

## Lab 1 – Introduction à JENKINS

Jenkins est un outil d'automatisation essentiel pour configurer l'intégration continue. C'est l'intégrateur qui vous aide à créer votre flux de travail de développement, de test et de déploiement et à créer des pipelines de tâches. Il ajoute également de la visibilité à toutes les parties prenantes, y compris les équipes de développement, d'assurance qualité et d'exploitation impliquées dans la création, les tests et le déploiement du produit.

Démarche officielle : <https://www.jenkins.io/doc/book/installing/linux/#red-hat-centos>

### Installation RPM

```
$ sudo -i
# wget -O /etc/yum.repos.d/jenkins.repo
https://pkg.jenkins.io/redhat-stable/jenkins.repo
# rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key
# yum upgrade
# yum install java-11-openjdk*
# cat > /etc/profile.d/java.sh <<'EOF'
export JAVA_HOME=$(dirname $(dirname $(readlink $(readlink $(which java))))
export PATH=$PATH:$JAVA_HOME/bin
EOF
# source /etc/profile.d/java.sh
# yum install jenkins
# systemctl enable --now jenkins
# hostnamectl set-hostname jenkins-server.lab.local
# IP=$(hostname -I | awk '{print $1}') eval 'echo "$IP
jenkins-server jenkins-server.lab.local" >> /etc/hosts'
```

### Installation de l'environnement avec Docker (Optionnel)

C'est la méthode la plus simple pour configurer Jenkins et c'est une option recommandée.

#### Installation de Docker Engine

<https://docs.docker.com/engine/install/centos/>

Pour valider l'environnement docker, exécutez.

```
[root@jenkins-server ~]# docker ps
```

Après avoir installé docker, récupérez l'image de docker Jenkins à partir de docker hub.

Il n'y a qu'une seule image docker officielle de Jenkins disponible, qui a été déconseillée au profit de jenkins/jenkins:its.

C'est la manière la plus simple d'installer Jenkins et nécessite un minimum d'efforts.

```
# docker pull jenkins/jenkins:lts
# docker run -idt --name jenkins -v
jenkins_home:/var/jenkins_home -v
/var/run/docker.sock:/var/run/docker.sock -p 8080:8080 -p
50000:50000 jenkins:lts
```

La documentation complète de l'image Jenkins Docker est disponible sur Git Hub . Ici, nous n'utilisons que quelques paramètres nécessaires.

**-p 8080:8080** -> Mappe le port de l'interface utilisateur Web Jenkins du conteneur Docker vers l'hôte Docker.

**-p 50000:50000** -> Mappe le port Build Slaves du conteneur Docker à l'hôte Docker.

**--name = jenkins** -> Nom du conteneur du docker Jenkins

**-v jenkins\_home:/var/jenkins\_home** -> Bind mount pour stocker les données et les configurations du conteneur Jenkins.

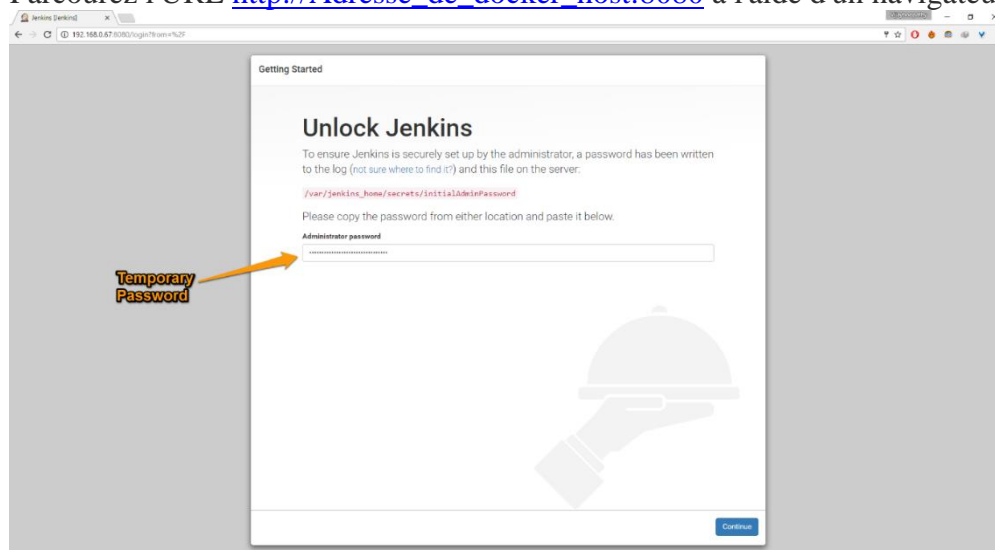
**jenkins/jenkins:lts** -> l'image utilisée pour créer le conteneur du docker Jenkins

Après cela, pour démarrer/arrêter jenkins avec docker, utilisez les commandes suivantes,

```
# docker start jenkins
# docker stop jenkins
```

## Accès à l'interface utilisateur Web de Jenkins:

Parcourez l'URL [http://Adresse\\_de\\_docker\\_host:8080](http://Adresse_de_docker_host:8080) à l'aide d'un navigateur.



- Puisque, nous accédons à l'interface Web de Jenkins pour la première fois, nous avons besoin du mot de passe administrateur pour nous y connecter.

Le chemin d'accès au fichier de mot de passe a été fourni par l'interface Web Jenkins.

```
# cat /var/lib/jenkins/secrets/initialAdminPassword
```

## Pour Docker

Nous pouvons utiliser la commande « Docker cp » pour copier ce fichier sur la machine hôte Docker.

