

Aprendiendo Git

Comandos Básicos

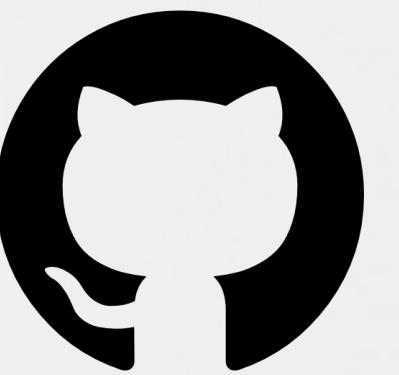


¿Qué es Git?

Es una herramienta de control de versiones, sirve para colaborar en proyectos online y guarda el historial de cambios para poder volver atrás.



Git



Github



Documentación



Algunos Comandos

Luego De La Instalación

```
→ $ git --version
```

Git mostrará la versión instalada actualmente en tu sistema.

Luego, para establecer globalmente el nombre y el correo del usuario que realizará las operaciones de Git en tu sistema.usamos

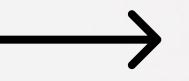
```
→ $ git config global user.name "tu_nombre"
```

```
→ $ git config global user.email "tu_email"
```



Algunos Comandos

Creando Un Repositorio



```
$ git init
```

Se crea un nuevo repositorio Git en el directorio actual o en el directorio especificado

Antes de commitear usamos el siguiente comando para obtener información sobre el estado actual del repositorio Git



```
$ git status
```



Algunos Comandos

Haciendo Commits

→ \$ git add

Se utiliza para agregar cambios en archivos específicos al staging area de Git.

→ \$ git commit -m

Se utiliza para crear un nuevo commit o confirmación en el repositorio Git.

→ \$ git push

Se utiliza para enviar los commits locales al repositorio remoto asociado



Algunos Comandos

Conectando A Github

```
→ $ git remote add origin <URL_remota>
```

Se utiliza para establecer la conexión entre un repositorio local de Git y un repositorio remoto en github.

```
→ $ git push -u origin <nombre_rama>
```

Se utiliza para realizar el push inicial de una rama local a un repositorio remoto, estableciendo una relación de seguimiento entre la rama local y la rama remota.



