TP Git

Author : Cedric Dumoulin

Date : 2 déc. 2014
Rev : 15 Sep. 2016

Objectifs

L'objectif de cet atelier est d'apprendre à utiliser GIT pour versionner ses projet et pour travailler à plusieurs.
Vous allez commencé par apprendre à travailler sur un dépôt local (clone d'un dépôt partagé), puis vous allez travailler sur un dépôt partagé.

Le livre suivant, en français, décrit comment utiliser GIT. Lisez les premiers chapitres si vous n'avez pas bien compris le cours, ou si vous voulez approfondir vos connaissances.
http://git-scm.com/book/fr/v1

Pour cet atelier, il est préférable de travailler dans un nouveau *workspace*.

Créer un dépôt partagé

Commencez par créer un dépôt partagé sur un hébergeur GIT. Vous pouvez par exemple créer un tel dépôt sur le site *github.com*

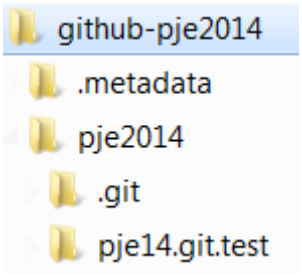
Créer un dépôt sur github.com

Vous devez d'abord créer un compte sur github, ou utiliser votre compte. Ensuite, suivez les indications pour créer un nouveau dépôt (repository).
Donnez lui le nom 'ta2015'.
Acceptez la création du 'README'.

Configurer Eclipse

Structure des répertoires

Travailler avec GIT implique de faire attention à la structure de vos répertoires. Vous ne pouvez pas utiliser les valeurs par défaut proposée par Eclipse.
La structure suivante vous donne un exemple d'organisation:



Vous devez adapter le nom du projet (ici github-pje2014) et le nom du package (ici pje14.git.test) à vos besoins.

Organisation des répertoires

Ne créez pas ces répertoire maintenant. Faites attention à leur localisation dans les différents wizards.

Le tableau suivant vous explique les répertoires:

nom	Rôle	Quand est-il créé ?
github-pje2014	workspace Eclipse	C'est vous qui le créez lors de la création du workspace Eclipse.
.metadata	Les métadonnées d'Eclipse.	Créé par Eclipse.
pje2014	workspace GIT.	Créé lors de la connexion avec le dépôt. Vous devez donner le nom est la localisation dans le wizard de connexion au dépôt.
.git	Clone du dépôt.	Créé par GIT lors de la connexion avec le dépôt.
pje14.git.test	projet java	C'est vous qui le créez en créant un projet java. Vous devez spécifiez la localisation et le nom du répertoire dans le wizard de création de projet. Le nom de répertoire doit être le nom du projet. Vous pouvez créer plusieurs projets.

Cloner le dépôt

Vous allez maintenant accéder à votre dépôt à partir d'Eclipse.
Passer en perspective GIT.
Dans la vue 'Git Repositories', cliquer sur 'clone a Git Repository'.

Dans 'Clone Git Repository', remplissez les champs.

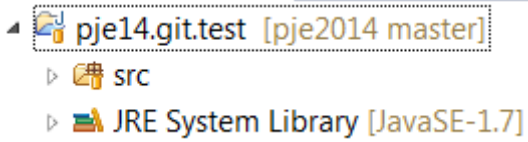
- Vous pouvez trouver l'URL de votre dépôt sur la page web de celui-ci.
- Faites attention a la localisation du workspace GIT

A cette étape, vous devez avoir une copie locale de votre dépôt. Celui-ci est visible dans la vue 'Git Repositories'.

Créer un projet Java

Changer de perspective.
Créer un projet Java.
Faites attention à la localisation de votre projet. N'acceptez pas la localisation par défaut. Ce doit être un nouveau répertoire dans le workspace GIT 'pje2014'.

A cette étape, vous devez avoir un projet contrôlé par GIT. Le nom de votre projet doit être suivie du nom du dépôt et de la branche.



Travailler en local

Exercez-vous à travailler en local. Basculez entre les perspectives GIt et JAVA pour vérifier les résultats.

Premier commit

- Créez une classe C1,
- Commit
- Vérifiez l'historique

Second Commit

- Créez une classe C2,
- Commit
- Vérifiez l'historique

Créer une Branche

- Créez une branche 'br1'
- Créez une classe C3,
- Commit
- Vérifiez l'historique, regardez bien les étiquètes !
- Créez une classe C4,
- Commit

- Vérifiez l'historique.

Vous avez maintenant une branche

Premier merge : Fast-Forward

Vous allez merger votre branche br1 dans master

- Changer de branche, aller sur master
  - Dans 'Git Repositories', double click sur la branche
- demander le merge de la branche br1
  - Comme l'historique est linéaire, GIT fait un fast-forward. Il ne faite que avancer les étiquetes locales.
- Vérifiez l'historique, regardez bien les étiquètes ! Où est br1 ? master ? origin/master ?

