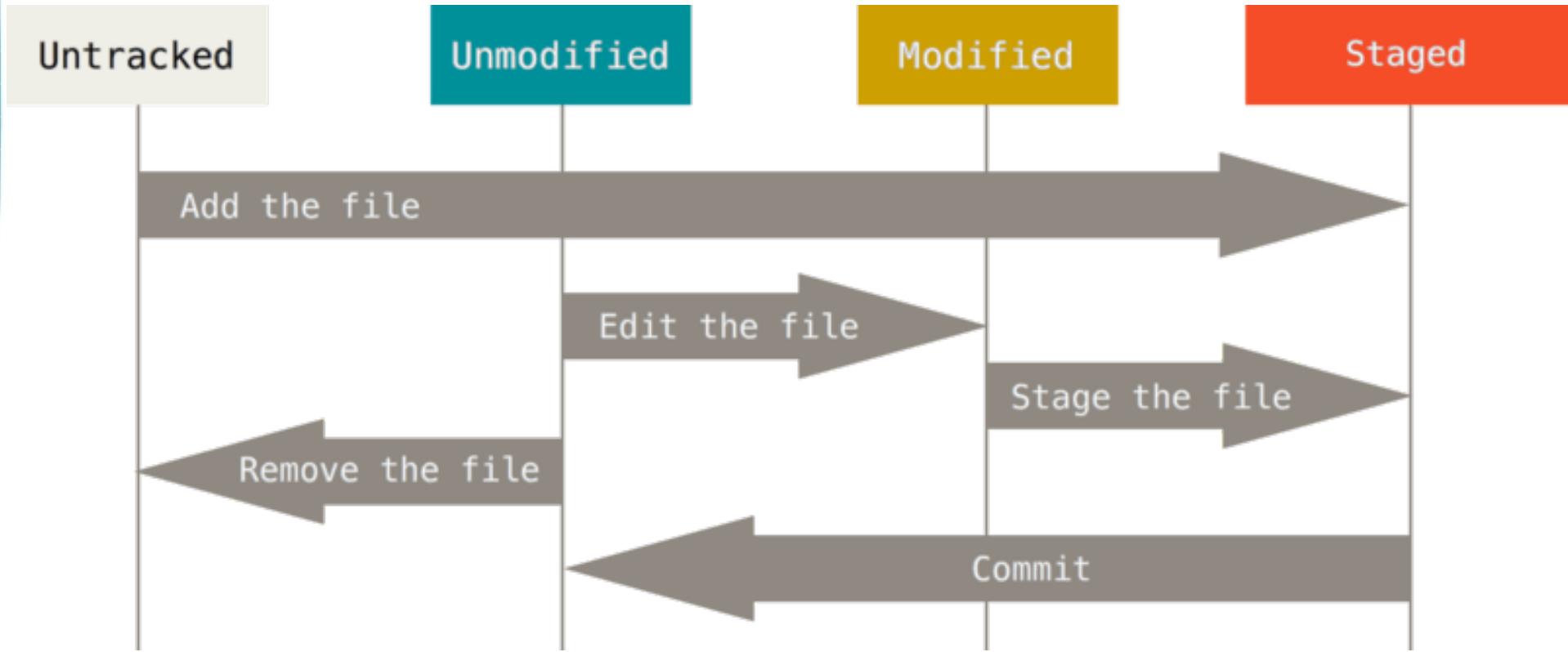
**ShortCut**

**git commit -a -m “Nota que identifica un estado del project”**

# Los tres estados del proyecto



# Estados del proyecto

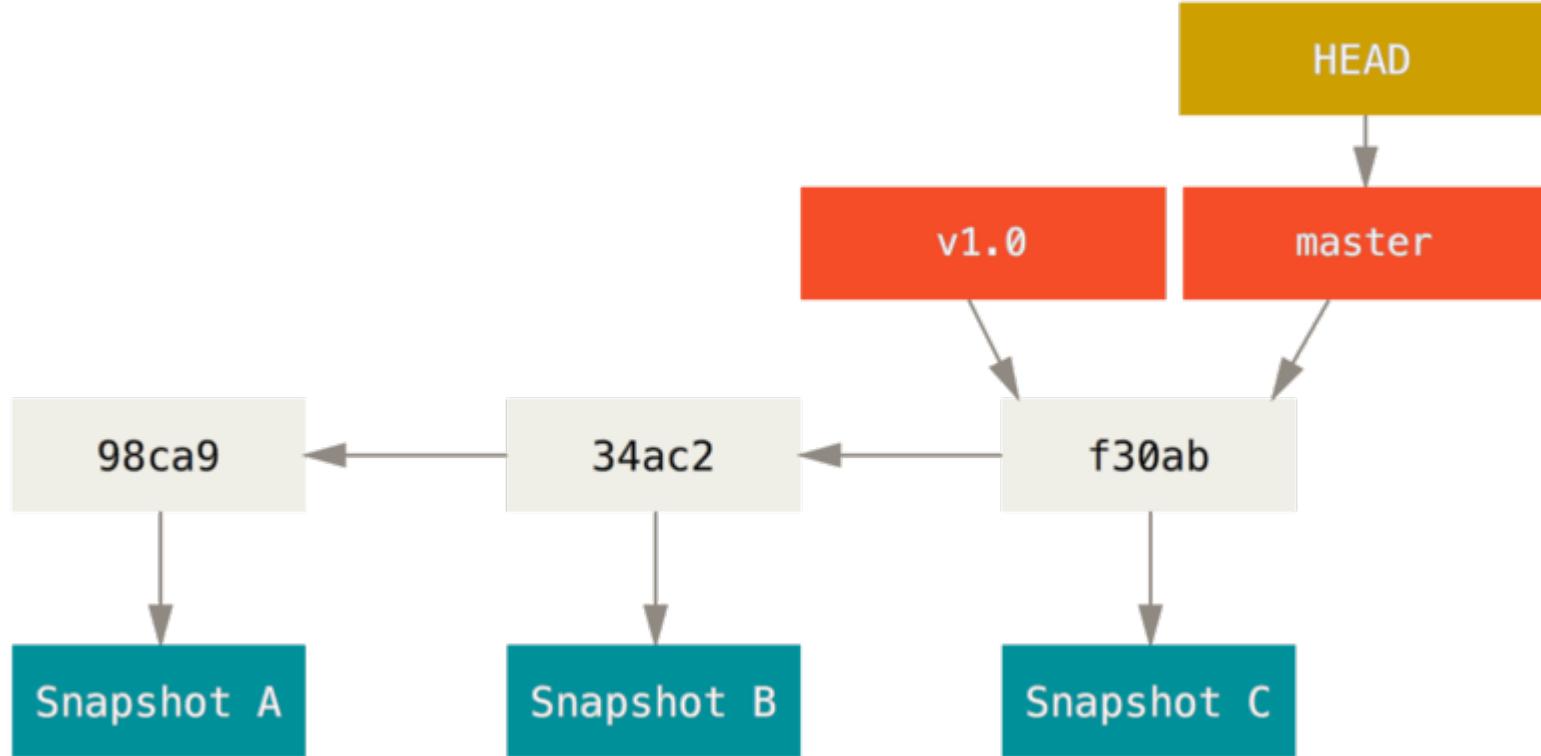


# Branch

## Ramas o branch

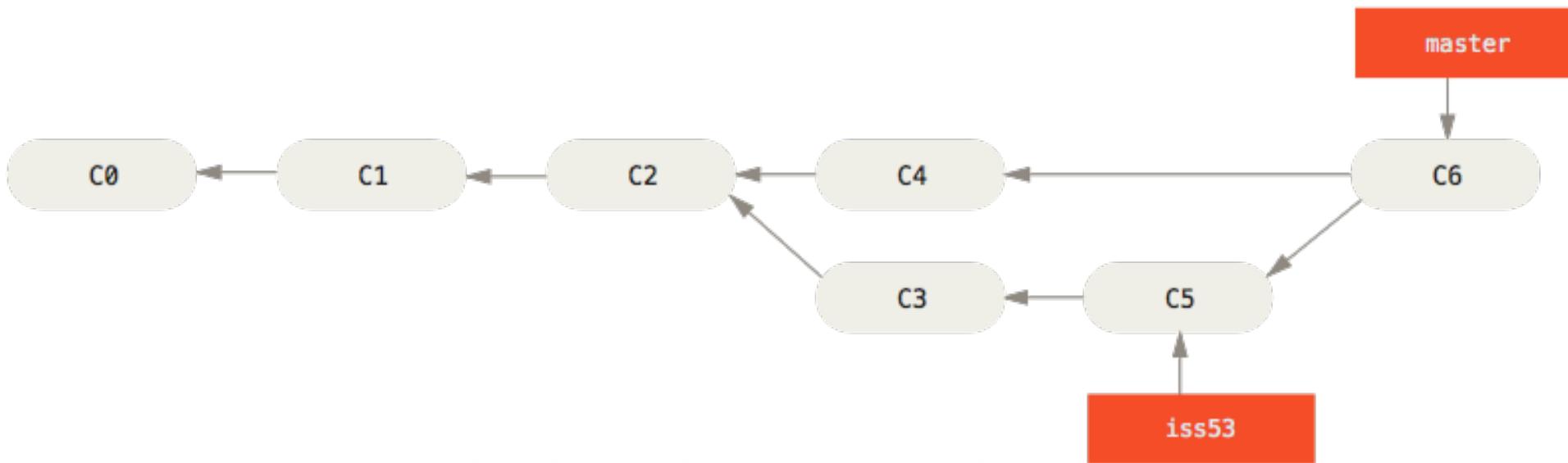
- ✓ Acciones básicas sobre un branch
- ✓ ¿ Para que sirven ?
- ✓ Crear un branch
- ✓ Merge un branch
- ✓ Tools to branch

# Ejemplo gráfico



# ¿ Para que sirven ?

- ✓ Para trabajar en una nueva versión experimental en paralelo con la versión **master** del código en producción"
- ✓ Para corregir bugs en versiones antiguas mientras sigue el desarrollo en **master**"



# ¿ Para que sirven ?

El código puede divergir en la rama respecto a la rama máster, pero luego se pueden mezclar los cambios con merge!

- ✓ Los commits en una rama son distintos a los de otras ramas"
- ✓ ¡Los ficheros que se ven en una rama son distintos cuando cambiamos de rama, aunque estamos en el mismo directorio!"

¡Siempre asegúrate antes de nada de en qué rama estás!"

- ✓ **Ojo con los ficheros abiertos:** ¡después de cambiarte a otra rama (en el mismo directorio, ¡recarga en tu editor todos los ficheros!"

Mejor: ¡rearranca el editor!"

