TP Git

Nathalie Morvan - 16 février 2017

Git est un logiciel de gestion de versions. Il est utilisé par les équipes de développements pour historiser leurs programmes source, sauvegarder ces programmes, et les partager facilement.

Git est un système de gestion de versions distribué. Chaque client possède tout l'historique des versions, comme le serveur.

SVN est un système de gestion de versions centralisé, où seul le serveur dispose de tout l'historique des fichiers.

GitHub est un site web permettant à des développeurs de partager leurs projets open-source, par exemple.

Question 1 – Installation

Installer Git sur votre machine, en laissant les paramètres par défaut.

https://git-for-windows.github.io/

Lancer « Git bash ». C'est une émulation bash permettant de lancer les commandes Git en mode commande.

Question 2 – Configuration

Lancer les deux commandes suivantes, en remplaçant *pseudo* et *votreemail*, ces valeurs seront ensuite utilisées pour créer votre compte GitHub :

git-config --global user.name "pseudo"

git-config --global user.email "votreemail"

Question 3 – Mon premier *repository* 

Créer un répertoire monPremierRepo. Ce sera votre « repository », autrement dit la racine de votre projet. Vous pouvez créer autant de « repository » que vous le souhaitez.

Une fois dans votre « repository », lancer la commande « git intit ». Cela va créer un répertoire .git et initialiser l'index de vos versions.

Toujours dans ce répertoire, créer un fichier.

Lancer la commande « git add . » pour ajouter tous les fichiers de votre répertoire à l'index git.

Puis lancer « git commit -m "mon premier commit" ». Vous venez de créer la 1ère version de votre fichier, dans le « repository ». Avant le « commit », les modifications sont dans un état « staged », elles sont dans la « staging area ».

La commande « git commit -a -m "mon premier commit" » permet de *commiter* des modifications sur des fichiers déjà indexés.

Tester les commandes « git log » et « git status ».

Modifier votre fichier, *commiter* la modification et relancer « git log ».

Question 4 – GitHub 

Pour sauvegarder ou partager votre code, vous pouvez avoir besoin d'un *remote repository*. Vous pouvez en créer un sur le site GitHub par exemple.

Pour cela, vous vous créer un compte avec les identifiants utilisés à la question 2.

On est dans le cas où le *repository* existe déjà en local. On va juste en créer vide un sur GitHub, puis y importer notre projet local.

Parenthèse :

Dans le cas où vous n'avez aucun projet en local, et que vous souhaitez importer un *repository* GitHub, récupérer l'URL du *repository* en question, et en local sur votre machine, dans le répertoire correspondant à votre future copie du projet, lancer la commande :

« git clone URLRecupereDansGitHub »

Sur GitHub, aller sur https://github.com/orgs/PPEDolto/teams

Choisir l'équipe Couloir, Fenetre ou Fond.

Dans votre équipe, créer un nouveau *repository* sans init (case décochée). Récupérer l'URL de ce repository.

Question 5 - Push 

Puis lancer les commandes suivantes sur votre machine, depuis votre *repository* local :

git remote add origin https://github.com/xxxx

git push -u origin master

Cela associe le nom *origin* à votre *remote repository*, puis y importe le contenu de votre repository local.

Revenez sur GitHub, votre *remote repository* s'est rempli. Vous pouvez voir l'historique des versions *commitées*.

Question 6 – Pull 

Faire une modification d'un fichier sur le *remote repository* puis lancer la commande suivante pour synchroniser votre *repository* local :

git pull origin master

Observer que la modification a été récupérée localement.

Conclusion

Ces questions vous ont permis d'appréhender les notions de base autour de Git. Il reste à creuser les commandes relatives à l'historique, à la gestion de conflit, à la création et à la fusion de branche.