Java Avancé

Cours 1 : Outillage

Arsène Lapostolet

19 Janvier 2024

Les outils du module

- Plateforme de développement : Machine Virtuelle Java (JVM)
- Langage de programmation : Java 21
- Environment de développement intégré : IntelliJ IDEA Community
- Système de Gestion de Version : Git et Github
- Système de build : Gradle

Git

Qu'est ce que Git ?

Système de Gestion de Version :

- Versionner le code : suivre précisement les changements
- Naviguer entre les versions
- Ne pas perdre de changements
- Revenir en arrière
- Coopération à plusieurs sur le même code

Concepts importants

- **Commit :** version du code sauvegardée à un instant T.
- **Branche :** historiques de commits qui évoluent en parallèle

Créer un dépôt local :

```bash
mkdir dossierPourMonDepot
cd dossierPourMonDepot
git init
```

Base d'un workflow Git

- Créer un commits
  1. `git add .` : ajouter tous les changements au suivi git
  2. `git commit -m "message de commit"` : crée un commit avec les changements trackés
- Créer une branche :
  1. `git branche nomDeLaBranche` : créer la branche
  2. `git checkout nomDeLaBranche` : se positionner sur la branche

Fusion

Pour appliquer les changement présents sur une branche à une autre :

```Bash
git checkout brancheCible
git merge brancheAFusionner
```

On appliquer les changement des la branche `brancheAFusionner` sur la branche `brancheCible`.

Attention aux conflits !

Dépôt distant

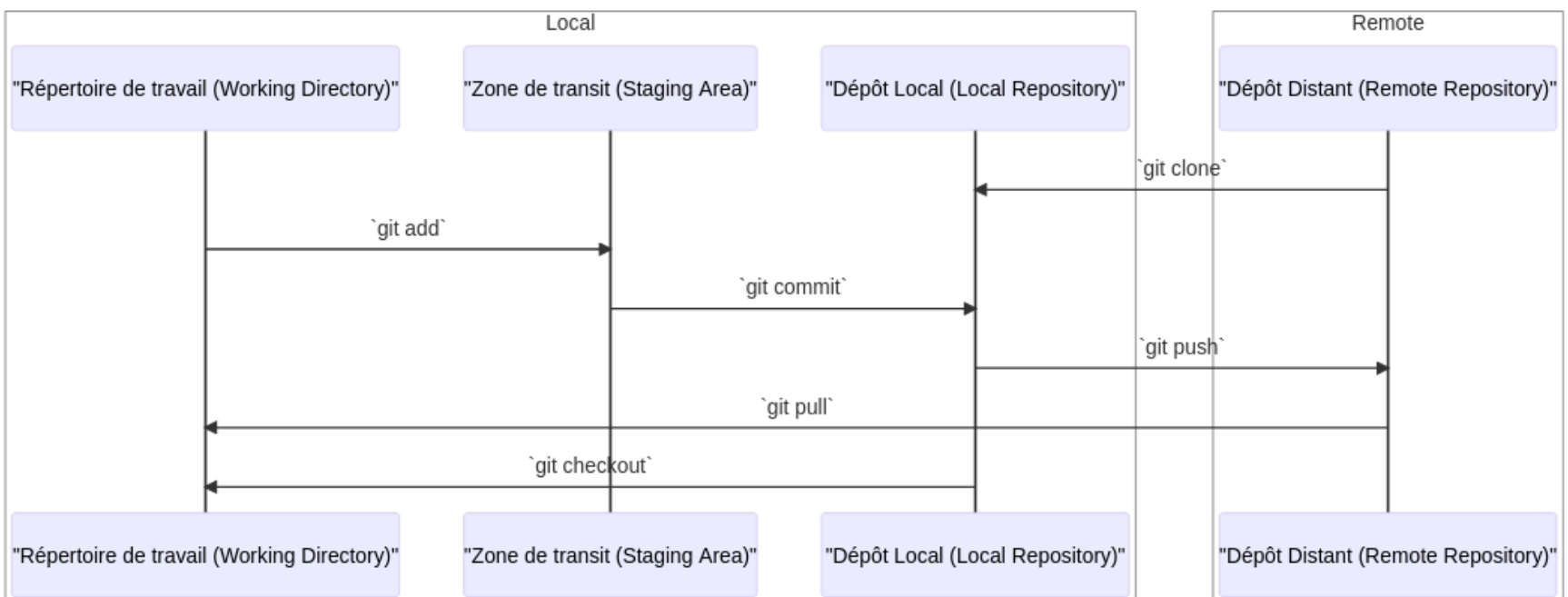
```Bash
git remote add nomLocalDuDepotDistant urlDuDepotDistant
```

1. Sauvegarder son travail en lieu sur
2. Coopérer avec d'autres personnes

Intéraction :

- `git push nomDelaBranche` : pousser une branche locale vers le dépôt
- `git pull nomDelaBranche` : récupérer une branche distante dans son dépôt local

Récap des espaces



Récap du workflow

1. Se positionner sur la branche principale : `git checkout master`
2. Mettre à jour la branche principale : `git pull origin master`
3. Créer une nouvelle branche à partir de la branche principale : `git branch maFeature`
4. Se positionner sur la nouvelle branche : `git checkout maFeature`
5. Faire des changements dans le code

6. Ajouter les changements à git : `git add .`
7. Créer un nouveau commit avec les changements : `git commit -m "message de commit"`
8. Pousser les changements : `git push origin maFeature`
9. Retourner sur master : `git checkout master`
10. Fusionner la branche de feature : `git merge maFeature`

Gradle

Qu'est ce que Gradle ?

- Outil de build
  - Structuer le projet
  - Décrire comment le compiler, packager, executer, tester