▂ ▃ ▄ ▅ ▆ ▇ █ ᵔᴥᵔ Doc git pour les nulos ᵔᴥᵔ █ ▇ ▆ ▅ ▄ ▃ ▂

1. **Tout d’abords il faut Download :**

* Source code : <https://github.com/git/git>
* Site web : <https://git-scm.com/>

1. **Pendant le DL, je vous propose une petite introduction à GIT :**

Vous écrivez un peu de code et souhaitez collaborer avec d'autres développeurs ? Vous voulez toujours savoir pourquoi une modification a été faite et ne plus risquer de perdre des jours de travail ? Vous avez clairement besoin de versionner votre code !

Un logiciel de gestion de versions est un outil incontournable pour tout développeur. Il en existe de nombreux, et dans cette doc pour les nulos, vous découvrirez Git, le logiciel créé par **Linus Torvald**, auteur du Kernel Linux. Vous découvrirez ce qu'est la gestion de version et les avantages que cela apporte. Puis, vous plongerez dans l'univers de Git : commit, branches, merge... tous ces termes n'auront plus de secrets pour vous ! Enfin vous apprendrez à utiliser GitHub pour héberger votre code et collaborer facilement sur des projets open-source.

1. **Maintenant les commandes principales :**

* **Git config :**

L’une des commandes git les plus utilisées est git config. On l’utilise pour configurer les préférences de l’utilisateur : son mail, l’algorithme utilisé pour diff, le nom d’utilisateur et le format de fichier etc. Par exemple, la commande suivante peut être utilisée pour définir le mail d’un utilisateur :

git config --global user.email [example@google.com](mailto:example@google.com)

* **Git init :**

Cette commande est utilisée pour créer un nouveau dépôt GIT :

git init

* **Git add :**

La commande git add peut être utilisée pour ajouter des fichiers à l’index. Par exemple, la commande suivante ajoutera un fichier nommé temp.txt dans le répertoire local de l’index:

git add temp.txt

* **Clone git :**

La commande git clone est utilisée pour la vérification des dépôts. Si le dépôt se trouve sur un serveur distant, utilisez:

git clone alex@93.188.160.58:/chemin/vers/dépôt

Inversement, si une copie de travail d’un dépôt local doit être créée, utilisez:

git clone /chemin/vers/dépôt

* **Git commit :**

La commande git commit permet de valider les modifications apportées au HEAD. Notez que tout commit ne se fera pas dans le dépôt distant.

git commit –m “Description du commit”

* **Git status :**

La commande git status affiche la liste des fichiers modifiés ainsi que les fichiers qui doivent encore être ajoutés ou validés. Usage:

git status

* **Git push :**

Git push est une autre commandes GIT de base. Un simple push envoie les modifications locales apportées à la branche principale associée :

git push origin master

* **Git checkout :**

La commande git checkout peut être utilisée pour créer des branches ou pour basculer entre elles. Par exemple nous allons créer une branche:

command git checkout -b <nom-branche>

Pour passer simplement d’une branche à une autre, utilisez:

git checkout <nom-branche>

* **Git remote :**

La commande git remote permet à un utilisateur de se connecter à un dépôt distant. La commande suivante répertorie les dépôts distants actuellement configurés:

git remote –v

Cette commande permet à l’utilisateur de connecter le dépôt local à un serveur distant:

git remote add origin <93.188.160.58>

* **Branche git :**

La commande git branch peut être utilisée pour répertorier, créer ou supprimer des branches. Pour répertorier toutes les branches présentes dans le dépôt, utilisez:

git branch

Pour supprimer une branche:

git branch –d <nom-branche>

* **Git pull :**

Pour fusionner toutes les modifications présentes sur le dépôt distant dans le répertoire de travail local, la commande pull est utilisée. Usage:

git pull

* **Git merge :**

La commande git merge est utilisée pour fusionner une branche dans la branche active. Usage:

git merge <nom-branche>

* **Git diff :**

La commande git diff permet de lister les conflits. Pour visualiser les conflits d’un fichier, utilisez

git diff --base <nom-fichier>

La commande suivante est utilisée pour afficher les conflits entre les branches à fusionner avant de les fusionner:

git diff <branche-source> <branche-cible>

Pour simplement énumérer tous les conflits actuels, utilisez:

git diff

* **Git log :**

L’ exécution de la commande git log génère le log d’une branche. Un exemple de sortie :

commit 15f4b6c44b3c8344caasdac9e4be13246e21sadw

Author: Alex Hunter <alexh@gmail.com>

Date: Mon Oct 1 12:56:29 2016 -0600

* **Git stash :**

L’une des moins connues, git stash aide à enregistrer les changements qui ne doivent pas être commit immédiatement. C’est un commit temporaire. Usage:

git stash

* **Git fetch :**

Git fetch permet à un utilisateur d’extraire tous les fichiers du dépôt distant qui ne sont pas actuellement dans le répertoire de travail local. Exemple d’utilisation:

git fetch origin



hihi