

Formation GIT

Logiciel de gestion de version décentralisé

Nicolas Aguirre

26 Mars 2013

Plan

- 1 Présentation de Git
- 2 Notions de Workflows
- 3 Base de données GIT
- 4 Les 3 zones
- 5 Commandes de base
- 6 Travailler avec des serveurs distants
- 7 Commandes avancées
- 8 Credits

Définition de GIT

- Logiciel de gestion de version décentralisé
- Permet de stocker un ensemble de fichiers en conservant la chronologie des modifications
- Permet de stocker l'ensemble des modifications et des relations localement sans avoir à se connecter à un serveur.

Historique

- Projet initié par Linus Torvalds pour le noyau linux.
- Première version le 7 Avril 2005.
- Logiciel de gestion de versions décentralisé.
- Semblable à Mercurial ou Bitkeeper.
- Version actuelle 1.8.0
- Licence GPLv2.
- <http://git-scm.com>

Git vs SVN

- GIT \neq SVN.
- Ne jamais se demander quelle est la version SVN de cette commande.
- svn checkout \neq git checkout.
- Dans svn trunk est LA branche.
- Dans git master est UNE branche parmi d'autres.

- Un workflow est un ensemble de tâches et d'opérations effectuées par:
 - ▶ Un développeur;
 - ▶ Un intégrateur;
 - ▶ Une société;

Workflow développeur

- Créer une branche de dev.
- Commiter comme bon vous semble.
- Aussi souvent que vous voulez.
- Pusher ou merger les commits en ensembles cohérents.

Workflow développeur

- Supprime la peur de ne pas être à jour.
- Permet de faire de la revue de code sur des ensembles cohérents.
- Permet de travailler par fonctionnalités.
- Permet d'expérimenter dans des branches.
- Le développeur devient un producteur de code source.

Workflow de l'intégrateur

- Récupération des commits dans une branche.
- Intégration par fonctionnalités.
- Test des fonctionnalités.
- Génération de branche par livraison.
- Génération de tags.

Workflow de la société

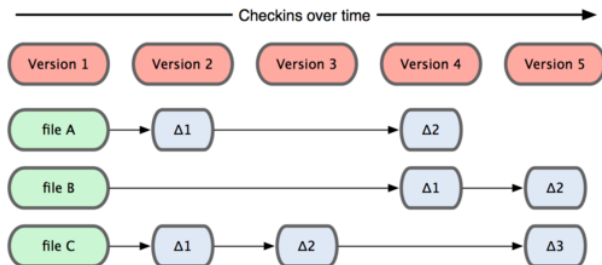
- Utilisation d'un serveur de référence.
- Possibilité de stocker toutes les branches des développeurs et des intégrateurs.
- Possibilitié de stocker uniquement les branches livrées.

Répertoire .git

- La base de donnée est stockée dans le répertoire .git à la racine de votre projet.

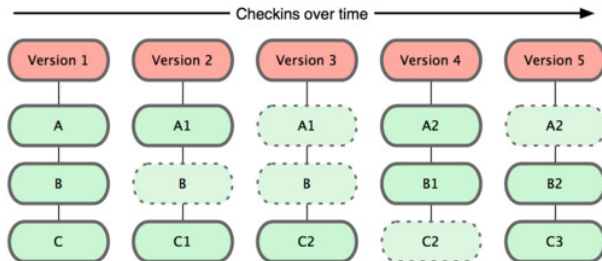
Comment sont stockées les données dans SVN

- Dans SVN l'information d'un commit est stockée sous forme de différences:



Comment sont stockées les données dans GIT

- Git fait un snapshot des fichiers a chaque version:



Comment sont stockées les données dans GIT

- Git indexe les fichiers d'après leur somme de contrôle SHA1.
- Les fichiers sont donc stockés uniquement si ils sont modifiés.
- Si un fichier est modifié il est stocké deux fois sur le disque.

