

Atelier 2 - Création et configuration de jobs Jenkins



Objectif

Cet atelier a pour objectif de démontrer le fonctionnement du serveur Jenkins en créant des projets (jobs) Jenkins. Nous allons donc créer et configurer des projets Maven basé sur un dépôt Git.

Introduction

Avant de commencer la création de jobs sur Jenkins, il est essentiel d'avoir le serveur **Jenkins** installé et configuré, un **projet** prêt à être compilé, testé ou déployé, et un **dépôt** Git pour stocker les codes. Nous allons détailler les étapes de création et de configuration de projets de type **freestyle** et **pipeline**.

Un job est l'équivalent d'un projet. Il va donc contenir un lien vers le code source (le dépôt où seront stockés les fichiers sources), des tâches de build et des actions post-build, que nous devons les paramétrer pour construire le projet (build).

I. Étapes pour créer un Job Freestyle sur Jenkins

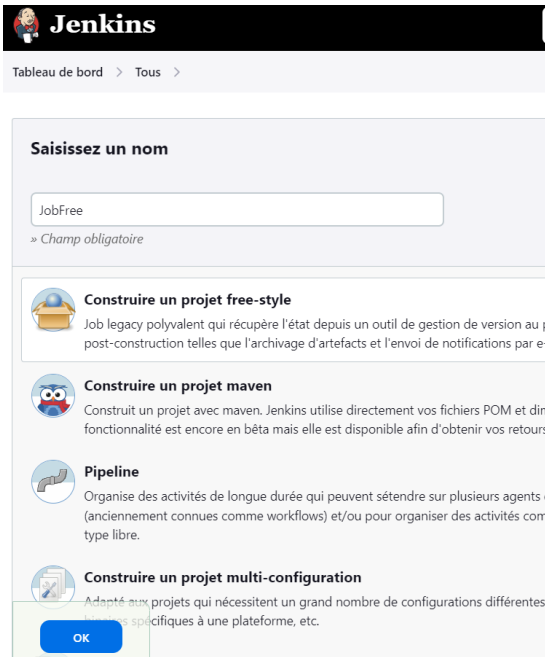
La création d'un Job Freestyle sur Jenkins implique les étapes suivantes :

1. Création d'un job freestyle

- Connectez-vous à l'interface web de Jenkins et cliquez sur le bouton « Nouveau Item ».

