



DevOps

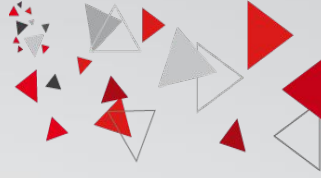
Chapitre 2 : Jenkins



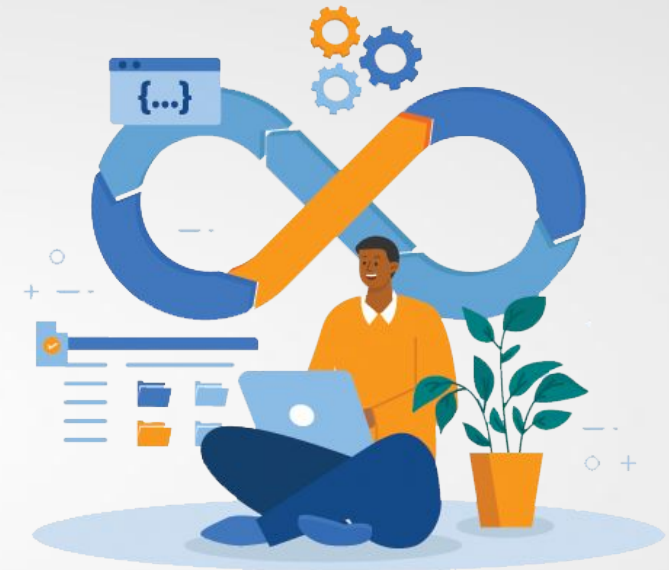
ESPRIT – UP ASI (Architecture des Systèmes d'Information)
Bureau E204



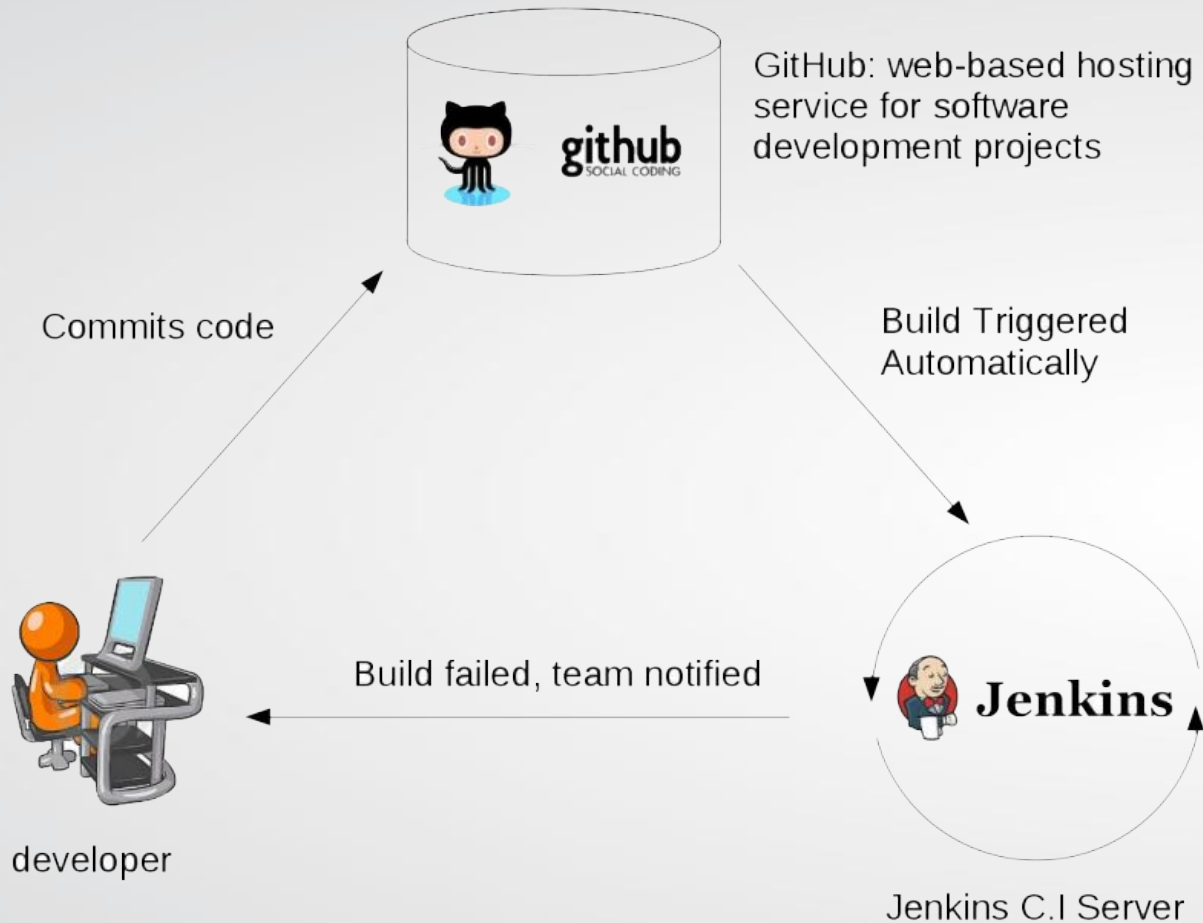
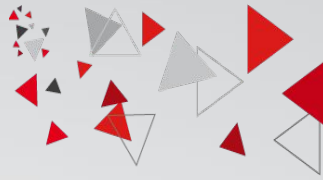
► Plan du cours



- Intégration Continue (Continuous Integration : **CI**)
- Définition de Jenkins
- Installation de Jenkins
- Configuration de Jenkins
- Configuration d'un projet avec Jenkins
- Travail à faire



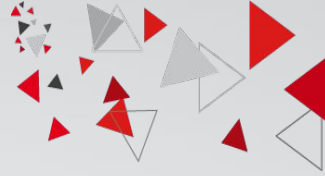
► Qu'est-ce que l'intégration continue ?



- Les développeurs soumettent régulièrement leur code à un référentiel partagé.
- Le système de contrôle de version est surveillé. Lorsqu'une soumission est détectée, une construction sera déclenchée automatiquement.
- Si la construction échoue, les développeurs seront immédiatement notifiés.



Besoin de l'intégration continue ?



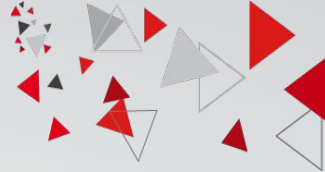
- **Détecter les problèmes** ou les bugs **le plus tôt possible** dans le cycle de développement.
- **Intégrer** l'ensemble de la base de code, la **construire** et la **tester** en permanence pour attraper les erreurs potentielles plus tôt dans le cycle de vie, ce qui **résulte** en un logiciel de meilleure qualité.



Jenkins – Définition



Jenkins



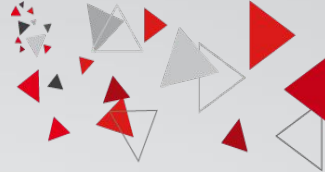
- **Jenkins** : Un serveur open source écrit en Java qui automatise l'intégration continue.
- **Interface Web** : Utilise un serveur web (comme Apache-Tomcat ou Jetty) pour fournir une interface web conviviale.
- **Intégration avec les Gestionnaires de Version** : S'interface avec des systèmes de gestion de versions (CVS, Git, Subversion SVN) et exécute des scripts (Groovy, Yaml, etc.).



