TP / Tutoriel

Outils d'analyse statique de code

L'objectif est de se familiariser avec différents outils d'analyse statique de code. Nous allons en tester deux : PMD et SonarQube. Ce genre d'outils permet d'identifier des problèmes au niveau du style de codage, de détecter des variables non utilisées, des bouts de code mort, des instanciations d'objets inutiles, etc.

PMD

PMD (https://pmd.github.io/) est un outil qui s'installe en local. Il permet de vérifier du code source, selon un ensemble de règles personnalisables, pour les fautes de style dans la programmation par exemple.

L'outil est téléchargeable en ligne, et intégrable sous forme de plugin à plusieurs environnements de développement intégrés.

Installation et configuration sous IntelliJ

- Ouvrez les préférences, et allez dans l'onglet Plugins.
- Installez les plugins QAPlug et QAPlug-PMD
- Allez ensuite dans les préférences > QAPlug > Coding Rules
- Explorez les règles.
- Pour l'exemple, nous allons activer toutes les règles contenant le mot « naming »

Analyse de code

Pour lancer l'analyse avec PMD, clic droit sur un projet ou un fichier > Analyse > Analyse du code

Exemple

- Téléchargez le code source de l'application Monopoly et lancez l'analyse PMD.
- Explorez les résultats. Essayez en activant d'autres règles.

SonarQube

SonarQube va plus loin dans l'analyse du code, et permet d'obtenir des rapports plus complets et des mesures sur la qualité de celui-ci, sa vulnérabilité, le taux de couverture du code par les tests, etc.

C'est un outil distribué, généralement utilisé dans les processus d'intégration continue. Par souci de simplification, nous allons installer un serveur local de démonstration.

Installation et configuration du serveur

- Télécharger la version « Community Edition » (https://www.sonarqube.org)
- Dézipper l'archive, aller dans le dossier, et exécuter la commande :

```
$ bin/[OS]/sonar.sh console
```

Le serveur SonarQube est lancé ! Vous pouvez y accéder à l'adresse http://localhost:9000 (connexion : admin/admin).

Le serveur centralise tous les rapports d'analyse qui lui sont envoyés. Maintenant, il faut installer l'outil local qui va scanner le code et envoyer au serveur ces rapports.

Installation du scanner

Il existe des scanners dédiés à plusieurs méthodes de build (maven, gradle, etc). Pour l'exemple, nous allons utiliser un scanner « basique », qui fonctionne en ligne de commande. Comme son nom l'indique, cet outil va scanner les fichiers et envoyer les résultats au serveur SonarQube.

- Téléchargez et installez le scanner en suivant la procédure décrite : https://docs.sonarqube.org/latest/analysis/scan/sonarscanner/
- Dans le dossier du projet à scanner, créez un fichier de configuration `sonar-project.properties`

```
# must be unique in a given SonarQube instance
sonar.projectKey=monopoly

# defaults to project key
sonar.projectName=Monopoly

# Path is relative to the sonar-project.properties file. Defaults to .
sonar.sources=src
sonar.java.binaries=out/production
sonar.exclusions=src/**/*Test.java


# Encoding of the source code. Default is default system encoding
sonar.sourceEncoding=UTF-8
```

Lancement de l'analyse

Dans le répertoire du projet, lancez la commande suivante :

```
$ sonar-scanner
```

Quand l'analyse est terminée, vous pouvez aller voir le résultat sur le site du serveur.