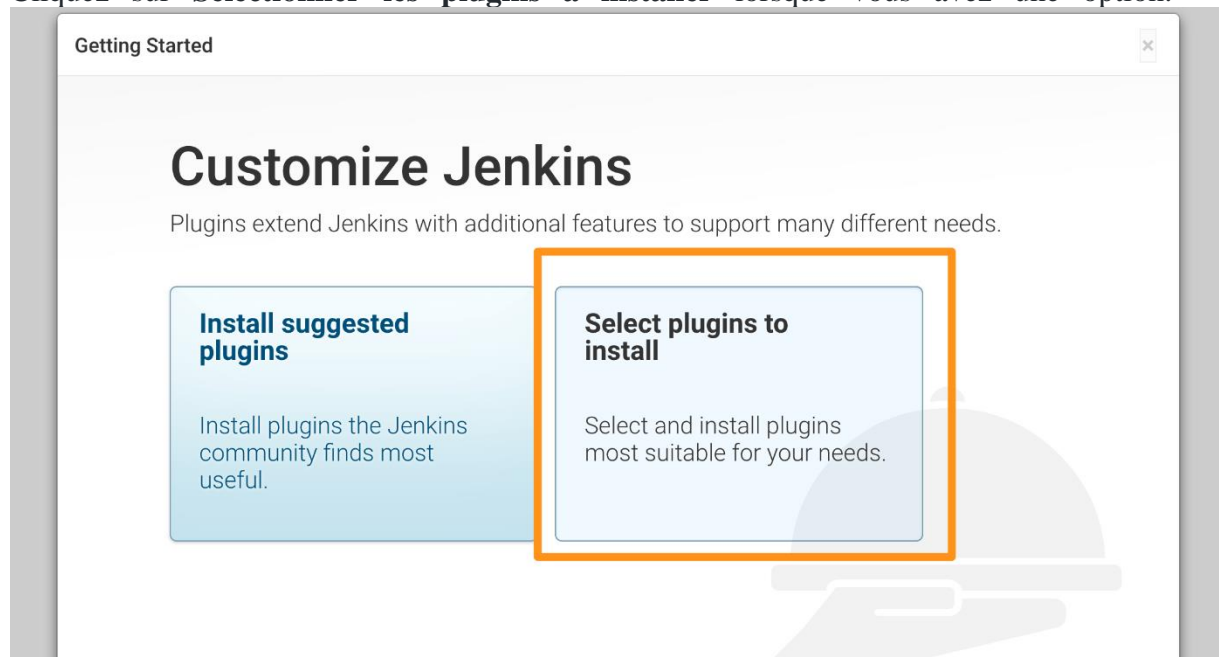
```
# docker cp
jenkins:/var/jenkins_home/secrets/initialAdminPassword /tmp
```

- Affichez le contenu de ce fichier pour obtenir le mot de passe administrateur.

```
# cat /tmp/initialAdminPassword
8a349405002444c89172bf1d519fddc5
```

Connectez-vous à l'interface Web de Jenkins en utilisant ce mot de passe.

- Cliquez sur **Sélectionner les plugins à installer** lorsque vous avez une option.



Cela vous permettra de choisir les plugins à installer sur la page suivante. Sur la page de sélection,

- Cliquez sur **Aucun** pour désélectionner tous les plugins

## Getting Started

Organization and Administration

Build Features

Build Tools

Build Analysis and Reporting

Pipelines and Continuous Delivery

Source Code Management

Distributed Builds

User Management and Security

Notifications and Publishing

All **None** Success

Note that the full list of plugins is not shown here. After initial setup is complete. [See the Wiki for more info](#)

### Organization and Administration

- ☐ **Dashboard View** [↗](#)  
Jenkins view that shows various cuts of build info
- ☐ **Folders Plugin** [↗](#)  
This plugin allows users to create "folders" to organize organization type etc). Folders are nestable and y
- ☐ **OWASP Markup Formatter Plugin** [↗](#)  
Uses policy definitions to allow limited HTML mar

### Build Features (0/10)

- ☐ **build-name-setter** [↗](#)
- ☐ **build timeout plugin** [↗](#)  
Aborts a build if it's taking too long
- ☐ **Config File Provider Plugin** [↗](#)  
Ability to provide configuration files (e.g. settings.

Créer un utilisateur administrateur

## Getting Started

### Create First Admin User

Username:

Password:

Confirm password:

Full name:

E-mail address:

Getting Started

# Instance Configuration

Jenkins URL:

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the BUILD\_URL environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

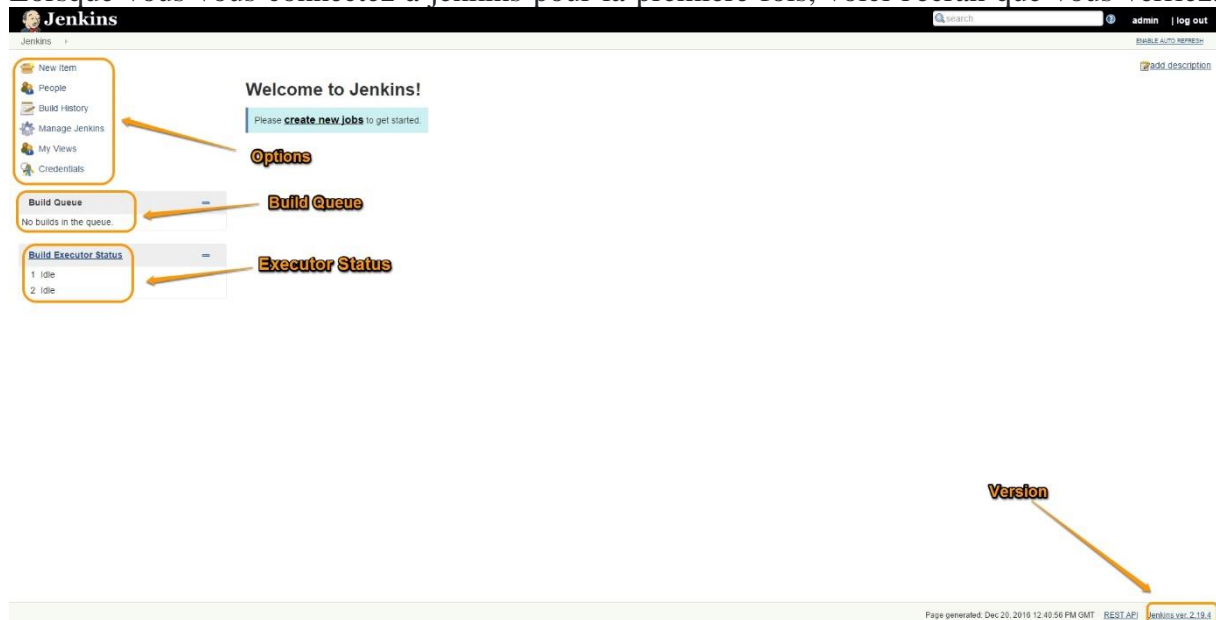
Jenkins 2.426.1 Not now Save and Finish

Maintenant, nous avons installé Jenkins avec succès et nous pouvons procéder aux configurations

## Configurations Jenkins

### Se familiariser avec la console Jenkins

Lorsque vous vous connectez à jenkins pour la première fois, voici l'écran que vous verriez.

