Git Informazioni

Sviluppatore Guida A Git

Imbrogli Foglio #1

Configurazione

$ git config user.name <nomeutente>	Definisce il nome utente per l'utente del *repository corrente* (configurazione locale) come coppia chiave-valore.
$ git config user.email <email>	Definisce il messaggio di posta elettronica per l'utente del *repository corrente* (configurazione locale) come coppia chiave-valore.
$ git config pull.rebase <false|true>	Imposta il comportamento predefinito delle operazioni pull, ovvero l'unione o la ribase per il repository corrente.
$ git config --global <any-command>	Il flag '--global' modifica l'ambito della configurazione dal file 'config' locale (repository corrente) a tutti i repository sul computer.

Informazioni di base

$ git init	Inizializza un repository Git nella directory corrente.
$ git clone <url>	Clona un repository Git remoto nella directory corrente, in una directory con il nome di quella remota.
$ git clone <URL> nome	Clona un repository Git remoto nella directory corrente, in una directory chiamata 'name'.
$ git add <file_name>	Aggiunge un file aggiornato con il nome file specificato, dall'albero di lavoro all'area di gestione temporanea.
$ git add <fileglob>	Aggiunge i file aggiornati corrispondenti dall'albero di lavoro all'area di gestione temporanea.
$ git add .	Aggiunge tutti i file aggiornati dall'albero di lavoro all'area di gestione temporanea.
$ git status	Stampa lo stato dei file modificati nell'albero di lavoro, contrassegnandoli rispettivamente come staged, unstaged e tracked.
$ git commit -m "<message>	Crea un commit con il messaggio specificato per le modifiche nell'area di gestione temporanea.

Ramificazione

$ git branch <branch_name>	Crea un nuovo ramo con il nome di ramo specificato. Impossibile creare due rami con lo stesso nome.
$ git branch -m <branch_name> <new_name>	Rinomina il ramo.
$ git branch -M <branch_name> <new_name>	Forza la ridenominazione del ramo. Se esistono nomi duplicati, il ramo che stai rinominando sostituirà quello vecchio.
$ git checkout <ramo>	Esegue il Check-Out (imposta l'albero di lavoro) sul ramo specificato.
$ git checkout -b <ramo>	Crea un nuovo ramo e ne esegue il check-out. Comando di convenienza che combina i due prima di questo.
$ git merge <branch>	Unisce il ramo specificato a quello attualmente estratto.
$ git rebase <branch>	Riapplica i commit del ramo attualmente estratto, oltre al ramo specificato nel metodo.
$ git log	Registra tutti i commit nel ramo attualmente estratto, con informazioni sull'oggetto commit.

$ git log -n	Registra gli ultimi 'n' commit nel ramo attualmente estratto, con le informazioni sull'oggetto di commit.
$ git log --oneline	Registra tutti i commit nel ramo attualmente estratto, con ogni commit in una singola riga contenente solo hash e messaggio.
$ git log --graph	Visualizza i commit registrati nell'interfaccia della riga di comando.

Filiali remote

$ git fetch <remote> <branch>	Aggiorna l'origine/ramo locale pertinente con il contenuto del ramo del repository remoto specificato.
$ git fetch <remoto> <remote_branch>:<local_branch>	Aggiorna l'origine/ramo locale specificato con il contenuto del ramo del repository remoto specificato.
$ git pull <remote> <branch>	Esegue un recupero del ramo specificato dal repository remoto specificato, quindi unisce il risultato nel ramo pertinente del repository locale.
$ git pull <remote> <remote_branch>:<local_branch>	Eseguire il pull dal ramo remoto specificato al ramo locale specificato.
$ git pull -u <remote> <branch>	Imposta a monte tra la filiale locale e quella remota.
$ git pull	Tira dal ramo e nel ramo che hai collegato tramite il flag '-u'.
$ git push <remote> <branch>	Invia i commit del ramo locale pertinente al ramo specificato del repository specificato.
$ git push <remoto> <local_branch>:<remote_branch>	Imposta in modo esplicito il ramo da e verso cui esegui il push.
$ git push -u <remote> <branch>	Come la spinta regolare, ma imposta il ramo remoto come a monte di quello corrente.
$ git push	Come il precedente, ma utilizza a monte del ramo corrente.

Avanzato

$ git stash push	Salva le modifiche indicizzate nella scorta (uno spazio locale dedicato per i lavori in corso).
$ git stash	Salva le modifiche non organizzate nella "scorta" per l'archiviazione temporanea.
$ git stash pop	Riapplica la modifica salvata in precedenza dalla scorta e la rimuove da essa.
$ git stash si applica	Riapplica le modifiche salvate in precedenza dalla scorta e le mantiene nella scorta.
$ git stash si applica | pop stash@{0}	Applica/fa scoppiare la scorta specificata dall'elenco delle scorte cambiate. L'ultimo è '0'.
$ git stash list	Elenca tutte le scorte salvate.
$ git tag <tag>	Assegna un'etichetta a 'HEAD'.
$ git tag -d <tag>	Elimina il tag.
$ git tag -a <tag>	Crea un oggetto Tag, non solo un'etichetta.
$ git push <remote> <tag>	Push tag al repository remoto.

$ git checkout <tag>	Checkout al tag specificato. Di solito, in modalità HEAD staccata.
$ git diff <file>	Verificare la differenza tra gli stati dei file in fasi e quelli non in prova.
$ git diff <commit1> <commit2>	Verificare la differenza tra i commit.
$ git diff <branch1> <branch2>	Controlla la differenza tra i rami.
$ git add -i	Modalità di gestione temporanea interattiva .

Ripristino e modifica della cronologia

$ git cherry-pick <riferimento>	Crea una copia del commit specificato, sotto un hash nuovo di zecca.
$ git rebase -i <riferimento>	Riapplica i commit del ramo corrente al riferimento specificato, ma consente azioni specifiche su ogni commit.
$ git reset <riferimento>	Reimposta un ramo sul commit specificato, apportando modifiche di cui non è stato eseguito il commit. Il valore predefinito è 'reset --mixed'.
$ git reset --hard <reference>	Reimposta un ramo su un determinato commit, ignorando le modifiche.
$ git reset --soft <reference>	Reimposta un ramo su un determinato commit, mantenendo le modifiche in fasi.
$ git revert <riferimento>	Crea un anticommit del commit specificato, eseguendo un nuovo commit nella cronologia con le modifiche opposte.
$ git revert --continue	Continuare a ripristinare dopo aver risolto il conflitto di ripristino.
$ git revert --abort	Interrompere il ripristino.
$ git commit --amend -m "nuovo messaggio"	Modifica l'ultimo commit.

Errori comuni

- Push prima di tirare mentre ci sono nuove modifiche sul repository remoto.
  Soluzione: tirare le modifiche prima di spingere le nostre.

- Unione o ribasatura durante le modifiche nell'area di gestione temporanea.
  Soluzione: eseguire il commit o l'archiviazione delle modifiche prima di unire, ribasare o estrarre.

- Commettere file errati o messaggi errati.

Soluzione: '$ git *commit -amend -m "new message'"*, dopo '*$ git add* file' dopo aver aggiunto i file pertinenti.

- Fatto un errore di battitura nel nome del ramo.
  Soluzione: '*$ git branch -m branch-name new-name*'.

- Rimuovere i file o le directory dall'indice.
  Soluzione: '*$ git reset HEAD*'.