Jenkins - Définition



Jenkins



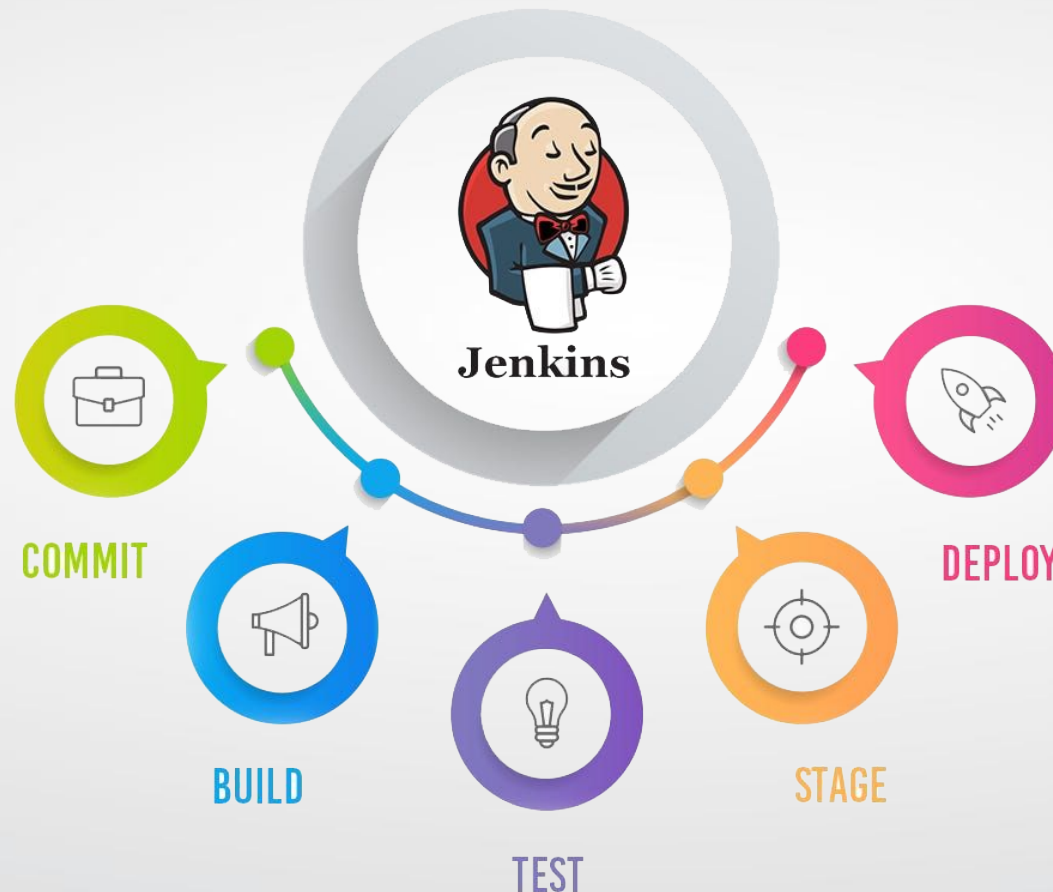
- **Automatisation de la Construction, des Tests et du Déploiement** : Automatise les étapes de construction, de tests et de déploiement, facilitant ainsi l'intégration continue et la livraison continue.
- **Amélioration Continue** : Permet aux développeurs d'intégrer rapidement des modifications au projet, accélérant ainsi l'amélioration continue du produit.



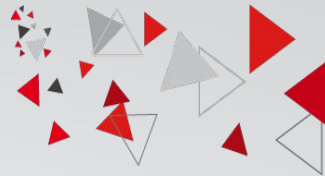
► Jenkins - Pipeline



- Un **Pipeline** est une **combinaison de plugins** qui permettent de soutenir l'intégration et la mise en place de pipelines de livraison continue en utilisant Jenkins.



► Installation de JDK 11



- Allez dans le répertoire de votre machine virtuelle Ubuntu, démarrez la machine virtuelle (`vagrant up`) et ouvrez un terminal (`vagrant ssh`).

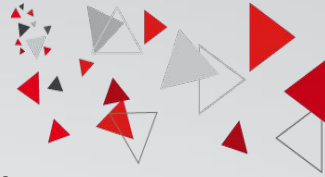
Pour plus d'informations, consultez le support “[1- Installation Vagrant-Ubunto](#)”

```
Sélection vagrant@vagrant: ~
PS D:\Vagrant\Ubuntu> vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Checking if box 'bento/ubuntu-22.04' version '202309.08.0' is up to date...
    default: /vagrant => D:/Vagrant/Ubuntu
==> default: Machine already provisioned. Run `vagrant provision` or use the `--provision`
==> default: flag to force provisioning. Provisioners marked to run always will still run.
PS D:\Vagrant\Ubuntu> vagrant ssh
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-83-generic x86_64)

Last login: Mon Sep 11 14:26:05 2023 from 10.0.2.2
vagrant@vagrant: $
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-83-generic x86_64)
```



► Installation de JDK 11



- Tapez les commandes suivantes pour installer la OpenJDK (par défaut, Ubuntu installe la version 11 :

```
sudo apt update
```

```
sudo apt install default-jdk
```



- Pour vérifier que le JDK 11 est bien installé, exécutez la commande suivante :

```
vagrant@vagrant:~$ java -version
openjdk version "11.0.20.1" 2023-08-24
OpenJDK Runtime Environment (build 11.0.20.1+1-post-Ubuntu-0ubuntu122.04)
OpenJDK 64-Bit Server VM (build 11.0.20.1+1-post-Ubuntu-0ubuntu122.04, mixed mode, sharing)
vagrant@vagrant:~$
```



► Installation de JDK 11



- Pour que les applications Java puissent trouver la machine virtuelle Java de manière précise, il est nécessaire de configurer la variable d'environnement "JAVA_HOME".

1- Ouvrir le fichier de configuration système /etc/environment :

```
vagrant@vagrant:~$ sudo nano /etc/environment
```



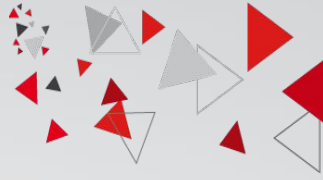
2- Ajouter la variable d'environnement suivante dans le fichier /etc/environment :

```
JAVA_HOME="/usr/lib/jvm/java-11-openjdk-amd64/"
```

```
vagrant@vagrant: ~  
GNU nano 6.2 /etc/environment *  
PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/ga  
JAVA_HOME="/usr/lib/jvm/java-11-openjdk-amd64/"
```



► Installation de JDK 11



3- Appliquer les modifications en utilisant la commande suivante :

```
source /etc/environment
```

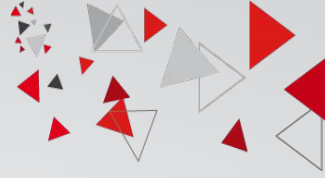
4- Utiliser la commande suivante pour vérifier que JAVA_HOME a été configuré correctement : `echo $JAVA_HOME`



```
vagrant@vagrant:~$ source /etc/environment
vagrant@vagrant:~$ echo $JAVA_HOME
/usr/lib/jvm/java-11-openjdk-amd64/
vagrant@vagrant:~$ |
```



► Installation de Maven



- Pour procéder à l'installation de Maven, veuillez ouvrir votre terminal et exécuter les commandes suivantes de manière consécutive :

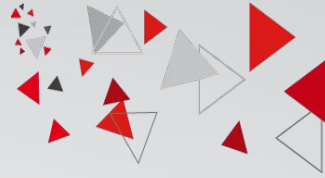
```
sudo apt install maven -y  
M2_HOME="opt/apache-maven-3.6.3"  
PATH="$M2_HOME/bin:$PATH"  
export PATH
```



```
vagrant@vagrant:~$ sudo apt install maven -y  
M2_HOME="opt/apache-maven-3.6.3"  
PATH="$M2_HOME/bin:$PATH"  
export PATH  
Reading package lists... Done
```