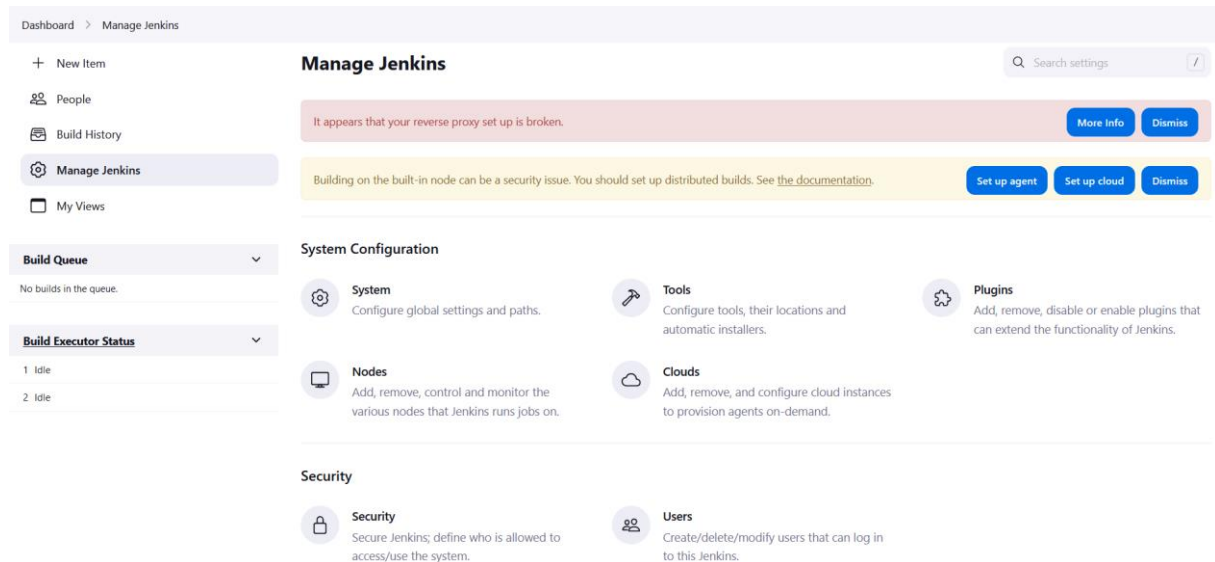


- Sur le côté gauche de l'écran, en haut se trouve le menu pour créer de nouveaux projets, pour gérer jenkins, pour créer des utilisateurs, etc.
- Juste en dessous du menu se trouve la file d'attente de construction. Tous les travaux programmés pour s'exécuter sont ajoutés à la file d'attente et apparaîtront ici.

- Sous la file d'attente de construction se trouve le statut de l'exécuteur de compilation. Cela montre l'état des travaux en cours d'exécution en temps réel.
- En bas à droite de la page se trouvent les informations sur la version de Jenkins affichées.

## Configuration de la sécurité globale

- Sélectionnez Manage Jenkins -> Security



- Vérifier
  - la case à cocher "Activer la sécurité" est cochée
  - Dans Security Realm, "Jenkins own database" est sélectionné
  - L'autorisation est définie sur "Les utilisateurs connectés peuvent tout faire"

Observez les configurations et vérifiez selon la capture d'écran ci-dessous



## Configure Global Security

☒ Enable security

TCP port for JNLP agents ☒ Fixed :  ☐ Random ☐ Disable

Agent protocols...

Disable remember me ☐

Access Control

### Security Realm

☐ Delegate to servlet container

☒ Jenkins' own user database

☐ Allow users to sign up

### Authorization

☐ Anyone can do anything

☐ Legacy mode

☒ Logged-in users can do anything

☐ Allow anonymous read access

Markup Formatter

Plain text

Treats all input as plain text. HTML unsafe characters like < and & are escaped to their resp

☒ Prevent Cross Site Request Forgery exploits

Crumbs

### Crumb Algorithm

☒ Default Crumb Issuer

☐ Enable proxy compatibility

### Plugin Manager

☐ Use browser for metadata download

☒ Enable Slave → Master Access Control

Rules can be tweaked [here](#)

Save

Apply

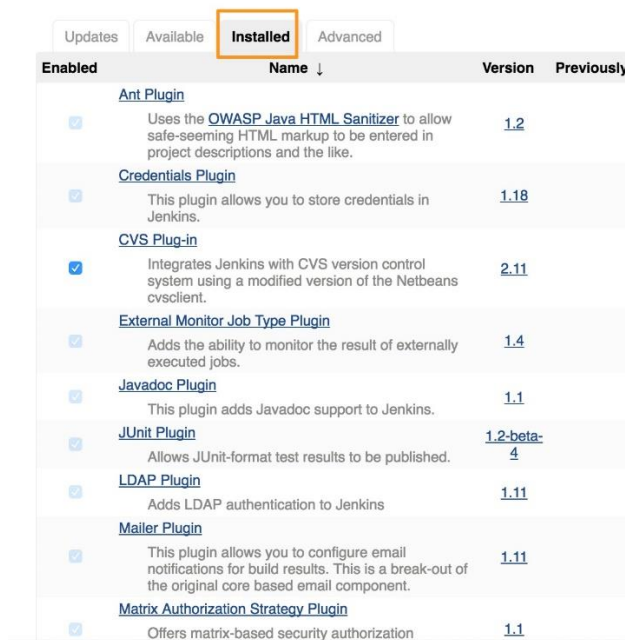
## Plugins

La vraie force de Jenkins réside dans son écosystème riche en plugins. C'est ainsi que les outils s'intègrent à Jenkins pour créer un flux de travail CI. Vous voulez déclencher des jobs Jenkins après chaque changement dans git, vous avez un plugin pour cela. Vous souhaitez envoyer une notification à vos développeurs sur une compilation réussie ou échouée, vous disposez d'un plugin de notification. Vous souhaitez utiliser un outil pour récupérer ou pousser les artefacts de construction, vous avez un plugin pour cela. C'est ainsi que la plupart des outils parlent à Jenkins.

