FINTECH DEVELOPER

Fondamenti di version control

Docente: Loredana Frontino

Titolo argomento: Storia dei VCS

Fondamenti di Version Control – Riassumendo

Recap generale

Un Version Control System (VCS) consente di gestire le modifiche apportate ai file di un progetto su cui tipicamente lavorano più persone garantendo :

- Supporto alla memorizzazione dei codici sorgenti
- Uno storico di ciò che è stato fatto
- Lavoro collaborativo e in parallelo

Fondamenti di Version Control – Storia dei VCS

Tipologie

Locale

Centralizzato

Distribuito

Fondamenti di Version Control – Storia dei VCS

Tipologie

Locale Centralizzato Distribuito

Composti da un database di versionamento speciale, il cui compito è quello di memorizzare i cambiamenti dei file

Non permette la collaborazione perché lavora in locale sulla macchina

Fondamenti di Version Control – Storia dei VCS

Tipologie

Locale

Composti da un database di versionamento speciale, il cui compito è quello di memorizzare i cambiamenti dei file

Non permette la collaborazione perché lavora in locale sulla macchina

Centralizzato

Il Database di versionamento viene spostato su un server. I computer si collegano al database e sincronizzano le proprie versioni

La presenza di un server non permette di lavorare offline e di recuperare le informazioni se questo viene distrutto.

Distribuito

Fondamenti di Version Control – Storia dei VCS

Tipologie

Locale

Composti da un database di versionamento speciale, il cui compito è quello di memorizzare i cambiamenti dei file

Non permette la collaborazione perché lavora in locale sulla macchina

Centralizzato

Il Database di versionamento viene spostato su un server. I computer si collegano al database e sincronizzano le proprie versioni

La presenza di un server non permette di lavorare offline e di recuperare le informazioni se questo viene distrutto.

Distribuito

È un sistema ibrido che comprende entrambe le tipologie viste in precedenza.

Prevede sempre un server con il database di versionamento, ma i client hanno una replica del database e la versione finale dei vari file.

Ogni collaboratore ha la propria copia di lavoro e, se una delle macchine non è più accessibile, una copia di lavoro si può recuperare. Permette di lavorare offline perché esiste una copia locale del progetto.

Fondamenti di Version Control – Storia dei VCS

Esempi di centralizzato e distirbuito

GIT	SVN
Git è open source; è stato sviluppato da Linus Torvalds nel 2005. ha la sua forza nella rapidità e nell'integrità dei dati	Apache Subversion è open source sotto Apache license.
strumento Distribuito	strumento Centralizzato
ogni utente ha la propria "copia" dei sorgenti in locale	esiste un repository centrale al quale bisogna collegarsi per scaricare il progetto, confrontare il codice, committare

Fondamenti di Version Control – Storia dei VCS

Esempi di centralizzato e distirbuito

GIT	SVN
NON serve una connessione per effettuare le operazioni	è necessaria una connessione per effettuare le operazioni
più difficile da imparare (ha una curva di apprendimento Maggiore), ha più concetti e comandi di SVN	molto più semplice da padroneggiare rispetto a git.
non ha un'interfaccia utente nativa	ha un'interfaccia utente semplice

Fondamenti di Version Control – Riassumendo

Step per avviare un aggiornamento di progetto

Ogni PC ha un repository **LOCALE** che viene generato **CLONANDO** il repository remoto (**PRODUZIONE**)

Le modifiche personali vengono verificate e committate sul repository locale

A questo punto, la situazione in **LOCALE** è definita, bisogna portare le modifiche sul repository remoto (**PRODUZIONE**)

L’operazione di "riversare" il repository **LOCALE** con quello **REMOTO** viene detta **PUSH**

