

presentation de Git



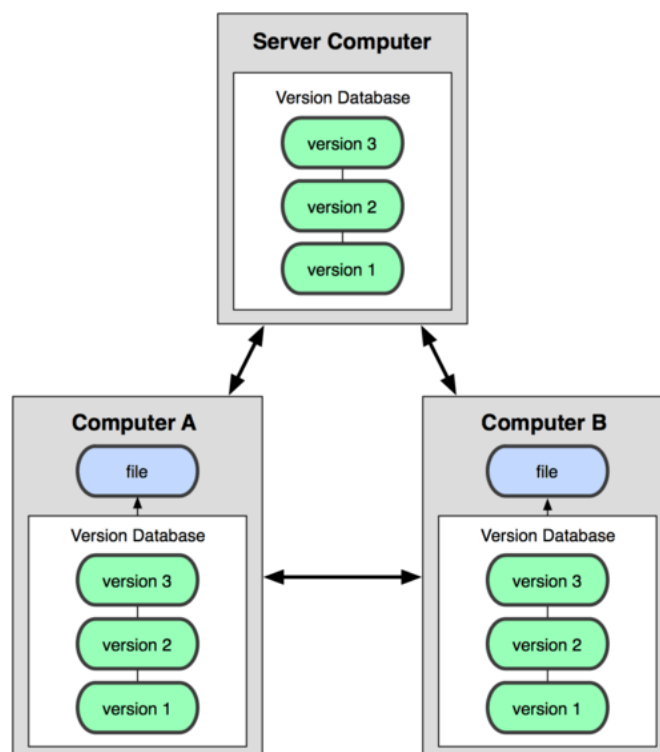
Figure 1: git

Concepts de base

- ▶ Décentralisé
- ▶ Rapide
- ▶ Flexible

Note: Inspiré par BitKeeper et créé en 2005 pour le remplacer après que la gratuité de ce dernier soit révoqué.

Décentralisé



Avantages

- ▶ Chaque clone est un fork
- ▶ Chaque clone est un backup
- ▶ Possibilité de versionner hors ligne
- ▶ **Très** rapide

Inconvénients

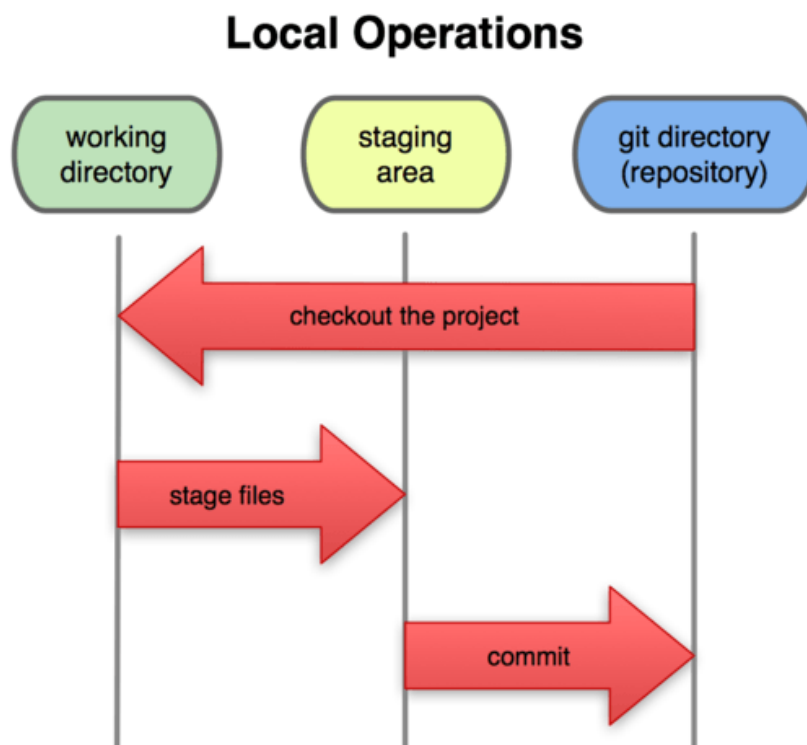
- ▶ Pas très performant pour les gros fichiers binaires

Note: Chaque clone est un backup : a condition de maintenir a jour toutes les branches.

