TD5 Programmation C/C++
Pointeurs & Git
(Annexe : Travail collaboratif avec *GitHub*)

1. Groupez-vous par trinôme (*étudiant A*, *étudiant B* et *étudiant C*).
2. Créez chacun un compte sur *GitHub*.
3. Créez un nouveau dépôt sur l'un des compte *GitHub* : sur le compte de l'*étudiant A*.
4. Les tâches ci-dessous seront exécutées (par les 3 étudiants) sur la machine et avec le compte *GitHub* de l'*étudiant A* :
   (a) Utilisez l'option *Create a new repository* comme indiqué dans Figure 1 pour créer votre dépôt.
      i. Remplissez les informations demandées sur cette page :
         – Nom de votre dépôt
         – Description (en option)
         – Public ou Privé (pour nous ça sera 'public' : l'option 'privé' est payante)
      ii. Et une fois terminé, cliquez sur *Create Repository*.
   (b) Initialisez votre dépôt (à partir de votre terminal).
      i. Créez le dossier dans lequel vous allez travailler et initialisez un nouveau dépôt *Git* :
         `git init`
      ii. Indiquez votre adresse mail dans la config de *Git* (adresse utilisée pour créer le compte *GitHub*) :
         `git config --global user.email "vous@mail.com"`
      iii. Créez un *README* pour votre dépôt.
         Bien qu'un fichier *README* ne soit pas un élément obligatoire d'un dépôt *GitHub*, c'est une très bonne habitude à prendre que d'en prévoir un.
         `echo "#README test1" >> README.md`
      iv. Indexez l'ajout de votre fichier *README* :
         `git add README.md`
      v. Soumettez les modifications indexées en zone de transit :
         `git commit -m "first commit"`
      vi. Crée un *remote* nommé *origin* pointant votre dépôt *GitHub* :
         `git remote add origin https://github.com/mcharfi2016/test1`
      vii. Envoyez vos changements à votre dépôt distant :
         `git push -u origin master`
   (c) Ajoutez *étudiant B* et *étudiant C* en tant que collaborateurs au projet de l'*étudiant A* (Figure 2).
5. Les collaborateurs *étudiant B* et *étudiant C* recevront alors des mails pour confirmer la participation au projet de l'*étudiant A* : acceptez cette invitation.
6. Refaites la même procédure d'initialisation de dépôt chez *étudiant B* et *étudiant C* (création et configuration du dépôt *Git*, connexion au compte *GitHub*).
7. **NB.** N'oubliez pas de télécharger la version à jour du dépôt avec les commandes *fetch* et *merge* avant d'essayer d'ajouter du nouveau
8. Ajoutez-moi en tant que collaborateur (mon nom utilisateur *GitHub* est *mcharfi2016*).
9. Revenez à l'énoncé principal ...

