Introduction à Git

EPFL Dojo - workshop git

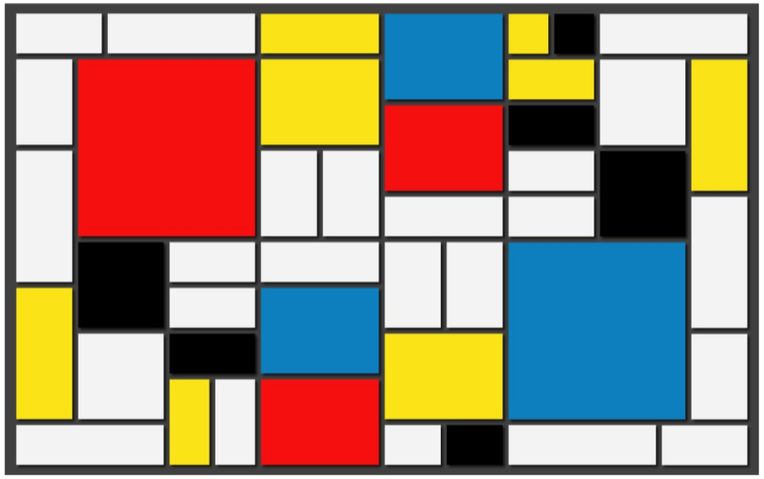
Lausanne, 27 janvier 2017

Nicolas Borboën <nicolas.borboen@epfl.ch>

Image Atlassian: https://www.atlassian.com/git/images/atlassian-getting-git-right.jpg

Git est un logiciel de gestion de versions décentralisé.