Dans ce Lab, nous allons voir un processus simple pour installer des plugins. Dans ce cadre, nous allons installer un plugin qui nous aidera à intégrer Jenkins à notre référentiel git.

## Explorer les configurations de plugins

- Dans «Administrer Jenkins», sélectionnez l'option «Gestion des plugins».
- Dans le volet Gérer les plugins, vous verrez les onglets suivants,
  - Mises à jour
  - Disponibles
  - installés



Updates	Available	Installed	Advanced
Enabled	Name ↓	Version	Previously
<input checked="" type="checkbox"/>	<a href="#">Ant Plugin</a> Uses the <a href="#">OWASP Java HTML Sanitizer</a> to allow safe-seeming HTML markup to be entered in project descriptions and the like.	<a href="#">1.2</a>	
<input checked="" type="checkbox"/>	<a href="#">Credentials Plugin</a> This plugin allows you to store credentials in Jenkins.	<a href="#">1.18</a>	
<input checked="" type="checkbox"/>	<a href="#">CVS Plug-in</a> Integrates Jenkins with CVS version control system using a modified version of the Netbeans cvsclient.	<a href="#">2.11</a>	
<input checked="" type="checkbox"/>	<a href="#">External Monitor Job Type Plugin</a> Adds the ability to monitor the result of externally executed jobs.	<a href="#">1.4</a>	
<input checked="" type="checkbox"/>	<a href="#">Javadoc Plugin</a> This plugin adds Javadoc support to Jenkins.	<a href="#">1.1</a>	
<input checked="" type="checkbox"/>	<a href="#">JUnit Plugin</a> Allows JUnit-format test results to be published.	<a href="#">1.2-beta-4</a>	
<input checked="" type="checkbox"/>	<a href="#">LDAP Plugin</a> Adds LDAP authentication to Jenkins	<a href="#">1.11</a>	
<input checked="" type="checkbox"/>	<a href="#">Mailer Plugin</a> This plugin allows you to configure email notifications for build results. This is a break-out of the original core based email component.	<a href="#">1.11</a>	
<input checked="" type="checkbox"/>	<a href="#">Matrix Authorization Strategy Plugin</a> Offers matrix-based security authorization	<a href="#">1.1</a>	

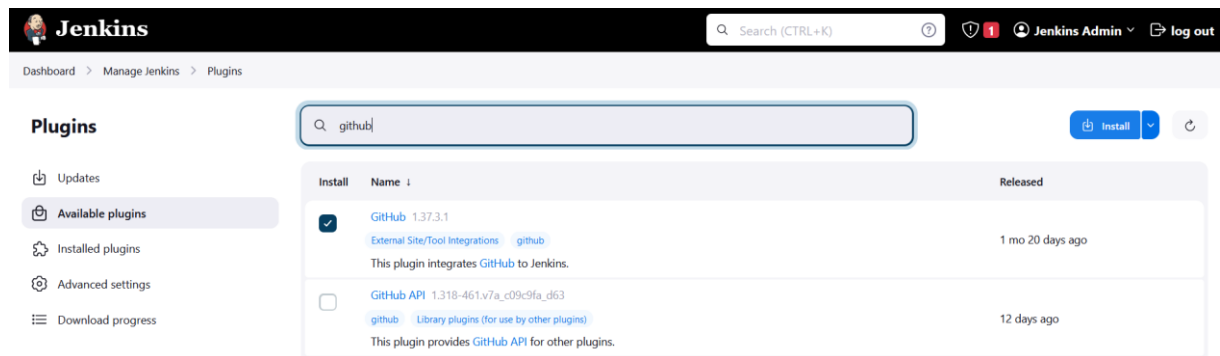
- Avancé

Sélectionnez "Installés" pour afficher la liste des plugins préinstallés avec Jenkins.

## Installer le plugin Github

- Dans "Gérer les plugins", sélectionnez l'onglet **Disponibles**.
- Dans le coin supérieur droit, vous devriez voir une zone de filtre, commencez à taper le terme de recherche dans cette zone. Pour cet exemple, nous taperions **github**.





Jenkins

Search (CTRL+K)

Dashboard > Manage Jenkins > Plugins

Plugins

Search: github

Install

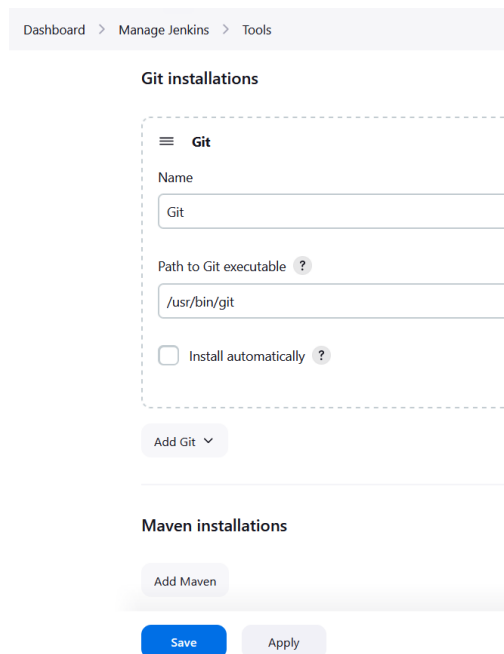
Install	Name	Released
<input checked="" type="checkbox"/>	GitHub 1.37.3.1 External Site/Tool Integrations <a href="#">github</a> This plugin integrates GitHub to Jenkins.	1 mo 20 days ago
<input type="checkbox"/>	GitHub API 1.318-461.v7a_c09c9fa_d63 <a href="#">github</a> Library plugins (for use by other plugins) This plugin provides GitHub API for other plugins.	12 days ago

## Installer Git sur le serveur jenkins

```
# yum install git -y
# which git
/usr/bin/git
```

Une fois installé il faut configurer le plugin :

## Dashboard -> Tools



Dashboard > Manage Jenkins > Tools

Git installations

Git

Name

Git

Path to Git executable ?

