

Un gestionnaire de versions
distribué : Git

Pourquoi un gestionnaire de version ?

- Travailler à plusieurs sur le même code
- Garder un historique de tout ce qui a été fait
- Revenir rapidement à une ancienne version
- Faciliter la sauvegarde du code source

On faisait comment avant Git ?

- Système D: copies du code source, FTP, clé USB, email, ...

Chapitre I^{er}

Dispositions relatives au mariage

I. – Le chapitre I^{er} du titre V du livre I^{er} du code civil est ainsi modifié :

1° Il est rétabli un article 143 ainsi rédigé :

« Art. 143. – Le mariage est contracté par deux personnes de sexe différent ou de même sexe. » ;

2° L'article 144 est ainsi rédigé :

« Art. 144. – Le mariage ne peut être contracté avant dix-huit ans révolus. » ;

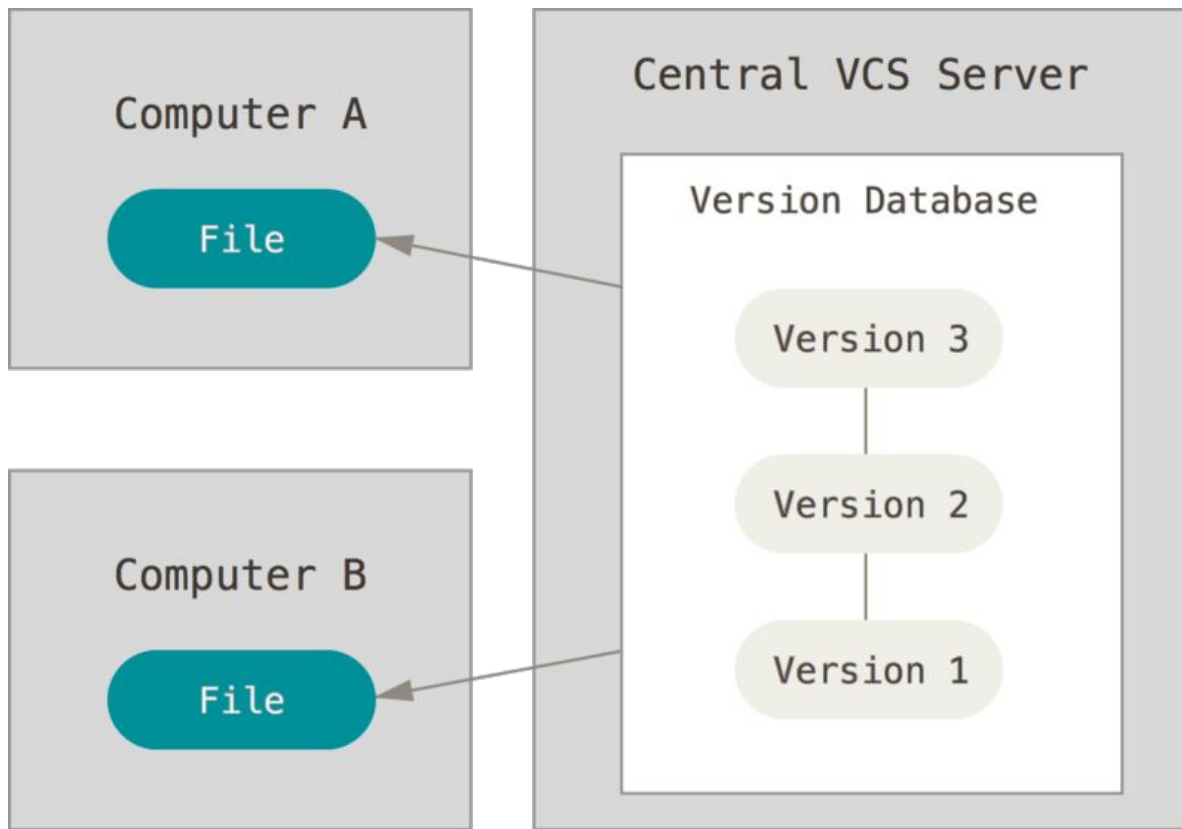
3° L'article 162 est complété par les mots : « , entre frères et entre sœurs » ;

4° L'article 163 est ainsi rédigé :

« Art. 163. – Le mariage est prohibé entre l'oncle et la nièce ou le neveu, et entre la tante et le neveu ou la nièce. » ;

On faisait comment avant Git ?

- Système D: FTP, clé USB, email, ...
- Gestionnaires de version centralisés: SVN, CVS, Perforce