Mondrian vs Pollock

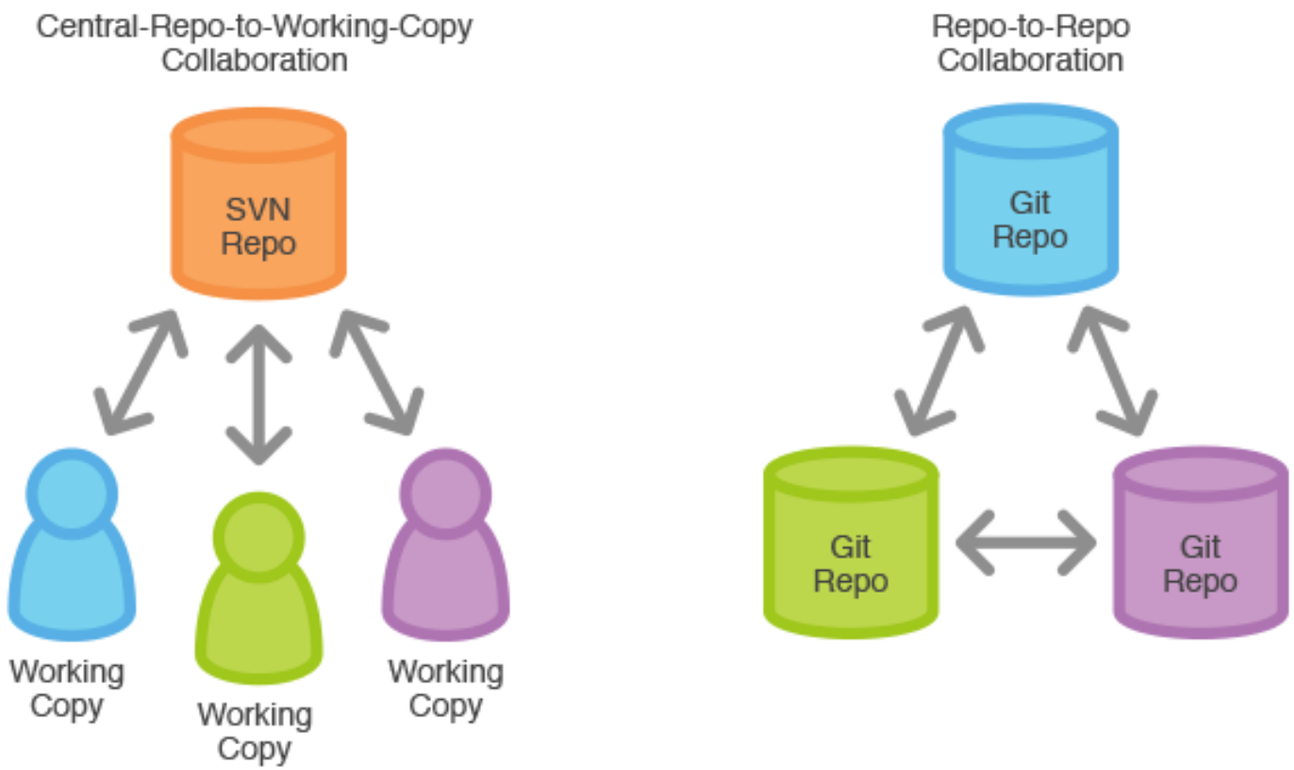


Cathedral vs Bazaar

SVN, CVS, Perforce (centralisés) vs Git, Mercurial, Bazaar (décentralisés)

Git est utile (gestion de versions)

Décentralisé ?



Utilisation locale

> La plupart des opérations de Git ne nécessitent que des fichiers et ressources locaux [...] Git vous fera penser que les dieux de la vitesse ont octroyé leurs pouvoirs à Git.

💡 git-scm.com - Rudiments de Git

Les tripes de Git

- **objects** → stockage du contenu `find .git/objects -type f`
- **tree** → répertoire
- **blob** → fichier (Binary Large OBject, soit en français : Gros Objet Binaire)
- **commit** → instantané
- **tag** → une étiquette pointant toujours sur un même commit
- **refs** → méchanisme interne pour représenter les branches et les tags

  💡 https://git-scm.com/book/fr/v1/Les-tripes-de-Git-Les-objets-Git

Utilisation (initialisation)

- Initialisation d’un dépôt Git dans un répertoire existant

  ```
  $ git init
  ```

  Cela crée un nouveau sous-répertoire nommé .git qui contient tous les fichiers nécessaires au dépôt — un squelette de dépôt Git.

- Récupération d'un dépôt distant dans un dépôt local

  ```
  $ git clone $URI
  ```

💡 https://git-scm.com/book/fr/v2/Les-bases-de-Git-Démarrer-un-dépôt-Git

Utilisation (modifications dans le dépôt)

- Placer de nouveaux fichiers sous suivi de version

  ```
  $ git add .
  ```

- Valider vos modifications

  ```
  $ git commit
  ```

  - il ne concerne qu'une chose et une seule ;
  - il est le plus petit possible (tout en restant cohérent).

- Effacer des fichiers

  ```
  $ git rm
  ```

- Déplacer des fichiers

  ```
  $ git mv
  ```

  💡 https://git-scm.com/book/fr/v2/Les-bases-de-Git-Enregistrer-des-modifications-dans-le-dépôt

Utilisation (état)

- Vérifier l’état des fichiers

  ```
  $ git status
  ```

- Visualiser l’historique des validations

  ```
  $ git log .
  ```

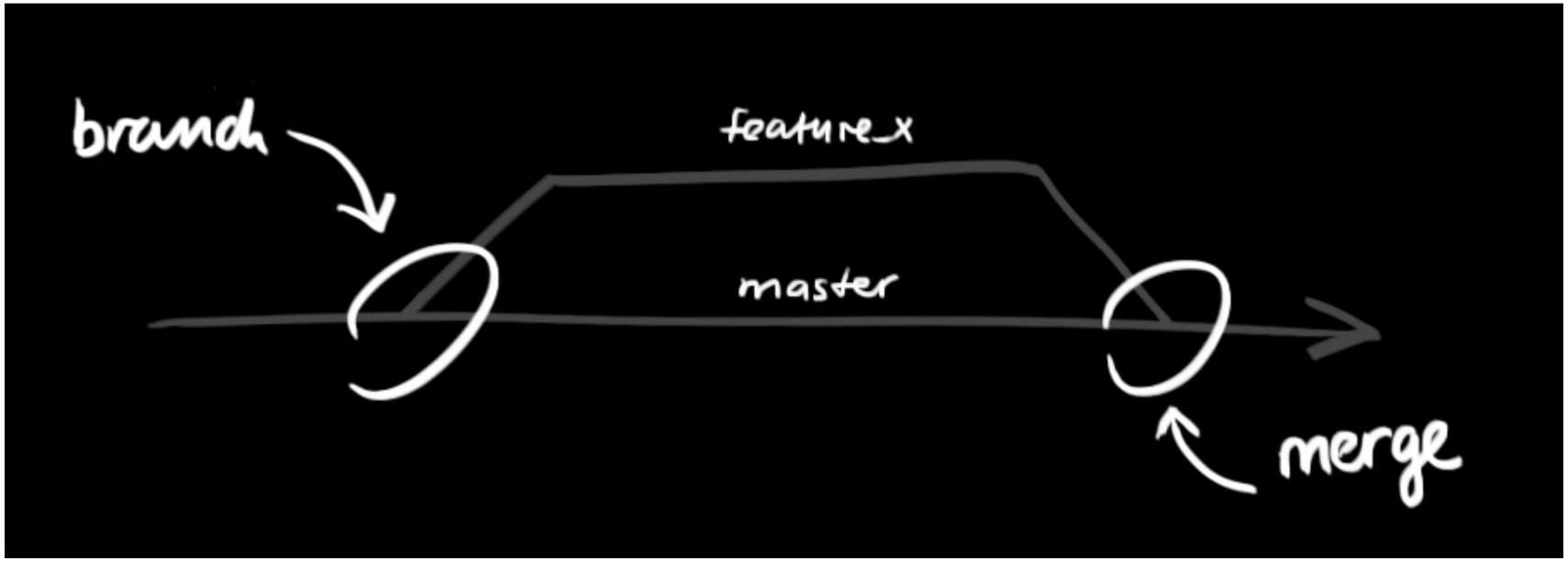
  - https://coderwall.com/p/euwpig/a-better-git-log