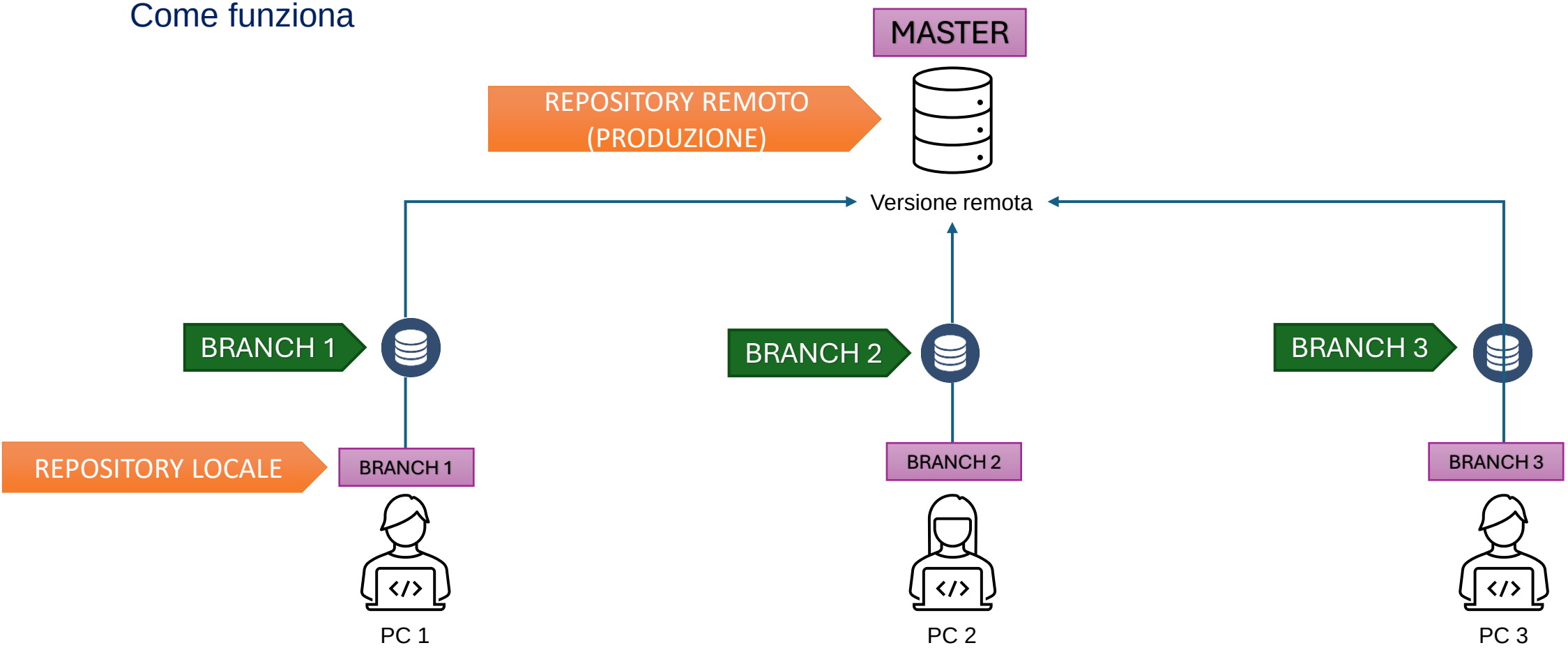
Può (anzi DEVE) essere effettuata anche l’operazione inversa, cioè "riversare" il repository **REMOTO** in quello **LOCALE**; questa operazione viene detta **PULL**