► Installation de Maven



- Pour vérifier que le Maven est bien installé, exécuter la commande suivante :

```
vagrant@vagrant:~$ mvn -version
Apache Maven 3.6.3
Maven home: /usr/share/maven
Java version: 11.0.20.1, vendor: Ubuntu, runtime: /usr/lib/jvm/java-11-openjdk-amd64
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "5.15.0-67-generic", arch: "amd64", family: "unix"
vagrant@vagrant:~$ |
```

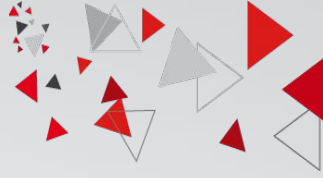


- Ouvrir le fichier de configuration système /etc/environment et définir la variable d'environnement de Maven "M2_HOME" :

M2_HOME="opt/apache-maven-3.6.3"

```
vagrant@vagrant: ~
GNU nano 6.2 /etc/environment *
PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games"
JAVA_HOME="/usr/lib/jvm/java-11-openjdk-amd64/"
M2_HOME="opt/apache-maven-3.6.3"
```

► Installation de Git



- Pour pouvoir utiliser Git avec Jenkins, vous devez installer Git sur votre machine virtuelle (VM). Aucune configuration de Git dans Jenkins ne sera nécessaire.

```
sudo apt install git
```

```
vagrant@vagrant:~$ sudo apt install git
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```



- Pour vérifier que le Git est bien installé, exécuter la commande suivante :

```
git --version
```

```
vagrant@vagrant:~$ git --version
git version 2.34.1
```



► Installation de Jenkins Jenkins

- En exécutant les commandes suivantes en ordre, vous **installez, configurez et démarrez *Jenkins*** sur votre système Ubuntu, prêt à être utilisé pour l'intégration continue et l'automatisation des tâches de développement.

```
wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add -  
sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ >  
/etc/apt/sources.list.d/jenkins.list'  
  
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 5BA31D57EF5975CA  
  
sudo apt update  
  
sudo apt install jenkins  
  
sudo systemctl start jenkins  
  
sudo systemctl enable jenkins
```



► Installation de Jenkins Jenkins

Voici une description simple de chaque commande :

1. **Ajouter la clé Jenkins** : Cette commande télécharge une clé de sécurité pour Jenkins et la rend utilisable par le système.

```
wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add -
```

2. **Ajouter la source Jenkins** : Cette commande configure le système pour télécharger les paquets Jenkins depuis un emplacement spécifique sur le web.

```
sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ >  
/etc/apt/sources.list.d/jenkins.list'
```

3. **Récupérer une autre clé** : Cette commande obtient une clé de sécurité supplémentaire nécessaire pour Jenkins.

```
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 5BA31D57EF5975CA
```



► Installation de Jenkins Jenkins

4. **Mettre à jour la liste des paquets** : Cette commande actualise la liste des logiciels disponibles pour installation, y compris Jenkins.

```
sudo apt update
```

5. **Installer Jenkins** : Cette commande installe Jenkins et ses composants associés.

```
sudo apt install jenkins
```

6. **Démarrer Jenkins** : Cette commande démarre Jenkins en tant que service.

```
sudo systemctl start jenkins
```

7. **Activer le démarrage automatique de Jenkins** : Cette commande configure Jenkins pour qu'il démarre automatiquement lorsque le système démarre.

```
sudo systemctl enable jenkins
```



► Installation de Jenkins Jenkins

- Pour vérifier l'installation de Jenkins, vous pouvez lancer la commande suivante :

```
sudo systemctl status jenkins
```

```
vagrant@vagrant: ~  
● jenkins.service - Jenkins Continuous Integration Server  
   Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: enabled)  
   Active: active (running) since Mon 2023-09-18 23:25:12 UTC; 15min ago  
     Main PID: 6917 (java)  
       Tasks: 43 (limit: 5629)  
      Memory: 1.3G  
         CPU: 2min 9.002s  
    CGroup: /system.slice/jenkins.service  
           └─6917 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080  
  
Sep 18 23:24:14 vagrant jenkins[6917]: WARNING: An illegal reflective access operation has occurred  
Sep 18 23:24:14 vagrant jenkins[6917]: WARNING: Illegal reflective access by org.codehaus.groovy.vmplugin.v7.Java7$1 (file:/var/cache/jenkins/w  
Sep 18 23:24:14 vagrant jenkins[6917]: WARNING: Please consider reporting this to the maintainers of org.codehaus.groovy.vmplugin.v7.Java7$1  
Sep 18 23:24:14 vagrant jenkins[6917]: WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations  
Sep 18 23:24:14 vagrant jenkins[6917]: WARNING: All illegal access operations will be denied in a future release  
Sep 18 23:25:11 vagrant jenkins[6917]: 2023-09-18 23:25:11.817+0000 [id=29]      INFO      jenkins.InitReactorRunner$1#onAttained: Complete  
Sep 18 23:25:12 vagrant jenkins[6917]: 2023-09-18 23:25:12.136+0000 [id=22]      INFO      hudson.lifecycle.Lifecycle#onReady: Jenkins is f  
Sep 18 23:25:12 vagrant systemd[1]: Started Jenkins Continuous Integration Server.  
Sep 18 23:25:13 vagrant jenkins[6917]: 2023-09-18 23:25:13.286+0000 [id=46]      INFO      h.m.DownloadService$Downloadable#load: Obtained  
Sep 18 23:25:13 vagrant jenkins[6917]: 2023-09-18 23:25:13.297+0000 [id=46]      INFO      hudson.util.Retrier#start: Performed the action
```



► Installation de Jenkins Jenkins

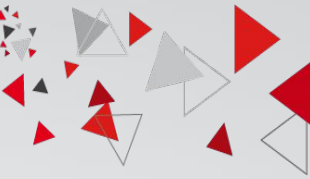
- Pour accéder à Jenkins, vous devez obtenir l'adresse IP de la machine virtuelle en utilisant la commande suivante : `ip addr show`

```
vagrant@vagrant: ~  
vagrant@vagrant:~$ ip addr show  
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000  
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00  
    inet 127.0.0.1/8 scope host lo  
        valid_lft forever preferred_lft forever  
    inet6 ::1/128 scope host  
        valid_lft forever preferred_lft forever  
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000  
    link/ether 08:00:27:10:0b:45 brd ff:ff:ff:ff:ff:ff  
    altnam enp0s3  
    inet 10.0.2.15/24 metric 100 brd 10.0.2.255 scope global dynamic eth0  
        valid_lft 66174sec preferred_lft 66174sec  
    inet6 fe80::a00:27ff:fe10:b45/64 scope link  
        valid_lft forever preferred_lft forever  
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000  
    link/ether 08:00:27:a7:5f:66 brd ff:ff:ff:ff:ff:ff  
    altnam enp0s8  
    inet 192.168.33.10/24 brd 192.168.33.255 scope global eth1  
        valid_lft forever preferred_lft forever  
    inet6 fe80::a00:27ff:fea7:5f66/64 scope link  
        valid_lft forever preferred_lft forever  
vagrant@vagrant:~$ |
```

► Installation de Jenkins



Jenkins



S'identifier [Jenkins]

Non sécurisé 192.168.33.10:8080/login?from=%2F

Démarrage

Débloquer Jenkins

Pour être sûr que Jenkins soit configuré de façon sécurisée par un administrateur, un mot de passe a été généré dans le fichier de logs ([où le trouver](#)) ainsi que dans ce fichier sur le serveur :

