Git and github

Version Control System

Il sistema di controllo delle versioni è un software che aiuta gli sviluppatori a lavorare insieme e a mantenere una cronologia completa delle attività svolte.

Di seguito sono elencati gli aspetti più importanti del software di controllo della versione:

- Consente agli sviluppatori di lavorare insieme;
- Evita che uno sviluppatore sovrascriva le modifiche di un’altro;
- Mantiene la cronologia di ogni versione.

Distributed Version Control System

- I sistemi di controllo della versione centralizzati usano un server centrale per mantenere la cronologia delle versioni.
- Il punto di debolezza di questi sistemi è il fatto che c'è un singolo punto e in caso di problemi nessuno può lavorare.
- Nei sistemi di controllo della versione distribuiti non solo i clients aggiornano l'albero delle modifche, ma fungono anche come repository per il backup del codice.
- Se il server smette di funzionare ogni client può aggiornare il server con la sua versione

Vantaggi di Git

Free and open source

- Git è rilasciato con una licenza open source.
- E’ disponibile gratuitamente su Internet
- Essendo open source puoi scaricare il codice ed eventualmente modificarlo per adattarlo alle tue esigenze.

Veloce e leggero

Molte delle operazioni sono compiute in locale, e questo consente a Git di essere performante e veloce.

La dimensione dei dati è contenuta, grazie a mecanismi di compressione che lo rendono efficiente.

La possibilità di perdere dati è veramente rara, poichè vengono mantenute copie multiple dello stesso archivio.

La gestione dei branch è molto semplice. E’ intuitivo crearne nuovi, cancellarli o fondere insieme due branches.

Sicurezza

- Git usa chiavi crittografiche dette hash (SHA1), per identificare gli oggetti del database.
- Ogni file e ogni commit è controllato e recuperato tramite il checksum dell'hash.
- Questo significa che è impossibile cambiare file, date o messaggi senza conoscere molto bene Git.
- Gli sviluppatori fanno cambiamenti nel loro spazio di lavoro e solo dopo il commit, questi cambiamenti entrano a far parte del repository.

Commits

- Il comando Commit aggiorna lo stato del repository
- Ogni commit è rappresentato con un hash (SHA1).
- Il commit può essere considerato come un nodo in una lista di nodi collegati fra loro.
- Ogni commit ha un riferimento al commit padre.
- Se un commit ha più di un genitore, allora significa che è stato creato dal merge di due branches

Branches

- Branches (rami) sono usati per creare linee parallele di sviluppo.
- Di default, Git ha un ramo chiamato master.
- Di solito un nuovo branch è usato per lavorare su una nuova funzionalità del software.
- Una volta che la nuova funzionalità è stata completata, viene fusa con il branch master e può essere cancellata.

Clone

- Clonare è l’operazione di copia dell’intero repository.
- Clone serve a sincronizzare l’archivio locale con quello remoto o per creare localmente una copia del repository.

Pull

- Pull è l’operazione che copia i cambiamenti da una versione remota del repository ad una locale.
- Pull è utilizzato per sincronizzare due repository.

Push

- Push è l’operazione che copia le modifiche locali del repository verso una copia remota.

https://git-scm.com/

Risorse

- https://git-scm.com/doc
- https://git-scm.com/video/what-is-version-control
- https://git-scm.com/video/what-is-git
- https://www.youtube.com/watch?v=mVnZVw4KJnc

I comandi - Local (1)

- git init «cartellagit»
  - (inizializza la cartella e la rende disponibile come repository)
- git status
  - (visualizza lo stato del repository)
- git add .|*
  - (aggiunge i files al traking - staging)
- git commit -m «primo commit del file»
  - (aggiunge il file al repository)
- git log
  - (mostra l’elenco dei commit)

I comandi – Local (2)

- git branch (-a)
  - (visualizza i branches)
- git checkout -b (nome branch alternativo)
  - (crea e attiva un nuovo branch)
- git checkout «master»
  - (attiva il branch master)
- git merge (nome branch alternativo)
  - (fonde il branch_alternativo in master)

I comandi – remote (1)

- git remote add origin + «url»
  - (crea una connessione con github)
- git push –u origin master
  - (copia il contenuto locale su origin/master)
- git remote
  - (visualizza le connessioni con github)
- git branch -r
  - (mostra i branches abbinati su github)

I comandi – remote (2)

- git clone + «url»
  - (copia da github e crea un repository locale)
- git push
  - (aggiorna il contenuto locale su origin/master)
- git fetch
  - (visualizza commit disponibili su github)
- git pull
  - (esegue fetch + merge ed aggiorna il repository locale da github)