* Résumé
* Procédure d’installation (url, etc)
* Présentations des fonctionnalités
* Mini-guide d’utilisation sur les principales fonctionnalités dans le cadre du TP Multi-Agents
* Les points clés du logiciel et les commandes associés

Git est un logiciel de gestion de versions décentralisé

utilisé pour les projets de développements open source ou commerciaux.

Il peut être installé dans différent environnement et notamment Windows 10. Une fois installé le

logiciel est accessible de différentes manières :

- Git CMD est identique à la ligne de commande habituelle Windows et permet d’exécuter les

fonctionnalités de Git au travers de la ligne de commande Windows.

- Git Bash émule un environnement Bash sous Windows sous forme de ligne de commande.

Git Bash permet d’exécuter toutes les commandes Git et la plupart des commandes Unix.

- Enfin, Git GUI est une interface graphique qui vous permet d’utiliser Git sans la ligne de

commande.

Un « repository », ou « Projet Git », regroupe l’ensemble des fichiers et/ou dossier associé au projet

ainsi que l’historique de l’ensemble des modifications effectuées. L’historique des modifications est

constitué d’un ensemble d’instantané appelés « commit ». L’ensemble des « commit » sont

accessibles grâce à la commande « git log ». Enfin les développements peuvent être organiser sous

forme de chantier, travaillant en parallèle, grâce au concept de « branche ».

Git étant un DVCS, les « repository » permettent à toute personne ayant une copie du « repository »

d’accéder à l’ensemble des éléments et à leur historique.

Travailler avec des “repository” permet d’organiser et de sécuriser les projets de développements

Les développeurs peuvent corriger des bugs, développer des nouvelles fonctionnalités, sans altérer

les axes majeurs de développements grâce à la notion de « branche ».

Au travers de plateforme comme GitHub, Git offres des fonctionnalités pour encore plus de

transparence et de collaboration.

En effet, GitHub est une plateforme sur le web qui facilite la gestion du code grâce à des

fonctionnalités de travail collaboratif et de gestion de version. Elle permet de travailler à plusieurs

sur des projets quel que soit la localisation de l’équipe. L’ensemble des ressources d’un projet

peuvent être gérées (code, fichier texte, librairie, photo,….).

|  |
| --- |
| git status : permet de voir le status du repertoire courant, c'est-à-dire determiner si c'est un repository git ou pas. |
|  |  |
|  |  |
|  | git init : permet de signaler que le dossier courant est un repository git |
|  |  |
|  |  |
|  | git status : si le dossier courant est un repository git, en faisant git status, on saura les fichiers non traqués et les differents commit. |
|  |  |
|  |  |
|  | // Commande à exécuter après chaque modification |
|  | git add nomdufichier : permet d'ajouter le fichier à l'index |
|  | git add . : permet d'ajouter tous les fichiers du repertoire courant |
|  | git commit -m "Message du commit: Ajout du fichier...." : commande à exécuter après chaque git add. |
|  |  |
|  | git log : permet d'afficher l'historique des commmit |
|  |  |
|  | Pour ne pas faire git add ..... à chaque modification, il faut ajouter l'option -a au git commit, cela permet de prendre en compte les modifications des fichiers deja ajoutés dans l'index. |
|  | git commit -a -m "Message du commit" |
|  | git commit -am "Astuce -am ça marche" |
|  |  |
|  | git checkout fifjjie655644rzhhd : se positionner à commit dont le sha est fifjjie655644rzhhd. Le sha s'obtient en faisant un git log |
|  |  |
|  | git checkout master : se positionner au dernier commit |
|  |  |
|  | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Récuperer du code d'un autre repository \*\*\*\*\*\* |
|  | Se postionner dans le repertoire où on veut stocker le repository |
|  | git clone urldurepository |
|  |  |
|  | \*\*\*\* Envoyer votre code dans un nouveau repository github \*\*\*\*\*\* |
|  | Créer un nvo repository github |
|  | Se placer dans le repository local |
|  | git remote add origin urldunvorepositorygithub |
|  | git push -u origin master |
|  |  |
|  | Pour envoyer les dernieres modifications faites sur la machine sur github, il faut exécuter : |
|  | git push -u origin master en local après chaque commit ou simplement |
|  | git push |
|  |  |
|  | \*\*\*\*\*\*\*\*\*\*\*\*\* Cloner un nvo repository github sur sa machine \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |
|  | git clone urlrepository |
|  | cd nomrepository |
|  | faire des modif et des commit |
|  | git push origin master : envoie des modifications sur github |
|  | git push : tout simple, marche aussi |
|  |  |
|  | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Récupeartion des modifications sur Github \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |
|  | Pour récuperer les dernières modifications sur Github, on se place dans le repository local puis on exécute |
|  | git pull origin master |
|  | git pull : marche aussi |
|  |  |
|  |  |
|  |  |
|  |  |
|  | ================================================================================================================================= |
|  | Cours Conduite de Projet Web |
|  | ================================================================================================================================= |
|  | Les commandes de bases : |
|  | clone |
|  | push: permet f'envoyer au depot des modifications lacales |
|  | pull |
|  | merge |
|  | fetch |
|  |  |
|  | git remote add origin https://.......git |
|  |  |
|  | ====================================================================================== |
|  | Gestion des branches |
|  | ====================================================================================== |
|  | git commit |
|  | git push |
|  | git outofhere |
|  |  |
|  | Tag : pas de changement |
|  | Branche : evolue |
|  |  |
|  | Operations sur les branches |
|  | Lister les branches locales et distantes : git branch // git branch -r // git branch -a |
|  | Creer de branches locales : git branch nomBranche |
|  | Supprimer des branches locales : git branch -d nomBranche |
|  | Diffuser une branche locale sur le depot distant |
|  | Basculer d'une branche à l'autre : git checkout nomBranche |
|  |  |
|  | Gestion des conflits |
|  | git merge featureX |
|  |  |
|  | Rebase |
|  | Tag |
|  | Un tag est une reference à un commit |
|  | Lister les tags : git tag |
|  |  |
|  | Gitflow |