

GIT

LUNCH AND LEARN

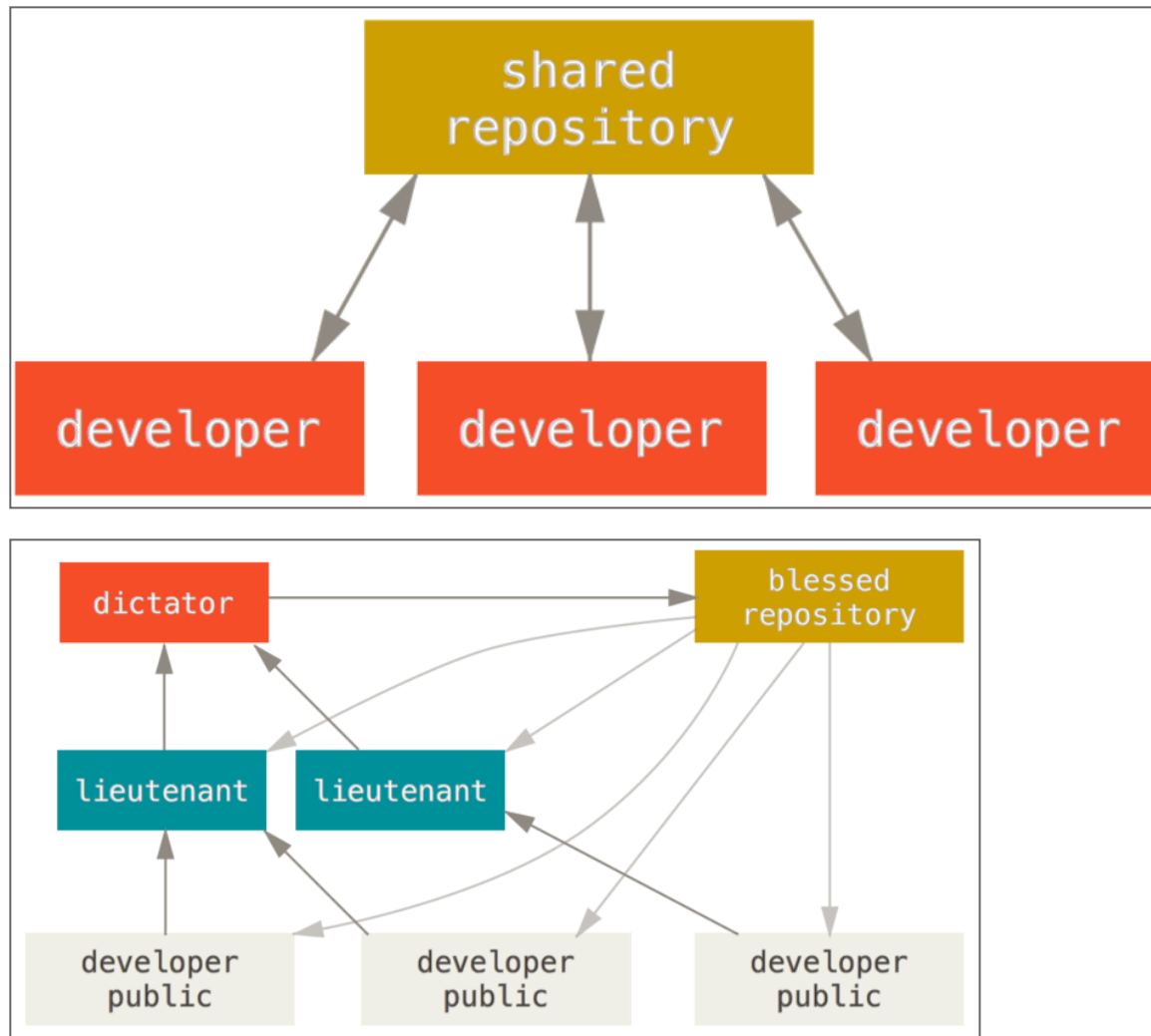


GIT = DISTRIBUTÉ

Chaque utilisateur a sa propre version d'un repository. Un remote fait référence à repository distant.

