

TP8 Maven2 : Génération de rapports

Table des matières

Objectifs.....	3
Liens utiles.....	3
Partie 1 : Génération de rapports	3
Générer la javadoc	3
Valider la qualité du code avec le plugin checkstyle.....	4
Rapport croisé de source.....	4
Couverture de test.....	5
Identifier patterns d'erreur avec PMD.....	5
Connaître l'activité du projet.....	6
Intégration, avec l'outil Sonar.....	7

Objectifs

- Générer des rapports maven

Liens utiles

- Site de Maven : <http://maven.apache.org/>
- Plugin Checkstyle : <http://maven.apache.org/plugins/maven-checkstyle-plugin/>
- FAQ MAVEN developpez.com : <http://java.developpez.com/faq/maven/>

Partie 1 : Génération de rapports

Générer la javadoc

- Ajoutez des commentaires à votre code projet Web (servlet)
- Ajouter le code suivant dans le pom.xml du client

```
<reporting>
...
<plugins>

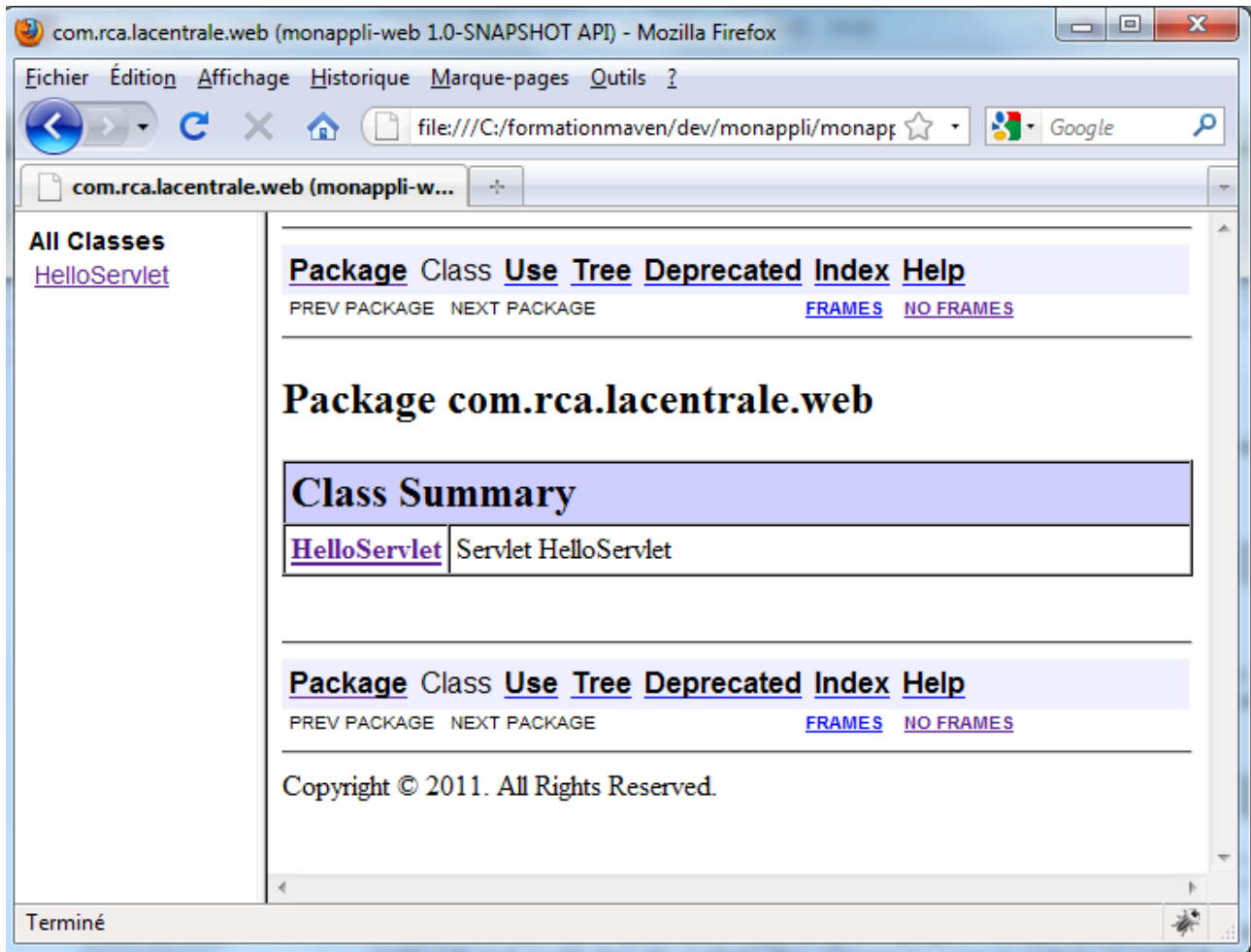
  <plugin>

    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-javadoc-plugin</artifactId>

  </plugin>

</plugins>
...
</reporting>
```