```
/var/lib/jenkins/secrets/initialAdminPassword
```

Veuillez copier le mot de passe depuis un des 2 endroits et le coller ci-dessous.

Mot de passe administrateur

Continuer



► Installation de Jenkins Jenkins

- Lors de l'installation de Jenkins, un fichier contenant le mot de passe initial d'administration est généré et stocké sur le système. Vous devez accéder à ce mot de passe pour débloquer Jenkins.
- Pour afficher le mot de passe, utilisez la commande suivante :

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



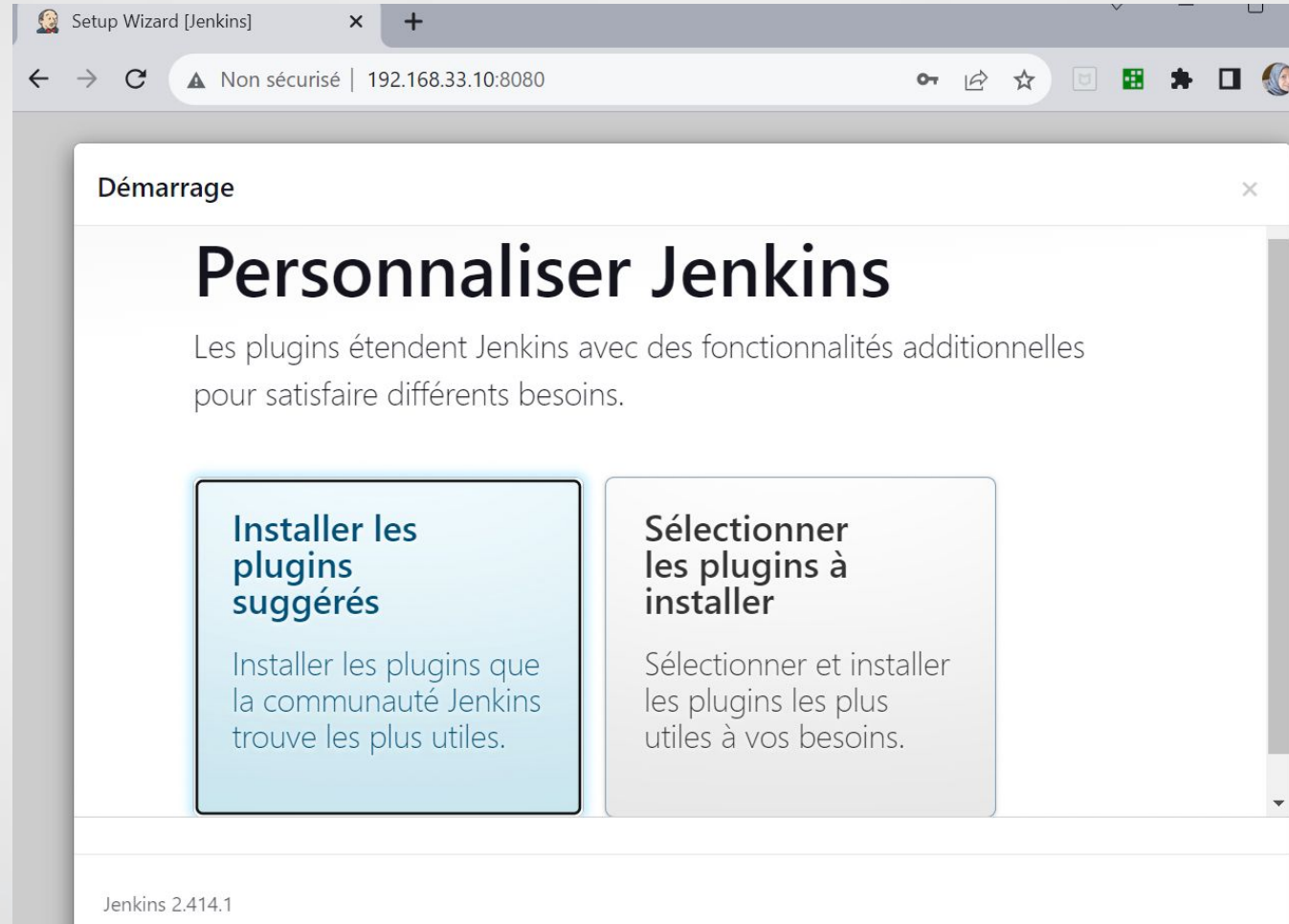
```
vagrant@vagrant: ~  
vagrant@vagrant:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword  
06eb0a14bc7e4ae7aa8253949d8ba92a  
vagrant@vagrant:~$ |
```

Veuillez copier le mot de passe depuis un des 2 endroits et le coller ci-dessous.

Mot de passe administrateur

► Installation de Jenkins Jenkins

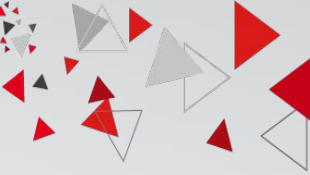
- Installer les plugins suggérés



Installation de Jenkins



Jenkins



Setup Wizard [Jenkins]

Non sécurisé | 192.168.33.10:8080

Installation en cours...

Installation en cours...

✓ Folders	Workspace Cleanup	Build Timeout	Credentials Binding
Timestamp	Workspace Cleanup	Ant	Gradle
Pipeline	GitHub Branch Source	Pipeline: GitHub Groovy Libraries	Pipeline: Stage View
Git	SSH Build Agents	Matrix Authorization	PAM Authentication

** Instance Identity
** JavaBeans Activation Framework (JAF) API
** JavaMail API
** Pipeline: Step API
** Token Macro

Build Timeout

** Credentials
** Plain Credentials
** Trilead API

** - dépendance requise

Jenkins 2.414.1



► Installation de Jenkins Jenkins

- Vous n'êtes pas obligé de créer de nouveaux utilisateurs, vous pouvez continuer à utiliser l'utilisateur "**admin**".

Démarrage

Créer le 1er utilisateur Administrateur

Nom d'utilisateur:

Mot de passe:

Confirmation du mot de passe:

Nom complet:

Adresse courriel:

Jenkins 2.414.1

[Continuer en tant qu'Administrateur](#) [Sauver et continuer](#)



► Installation de Jenkins Jenkins

- Pour personnaliser le numéro de port de Jenkins, vous pouvez simplement le modifier dans cette fenêtre. Vous avez la possibilité de choisir de maintenir le port 8080 ou d'en sélectionner un autre selon vos préférences.

Démarrage

Configuration de l'instance

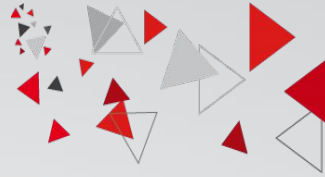
URL de Jenkins :

L'URL de Jenkins est utilisée pour fournir l'URL de base pour les liens absolus vers les diverses ressources Jenkins. Cela signifie que cette valeur est nécessaire pour le bon fonctionnement de nombreuses fonctionnalités de Jenkins, notamment les notifications par mail, les mises à jour des statuts des pull requests, et la variable d'environnement BUILD_URL fournie pour les étapes de build.

La valeur par défaut affichée **n'est pas encore sauvegardée** et est générée à partir de la requête actuelle, lorsque c'est possible. Il est fortement recommandé d'utiliser comme valeur l'URL qui est censée être utilisée par les utilisateurs. Cela évitera des confusions lors du partage ou de la visualisation de liens.



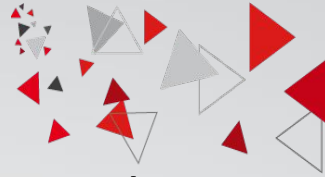
► Configuration de Jenkins



- Vous pouvez changer le mot de passe de "**admin**" à "**jenkins**" par exemple :



► Configuration de Jenkins

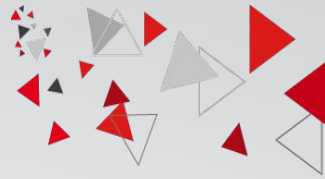


- Jenkins offre une interface web conviviale et intuitive, permettant un accès direct à toutes les configurations disponibles et fournissant des informations complètes sur tous les jobs.

The screenshot shows the Jenkins web interface. At the top is a black header with the Jenkins logo, a search bar labeled 'rechercher (CTRL+K)', and user information 'admin' with a 'se déconnecter' link. Below the header is a 'Tableau de bord' section. On the left is a sidebar with links: '+ Nouveau Item', 'Utilisateurs', 'Historique des constructions', 'Administrer Jenkins', and 'Mes vues'. The main content area has a 'Bienvenue sur Jenkins !' message, followed by 'Commencer à créer votre projet' with a 'Créer un job' button, and 'Configurer un build distribué' with buttons for 'Mettre en place un agent', 'Configurer un cloud', and a link 'En apprendre plus sur les builds distribués'.



► Configuration de Jenkins



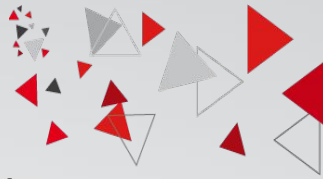
- Pour installer des plugins, il vous suffit d'accéder à la fenêtre de "**Plugins**".

The screenshot shows the Jenkins 'Plugins' page. The top navigation bar includes the Jenkins logo, a search bar with the text 'rechercher (CTRL+K)', and user information 'admin' with a dropdown arrow and a 'se déconnecter' link. The breadcrumb trail reads 'Tableau de bord > Administrer Jenkins > Plugins'. The left sidebar contains links to 'Updates', 'Available plugins' (which is highlighted), 'Installed plugins', 'Advanced settings', and 'Download progress'. The main content area is titled 'Plugins' and features a search bar labeled 'Search available plugins'. To the right of the search bar are an 'Install' button and a refresh icon. Below these elements is a table of installed plugins.

Install	Name ↓	Released
<input type="checkbox"/>	Command Agent Launcher 107.v773860566e2e Agent Management Allows agents to be launched using a specified command.	1 mo. 12 j ago
<input type="checkbox"/>	Oracle Java SE Development Kit Installer 73.vddf737284550 Allows the Oracle Java SE Development Kit (JDK) to be installed via download from Oracle's website.	1 mo. 16 j ago



► Configuration de Jenkins - plugins

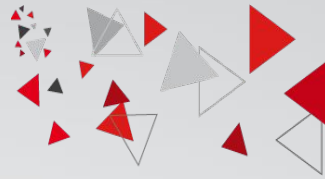


- Pour créer notre chaîne d'intégration continue, nous allons installer les plugins suivants dans Jenkins (Installez ces plugins sans redémarrer, puis redémarrez Jenkins à la fin) :

- Git plugin (normalement déjà installé, mais vérifier)
- Maven Integration
- Sonargraph Integration
- SonarQube Scanner



► Configuration de Jenkins - plugins








**Jenkins**


Tableau de bord > Administrer Jenkins > Plugins

 Updates

 Available plugins

 Installed plugins

 Advanced settings

 Download progress

Download progress

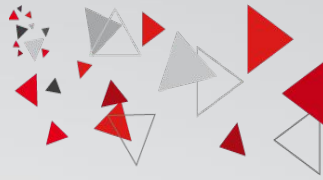
Préparation

- Checking internet connectivity
- Checking update center connectivity
- Success

Ionicons API	✓ Succès
Folders	✓ Succès
OWASP Markup Formatter	✓ Succès
Structs	✓ Succès
bouncycastle API	✓ Succès
Instance Identity	✓ Succès
JavaBeans Activation Framework (JAF) API	✓ Succès
JavaMail API	✓ Succès



► Configuration de Jenkins - plugins



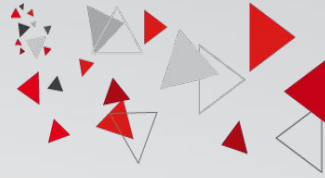
- Après avoir installé les plugins, vous devez redémarrer Jenkins :