Les commits sont appliqués localement, push et pull servent à mettre à jour les repositories entre eux.

GIT = DISTRIBUÉ



OUTILS

- Sourcetree (Windows, OSX)
- Intégration dans IDE: Visual Studio, VS Code, JetBrains, XCode
- Shell prompt
 - <https://github.com/magicmonty/bash-git-prompt>
 - fish
 - Git bash (Windows)

SHORTCUTS

```
git config --global alias.co checkout  
git config --global alias.br branch  
git config --global alias.ci commit  
git config --global alias.st status
```

BRANCHES

Un branche est l'équivalent d'un bookmark vers un commit. Créer une branche ne coûte rien. HEAD correspond au dernier commit d'une branche.

```
# Lister les branches
git branch
git branch -a
git branch -av

# Changer de branche
git checkout |branch_name|

# Créer une branche
git checkout -b |branch_name|
```

BRANCHES

```
# Renommer une branche  
git branch -m |new_name|  
  
# Effacer un branche  
git branch -D |branch_name|
```

CHANGEMENTS

```
git add |filename|  
git add -u  
git reset |filename|  
git reset  
git rm |filename|
```

- Tracked: fichier qui fait partie du repository
- Staged: addition prévue lors du prochain commit.

CHANGEMENTS

```
git diff  
git diff --cached  
git diff |branch|  
git diff |commit|  
git difftool
```

CHANGEMENTS

Sélection de hunks.

```
git add -p |filename|
```

Permet de commiter seulement une partie des modifications à un fichier.

CHANGEMENTS

.gitignore -- .git/info/exclude

```
builds/  
*.bak
```

CHANGEMENTS

Sauvegarde temporaire avec git stash

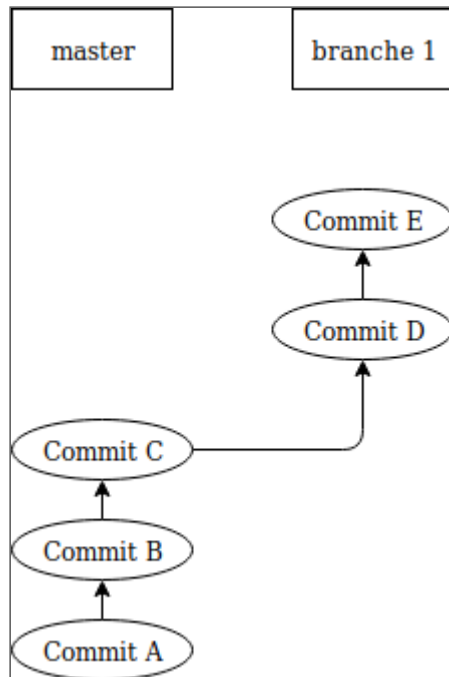
```
git stash  
git stash save |name|  
git stash -u  
git stash list  
git stash apply  
git stash pop
```

CHANGEMENTS

Revenir au départ

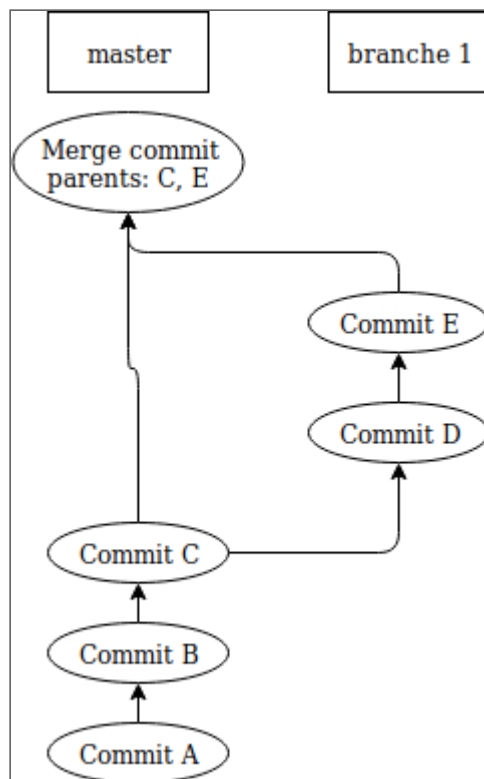
```
git checkout |filename|  
git reset --hard  
git clean -f
```

