

Rapport TP1 LO43:

Roles des commandes Git

Réaliser par : Imane ZENOUAKI

Introduction :

Git : est un logiciel de gestion de versions décentralisé. C'est un logiciel libre créé par Linus Torvalds, auteur du noyau Linux, et distribué selon les termes de la licence publique générale GNU version 2. En 2016, il s'agit du logiciel de gestion de versions le plus populaire qui est utilisé par plus de douze millions de personnes.

Après avoir réalisé le Tutoriel demandé sur le site try.github.io, j'ai utilisé l'outil "Visualizing Git" pour tester quelques commandes.

1- Git Config :

On utilise cette commande pour définir et configurer les informations de l'utilisateur, dans mon cas, j'ai choisi de configurer l'email de l'utilisateur comme sur la figure 1.

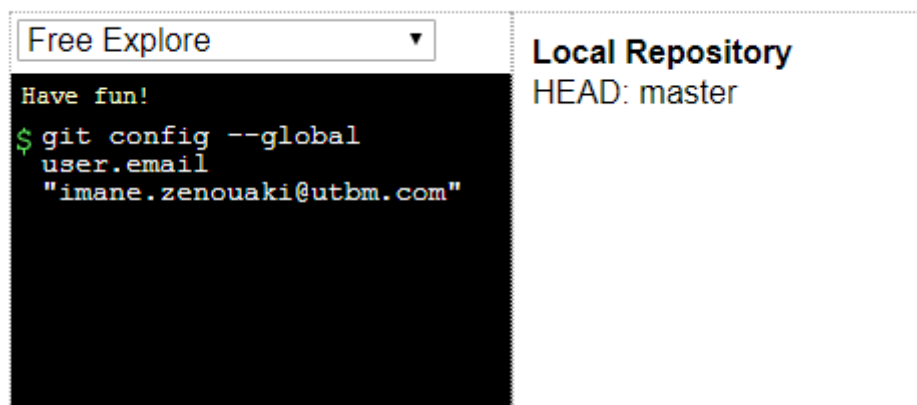
The image shows a screenshot of the 'Visualizing Git' web application. At the top left, there is a dropdown menu labeled 'Free Explore'. Below it, a terminal window displays the following text: 'Have fun!', '\$ git config --global', 'user.email', and '"imane.zenouaki@utbm.com"'. To the right of the terminal, the text 'Local Repository' is displayed in bold, with 'HEAD: master' below it.

Figure 1 Commande config

2- Commande Git init :

J'ai pas pu tester cette commande vu que je travaille seulement avec l'outil "Visualizing Git". Mais généralement Git init sert à créer un nouveau repertoire ou dépôt Git.

3- Commande Git status :

De même pour cette commande, son utilité est d'afficher la liste des fichiers modifiés et les fichiers qui doivent encore être ajoutés ou validés.

4- Commande Git diff :

Elle permet de lister les conflits.

5- Commande Git blame :

Cette commande sert à connaître qui a fait les modifications et sur quelle partie du fichier sur lesquels travaillent plusieurs personnes.

6- Commande Git merge :

Comme il est affiché dans la figure 2, La commande git merge est utilisée pour fusionner une branche dans la branche active, et je l'ai utilisée pour la branche master et mybranch et puis avec mabbranch.

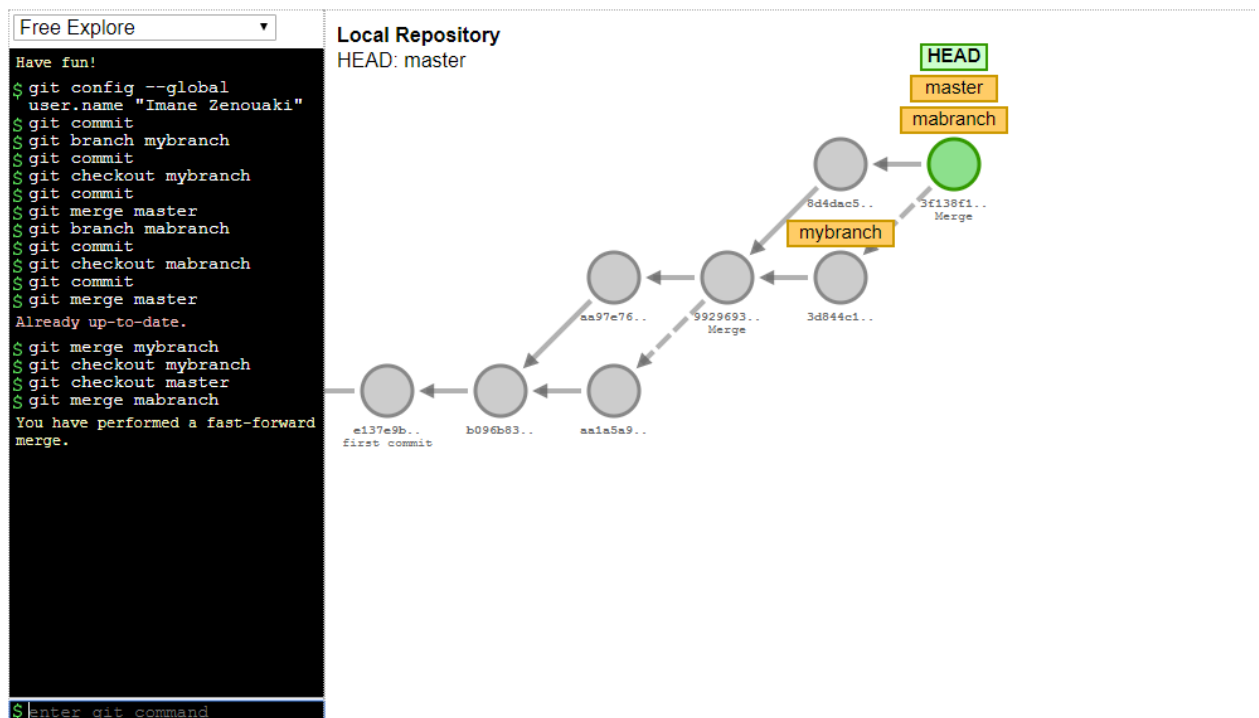


Figure 2

Pour le réaliser j'ai utilisé d'autres commandes, comme Git checkout, Git commit et Git branch.

Git Commit : Pour valider les modifications apportées au HEAD.

Git brach : Pour pour répertorier ou créer des branches.

Git checkout : Pour basculer entre les branches.