Algunos Comandos

Creando Una Branch

→ \$ git branch <nombre_rama>

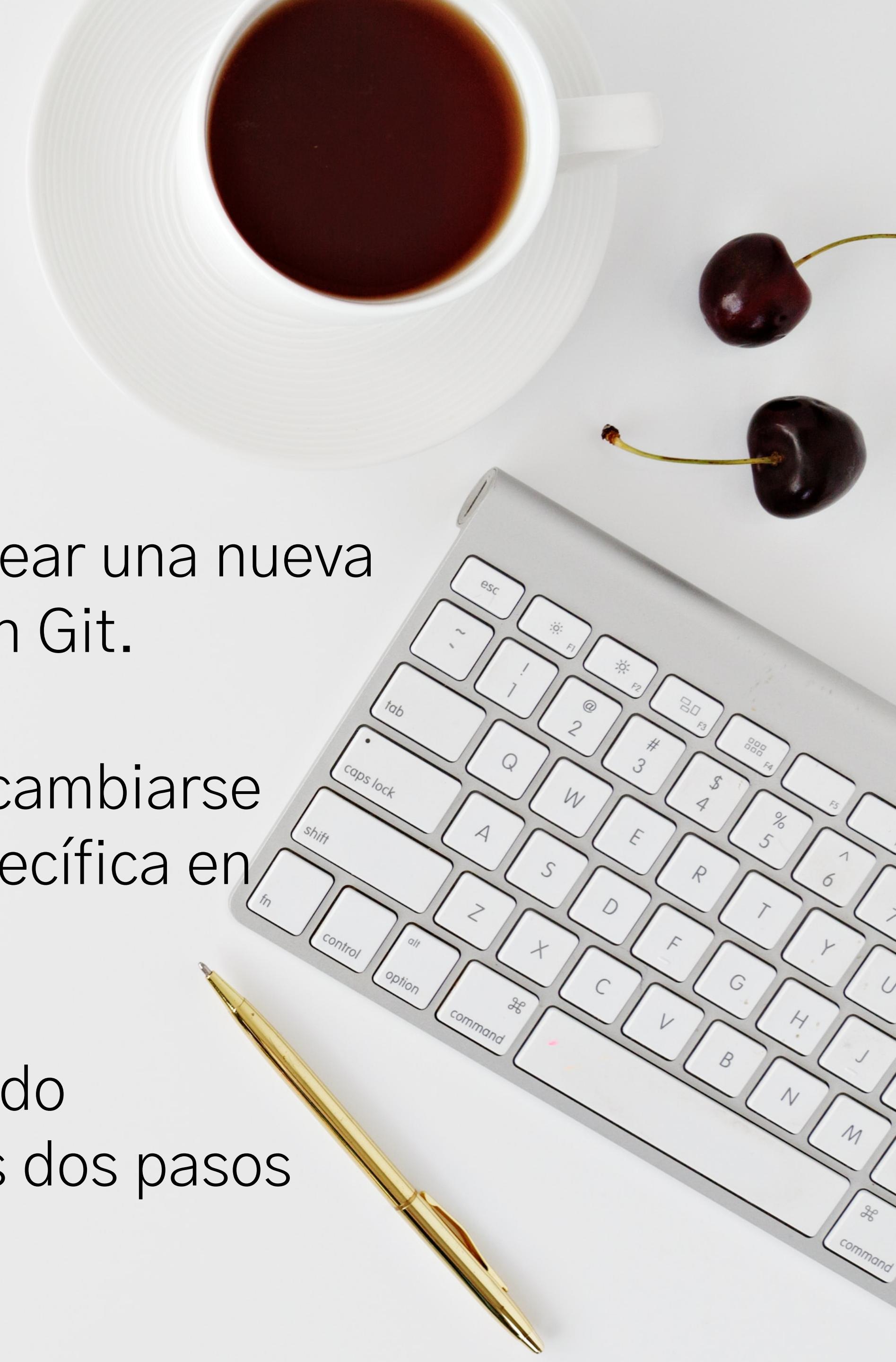
Se utiliza para crear una nueva rama (branch) en Git.

→ \$ git checkout <nombre_rama>

Se utiliza para cambiarse a una rama específica en Git

→ \$ git checkout -b <nombre_rama>

Este comando combina los dos pasos anteriores



Algunos Comandos

Borrando Commits

→ \$ git reset

Se utiliza para eliminar commits en Git y ajustar la posición de la rama actual.

→ \$ git revert

Se utiliza en Git para deshacer los cambios realizados en un commit específico creando un nuevo commit que revierte esos cambios.

A diferencia de `git reset`, que elimina los commits, `git revert` conserva el historial de commits y crea un nuevo commit que deshace los cambios introducidos por el commit que deseas revertir.



Algunos Comandos

Borrando Con Git Reset

Borrando de manera “soft”