Objectif de git

- ▶ Rapidité
- ▶ Simplicité
- ▶ Prise en charge de milliers de branches parallèles
- ▶ Architecture distribuée
- ▶ Capable de gerer efficacement de gros projet (*ex:Le noyau Linux*)
- ▶ Garantir l'intégrité

Les trois états

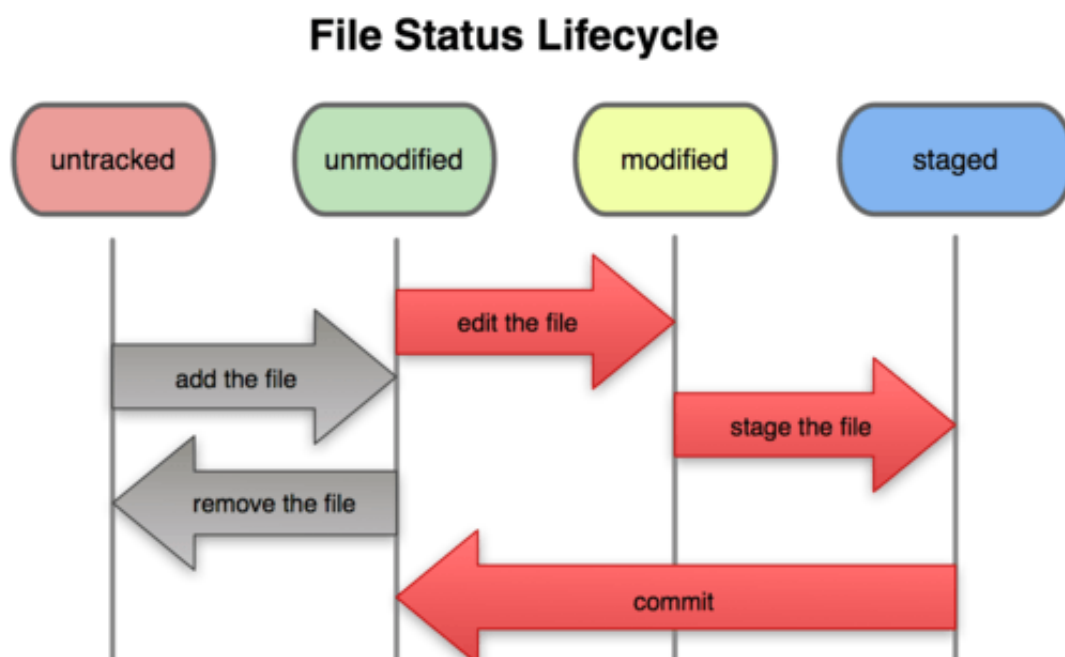


Le dossier git

Le répertoire de travail est une extraction unique d'une version du projet. Ces fichiers sont extraits depuis la base de données compressée dans le répertoire Git et placés sur le disque pour pouvoir être utilisés ou modifiés.

La zone d'index est un simple fichier, généralement situé dans le répertoire Git, qui stocke les informations concernant ce qui fera partie du prochain instantané.

Utilisation standard



Installer git



Figure 5: one does not simply install git on windows

Officiel

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

Note: Binaire, installeur et source pour
Windows, OS X, Linux, Solaris

Windows

<http://msysgit.github.io/>

LINUX

- ▶ Ubuntu: `sudo apt-get install git`

OS X

Avec Homebrew `brew install git`

Note: Ok mais a quoi ça ressemble ?

Interface graphique ?