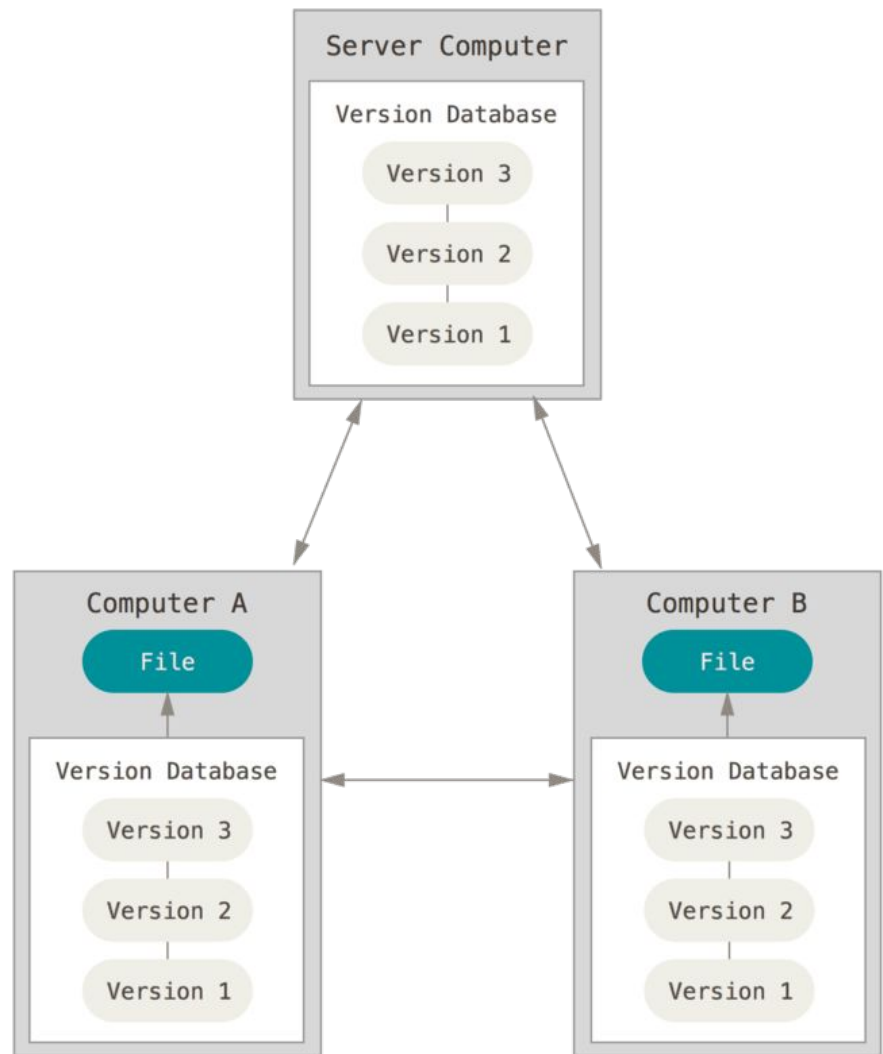


Fonctionnement d'un système de gestion de version centralisé.

Source: <https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control>

On faisait comment avant Git ?

- Système D: FTP, clé USB, email, ...
- Gestionnaires de version centralisés: SVN, CVS, Perforce
- Gestionnaires de version décentralisés: BitKeeper, Bazaar



Fonctionnement d'un système de gestion de version décentralisé.

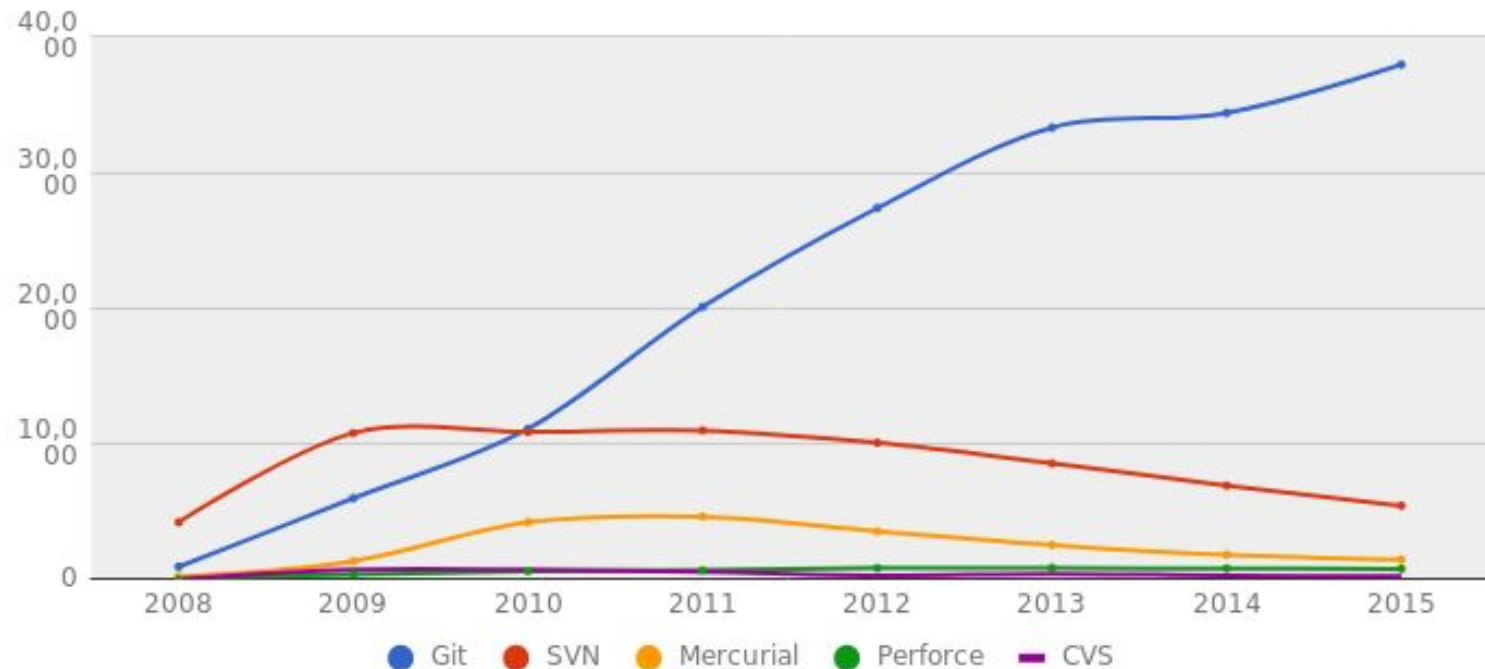
Source:

<https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control>

Un peu d'histoire

- Développé commencé par Linus Torvalds en Avril 2005
- Conçu pour gérer les sources du noyau Linux
- Popularisé par la sortie de Github en 2008
- Très populaire dans l'Open Source
- VCS (Version Control Software) le plus utilisé dans le monde en 2018