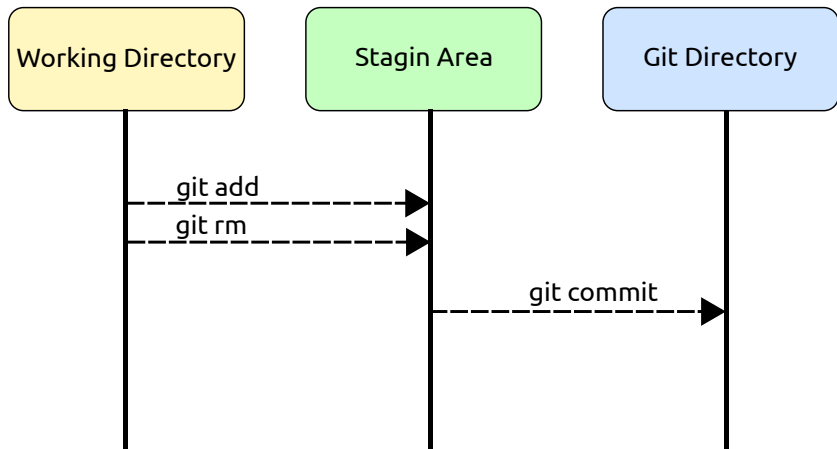
Toutes les informations sont stockées en local

- Toutes les informations sont stockées en local sur votre disque.
- Pas de latence due au réseau.
- L'historique de tout le projet est local.
- Tous les objets git ont une somme de contrôle SHA-1.

Les 3 zones

- la zone “working directory”
- la zone “staging area”
- la zone “git directory(repository)”

Les 3 zones



Git directory

- C'est ce qui est copié lorsque vous clonez un dépôt d'un autre endroit.
- C'est la base de donnée de git.
- C'est l'endroit où il stocke tous les commits, et toutes les relations entre commits.

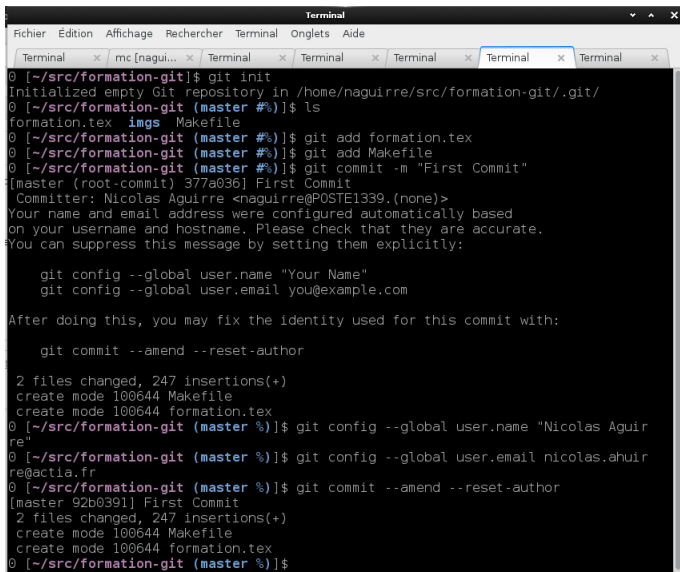
Working directory

- C'est un snapshot d'une des version du projet.
- Les fichiers sont issues de la base de données du git directory.
- Git les place sur le disque pour que vous puissiez les utiliser ou les modifier.

Staging Area

- C'est un fichier qui contient les informations de ce qui ira dans votre prochain commit.

Mon premier commit



```
Terminal
Fichier  Edition  Affichage  Rechercher  Terminal  Onglets  Aide
Terminal x mc[nagui... x Terminal x Terminal x Terminal x Terminal x Terminal x

0 [~/src/formation-git]$ git init
Initialized empty Git repository in /home/naguirre/src/formation-git/.git/
0 [~/src/formation-git (master #)]$ ls
formation.tex  imgs  Makefile
0 [~/src/formation-git (master #)]$ git add formation.tex
0 [~/src/formation-git (master #)]$ git add Makefile
0 [~/src/formation-git (master #)]$ git commit -m "First Commit"
[master (root-commit) 377a036] First Commit
  Committer: Nicolas Aguirre <naguirre@POSTE1339.(none)>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

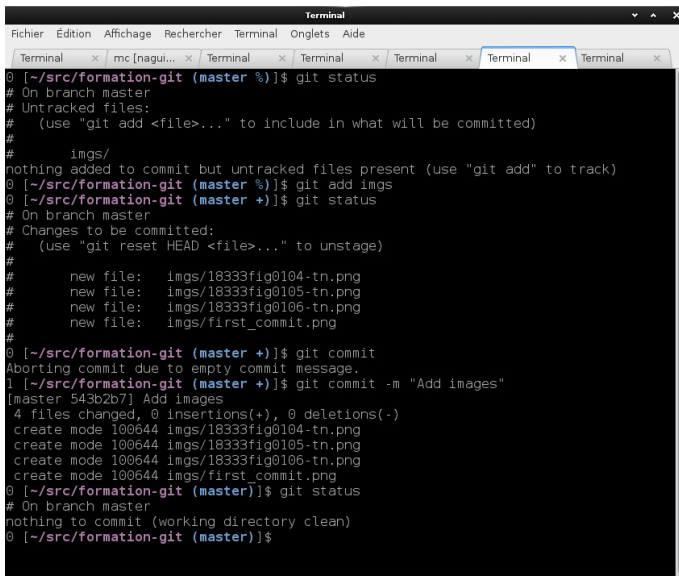
    git config --global user.name "Your Name"
    git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

2 files changed, 247 insertions(+)
 create mode 100644 Makefile
 create mode 100644 formation.tex
0 [~/src/formation-git (master %)]$ git config --global user.name "Nicolas Aguirre"
0 [~/src/formation-git (master %)]$ git config --global user.email nicolas.aguirre@actia.fr
0 [~/src/formation-git (master %)]$ git commit --amend --reset-author
[master 92b0391] First Commit
 2 files changed, 247 insertions(+)
 create mode 100644 Makefile
 create mode 100644 formation.tex
0 [~/src/formation-git (master %)]$
```

