Git commit : créé une nouvelle version du programme, reliée à la précédente

Git branch [nom] : créer une branche au code pointé par le dernier checkout

Git checkout : permet de repositionner les commit à un autre endroit (changer de branche, revenir à une version précédente)

Git merge [branch] : créé un nouveau commit à partir de la branche indiquée et le commit pointé par le checkout

Git rebase : fusionne le contenu de 2 commit

^ : replace le checkout sur le commit précédent

Git reset [commit] : créé une nouvelle version du commit utilisant sa version précédente

Git revert [commit] : annule le dernier commit effectué en fonction de la position

Git rebase -i [commit]

Git cherry-pick [commit] créé une copie d'un commit sous le main

Git clone : clone la branche actuelle, avec ses commits

Git fetch : après un clonage, reprend les commits du clone dans la version originale

Git pull : git fetch + git merge

Git config : permet de renseigner son compte github sur sa machine

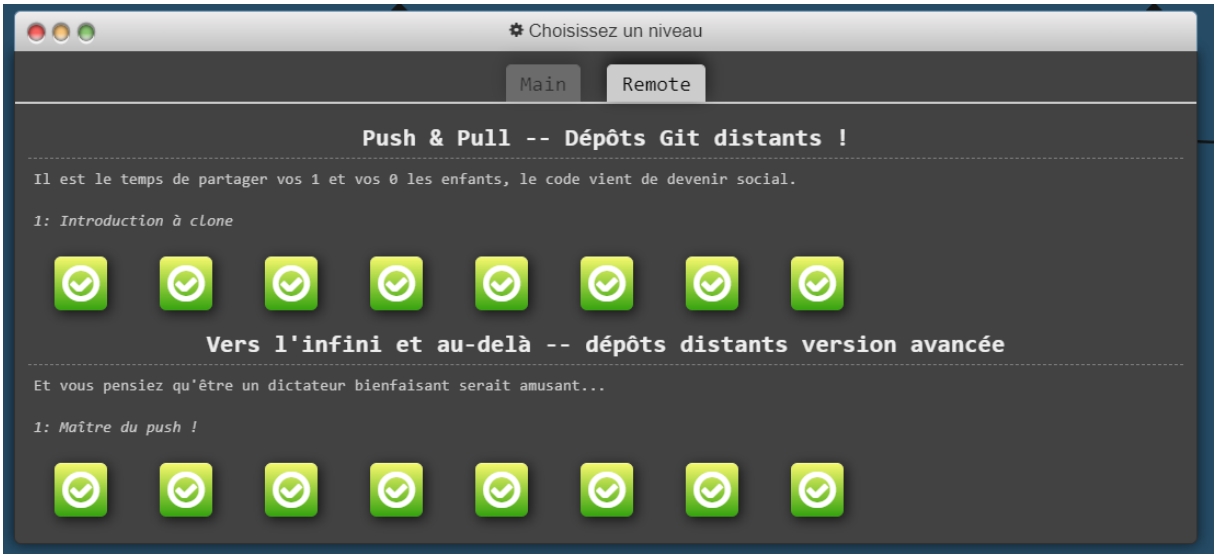
Git init : initialise un repo git

Git status : montre où est le positionnement actuel (avec les checkout), dans quelle branche nous sommes

Git add : ajoute des fichiers à l'index

Git diff : affiche les différences entre deux commit

Git blame : affiche qui a fait les différences (de git diff)




Séquence d'introduction
Une introduction en douceur à la majorité des commandes Git
1: Introduction aux commits avec Git
Montée en puissance
Le prochain excellent plat de pur Git. J'espère que vous êtes affamés
1: Détacher votre HEAD
Déplacer le travail
Soyez à l'aise pour modifier l'arbre Git
1: Introduction à cherry-pick
Un assortiment
Un assortiment de techniques et astuces pour utiliser Git
1: Choisir seulement 1 commit
Sujets avancés
Pour les plus courageux !
1: Rebaser plus de 1000 fois