Questions on Stack Overflow, by Year

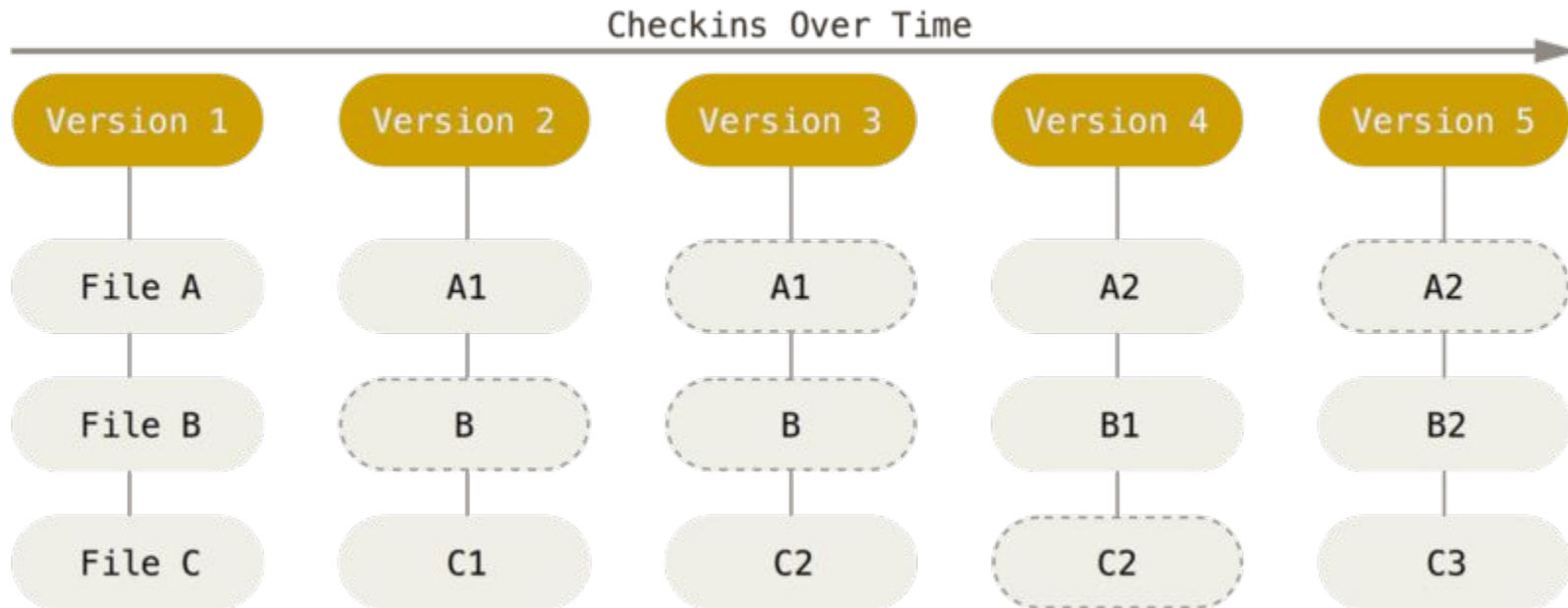


Nombre de questions Stackoverflow à propos des principaux VCS entre 2008 et 2015.

Source: <https://rhodecode.com/insights/version-control-systems-2016>

Fondamentaux de Git: it's only snapshots!

- Une version = un snapshot complet des sources



Stockage des sources sous la forme de snapshots

Source: <https://git-scm.com/book/en/v2/Getting-Started-Git-Basics>

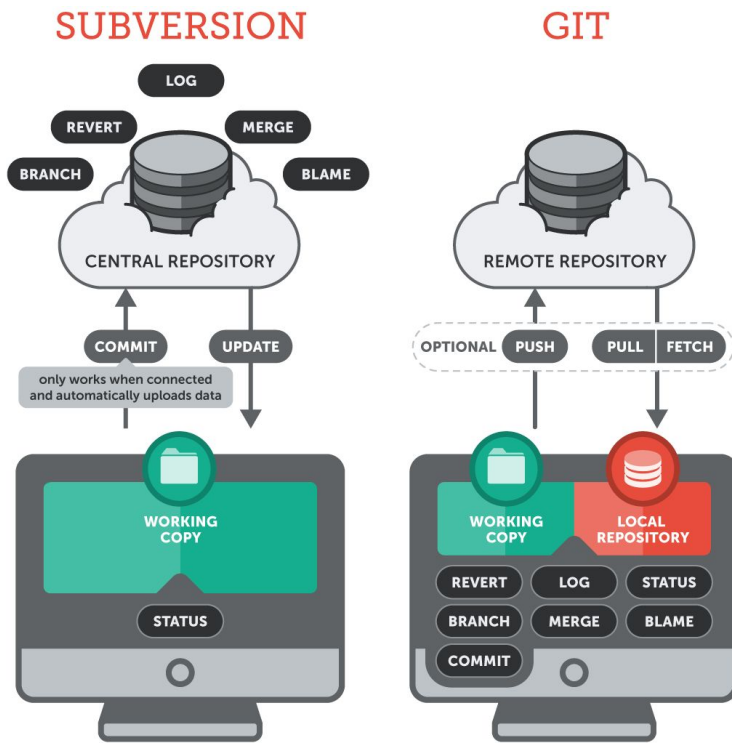
Fondamentaux de Git: it's only patches!

- Conserve la différence ligne par ligne entre les fichiers

```
--- a/source/css/_partial/article.styl
+++ b/source/css/_partial/article.styl
@@ -125,7 +125,7 @@
     max-width: 240px
     max-height: 96px
     display: block
-    margin-right: 12px
+    margin-left: 12px
     margin-top: 12px
     float: right
     clear: right
Stage this hunk [y,n,q,a,d,/,j,J,g,e,?]? y
@@ -379,4 +379,4 @@ $article-gallery-ctrl
     right: 0
     &:before
       content: "\f054"
       right: 15px
\ No newline at end of file
right: 15px
```