💡 https://git-scm.com/book/fr/v2/Les-bases-de-Git-Visualiser-l’historique-des-validations

Utilisation (branches)

Une branche n'est qu'une étiquette qui pointe vers un commit.

Elle permet d'avoir des environnements de développement facilement (feature, bug fix)



💡 https://git-scm.com/book/fr/v1/Les-branches-avec-Git-Travailler-avec-les-branches

Utilisation (flow)

L'utilisation des branches peut-être liées à des bonnes pratiques, par exemple :

- https://leanpub.com/git-flow/read
- http://nvie.com/posts/a-successful-git-branching-model/

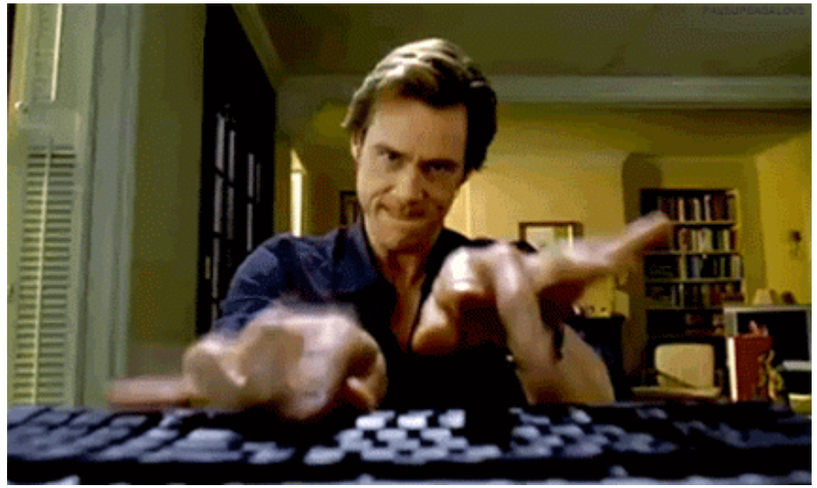
Utilisation (conflits)

Lors qu'un fichier est modifié par un autre utilisateur, ou que vous voulez fusionner deux branches, il arrive qu'il soit nécessaire de résoudre des conflits.