Fondamenti di Version Control – Branches

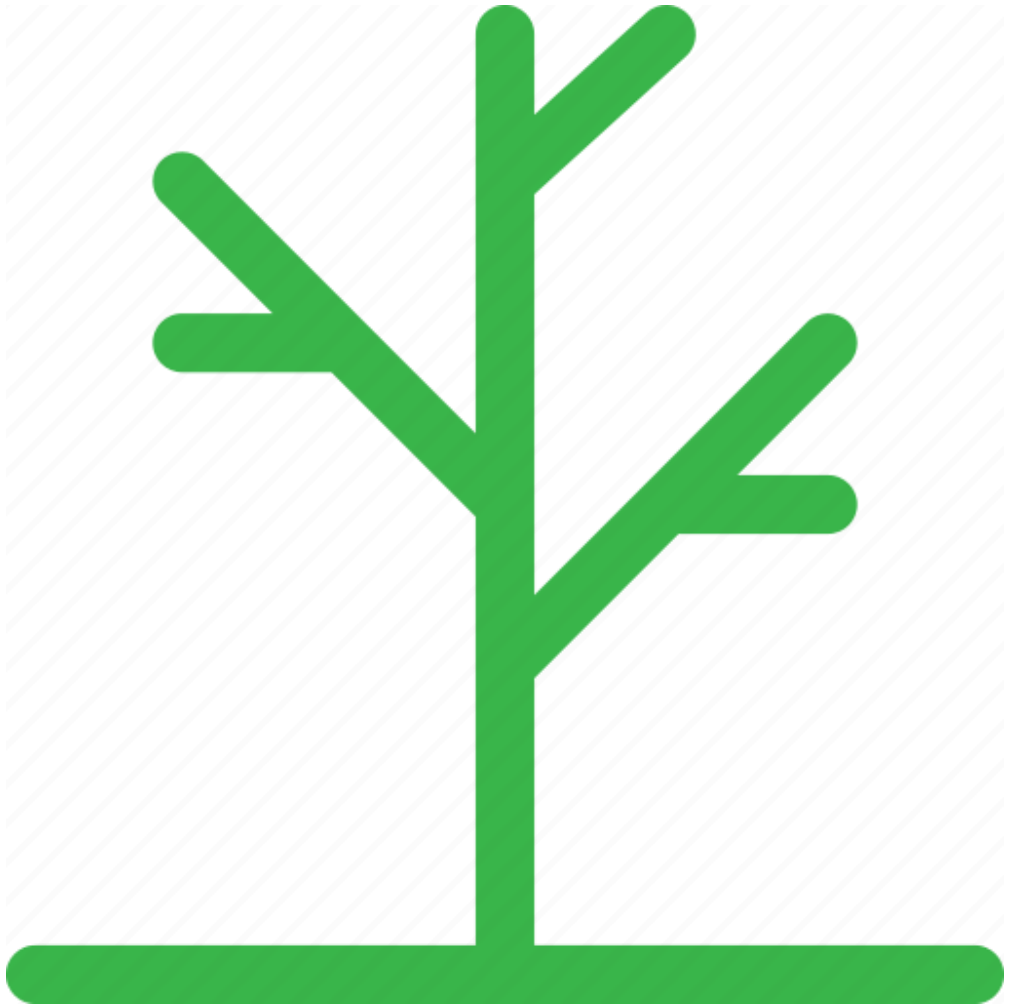
Git introduce il concetto dei **branches** (letteralmente "rami") che permettono di creare delle **ramificazioni del progetto**