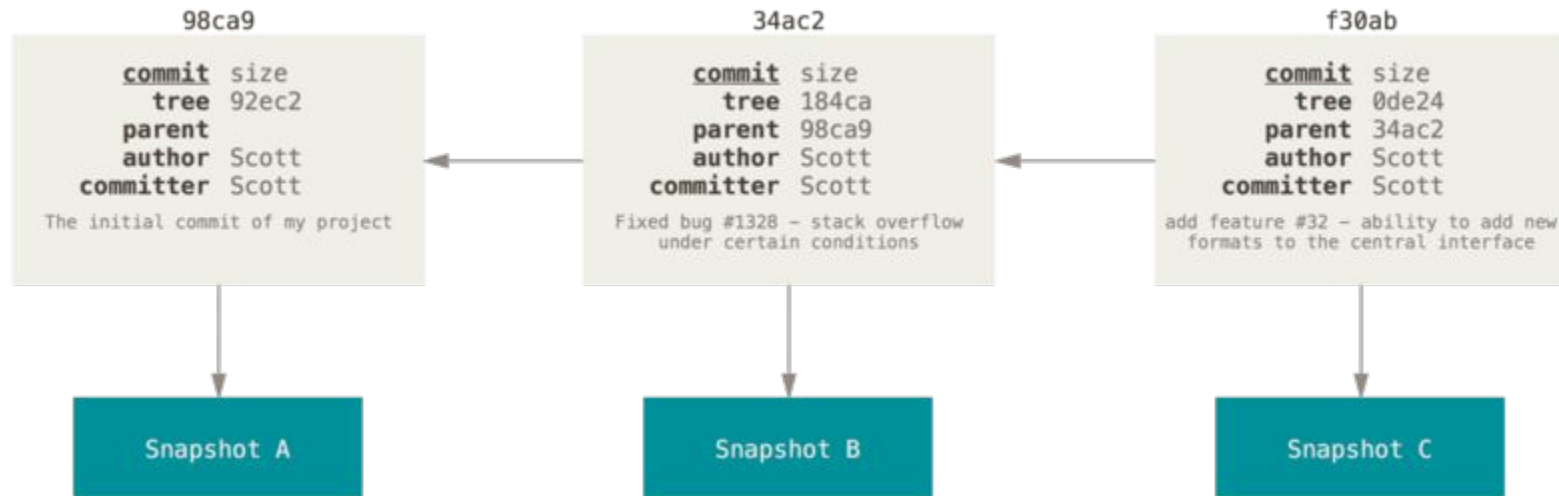
Fondamentaux de Git: it's only local!

- La majorité des opérations se fait uniquement en local



Fondamentaux de Git: it's maintaining integrity!

- Utilisation d'une somme de contrôle (Hash SHA1) pour assurer l'intégrité de l'arbre des modifications



Visualisation de l'arbre des commits

Source: <https://git-scm.com/book/en/v2/Git-Branching-Branches-in-a-Nutshell>

Initialisation d'un dépôt

- Création en local avec *git init*

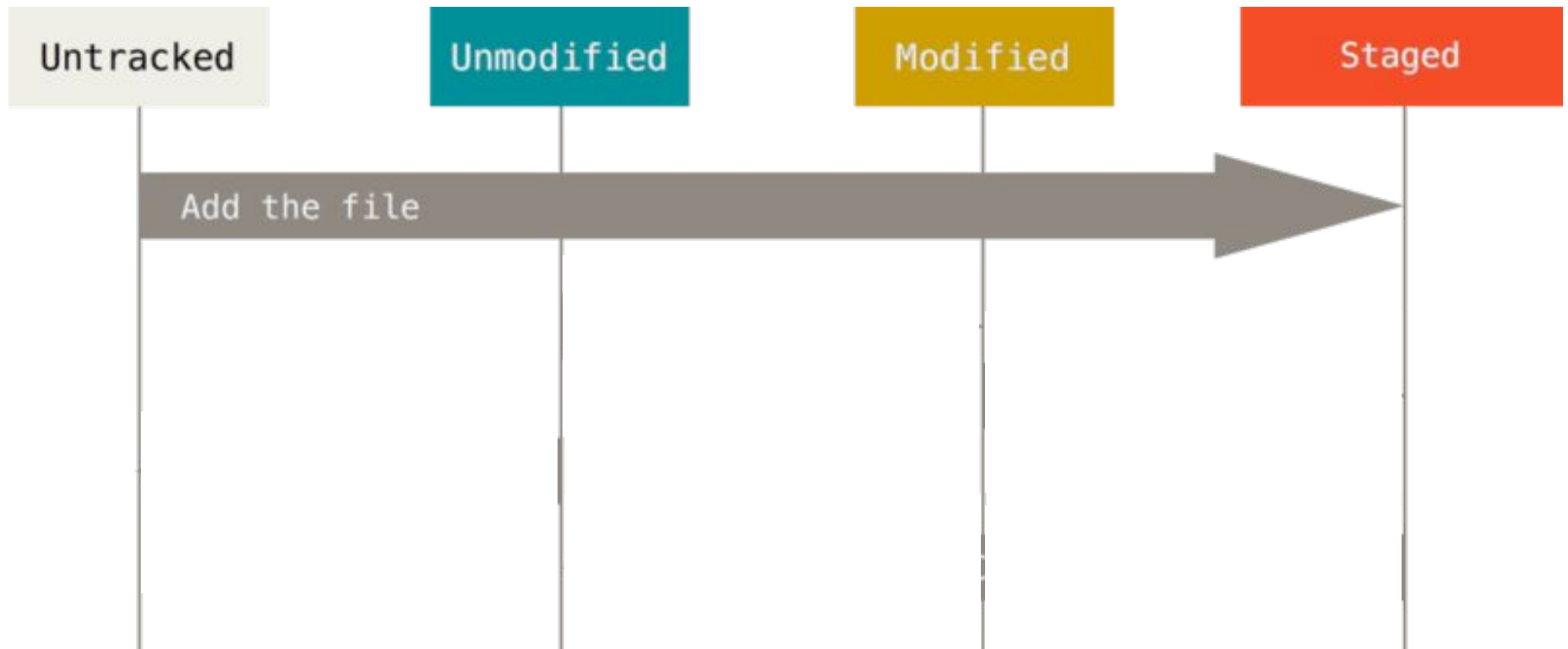
```
aschen@gman perso/decentralized_vcs » git status
fatal: Not a git repository (or any parent up to mount point /home/shadow)
Stopping at filesystem boundary (GIT_DISCOVERY_ACROSS_FILESYSTEM not set).
aschen@gman perso/decentralized_vcs » git init .
Initialized empty Git repository in /home/shadow/projets/perso/decentralized_vcs/.git/
aschen@gman perso/decentralized_vcs (master) » git status
On branch master

Initial commit

nothing to commit (create/copy files and use "git add" to track)
```