/usr/bin/git

☐ Install automatically ?

Add Git

Maven installations

Add Maven

Save Apply

## Apply & Save

## Création du premier projet

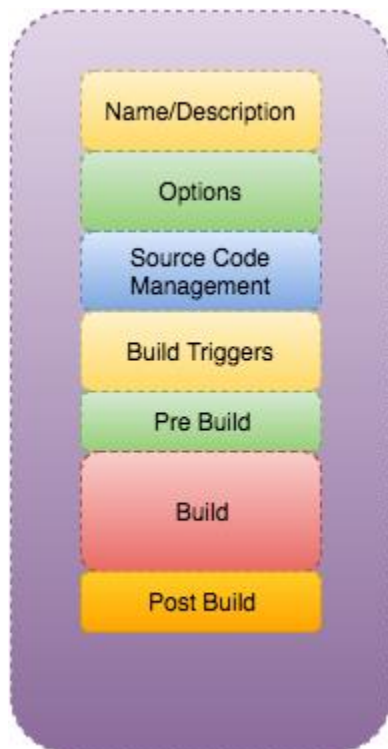
Dans cette session, nous allons créer et lancer notre premier projet avec jenkins. Nous utiliserons un projet de style libre pour cet exemple.

### Types de Jobs

Avec Jenkins, vous pouvez créer les types de projets ou d'emplois suivants.

- Style libre
- Maven
- Travail externe
- Configuration multiple

### Jenkins Jobs Anatomie



Un travail jenkins de style typique comprend les sections suivantes.

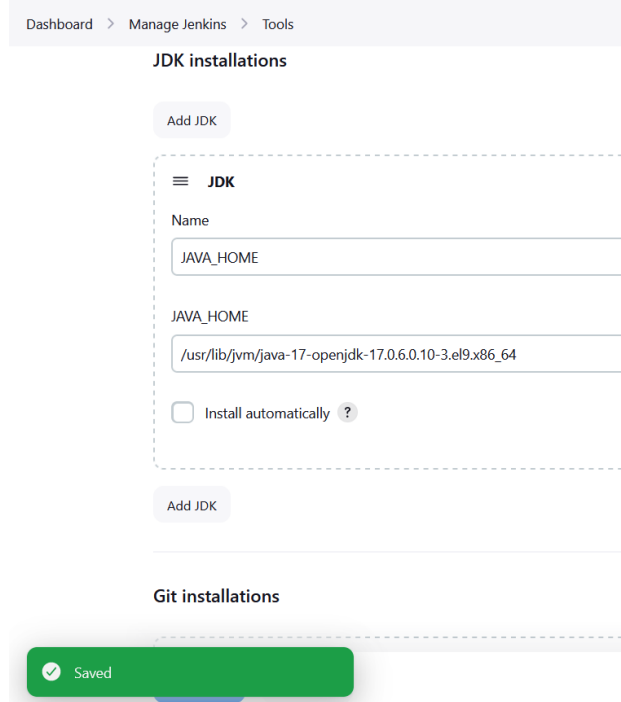
- Nom / description
- Option avancée
- Gestion du code source
- Créer des déclencheurs
- Pré-construction
- Construction
- Post construction

## Créer un travail simple

Nous allons tester un job Java, pour cela il faut configurer le plugin JDK de jenkins

Sur le serveur jenkins :

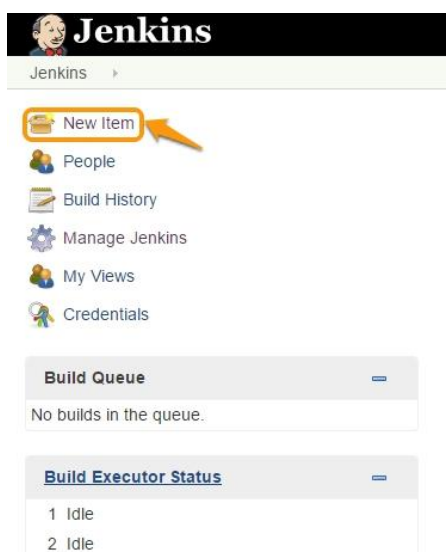
```
# echo $JAVA_HOME  
/usr/lib/jvm/java-17-openjdk-17.0.6.0.10-3.el9.x86_64
```



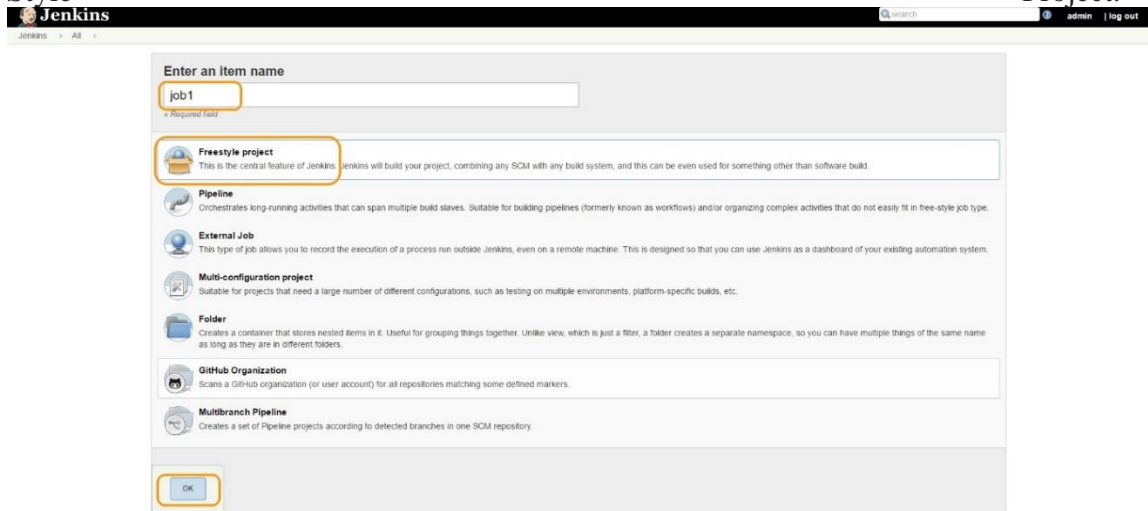
Apply & Save

On peut maintenant créer un travail simple en utilisant jenkins pour exécuter un programme hello world.

