## Dépôt

Créer un dépôt dans un répertoire :

cd c:\repertoire\du\futur\depot

git init

Vérifier l’état des fichiers dans le dépôt :

git status

Annuler les modifs d'un fichier pas encore stagé :

git checkout -- chemin/vers/fichier.txt

## Staging (=prochain commit)

Inclure tous les fichiers modifiés :

git add --all ou git add -A

Inclure tous les fichiers d’un répertoire :

git add chemin/vers/repertoire

Inclure <chemin/vers/fichier> :

git add <chemin/vers/fichier>

Inclure uniquement les fichiers déjà versionnés :

git add -u

Inclure modifs bloc par bloc, fichier par fichier :

git add -p [<fichier>]

Déstager (ne plus inclure) <fichier> :

git reset -- <fichier>

## Suppression

Sup. un fichier de l'index, garder copie de travail :

git rm --cached chemin/vers/fichier.txt

Sup. un fichier de l'index ET de copie de travail :

git rm chemin/vers/fichier.txt

Sup. un fichier de copie de travail, pas de l'index :

rm chemin/vers/fichier.txt

Sup. fichiers non versionnés de la copie de travail :

git clean

## Stash (=commits temporaires)

git stash list

git stash save ["message"]

git stash pop

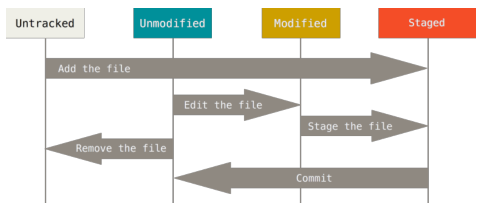
git stash apply [id de stash]

git stash show [id de stash]

git stash drop [id de stash]

git stash clear

## Interactions avec serveur distant



Cloner un dépôt :

cd c:\un\repertoire

git clone <url>

Récupérer les modifications du serveur sans les appliquer sur la copie de travail :

git fetch

Récupérer les modifications du serveur et les appliquer sur la copie de travail :

git pull

Envoyer vos commits sur le serveur :

git push

Envoyer <branche\_locale> sur le serveur :

git push -u origin <branche\_locale>

Envoyer les modifs de <branche> sur le serveur :

git push origin <branche>

Envoyer les tags créés précédemment :

git push --tags

## Renommage/déplacement

Renommer un fichier :

git mv chemin/source.txt chemin/dest.txt

## Commits

Commit avec éditeur Vim :

git commit

Taper "i", puis message, puis "Echap : w q"

Commit avec un message d’une ligne :

git commit -m "message"

Commit avec message multi-lignes :

git commit -m "ligne 1" -m "ligne 2"

Commit avec staging auto (git add --all implicite) :

git commit -a -m "message"

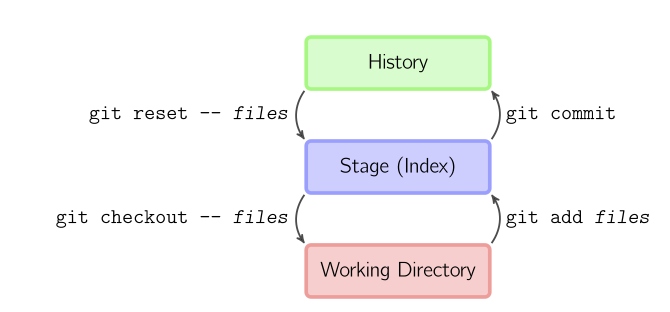
Marquer le commit courant avec un tag

git tag <nom\_tag>

Annuler les modifs de <fichier> depuis un commit particulier :

git checkout <commit\_id> <fichier>

Annuler les modifs de <fichier> depuis le dernier commit :



git checkout HEAD <fichier>

Annuler toutes les modifs du dépôt depuis le dernier commit :

git revert HEAD

Annuler un commit et les suivants en gardant les modifications (ces commits ne doivent pas avoir été pushés) :

git reset <commit\_id>

Annuler un commit et les suivants sans garder les modifications (ces commits ne doivent pas avoir été pushés) :

git reset --hard <commit\_id>

Annuler les modifications d'un commit particulier, mais pas celles des commits suivants :

git revert <commit\_id>

Intégrer les modifs du commit spécifié dans la branche courante :

git cherry-pick <commit\_id>

## Branches

Lister les branches distantes :

git branch -r

Se placer sur une branche existante :

git checkout <nom\_branche>

Créer nouvelle branche et se positionner dessus :

git checkout -b <nom\_branche>

Supprimer une branche :