MERGES



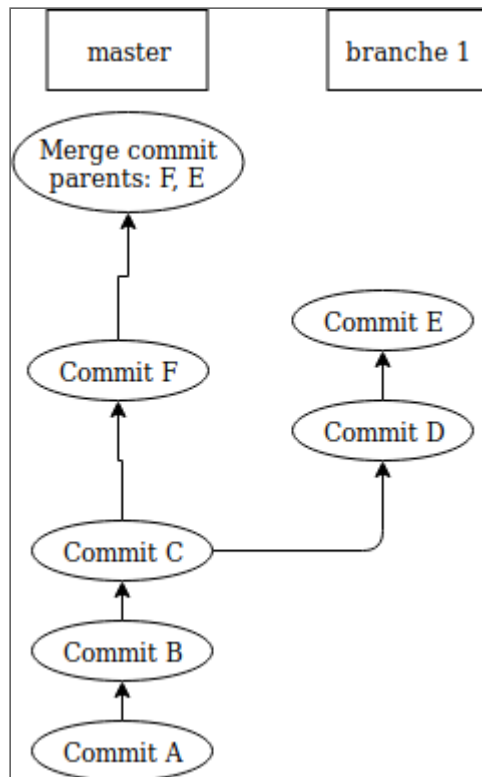
MERGES

Merge



MERGES

Merge



MERGES

Fast-forward merge

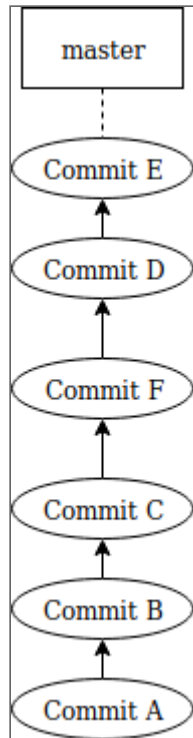
What is a fast-forward merge?

It will just shift the
master HEAD



MERGES

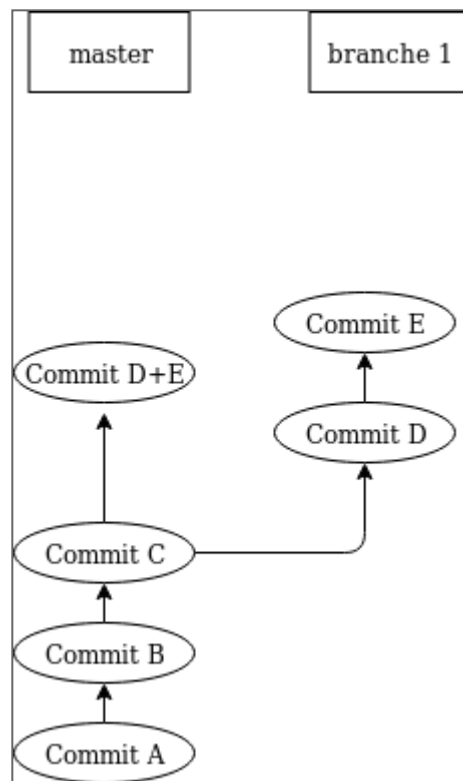
Rebase



MERGES

Squash

Remplace les commits de la branche par un seul nouveau commit.