```
sudo systemctl restart jenkins
```

```
vagrant@vagrant: ~  
vagrant@vagrant:~$ sudo systemctl restart jenkins  
vagrant@vagrant:~$ |
```



► Configuration de Jenkins - Outils



- Pour configurer les outils, vous devez accéder à la fenêtre "Tools".

The screenshot shows the Jenkins web interface. At the top is a black header with the Jenkins logo, a search bar containing 'rechercher (CTRL+K)', and user information 'admin' with a dropdown arrow and a 'se déconnecter' link. Below the header, a breadcrumb trail reads 'Tableau de bord > Administrer Jenkins'. The main content area is titled 'Administrer Jenkins' and includes a 'Search settings' bar. On the left is a sidebar with navigation links: '+ Nouveau Item', 'Utilisateurs', 'Historique des constructions', 'Administrer Jenkins' (highlighted with a gear icon), and 'Mes vues'. Below these is a 'File d'attente des constructions' section showing an empty queue. The main area is divided into 'System Configuration' with three options: 'System' (gear icon), 'Tools' (hammer icon), and 'Nodes' (monitor icon). The 'Tools' option is highlighted and includes the description: 'Configurer les outils, leur localisation et les installeurs automatiques.'



► Configuration de Jenkins - JDK



Jenkins

Tableau de bord > Administrer Jenkins > Tools

Tools

Configuration Maven

Fournisseur de réglages par défaut

Utiliser les réglages Maven par défaut

Fournisseur de réglages globaux par défaut

Utiliser les réglages globaux Maven par défaut

Installations Sonargraph Build

Ajouter Sonargraph Build

Installations JDK

Ajouter JDK

- Faites référence au JDK déjà installé sur votre VM et enregistrez cette configuration :

```
vagrant@vagrant:~$ echo $JAVA_HOME  
/usr/lib/jvm/java-11-openjdk-amd64/
```

Ajouter JDK

JDK

Nom

JAVA_HOME

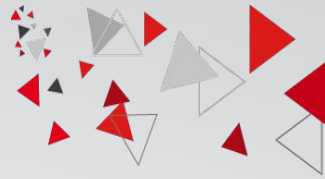
JAVA_HOME

/usr/lib/jvm/java-11-openjdk-amd64/

☐ Install automatically ?



► Configuration de Jenkins - Maven



- Faites référence au Maven déjà installé sur votre VM et enregistrez cette configuration :