Fondamenti di Version Control – Branches

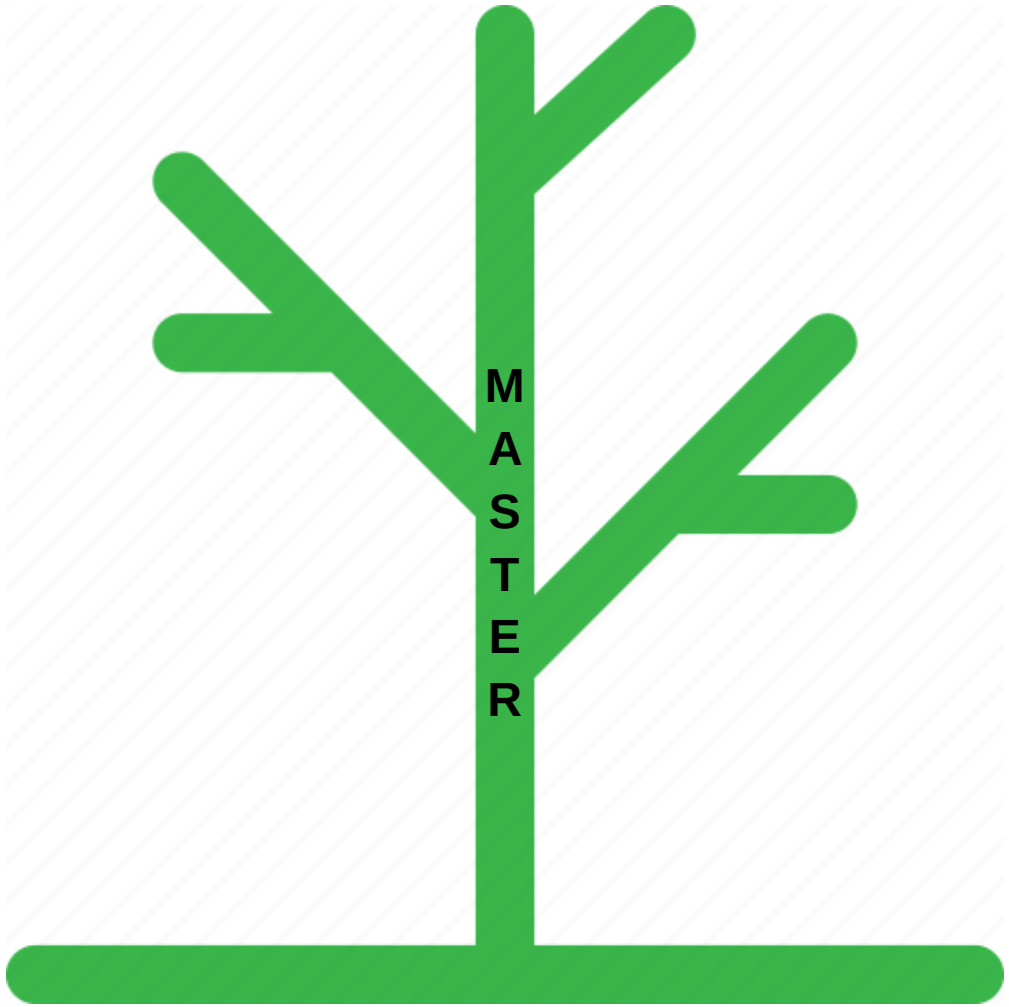
Come funziona



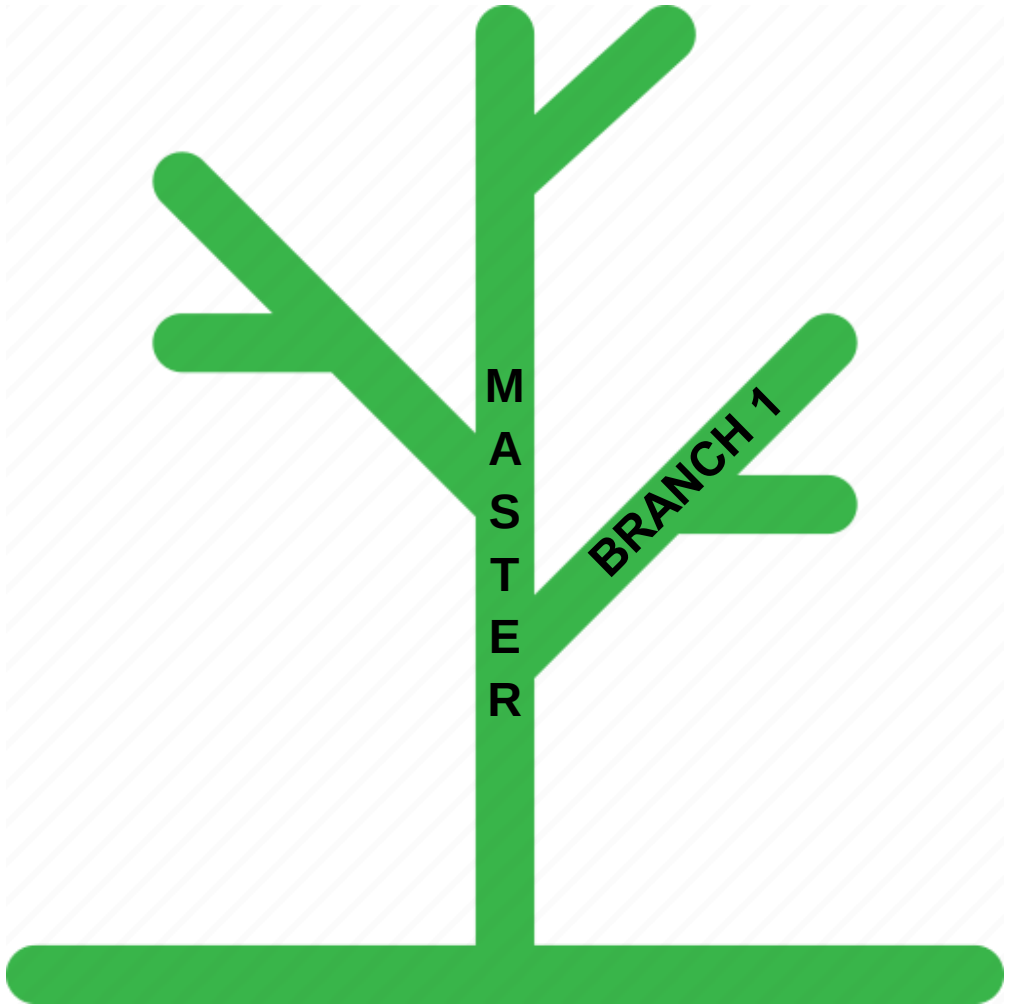
Fondamenti di Version Control – Branches



