Abdessamad KRIOUITA

**Travaux Pratique**

**GitHub**

1 Prise en main de git

1.1 Création de compte sur GitHub :

GitHub est un service Web d’hébergement de d´dépôts distants, utilisant git. GitHub comporte des fonctionnalités  
supplémentaires destinées `a la collaboration, telles que le suivi des bugs, les demandes d’ajout de fonctionnalités, ou la  
gestion de tâches. . .A noter qu’il existe d’autres sites Web d’hébergement basées sur git, tels que GitLab, ou BitBucket. `  
Allez sur https://github.com/ et suivez les instructions pour la création et l’activation de votre compte. Sur un  
terminal, spécifiez l’adresse mail avec laquelle vous allez effectuer vos commit :  
git config -- global user. Email " votre.email@gmail .com "

1.2 Création d’un d´dépôt :Nous allons maintenant créer un dépôt distant. Sur GitHub :  
| Cliquez sur le \+" en haut `a droite de la page, puis \New repository".  
| Donnez un nom `a votre d´dépôt, par exemple PremierDepot, ainsi qu’une description. Par d´défaut, votre d´dépôt est  
public, c’est-`a-dire que tout le monde a accès `a votre code (droits de lecture). Terminez en cliquant sur \Create  
repository".  
Suivez les instructions fournies par GitHub pour créer votre copie locale du d´dépôt. Dans l’ordre :  
1. git init permet d’initialiser un d´dépôt local vide dans votre répertoire courant ;  
2. git add permet d’indexer un fichier ou un répertoire ;  
3. git commit permet de valider les modifications sur votre d´dépôt local ;  
4. git remote add <nom> <url> : permet de lier votre d´dépôt local (initie par init) `a un d´dépôt distant (url sur  
GitHub) ;  
5. git push permet d’envoyer les modifications indexées au d´dépôt distant.

1.3 Premiers commits :Vous voulez mettre sous version l’exercice 2 du TP1.  
1. Copiez l’ensemble du répertoire, y compris les exécutables du TP1, dans votre répertoire où le dépôt git a été initialisé.  
2. Utilisez les commandes git status et git diff en vous aidant de la documentation au besoin. Que constatez-vous ?  
3. Indexez l’ensemble des fichiers (y compris les exécutables), puis vérifiez l’indexation avec git status.  
4. Poussez les modifications sur votre d´dépôt distant.  
Ces ´étapes sont un minimum pour sauvegarder votre travail sur le serveur distant. Il est important de ne pas oublier de  
faire un git status `a chaque ´étape : un fichier que vous ne désiriez pas ajouter peut être ajouté par mégarde.

1.4 Supprimer des fichiers dans le d´dépôt distant :Nous avons mis l’ensemble du projet sous git. Il s’agit d’une mauvaise pratique : les exécutables et les fichiers résultant  
de la compilation (.o) ne doivent pas ^être mis sous version. Il est alors possible de supprimer des fichiers indexes en utilisant :  
git rm -- cached fichier  
1 Florent Génelard  
TP  
Git  
Attention : Ne pas oublier l’option --cached, sinon votre fichier sera aussi supprimé de votre d´dépôt local !  
Question Supprimez le/les exécutables de votre d´dépôt distant, puis commutez et poussez vers votre d´dépôt distant.  
N’oubliez pas git status pour vérifier les modifications apportées, `a chaque ´étapes.  
Une autre option, est de d´finir, `a la racine de votre d´dépôt local, un fichier nommé .gitignore. Ce fichier contient  
l’ensemble des fichiers qui ne doivent pas ^être ajoutes `a votre d´dépôt distant.  
Question En utilisant des expressions régulières, définissez un fichier .gitignore qui permette d’ignorer les fichiers  
avec l’extension .o, les fichiers temporaires, ainsi que l’exécutable dans le d´dépôt. Commettez et poussez vos modifications.  
Comment cela simplifie-t-il l’ajout de nouveaux fichiers `a votre projet ?

2 Collaboration entre différents utilisateurs

Le but de cet exercice est de vous initier a l’utilisation de git en tant qu’outil de collaboration.

2.1 Manipulation d’un d´dépôt existant :Récupérez le dépôt `a l’adresse https://github.com/fgrelard/GLGit.  
Forkez le dépôt en suivant ces étapes :  
1. Sur GitHub, cliquez sur l’icône du fork.  
2. Une copie est ajoutée sur votre espace GitHub. Clonez-la afin d’obtenir un dépôt local.  
Il s’agit d’un programme en C, dédie `a la manipulation de voies m´métaboliques et d’enzymes. L’utilisateur peut spécifier  
leur nom `a partir de la ligne de commande ou d’un fichier.

