[Projet Tech']
Gestion de code source avec Git

Fabien Badeig

Institut Henri Fayol

Tue 17 Sept, 2018

Table of contents

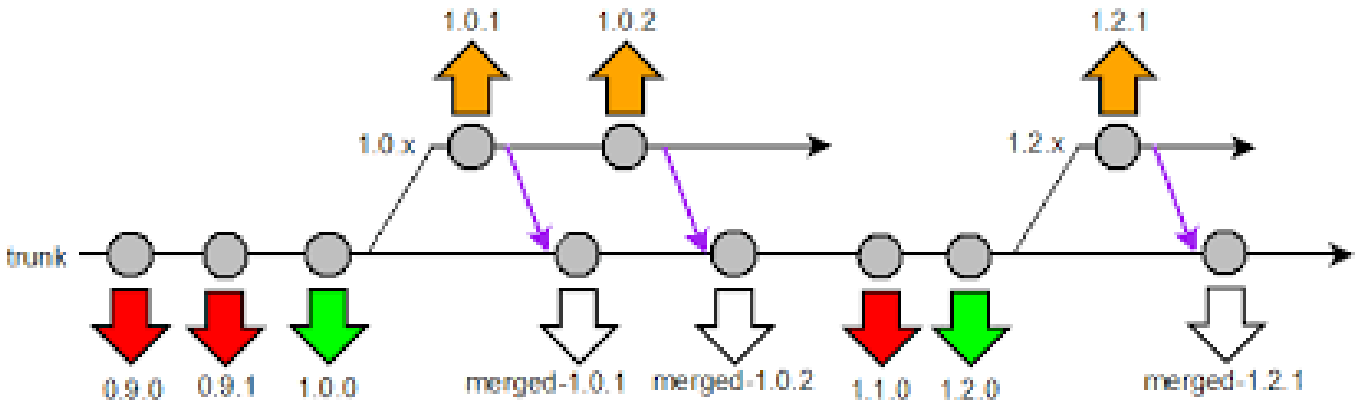
Définition

Git

Git est un gestionnaire de versions distribué. Pour faire simple, c'est un outil permettant de travailler :

- à plusieurs : accès concurrentiel
- dans le temps : historique des modifications, gestion de tags, pas de perte de codes, ...
- à distance : avec des environnements locaux personnalisés et gardant l'historique d'évolution

sur un ensemble de fichiers formant un projet.

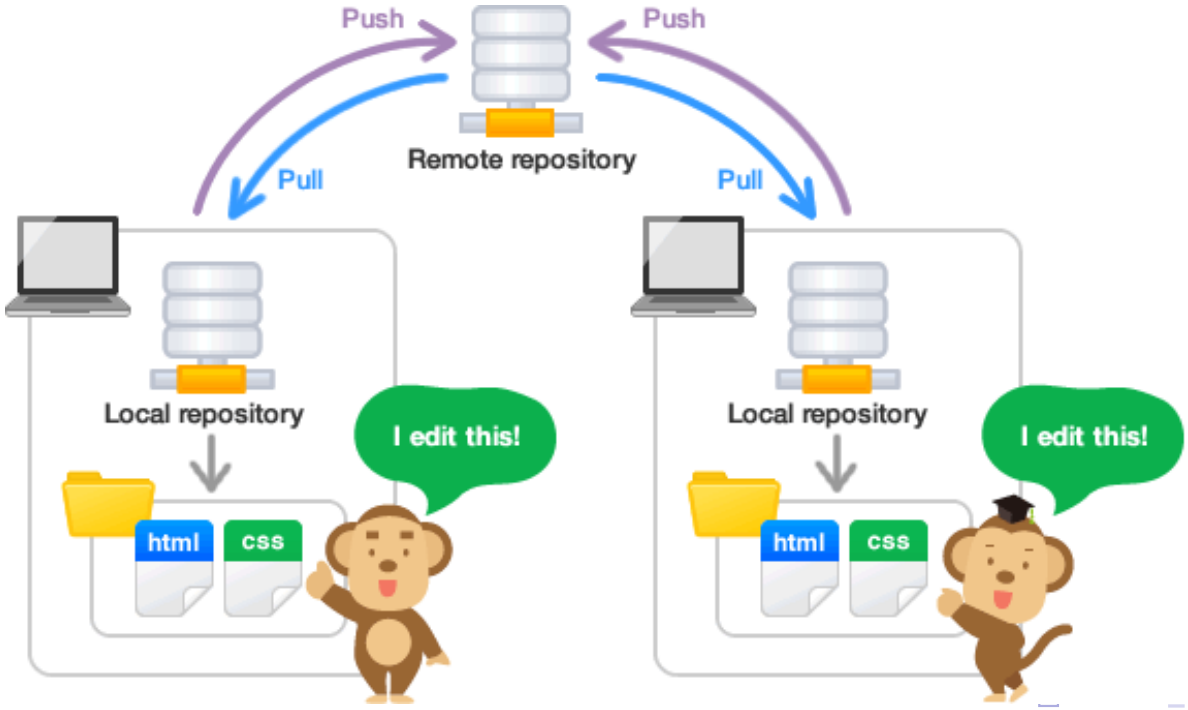


Fonctionnement

Structure

Git est organisé en trois couches

1. dépôt distant (sur un serveur)
2. dépôt local (sur votre machine)
3. dossier de travail (sur votre machine)