1. À partir de la page principale de Jenkins, cliquez sur **Nouvel élément**.



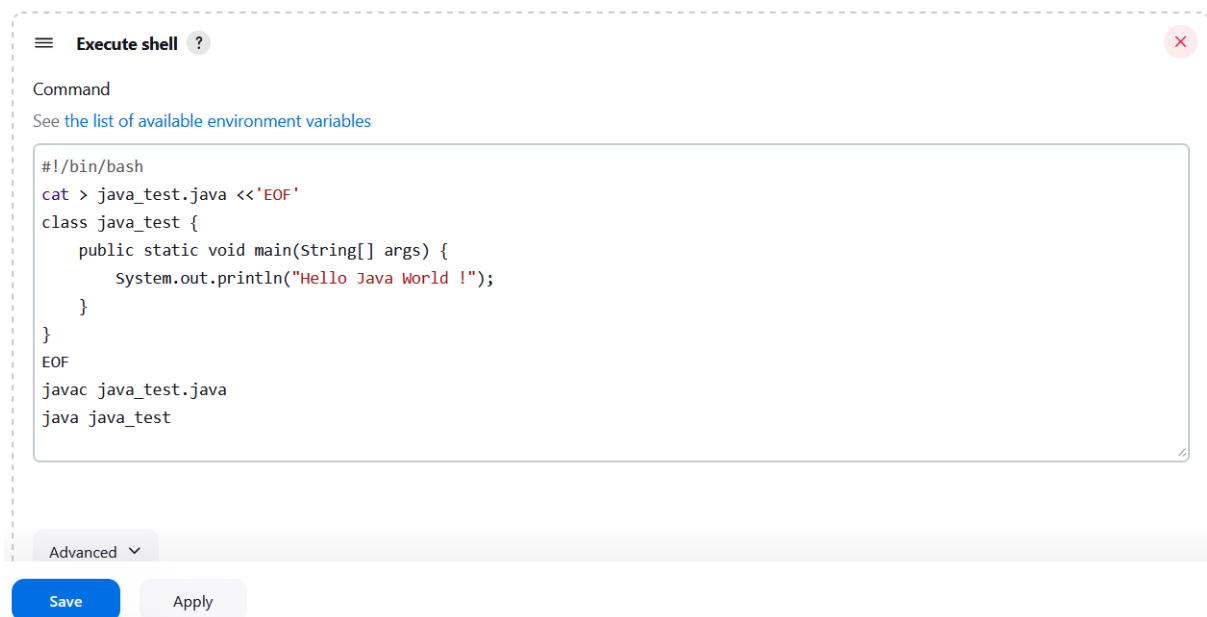
- Donnez un nom au projet dans Nom de l'élément, c'est-à-dire "job1". Sélectionnez Free Style Project.



L'écran suivant ouvre la page de configuration du travail. Sur la page de configuration du travail,

- Ajoutez une description de job par exemple, "Notre premier travail Jenkins".
- Ignorez la « gestion du code source » et « Ce qui déclenche le build », puis faites défiler jusqu'à **Build steps**.
- Dans "Ajouter une étape de construction", sélectionnez **Exécuter un script Shell** et fournissez l'exemple de code suivant,