```
vagrant@vagrant:~$ echo $M2_HOME  
opt/apache-maven-3.6.3
```

Installations Maven

Ajouter Maven

☰ Maven

Nom

! Required

☒ Install automatically ?

☰ Install from Apache

Version

Ajouter un installateur ▼

← Décocher ce choix

Installations Maven

Ajouter Maven

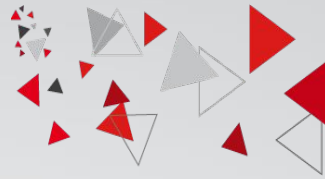
☰ Maven

Nom

MAVEN_HOME



► Configuration de Jenkins - Git



- Rien à configurer

Git installations

≡ Git

Name

Default

Path to Git executable ?

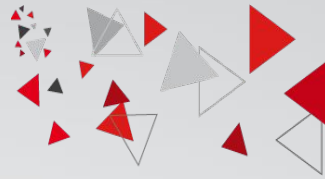
git

☐ Install automatically ?

Add Git ▼



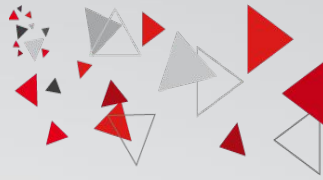
► Configuration d'un projet - Job



- Les tâches ou "**jobs**" sont essentielles pour réaliser la construction (build) dans Jenkins.
- Un projet dans Jenkins est symbolisé par un "**job**" qui englobe plusieurs étapes du processus de build. Tous les projets dans Jenkins suivent ces trois étapes :
 1. Création du job.
 2. Configuration du job, notamment la configuration des étapes du build.
 3. Lancement du build.
- Toutes les étapes du build dans les jobs sont gérées directement grâce à des plugins.



► Configuration d'un projet - Job

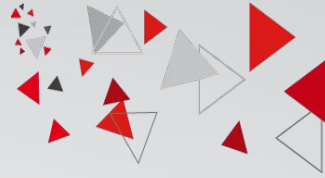


– Les **étapes** pour compiler un projet sont les suivantes :

1. Récupération du projet.
2. Compilation.
3. Exécution des tests unitaires automatiques (JUnit).
4. Exécution des tests de qualité (Sonar).
5. Préparation de la version à distribuer.
6. Distribution de la version (Nexus).



► Configuration d'un projet - Job



- Il y a deux façons de configurer la compilation d'un projet :
 - **Méthode formulaire** (*freestyle*): Vous pouvez configurer le projet en utilisant un formulaire à remplir.
 - **Méthode pipeline** : L'alternative est d'utiliser un *pipeline*, qui configure le projet à l'aide de scripts (en utilisant Groovy). Cette approche permet également de paralléliser les étapes du projet et offre une interface plus conviviale pour lire les logs.



► Configuration d'un projet - *Freestyle*

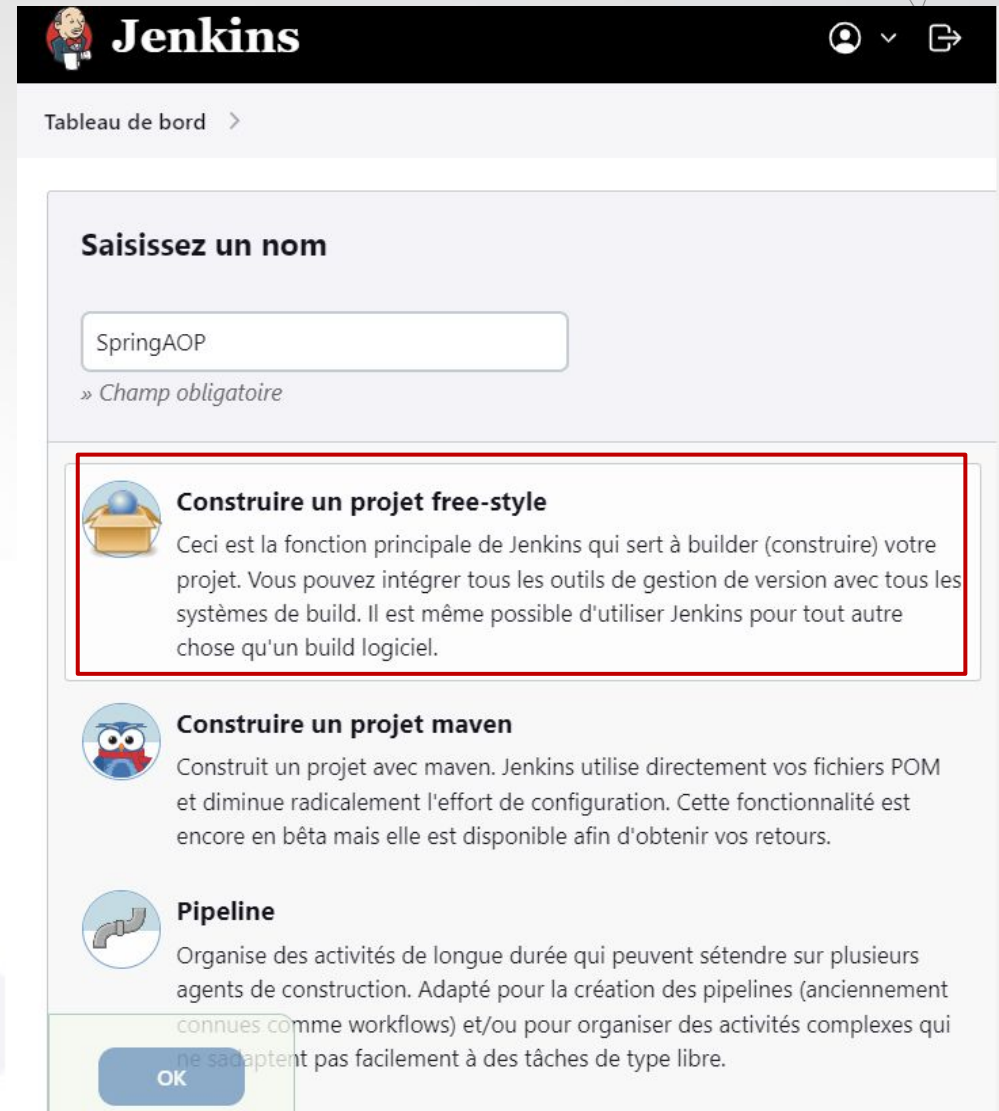
- Pour démontrer le fonctionnement de Jenkins, nous allons utiliser un projet Spring Boot.
- Pour commencer, vous devez créer un projet Jenkins de type "*Freestyle*".

Bienvenue sur Jenkins !

Vos jobs Jenkins seront affichés sur cette page. Pour commencer, vous pouvez mettre en place un build distribué ou commencer à créer un projet.

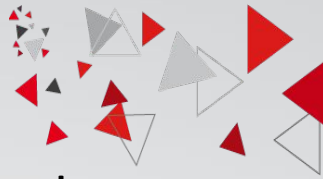
Commencer à créer votre projet

Créer un job



The screenshot shows the Jenkins dashboard. At the top is the Jenkins logo and a navigation bar. Below it is a 'Tableau de bord' (Dashboard) link. The main content area has a section 'Saisissez un nom' (Enter a name) with a text input field containing 'SpringAOP' and a note '» Champ obligatoire' (» Required field). Below this is a red-bordered box containing the 'Construire un projet free-style' (Build a free-style project) option, which is highlighted. Below this box are two other options: 'Construire un projet maven' (Build a maven project) and 'Pipeline'. The 'Construire un projet free-style' option includes a description: 'Ceci est la fonction principale de Jenkins qui sert à builder (construire) votre projet. Vous pouvez intégrer tous les outils de gestion de version avec tous les systèmes de build. Il est même possible d'utiliser Jenkins pour tout autre chose qu'un build logiciel.' The 'Construire un projet maven' option includes a description: '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.' The 'Pipeline' option includes a description: '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.'

► Configuration d'un projet – *Freestyle*



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

Tableau de bord > SpringAOP > Configuration

Configure

- General
- Gestion de code source
- Ce qui déclenche le build
- Environnements de Build
- Build Steps
- Actions à la suite du build

General

Enabled ☒

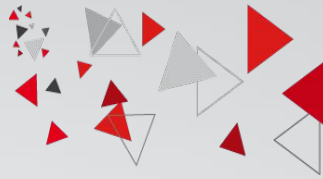
Description

Plain text [Prévisualisation](#)

- ☐ Ce build a des paramètres ?
- ☐ GitHub project
- ☐ Supprimer les anciens builds ?
- ☐ Throttle builds ?
- ☐ Exécuter des builds simultanément si nécessaire ?

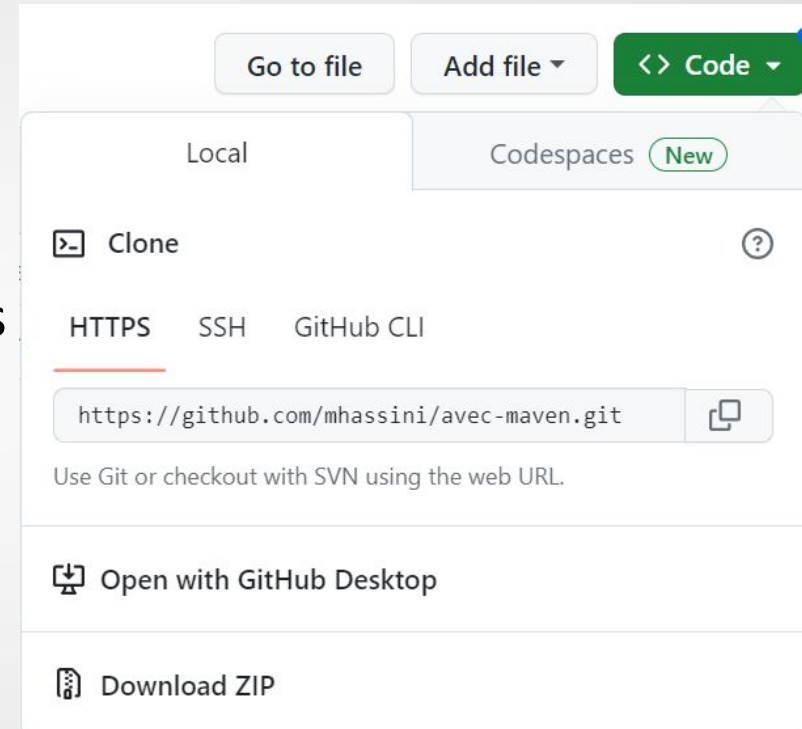


► Configuration d'un projet – *Freestyle*

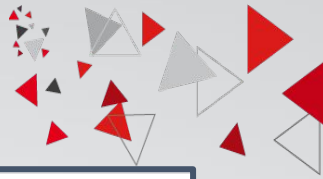


- Pour obtenir du code à partir de Git, vous avez plusieurs options :
 - utilisez des dépôts Git publics
 - configurez des clés SSH
 - saisissez des noms d'utilisateur et des mots de passe dans la section "Identifiants" afin que Jenkins puisse récupérer le code Git.
- Copiez l'URL de votre projet Git. Voici un exemple que vous pouvez utiliser à des fins de test :

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



► Configuration d'un projet - *Freestyle*



☒ GitHub project

Project url ?

`https://github.com/mhassini/avec-maven`

Avancé ▾

☐ Git ?

Repositories ?

Repository URL ?

`https://github.com/mhassini/avec-maven.git`

Credentials ?

- aucun - ▾

Ajouter ▾

Avancé ▾

Add Repository

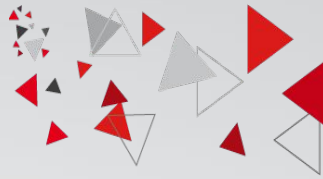
Branches to build ?

Branch Specifier (blank for 'any') ?

`*/master`



► Configuration d'un projet - *Freestyle*



- 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 ?

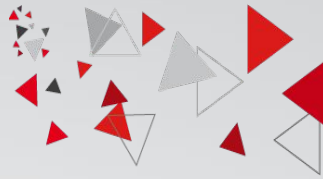
☒ Construire périodiquement ?

Planning ?

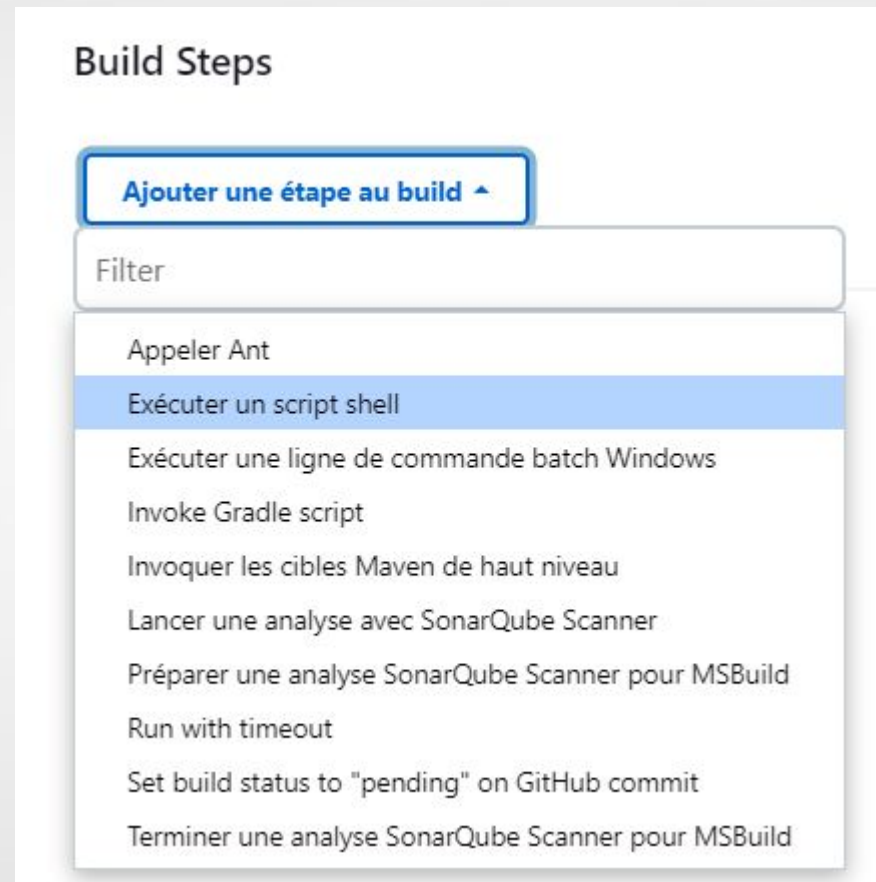
⚠ Vous voulez vraiment dire "chaque minute" avec l'expression "*****"? Peut-être vouliez-vous dire "H*****"?



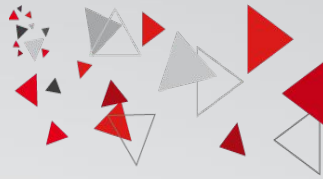
► Configuration d'un projet – *Freestyle*



- Pour les autres tâches (2, 3, 4 et 5), on peut les configurer à travers la partie Build.



► Configuration d'un projet - *Freestyle*



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

Build Steps

☰ Exécuter un script shell ?

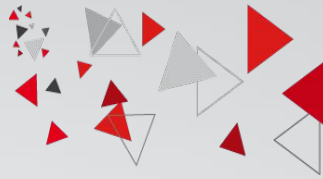
Commande

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