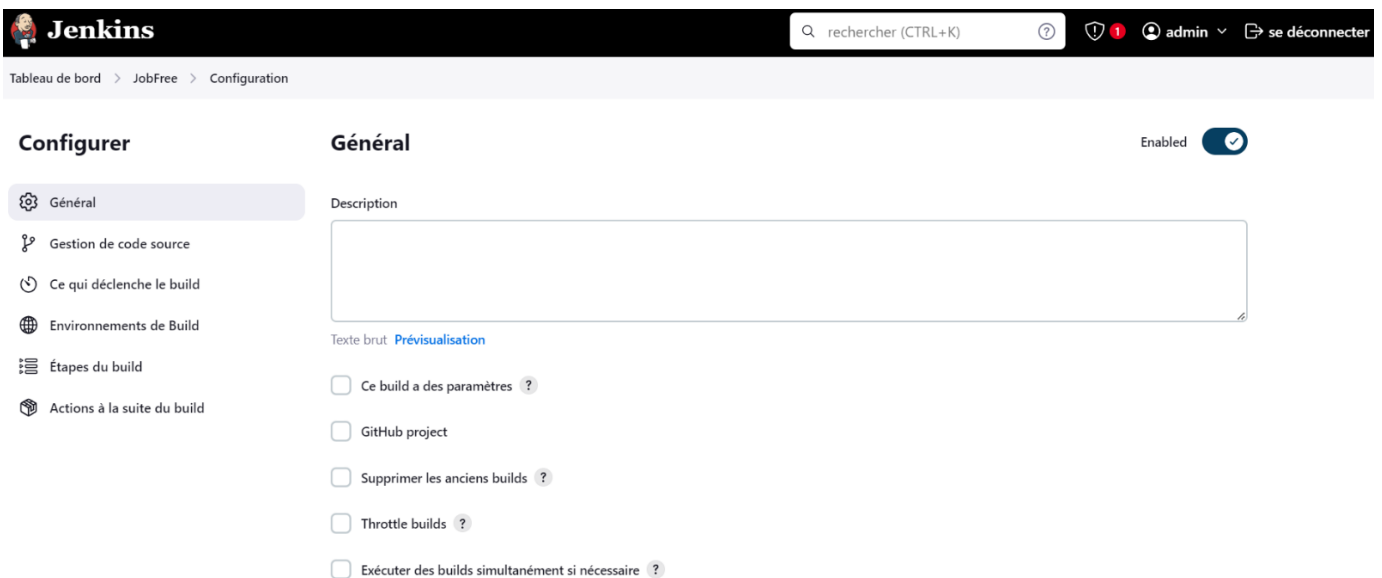


- Donnez un nom à votre Job et choisissez le type « projet free-style ». Cliquez sur « OK » pour accéder à la page de configuration du Job.