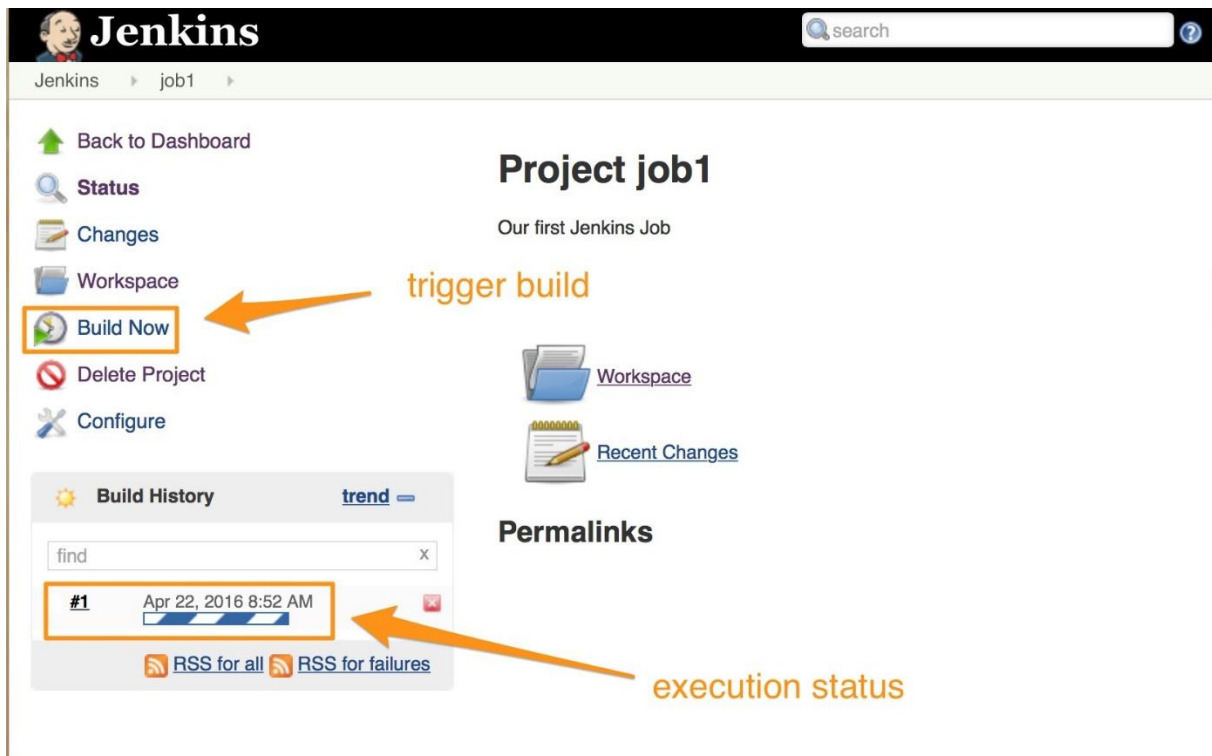
#### Build Steps



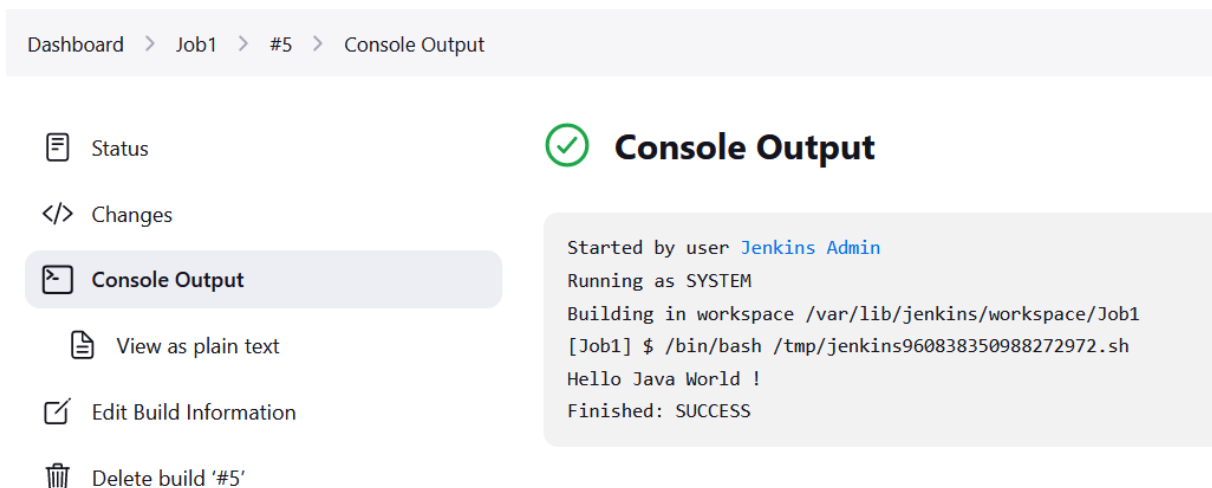
Apply & Save pour accéder à la page du projet.

## Créer un travail pour la première fois

Cliquez sur **Build Now** pour lancer un build. Une fois la construction démarrée, vous verrez le statut dans la section **Historique de build**.

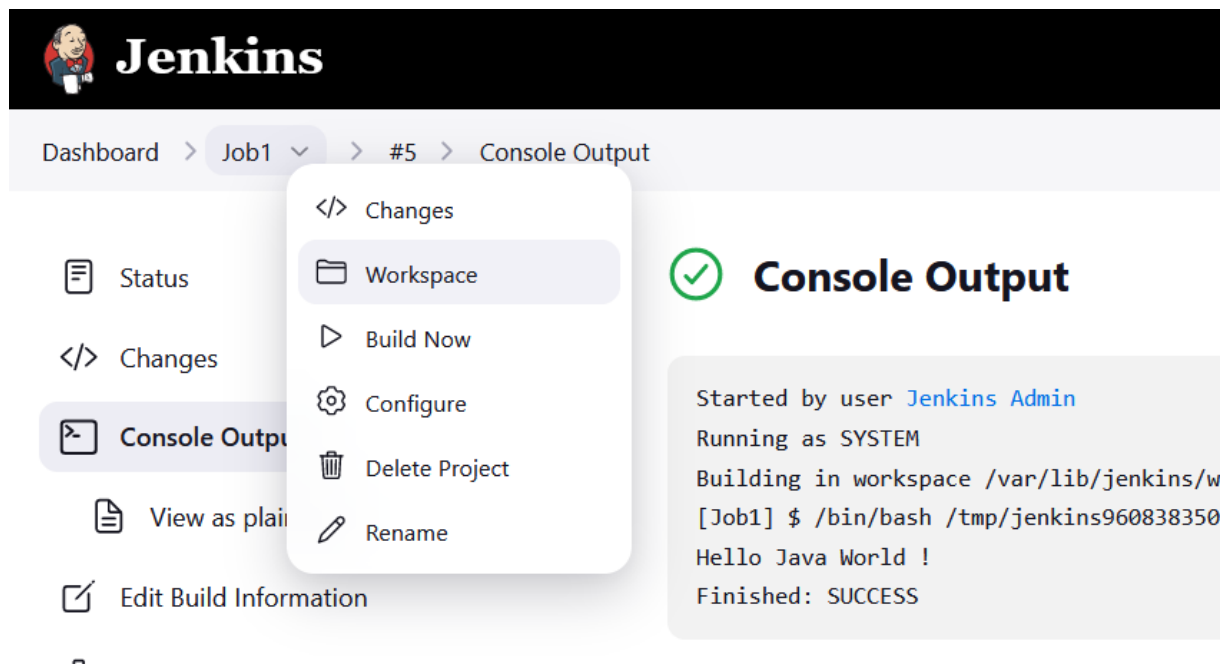


Une fois la construction terminée, cliquez sur le numéro de version qui commence par #. En cliquant sur le numéro de build, par exemple **#1**, vous accédez à la page qui affiche les statistiques de build. L'option intéressante sur cette page peut être **l'état de la console**, qui affiche la sortie au moment de l'exécution.