```
echo "c'est mon premier projet freestyle Jenkins !"
date
```



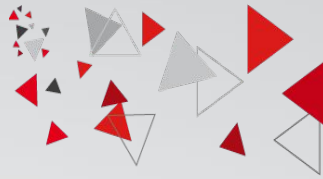
► Configuration d'un projet - *Freestyle*



- **Exemple 2** : Vérifiez l'installation de maven



► Configuration d'un projet – *Freestyle*



- Enregistrez la configuration, puis lancez le build.

Tableau de bord > SpringAOP >

État

Modifications

Répertoire de travail

Lancer un build

Configurer

Supprimer Projet

GitHub

Renommer

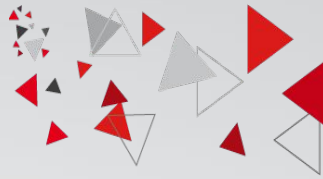
Projet SpringAOP


Liens permanents




- [Dernier build \(#2\), il y a 2 mn 14 s](#)
- [Dernier build stable \(#2\), il y a 2 mn 14 s](#)
- [Dernier build avec succès \(#2\), il y a 2 mn 14 s](#)
- [Last completed build \(#2\), il y a 2 mn 14 s](#)



► Configuration d'un projet - *Freestyle*



- En cas d'échec de build, l'icône d'avertissement  apparaît à côté du nom du projet sur le tableau de bord.

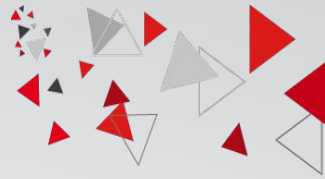
		freestyle1	3 mn 43 s #2	2 mn 13 s #3	0.34 s	
---	---	------------	--------------	--------------	--------	---

- Si la construction est réussie, l'icône  s'affiche.

S	M	Nom du projet ↓	Dernier succès	Dernier échec	Dernière durée	
		SpringAOP	5 mn 52 s #2	s. o.	2.4 s	



► Configuration d'un projet - *Pipeline*



- Le **pipeline** Jenkins est un code qui permet à plusieurs utilisateurs de modifier et d'exécuter un processus de manière collaborative.
- Il résout ce problème en introduisant un nouveau langage basé sur *Groovy* pour configurer les tâches.



- Vous pouvez enregistrer le pipeline dans un fichier (pour la gestion de version) et il peut gérer divers scénarios.

<https://www.jenkins.io/doc/book/pipeline/getting-started/>



Pipeline


► Configuration d'un projet – Pipeline


- Pour illustrer comment Jenkins fonctionne, nous allons utiliser un projet Spring Boot (que vous obtiendrez pendant le cours Git).
- Pour commencer, vous devez créer un "*pipeline*" en tant que type de projet.


Tableau de bord > Tous >


Saisissez un nom

» Champ obligatoire

 **Construire un projet free-style**
Ceci est la fonction principale de Jenkins qui sert à builder (construire) votre projet. Vous pouvez intégrer tous les outils de gestion de version avec tous les systèmes de build. Il est même possible d'utiliser Jenkins pour tout autre chose qu'un build logiciel.

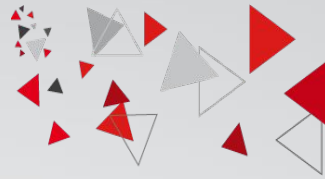
 **Construire un projet maven**
Construit un projet avec maven. Jenkins utilise directement vos fichiers POM et diminue radicalement l'effort de configuration. Cette fonctionnalité est encore en bêta mais elle est disponible afin d'obtenir vos retours.

 **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 multiples, comme des environnements de test multiples, des binaires spécifiques, etc.



► Configuration d'un projet - Pipeline



- **Exemple 1** : Affichez un message “Hello world !” avec Groovy

Pipeline

Definition

Pipeline script ▼

Script ?

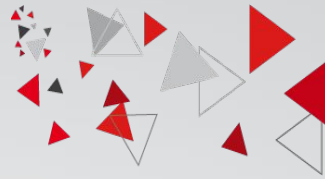
```
1 pipeline {  
2   agent any  
3  
4   stages {  
5     stage('Hello') {  
6       steps {  
7         echo 'Hello World'  
8       }  
9     }  
10  }  
11 }  
12
```

Hello World ▼

Sauver Apply



► Configuration d'un projet - Pipeline



- Exemple 2 : Récupérerez le code à partir de Git

Pipeline

Definition

Pipeline script

Script ?

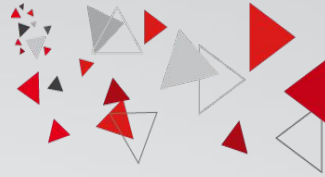
```
1 pipeline {
2   agent any
3
4   stages {
5     stage('Checkout GIT') {
6       steps {
7         echo 'Pulling...'
8         git branch: 'main',
9         url: 'https://github.com/OnsBENSALAH/ski'
10        credentialsId: 'ghp_6hnee2v9zsLui0THILUoK3qKAcdy80yWGgA'
11      }
12    }
13  }
14 }
15
16
```

Si le repository est privé

☒ Use Groovy Sandbox ?



► Configuration d'un projet - *Pipeline*



- **Exemple 3** : Exécutez une commande Maven

Pipeline

Definition

Pipeline script ▼

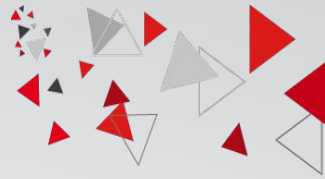
Script ?