git status



```
Terminal
Fichier  Édition  Affichage  Rechercher  Terminal  Onglets  Aide
Terminal x mc [nagui... x Terminal x Terminal x Terminal x Terminal x Terminal x
0 [~/src/formation-git (master %)]$ git status
# On branch master
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#       imgs/
nothing added to commit but untracked files present (use "git add" to track)
0 [~/src/formation-git (master %)]$ git add imgs
0 [~/src/formation-git (master +)]$ git status
# On branch master
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       new file:   imgs/18333fig0104-tn.png
#       new file:   imgs/18333fig0105-tn.png
#       new file:   imgs/18333fig0106-tn.png
#       new file:   imgs/first_commit.png
#
0 [~/src/formation-git (master +)]$ git commit
Aborting commit due to empty commit message.
1 [~/src/formation-git (master +)]$ git commit -m "Add images"
[master 543b2b7] Add images
 4 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 imgs/18333fig0104-tn.png
 create mode 100644 imgs/18333fig0105-tn.png
 create mode 100644 imgs/18333fig0106-tn.png
 create mode 100644 imgs/first_commit.png
0 [~/src/formation-git (master)]$ git status
# On branch master
nothing to commit (working directory clean)
0 [~/src/formation-git (master)]$
```

Visualisation

Historique des commits: `git log`

```
$ git log
```

Visualisation des différences: `git diff` et `git diff --staged`

```
$ git diff
```

```
$ git diff --staged
```

Récapitulatif des commandes de base

```
$ git init  
$ git add  
$ git rm  
$ git commit  
$ git status  
$ git diff  
$ git log  
$ git tag  
$ git branch  
$ git checkout
```


Outil graphique: gitg, gitk, git gui

- gitg et gitk sont des outils graphique permettant de visualiser plus facilement l'arbre de commits.
- git gui est un outil graphique permettant de commiter de manière graphique.
- il existe également Tortoise git sur windows

Buildroot

Buildroot - Mozilla Firefox


File Edit View History Bookmarks Tools Help

Buildroot

buildroot.uclibc.org/git.html

Google

Most Visited Linux Mint Community Forums Blog News Brasil (Em Portuguese) The JeeLabs Shop - R...



Git Access

The buildroot repository can be browsed online through cggit at <http://git.buildroot.net/buildroot>. To grab a copy of the repository use

```
git clone git://git.buildroot.net/buildroot
```

Or if you're behind a firewall blocking git:

```
git clone http://git.buildroot.net/git/buildroot.git
```

Please use the native git protocol if at all possible, as it's a lot more efficient than HTTP.

If you are not already familiar with using Git, we recommend you visit [the Git website](#).

Once you've checked out a copy of the source tree, you can update your source tree at any time so it is in sync with the latest and greatest by entering your buildroot directory and running the command:

```
git pull
```

Because you've only been granted anonymous access to the tree, you won't be able to push your changes to the repo. Changes can instead be submitted for inclusion by posting them to the buildroot mailing list or to the [Bug and Patch Tracking System](#).

