# git

## Installation

• **Installer git sur Windows :** http://windows.github.com

• **Installer git sur Linux :** sudo apt install git-all

## Configuration des outils

• **Définir un nom d’utilisateur :** git config --global user.name nom

• **Définir un courriel d’utilisateur :** git config --global user.email courriel

• **Activer la colorisation de la sortie en ligne de commande :** git config --global color.ui auto

## Création de dépôts

• **Créer un dépôt local à partir du nom spécifié :** git init projet

Git flow init• **Télécharger un projet et tout son historique de versions :** git clone url

## Effectuation de changements

• **Lister tous les nouveaux fichiers et les fichiers modifiés à commiter** (les fichiers dans .gitignore sont ignorés)**:** git status

• **Ajouter un instantané d’un fichier, en préparation pour le suivi de version :** git add fichier

• **Ajouter un instantané des fichiers, en préparation pour le suivi de version :** git add --all

• **Voir les modifications de fichier qui ne sont pas encore indexées :** git diff

• **Voir les différences de fichier entre la version indexée et la dernière version :** git diff ‑‑staged

• **Enlever un fichier de l’index mais conserver son contenu :** git restore fichier

• **Enregistrer des instantanés de fichiers de façon permanente dans l’historique des versions :** git commit -m "message"

• **Afficher la liste des dépôts distants liés au dépôt local :** git remote -v

• **Ajouter le dépôt distant situé un certain emplacement :** git remote add dépôt emplacement (exemple : git remote add origin https://github.com/Helene-Singer/Python.git)

## Groupement des changements

• **Lister toutes les branches locales dans le dépôt courant :** git branch

• **Créer une nouvelle branche :** git branch branche

• **Créer une nouvelle branche, si elle n’existe pas, et basculer dessus :** git checkout branche

• **Basculer sur la branche spécifiée et mettre à jour le répertoire de travail :** git switch branche

• **Combiner dans la branche courante l’historique de la branche spécifiée :** git merge branche

• **Renommer la branche sur laquelle on est :** git branch -M nouveau\_nom

• **Supprimer la branche spécifiée :** git branch -d branche

## Changements au niveau des noms de fichiers

• **Supprimer un fichier du répertoire de travail et mettre à jour l’index :** git rm fichier

• **Supprimer un fichier du système de suivi de version mais le préserver localement :** git rm ‑‑cached fichier

• **Renommer un fichier et préparer le changement pour un commit :** git mv fichier nouveau\_fichier

## Exclusion du suivi de version

• **Lister tous les fichiers exclus du suivi de version dans ce projet :** git ls-files -‑other ‑‑ignored ‑‑exclude-standard

## Enregistrement des fragments

• **Enregistrer de manière temporaire tous les fichiers sous suivi de version qui ont été modifiés** (remiser) **:** git stash

• **Appliquer une remise et la supprimer immédiatement :** git stash pop

• **Lister toutes les remises :** git stash list

• **Supprimer la remise la plus récente :** git stash drop

## Vérification de l’historique des versions

• **Voir l’historique des versions pour la branche courante :** git log

• **Voir l’historique des versions, y compris les actions de renommage, pour le fichier spécifié :** git log --follow fichier

• **Voir les différences de contenu entre deux branches :** git diff branche1 branche2

• **Voir les modifications de métadonnées et de contenu inclues dans le commit spécifié :** git show commit

## Refaire des commits

• **Remplacer le dernier commit après avoir fait les modifications nécessaires :** git commit ‑‑amend -m "message"

• **Annuler tous les commits après le commit spécifié, en conservant les modifications localement :** git reset commit

• **Supprimer l’historique et les modifications effectuées après le commit spécifié :** git reset ‑‑hard commit

## Synchronisation des changements

• **Récupérer l’historique du dépôt nommé :** git fetch dépôt

• **Fusionner la branche du dépôt dans la branche locale courante :** git merge dépôt/branche

• **Envoyer tous les commits de la branche locale vers le dépôt distant :** git push dépôt branche (premier push d’un dépôt : git push -u origin main)

• **Récupérer l’historique du dépôt nommé et incorpore les modifications** (uniquement la branche sur laquelle on est)**:** git pull

• **Récupérer l’historique du dépôt nommé et incorpore les modifications** (toutes les branches)**:** git pull --all

• **Envoyer tous les commits de la branche locale vers le dépôt distant si le dépôt distant a rencontré des changements :**

git pull –rebase

git push

## Nommage des commits

• **Ajout d’une caractéristique :** feat

• **Correction de bug :** fix

• **Modification de la documentation :** docs

• **Modification du style (espaces blancs, formatage, guillemets manquants, etc.) :** style

• **Remaniement de code** (ne fixe pas de bug et n’ajoute pas de caractéristique) **:** refactor

• **Amélioration de performance :** perf

• **Ajout ou correction de tests :** test

• **Modification qui affecte le système de build ou les dépendances externes :** build

• **Modification des fichiers de configuration ou des scripts d’intégration continue :** ci

• **Modification qui n’affecte pas les fichiers test ou src :** chore

• **Annulation du commit précédent :** revert

git commit -am "message" = git add . + git commit -m "message"

gitk = interface graphique de Git

git push --all : copie toutes les branches sur git