

# Intégration sur Jenkins et Anayle sur SonarQube

## Configuration de SonarQube :

Après la phase d'installation de SonarQube (décrite dans le fichier « Documentation des tests SOSIE2), on le lance dans le navigateur avec la l'url suivante : « localhost:9000 ».

Une fois authentifié, il suffit de choisir le langage du projet (java) et type de projet (Maven). À ce moment là une commande va être générée qu'on va l'ajouter dans notre script de build du projet afin de faire l'analyse après chaque build.

Dans notre cas, la commande générée est la suivante :

```
mvn sonar:sonar \
-Dsonar.host.url=http://localhost:9000 \
-Dsonar.login=46671055f0e264c924c05fe404aae0eae051d6fa
```

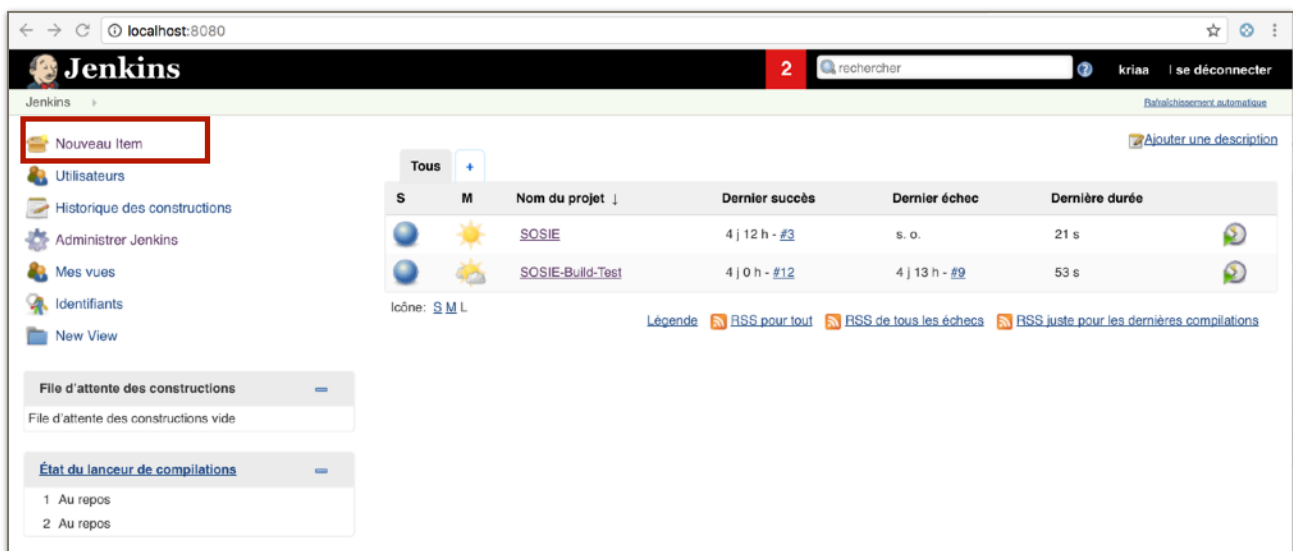
## Configuration de Jenkins :

Après la phase d'installation de Jenkins (décrite dans le fichier « Documentation des tests SOSIE2), on le lance avec la commande « java -jar jenkins.war —httpPort=8080 » .

### Créer un nouveau projet :

Pour créer un nouveau projet, il suffit de :

- cliquer sur le bouton « Nouveau Item » :



- choisir le type « projet free-style » et saisir le nom.

localhost:8080/view/all/newJob

Jenkins

2

rechercher

kriaa | se déconnecter

Jenkins » Tous »

### Saisissez un nom

Projet-SOSIE

» 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.

**Pipeline**  
Orchestrates long-running activities that can span multiple build slaves. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

**Construire un projet multi-configuration**  
Adapté aux projets qui nécessitent un grand nombre de configurations différentes, comme des environnements de test multiples, des binaires spécifiques à une plateforme, etc.

**Folder**  
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

## Configurer le projet :

Ajouter un nom et une description du projet

localhost:8080/view/all/newJob

Jenkins

2

rechercher

kriaa | se déconnecter

Jenkins » Projet-SOSIE »

### Projet-SOSIE

General

Gestion de code source

Ce qui déclenche le build

Environnements de Build

Build

Actions à la suite du build

Nom du Projet: Projet-SOSIE

Description: Guide de création d'un nouveau projet sur Jenkins

[Plain text] [Prévisualisation](#)

☐ Ce build a des paramètres

☐ GitHub project

☐ Supprimer les anciens builds

☐ Throttle builds

☐ Désactiver le projet

☐ Exécuter des builds simultanément si nécessaire

Avancé...

## Choisir l'emplacement du code source :

Dans notre cas, notre code source est dans git.

The screenshot shows the 'Gestion de code source' (Source Code Management) configuration page. It features a sidebar with radio buttons for 'Aucune' (None) and 'Git' (selected). The main area is divided into sections: 'Repositories' with a text field for 'Repository URL' containing 'https://github.com/sirineKr/SOSIE-Spring.git' and a dropdown for 'Credentials' set to '- aucun -', with buttons for 'Ajouter' (Add), 'Avancé...' (Advanced), and 'Add Repository'; 'Branches to build' with a 'Branch Specifier (blank for 'any')' field containing '\*/master' and an 'Add Branch' button; 'Navigateur de la base de code' (Code Base Navigator) with a dropdown set to '(Auto)'; and 'Additional Behaviours' with an 'Ajouter' (Add) button. At the bottom, there is a radio button for 'Subversion'.

