Sonar XSL

A Schematron-based SonarQube plugin for XSL code quality measurement.
Code Quality
What is « Good Code » ?

• **Code should be easily maintainable**
  • Clear architecture
  • human-readable code

• **Code should be reliable**
  • Wich means that is should be tested

• **Code should not expose vulnerabilities**
What is « Good Code » ?

• **Code should be documented**
  
  • To be easily understood: for maintainance, debugging, enhancement…
  
  • For external usage (Public API, Library...)
Some Coding Standards exists for many languages

- **Example, for Java:**
  - Sun Code Conventions
  - Google Java Style
  - Etc.

- **For JS:**
  - StandardJS
  - EsLint / JsLint
  - Etc.
Some Coding Standards for many languages

- **Example, for Java**:  
  - Sun Code Conventions  
  - Google Java Style  
  - Etc.

- **For JS**:  
  - StandardJS  
  - EsLint / JsLint  
  - Etc.

- They are designed to be checked programmatically.
The XSLQuality StyleSheet

In 2009, Mukul Gandhi released his XSL Quality XSLT.

A tool written in XSLT 2.0 that checks the conformance of an XSL Stylesheet to a set of 25 rules that he defined.
• The XSLT-Quality Schematron

Late 2017 Matthieu Ricaud Dussarget released a Schematron based on this work.
SonarQube
SonarQube

• **A code quality measurement tool**
  • That allows to quantify, with metrics, the code quality of an application.

• **SonarQube analyses the source code**
  • And checks the respect of some best-practice rules
  • These rules are defined in some Plugins
  • SonarQube embeds a dozen plugins
    − For the main programming languages (Java, Python, C#, Js...)
    − These plugins are maintained by an open-source community
  • An API allows to extend SonarQube by creating plugins
    − To add new rules
    − To handle some new programming languages
    − This API is available in Java (SonarQube is written in Java)
• **Rules...**

• ...of different types
  - Code smell – Coding style / Best-practices
  - Bug – Potential source of failure
  - Vulnerability – Security concerns

• ...of different severity level
  - Info
  - Minor
  - Major
  - Blocker
  - Critical

• ...on different issues
  - Coding Style
  - Test coverage
  - Etc.
• When a rule is broken in the code
  
  • SonarQube raises it as an Issue
    - Traceability of fix actions
  
  • This mechanism can be integrated to the project workflow tools (Jira, Trello, Slack, Mantis, Github Issue...)
• The Quality Gate:

  • Requirements definition for a project
    - Minimal/maximal values admitted for the metrics related to the project
  • Each analysis of the project ends with a « verdict »:
    *Success*, *Warning* or *Failed*
  • Key actions should depend on this result:
    Examples:
    - No delivery if the analysis ends to *failed*
    - Improvements planning
    - Etc.
• Exemple : the default « Sonar Way » Quality Gate

Sonar way | Built-in

Conditions
Only project measures are checked against thresholds. Sub-projects, directories and files are ignored.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Over Leak Period</th>
<th>Operator</th>
<th>Warning</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coverage on New Code</td>
<td>Always</td>
<td>is less than</td>
<td></td>
<td>80.0%</td>
</tr>
<tr>
<td>Duplicated Lines on New Code (%)</td>
<td>Always</td>
<td>is greater than</td>
<td></td>
<td>3.0%</td>
</tr>
<tr>
<td>Maintainability Rating on New Code</td>
<td>Always</td>
<td>is worse than</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>Reliability Rating on New Code</td>
<td>Always</td>
<td>is worse than</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>Security Rating on New Code</td>
<td>Always</td>
<td>is worse than</td>
<td>A</td>
<td></td>
</tr>
</tbody>
</table>
The Sonar XSL Plugin
The Sonar XSL Plugin

Module

Schematron-Sonar
- Main reactor: drives Schematron validation in the context of a SonarQube plugin
- Create Sonar Rules
  - Based on Schematron asserts/reports
- Validate the XML sources
  - Against the Schematron: raises up a Sonar issue when a Schematron report/assert is triggered

This module could be used on any XML-Based Language

XSLT-Quality
- Generic XSL Coding rules

Custom Rules
- Enterprise/Project specific rules
  - (as much Schematron packages as needed)

Sonar XSLT Plugin
- Definition of the language and his properties
- (targeted file suffixes, etc.)

(eventually embedding project/enterprise-specific custom rules)
The Sonar XSL Plugin

- Finalise the code
- Deployment and use in real-world
  - Experimental usage at *Editions Lefebvre Sarrut*
- Documentation and communication
Conclusion

• **SonarQube is not a « punitive » tool**

  • It allows developpers (and not only them) to have more visibility on the quality of what they do
  • It is a documentation repository
Any Questions?