Second merge : vraie merge

- Basculer sur br1
- Créez une classe C5,
- Commit
- basculer sur master
- Créez une classe C6,
- Commit
- Vérifiez l'historique,
  - regardez bien les étiquètes. Vous devez voir maintenant deux branches distinctes.
  - Utilisez la toolbar de History pour bien afficher la totalité des commits de votre dépôt.
- demander le merge de la branche br1
  - Le merge s'effectue sur deux branches distinctes. GIT crée un nouveau noeud pour merger les deux branches
- Vérifiez l'historique afin de visualiser le nouveau noeud.

Rebase

A venir :-(

Travailler à plusieurs

Jusqu'ici, vous avez travaillé avec votre dépôt local. C'est ce que vous devez faire quand vous développez une nouvelle fonctionnalité. Une fois la fonctionnalité finalisée, il faut la partager dans le dépôt commun.

Créer un second clone

Pour simuler le partage, il faut créer un second clone de votre dépôt distant.

Pour cela, créez un deuxième workspace Eclipse. Vous pouvez le faire sur votre machine, ou sur le compte de votre binôme.
Voir Cloner le dépôt.

Par la suite, nous distinguerons les deux espaces de travail Eclipse par **'Eclipse1'** et **'Eclipse2'**.
Dans **Eclipse2**, vous devriez voir le projet de **Eclipse1**, ainsi que son historique.
A ce niveau, les espaces de travail de **'Eclipse1'** et **'Eclipse2'**sont différents. Si ce n'est pas le cas, c'est que vous avez fait des 'pull' à la place de 'commit'. Dans ce cas, créez une nouvelle classe dans **Eclipse1** afin d'avoir un espace de travail différent.

Pousser les modifications

Pour partager les modifications, il faut les 'pousser' (push) dans le dépôt distant.
Dans votre historique, vous avez une étiquètes 'origin/master' et une étiquète 'master'. Ces étiquètes référencent le dernier commit de la branche correspondante. La première étiquète 'origin/master' correspond au commit de 'master' dans le dépôt distant. La seconde correspond au dernier commit que vous avez fait dans la branche locale 'master'.
Le but de push est de pousser les commits que vous avez fait dans la branche locale master vers la branche distante master, puis de faire avancer l'etiquete origin/master.
Comment pousser les commit :

- Dans Eclipse1
- sélectionnez la branche master
- push
- Vérifiez l'historique;
    - ou est l'étiquète 'origin/master' ? master ?

Récupérer (tirer) les modifications

Pour récupérer les modifications du dépôts distant, il faut les tirer (pull) dans votre dépôt local.
GIT fait le pull se fait en deux étapes :

- un fetch - qui rapatrie dans votre dépot local tous les commits distant que vous n'avez pas encore.
    - Dans cette étape, origin/master peut etre modifié par GIT si des commit distants existent dans cette branche.
- un merge - qui merge la branche origin/master avec votre branche master locale
    - GIT fera un fast-forward si possible

    - Sinon, il creera un nouveau noeud merge

Ces deux etapes peuvent se faire séparément, mais GIT offre une commande faisant les deux à la fois : 'pull'.

- Allez dans Eclipse2 'Git Repository',
- sélectionnez votre dépôt, puis click-droit->pull
    - Cela doit synchroniser votre dépot local et fusionner votre branche master.
- Vérifiez l'historique;
    - ou est l'étiquète 'origin/master' ? master ?

Gérer les conflits

La plupart du temps, le merge se passes sans probleme. GIT arrive a fusionner les modifications de chacun.
La fusion se fait par fichier, ligne à ligne: GIT remplace les lignes modifiés par la nouvelle ligne.
Les conflits arrivent si vous modifiez des lignes qui ont été modifié par quelqu'un d'autre depuis votre dernière synchronisation avec le dépôt distant. Dans ce cas, GIT interrompt le merge, et vous demande de régler le conflit.
GIT indique le conflit dans le fichier concerné à l'aide de chevron '<<<<<<<<<<<< et >>>>>>>>>>>>'.
Vous devez résoudre le conflit, enlever les chevrons, et finir le processus de merge en plaçant le fichier résolu dans la *stagged area*, puis en faisant un commit.

- Dans Eclipse1 et Eclipse2, modifiez la même ligne du même fichier.
- Allez dans Eclipse1
- pousser vos modifications
- Allez dans Eclipse2
- pousser vos modifications
    - Eclipse refuse, il vous demande de faire 'pull' auparavant
- Tirez les modifications du dépôt distant
    - Le merge s'interrompt
    - réglez le conflit
- placez votre fichier résolu dans le staged area
- Dans 'Git Repository', sectionnez votre répertoire distant
- Cherchez le menu permettant de terminer le pull
- Vous avez maintenant une version locale contenant vos modifications et les modifications du serveur
- Faites un push afin pousser vos modifications sur le serveur.

- Allez dans Eclipse1
- Tirez les modifications du serveur
    - Vous devez voir les modifications effectué dans Eclipse2

Appliquer GIT

Mettez votre appli "operation entre 2 nombres" dans le dépot GIT