Cosa sono GIT e il Version Control?

Git è **un** sistema di **controllo versione distribuito**, progettato per gestire i **cambiamenti** nel codice sorgente di un progetto software.

Consente a più persone di lavorare contemporaneamente sugli stessi file, tenendo traccia delle modifiche apportate nel tempo.

Git **memorizza le versioni del codice** in modo **efficiente** e consente di **tornare indietro** a versioni precedenti o di confrontare le modifiche effettuate.

È ampiamente utilizzato nello sviluppo di software per collaborare, gestire le modifiche e mantenere un flusso di lavoro organizzato tra i membri del team.

Repository

- Una repository (o repo) è un **luogo** in cui vengono conservati, **archiviati** e organizzati **i file di un progetto**.
- Può essere considerata come una cartella virtuale che contiene tutti i documenti, il codice sorgente e gli asset di un progetto.
- Una repository può essere pubblica, accessibile a tutti, o privata, accessibile solo ai membri del team autorizzati.

GIT vs GITHUB

GIT

- Sistema di controllo versione distribuito
- Gestisce i cambiamenti nel codice sorgente
- Memorizza le versioni del codice in modo efficiente
- Consente di lavorare offline
- Utilizzato per tenere traccia delle modifiche localmente

GITHUB

- Sito web basato su Git
- Servizio di hosting per repository Git
- Permette di condividere e collaborare sui progetti
- Offre funzionalità di social coding
- Consente di lavorare online e collaborare con altri

GIT vs GITHUB

In sintesi

- Git è uno **strumento** di version control utilizzato **localmente**
- GitHub è un **sito web basato su git** che consente di condividere repository
- Git e GitHub sono strumenti complementari
- Git è la tecnologia sottostante, mentre GitHub offre una piattaforma online per sfruttarne le potenzialità

Principali comandi di git

Tutti i comandi devono essere preceduti dal comando “git”

- clone
  - Clona (scarica) una repository originariamente non situata nella nostra macchina in una cartella di nostra scelta
- add
  - Informa GIT dei file di cui tenere traccia
- commit
  - Salva i cambiamenti effettuati ai file di cui si tiene traccia
- push
  - Esegue l’upload **dei commit** da una repository **locale** ad una **remota** (es. su Github, Gitlab, Bitbucket)
- pull
  - Esegue il download **dei commit** da una repository **remota** ad una **locale** (l’opposto di push)

DEMO: Primi Passi con GITHUB

- Registrazione a Github
- Esplorazione delle principali aree di Github: dashboard, pagina personale, repository
- Creazione di una repository
- Modifica di una repository

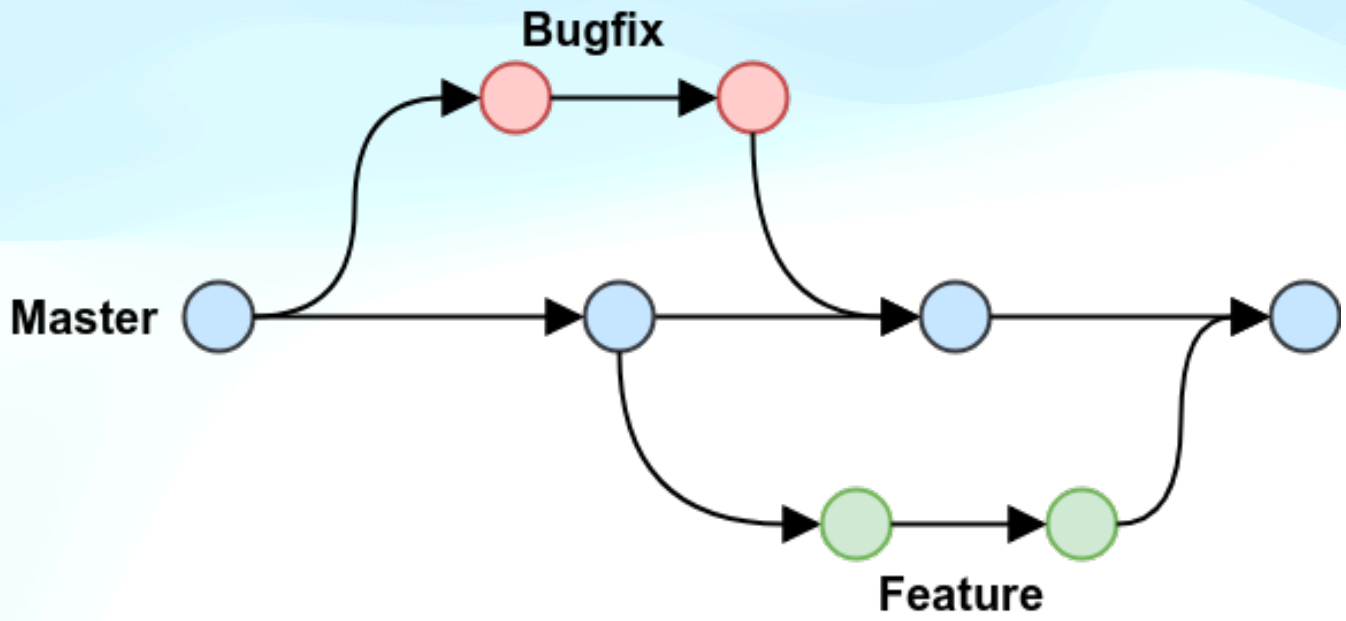
Installazione di GIT

https://www.atlassian.com/git/tutorials/install-git

Demo: Gestione chiavi SSH

https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent

Branching



Branching

I branch (rami) in Git sono versioni separate del codice sorgente che permettono di **lavorare** su nuove funzionalità o correzioni di bug **in modo isolato**.

Servono a separare lo sviluppo di diverse caratteristiche o miglioramenti, senza influenzare il codice principale.

Ogni branch ha la sua linea di sviluppo indipendente, permettendo di sperimentare, testare e iterare sulle modifiche senza alterare direttamente il codice nella branch principale (solitamente chiamata "master" o “main").

Una volta che il lavoro su un branch è completo e testato, può essere unito (merged) nella branch principale, incorporando così le modifiche nel codice principale.

Repository

Tutti i comandi devono essere preceduti dal comando “git”

- **branch**
  - Elenca i branch
- **checkout**
  - Cambia branch
- **checkout -b <nome del branch>:**
  - Crea e cambia il branch
- **merge <nome del branch>**
  - Unisce il branch specificato in quello corrente
- **git diff <nome del branch>**
  - Mostra le differenze fra due branch
- **branch -d <nome del branch>**
  - Cancella un branch

Annullare i commit

- Reset
  - Riporta indietro il puntatore dei commit ad un commit presente
- Reset —hard
  - Riporta indietro i file e le cartelle ad un punto precedente

Demo: Reset e Forking