```
1 pipeline {
2   agent any
3
4   stages {
5     stage('Testing Maven') {
6       steps {
7         sh "mvn -version"
8       }
9     }
10  }
11 }
12
13
```





☒ Use Groovy Sandbox ?



► Configuration d'un projet – *Pipeline*



- Pour suivre les meilleures pratiques, il est recommandé d'utiliser un fichier appelé "*Jenkinsfile*" stocké dans le référentiel de code (Git) pour définir le pipeline.

	sirineDevOps Create Jenkinsfile	1ef97a6 30 minutes ago	🕒 14 commits
	src/main/java/tn/esprit/esponline	commit from user3	19 days ago
	Jenkinsfile	Create Jenkinsfile	30 minutes ago
	pom.xml	Update pom.xml	2 hours ago



► Configuration d'un projet – Pipeline



- Vous pouvez configurer le pipeline Jenkins en pointant simplement vers le fichier *Jenkinsfile*

Tableau de bord > Station Ski > Configuration

Pipeline

Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

Repository URL ?

<https://github.com/OnsBENSALAH/ski.git>

Credentials ?

- aucun -

Ajouter ▼



► Configuration d'un projet – Pipeline



- **Bonus** : déclenchez un build à distance en accédant à une URL spécifique via votre navigateur.

pipeline-1 >

Configuration

Build Triggers

- ☐ Construire après le build sur d'autres projets ?
- ☐ Construire périodiquement ?
- ☐ 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

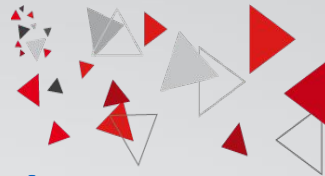
pipeline-1-token

Utilisez l'URL qui suit pour lancer un build à distance : JENKINS_URL/job/pipeline-1/build?token=TOKEN_NAME ou /buildWithParameters?token=TOKEN_NAME

<http://192.168.33.10:8080/job/pipeline-1/build?token=pipeline-1-token>



► Configuration d'un projet - Pipeline



Sur le navigateur : <http://192.168.33.10:8080/job/pipeline-1/build?token=pipeline-1-token>

Sur Jenkins :

Tableau de bord > pipeline-1 >

- ⚙ Configurer
- 🗑 Supprimer Pipeline
- 🔍 Full Stage View
- ✎ Renommer
- ❓ Pipeline Syntax

Recent Changes

Stage View

Average stage times:
(Average full run time: ~1s)

	Git	Maven Build	Maven Test
Average	141ms	118ms	111ms
#13 Sep 18 21:24 No Changes	164ms	141ms	106ms
#12			

Historique des builds tendance ▼

🔍 Filter builds...

✅ #13 18 sept. 2022 20:24




► Travail à faire



- En utilisant Groovy, créez un flux d'intégration continue qui démarre automatiquement dès qu'un push est détecté dans le référentiel Git, en utilisant un fichier Jenkinsfile. Ce flux inclut les étapes suivantes :
 - Récupération du code source depuis le référentiel Git
 - Affichage de la date système
- Assurez-vous de **décocher** la case "Ce qui déclenche le build" dans tous vos projets (freestyle ou pipeline) après vos tests afin d'éviter qu'un travail ne s'exécute automatiquement chaque minute de façon indéfinie.





*"Apprendre par le projet, c'est découvrir
▶ par l'action, créer par la compréhension, et
réussir par la persévérance."*