Le cycle de vie des fichiers

- 2 types de fichiers: ***tracked*** et ***untracked***
- 3 états pour les fichiers suivis (***tracked***) par Git:
 - non-modifié (***unmodified***): fichier n'ayant pas subi de modifications
 - modifié (***modified***): fichier ayant été modifié
 - indexé (***staged***): fichier dont les modifications seront sauvegardées au prochain ***commit***



Les différents états du cycle de vie d'un fichier géré par Git

Source: <https://git-scm.com/book/en/v2/Git-Basics-Recording-Changes-to-the-Repository>

Le cycle de vie des fichiers

- 2 types de fichiers: ***tracked*** et ***untracked***
- 3 états pour les fichiers suivis (***tracked***) par Git:
 - non-modifié (***unmodified***): fichier n'ayant pas subi de modifications depuis le dernier **commit**
 - modifié (***modified***): fichier ayant été modifié
 - indexé (***staged***): fichier dont les modifications seront sauvegardées au prochain **commit**
- Connaître l'état actuel du dépôt: ***\$ git status***

```
aschen@gman perso/decentralized-vcs (master *+%) » git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   README.md

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   ex01/.keep

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        ex02/
```

Résultat de la commande **git status**

Enregistrer des modifications

- Modifier son fichier

Working
directory

Staging
area

Git
repository

```
merge 4138180 e9aaf6b
Author: Kevin Blondel <kblondel@kalliope.com>
Date:   Tue Nov 13 10:23:40 2018 +0100

    Merge pull request #1214 from kuzileio/kzl-832-document-count-body-optional

    [kzl-832] Document.count - body is optional

commit e9aaf6b82dff05664d92b18074ced7e771d63397
Author: Benoit Vidis <contact@benoitvidis.com>
Date:   Mon Nov 12 11:35:19 2018 +0100

    [kzl-832] Document.count - body is optional

author: Benoit Vidis <contact@beno.fr>
date:   Thu Oct 25 16:13:24 2018 +0200

    document.validate should not require an id (#1208)

author: Alexandre Bouthinon <bouthinon.alexandre@gmail.com>
date:   Tue Oct 23 13:44:04 2018 +0200

    [kzl-832] Fix API deployment
```

Historique des commits par la commande **git log**

TD: Comprendre les différentes zones

- Initialiser un dépôt
 - Créer un nouveau fichier
 - Afficher git status
 - Ajouter le fichier à la zone de staging
 - Afficher git status
 - Commit le fichier
 - Afficher git status
- Modifier le fichier
 - Ajouter le fichier à la zone de staging
 - Annuler les modifications
-
- Modifier le fichier
 - Ajouter le fichier à la zone de staging
 - Commit le fichier
 - Annuler le commit mais conserver les modifications
 - Annuler les modifications

Comment faire un bon commit ?

- Garder le même contexte pour les modifications entrant dans le commit



The image shows a screenshot of a code editor with a commit message being written. The commit message is "beam ssa dead: Speed up optimization of switch instructions". A semi-transparent overlay box is positioned over the commit message, containing the text "When pushed, this commit will " followed by a blank line and a closing quote. Below this, a purple arrow points down to a list of guidelines: "✓ DO Complete the sentence with your commit message" and "✗ DONT Use WIP, Fix, Update". The background shows several commit messages in a list, including "ugly fix", "toast tak", "add app", and "qr code erreur", all by "Aschen" and dated "Jul 3, 2015".

When pushed, this commit will " _____ "

↓

✓ DO Complete the sentence with your commit message

✗ DONT Use WIP, Fix, Update



Revenir en arrière

- Enlever un fichier des fichiers **staged**
 - `$ git reset HEAD <fichier>` (*<fichier> ne sera pas intégré au prochain commit*)

Untracked

Unmodified

Modified

Staged

\$ git add <fichier>

\$ git reset HEAD <fichier>

*Enlever un fichier de la zone de staging avec **git reset HEAD <fichier>***

```
aschen@gman perso/decentralized-vcs (master *) » git add README.md
aschen@gman perso/decentralized-vcs (master +) » git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   README.md

aschen@gman perso/decentralized-vcs (master +) » git reset HEAD README.md
Unstaged changes after reset:
M       README.md
aschen@gman perso/decentralized-vcs (master *) » git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
aschen@gman perso/decentralized-vcs (master *) » █
```

Enlever un fichier de la zone de staging avec **git reset HEAD <fichier>**



Revenir en arrière

- Retirer un fichier des fichiers du prochain commit (**staged => modified**)
 - `$ git reset HEAD <fichier>`
- Annuler les modifications faites sur un fichier (**modified => unmodified**)
 - `$ git checkout -- <fichier>`

