

Exercice 1 : prise en main de GIT

L'objectif de cet exercice est d'apprendre à manipuler les fonctions de bases de git telles que `init`, `add`, `commit`, `status`, `log`, `revert`.

Pour cela nous travaillerons en ligne de commande, en utilisant directement un terminal si vous êtes sous Unix / Linux ou l'outil "GIT BASH" que l'on peut télécharger ici : <https://gitforwindows.org/>

1. Nous allons commencer par créer un dépôt.

Pour cela :

- a. Créer un répertoire « **tp-git_votre-nom** » (il est important d'inclure votre nom pour que le nom du répertoire vous soit propre), et y inclure un fichier texte "info.txt" contenant vos noms, prénom et adresse.
- b. Se placer dans ce dossier via le terminal et lancer la commande « `git init` ». Vérifier qu'un dossier caché ".git" qui a été créé (via la commande « `ls -a` »)

2. La commande « `git status` » permet de dresser l'état des lieux de votre projet.

Que constatez-vous à ce stade ?

3. Nous allons à présent voir comment ajouter un fichier au projet.

Pour cela :

- a. Indexez le fichier « info.txt » via la commande "`git add info.txt`" et regarder les changements apportés via la commande "`git status`".
- b. Enregistrez le document dans le dépôt via la commande "`git commit`". Cela aura pour effet de vous ouvrir un éditeur de texte vous invitant à décrire le commit. En pratique, on peut directement le faire via l'option "`-m`".
- c. On peut alors vérifier que notre répertoire local est bien synchronisé via la commande « `git status` ».

4. Nous allons à présent voir comment apporter une modification à un fichier du dépôt.

Pour cela :

- a. Ajouter une information à votre profil (e.g., votre âge) en éditant le fichier « `info.txt` » et appelez la commande « `git status` » pour voir les changements apportés.
- b. Avant de soumettre les modifications, utiliser la commande « `git diff` » pour vérifier les modifications apportées.
- c. Enfin, enregistrez le changement sur le dépôt via « `git add` » puis « `git commit` » (suivre les instructions données par « `git status` »). Notez qu'en pratique on peut faire ça d'un seul coup via la commande « `git commit -a` » (qui ajoute et commit en même temps donc).

5. Appelez la commande "**git log**" pour voir l'historique des opérations.

6. Nous allons maintenant voir comment restaurer un fichier à une version antérieure.

Pour cela :

- a. changez votre date de naissance dans le fichier pour qu'il contienne une erreur, et enregistrez le sur le dépôt.
- b. Appelez la commande "**git log**" pour voir l'ensemble de l'historique.
- c. Revenez à la dernière version correcte (ici, celle du 2^{ème} commit) grâce à la commande « **git checkout COM_ID info.txt** » où « **COM_ID** » est l'identifiant du commit voulu (qui commence ici par 457cc9, en pratique, prendre les 6 premiers caractères est suffisant). Vérifier que c'est bien le cas, i.e., que le fichier est dans l'état attendu.
- d. Appeler la commande « **git status** » pour faire l'état des lieux. Que constatez-vous ?
- e. Enregistrer la modification sur le dépôt, et appelez la fonction « **git log** » pour voir l'historique du dépôt.

Pour aller plus loin :

1. Ajouter un autre fichier au dépôt.
2. Supprimer et/ou déplacer le fichier ajouté via les commandes **git rm** et **git mv**.
3. Revenir à la version du fichier « **info.txt** » du 2^{ème} commit comme précédemment via la commande « **git checkout XXXX info.txt** ».
 - i. Vérifier que le fichier est bien dans la version voulue
 - ii. Décider finalement d'annuler les modifications :
 1. « unstager » le fichier via la commande « **git reset** »
 2. Annuler les modifications via « **git checkout -** »

Pour aller moins loin...

Notez que toutes ces opérations peuvent être réalisées sous Windows sans passer par la ligne de commande via l'outil « GIT GUI » (et sans doute bien d'autres du même type). Ajouter un fichier dans votre répertoire de travail et ouvrez l'interface graphique pour voir comment procéder pour l'enregistrer sur le dépôt.