GESTION DES CONFLITS

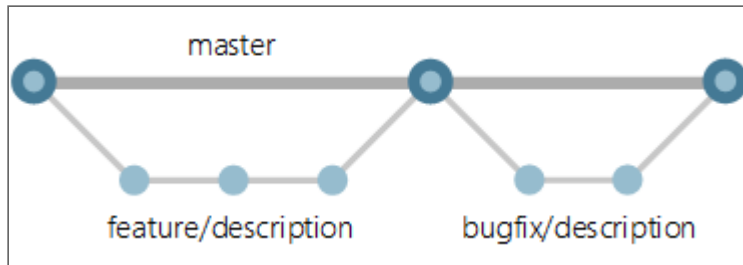
```
GIT MERGETOOL  
GIT MERGE --ABORT
```

WORKFLOW

Toujours travailler sur une branche, ne pas commiter directement sur master.
Github et Bitbucket peuvent être configurés pour l'interdire.

BRANCHES ET PULL REQUESTS

master est la branche commune, les changements sont intégrés via des pull requests.



REBASE D'UNE BRANCHE SUR MASTER

Si une branche de travail est utilisée sur une longue période de temps, rebaser permet de gérer les conflits plus facilement.

```
git fetch  
git rebase master
```

MODIFICATION DE L'HISTORIQUE

Utile sur les branches de travail

```
# Réinitialiser à un commit spécifique  
git reset --hard |commit|  
  
# Réinitialiser une branche locale à partir d'une branche  
# remote  
git reset --hard |origin/branch|
```


MODIFICATION DE L'HISTORIQUE

```
# Modifier le dernier commit  
git commit --amend
```

```
# Rebase interactif  
git rebase -i HEAD~2
```

MODIFICATION DE L'HISTORIQUE

cherry-pick permet d'appliquer un commit spécifique.

```
git cherry-pick |commit|
```

MODIFICATION DE L'HISTORIQUE

Si la branche locale a divergée de la branche remote.

```
git push --force
```

Ne jamais faire sur une branche publique. Utiliser git revert pour annuler un commit.

```
git revert |commit|  
# Revert d'une branche  
git revert -m 1 |commit|
```

MODIFICATION DE L'HISTORIQUE

Appliquer les modifications d'une branche sans garder les commits.

```
git merge --no-commit --squash |branch|  
git reset
```

HISTORIQUE

```
git log  
git log |filename|  
git log --stat  
  
git show |commit|  
  
git blame |filename|
```

HISTORIQUE

```
git grep -n |expr|
```

```
# Recherche dans le contenu sur le git
```

```
# Recherche à partir du dossier courant
```

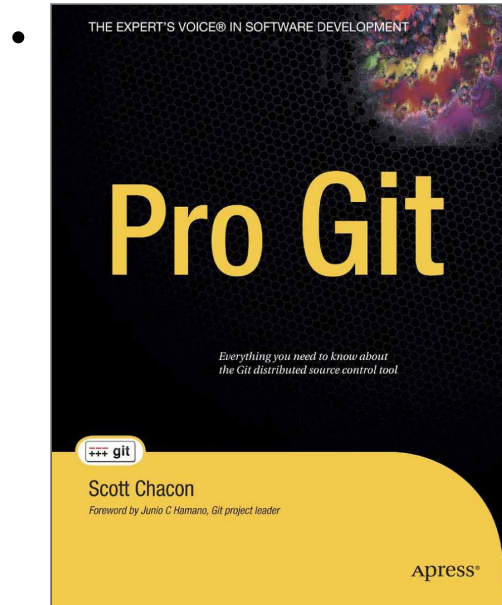
```
# Souvent plus efficace/pertinent que grep directement
```

HISTORIQUE

```
# Checkout d'un commit spécifique  
git checkout |commit|  
git checkout -b |branch| |commit|
```

```
# "Detached HEAD" signifie un checkout qui n'est pas sur  
# une branche.
```

RESSOURCES



- `git help |command|`
- <https://github.com/git-tips>
- <https://www.atlassian.com/git/tutorials/comparing-workflows>
- <https://docs.microsoft.com/en-us/vsts/repos/git/git-branching-guidance>

FAIRE UN COMMIT

```
git clone https://github.com/ekse/rdm
```

```
git status
```

```
git add
```

```
git commit
```

```
git push
```