Untracked

Unmodified

Modified

Staged

\$ git checkout -- <fichier>

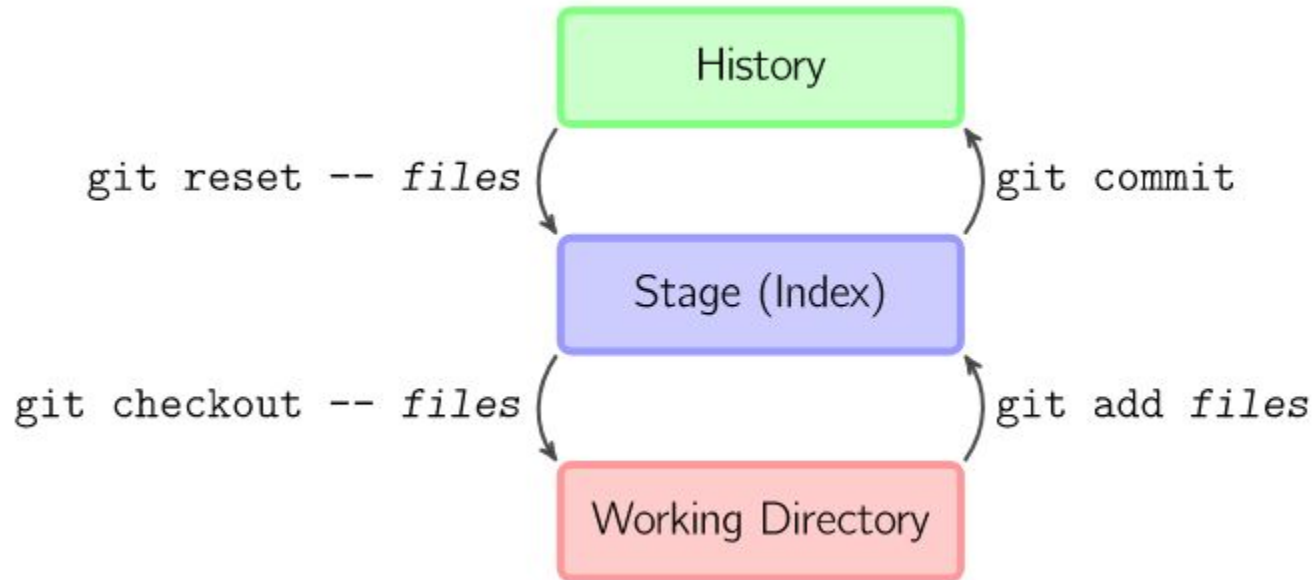
*Enlever un fichier de la zone de staging avec **git reset HEAD <fichier>***

```
aschen@gman perso/decentralized-vcs (master *) » cat README.md
\# Decentralized Version Control Systems
Hey
aschen@gman perso/decentralized-vcs (master *) » git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
aschen@gman perso/decentralized-vcs (master *) » git checkout -- README.md
aschen@gman perso/decentralized-vcs (master) » cat README.md
\# Decentralized Version Control Systems
aschen@gman perso/decentralized-vcs (master) » git status
On branch master
nothing to commit, working directory clean
```

Annuler les modifications faites sur un fichier avec **git checkout -- <fichier>**



Résumé des processus d'ajout de fichiers et des retours en arrière possibles

Source: <https://marklodato.github.io/visual-git-guide/index-fr.html>

Développer en parallèle avec les branches

- Une branche est un historique de commits qui se suivent
 - Par défaut, les commits sont sur la branche **master**
 - Chaque branche avance de manière indépendante
- Création d'une branche à partir du commit courant
 - ***\$ git branch <nom de la branche>***
- Utiliser une branche
 - ***\$ git checkout <nom de la branche>***
- Créer + utiliser une branche
 - ***\$ git checkout -b <nom de la branche>***

Free Explore

Have fun!

Local Repository

HEAD: master

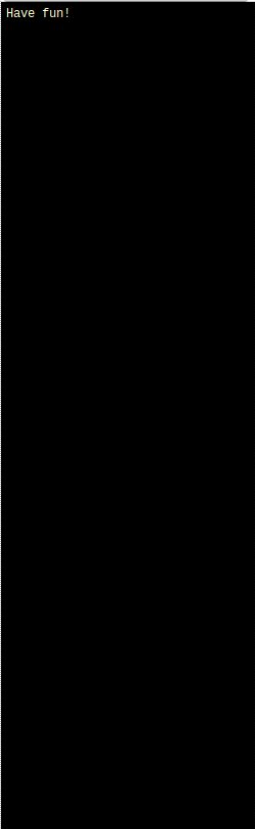


\$ enter git command

Se déplacer dans l'arborescence des commits

- Emplacement courant symbolisé par **HEAD**
- Déplacer la **HEAD**
 - `$ git checkout <branche|commit>`
- Déplacer la **HEAD** relativement
 - `$ git checkout <branche|commit>^` : un commit en arrière
 - `$ git checkout <branche|commit>~2` : deux commits en arrière

Free Explore



\$ enter git command

Local Repository

HEAD: master



Annuler des commits: git reset

- Modifications locales seulement
 - `$ git reset <commit>`
 - `$ git reset HEAD~1`
- **Conserve les modifications mais supprime les commits**
- Utilisable sur un fichier à la fois
 - `$ git reset HEAD <fichier>`

```
aschen@gman perso/decentralized-vcs (master) » cat README.md
\# Decentralized Version Control Systems
Hey
aschen@gman perso/decentralized-vcs (master) » git log --pretty=oneline | head
f8479a6ef61f70d675cf900c6d50019f955e9301 Add hey to README.md
d8a2cel677a092c0a0ce486cc91088d747c5f262 add exo2
4b1c40b4a0b853ca8b3e7fe100e8aa9656f318b9 work
91ee56ed7af5cbb6262f4a6f94fa0ac35b0f889b first commit
aschen@gman perso/decentralized-vcs (master) » git reset --soft d8a2ce
aschen@gman perso/decentralized-vcs (master +) » git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   README.md

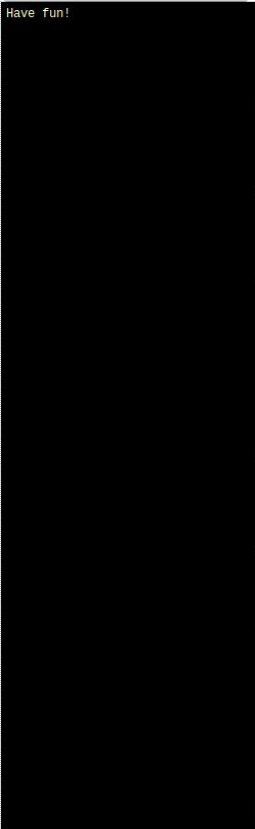
aschen@gman perso/decentralized-vcs (master +) » git reset HEAD README.md
Unstaged changes after reset:
M       README.md
aschen@gman perso/decentralized-vcs (master *) » git checkout -- README.md
aschen@gman perso/decentralized-vcs (master) » cat README.md
\# Decentralized Version Control Systems
```