Puis lancez : **mvn site**



Valider la qualité du code avec le plugin checkstyle

- Ajoutez à la section <reporting> du projet client le plugin checkstyle

```
<project>
...
<reporting>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-checkstyle-plugin</artifactId>
    </plugin>
  </plugins>
</reporting>
...
</project>
```

- Lancez : **mvn site**
- Observez la section 'reporting' ajoutée dans le rapport récupéré dans \target\site\index.html
- Quelle est la norme de codage à laquelle se réfère le rapport par défaut ?
- Comment imposer la norme de codage d'IBM ?
- Modifiez la couche web de façon à diminuer le nb d'erreur ?

Liens utiles :

Site de de l'outil CheckStyle : <http://checkstyle.sourceforge.net/>

Site du plugin Maven : <http://maven.apache.org/plugins/maven-checkstyle-plugin/>

HelloServlet.java		
Violation	Message	Line
✗	File does not end with a newline.	0
✗	Missing package-info.java file.	0
✗	Using the '.' form of import should be avoided - java.io.*.	2
✗	Using the '.' form of import should be avoided - javax.servlet.*.	3
✗	Using the '.' form of import should be avoided - javax.servlet.http.*.	4
✗	First sentence should end with a period.	5
✗	Line is longer than 80 characters.	8
✗	Line is longer than 80 characters.	9
✗	Method 'doGet' is not designed for extension - needs to be abstract, final or empty.	9
✗	Missing a Javadoc comment.	9
✗	Parameter request should be final.	9
✗	Parameter response should be final.	9

Rapport croisé de source

```
<reporting>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-jxr-plugin</artifactId>
    </plugin>
  </plugins>
</reporting>
```

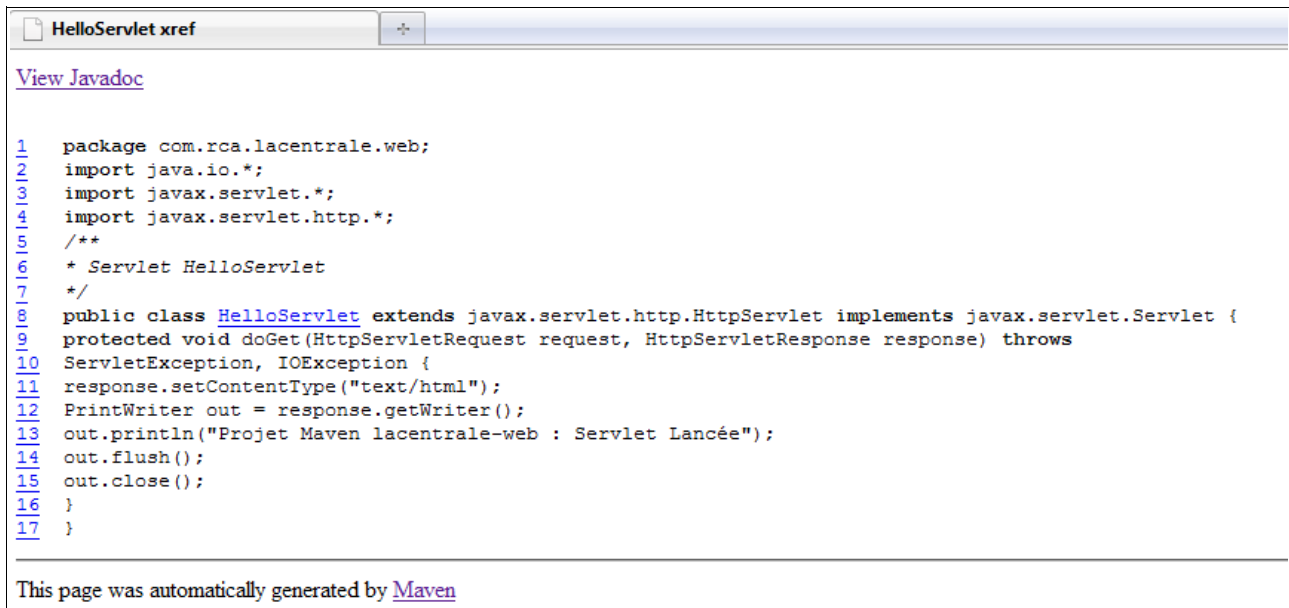
Lien utile : <http://maven.apache.org/plugins/maven-jxr-plugin/>