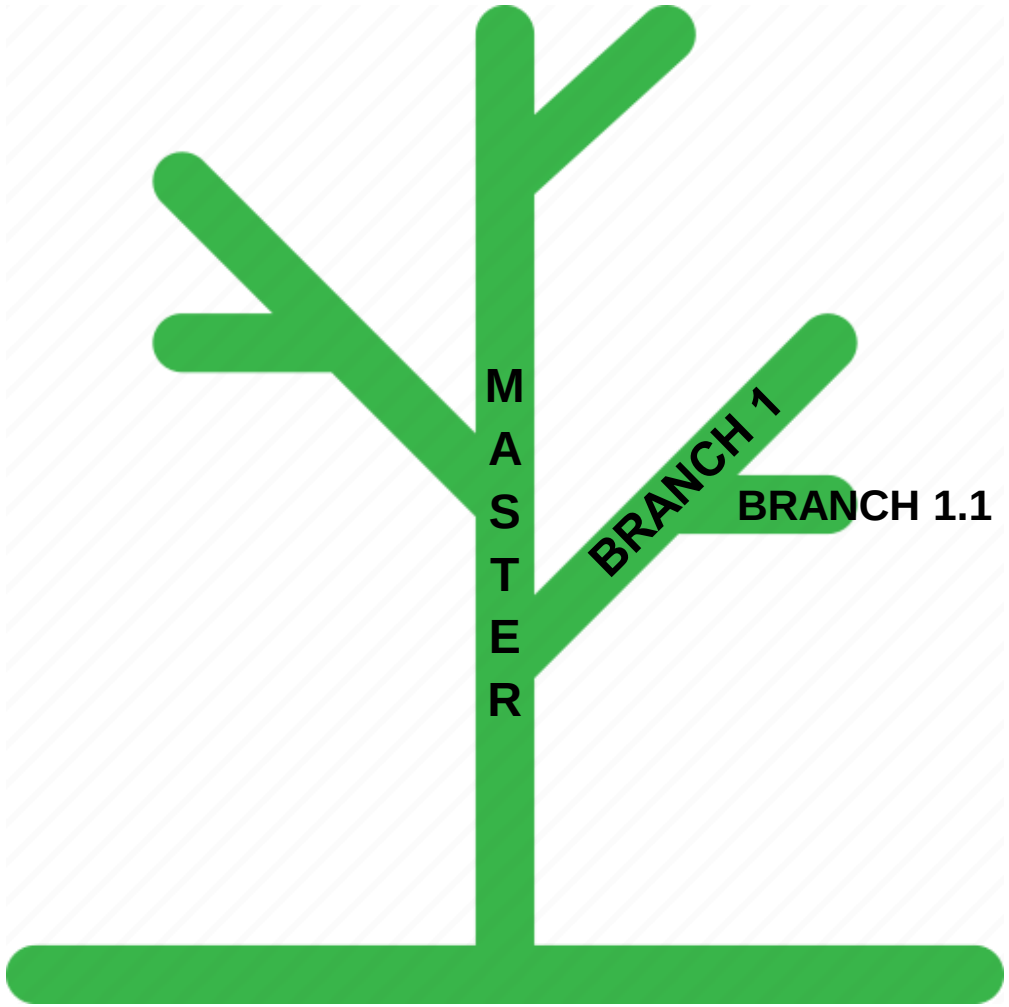
Fondamenti di Version Control – Branches