Afin de pouvoir résoudre les conflits visuellement, il existe des outils adéquats, par exemple Meld.
Pour définir l'outils utilisé pour résoudre les conflits, utiliser la commande:

```
$ git config merge.tool meld
```

Pratique



Pratique

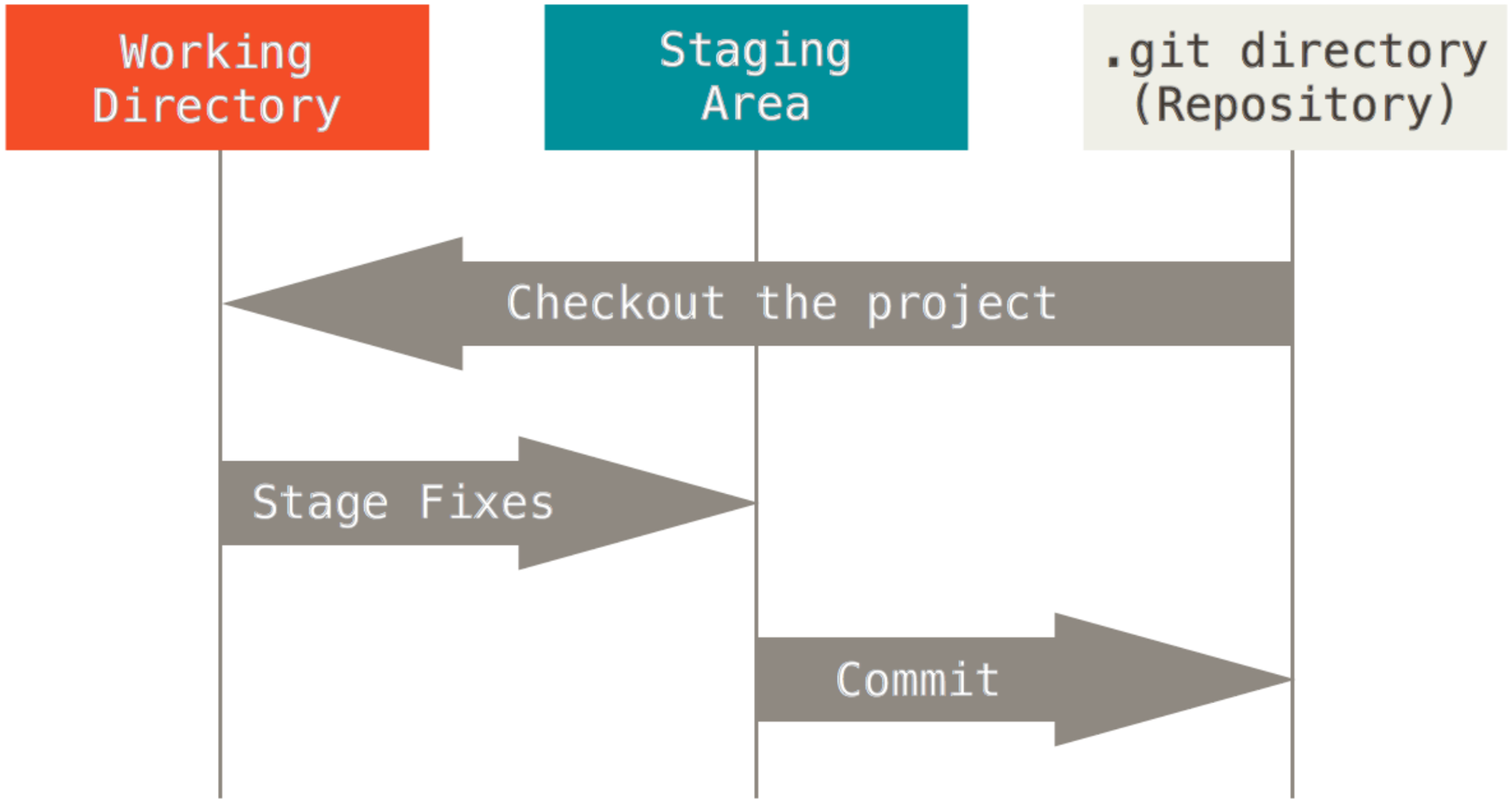
1. Installation git 💡 doc
2. Configuration de l'utilisateur 💡 doc

```
$ git config --global user.name "John Doe"
```

```
$ git config --global user.email johndoe@example.com
```

- Créer un dépôt local `$ git init` dans un répertoire
- Manipulation:
  - création fichier, édition fichier ;
  - `$ git add fichier` "Staging aera"
  - `$ git commit -m"mon message de log" fichier` "Repository aera"
- Utiliser `$ git status` entre les étapes

Pratique


Working Directory
Staging Area
.git directory (Repository)
Checkout the project
Stage Fixes
Commit

Pratique

- Créer un conflit
  - Modifier un fichier et le commiter
  - Création d'une branche sur le répertoire courant et switcher dessus

