

Referencia Rápida de Git

Ramas, merge, rebase, stash, remotos

Configuración

Config de usuario

```
git config --global user.name "Your Name"
git config --global user.email "you@example.com"
git config --global core.editor vim
git config --list # ver todas las opciones
```

Init y Clone

```
git init # nuevo repo en dir actual
git clone <url> # copiar repo remoto
git clone <url> dir # clonar en dir/
```

Básico

Estado y preparación

```
git status # estado del árbol de trabajo
git add file.txt # preparar un archivo
git add . # preparar todos los cambios
git add -p # preparar de forma interactiva
```

Commits

```
git commit -m "msg" # commit de cambios preparados
git commit -am "msg" # preparar tracked + commit
git commit --amend # editar último commit
```

Diff

```
git diff # cambios sin preparar
git diff --staged # cambios preparados
git diff HEAD~1 # vs commit anterior
git diff branchA branchB
```

Ramas

Gestión de ramas

```
git branch # listar ramas locales
git branch -a # listar todas (incl. remotas)
git branch feat # crear rama
git branch -d feat # eliminar (seguro)
git branch -D feat # eliminar (forzado)
```

Cambiar de rama

```
git checkout feat # cambiar a rama
git checkout -b feat # crear + cambiar
git switch feat # cambiar (moderno)
git switch -c feat # crear + cambiar (moderno)
```

Merge

```
git merge feat # merge feat en rama actual
git merge --no-ff feat # crear siempre merge commit
git merge --abort # cancelar merge con conflicto
```

Remoto

Gestionar remotos

```
git remote -v # listar remotos
git remote add origin <url> # agregar remoto
git remote remove origin # eliminar remoto
```

Fetch, Pull, Push

```
git fetch origin # descargar actualizaciones
git pull # fetch + merge
git pull --rebase # fetch + rebase
git push origin main # push al remoto
git push -u origin feat # push + establecer upstream
```

Log

Ver historial

```
git log # historial completo
git log --oneline # formato compacto
git log --oneline -10 # últimos 10 commits
git log --graph --oneline --all # gráfico de ramas
git log -p # mostrar parches (diffs)
git log --stat # mostrar estadísticas de cambios
```

Filtrar log

```
git log --author="Alice"
git log --since="2024-01-01"
git log -- path/file # commits que tocan el archivo
git log -S "keyword" # buscar cambios en contenido
```

Stash

```
git stash # guardar cambios de trabajo
git stash push -m "wip" # guardar con mensaje
git stash list # listar todos los stashes
git stash pop # aplicar + eliminar stash superior
git stash apply # aplicar pero conservar stash
git stash drop # eliminar stash superior
git stash clear # eliminar todos los stashes
```

Deshacer cambios

Desestabilizar y descartar

```
git restore file.txt # descartar cambios de trabajo
git restore --staged file.txt # desestabilizar archivo
git checkout -- file.txt # descartar (legado)
```

Reset

```
git reset --soft HEAD~1 # deshacer commit, conservar staged
git reset --mixed HEAD~1 # deshacer commit, conservar working (por defecto)
git reset --hard HEAD~1 # deshacer commit, descartar todos los cambios
```

Revert

```
git revert <commit> # nuevo commit deshaciendo cambios
git revert HEAD # deshacer último commit (seguro)
```

Tags

```
git tag v1.0 # tag ligero
git tag -a v1.0 -m "Release" # tag anotado
git tag # listar tags
git push origin v1.0 # push de tag único
git push origin --tags # push de todos los tags
git tag -d v1.0 # eliminar tag local
```

.gitignore

Ejemplos de patrones

```
# .gitignore
*.log # todos los archivos .log
build/ # directorio build
!important.log # excepción (sí rastrear)
/TODO # solo TODO en raíz
doc/**/*.pdf # pdfs en subárbol doc/
```

Patrones comunes

```
*.ext # Todos los archivos con esa extensión
dir/ # Directorio completo
!pattern # Negar (re-incluir)
**/name # Coincidir en cualquier subdirectorio
? # Comodín de un solo carácter
```