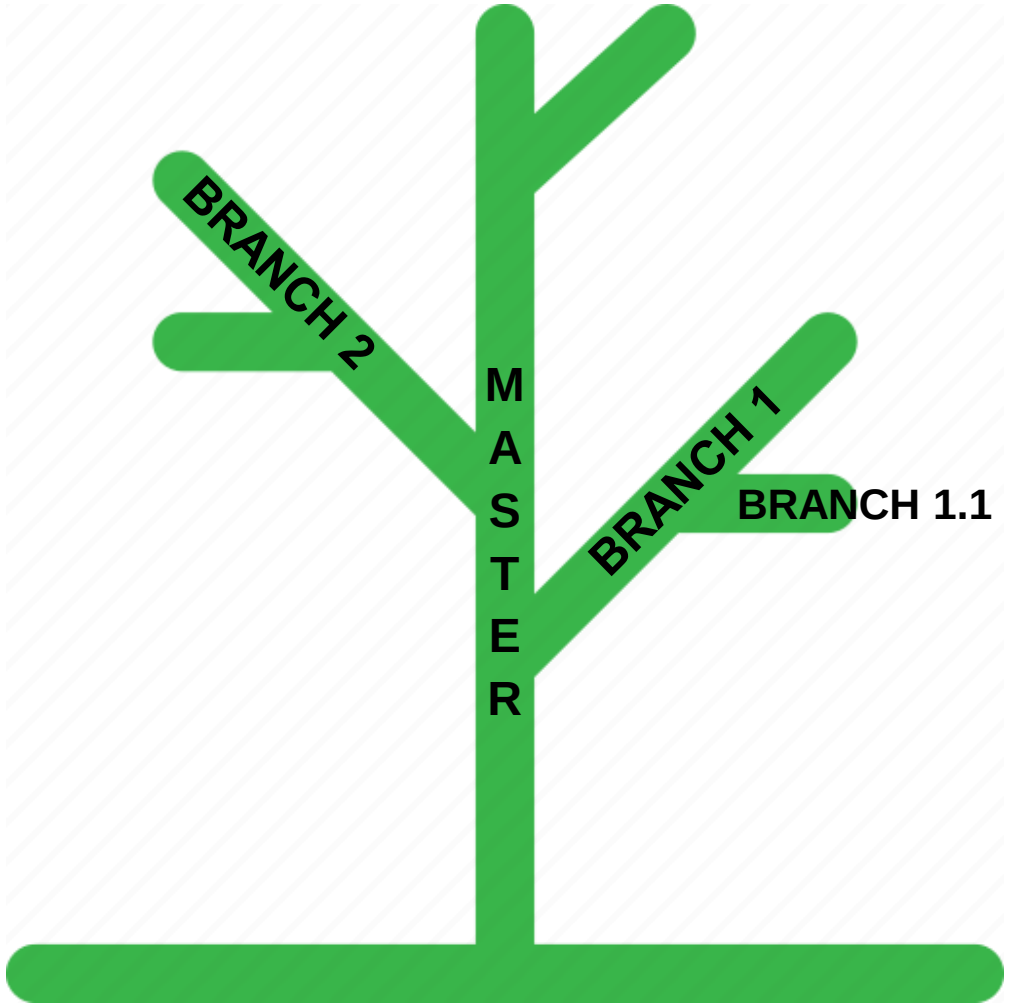
Fondamenti di Version Control – Branches



