# Guía de Git: Básico, Medio y Avanzado

## Nivel Básico: Fundamentos

### 🔧 Configuración Inicial

git config --global user.name "Tu Nombre"

git config --global user.email "tu@correo.com"

### 📁 Crear o Clonar Repositorio

git init # Crear un nuevo repositorio local

git clone URL\_DEL\_REPO # Clonar un repositorio remoto

### 💾 Flujo Básico

git status # Ver archivos modificados

git add archivo.txt # Agregar archivo al staging

git add . # Agregar todos los cambios

git commit -m "Mensaje" # Confirmar cambios

### 🔄 Trabajar con Remoto

git remote add origin URL # Conectar con repositorio remoto

git push -u origin main # Subir por primera vez

git push # Subir cambios

git pull # Obtener cambios del remoto

## Nivel Medio: Gestión y Ramas

### 🌿 ¿Qué es una Rama?

Una rama en Git es una versión paralela del proyecto. Te permite trabajar de forma aislada en nuevas características, corregir errores o experimentar sin afectar la rama principal (main).

### 📌 Conceptos Clave

| **Concepto** | **Explicación** |
| --- | --- |
| main | Rama principal del proyecto. |
| branch | Línea separada de desarrollo. |
| merge | Une los cambios de una rama en otra. |
| rebase | Mueve una rama encima de otra para una historia más limpia. |
| HEAD | Puntero que indica en qué rama y commit estás. |

### 🛠️ Gestión de Ramas

git branch # Ver ramas

git branch nueva-rama # Crear nueva rama

git checkout nueva-rama # Cambiar de rama

git switch nueva-rama # Alternativa moderna

git merge rama-a-unir # Unir rama actual con rama-a-unir

### Deshacer Cambios

git checkout archivo.txt # Deshacer cambios en archivo

git reset archivo.txt # Quitar archivo del staging

git reset --hard HEAD # Deshacer todo hasta el último commit

### 📜 Revisar Historia

git log # Ver historial

git log --oneline --graph # Vista resumida y visual

git diff # Ver diferencias no confirmadas

git show <commit> # Ver detalles de un commit

## 🔖 Etiquetas (Tags)

### ✅ ¿Qué es una Etiqueta?

Una etiqueta (tag) es un marcador permanente que se coloca sobre un commit específico para identificar una **versión estable**, como v1.0, v2.1.3, etc.

A diferencia de las ramas, las etiquetas no cambian. Son como una fotografía del estado del código en un momento importante.

### 🎯 Usos comunes

* Marcar versiones de lanzamiento (releases).
* Identificar hitos importantes (v1.0, beta, final).
* Asociar una versión con un estado del código.
* Facilitar el despliegue y distribución del software.

### 🛠️ Tipos de Etiquetas

| **Tipo** | **Descripción** |
| --- | --- |
| Ligera | Solo apunta a un commit (sin metadata). |
| Anotada | Incluye autor, fecha y mensaje descriptivo. |

### 📌 Comandos para Etiquetas

git tag v1.0 # Crear etiqueta ligera

git tag -a v1.0 -m "Versión estable" # Crear etiqueta anotada

git tag # Ver etiquetas

git show v1.0 # Ver detalles de una etiqueta

git push origin v1.0 # Subir una etiqueta

git push origin --tags # Subir todas las etiquetas

git tag -d v1.0 # Eliminar etiqueta local

git push origin --delete tag v1.0 # Eliminar etiqueta remota

## 🔴 Nivel Avanzado: Control Preciso y Colaboración

### Reescribir Historia (⚠️ Peligroso si ya se compartió)

git commit --amend # Editar último commit

git rebase -i HEAD~3 # Reescribir los últimos 3 commits

### Guardar Trabajo Temporal con Stash

git stash # Guardar cambios sin confirmar

git stash list # Ver todos los stashes

git stash pop # Recuperar el último stash

### 🌐 Repositorio Remoto Avanzado

git remote -v # Ver remotos actuales

git remote add origin URL # Agregar un nuevo remoto

git remote set-url origin URL # Cambiar URL de remoto existente

### 🔍 Buscar y Filtrar

git grep "texto" # Buscar texto en archivos versionados

git log --author="nombre" # Commits por autor

git log --since="2 weeks ago" # Commits recientes

### 📦 Submódulos

git submodule add URL ruta/ # Agregar submódulo

git submodule init

git submodule update

## 🛠️ Comandos Útiles Extra

git clean -fd # Eliminar archivos no rastreados

git blame archivo.txt # Ver quién cambió cada línea

git reflog # Ver historial de movimientos de HEAD

## 🔁 Forzar git pull y git push

### ⚠️ Advertencia

Usar --force puede sobrescribir cambios locales o remotos. Úsalo solo si sabes lo que haces.

### 🔽 Forzar git pull (destruye cambios locales)

git fetch origin

git reset --hard origin/main

### 🔼 Forzar git push (sobrescribe remoto)

git push --force

# o más seguro:

git push --force-with-lease

| **Opción** | **Descripción** |
| --- | --- |
| --force | Fuerza el push sin verificar si alguien más subió cosas. |
| --force-with-lease | Solo forza si nadie más subió algo nuevo. Más seguro. |