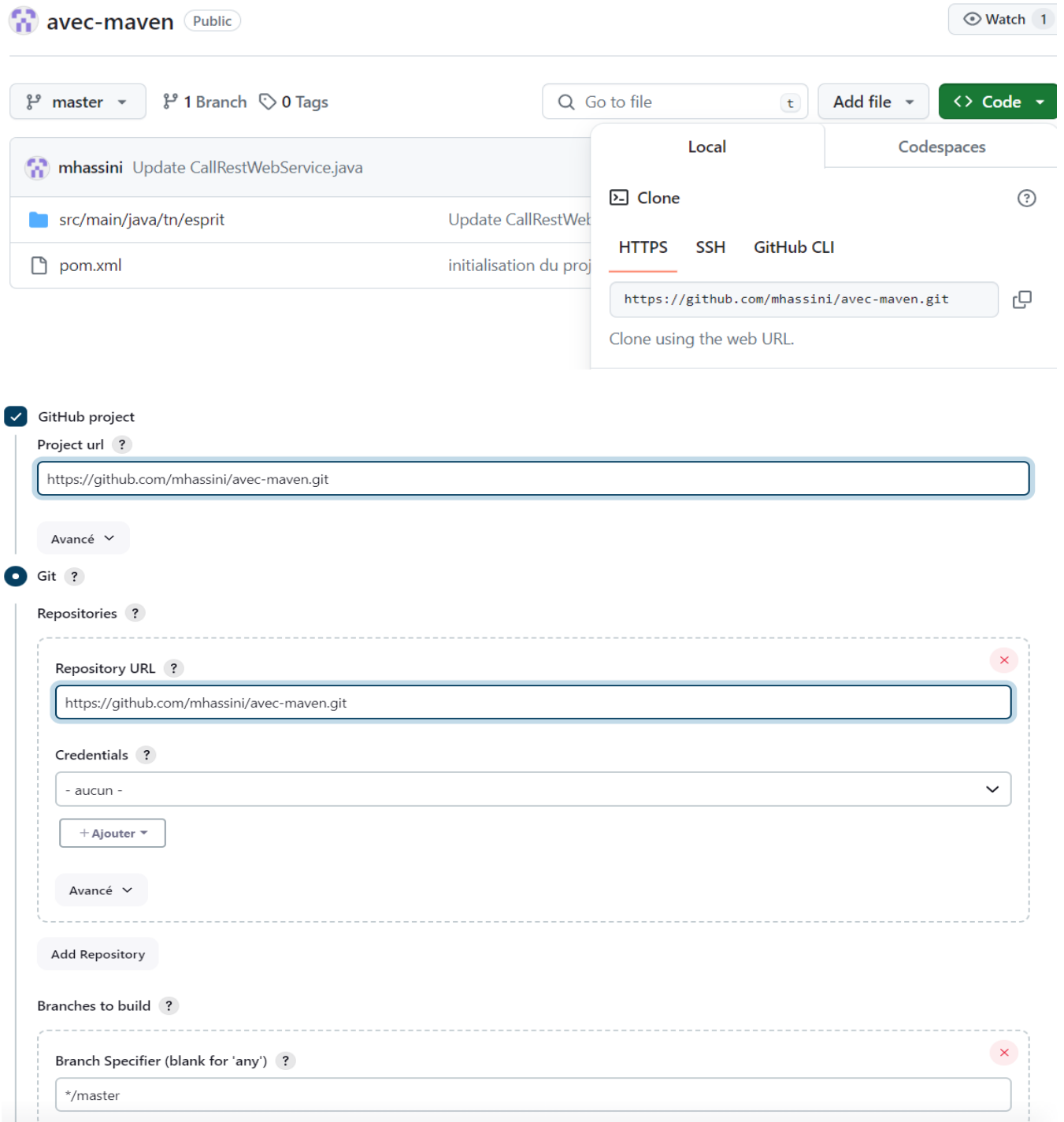
2. Configuration d'un job freestyle

- Après avoir créé le job, vous devez ensuite le configurer pour compiler le projet. La première section de la page de configuration contient les détails généraux du projet.



- Pour obtenir du code à partir de Git, vous avez plusieurs options :
 - utilisez des dépôts Git publics (comme dans notre cas)
 - configurez des clés SSH

- saisissez des noms d'utilisateur et des mots de passe dans la section "Identifiants" afin que Jenkins puisse récupérer le code Git.
- Copiez l'URL de votre projet Git. Voici un exemple que vous pouvez utiliser à des fins de test : <https://github.com/mhassini/avec-maven.git>



avec-maven Public Watch 1

master 1 Branch 0 Tags Add file Code

mhassini Update CallRestWebService.java

src/main/java/tn/esprit Update CallRestWeb

pom.xml initialisation du proj

Clone ?

HTTPS SSH GitHub CLI

<https://github.com/mhassini/avec-maven.git> Copy

Clone using the web URL.

☒ **GitHub project**

Project url ?

<https://github.com/mhassini/avec-maven.git>

Avancé ▼

☒ **Git** ?

Repositories ?

Repository URL ? ✕

<https://github.com/mhassini/avec-maven.git>

Credentials ?

- aucun - ▼

+ Ajouter ▼

Avancé ▼

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ? ✕

*/master

- Sélectionnez l'action pour déclencher périodiquement le flux d'intégration continue. Cela peut être un processus qui s'exécute régulièrement, par exemple chaque minute, ou qui vérifie si une nouvelle version a été poussée sur Git, ...

Ce qui déclenche le build

- ☐ Déclencher les builds à distance (Par exemple, à partir de scripts) ?
- ☐ Construire après le build sur d'autres projets ?
- ☒ Construire périodiquement ?

Planning ?

H/5 * * * *

Aurait été lancé à Saturday, June 29, 2024 at 11:08:30 PM Coordinated Universal Time; prochaine exécution à Saturday, June 29, 2024 at 11:13:30 PM Coordinated Universal Time.

- ☐ GitHub hook trigger for GITScm polling ?
- ☐ Scrutation de l'outil de gestion de version ?

- Dans la section « Etapes du build », vous pouvez ajouter des étapes de construction selon vos besoins, comme l'exécution de scripts de compilation, de tests, ou de déploiement. Vous pouvez utiliser des outils comme Java, Docker ou Git pour configurer vos étapes.

Étapes du build

Ajouter une étape au build ^

Filter

Appeler Ant

Exécuter un script shell

Exécuter une ligne de commande batch Windows

Invoke Gradle script

Invoquer les cibles Maven de haut niveau

Run with timeout

Set build status to "pending" on GitHub commit

Exemple 1 : Écrivez un message en incluant la date système, enregistrez-le, puis attendez une minute (le job se lancera automatiquement toutes les minutes).

