

# Git (libro Amazon)

[devops](#), [cursos](#), [git](#)

## uso básico

### configuración inicial

- git config --global user.name <nombre>
- git config --global user.email <email>
- git config --global core.editor <EDITOR>
- git config --global push.default {matching|simple|current|nothing} : ver *sincronizando directorios*
- git config --global core.excludesFile <FILE> : fichero global de exclusión de ficheros
- git config --list

### iniciando repositorio

- git init
- git clone <http|https|ssh>
- git config --list : en el directorio del repositorio te da info sobre el mismo

## trabajando con Git

- Directorio de trabajo → Index → Head
- git status
- git add
  - git add -u : añade al Index los archivos que deben ser borrados
  - git add -A = git add -u & git add .
- git commit
  - git commit -m «<COMENTARIO>»
  - git commit -a : git add -A & git commit

## sincronizando repositorios

- git clone
- git remote -v[vv] : lista información de los repositorios remotos
  - git remote add <ALIAS> <DIRECCION\_REPOSTITORIO>
  - git remote rm <ALIAS>
  - git remote rename <ALIAS> <NUEVO\_ALIAS>
- git pull <ALIAS> <RAMA>
  - git pull = git fetch & git merge
  - ALIAS por defecto es **origin**
  - RAMA por defecto es **master**
- git push <ALIAS> <RAMA>
  - hemos de estar al día en nuestro repositorio local para poder hacer un push
  - comportamiento por defecto v2.x : **simple** en lugar de **matching**
    - simple: solo sube rama activa a la rama de la que has hecho pull, si no tiene el mismo nombre da error
    - matching: sube todas las ramas, si no existe la crea
    - current: sube los cambios de la rama activa a la rama del mismo nombre, si no existe, se

- crea
- nothing: test/debug
- upstream : idem *simple* pero no da error si tiene otro nombre

## .gitignore

- <https://github.com/github/gitignore>

## viendo el historial

- git log
  - parámetros:
    - -#\_entradas
    - –oneline
    - -p : más detalle, con diff
    - –graph

## borrado de archivos

- git rm <ARCHIVO> : borra archivo + borra archivo del Index
  - git rm –cached <ARCHIVO> : borra solo del Index
  - git reset HEAD <ARCHIVO> : idem anterior

## arrepentimientos

### rehacer un commit

- git commit –amend : si no hay modificación de archivos (no has modificado en Index), edita el comentario
- si has olvidado algún archivo, lo añades y ejecutas la instrucción anterior

### deshacer cambios de un archivo

- git checkout – <ARCHIVO> : deshace los cambios que has hecho, lo recupera del HEAD

### viviendo al pasado

- git reset –hard <HASH\_COMMIT>:
  - deshace commits posteriores al indicado
  - recupera los archivos del commit indicado
  - DESAPARECEN TODOS LOS CAMBIOS POSTERIORES
  - se recomienda hacer un PUSH o ejecutarlo sobre otra rama

## resolviendo conflictos

- hay commits posteriores en tu rama, error al hacer push
  - hacer pull
  - resolver conflictos, si los hubiese (primero aparece lo tuyo, entre <<<< y ===== y lo que hay en el remoto está entre ===== y >>>>>)

- hacer push

## viendo/recuperando archivos antiguos

- git show <HASH\_COMMIT>
- git show <HASH\_COMMIT>:path/to/file
- git show <HASH\_COMMIT>:path/to/file > archivo\_copia

## más usos de Git

### organización

no hay reglas de como organizar tu trabajo, aunque si que hay reglas de como **NO** hacerlo... se ha de organizar para que el trabajo de los otros no te distraiga

### qué poner en el directorio principal

- README.md : fichero mostrado en la página principal de Github. Explicar en pocas palabras de que va el proyecto, como instalarlo, prerequisitos, licencia, navegación por el repositorio
- INSTALL.md : instrucciones detalladas de instalación
- .gitignore
- LICENSE
- TODO

### estructura habitual con directorio test

- Git tiene una estructura plana (el todo es tratado en un conjunto, al contrario que CVS o Subversion, que podían tratar un directorio como un proyecto independiente)
- Git permite trabajar con **submódulos** para emular este comportamiento. Con sentido en proyectos de terceros del que depende tu proyecto o proyectos/equipos muy grandes
  - git submodule add <URL\_REPOITORIO> <DIRECTORIO> + cd <DIRECTORIO> + git submodule init + git submodule update + git pull

### flujos de trabajo

From:  
<https://miguelangel.torresegea.es/wiki/> - miguel angel torres egea



Permanent link:  
<https://miguelangel.torresegea.es/wiki/info:libros:git?rev=1535967743>

Last update: **03/09/2018 02:42**