**Intégration de SonarQube avec Maven**

SonarQube est une plateforme open-source pour l'inspection continue de la qualité du code. Il permet de détecter les bugs, les vulnérabilités et les mauvaises pratiques dans le code source. Voici un exercice pour intégrer SonarQube avec un projet Maven.

**Objectifs**

* Installer SonarQube et SonarQube Scanner.
* Configurer un projet Maven pour l'analyse de la qualité du code avec SonarQube.
* Exécuter une analyse de code avec SonarQube.

**Prérequis**

* Java Development Kit (JDK) 11 ou supérieur installé.
* Maven installé.
* Docker installé (optionnel, pour une installation facile de SonarQube).

**Étapes de l'exercice**

**1. Installation de SonarQube**

1. **Installer SonarQube avec Docker**
   * Téléchargez et exécutez l'image Docker de SonarQube :

docker run -d --name sonarqube -p 9000:9000 sonarqube

* + Attendez quelques minutes que SonarQube se lance. Vous pouvez vérifier le statut du conteneur avec :

docker ps -a

1. **Accéder à l'interface web de SonarQube**
   * Ouvrez un navigateur et accédez à http://localhost:9000.
   * Utilisez les identifiants par défaut pour vous connecter (login: admin, mot de passe: admin).

**2. Configuration de SonarQube**

1. **Créer un projet SonarQube**
   * Dans l'interface web de SonarQube, allez dans **Projects > Create new project**.
   * Donnez un nom au projet et suivez les instructions pour générer un token.
2. **Configurer Maven pour SonarQube**
   * Ouvrez le fichier pom.xml de votre projet Maven et ajoutez le plugin SonarQube :

<build>

<plugins>

<plugin>

<groupId>org.sonarsource.scanner.maven</groupId>

<artifactId>sonar-maven-plugin</artifactId>

<version>3.9.1.2184</version>

</plugin>

</plugins>

</build>

1. **Ajouter les informations du serveur SonarQube dans settings.xml de Maven**
   * Ouvrez ou créez le fichier settings.xml dans le répertoire .m2 de votre utilisateur (~/.m2/settings.xml sous Linux/macOS ou C:\Users\<YourUser>\.m2\settings.xml sous Windows) et ajoutez :

<servers>

<server>

<id>sonar</id>

<username>votre\_token\_sonar</username>

<password></password>

</server>

</servers>

**3. Exécution de l'analyse SonarQube**

1. **Analyser le projet avec SonarQube**
   * Exécutez la commande suivante dans le répertoire de votre projet Maven :

mvn clean verify sonar:sonar -Dsonar.projectKey=nom\_du\_projet -Dsonar.host.url=http://localhost:9000 -Dsonar.login=votre\_token\_sonar

1. **Vérifier les résultats**
   * Allez dans l'interface web de SonarQube et sélectionnez votre projet pour voir les résultats de l'analyse.

**Conclusion**

En suivant ces étapes, vous avez :

* Installé et configuré SonarQube.
* Configuré un projet Maven pour l'analyse de qualité de code avec SonarQube.
* Exécuté une analyse de code et visualisé les résultats dans l'interface web de SonarQube.

SonarQube est un outil puissant pour améliorer la qualité de votre code en identifiant les problèmes potentiels et en vous aidant à les corriger de manière proactive. Pour des configurations plus avancées et des intégrations continues, consultez la documentation officielle de SonarQube : SonarQube Documentation.

**Intégration de SonarQube avec Jenkins**

L'intégration de SonarQube avec Jenkins permet d'automatiser l'analyse de la qualité du code dans votre pipeline CI/CD. Voici un exercice détaillé pour configurer cette intégration.

**Prérequis**

* Jenkins installé et en cours d'exécution.
* SonarQube installé et en cours d'exécution.
* Projet Maven configuré pour SonarQube.
* Java Development Kit (JDK) 11 ou supérieur installé.
* Maven installé.

**Étapes de l'exercice**

**1. Installer et configurer SonarQube**

1. **Installer SonarQube avec Docker**

docker run -d --name sonarqube -p 9000:9000 sonarqube

1. **Accéder à l'interface web de SonarQube**
   * Ouvrez un navigateur et accédez à http://localhost:9000.
   * Connectez-vous avec les identifiants par défaut (login: admin, mot de passe: admin).
2. **Créer un projet SonarQube**
   * Dans l'interface web de SonarQube, allez dans **Projects > Create new project**.
   * Donnez un nom au projet et générez un token.

**2. Configurer Jenkins**

1. **Installer les plugins nécessaires dans Jenkins**
   * Allez dans **Manage Jenkins > Manage Plugins > Available**.
   * Recherchez et installez les plugins suivants :
     + SonarQube Scanner
     + Maven Integration
     + Pipeline
2. **Configurer SonarQube dans Jenkins**
   * Allez dans **Manage Jenkins > Configure System**.
   * Faites défiler jusqu'à la section **SonarQube Servers**.
   * Cliquez sur **Add SonarQube** et configurez l'URL du serveur SonarQube (ex: http://localhost:9000) et le token généré.
3. **Configurer Maven dans Jenkins**
   * Allez dans **Manage Jenkins > Global Tool Configuration**.
   * Configurez Maven en ajoutant une installation Maven (vous pouvez indiquer la version ou le chemin d'installation de Maven).

**3. Configurer un Job Jenkins pour analyser avec SonarQube**

1. **Créer un nouveau job de type Pipeline**
   * Allez dans **New Item**, donnez un nom au job et sélectionnez **Pipeline**.
2. **Définir le pipeline dans Jenkins**
   * Dans la section **Pipeline**, sélectionnez **Pipeline script** et ajoutez le script suivant :

pipeline {

agent any

tools {

maven 'Maven' // Assurez-vous que le nom correspond à l'installation Maven configurée dans Jenkins

}

environment {

scannerHome = tool 'SonarQube Scanner' // Assurez-vous que le nom correspond à l'installation du scanner SonarQube configurée dans Jenkins

}

stages {

stage('Clone Repository') {

steps {

git 'https://github.com/votre-utilisateur/mon-projet.git'

}

}

stage('Build') {

steps {

sh 'mvn clean install'

}

}

stage('SonarQube analysis') {

steps {

withSonarQubeEnv('SonarQube') {

sh 'mvn sonar:sonar -Dsonar.projectKey=mon-projet'

}

}

}

stage('Quality Gate') {

steps {

script {

timeout(time: 1, unit: 'HOURS') {

waitForQualityGate abortPipeline: true

}

}

}

}

}

}

**4. Exécution du job Jenkins**

1. **Lancer le job**
   * Cliquez sur **Build Now** pour lancer le job.
2. **Vérifier les résultats**
   * Suivez la progression du job dans la console de Jenkins.
   * Une fois le job terminé, accédez à l'interface web de SonarQube (http://localhost:9000) et consultez les résultats de l'analyse de votre projet.

**Conclusion**

En suivant ces étapes, vous avez configuré Jenkins pour automatiser l'analyse de la qualité du code avec SonarQube. Vous avez intégré SonarQube dans un pipeline Jenkins, ce qui permet de surveiller continuellement la qualité du code de votre projet et de réagir rapidement aux problèmes potentiels.

Pour des configurations plus avancées et des intégrations continues, vous pouvez consulter la documentation officielle de Jenkins et SonarQube :

* Documentation Jenkins
* Documentation SonarQube