    `$ git checkout -b maBranche`
  - Editer les mêmes lignes du fichier dans la branche et le commiter
  - Retrouner sur la branche `master` : `$ git checkout master`
  - Fusionner `maBranche` dans `master` : `$ git merge maBranche`
  - Identifier le conflit avec `$ git status`
  - Résoudre le conflit, par exemple avec Meld

Pratique (conflits)

```
~/gitTest(branch:master) » git merge maBranche
Auto-merging fichier
CONFLICT (content): Merge conflict in fichier
Automatic merge failed; fix conflicts and then commit the result.
```

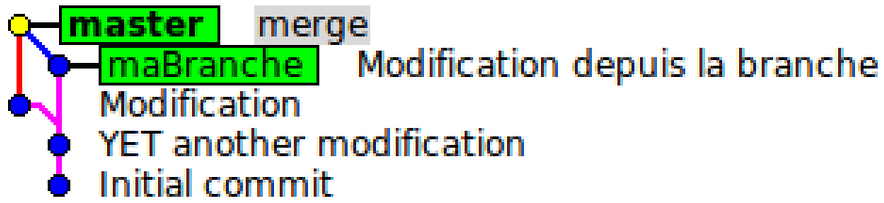
- Ouvrir l'outil de résolution de conflits :

  `$ git mergetool`
- Résoudre les conflits (le fichier central est le fichier final)
- Utiliser `$ git add fichier` et `$ git commit -m"Merge"`

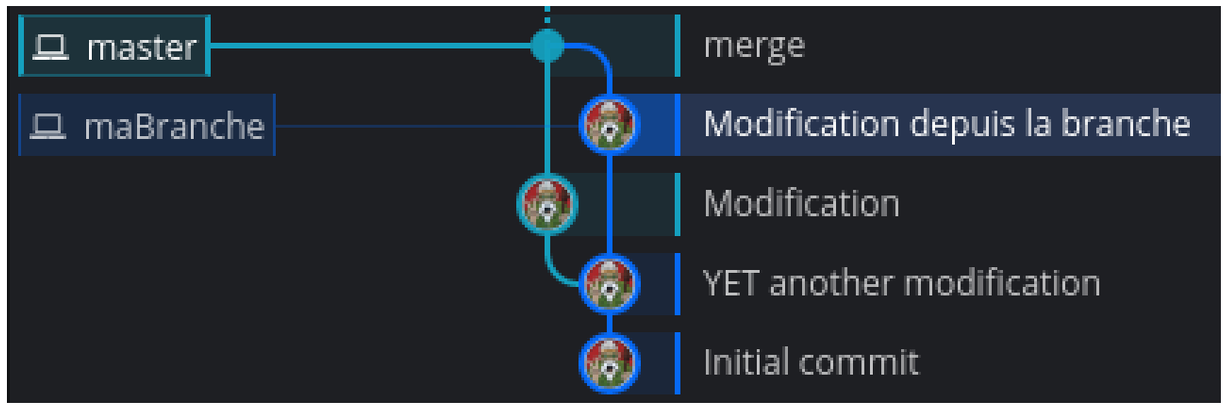
  ⇒ conflits résolus

Pratique (conflits)

- Utiliser une commande pour voir le graph des commits, par exemple:
  - `$ git log --graph` ou `$ git log --graph --oneline --decorate --all`
  - `$ gitk`