Étapes du build

Exécuter un script shell ?

Commande

Voir [la liste des variables d'environnement disponibles](#)

```
echo "c'est mon premier job freestyle Jenkins!"
```

Avancé ▾

Ajouter une étape au build ▾

Exemple 2 : Vérifiez l'installation de maven.

☰ Exécuter un script shell ?

Commande

Voir [la liste des variables d'environnement disponibles](#)

```
mvn -version
```

Vous pouvez tester ces deux exemples et visualiser le résultat.

3. Exécution et visualisation du résultat

- Une fois la configuration du job est terminée, cliquez sur **Sauvegarder**. Vous pouvez lancer le build et consulter les résultats de son exécution.


 **Jenkins**

Tableau de bord > JobFree >

📄 État

</> Modifications

📁 Répertoire de travail

▶ Lancer un build

⚙️ Configurer

🗑️ Supprimer Projet

🔄 GitHub

✎ Renommer

✅ JobFree

Liens permanents

- [Dernier build \(#1\), il y a 31 s](#)
- [Dernier build stable \(#1\), il y a 31 s](#)
- [Dernier build avec succès \(#1\), il y a 31 s](#)
- [Dernier build complété \(#1\), il y a 31 s](#)

☀ Historique des builds tendance ▼

🔍 Filter... /

✅ #1

23 juin 2024 23:14

- L'icône verte à gauche de la date du build indique que tout s'est bien passé. Vous pouvez consulter le tableau de bord pour plus de détails.

Tableau de bord >

+ Nouveau Item ✎ Ajouter une description

Historique des constructions Tous +

Administrer Jenkins

Mes vues

File d'attente des constructions ▼

File d'attente des constructions vide

S	M	Nom du projet ↓	Dernier succès	Dernier échec	Dernière durée
✓	☀	JobFree	1 mn 57 s #1	s. o.	4.6 s

Icône: S M L ...

- En cas d'échec de build, vous pouvez consulter la sortie de la console pour déterminer les problèmes lors de la compilation et de les corriger.

État ✖ Sortie de la console

</> Modifications

Sortie de la console

Voir en texte brut

Informations de la construction

Supprimer le build "#26"

Timings

← Build précédent

```

Started by user admin
Running as SYSTEM
Building on the built-in node in workspace /var/lib/jenkins/workspace/JobFree
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/JobFree/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/mhassini/avec-mmaven.git # timeout=10
Fetching upstream changes from https://github.com/mhassini/avec-mmaven.git
> git --version # timeout=10
> git --version # 'git version 2.34.1'
> git fetch --tags --force --progress -- https://github.com/mhassini/avec-mmaven.git +refs/heads/*:refs/remotes/origin/* # timeout=10
ERROR: Error fetching remote repo 'origin'
hudson.plugins.git.GitException: Failed to fetch from https://github.com/mhassini/avec-mmaven.git
    at hudson.plugins.git.GitSCM.fetchFrom(GitSCM.java:999)
  
```

II. Étapes pour créer un Job Pipeline sur Jenkins

La création d'un Job Pipeline sur Jenkins implique les étapes suivantes :

1) Création d'un job pipeline sur Jenkins

- Pour commencer, vous devez créer un "pipeline" en tant que type de projet.


Jenkins rechercher (CTRL+K) ? 1

Tableau de bord > Tous >


Saisissez un nom

JobPipeline


» Champ obligatoire

 **Construire un projet free-style**


Job legacy polyvalent qui récupère l'état depuis un outil de gestion de version au plus, exécute les étapes de build en série, suivi d'étapes post-construction telles que l'archivage d'artefacts et l'envoi de notifications par e-mail.