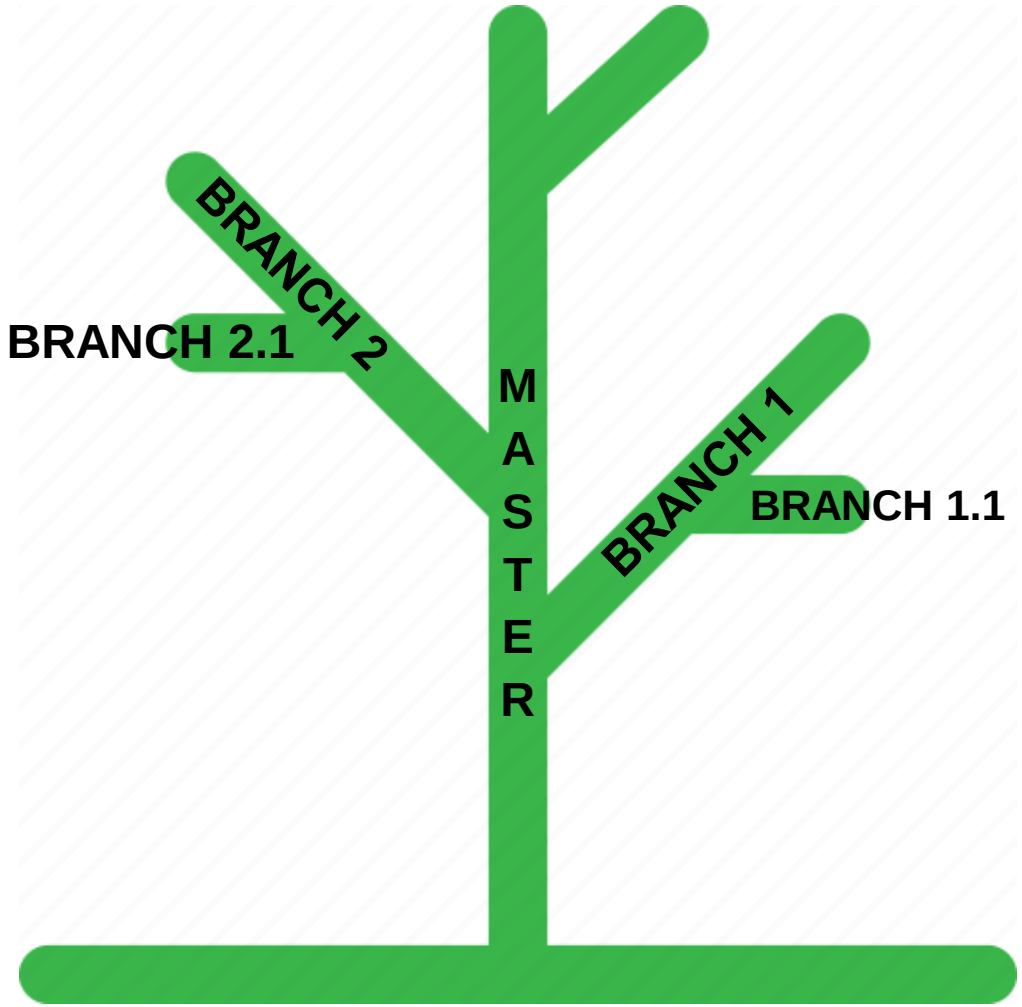
Fondamenti di Version Control – Branches