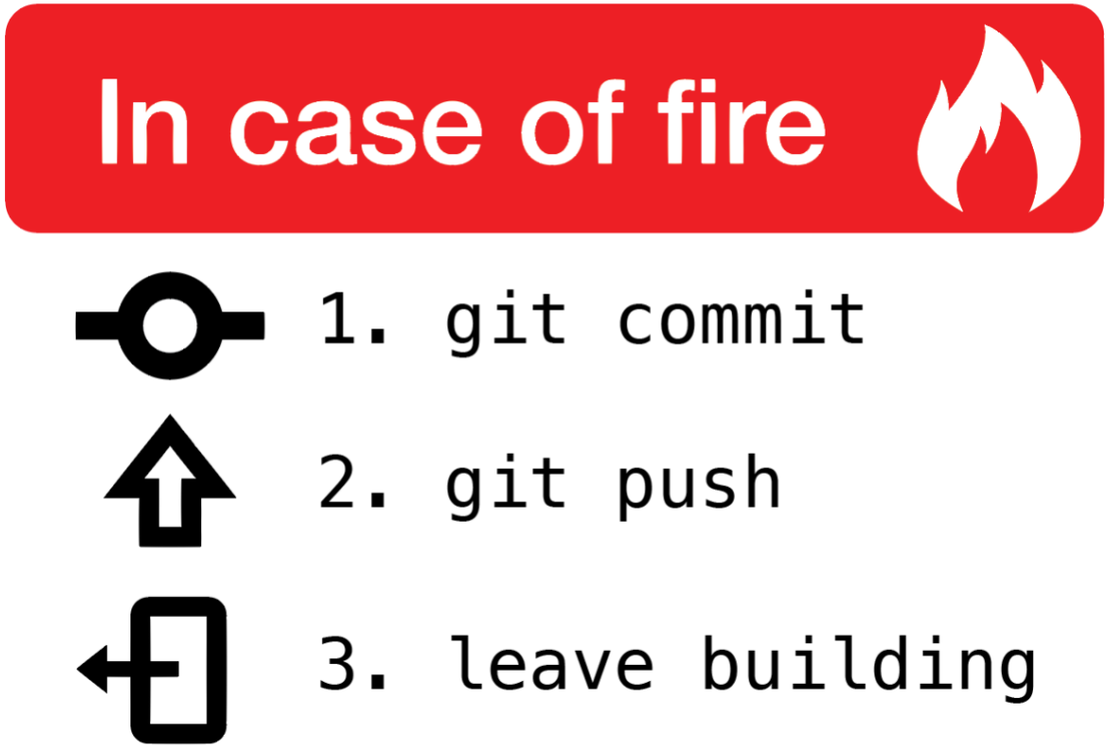
  - GitKraken



ou autres GUI https://git-scm.com/download/gui/

Conclusion


VERSION CONTROL
ALL THE THINGS!
memegenerator.net


In case of fire
1. git commit
2. git push
3. leave building

Ressources [1]

- Starter Pack
  - Le site officiel : https://git-scm.com/
    - Les multiples clients : https://git-scm.com/downloads
    - Les pages de documentation : https://git-scm.com/doc
    - Le livre, en français : https://git-scm.com/book/fr/
  - Juste un petit guide pour bien démarrer avec git. no deep shit ;)

    Un site sur une page résumant le nécessaire pour débuter.

Ressources [2]

- Cours
  - Learn Git in 7 minutes

    Un tutorial pour débutant avec une vidéo et une infographie.
  - CodeSchool / GitTry

    Les tutos officiels pour Git.
  - Learn Git Branching

    Un super site avec rendu visuel pour mieux comprendre le fonctionnement des branches.
- Intro
  - https://github.com/sdfepfl/git-basics
  - https://c4science.ch/source/git-demo/

Ressources [3]

- Divers
  - http://think-like-a-git.net/
  - http://marklodato.github.io/visual-git-guide/index-fr.html
  - https://help.github.com/articles/git-and-github-learning-resources/
  - http://danielkummer.github.io/git-flow-cheatsheet/index.fr_FR.html
  - http://www.askaswiss.com/2016/01/12-useful-advanced-git-commands.html
  - https://www.miximum.fr/blog/enfin-comprendre-git/

Ressources [4]

- Awesome
  - https://github.com/dictcp/awesome-git
  - https://github.com/git-tips/tips

Ressources [5]

- Cheat Sheets
  - https://services.github.com/on-demand/downloads/fr/github-git-cheat-sheet.pdf
  - http://rogerdudler.github.io/git-guide/files/git_cheat_sheet.pdf
  - https://github.com/arslanbilal/git-cheat-sheet
  - https://www.git-tower.com/blog/git-cheat-sheet/
  - https://www.atlassian.com/dms/wac/images/landing/git/atlassian_git_cheatsheet.pdf
  - http://www.cheat-sheets.org/saved-copy/git-cheat-sheet.pdf

Ressources [6]

- Infographie
  - Learn Git in 7 minutes
  - https://github.com/aaasen/github_globe

A propos 💡

Cette présentation a été faite avec le logiciel Marp — Markdown Presentation Writer — dans le cas des dojos de programmation pour les apprentis informaticiens de l'EPFL.

Ces slides sont disponibles sur le dépot gitro — une petite introduction à Git.