Introduzione a Git - Cos’è e come installarlo

Prima ancora di essere un web designer sono uno sviluppatore freelance. Lavorando a molti progetti in remoto ho capito quanto sia importante utilizzare strumenti che facilitino la collaborazione online e permettano la condivisione di codice, testi e, perché no, dei propri design. Un modo per farlo è utilizzare un sistema di **version control**, come il **programma Git**.

Un **sistema di controllo versione** serve a registrare i cambiamenti che si fanno su un file o su una serie di file nel tempo, così da poter richiamare una versione specifica di quei dati in qualsiasi momento.

Perché usare un version control system?

Alcuni anni fa, prima di conoscere i version control systems, ogni volta che mi trovavo a dover fare un backup del mio codice, dovevo copiare tutti i file in un'altra directory che chiamavo "yyyy.mm.dd-backup-project". Questo è un metodo molto usato perché semplice e veloce, ma è anche incredibilmente scomodo e soggetto ad errori.

Per questo motivo, quando ho iniziato a gestire i progetti utilizzando questi sistemi, ho fatto i salti di gioia. Grazie a un version control system posso:
- ripristinare i file ad uno stato precedente
- ripristinare l'intero progetto allo stato precedente
- confrontare i cambiamenti all'interno dei file
- nel caso di lavoro in team, vedere chi ha realizzato determinate modifiche, quando e perché
… e molto altro ancora.

Per capire meglio l’utilità dei sistemi di controllo versione mi basti dire che prima, se facevo dei pasticci o per errore cancellavo dei file, rischiavo l'infarto, ora invece, se succede, scrivo 4 parole sul terminale e tutto torna a posto.

Perché Git?

Una domanda che potreste farmi è: "Ci sono tanti tipi di controllo versione, perché dovremmo usare proprio Git?".

I motivi sono molteplici.

Gratis e open source

Git è nato dopo che molti sviluppatori di **Linux** furono costretti ad abbandonare l'utilizzo di un controllo versione chiamato **BitKeeper** poiché il detentore dei diritti aveva ritirato la possibilità di

utilizzarlo gratuitamente.

Torvald Linus allora si mise a cercare un sistema simile ma non trovò nulla che lo soddisfacesse. Nel **2005** quindi fece partire lo sviluppo di Git e pochi mesi dopo venne rilasciata la prima versione gratis e open source.

Veloce e sicuro perché è un sistema distribuito

I sistemi di controllo versione possono essere di tre tipi:

- Locale
- Centralizzato
- Distribuito

1) I primi funzionano attraverso un semplice database installato in locale che registra tutti i cambiamenti dei file. Se però il computer contenente i dati incorre in qualche problema, non sarebbe possibile recuperarli.

2) I sistemi di controllo centralizzati hanno un singolo server che contiene tutte le versioni dei file. Se per qualche motivo, però, il server dovesse avere dei problemi, nessuno potrebbe più riuscire a collaborare e a salvare le proprie modifiche.

3) Se si uniscono i primi due sistemi, nascono i sistemi di controllo distribuiti. Tutti gli utenti che lavorano al progetto hanno una copia completa del repository (l'insieme di tutte le versioni dei vari file). In questo modo è come se esistesse una copia di backup per ogni persona che lavora al progetto, senza che i membri del team abbiano bisogno di connettersi al server per visualizzare la storia di un progetto.

Quest’ultimo è il sistema utilizzato da Git, per questo motivo risulta più sicuro e veloce degli altri sistemi. **Sicuro** perché la copia completa del repository è presente nel server ed è posseduta da tutti gli utenti che lavorano al progetto. **Veloce** perché opera in locale quindi c’è bisogno di connettersi al server solamente per eseguire un aggiornamento dei dati.

Stabile e popolare

Grazie a queste e a molte altre caratteristiche che non ho elencato, Git è diventato estremamente popolare in quest'ultimo periodo e viene usato da tutte le più grosse company della Silicon Valley (Google, Facebook, Linux, Android, Apple, Twitter, etc. etc.).

Come opera Git?

È molto importante capire questo punto per affrontare al meglio il processo di apprendimento.

I file controllati da Git possono assumere tre stati principali:

- **Modified**: il file è stato modificato ma non è stato contrassegnato in modo da essere aggiunto al database
- **Staged**: Il file modificato è stato contrassegnato in modo da essere inserito nel database
- **Committed**: il file modificato è stato registrato nel database

Quindi il flusso di lavoro in Git funziona come segue:

1. Modifico i file nella mia directory di lavoro e i file assumono lo stato di "modified"
2. Eseguo l'operazione di stage dei file e li contrassegno in modo che vengano aggiunti al database nel commit successivo
3. Eseguo il commit che immagazzina in modo permanente i miei file modificati all'interno del database

"Immaginate il flusso di lavoro di un ristorante. Il cuoco prepara il piatto (status = modified), non appena ha finito mette il piatto nel bancone (status = staged). A questo punto se il cuoco vuole aggiungere altri ingredienti può sempre prendere il piatto dal bancone, aggiungere gli ingredienti e rimettere il piatto nel bancone oppure può chiamare il cameriere e ordinargli di portare il piatto al cliente *(status = committed)*".

Primo passo: installiamo Git

Installare Git è semplicissimo, vi basterà connettervi al link e seguire l'installazione guidata:

http://git-scm.com/download

Per controllare che l'installazione sia avvenuta con successo, apriamo il nostro terminale (o prompt dei comandi) e digitiamo il comando:

Se è stato installato correttamente allora riceverete in risposta la versione di Git presente nel vostro sistema.

GUI per Git

Quando Git è installato nel sistema è possibile effettuare tutte le operazioni da riga di comando oppure utilizzare alcuni software che facilitano questo processo.

Windows

- SourceTree

- Github

- TortoiseGit

- Msysgit

Mac

- SourceTree

- Github

- GitX

Linux

- SmartGit

- GitCola

Questa guida vi introdurrà alle operazioni da riga di comando così da aiutarvi a capire in modo più approfondito come opera questo sistema di controllo versione.

Configurazione di Git

Ora che è stato installato è opportuno personalizzare il nostro ambiente di lavoro. Questa configurazione è necessaria solo la prima volta, le informazioni da voi inserite rimarranno invariate dopo ogni aggiornamento. Tuttavia, se ne avete bisogno, potete cambiarle in ogni momento.

La prima cosa che occorre fare è impostare il proprio nome utente e indirizzo mail. Ciò è importante così che gli altri componenti del team possano riconoscere chi ha fatto determinati cambiamenti. Per far ciò basterà eseguire i seguenti comandi nel terminale:

L'opzione --global ci permette di eseguire questa modifica una sola volta, dopo di che Git utilizzerà sempre queste informazioni per qualsiasi operazione fatta sul sistema.

Se volete controllare le informazioni che avete inserito, vi basterà eseguire:

Ognuna di queste istruzioni vi restituirà le informazioni che avete inserito.

Conclusioni

In questa prima guida abbiamo imparato quindi cos'è un sistema di version control, perché Git è la scelta migliore tra i tanti, come installarlo e come eseguire la prima configurazione.

Nel prossimo articolo di questa guida Git, invece, impareremo ad utilizzare questo sistema in modo da poter gestire al meglio i nostri progetti.

INTRODUZIONE A GIT: INDICE LEZIONI

- Cos'è e come installarlo
- Quali sono le funzioni principali di Git
- Usare Git come strumento di collaborazione online