Coding Bootcamp

Web fundamentals

Git

Programma

Introduzione a Git

Installazione e prima configurazione

Creare un nuovo progetto

Lavoriamo nel progetto

Importare un progetto

Lavoriamo in remoto

I branch

Introduzione a Git

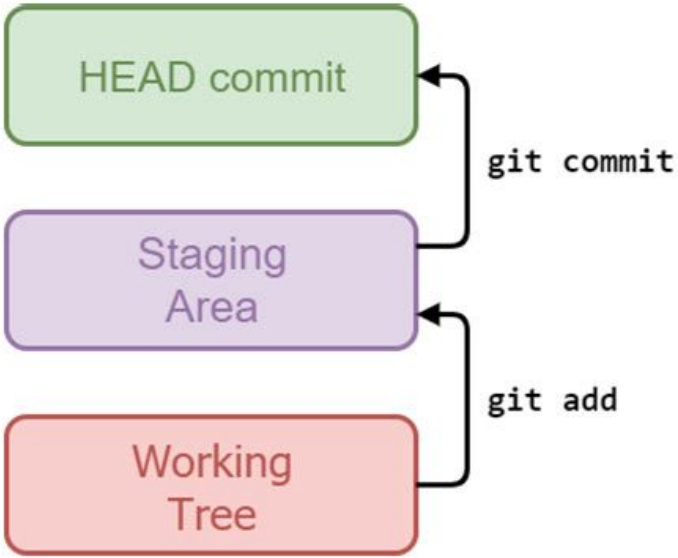
Git è un software di **controllo di versione** (VCS, Version Control System).
Sostanzialmente un meccanismo che permette di tenere traccia nel tempo delle modifiche apportate ad un progetto.

All'interno di un **team di lavoro** fornisce gli strumenti per:

- Monitorare i cambiamenti effettuati;
- risalire ad una modifica che potrebbe aver causato un malfunzionamento;
- annullare una modifica ripristinando la versione precedente di un file o di un progetto.

È un software gratuito e open source, e le sue eccellenti prestazioni lo rendono il VCS più **diffuso** e **utilizzato**.

Tranquilli niente di complicato



Installazione e prima configurazione

Sia per Mac che per Windows l'installazione è un processo semplicissimo e gratuito, basta usare i semplici installer.

windows: https://gitforwindows.org/
mac: https://sourceforge.net/projects/git-osx-installer/

Se l'installazione va a buon fine utilizziamo il terminale per lanciare le istruzioni per la prima configurazione. Usiamo la nostra identità per lavorare o contribuire ad un progetto:

git config **--global** user.name "Mario Rossi"
git config **--global** user.email mario.rossi@gmail.com

"Chi ha fatto questa azione?"

Creare un nuovo progetto

Esistono principalmente due modi: il primo è **iniziare** un progetto Git da una directory sul nostro computer. Il secondo è **importare** un progetto esistente su un altro server.

Creiamo il nostro primo progetto git:

Portiamoci nella cartella del progetto tramite il terminale e lanciamo il comando:

> git init

Questo comando crea all'interno del vostro progetto la directory **.git** che conterrà tutti i file che faranno funzionare il repository.

Lavoriamo nel progetto

Tracciamo un file e lo aggiungiamo allo Stage:

> **git add** <nome file>

Aggiungiamo il file nell'Head:

> **git commit** -m "Messaggio del commit"

Annulliamo le modifiche:

> **git restore** <nome file>

Per togliere un file dallo stage:

> **git restore --staged** <nome file>

Visualizzare e ripristinare una precedente versione del nostro progetto:

> **git log**

> **git revert** <hash>

Importare un progetto

Il secondo metodo è importare un progetto esistente su un altro server.

Cloniamo un repository

Portiamoci nella cartella dove vogliamo clonare il progetto e tramite terminale lanciamo i comandi:

> **mkdir** wp-clone
> **cd** wp-clone
> **git clone** https://github.com/WordPress/WordPress.git

Git creerà sul tuo computer una copia completa del progetto **completa di cronologia** delle modifiche.

Lavoriamo in remoto

Colleghiamo il repository sul server (Github):

> **git remote add <nome> <url>**

Scarichiamo i commit dal server remoto senza intaccare il lavoro in locale:

> **git fetch**

Facciamo il merge in locale:

> **git merge**

Fetch e merge in un unico comando:

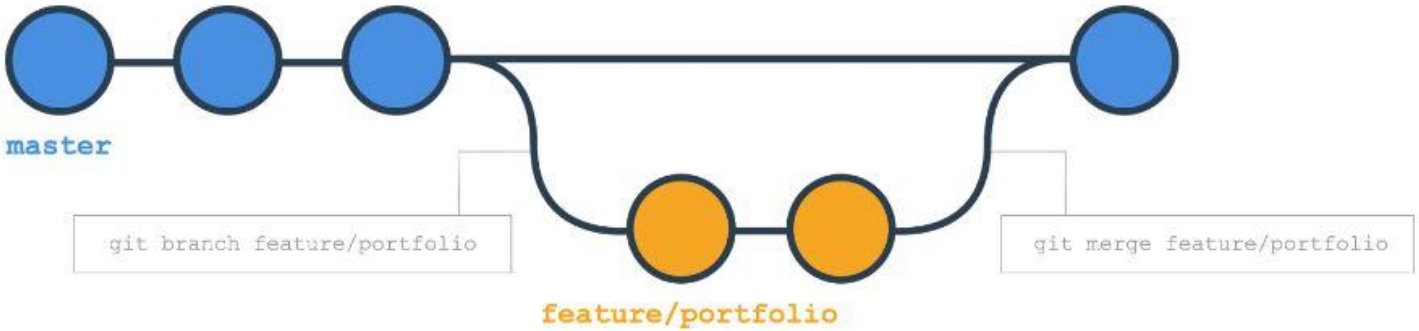
> **git pull**

Carichiamo il nostro lavoro sul repository remoto:

> **git push**

I Branch
Le ramificazioni

Una branch **è un percorso diverso e staccato** da quello principale (master) del progetto. Questo è molto utile e da la possibilità di avere dei flussi paralleli e indipendenti in cui possiamo sperimentare nuove funzioni o correggere bug **senza intaccare il progetto principale** per poi, eventualmente, **unirli (merge)** nel ramo principale o in altri rami.
Git incentiva molto l'uso dei branch perché renderanno il lavoro molto più **agile e semplice da gestire**.



I Branch
lavorare con le ramificazioni

Visualizziamo la lista dei rami:

> git branch

Creiamo un nuovo ramo:

> git branch <nome>

Cancelliamo un ramo:

> git branch -d <nome>

Saltiamo tra un ramo e l'altro:

> git checkout <nome>

Uniamo due rami:

> git merge <nome> -m "messaggio"

Se aggiungiamo l'opzione **--no-ff** diciamo a Git di creare un commit relativo a questo merge così da poterla identificare in futuro.

Uniamo due rami senza vim/vi(o il tuo $EDITOR)

> git merge <nome> --no-edit

Visualizziamo una lista dei rami non uniti:

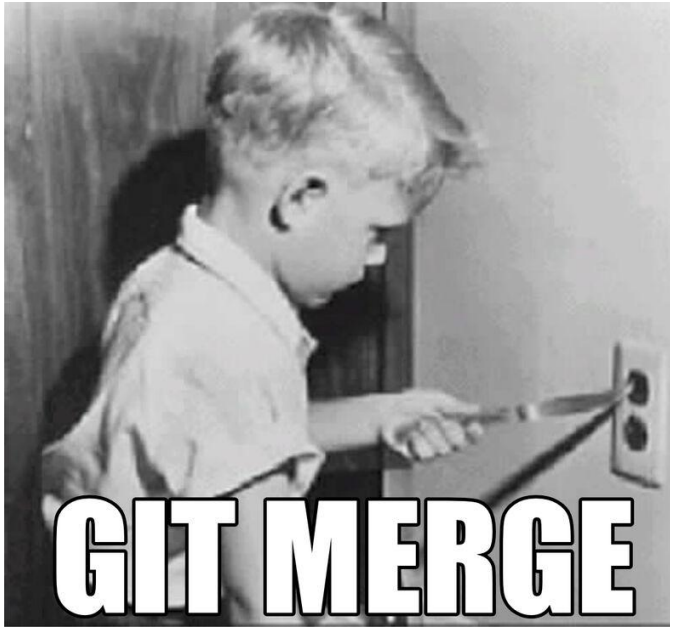
> git branch --no-merge

Risolvere un conflitto

Potrebbe capitare che tutta l'automazione di Git non vada a buon fine, specialmente se abbiamo fatto modifiche allo stesso file in più rami che sono in conflitto tra di loro. Git non sarà in grado di gestire la fusione del file in modo pulito e fin tanto che il conflitto ci sarà, la fusione tra rami non avverrà.

Esempio di conflitto:

```
<<<<<<< HEAD
<span> codice contenuto in locale </span>
=======
<span> codice del commit </span>
>>>>>>> 6ef9c569a908bbd3a2fd296a02ffe5bb41a41bd4
```



Esercizio

- Installa git sul tuo pc o mac;
- inizializza un repository Git in locale;
- crea il file index.html, inseriscilo nello stage e fai il tuo primo commit;
- crea un nuovo branch, spostati su di esso e apporta delle modifiche al file index.html, fai il commit e successivamente un merge del nuovo ramo con quello master;
- creati un account su gitHub: https://github.com/
- collega il repository remoto a quello locale e procedi con la sincronizzazione;
- modifica il file index.html da remoto e aggiorna la copia locale del tuo repository.

Se non ne hai abbastanza divertiti finché non dovrai risolvere un conflitto ;)