## Mettre les commande nécessaire pour lancer le build du projet et faire l'analyse sur SonarQube :

Dans cette partie, nous avons mis deux commandes : la première pour faire le build du projet et la deuxième (commande générée à partir du SonarQube) pour faire l'analyse à la fin du build.

The screenshot shows the 'Build' configuration page. It has a title 'Build' and a section 'Exécuter un script shell' (Execute shell script). Below this, there is a 'Commande' (Command) text area containing the following commands:

```
mvn clean install
mvn sonar:sonar \
-Dsonar.host.url=http://localhost:9000 \
-Dsonar.login=46671055f0e264c924c05fe404aae0eae051d6fa
```

Below the command area, there is a link 'Voir la liste des variables d'environnement disponibles' (View the list of available environment variables) and an 'Avancé...' (Advanced) button. At the bottom, there is a button 'Ajouter une étape au build' (Add step to build).

## Choisir les actions à faire après le build :

Dans notre cas, nous voulons publier le rapport des résultats des tests JUnit :

### Actions à la suite du build

Publier le rapport des résultats des tests JUnit

XML des rapports de test

target/surefire-reports/\*.xml

Une configuration du type [Fileset 'includes'](#) qui indique où se trouvent les fichiers XML des rapports de test, par exemple 'myproject/target/test-reports/\*.xml'. Le répertoire de base (basedir) du fileset est [la racine du workspace](#).

☐ Retain long standard output/error

Health report amplification factor

1,0

1% failing tests scores as 99% health. 5% failing tests scores as 95% health

Allow empty results

☐ Do not fail the build on empty test results

Ajouter une action après le build ▾

Sauver

Apply

Une fois terminé, on lance le build du projet (il faut que sonarQube est lancé sur le port 9000) :

Jenkins

2

rechercher

kriaa

se déconnecter

Jenkins

Nouveau Item

Utilisateurs

Historique des constructions

Administrer Jenkins

Mes vues

Identifiants

New View

File d'attente des constructions

File d'attente des constructions vide

État du lanceur de compilations

1 Au repos

2 [Projet-SOSIE](#) #1

Tous +

S	M	Nom du projet ↓	Dernier succès	Dernier échec	Dernière durée	
		<a href="#">Projet-SOSIE</a>	s. o.	s. o.	ND	
		<a href="#">SOSIE</a>	4 j 18 h - #3	s. o.	21 s	
		<a href="#">SOSIE-Build-Test</a>	4 j 6 h - #12	4 j 19 h - #9	53 s	

Icône: S M L

Légende RSS pour tout RSS de tous les échecs RSS juste pour les dernières compilations

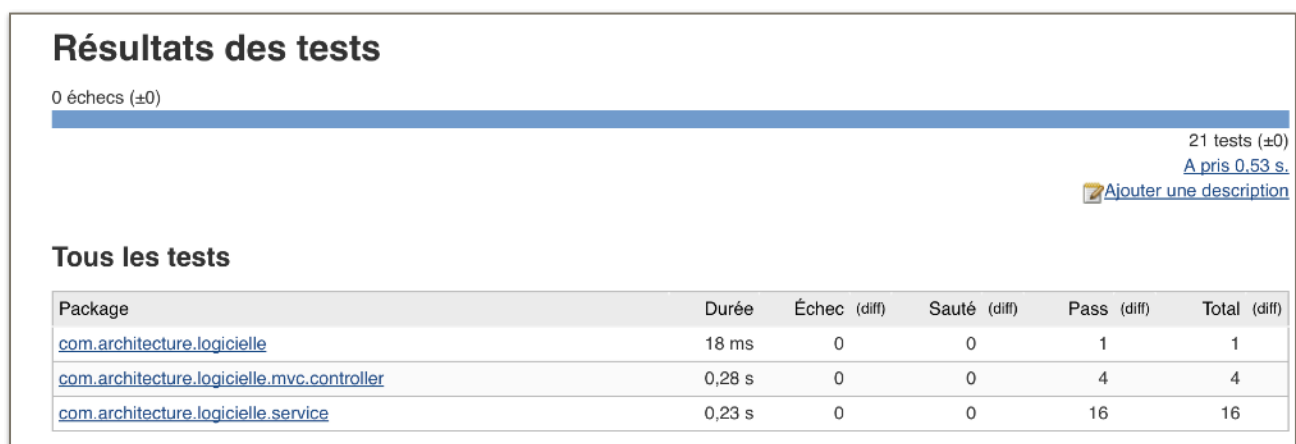
Rafraîchissement automatique

Ajouter une description

Le résultat des tests est la suivante :



Lorsque nous cliquons sur « Derniers résultats des tests » :



Dans console output du projet, on trouve lien pour consulter l'analyse sur SonarQube :

```
[INFO] ANALYSIS SUCCESSFUL, you can browse http://localhost:9000/dashboard/index/org.springframework.gs-mysql-data
[INFO] Note that you will be able to access the updated dashboard once the server has processed the submitted
analysis report
[INFO] More about the report processing at http://localhost:9000/api/ce/task?id=AWB-lBTKtlbHD5p9bbRM
[INFO] Task total time: 10.202 s
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 13.302 s
[INFO] Finished at: 2017-12-22T15:15:22+01:00
[INFO] Final Memory: 41M/539M
[INFO] -----
Enregistrement des résultats des tests
Finished: SUCCESS
```

Sur SonarQube :

