**GIT**

Git è un software di **controllo versione distribuito (distributed version control**) utilizzabile da interfaccia a riga di comando. Permette di salvare in locale e in remoto su un server, più versioni di uno o più file e progetti.

Un sistema di **controllo versione distribuito** o **decentralizzato** (o DVCS da Distributed Version Control System) permette di tenere traccia delle modifiche e delle versioni apportate a dei file o progetti, senza la necessità di dover utilizzare un server centrale. La definizione di sistema **distribuito** deriva dalla possibilità di avere i file sia in locale che in remoto, quindi nel caso in cui il server che immagazzina i file dovesse essere compromesso con la conseguente perdita dei progetti salvati, non si ha la perdita di questi ultimi vista la loro permanenza nel sistema locale. Con questo sistema gli sviluppatori possono collaborare individualmente e parallelamente su un dato **deposito** dati di sviluppo (detto repository), registrare le proprie modifiche (mediante validazione – commit) ed in seguito condividerle o unirle (merge) a quelle di altri.

Questo sistema permette diverse modalità di collaborazione, proprio perché il server è soltanto un mero strumento d'appoggio.

In breve git pemette

* Lavorare in team su uno stesso progetto in remoto
* Funzionalità di backup
* Confronto modifiche tra versioni differenti tra file e progetti
* Risalire a vecchi progetti nel caso le nuove versioni siano corrotte

Per gestire i vari depositi ed effettuare le relative funzioni che offre Git, è necessario utilizzare l’apposita finestra di comando, detto **bash**.

Per aprire il bash di git in Windows si va nella cartella interessata e si clicca “tasto destro mouse” sulla cartella (o nella cartella aperta in un punto qualsiasi) e si seleziona “git bash HERE”, il quale aprirà il command line di Git. Su sistema UNIX e Linux invece il bash di git è incluso direttamente nel terminale, quindi direttamente da questo è possibile eseguire i vari comandi nella cartella desiderata.

La cosa positiva nell’utilizzo di questo tipo di sistema è appunto accedere a più versioni differenti di un file senza dover adottare dei nomi particolari che distinguano le sue varie versioni. Se ad esempio avessimo un file e volessimo distinguere le varie versioni realizzate, si potrebbe mettere in coda al nome una sigla relativa al numero della versione, come per esempio filev1, filev1.2, filev2, filev2.2, final\_file, etc….Risulta ovvio che quando si andranno a gestire molte versioni di file differenti, magari appartenenti ad un unico progetto, quest’approccio può risultare frustrante e confusionario, specialmente se i file diversi interagiscono tra loro. Git permette invece di creare più depositi relativi alle varie versione del progetto, evitando quindi ulteriore lavoro per collegare i file di nome differente. Nel caso in cui stessimo creando una libreria da utilizzare in progetto realizzato in C, ogni qualvolta che si crea una versione della libreria, con il vecchio approccio sarebbe necessario cambiare il nome della libreria nella direttiva #include, con Git questa scocciatura non si presenta in quanto i nomi dei file saranno sempre gli stessi e per passare da una versione all’altra della libreria sarà sufficiente passare nella reposy desiderata. Inoltre nel caso in cui delle modifiche ad un file sorgente dovessero compromettere l’integrità del programma in modo irreversibile, sarà sufficiente tornare all’ultima versione funzionante salvata.

Questo tutorial si basa sulla playlist del canale <https://www.youtube.com/channel/UCJbPGzawDH1njbqV-D5HqKw>.

Il link del primo video sulla playlist è : https://www.youtube.com/watch?v=cEGIFZDyszA&list=PL6gx4Cwl9DGAKWClAD\_iKpNC0bGHxGhcx