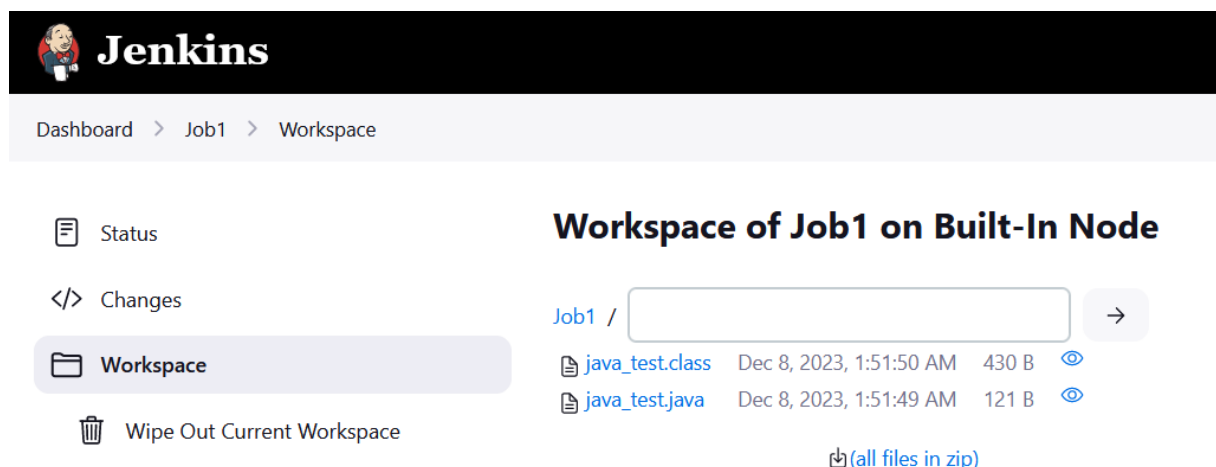


Notez le répertoire « Building in workspace »

Revenez au Workspace du Job1



Notez les fichiers présents dans le répertoire workspace



## Ajout de déclencheurs aux jobs

### Créer des déclencheurs

Les déclencheurs de construction décident quand une tâche Jenkins est exécutée. Que cela se produise sur la base d'un événement externe, par exemple un push to git repository, une exécution planifiée, ou un travail est exécuté après qu'un autre travail est terminé, il existe de nombreuses options pour déclencher des compilations.

Depuis la page du projet, cliquez sur **Configurer**.

## Types de déclencheurs

1. Déclencher les builds à distance (Par exemple, à partir de scripts)
2. Construire après le build sur d'autres projets
3. Construire périodiquement
4. GitHub hook trigger for GITScm polling
5. Scrutation de l'outil de gestion de version

## Déclencher les builds à distance

Les jobs peuvent être déclenchés à distance en dehors de Jenkins. Ceci est très utile lorsque vous souhaitez que les travaux soient déclenchés en fonction d'un événement ou d'une partie de la logique que vous avez écrite dans le cadre de votre script. C'est également de cette manière que vous déclencheriez le travail en fonction des activités effectuées sur les référentiels. par exemple, ajouter un nouveau commit à git hub.

Exemple:

- Cliquez sur *job1* puis sélectionnez Configure
- Dans **Build Trigger** , cochez **Trigger Builds à distance** .

**Build Triggers**

☒ Trigger builds remotely (e.g., from scripts) ?

Authentication Token  ?

Use the following URL to trigger build remotely: JENKINS\_URL/job/job1/build?token=TOKEN\_NAME or /buildWithParameters?token=TOKEN\_NAME  
Optionally append &cause=Cause+Text to provide text that will be included in the recorded build cause.

☐ Build after other projects are built ?

☐ Build periodically ?

☐ GitHub hook trigger for GITScm polling ?

☐ Poll SCM ?

- Définir un jeton
- Sauvegardez le travail.

- Pour déclencher le travail, vous avez besoin de deux choses
  - utilisateur
  - jeton API de l'utilisateur
- Nous utiliserons le jeton d'API de l'utilisateur admin (premier utilisateur que nous avons créé) pour déclencher cette tâche. Vous pouvez le trouver sur **jenkins\_homepage -> Utilisateurs -> admin -> configurer**

### API Token

Current token(s)

There are no registered tokens for this user.

### API Token

Current token(s)

There are no registered tokens for this user.





### API Token

Current token(s)

1138e723af862e614e3c05a9c45d4513b4

⚠ Copy this token now, because it cannot be recovered in the future.



- Accédez au déclencheur à partir du navigateur ou utilisez curl

user:<API\_TOKEN>@<Jenkins\_URL>/job/job1/build?token=<JOB\_TOKEN>

Example:

http://admin:552dab89b070c0fcc3fad281c51318ad@10.40.1.14:8080/job/job1/build?token=mytoken

- Cela déclenchera la construction.

**Historique des builds**

tendance

**#3**

(pending—En période d'attente. Expire dans 1.6 s)

**#2**

20 déc. 2020 19:31

**#1**

20 déc. 2020 18:57



## **Construire après le build sur d'autres projets (pipeline)**

L'une des caractéristiques importantes de Jenkins est sa capacité à créer un pipeline de Jobs, alors qu'en fonction du résultat d'un travail, un autre peut être déclenché. Par exemple seulement si vous êtes capable de compiler le code, vous voudrez peut-être procéder au test, sinon il est inutile de le faire. L'utilisation de Build après que d'autres projets sont déclenchés, cela peut être facilement réalisé. Nous allons créer un pipeline de travaux en utilisant cette fonctionnalité dans le prochain Lab.

## **Construire périodiquement**

Semblable à la création de cronjobs ou de travaux planifiés, il est possible de définir un calendrier d'exécution avec jenkins.

## **Scrutation de l'outil de gestion de version**

Cette option permet à jenkins de vérifier régulièrement le système de gestion du code source, par exemple un référentiel git distant pour vérifier s'il y a des mises à jour, et de lancer un travail basé sur celui-ci. Idéalement, les hooks / webhooks de commit avec git devraient déclencher les builds, mais cela n'est pas toujours possible. Par exemple, si vous hébergez des jenkins dans un réseau privé, qui n'est pas accessible par le référentiel git hébergé sur le cloud, le déclenchement d'un webhook ne sera pas possible. Dans de tels cas, la meilleure option suivante est d'interroger le référentiel git à intervalles réguliers et de déclencher les builds.

## **GitHub hook trigger for GITScm polling**

Si Jenkins reçoit le hook PUSH GitHub du référentiel défini dans la section Git SCM, il déclenchera la logique d'interrogation Git SCM. La logique d'interrogation appartient donc en fait à Git SCM.