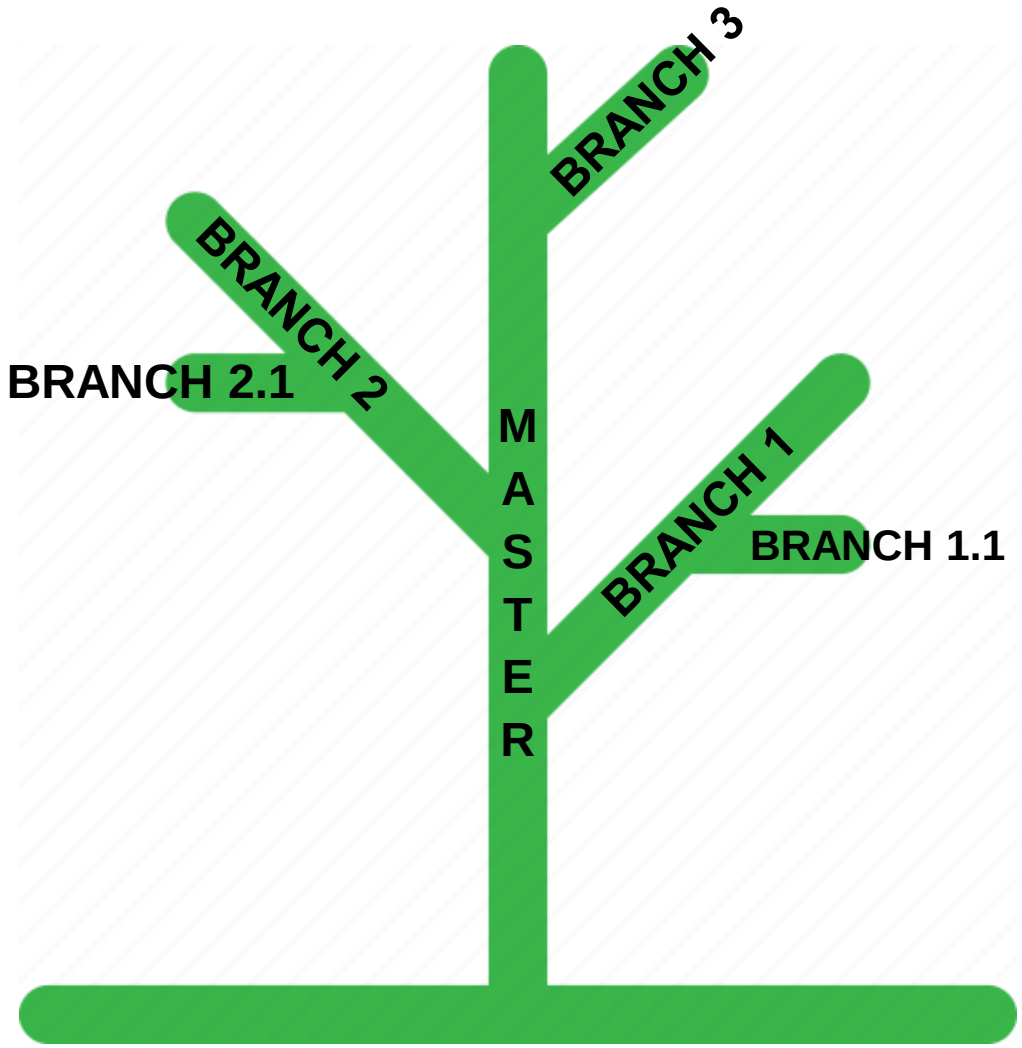
Fondamenti di Version Control – Branches



Fondamenti di Version Control – Branches



Fondamenti di Version Control – Branches



Fondamenti di Version Control – Branches

Git introduce il concetto dei **branches** (letteralmente "rami") che permettono di creare delle **ramificazioni del progetto**

La struttura va vista proprio come un albero

- Le radici sono il master
- Dal master si diramano i branches, 1 per ogni sviluppo (sviluppatore) diverso
- Capita che si crei un branch da un branch e non da master, ma non è la norma

Glossario

- **Repository Remoto**: "contenitore ufficiale" del codice del progetto, normalmente situato su un server aziendale o remoto
- **Repository Locale**: "contenitore ufficioso" del codice del progetto, esiste sulla macchina su cui si sviluppa e viene normalmente creato clonando (vedi sotto) dal repository remoto
- **Clone**: operazione con cui viene creato un repository locale facendo una sorta di "copia-incolla" dal repository remoto; questo permette di avere il collegamento tra repository locale e remoto senza dover fare altri passaggi (e quindi poter fare push, pull, eccetera)
- **Branch**: letteralmente "ramo", in effetti è una diramazione del master. Se si immagina il master come il tronco di un albero, ogni branch diventa un ramo che parte da quel tronco e si sviluppa

Link utili

Download git: https://git-scm.com/downloads

Github: https://github.com/