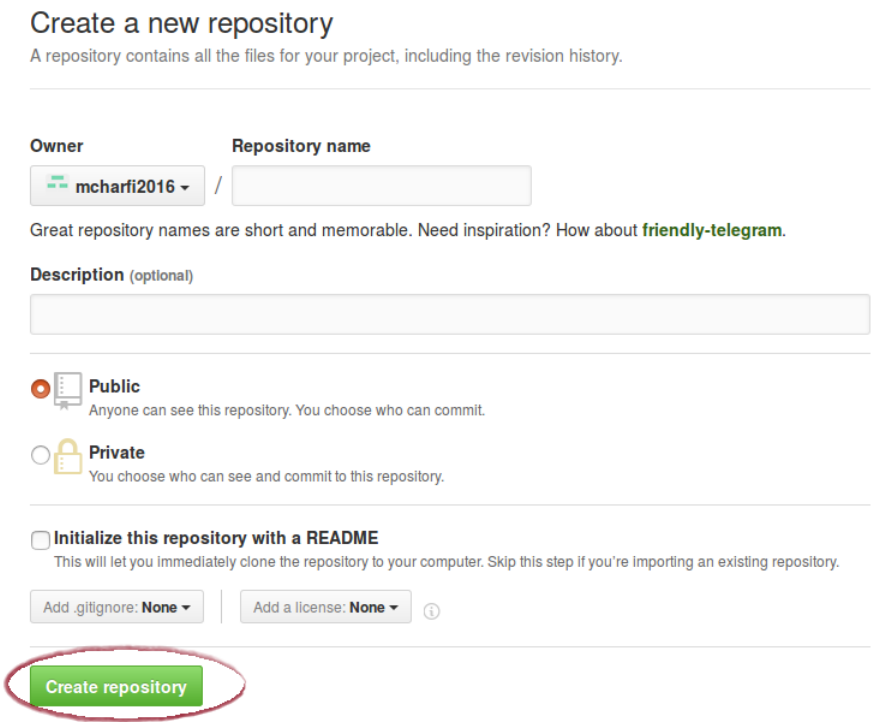

FIGURE 1 – Création d'un nouveau dépôt sur votre compte *GitHub*

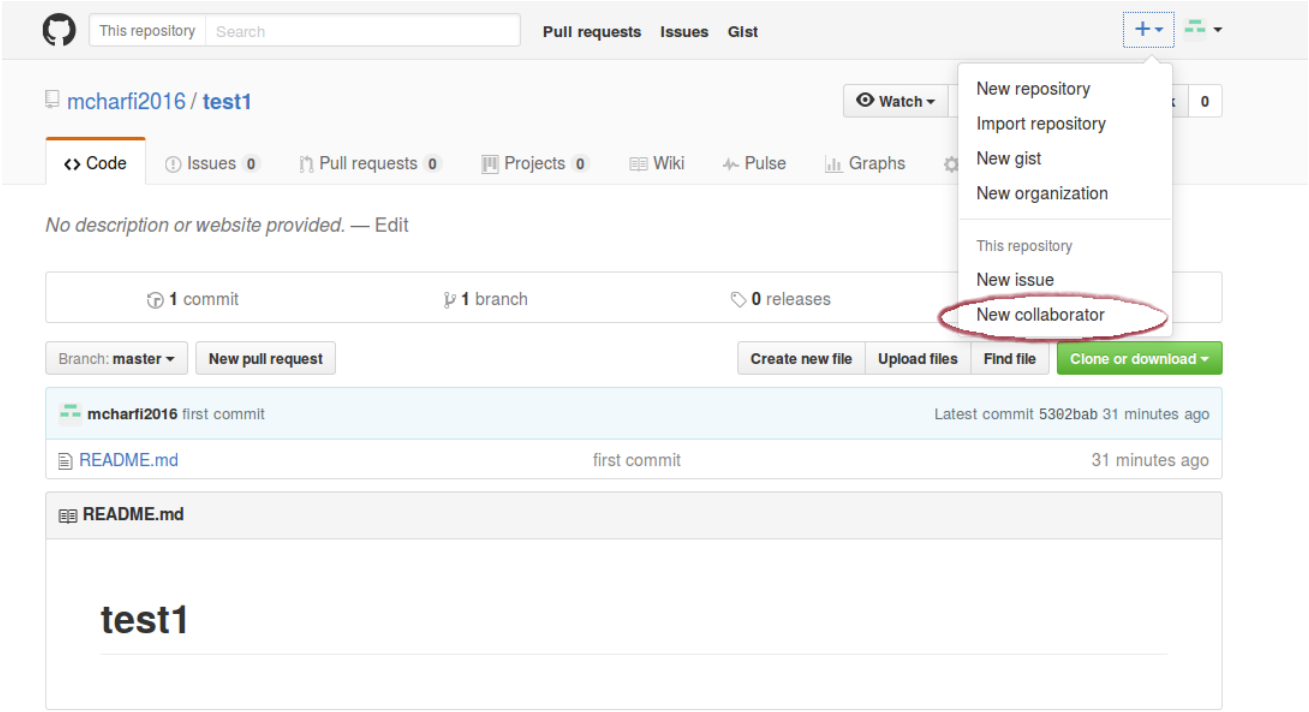

FIGURE 2 – Ajout de collaborateur au dépôt via le compte *GitHub*

Rappel quelques commandes *Git*

- **git init** : initialise un nouveau dépôt *Git*.
- **git config** : raccourci de "configurer", ceci est tout particulièrement utile quand vous paramétrez Git pour la première fois.
- **git help** : permet d'afficher les commandes les plus courantes de *Git*, on peut aussi l'utiliser avec le nom d'une commande *Git* spécifique pour voir comment l'utiliser et la configurer.
- **git status** : vérifie le statut de votre dépôt (voir quels fichiers sont à l'intérieur, quelles sont les modifications à *commiter* et sur quelle branche du dépôt vous travaillez).
- **git add [*nomFichier*]** : Ajoute un instantané du fichier, en préparation pour le suivi de version (elle n'ajoute pas de nouveaux fichiers dans votre dépôt).
- **git commit -m "[*message descriptif*]"** : enregistre les modifications apportées au dépôt.
- **git branch [*nomBranche*]** : liste toutes les branches locales dans le dépôt courant ; quand elle est suivie d'un nouveau nom elle permet de construire une nouvelle branche, ou une chronologie des *commits*, des modifications et des ajouts de fichiers qui sont complètement les vôtres.
- **git checkout [*nomBranche*]** : bascule sur la branche spécifiée et met à jour le répertoire de travail.
- **git merge [*nomBranche*]** : combine dans la branche courante l'historique de la branche spécifiée.
- **git log** : montre l'historique des versions pour la branche courante.
- **git remote add *origin url*** : récupère les données d'un dépôt déclaré.
- **git push** : envoie tous les *commits* de la branche locale vers *GitHub*.
- **git pull** : récupère tout l'historique du dépôt nommé et incorpore les modifications.

Sources

- https ://services.github.com/on-demand/downloads/fr/github-git-cheat-sheet.pdf
- http ://christopheducamp.com/2013/12/15/github-pour-nuls-partie-1/
- https ://forge.cnrs.fr/git/commandes_GIT.pdf