



# Jenkins

## Installation du docker Jenkins

- Dans un terminal Ubuntu, exécuter les commandes suivantes :

```
docker pull frouland/myjenkins:0.2
docker run -d --name JenkinsCI -e http_proxy='proxy.insa-rouen.fr:3128' -e https_proxy='proxy.insa-rouen.fr:3128' -p 8080:8080 -p 50000:50000 frouland/myjenkins:0.2
```

- Dans un navigateur, ouvrir l'URL : <http://10.0.0.100:8080/>
- Ajout du mot de passe admin pour débloquer Jenkins :

```
docker exec -it JenkinsCI cat /var/jenkins_home/secrets/initialAdminPassword
```

- Mise à jour du proxy dans Jenkins pour les plugins : <http://10.0.0.100:8080/pluginManager/advanced>
- Installation des plugins par défaut
- Aller dans "**Administrer Jenkins > Configuration globale des outils**" pour configurer les outils suivants :

### 1. JDK (décocher Install automatically) :

- Nom : java-1.8-openjdk
- JAVA\_HOME : /usr/lib/jvm/java-1.8-openjdk

JDK

Installations JDK

Ajouter JDK

JDK

Nom

java-1.8-openjdk

JAVA\_HOME

/usr/lib/jvm/java-1.8-openjdk

☐ Install automatically

Supprimer JDK

Ajouter JDK

Liste des installations JDK sur ce système

### 2. Maven (décocher Install automatically) :

- Nom : M3
- MAVEN\_HOME : /usr/share/maven

Maven

Installations Maven

Ajouter Maven

Maven

Nom

M3

MAVEN\_HOME

/usr/share/maven

☐ Install automatically

Supprimer Maven

Ajouter Maven

Liste des installations Maven sur ce système

## Configuration du proxy pour Maven dans le docker JenkinsCI

- Dans un terminal Ubuntu, exécuter les commandes suivantes :

```
docker exec -it JenkinsCI bash
dos2unix /usr/share/maven/conf/settings.xml
```

- Editer le fichier "**/usr/share/maven/conf/settings.xml**" et ajouter les lignes suivantes dans la section **proxies** :

```
<proxy>
  <id>optional</id>
  <active>true</active>
  <protocol>http</protocol>
  <username></username>
  <password></password>
  <host>proxy.insa-rouen.fr</host>
  <port>3128</port>
```

```
<nonProxyHosts>local.net|some.host.com|10.0.0.100</nonProxyHosts>
</proxy>
```

## Création d'un projet free-style

- Créer un projet free-style nommé "**Petclinic**" qui devra :
  1. Récupérer les sources de **spring-framework-petclinic** dans votre GitHub
    - Créer une clé SSH dans le docker JenkinsCI avec la commande :

```
ssh-keygen -t rsa -b 4096 -C "<votre_adresse_mail>"
```

- Ajouter la nouvelle clé SSH "\*\*\*~/.ssh/id\_rsa.pub\*\*\*" dans votre compte GitHub
- Créer un credential de type "\*\*\*SSH Username with private key\*\*\*"
- Renseigner un ID, un username, la "\*\*\*Private Key\*\*\*" et la passphrase (si nécessaire)

### Gestion de code source

☐ Aucune

☒ Git

Repositories

Repository URL:

Credentials:  [Ajouter](#)

Avancé...

Add Repository

Branches to build

Branch Specifier (blank for 'any'):

Add Branch

Navigateur de la base de code: (Auto)

Additional Behaviours: [Ajouter](#)

☐ Mercurial

☐ Subversion

2. Créer une tâche du build avec le type "**Invoquer les cibles Maven de haut niveau**"
3. Sélectionner la version de maven
4. Appel des cibles "**clean package**"
5. Dans "**Avancé...**", ajouter la propriété "**skipTests=true**" pour éviter le lancement des tests

### Build

**Invoquer les cibles Maven de haut niveau**

Version de Maven:

Cibles Maven:

POM:

Propriétés:

Options de la JVM:

☐ Inject build variables

☐ Utiliser un repository Maven privé

Settings file:

Global Settings file:

6. Lancer le build manuellement

**Jenkins**

Jenkins > Petclinic >

- Retour au tableau de bord
- État
- Modifications
- Répertoire de travail
- Lancer un build**
- Supprimer Projet
- Configurer
- Favoris
- Open Blue Ocean
- Rename

7. Consulter le log du build et l'espace de travail

**Jenkins**

Jenkins > Petclinic > #21

Retour au projet

État

Modifications

**Console Output**

View as plain text

Informations de la construction

Delete build '#21'