2.2 Processus de résolution de bugs :Un bug s’est glissé dans le programme après un commit. La description du bug est donnée dans la partie \Issues" du  
d´dépôt distant de fgrelard.

2.2.1 Identification du commit coupable :Votre but est d’identifier le commit ayant induit le bug.  
Question Utilisez log, diff, checkout et git afin d’identifier le commit responsable de l’apparition du bug.

2.2.2 Création d’une branche :Afin de résoudre un bug complexe ou d’ajouter des fonctionnalités, il est souvent n´nécessaire de modifier plusieurs parties  
du code. On crée donc une branche, où l’on fait tous les commits dédies `a la résolution du bug. L’idée est de maintenir une  
version stable, dans la branche master, séparée de la version en développement, qui peut contenir des bugs.  
Questions En vous aidant de la documentation :  
1. Créer une branche locale nommée fix suppression avec la commande git Branch.  
2. Poussez la branche locale sur le dépôt distant avec la commande git push.  
3. Déplacez-vous sur la branche locale avec git checkout. Les fichiers modifies sur cette branche ne seront pas  
modifies sur master.

2.2.3 Résolution du bug :On peut ensuite résoudre le bug.  
1. Modifiez le fichier concerné par le bug en le réinitialisant `a une de ses versions précédentes, en utilisant la variable  
HEAD.  
2. Commettez, puis poussez vos modifications sur la branche fix suppression.  
2 Florent Génelard TP Git.

2.2.4 Validation et suppression de la branche :1. Assurez-vous que la résolution du bug n’a pas induit d’autres bugs dans l’exécution du programme, en le testant.  
2. Vous pouvez ensuite fusionner la branche fix suppression dans master et la supprimer :

(a) déplacez-vous sur la branche master  
(b) fusionnez la branche fix suppression dans master avec git merge  
(c) supprimez la branche locale avec git branch -d  
(d) supprimez la branche distante avec git push -d

2.2.5 Pull requests :Votre correction du bug peut intéresser l’auteur original du projet, fgrelard. Sur GitHub, ouvrez une pull-request.  
Les pull-request sont un ensemble de commits qui peuvent ^être intègres directement par l’auteur du projet dans son dépôt  
distant, et sont ainsi un outil puissant pour travailler a plusieurs.

3 Fusion et conflits

Le but de cet exercice est de manipuler les fusions (merge) de commits. Une fusion peut ^être conflictuelle lorsque deux  
utilisateurs travaillent en parallèle sur les mêmes lignes du même fichier.

3.1 Importation du projet :1. Utiliser l’outil d’import de GitHub disponible `a l’adresse : https://github.com/new/import. Cet outil permet de faire l’importation d’un projet existant. Entrez l’URL suivante dans \Your old’s repository URL" : https://github.com/githubschool/on-demand-merge-conflict.git. Une copie du dépôt doit ^être faite sur faite sur  
votre espace GitHub, de façon similaire `a un fork  
2. Clonez ce dépôt en local.  
Ce ddépôt contient du code permettant d’afficher un CV en ligne.

3.2 Première fusion :Vous voulez changer le titre et la description de votre CV.

1. Créez une branche nommée mytitle  
2. Modifiez les lignes 2 et 3 du fichier config.yml en spécifiant le titre et la description voulue.  
3. Commitez et poussez vers le dépôt distant, en spécifiant le nom de la branche courante.  
4. Déplacez-vous sur la branche master  
5. Fusionnez (merge) les modifications avec la commande git merge, puis commitez et poussez.  
La branche master contient maintenant les modifications effectuées sur la branche mytitle. Supprimez la branche  
mytitle sur le dépôt local et distant (cf. exercice précédent).

3.3 Fusion conflictuelle :Vous voulez spécifier votre nom et vos informations personnelles.

1. Déplacez-vous sur la branche username-config  
2. Sur cette branche, ouvrez le fichier config.yml  
3. Les lignes 12 `a 19 doivent ^être modifiées avec vos informations personnelles. Sauvegardez vos modifications.  
4. Commitez puis poussez vos modifications.  
5. Déplacez-vous sur la branche master et fusionnez la branche username-config `a l’aide de merge.  
Un conflit apparait, puisque l’utilisateur Bob Belcher a modifié les mêmes lignes que vous sur la branche master. Ouvrez  
le fichier config.yml `a nouveau, le fichier doit se présenter de la façon suivante :  
3 Florent Génelard

TP Git  
<<<<<<< commit Bob  
# version de Bob  
=======  
# ma version  
>>>>>>>  
Modifiez le fichier pour ne garder que votre version : `a ce stade, le conflit est considère comme résolu. Enfin, commitez  
puis poussez vers votre d´dépôt distant.