Quelle est la valeur ajoutée de ce plugin ? En particulier, montrez sa complémentarité avec CheckStyle

HelloServlet.java

Violation	Message	Line
✖	File does not end with a newline.	0
✖	Missing package-info.java file.	0
✖	Using the '*' form of import should be avoided - java.io.*.	2
✖	Using the '*' form of import should be avoided - javax.servlet.*.	3
✖	Using the '*' form of import should be avoided - javax.servlet.http.*.	4
✖	First sentence should end with a period.	5
✖	Line is longer than 80 characters.	8
✖	Line is longer than 80 characters.	9
✖	Method 'doGet' is not designed for extension - needs to be abstract, final or empty.	9
✖	Missing a Javadoc comment.	9
✖	Parameter request should be final.	9
✖	Parameter response should be final.	9

Désormais vous pouvez passer du rapport CheckStyle au code source en cliquant sur le numéro de ligne associé au commentaire CheckStyle.



```
1 package com.rca.lacentrale.web;
2 import java.io.*;
3 import javax.servlet.*;
4 import javax.servlet.http.*;
5 /**
6  * Servlet HelloServlet
7  */
8 public class HelloServlet extends javax.servlet.http.HttpServlet implements javax.servlet.Servlet {
9     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
10     ServletException, IOException {
11         response.setContentType("text/html");
12         PrintWriter out = response.getWriter();
13         out.println("Projet Maven lacentrale-web : Servlet Lancée");
14         out.flush();
15         out.close();
16     }
17 }
```

This page was automatically generated by [Maven](#)

Couverture de test

A quel point les développeurs ont réalisé des tests unitaires ?

Quels parties de l'application n'a pas été testée

```
<reporting>
  <plugins>
    ...
    <plugin>
      <groupId>org.codehaus.mojo</groupId>
      <artifactId>cobertura-maven-plugin</artifactId>
    </plugin>
  </plugins>
</reporting>
```

Lien utile : <http://mojo.codehaus.org/cobertura-maven-plugin/usage.html>

Identifier patterns d'erreur avec PMD

Identifiez :


- Code mort (Ex : variables ou paramètres non utilisés)
- Duplication de code (Code copié/collé = possible bug copié/collé)
- Code 'compliqué' (Ex : trop de if...else)

- Ajoutez à la section <reporting> du projet persist le plugin PMD

```
<project>
...
<reporting>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-pmd-plugin</artifactId>
      <version>2.5</version>
    </plugin>
  </plugins>
</reporting>
...
</project>
```

Quels sont les deux nouveaux rapports générés ?

Generated Reports

This document provides an overview of the various reports that are automatically generated by [Maven](#) . Each report is briefly described below.

Overview

Document	Description
Checkstyle	Report on coding style conventions.
Cobertura Test Coverage	Cobertura Test Coverage Report.
CPD Report	Duplicate code detection.
JavaDocs	JavaDoc API documentation.
PMD Report	Verification of coding rules.
Source Xref	HTML based, cross-reference version of Java source code.

© 2011

Qu'est ce que le rapport 'CPD Report' ?

Qu'est ce que le rapport 'PMD Report' ?

Connaître l'activité du projet

Quels et combien de fichiers modifiés par un développeur ?

- Ajoutez à la section <reporting> du projet persist le plugin changelog

```
<project>
...
<reporting>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-changelog-plugin</artifactId>
    </plugin>
  </plugins>
</reporting>
...
<scm>
  <connection>scm:svn:svn://IPSERVEUR/repository1/monappli-web</connection>
  <url>http://IPSERVEUR/svn/monappli-web</url>
</scm>
...
</project>
```

- Lancez : mvn site
 - Que s'est t'il passé ?

