Git + Github = versioning en ligne

Benoît Charroux

1. Introduction

Le but de ce TP est d'utiliser un système de versioning de code (git) conjointement avec un gestionnaire de dépôt de code en ligne (Github). Github (https://github.com/) peut héberger gratuitement le code de tout projet Open source.

Pour utiliser Github, il faut créer un compte. Vous pouvez créer un compte par équipe de projet mais vous pouvez aussi avoir un compte personnel (recommandé) afin de pouvoir participer à plusieurs projets.

Le principe de git étant que chaque développeur a sur sa machine une copie (ou plusieurs si on utilise le principe des branches) du code présent sur le serveur, il faut aussi disposer sur sa machine d'un client git. Git peut être installé sur Windows, Linux ou Mac. Sur Windows il y a une version graphique mais il n'est pas recommandé de s'en servir car la version ligne de commande propose les mêmes commandes quel que soit le système. Ces commandes sont d'inspiration Unix (utiliser « ls » par exemple pour obtenir la liste des fichiers du dossier courant). Avant d'installer un client git sur votre machine, tester la présence de git via la commande git. Si git n'est pas installé suivez la procédure d'installation « Set up Git » : https://help.github.com/articles/set-up-git/#platform-all

2. Création d'une application sur votre machine et envoyer là vers le serveur

Dans le langage que vous voulez (Java, C…) créez une application. Votre application peut avoir plusieurs dossiers (packages Java…). Compilez (si le langage le permet) et testez l'application.

Comment pouvez-vous envoyer votre application sur Github ?

Indications : utiliser les commandes git init, git add, git rm, git mv, git commit, git fetch, git pull et git push.

3. Travail de codage

Comment faire des modifications du code dans votre application avec la possibilité de revenir en arrière (annuler des modifications) ?

Indications : utilisez les commandes git commit, git reset et git log.

4. Tester des modifications de code sans affecter le code existant

Une pratique courante en codage consiste à vouloir tester des modifications de code sans prendre le risque de modifier le code déjà écrit.

Comment tester l'ajout des nouvelles fonctionnalités à votre application ?

Indications : utiliser les commandes git branch, git checkout et git merge.

Comment intégrer des modifications de code dans le code existant ? Que se passe-t-il si une partie du code que vous avez modifié affecte le code original ? Comment résoudre ces conflits ?

Indications : utiliser les commandes git merge et git push.

5. Comment participer à un projet ?

Quels sont les principes vus ci-dessus que vous pouvez utiliser pour coder à plusieurs sur le même projet ? Comment un développeur qui vient de terminer une modification de code peut-il demander l'intégration de son code ?

Indications : commencer par faire un fork pour récupérer une copie séparée de l'original, puis utiliser les commandes git clone, git remote, git fetch, git merge et git push.