About
[Latest News](#)
[Download](#)
[Browse Source](#)
[Accessing Source](#)
[Bug Tracking](#)
[Documentation](#)
[Mailing Lists](#)
[Autobuilder](#)
[Patchwork](#)

Related Sites
[BusyBox](#)
[uClibc.org](#)
[uClibc++](#)
[udhcp](#)
[Scratchbox](#)
[OpenEmbedded](#)
[uCdot](#)
[LinuxDevices](#)
[Slashdot](#)
[Freecode](#)
[Linux Today](#)
[Linux Weekly News](#)
[Linux HOWTOs](#)

Copyright © 1999-2005 Erik Andersen, 2006-2012 The Buildroot developers
Mail all comments, insults, suggestions and bribes to
The Buildroot developers buildroot@uclibc.org

git clone

```
$ git clone http://git.buildroot.net/git/buildroot.git
```

- Permet de copier à l'identique le dépôt distant sur sa machine.
- Les tags, branches et l'intégralité de tous les commits sont récupérés.

Seveurs distants

```
$ git remote -v  
origin git://git.buildroot.net/buildroot (fetch)  
origin git://git.buildroot.net/buildroot (push)
```

Récapitulatif des commandes

- push : action d'envoyer les commits ainsi que les relations entre commits vers le serveur distant (origin par défaut).
- fetch : action de récupérer les commits ainsi que les relations entre commits depuis le serveur distant.

Ajout d'un serveur

```
$ git remote add forge3 git@forge3:buildroot  
$ git remote -v  
origin git@forge3:buildroot (fetch)  
origin git@forge3:buildroot (push)
```

Checkout

```
$ git checkout (tag ou sha1 ou branch)
```

Git présente sur le système de fichier la version qu'on lui demande.

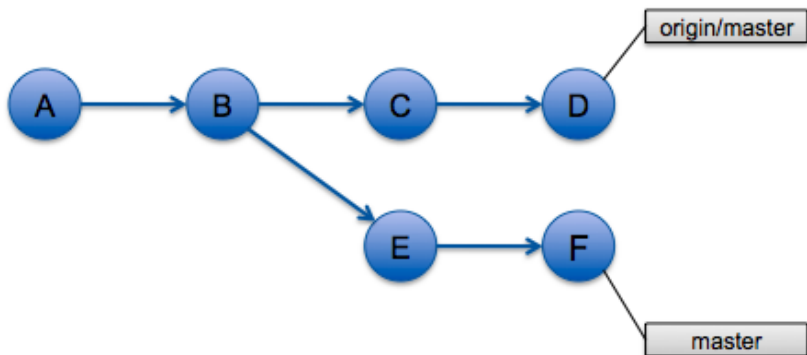
Pull/Push

```
$ git pull
```

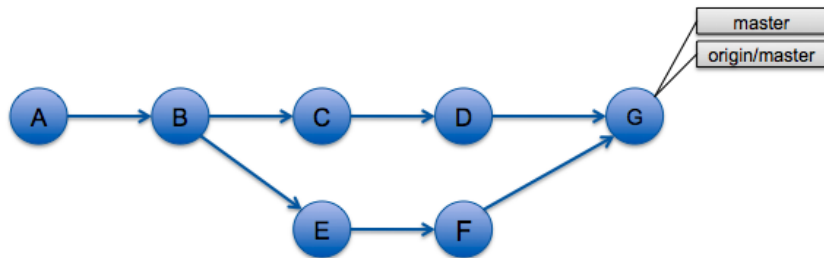
```
$ git pull --rebase
```

- Git récupère les commits distant et merge la branche actuelle avec la branche distante.
- Par défaut git merge les branches lors d'un pull.
- Rebase la branche actuelle avec la branche distante.

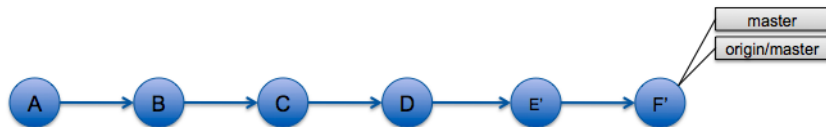
Merge/Rebase



Merge/Rebase



Merge/Rebase



Pour aller plus loin

```
$ git stash  
$ git reflog  
$ git reset
```

Credits

- <http://blog.octo.com/git-dans-la-pratique-22/>
- pour récupérer la formation:
- `git clone https://github.com/naguirre/formation-git.git`