#pour branche locale :

git branch -d <nom\_branche>

#pour branche distante :

git push origin --delete <nom\_branche>

Fusionner une branche (merge) :

git checkout <branche\_cible>

git pull

git merge --no-ff <branche\_a\_fusionner>

## Historique

Voir les 5 derniers commits :

git log -n 5

Voir les modifications depuis le dernier commit :

git diff [fichier/commit/branche]

Voir modifs entre deux commits ou branches :

git diff <commit\_id\_1> <commit\_id\_2>

Qui a changé quoi et quand dans <fichier>

git blame <fichier>

# Plugin Egit pour Eclipse

**T** = Clic droit sur le projet, menu "Team"

**T** = Clic droit sur une ou plusieurs ressources, menu "Team"

## Serveur distant

Récupérer les modifications et les appliquer

**T** > Pull

Récupérer les modifications sans les appliquer

**T** > Fetch from upstream

Envoyer vos commits sur le serveur

**T** > Push to upstream

## Staging

Avec Egit, le staging n'a aucun effet sur la liste des fichiers qui seront cochés lors du prochain commit.

Stager un fichier pour le prochain commit :

**T** > Add to Index

Stager tous les fichiers pour le prochain commit (sans confirmation)

**T** > Add to Index

Déstager un fichier

**T** > Remove from Index

## Commits

Commiter uniquement les fichiers sélectionnés :

**T** > Commit

Taper un message

Bouton "Commit" : faire un commit local

Bouton "Commit and push" : idem puis fait un Push

Commiter tous les fichiers modifiés :

**T** > Commit

Taper un message

Bouton "Commit" : faire un commit local

Bouton "Commit and push" : idem puis fait un Push

## Branches

Créer une branche locale liée à une branche distante ( à faire 1 seule fois)

**T** > Switch To > New branch > liste "Source ref" >

Sélectionner refs/remotes/origin/nom\_branche et Finish

Se placer sur une branche locale existante

**T** > Switch To > Sélectionner une branche existante

Créer une nouvelle branche :

**T** > Switch To > New Branch

Choisir la branche source (ex : ref/heads/nom\_branche)

Taper un nom de branche, Finish

Supprimer une branche :

**T** > Advanced > Delete branch

Dans Local, sélectionner la branche, OK

Fusionner une branche :

Se placer sur la branche destination

**T** > Merge, choisir la branche à fusionner, Merge

# TortoiseGit

**R** = Clic droit sur la racine du dépôt

**F** = Clic droit sur un fichier ou répertoire

## Serveur distant

Récupérer les modifications et les appliquer

**R** > TortoiseGit > Pull…

Récupérer les modifications sans les appliquer

**R** > TortoiseGit > Fetch…

Envoyer vos commits sur le serveur

**R** > TortoiseGit > Push…

## Staging

TortoiseGit gère le staging des fichiers automatiquement par l'intermédiaire de cases à cocher lors d'un commit.

## Commits

Commiter uniquement les fichiers sélectionnés (cela ne fonctionne que si les fichiers ont été stagés) :

**F** > Git Commit -> "nom branche"…

Commiter tous les fichiers modifiés :

Sur un fichier non stagé :

**F** > Git Commit -> "nom branche"…

Ou bien :

**R** > Git Commit -> "nom branche"…

## Branches

Se placer sur une branche existante

**R** > TortoiseGit > Switch/Checkout… > Choisir branche > OK

Créer une nouvelle branche

**R** > TortoiseGit > Create Branch…

Taper un nom de branche, choisir la branche source, OK

Supprimer une branche

**R** > TortoiseGit > Switch/Checkout… > bouton "…" > Clic droit sur la branche à supprimer > Delete Branch

Fusionner une branche

Se placer sur la branche destination

**R** > TortoiseGit > Merge… > From > Choisir branche source > OK

## Revert

Les fichiers sélectionnés :

**F** > TortoiseGit > Revert… > Ok

Tous les fichiers modifiés :

**R** > TortoiseGit > Revert… > Cocher les fichiers à revert > Ok