

Comenzamos

18:10

Versionado de Software Git Deshacer cambios

Tecnología de la Información en las Organizaciones

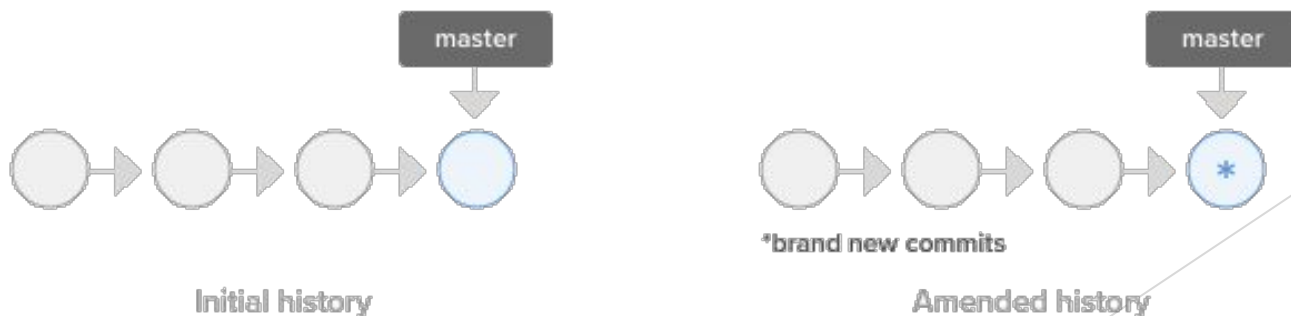
2020

Motivación

- ▶ ¿Cómo deshacer los cambios realizados sobre un archivo?
- ▶ ¿Cómo borrar el último commit?
- ▶ ¿Cómo borrar los últimos 2 commits pero manteniendo los cambios?
- ▶ ¿Cómo convertir un archivo preparado (en la *staging area*) a modificado?
- ▶ ¿Cómo guardar temporalmente los archivos sin tener que subirlos ni descartarlos?

git commit --amend

- ▶ Permite modificar el último commit realizado
- ▶ Se utiliza cuando realizamos un commit antes de tiempo y queremos incorporar algún archivo o cambiar el mensaje de confirmación
- ▶ **git commit --amend -m <mensaje>**



git commit --amend

```
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)

       modified:   js/main.js

no changes added to commit (use "git add" and/or "git commit -a")

braia@DESKTOP-LP0G5F5 MINGW64 ~/Desktop/tio-rauch (test)
$ git add .

braia@DESKTOP-LP0G5F5 MINGW64 ~/Desktop/tio-rauch (test)
$ git commit -m "Cambios en main.js"
[test 4437085] Cambios en main.js
 1 file changed, 1 insertion(+), 1 deletion(-)

braia@DESKTOP-LP0G5F5 MINGW64 ~/Desktop/tio-rauch (test)
$ git status
On branch test
Your branch is ahead of 'origin/PRUEBA' by 3 commits.
  (use "git push" to publish your local commits)

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)

       modified:   js/service.js

no changes added to commit (use "git add" and/or "git commit -a")

braia@DESKTOP-LP0G5F5 MINGW64 ~/Desktop/tio-rauch (test)
$ git add .

braia@DESKTOP-LP0G5F5 MINGW64 ~/Desktop/tio-rauch (test)
$ git commit --amend -m "cambios en main.js y service.js"
[test 823493f] cambios en main.js y service.js
 Date: Sat Apr 25 14:11:55 2020 -0300
 2 files changed, 2 insertions(+), 2 deletions(-)

braia@DESKTOP-LP0G5F5 MINGW64 ~/Desktop/tio-rauch (test)
$ git log
commit 823493faeab869c986a7cef765416b4080b46822 (HEAD -> test)
Author: Braian Varona <brvarona@gmail.com>
Date:   Sat Apr 25 14:11:55 2020 -0300

    cambios en main.js y service.js
```

Modificamos y
commiteamos
nuestros cambios

Nos olvidamos
de modificar un
archivo

Los agregamos y
realizamos un amend
para mantenerlo en
nuestro último
commit. También
actualizamos el
nombre

git checkout -- [archivo]

- ▶ Permite revertir archivos individuales o repositorios enteros
- ▶ Se utiliza cuando queremos deshacer un cambio realizado sobre un archivo determinado que se encuentra en nuestro working directory (archivo modificado)
- ▶ **git checkout -- <archivo>**