Git Build Data

No Tags

Open Blue Ocean

Build précédent

### Sortie de la console

```

Started by user Administrateur
Running as SYSTEM
Building in workspace /var/jenkins_home/workspace/Petclinic
using credential github
> git rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/frouland/spring-framework-petclinic # timeout=10
Fetching upstream changes from https://github.com/frouland/spring-framework-petclinic
> git --version # timeout=10
using GIT_SSH to set credentials
> git fetch --tags --force --progress -- https://github.com/frouland/spring-framework-petclinic +re
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision e7f3d4de2f547ebe84122b0d0e598d7847f3c846 (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f e7f3d4de2f547ebe84122b0d0e598d7847f3c846 # timeout=10

```

**Jenkins**

Jenkins > Petclinic >

Retour au tableau de bord

État

Modifications

**Répertoire de travail**

Effacer l'espace de travail

Lancer un build

Supprimer Projet

Configurer

Favoris

Open Blue Ocean

Rename

### Workspace of Petclinic on maître

.git		
.mvn/wrapper		
src		
target		
.editorconfig	12 déc. 2019 16:06:59	192 B <a href="#">vue</a>
.gitignore	12 déc. 2019 16:06:59	87 B <a href="#">vue</a>
.travis.yml	12 déc. 2019 16:06:59	31 B <a href="#">vue</a>
Dockerfile	12 déc. 2019 16:06:59	58 B <a href="#">vue</a>
Jenkinsfile	12 déc. 2019 16:06:59	820 B <a href="#">vue</a>
mvnw	12 déc. 2019 16:06:59	6.95 KB <a href="#">vue</a>
mvnw.cmd	12 déc. 2019 16:06:59	5.06 KB <a href="#">vue</a>
pom.xml	12 déc. 2019 16:06:59	18.31 KB <a href="#">vue</a>
readme.md	12 déc. 2019 16:06:59	7.94 KB <a href="#">vue</a>
sonar-project.properties	12 déc. 2019 16:06:59	332 B <a href="#">vue</a>
tutorial.md	12 déc. 2019 16:06:59	3.38 KB <a href="#">vue</a>

[\(Tous les fichiers dans un zip\)](#)

**Historique des builds** [tendance](#)

find X

#21 12 déc. 2019 16:08

8. Ajouter une nouvelle tâche dans le build pour renommer le fichier **petclinic.war** généré dans le dossier target du workspace.

Le fichier sera renommé de la façon suivante : petclinic-NUM\_BUILD-TIMESTAMP.war

Exemple : petclinic-4-20191210102322.war

Astuce :

DATE\_WITH\_TIME=`date "+%Y%m%d-%H%M%S"`;

NEW\_NAME="petclinic\_\${BUILD\_NUMBER}-\${DATE\_WITH\_TIME}.war";

Exécuter un script shell

Commande

```
DATE WITH TIME=`date "+%Y%m%d-%H%M%S"`;
NEW_NAME="petclinic_${BUILD_NUMBER}-${DATE_WITH_TIME}.war";
mv $WORKSPACE/target/petclinic.war $WORKSPACE/target/$NEW_NAME;
```

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

Avancé...

Ajouter une étape au build

9. Ajouter une action à la suite du build pour archiver l'artefact généré (fichier à archiver : **target/\*.war**)

Actions à la suite du build

Archiver des artefacts

Fichiers à archiver

target/\*.war

Exclusions

☐ Ne pas faire échouer la construction si l'archive retournée est vide

☒ Archiver les artefacts seulement si la construction est un succès

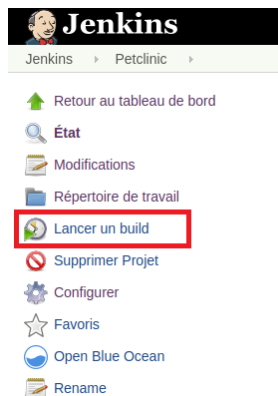
☐ Marquer d'une empreinte numérique tous les artefacts archivés

☒ Utilise les exclusions par défaut

☒ Treat include and exclude patterns as case sensitive

Ajouter une action après le build

10. Lancer le build manuellement



12. (En option) Ajouter une action pour supprimer le workspace une fois le build terminé

Delete workspace when build is done

Patterns for files to be deleted

Ajouter

Apply pattern also on directories

☐

Clean when status is

☒ Success
 ☒ Unstable
 ☒ Failure
 ☒ Not Built
 ☒ Aborted

Don't fail the build if cleanup fails

☐

External Deletion Command

Disable deferred wipeout

☐

Ajouter une action après le build