

## DÉPÔT

### git init --bare <dir>.git

Crée un dépôt sans *working copy*

### git init .

Crée un dépôt dans le répertoire courant

### git clone <url> <local\_dir>

git clone ssh://[user@host.xz[:port]/path/to/repo.git/ .

git clone ssh://[user@host.xz[:port]/~[user]/path/to/repo.git/ .

Crée une copie locale du dépôt distant et le référence sous le nom *origin*

### git fetch

Rapatrie les modifications du dépôt *origin*

### git pull

Rapatrie les modifications du dépôt à l'origine de la branche courante et effectue un *merge* sur la branche courante

### git pull --rebase

Rapatrie les modifications du dépôt à l'origine de la branche courante et *rebase* la branche courante sur ces modifications.

### git push

Envoie les *commits* locaux sur le dépôt *origin* (les branches locales non associées à une branche distante ne sont pas envoyées).

### git push origin <local\_branch>:<remote\_branch>

Crée la branche <remote\_branch> sur le dépôt *origin* à partir de la branche locale <local\_branch> et y envoie les *commits* locaux

### git push origin :<remote\_branch>

Supprime la branche <remote\_branch> sur le dépôt *origin*

## WORKING COPY

### git status

Donne l'état de la branche courante du dépôt local

### git status .

Donne l'état du répertoire courant et de ses sous-répertoires dans la branche courante du dépôt local

### git log

Affiche les logs de *commits* de la branche courante

### git log -n <n>

Affiche les logs des <n> derniers *commits* de la branche courante

### git diff

Affiche les différences entre la *working copy* et la *staging area* (les nouveaux fichiers sont ignorés)

### git diff HEAD

Affiche les différences entre la *working copy* et le dernier *commit* (les nouveaux fichiers sont ignorés)

### git add <file>

git add <dir>

git add <dir>/\*.c

git add <dir>/^\*.c

Ajoute un élément à la *staging area* :

- un fichier simple
- tous les fichiers de <dir> et de ses sous-répertoires
- tous les fichiers \*.c de <dir> (shell asterisk expand)
- tous les fichiers \*.c de <dir> et de ses sous-répertoires (git asterisk expand)

### git rm <file>

Supprime un fichier de la *working copy* et le marque comme supprimé dans la *staging area*

### git rm --cached <file>

Supprime un fichier de la *staging area*

### git commit [-m "<message>"]

Fait un *commit* sur la branche courante à partir de la *staging area*

### git commit --amend

Modifie le dernier *commit*...

### git stash

Sauvegarde les modifications locales et remet la *working copy* dans l'état de *HEAD*

### git stash pop

Applique le dernier *stash* sur la *working copy* et le supprime

### git stash apply

Applique le dernier *stash* sur la *working copy*

### git stash drop

Supprime le dernier *stash*

### git stash branch <new\_branch>

Crée la branche <new\_branch>, bascule dessus, y applique le dernier *stash* et le supprime. L'origine de la branche est le *commit* parent du *stash*

## BRANCHES

### git branch

Affiche les branches du dépôt local

### git branch -d <branch>

Supprime la branche <branch>

### git branch <new\_branch>

Crée la branche <new\_branch> depuis le *HEAD*

### git checkout -b <new\_branch>

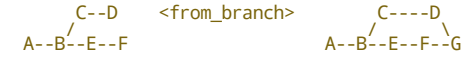
Crée la branche <new\_branch> depuis le *HEAD* et bascule dessus

### git checkout <branch>

Bascule la *working copy* sur la branche <branch>

### git merge <from\_branch>

Merge la branche <from\_branch> sur la branche courante



### git rebase <base\_branch>

*Rebase* la branche courante au dessus de la branche <base\_branch>



### git rebase -i HEAD~<n>

*Rebase* interactif des <n> derniers *commits* (edition, fusion, suppression...)

## DIVERS - CONFIGURATION

### git config [--global] user.name "<Firstname Lastname>"

Configure le nom d'utilisateur pour le dépôt ou globalement (--global)

### git config [--global] user.email "<user@domain.tld>"

Configure l'e-mail d'utilisateur pour le dépôt ou globalement (--global)

### .git/config

Fichier de configuration du dépôt

### .git/info/exclude

Fichier privé des exclusions (non partagé avec le dépôt)

### .gitignore

Fichier(s) public(s) des exclusions (versionné dans le dépôt)

### gitk

Outil graphique de parcours de l'historique du dépôt

### git gui

Outil graphique de gestion du dépôt (*commits*, *staging area*, branches...)

### man git <git\_command>

Manuel d'une commande

## REMARQUES

- *HEAD* = dernier *commit* de la branche courante
- Ne jamais *rebase* un travail déjà publié sur un autre dépôt

## LIENS

<http://git-scm.com/>

<http://progit.org/book/fr/>

<http://www.alexgirard.com/git-book/>