git checkout -- [archivo]

```
braia@DESKTOP-LP0G5F5 MINGW64 ~/Desktop/tio-rauch (test)
$ git status
On branch test
Your branch is ahead of 'origin/PRUEBA' by 3 commits.
  (use "git push" to publish your local commits)

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)

        modified:   js/service.js

no changes added to commit (use "git add" and/or "git commit -a")

braia@DESKTOP-LP0G5F5 MINGW64 ~/Desktop/tio-rauch (test)
$ git checkout -- js/service.js

braia@DESKTOP-LP0G5F5 MINGW64 ~/Desktop/tio-rauch (test)
$ git status
On branch test
Your branch is ahead of 'origin/PRUEBA' by 3 commits.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

Modificamos un
archivo por
error

Eliminamos esa
modificación

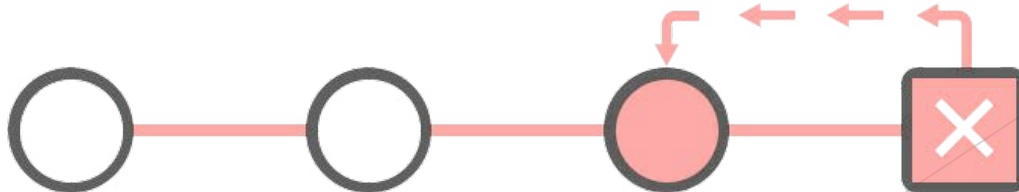
Comprobamos que
el cambio no
aparece

git reset

- ▶ Permite volver a versiones anteriores y limpiar el repositorio.
- ▶ Se utiliza cuando queremos deshacer un archivo preparado (git add) o para deshacer commits (git commit).
- ▶ Los cambios que no estén en la *Staging Area* seguirán en estado “Modified” a no ser que usemos --hard
- ▶ Se le puede especificar a qué momento queremos volver (commit, tag, etc)

git reset

- ▶ **git reset --hard**: deshacer todos los cambios en archivos modificados y preparados.
- ▶ **git reset HEAD <archivo>**: deshacer el archivo preparado.
- ▶ **git reset HEAD~<N>**: deshacer los últimos N commits manteniendo las modificaciones.
- ▶ **git reset <HASH>**: se deshacen los commits posteriores al commit especificado



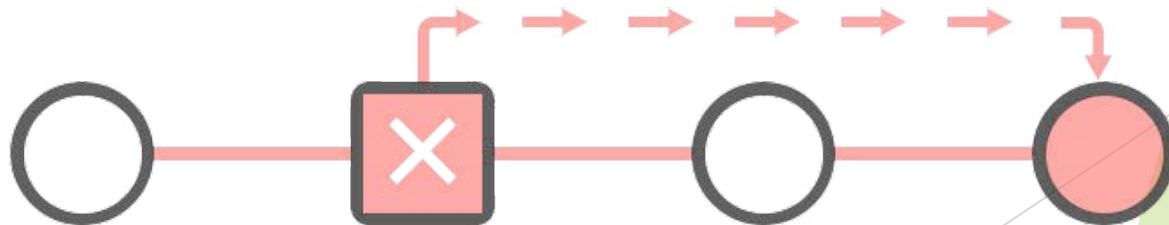
hard vs soft

- ▶ **git reset --soft:** los cambios producidos por los commits eliminados se mantendrán en el repositorio local.
- ▶ **git reset --hard:** se pierden las modificaciones realizadas por los commits eliminados.



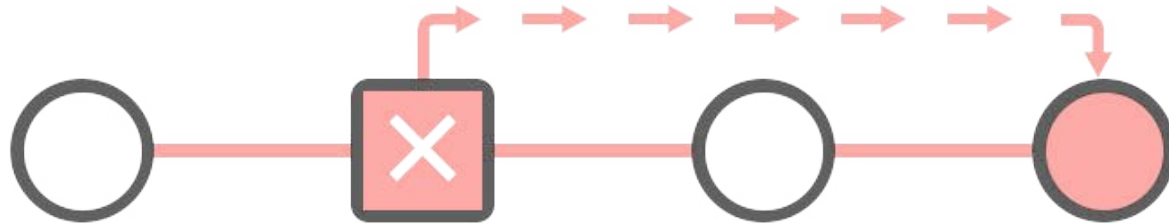
git revert

- ▶ Permite revertir el estado del proyecto, generando un nuevo commit que revierte los cambios realizados.
- ▶ Los commits no son eliminados del historial de cambios, conservando dicha información.
- ▶ Es más seguro que un reset.
- ▶ **git revert <HASH>**

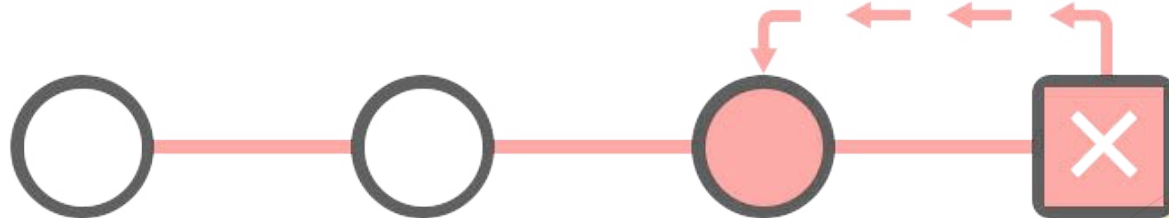


Reset and revert

Reverting



Resetting



git stash

- ▶ Permite descartar los cambios, pero guardar provisionalmente los archivos modificados.
- ▶ Se utiliza cuando se requiere resolver un bug o desarrollar una nueva funcionalidad pero sin realizar el commit del trabajo realizado hasta el momento.
- ▶ Estos cambios se descartan, pero se guardan para continuar trabajando más adelante.
- ▶ **git stash save <TÍTULO>**

git stash

- ▶ **git stash pop**: los cambios guardados en el stash vuelven al working directory.
- ▶ **git stash list**: se muestra el contenido de la pila de stash.
- ▶ **git stash apply <STASH>**: se aplica sobre el working directory los cambios del stash especificado.
- ▶ **git stash drop <STASH>**: elimina de la lista el stash especificado.

Resumen

- ▶ **git commit --amend -m <mensaje>**: sobrescribe el último commit realizado.
- ▶ **git checkout -- <archivo>**: deshace los cambios realizados sobre un archivo.
- ▶ **git reset**: permite volver a versiones anteriores
- ▶ **git revert <hash>**: revierte el estado sin perder el historial de cambios
- ▶ **git stash**: descarta los cambios, pero guarda una copia de los archivos modificados

Bibliografía

<https://git-scm.com/book/es/v2/Fundamentos-de-Git-Deshacer-Cosas>

Leer:

2.4 Fundamentos de Git - Deshacer Cosas