Revenir à un état antérieur du dépôt avec **git reset --soft <commit>**

```
aschen@gman perso/decentralized-vcs (master) » cat README.md
\# Decentralized Version Control Systems
Hey
aschen@gman perso/decentralized-vcs (master) » git log --pretty=oneline | head
46dd282d2ebc3ddd91ad6483f0a4a69ac75afb4f Add hey to README.md
d8a2ce1677a092c0a0ce486cc91088d747c5f262 add exo2
4b1c40b4a0b853ca8b3e7fe100e8aa9656f318b9 work
91ee56ed7af5cbb6262f4a6f94fa0ac35b0f889b first commit
aschen@gman perso/decentralized-vcs (master) » git revert 46dd282
[master a0a0f1d] Revert "Add hey to README.md"
 1 file changed, 1 deletion(-)
aschen@gman perso/decentralized-vcs (master) » cat README.md
\# Decentralized Version Control Systems
aschen@gman perso/decentralized-vcs (master) » git log --pretty=oneline | head
a0a0f1da7ce362ce59b34f08946edea2d060c68a Revert "Add hey to README.md"
46dd282d2ebc3ddd91ad6483f0a4a69ac75afb4f Add hey to README.md
d8a2ce1677a092c0a0ce486cc91088d747c5f262 add exo2
4b1c40b4a0b853ca8b3e7fe100e8aa9656f318b9 work
91ee56ed7af5cbb6262f4a6f94fa0ac35b0f889b first commit
```

Utilisation de **git revert <commit>** pour annuler un commit tout en conservant l'historique

Free Explore



\$ enter git command

Local Repository

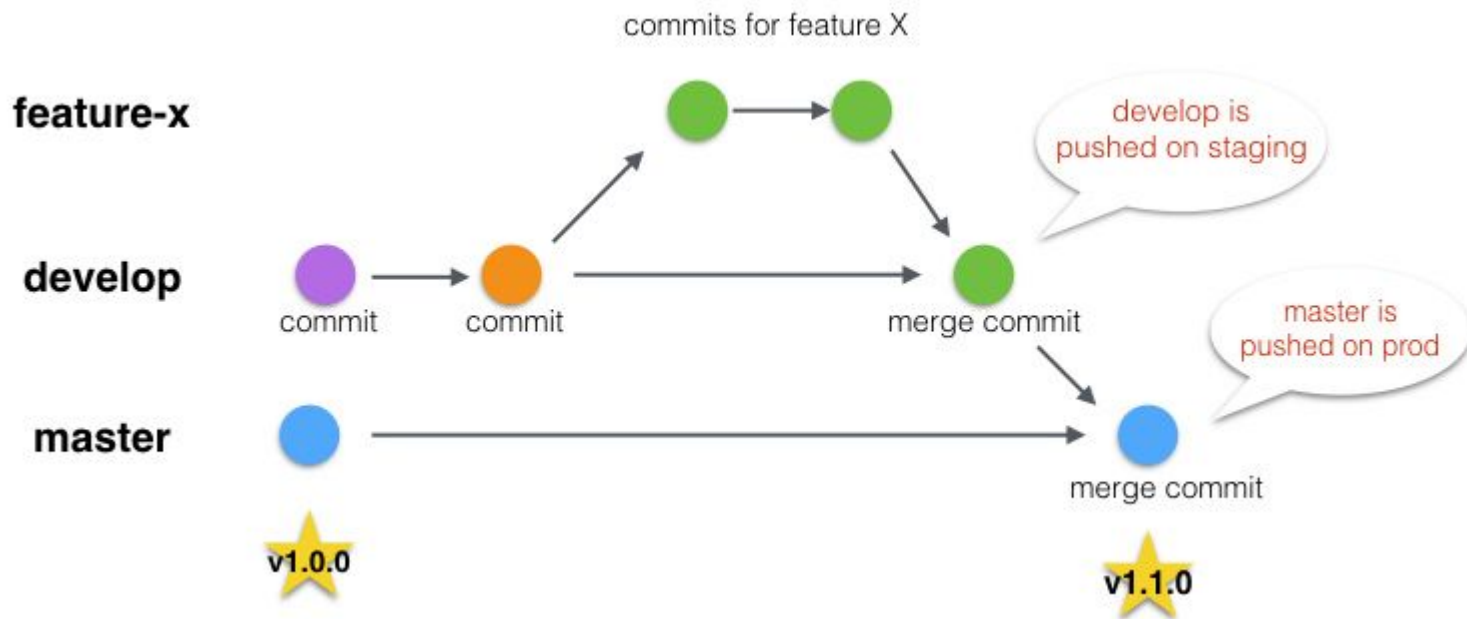
HEAD: master



Fusion de branches



- Une branche = une fonctionnalité en cours de développement



Free Explore

Have fun!