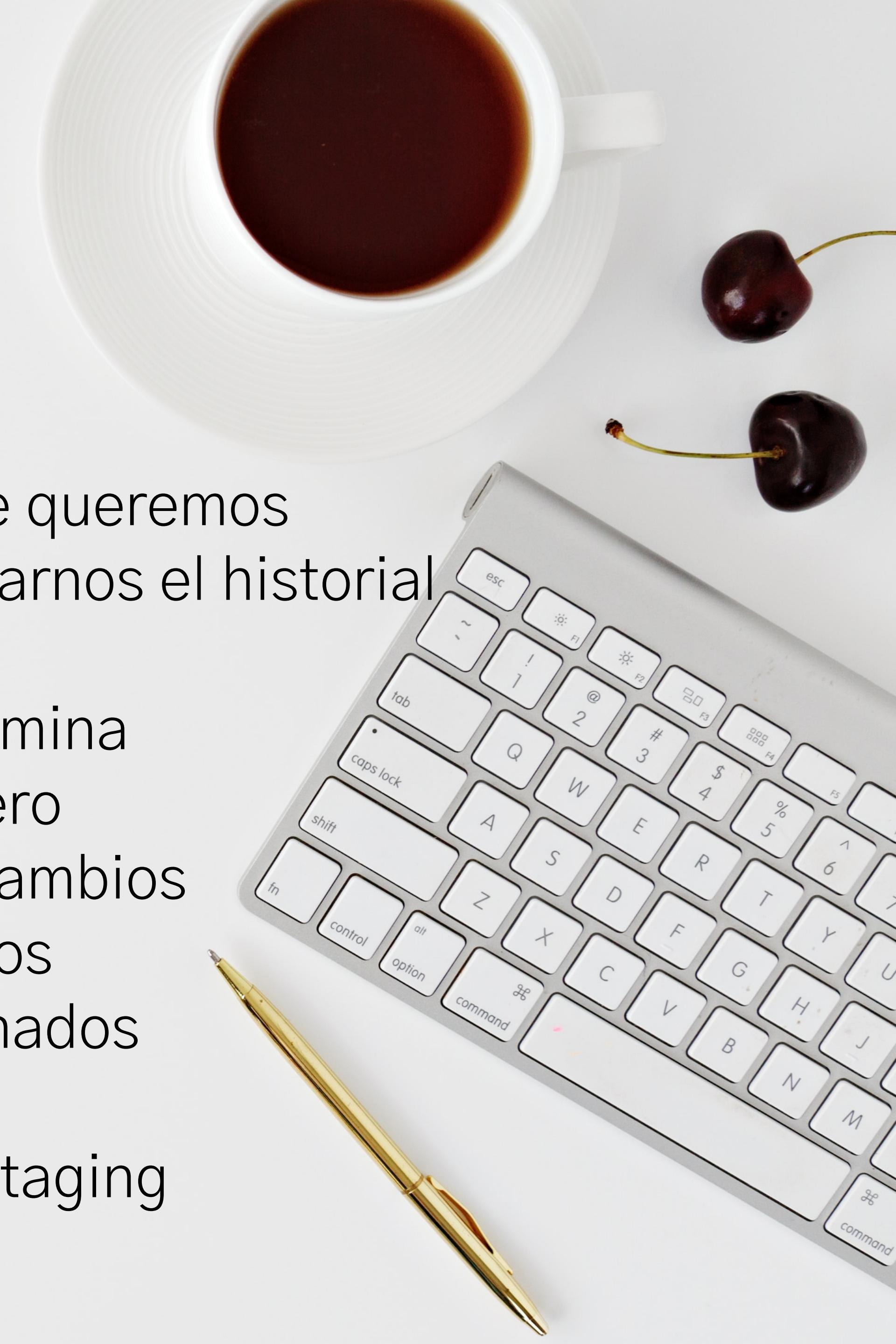
```
→ $ git log
```

Primero identificamos el commit que queremos eliminar, la función de `git log` es darnos el historial de commits.

```
→ $ git reset --soft HEAD~3
```

Por ejemplo, este comando eliminaría los últimos 3 commits y mantendría los cambios en el área de preparación para que puedas realizar nuevos commits con esos cambios.

Esta opción elimina los commits pero mantiene los cambios realizados en los commits eliminados en el área de preparación (staging area).



Algunos Comandos

Borrando Con Git Reset

Otra manera de borrar

```
→ $ git log
```

Primero identificamos el commit que queremos eliminar.