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

Note: Des interface graphique existe... Vous pouvez les trouver la. À vous de choisir celle qui vous convient. Recommandé :

- ▶ SmartGit (crossplatform, gratuit pour usage personnel)
- ▶ SourceTree (sous mac, le plus populaire)
- ▶ GitHub for Mac / Windows (attention : spécifique à GitHub)

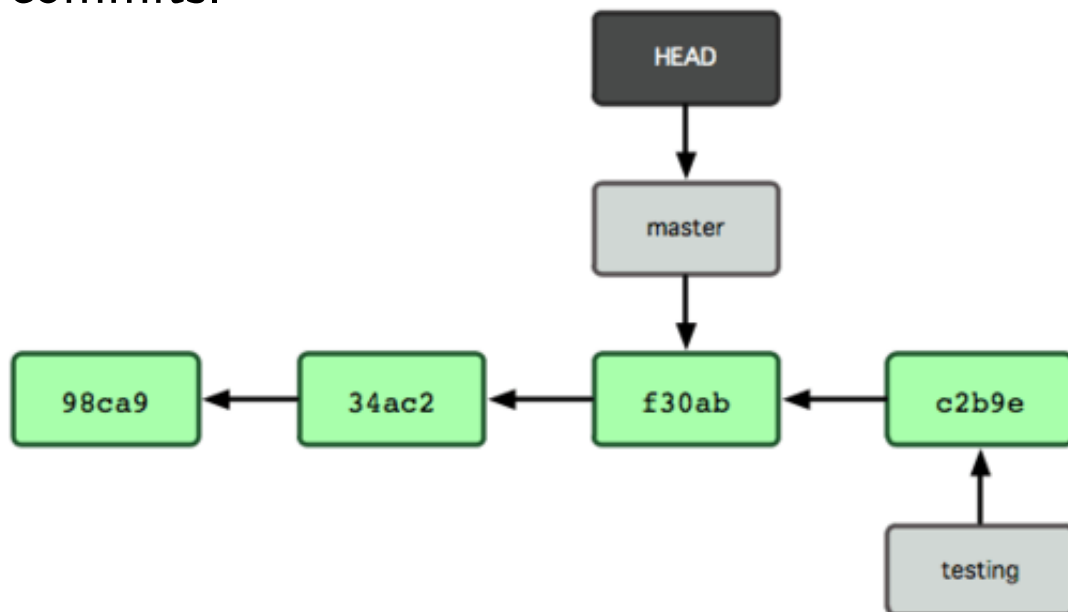
Configuration

<http://git-scm.com/book/fr/Personnalisation-de-Git-Configuration-de-Git>

Note: Et voilà, git est prêt !

Branches

Les branches sont des “pointeurs” vers des commits.