 **Construire un projet maven**

Construit un projet avec maven. Jenkins utilise directement vos fichiers POM et diminue radicalement l'effort de configuration. Cette fonctionnalité est encore en bêta mais elle est disponible afin d'obtenir vos retours.

 **Pipeline**

Organise des activités de longue durée qui peuvent s'étendre sur plusieurs agents de construction. Adapté pour la création des pipelines (anciennement connues comme workflows) et/ou pour organiser des activités complexes qui ne s'adaptent pas facilement à des tâches de type libre.

 **Construire un projet multi-configuration**

Adapté aux projets qui nécessitent un grand nombre de configurations différentes, comme des environnements de test multiples, des tests spécifiques à une plateforme, etc.

OK

- Ensuite, vous pouvez définir le script du pipeline pour construire le projet. Voici trois exemples, vous pouvez les tester séparément en créant trois jobs différents ou dans le même job (une phase (stage) pour chacun).

Exemple 1 : Affichez un message "Hello world !" avec Groovy.

Definition

Pipeline script

Script ?

```

1 pipeline {
2   agent any
3
4   stages {
5     stage('Hello') {
6       steps {
7         echo 'Hello World'
8       }
9     }
10  }
11 }
12

```

Use Groovy Sandbox ?

Pipeline Syntax

Sauvegarder Appliquer

Exemple 2 : Récupérerez le code à partir de Git (ajoutez credentialsId si le repository est privé).

Pipeline

Definition

Pipeline script

Script ?

```

1 pipeline {
2   agent any
3   tools {
4     maven 'M2_HOME'
5   }
6   stages {
7     stage('GIT') {
8       steps {
9         git branch: 'master',
10        url: 'https://github.com/hwafa/atelier-jenkins.git',
11        credentialsId: 'jenkins-example-github-pat'
12      }
13    }
14  }
15 }
16 }
17

```

try sample Pipeline...

Exemple 3 : Exécutez une commande Maven.

Pipeline

Definition

Pipeline script

Script ?

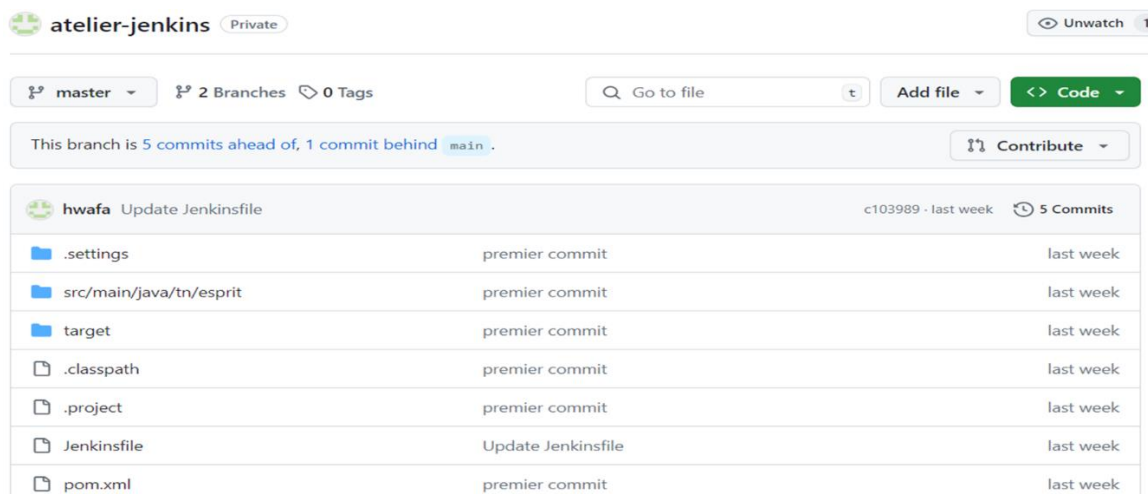
```

1 pipeline {
2   agent any
3   tools {
4     maven 'M2_HOME'
5   }
6   stages {
7     stage('MAVEN') {
8       steps {
9         sh "mvn -version"
10      }
11    }
12  }
13 }
14 }
15
```

try sample Pipeline...