Local Repository

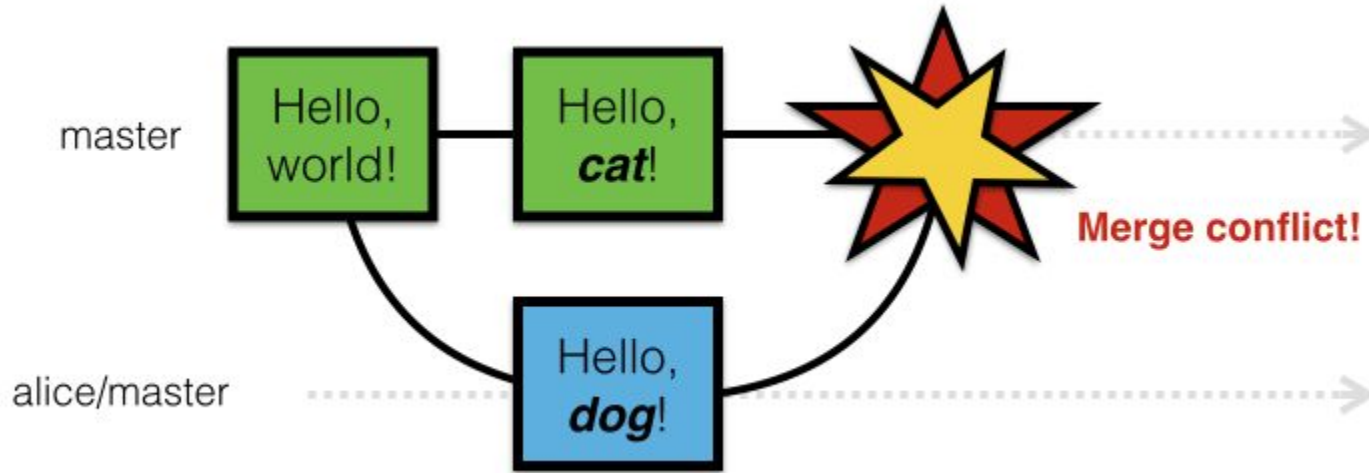
HEAD: master



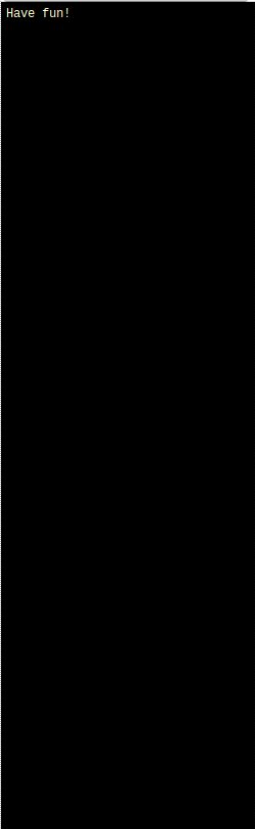
\$ enter git command

Résoudre les conflits de fusion

- Apparaissent lorsqu'une même ligne est modifiée sur les deux branches



Free Explore



\$ enter git command

Local Repository

HEAD: master



```
aschen@gman perso/decentralized-vcs (change-readme) » git log --pretty=oneline | head
015b5a48837b242175679af1e4697108d3207734 Add hello to README
46dd282d2ebc3ddd91ad6483f0a4a69ac75afb4f Add hey to README.md
d8a2cel677a092c0a0ce486cc91088d747c5f262 add exo2
4b1c40b4a0b853ca8b3e7fe100e8aa9656f318b9 work
91ee56ed7af5cbb6262f4a6f94fa0ac35b0f889b first commit
aschen@gman perso/decentralized-vcs (change-readme) » cat README.md
\# Decentralized Version Control Systems
Hello
aschen@gman perso/decentralized-vcs (change-readme) » git checkout master
Switched to branch 'master'
aschen@gman perso/decentralized-vcs (master) » git log --pretty=oneline | head
020f87cece5488f403cc0531d6fb8c7daf49b138 Add hi to README
46dd282d2ebc3ddd91ad6483f0a4a69ac75afb4f Add hey to README.md
d8a2cel677a092c0a0ce486cc91088d747c5f262 add exo2
4b1c40b4a0b853ca8b3e7fe100e8aa9656f318b9 work
91ee56ed7af5cbb6262f4a6f94fa0ac35b0f889b first commit
aschen@gman perso/decentralized-vcs (master) » cat README.md
\# Decentralized Version Control Systems
Hi
```

Exemple d'un conflit entre deux branche.

Ici chaque branche a avancé d'un commit ayant modifié la même ligne du même fichier.

```
aschen@gman perso/decentralized-vcs (master) » git merge change-readme
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.
aschen@gman perso/decentralized-vcs (master ?) » cat README.md
\# Decentralized Version Control Systems
<<<<<< HEAD
Hi
=====
Hello
>>>>>> change-readme
aschen@gman perso/decentralized-vcs (master ?) » git status
On branch master
You have unmerged paths.
  (fix conflicts and run "git commit")

Unmerged paths:
  (use "git add <file>..." to mark resolution)

        both modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
```

Exemple d'un conflit entre deux branches.

Ici Git nous indique à quel endroit il y a un conflit pour que nous puissions le résoudre.

```

aschen@gman perso/decentralized-vcs (master ?) » cat README.md
\# Decentralized Version Control Systems
Hello
aschen@gman perso/decentralized-vcs (master ?) » git add README.md
aschen@gman perso/decentralized-vcs (master +) » git commit -m "merge ok"
[master 04c0fcc] merge ok
aschen@gman perso/decentralized-vcs (master) » git log | head -n 20
commit 04c0fccf22ea77abd5e8dfb95f58715fdeb502e
Merge: 020f87c 015b5a4
Author: Aschen (Adrien Maret) <amaret93@gmail.com>
Date: Sat Nov 17 17:23:43 2018 +0100

    merge ok

commit 020f87cece5488f403cc0531d6fb8c7daf49b138
Author: Aschen (Adrien Maret) <amaret93@gmail.com>
Date: Sat Nov 17 17:18:59 2018 +0100

    Add hi to README

commit 015b5a48837b242175679af1e4697108d3207734
Author: Aschen (Adrien Maret) <amaret93@gmail.com>
Date: Sat Nov 17 17:18:44 2018 +0100

    Add hello to README

```

Merge commit

Commit de la branche *master*

Commit de la branche *change-readme*

Exemple d'un conflit entre deux branches.

Ici nous avons résolu le conflit avec un **merge commit**.

L'historique contient bien tous les commits de chaque branche

1. Créer un fichier avec plusieurs lignes de contenu
2. Changer de branche, modifier une des ligne puis commiter
3. Retourner sur master
4. Changer de branche, modifier la même ligne puis commiter
5. Retourner sur master
6. Merge la première puis la deuxième branche dans master
7. Comprendre et résoudre le conflit