Le répertoire */target/site* situé dans votre projet contient maintenant trois rapports d'activité :

- **changelog** : rapport indiquant toutes les activités sur le SCM.
- **dev-activity** : rapport indiquant par développeur le nombre de commits, de fichiers modifiés.
- **file-activity** : rapport indiquant les fichiers qui ont été révisés.

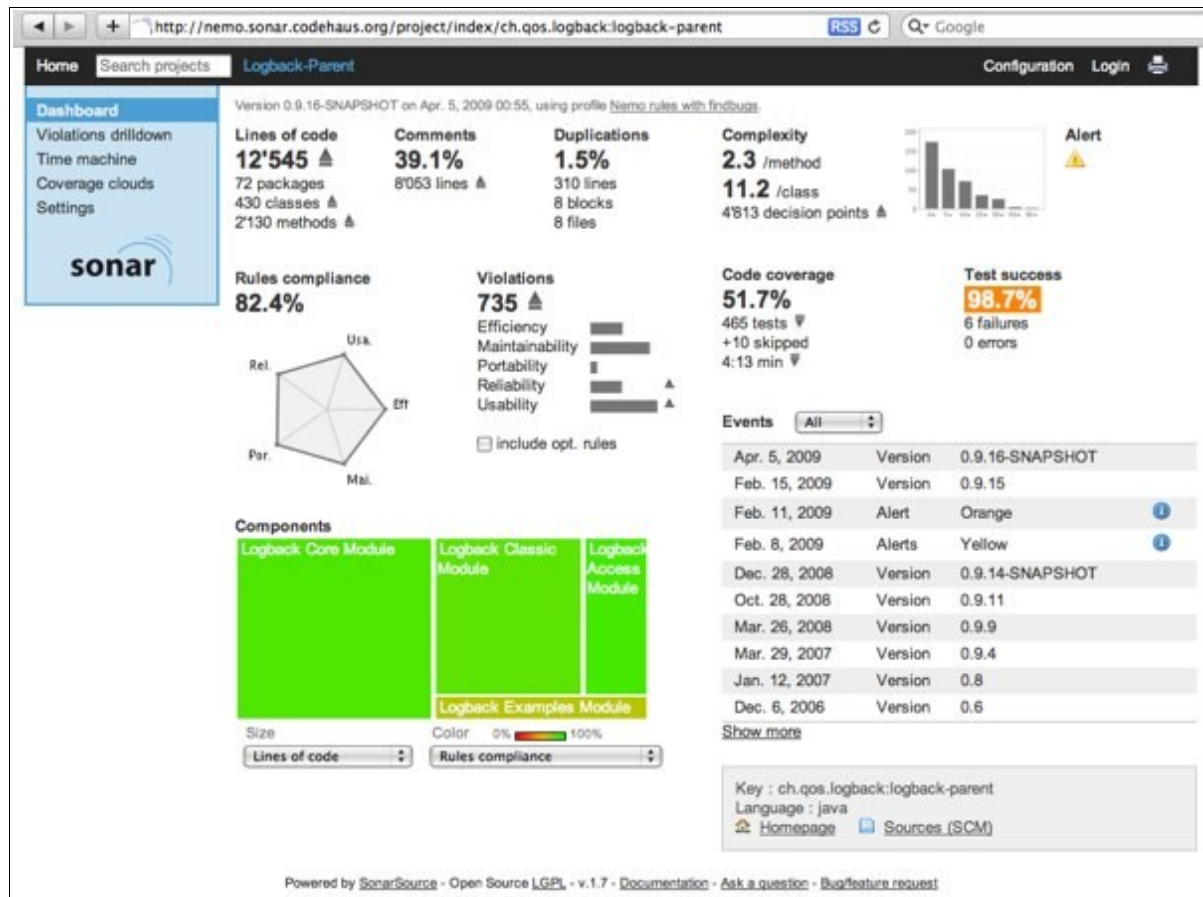
Intégration, avec l'outil Sonar

Téléchargez Sonar : <http://sonar.codehaus.org/downloads>

Installez Sonar : <http://sonar.codehaus.org/documentation/>

Lancez : mvn sonar:sonar

Formation MAVEN - TP 'Génération de site et création de rapports'



FIN