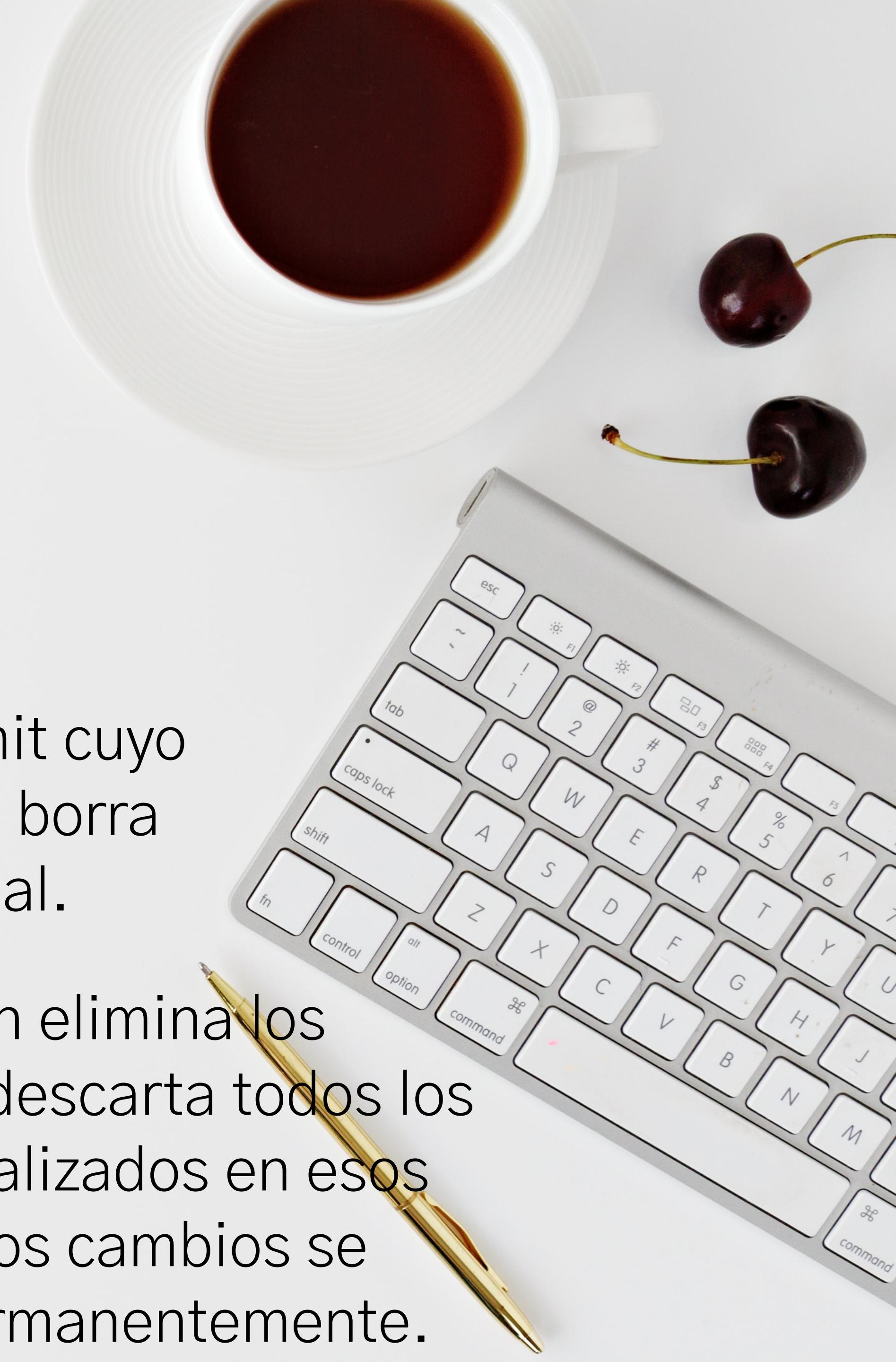
```
→ $ git reset <hash_commit>
```

Git borra el commit cuyo hash le pasamos, borra también el historial.

Borrando de manera “hard”

```
→ $ git reset --hard <hash_commit>
```

Esta opción elimina los commits y descarta todos los cambios realizados en esos commits. Los cambios se pierden permanentemente.



Algunos Comandos

Borrando Con Git Revert

→ `$ git log`

Primero identificamos el commit que queremos eliminar.

→ `$ git revert <hash_commit>`

Git creará un nuevo commit que deshace los cambios del commit especificado

Una vez que se complete el comando, Git habrá creado un nuevo commit que revierte los cambios introducidos por el commit original.

→ `$ git add .`

Guardamos los cambios y luego hacemos `git commit`

→ `$ git revert --continue`

Con esto terminamos el proceso para borrar el commit deseado.



Algunos Comandos

Otro Par De Comandos

→ \$ git merge <rama_secundaria>

En Git se utiliza para combinar los cambios de una rama en otra rama

→ \$ git rebase <rama_secundaria>

En Git reaplica los commits de una rama en la parte superior de otra rama, lo que resulta en una línea de tiempo más lineal y limpia.

→ \$ git clone URL_repositorio

Se utiliza en Git para crear una copia local de un repositorio remoto