Commandes de base

Commandes

- copie d'un dépôt :

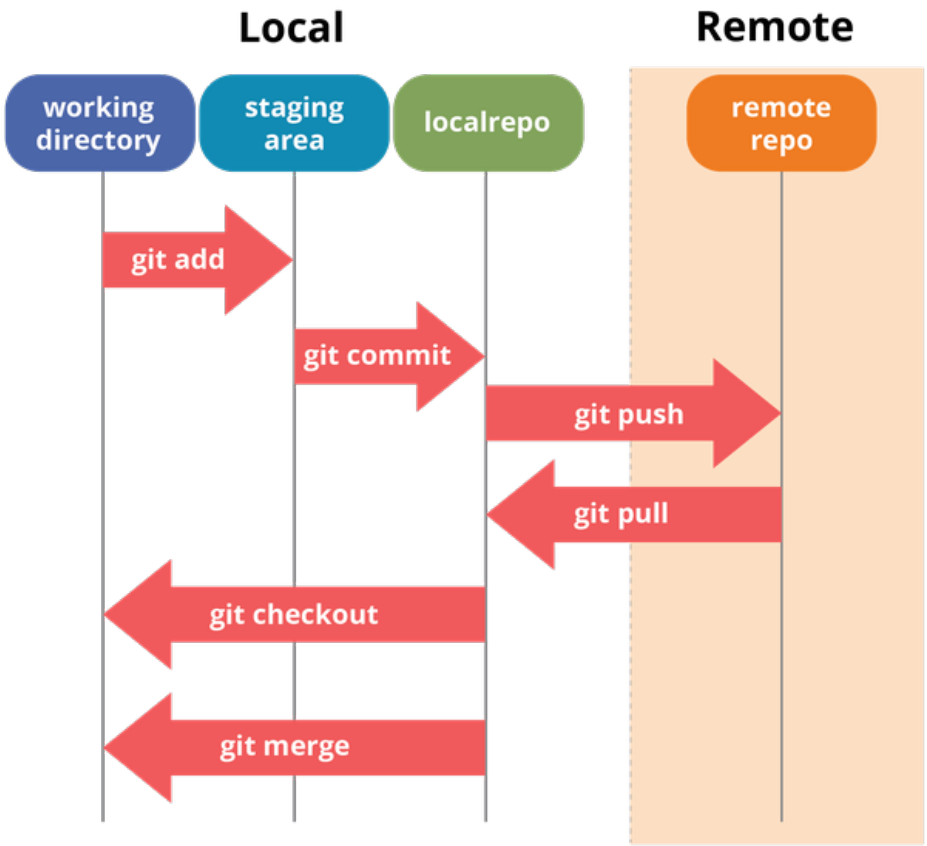
```
>$ git clone <url>
```

- modification d'un fichier

```
>$ git add <files>
>$ git commit <files>
   -m "<description>"
>$ git push <serveur>
            <branche>
```

- mise à jour du dépôt local

```
>$ git pull
```



Projet

Contexte

- Plusieurs développeurs
- Une instance de test
- Une instance de production ("version stable")
- Des mises à jour (correction de bugs)
- Des nouvelles fonctionnalités

Règles

1. **UN SEUL** mainteneur qui gère les releases
2. **NE JAMAIS** commiter sur la branche master
3. **NE JAMAIS** faire de rebase de master sur une autre branche
4. Respecter le workflow de développement

Gestion

Workflow : correction de bug/ nouvelle fonctionnalité

- Créer une branche depuis master
- Commiter ses ajouts sur sa branche
- Vérifier que son code fonctionne toujours avec le code de master avec un rebase depuis sa branche sur master

```
>$ (master) git checkout -b featureA
>$ (featureA) git commit -a -m "featureA part 1"
>$ (featureA) git rebase master
```

Travail du mainteneur

```
$ (master) git merge --no-ff featureA
$ (master) git branch -d featureA
$ (master) git push origin :featureA
$ (master) git tag 1.0
$ (master) git checkout -b stable1.0
$ (stable1.0) git push origin stable1.0
```

Commandes utiles

Commandes

- Etat du dépôt :

```
>$ git status
```

- Comparaison du contenu d'un fichier sur la copie de travail avec sa version sur le dépôt :

```
>$ git diff <file>
```

- historique des logs :

```
>$ git log
```