2) Création d'un job pipeline à partir de Jenkinsfile

- Pour commencer, vous devez créer un fichier " Jenkinsfile " et insérer le script pipeline que vous voulez exécuter dedans, ensuite sauvegarder le fichier dans le référentiel de code (Git).
- Pour configurer le job pipeline à partir de Jenkinsfile, il suffit de pointer jenkins vers ce fichier.



The screenshot shows the Jenkins job configuration page for 'atelier-jenkins' (Private). The job is currently on the 'master' branch, which is 5 commits ahead of and 1 commit behind the 'main' branch. The job has 2 branches and 0 tags. The 'Add file' button is visible, along with a 'Code' button. Below the branch information, a table lists the files in the repository:

File	Commit	Time
.settings	premier commit	last week
src/main/java/tn/esprit	premier commit	last week
target	premier commit	last week
.classpath	premier commit	last week
.project	premier commit	last week
Jenkinsfile	Update Jenkinsfile	last week
pom.xml	premier commit	last week

Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

Repository URL ?

https://github.com/hwafa/atelier-jenkins.git

Credentials ?

jenkins-example-github-pat

+ Ajouter

Avancé

Branches to build ?

Branch Specifier (blank for 'any') ?

*/master

Add Branch

Navigateur de la base de code ?

(Auto)

Additional Behaviours

Ajouter

Script Path ?

Jenkinsfile

3) Configuration d'un job pipeline sur jenkins

- Les Jobs Jenkins peuvent être lancés de différentes manières :
 - a. Manuellement
 - b. Selon un crône répétitif (builds réguliers)
 - c. Déclencher des builds à distance
 - d. Utilisation des WebHooks
- Déclenchez un build à **distance** en accédant à une URL spécifique via votre navigateur.

Build Triggers

- ☐ Construire après le build sur d'autres projets ?
- ☐ Construire périodiquement ?
- ☐ Construire dès qu'une dépendance SNAPSHOT est modifiée ?
- ☐ GitHub hook trigger for GITScm polling ?
- ☐ Scrutation de l'outil de gestion de version ?
- ☐ Période d'attente ?
- ☒ Déclencher les builds à distance (Par exemple, à partir de scripts) ?

Jeton d'authentification

Utilisez l'URL qui suit pour lancer un build à distance : `JENKINS_URL/job/JobPipeline/build?token=TOKEN_NAME` ou `/buildWithParameters?token=TOKEN_NAME`
 Optionally append `&cause=Cause+Text` to provide text that will be included in the recorded build cause.

Sur le navigateur, tapez l'URL suivante :

`192.167.33.10:8080/job/JobPipeline/build?token=pipeline-jenkinsfile-token`

Le job se déclenche à distance en tapant cette URL, voici le résultat sur Jenkins :

S	M	Nom du projet ↓	Dernier succès	Dernier échec	Dernière durée	
✓	☀	JobFree	17 mn #5	20 mn #4	0.82 s	▶
✓	☀	JobPipeline	10 s #1	s. o.	5.5 s	▶

Vous pouvez lancer les jobs pipeline créés ci-dessus en appliquant cette manière d'automatisation. Essayez de pratiquer une autre méthode « déclencheurs de build » pour vos jobs freestyle.

Conclusion

Pour cet atelier, nous avons mis un projet sous Jenkins c'est-à-dire nous avons créé un job et nous avons effectué un premier build. La phase de configuration est primordiale dans la construction de projets Jenkins. Pour configurer le build, on pointe le serveur jenkins vers le dépôt git pour récupérer le code source, on crée des tâches (des actions) à effectuer lors du build et on configure même des actions à effectuer après le build.

Si le build réussit, toutes les tâches ont été acceptées, sinon cela signifie qu'une tâche a échoué et on peut voir d'où vient cette erreur facilement en consultant le journal d'exécution.