Git Commands

Ludovico Latini

April 2022

Contents

1 Un po di storia

Git è un software per il controllo di versione distribuito utilizzabile da interfaccia a riga di comando, creato da Linus Torvalds nel 2005.

L'idea Linus era quella di tenere traccia delle proprie versioni del kernel linux, nasce per essere a prova di *git* (slang inglese che vuol dire idiota). Il sistema prese molto piede poi negli anni successivi diventanto uno degli strumenti per tenere tracce di progetti più usati al mondo. Inoltre il software è attualmente considerato uno dei migliori software per collaborare su progetti in team ed è quindi lo stromento più utilizzato per la pubblicazione del codice di programmi open source.

Basati su git sono stati creati numerosi software e si→ti che mettono a disposizione, insieme a delle funzionalità proprie, sia a pagamento che non, dei server git da poter utilizzare.

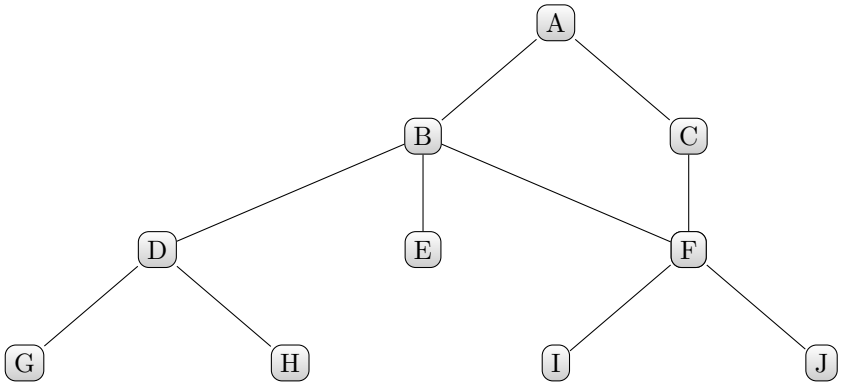
I siti più noti che offrono questo srvizio sono *GitHub* e *GitLab*.

2 Comandi base

Ecco una lista dei comandi base che si possonoutilizzare per compiere le operazioni più elementari di git:

- Creare una nuova repository → ***git init***
- Clonare una repository → ***git clone*** *(indirizzo repository)*
- Scaricare una copia del repository online → ***git pull***
- Aggiungere alla propria repository un file o più → ***git add*** *(nome di uno o più file oppure . per indicare tutta la cartella)*
- Creare un commit → ***git commit*** *(anche con opzione* ***-m*** *per specificare direttamente il nome del commit seguita da* ***"nome del commit"***
- "Pushare" o pubblicare nella repo online uno o più commit → ***git push***
- Vedere le modifiche ai file fatte dall'ultimo commit → ***git status***
- Vedere info per ogni commit → ***git log***

3 Commit e muoversi tra di essi



Il grafico rappresenta una possibile situazione in cui possiamo vedere diversi commit rappresentati da delle lettere maiuscole e diversi brach rapresentati da biforcazioni. La succesione A,B,D,F rappresenta uno stesso branch il quale condivide un commit con il branch formato da A,C,F,J.

Nell'attuale situazione è possibile muoversi in diversi modi:

- Muoversi tra vari commit → ***git checkout*** *(id del commit)*
- Muoversi di x commit indietro → ***git checkout HEAD~x*** *(dove ~ si chiama tilde e x rappresenta il numero di step*
- Muoversi ad un commit precedente di un dato branch → ***git checkout HEAD^x*** *(dove ^ è il cappello e x è un numero che nel caso in cui fosse 1 si può omettere)*

Ecco degli esempi pratici di come spostarci se si parte dal presupposto che si sta in A:

- Da A a B → git checkout HEAD^ / HEAD^1 / HEAD~1
- Da A a C → HEAD^2
- Da A a D → HEAD^^ / HEAD^1^1 / HEAD~2
- Da A a G → HEAD^^^ / HEAD~3

4 Branch comandi utili

- Creare un branch → ***git branch*** *(nome del branch)*
- Spostarsi tra un branch ed un altro → ***git branch*** *(nome del branch)*
- Eliminare un branch → ***git brach -D*** *(nome del branch)*
- cambiare il commit in cui un branch punta ***git reset –hard*** *(id del commit)*

5 Merge comandi utili

Ora che sappiamo come creare un branch come commitare modifica e come aggiornare la repo locale ciò che manca è sapere come fare il merge con uno o più branch:

- Fare un merge → ***git merge*** *(ID del commit o nome del branch)*

Quando si fa un merge git potrebbe riscontrare dei conflitti se così fosse i conflitti vanno risolti e posso essere visualizzati all'interno del file che li crea, in cui si può trovare descritto il conflitto nel seguente modo: parte di codice del commit o del branch in cui stiamo sotto HEAD e parte di codice incriminata ed appartenente al branch o al commit con cui stiamo facendo il merge sotto "nome branch"

Una volta risolti i conflitti si prosegue facendo ***git add*** e ***git commit*** e il merge andrà a buon fine.

Se invece non si è sicuri di come risolvere il conflitto allora si può annullare il merge in questo modo:

- Abortire un megre → ***git merge –abort***

6 Altri comandi utili

- Fare un rebase → ***git rebase*** *(ID commit o nome branch)*
- Recuperare un file eliminato → ***git checkout*** *(ID commit o nome branch) (nome file)*
- Eliminare un commit errato → ***git revert*** *(ID commit errato)*
- Visualizzare una cronologia delle HEAD attraversate con l'account → ***git reflog***