

# Documentazione Granulare dei Comandi Git

Generata automaticamente con GPT-5

October 14, 2025

## Contents

# 1 Inizializzazione e Configurazione

```
# Inizializza un nuovo repository Git
git init

# Clona un repository remoto
git clone https://github.com/user/repo.git

# Imposta nome e email globali
git config --global user.name "Nome Cognome"
git config --global user.email "email@example.com"

# Visualizza configurazioni correnti
git config --list
```

# 2 Lavoro Quotidiano

```
# Mostra lo stato dei file tracciati e non
git status

# Aggiunge file specifici o tutti
git add nomefile.txt
git add .

# Crea un commit con messaggio
git commit -m "Descrizione del cambiamento"

# Modifica l'ultimo commit (mantiene lo stesso hash)
git commit --amend --no-edit
git commit --amend -m "Nuovo messaggio"

# Mostra differenze tra working directory e ultimo commit
git diff

# Mostra differenze tra commit e area di staging
git diff --staged
```

# 3 Gestione Branch

```
# Mostra tutti i branch locali
git branch

# Crea e sposta su un nuovo branch
git checkout -b feature/nuova-funzionalita

# Sposta su un branch esistente
git checkout main

# Rinomina branch corrente
git branch -m nuovo-nome

# Cancella branch locale
git branch -d nome-branch
```

## 4 Merge e Conflitti

```
# Unisci branch 'feature' in quello corrente
git merge feature

# Merge senza fast-forward (mantiene la cronologia)
git merge --no-ff feature

# Se si verifica un conflitto:
# 1. Risolvi i conflitti nei file
# 2. Poi aggiungi i file risolti
git add file_conflitto.txt

# 3. Completa il merge
git merge --continue

# Annulla il merge in corso
git merge --abort
```

## 5 Revert, Reset e Stash

```
# Annulla un commit specifico (senza rimuoverlo dalla storia)
git revert <hash>

# Riporta il branch a uno stato precedente (attenzione)
```

```
git reset --hard <hash>

# Rimuove solo dallo staging area
git reset HEAD nomefile.txt

# Salva modifiche temporaneamente (stash)
git stash

# Lista degli stash salvati
git stash list

# Applica l'ultimo stash
git stash apply

# Applica e rimuove lo stash
git stash pop
```

## 6 Operazioni Remote

```
# Aggiunge un remote
git remote add origin https://github.com/user/repo.git

# Verifica i remote
git remote -v

# Push iniziale
git push -u origin main

# Pull delle modifiche remote
git pull

# Sincronizza branch remoto
git fetch origin
git merge origin/main
```

## 7 Comandi Situazionali e di Debug

```
# Mostra il log con grafico dei branch
git log --oneline --graph --all
```

```
# Mostra chi ha modificato una riga
git blame nomefile.txt

# Pulisce i file non tracciati
git clean -fd

# Mostra file modificati in un commit specifico
git show --name-only <hash>

# Ripristina un file da un commit
git checkout <hash> -- nomefile.txt

# Mostra differenze tra due branch
git diff branch1..branch2

# Mostra differenze tra working directory e remoto
git diff origin/main

# Ricrea branch remoto cancellato
git push origin nome-branch

# Cancella branch remoto
git push origin --delete nome-branch
```

## 8 Risolvere Conflitti Complessi

```
# Mostra file con conflitti
git diff --name-only --diff-filter=U

# Accetta versione locale o remota in blocco
git checkout --ours .
git checkout --theirs .

# Dopo aver risolto tutto:
git add .
git merge --continue
```

## 9 Comandi “Salvavita”

```
# Recupera commit persi (es. dopo un reset)
```

```
git reflog

# Mostra tutti i branch (locali e remoti)
git branch -a

# Elimina cache Git (utile per .gitignore)
git rm -r --cached .

# Reinstalla submodule
git submodule update --init --recursive
```