MELLAH Youssef 04/05/2019

ENSAO

TP : Introduction à la gestion de versions avec Git

Exercice 1 : Dépôt local

- Question 1 - Configuration initiale
  - Nous allons tout d'abord renseigner quelques informations qui seront inscrites dans vos commits.
  - git config --global --add user.name "Prénom Nom"
  - git config --global --add user.email "prenom.nom@telecomnancy.eu"
  - git config --global merge.conflictstyle diff3
  - Vérifier la prise en compte de la configuration : cat ~/.gitconfig

- Question 2 - Versionner un répertoire local
  - Créer un répertoire local : mkdir tp_git, et faites en votre répertoire courant : cd tp_git
  - Versionner ce répertoire : git init, vérifier que le répertoire contenant la base de données de git « .git » a bien été créé à la racine : ls -la .git
  - Ajouter deux fichiers texte dans le répertoire : recipe_appbread.txt, recipe_soup.txt .
  - Indexer ces deux fichiers : git add nom_fichier
  - Vérifier la prise en compte en observant l'état de l'index (zone de staging) : git status
  - Enregistrer cette version dans le dépôt : git commit
  - git ouvre alors un éditeur de texte et vous invite à renseigner le message du commit, entrer un message explicite « Ajout des recettes de soupe à l'oignon et d'apple bread ». L'option git commit -m "message" permet de tout faire en une commande.
  - Vérifier la prise en compte du commit en observant le log : git log , observer l'identifiant SHA1.

- Question 3 - Observer les modifications en cours
  - Faire une première modification dans un fichier et sauvegarder.
  - Observer les modifications perçues par git grâce à git status, git diff et git diff --cached
  - Indexer le fichier précédemment modifié : git add fichiers
  - Observer à nouveau avec les mêmes commandes afin de percevoir ce que fait chacune.
  - Commiter cette modification : git commit -m "message" et observer une dernière fois.

Exercice 2 : Annuler ou corriger des modifications

- Question 1 – Annuler une modification du working directory
  - Modifier et sauvegarder un fichier.
  - Cette modification est une erreur, rétablir le fichier tel que définit dans un commit (le dernier) :
    - git checkout SHA1_ancien_commit nom_fichier
- Question 2 – Annuler une modification indexée
  - Modifier et sauvegarder un fichier. Indexer la modification.
  - Modifier une nouvelle fois le même fichier. Malheureusement, cette nouvelle modification est une erreur.
  - Rétablir le fichier dans l'état où il a été précédemment indexé : git checkout nom_fichier
  - Finalement, vous décidez de ne pas enregistrer les modification de ce fichier pour le prochain commit, désindexer le : git reset nom_fichier
  - Vérifier l'état de l'index : git status
  - Rétablir l'ensemble du dépôt tel que définit dans le dernier commit : git reset --hard

- Question 3 – Annuler une modification enregistrée
  - Modifier et sauvegarder un fichier. Indexer puis enregistrer la modification. Vérifier la prise en compte de l'enregistrement avec git log.
  - Encore une erreur ! Annuler complètement le dernier commit : git revert SHA1_dernier_commit
  - Modifier et sauvegarder un fichier. Indexer puis enregistrer la modification.
  - Vous avez oublié de modifier le second fichier. Le modifier et indexer le changement.
  - On souhaite maintenant corriger le dernier commit pour y enregistrer les modifications du second ficher : git commit --amend (corriger si besoin le message du commit)
  - Vérifier que le dernier commit inclut bien désormais la modification du second fichier :
    - git show HEAD

Exercice 3 : Manipuler les branches

- Question 1 – Créer une nouvelle branche
  - Créer une nouvelle branche de développement : git branch experimental
  - Changer de branche pour la nouvelle : git checkout experimental
  - Lister les branches : git branch
  - Vérifier que la HEAD est bien positionnée sur la nouvelle branche : cat .git/HEAD

- Question 2 – Merger (fusionner) deux branches
  - Ajouter une ligne dans un fichier et le sauvegarder. Indexer puis enregistrer la modification.
  - Vérifier que le commit est bien enregistré : git log
  - Revenir sur la branche principale : git checkout master
  - Observer que la modification n'est plus présente dans le fichier précédemment édité ! C'est l'intérêt des branches : plusieurs versions co-existent en parallèle.
  - Vérifier que le précédent commit est absent de cette branche : git log
  - Pour intégrer les modification de la branche experimental, il faut faire un merge : ▪ git merge experimental
  - Vérifier que les modifications ont bien été intégrées
  - Supprimer la branche : git branch -d experimental

Exercice 4 : Utiliser un dépôt distant

A partir de cet exercice, il faut travailler en binôme.

- Question 1 – Créer un projet sur GitLab
  - Aller sur le GitLab et créer un compte
  - Créer un nouveau projet et choisir comme nom de projet « tp_git_votrenomdefamille »
  - Dans le menu configuration/membres de l'interface web, ajouter votre binôme en tant que développeur ainsi que MELLAH Youssef (ysfmell) (pour la validation du TP)

- Question 2 – Lier le dépôt local a un dépôt distant
  - Il faut lier votre dépôt local à votre dépôt distant sur la forge :
    - git remote add origin URL(`https://github.com/user/repo.git`)
    - « origin » est simplement ici le nom par défaut donné en local à ce dépôt distant
  - En cas d'erreur, l'adresse du dépôt distant peut être changée :
    - git remote set-url origin nouvelle_adresse
  - Lister les dépôts distants : git remote show
  - Envoyer vos enregistrements sur le dépôt distant : git push origin master
    - Si le dépôt distant n'est pas vide, il faut préalablement le merger avec le dépôt local :
      - git pull origin master
  - L'historique des commits doit maintenant apparaître sur l'interface web de la forge : sous-menu « Repository » dans le menu de gauche, section « Commits »
  - Parcourez l'ensemble des menus sous l'arborescence « Repository »

- Question 3 – Récupérer un autre dépôt distant
  - Positionnez-vous en dehors de votre dépôt local : cd ..
  - Récupérer le dépôt distant de votre binôme : git clone URL
  - Ajouter une ligne dans un fichier et le sauvegarder. Indexer puis enregistrer la modification.
  - Il faut maintenant propager les modification enregistrées localement au dépôt disant :
    - git push

- Question 4 – Récupérer les modifications enregistrées sur le dépôt distant
  - Votre binôme a enregistré des modifications sur votre dépôt distant, les récupérer :
    - git pull origin master
  - Observer l'historique des commits et vérifier que les modifications ont été prises en compte

Exercice 5 : Gérer les conflits

- Question 1 – Générer un conflit
  - Créer un conflit en créant une branche « conflit » et en enregistrant deux modifications différentes d'une même ligne d'un fichier, l'une sur la branche master, l'autre sur celle nouvellement créée.
  - Se replacer sur la branche master et fusionner les branches : git merge doit alors vous notifier que des conflits sont apparus

- Question 2 – Résoudre un conflit
  - Ouvrir le (ou les) fichier(s) incriminé(s), les marqueurs de conflit <<<<< ||||| >>>>> doivent être présents
  - Résoudre le problème en choisissant pour chaque conflit la modification pertinente ou en proposant une nouvelle modification. Ne pas oublier de supprimer les symboles marqueurs de conflit. Indexer le fichier.
  - La résolution du conflit et la validation du merge se fait en enregistrant les modifications. : git commit