**Los repos recién creados tienen una rama master**

# Crear un branch

Creando un nuevo branch

`git branch branch_name`

Accediendo al area de trabajo del branch

`git checkout branch_name`

ShortCut

`git checkout -b branch_name`



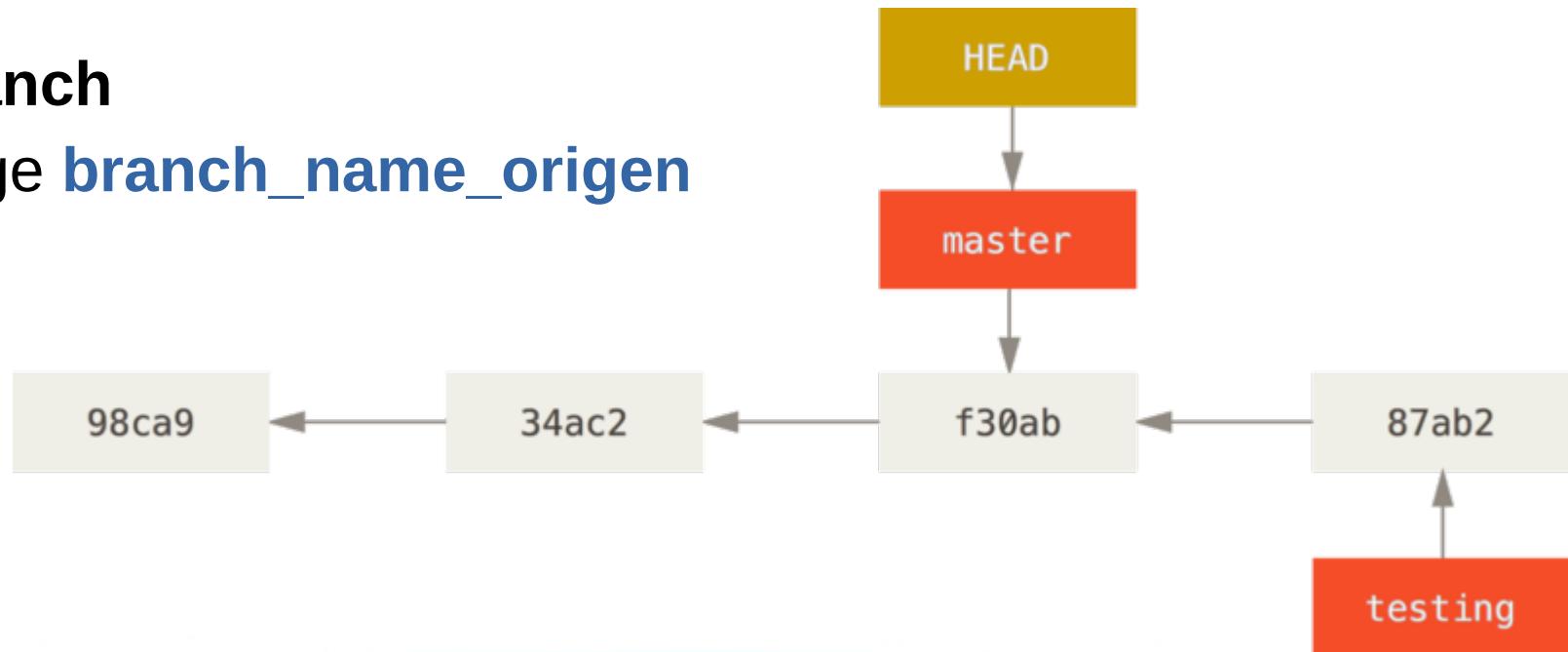
# Merge branch

Situarse en el branch destino

`git checkout branch_name_destino`

Merge branch

`git merge branch_name_origen`



# Tools to branch

**Delete the branch**

```
git branch -d branch_name
```

**Views branch**

```
git log --oneline --decorate --graph --all
```

# Remote

## Servidor de repositorios

- ✓ Configurar **github** para guardar un repositorio
- ✓ Agregar un **remote**
- ✓ Guardar proyecto en un repositorio **git**

# Configurar github para guardar un repositorio



<https://help.github.com/en/articles/checking-for-existing-ssh-keys>

<https://help.github.com/en/articles/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent>

<https://help.github.com/en/articles/adding-a-new-ssh-key-to-your-github-account>



# Agregar un remote

Agregar *remote* por defecto [ origin ]

`git remote add remote_alias url.git`

Vizualizar remote's agregados

`git remote`

Vista detallada de [ fetch , push ]

`git remote -v`

# Guardar proyecto en un repositorio git

Enviar proyecto al repositorio

*git push remote\_alias*

Enviar rama del proyecto al repositorio

*git push remote\_alias branch\_name*

Actualizar repositorio local con nuevos cambios en el servidor

*git fetch remote\_alias*

# URL's

<https://git-scm.com/>

<https://git-scm.com/book>

<https://git-scm.com/download/linux>

# Contactos y sugerencias



<https://www.facebook.com/juanvladimir13>



<https://twitter.com/juanvladimir13>



<https://www.linkedin.com/in/juanvladimir13>



<https://www.instagram.com/juanvladimir13>



@juanvladimir13



[https://www.youtube.com/channel/UCk9R\\_mLgbcENR\\_BPF9M9asQ/videos](https://www.youtube.com/channel/UCk9R_mLgbcENR_BPF9M9asQ/videos)



juanvladimir13@gmail.com



@juanvladimir13



<http://juanvladimir13.wordpress.com>



<http://juanvladimir13.blogspot.com/>



<https://github.com/juanvladimir13/>



<https://bitbucket.org/juanvladimir13>