Fusionner des branches

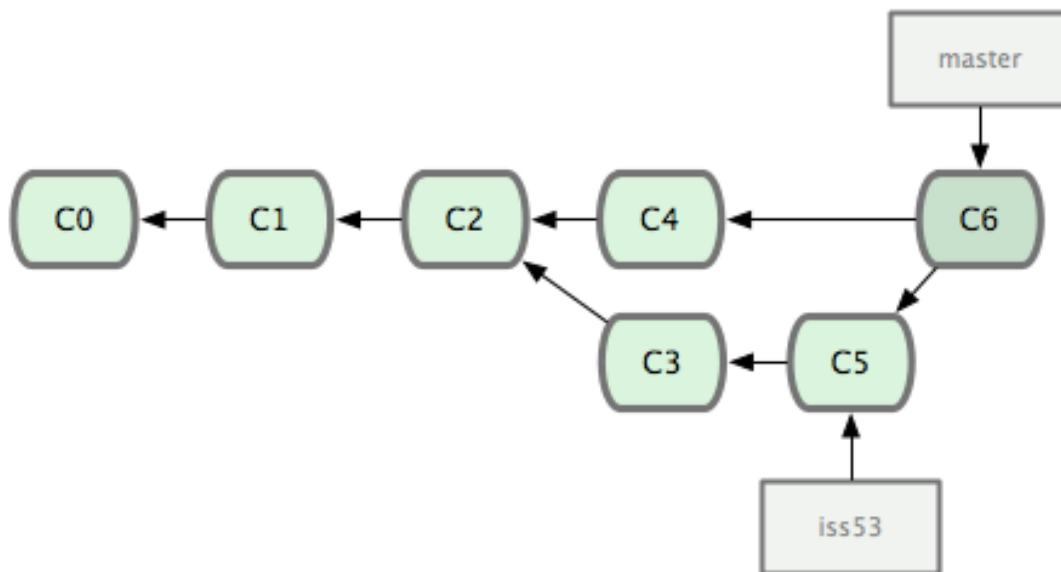


Figure 6: schema common workflow

En général ça marche sans problème

**BRANCHED FROM MASTER 3
WEEKS AGO**

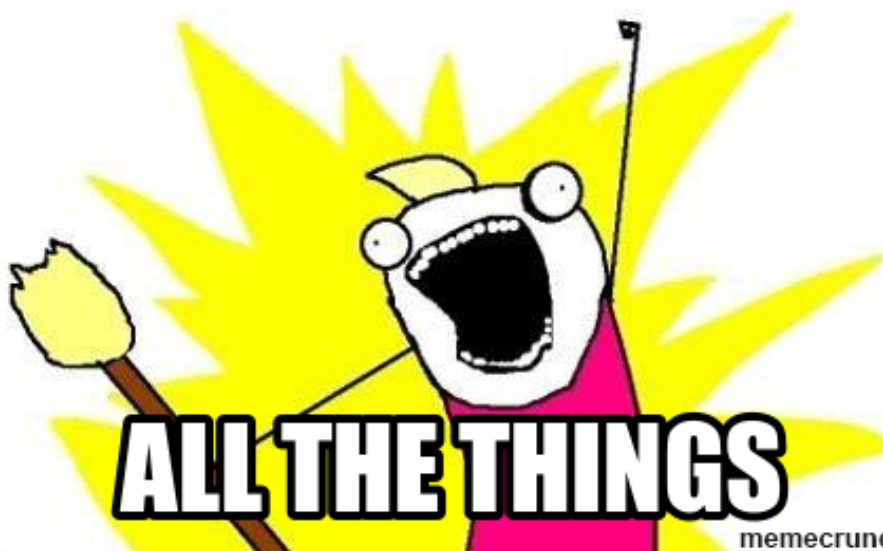


**MERGED BACK WITHOUT
ANY CONFLICTS**

memecrunch.com

Et maintenant ?

GIT



ALL THE THINGS

memecrunch.com

Resources

Très (très) inspirée de
`https://github.com/p-j/isep-git/
blob/master/data/slides.md`

- ▶ “Pro Git” Book
- ▶ Git reference
- ▶ Github git challenges
- ▶ Cheat sheet
- ▶ Git Visual Cheat sheet

Questions ?

Master comme branche stable

- ▶ On ne commit dans master que du code stable
- ▶ On commit le code instable dans une branche de développement
- ▶ On déploie des patchs sur master
- ▶ On merge les patchs dans la branche de développement

Master comme branche de développement

- ▶ On commit le code instable dans master
- ▶ On créer des tags pour marquer les étapes stables
- ▶ On créer une branche à partir d'un tag pour les patchs
- ▶ On merge les patchs dans master

Compléments

Commandes de base

L'aide dans git

```
$ git --help
```

```
$ git add --help
```

```
$ git <sub-command> --help
```

Créer un dépôt local

```
$ git init
```

Cloner un dépôt existant

To be continued...

.gitignore

```
$ cat .gitignore
.DS_Store
.svn
log/*.log
tmp/**
node_modules/
```

- ▶ Les lignes vides ou démarrant par un # sont ignorées
- ▶ Il est possible d'utiliser des jokers

Remotes

- ▶ Autre clones du même dépôt
- ▶ Peut être local (un autre checkout) ou distant (collègue, serveur central)
- ▶ On peut configurer une valeur par défaut pour push et pull

```
$ git remote -v
origin  git@github.com:p-j/isep-fsnp.g
origin  git@github.com:p-j/isep-fsnp.g
```

Push to remote

Stashing

```
$ git stash  
$ git stash pop  
$ git stash list  
$ git stash apply  
$ git stash drop  
$ git stash clear
```

Note: Utilisez git stash quand vous voulez enregistrer l'état actuel du répertoire de travail et de l'index, et revenir à un répertoire de travail propre. La commande enregistre

Couleur

```
$ git config --global color.ui true
```