

Softwarequalitätssicherung

Projektvorstellung

Dennis Jongebloed, Julian Hinxlage, Johannes Theiner

Fachbereich Technik
Abteilung Elektrotechnik und Informatik

23. April 2020

Übersicht

Vorstellung

Warum gerade PMD ?

Qualitätssicherung

2020-04-23

↳ Übersicht

Übersicht

Vorstellung

Warum gerade PMD ?

Qualitätssicherung

PMD



Tool zur statischen Analyse von
Java, JavaScript, Salesforce.com,
Apex, Visualforce, PLSQL,
Apache Velocity, XML und XSL
Anwendungen.

2020-04-23

└ Vorstellung
└ Was ist PMD ?
└ PMD



PMD

Enthält zusätzlich einen CPD(copy-paste-detector) mit Unterstützung von Java, C, C++, C#, Groovy, PHP, Ruby, Fortran, JavaScript, PLSQL, Apache Velocity, Scala, Objective C, Matlab, Python, Go, Switch, Salesforce.com, Apex und Visualforce.

2020-04-23

└ Vorstellung
└ Was ist PMD ?
└ PMD

Warum PMD ?

Die Dokumentation von PMD ist sehr übersichtlich gestaltet, die Beschreibungen sind umfangreich. Der Code besitzt eine gute Codierungskonvention und die Kommentare sind oft hilfreich.

2020-04-23

- └ Warum gerade PMD ?
 - └ Warum PMD ?
 - └ Warum PMD ?

- ▶ dieser gegen den master Branch erstellt werden
- ▶ ohne Fehler gebaut werden können

Richtlinien

Um einen erfolgreichen Pullrequest zu erstellen muss:

- ▶ dieser gegen den master Branch erstellt werden
- ▶ ohne Fehler gebaut werden können



Travis CI



Travis CI

PMD nutzt das Tool Travis CI für die Continuous Integration. Dabei wird der Code automatisch kompiliert, wenn ein neuer Commit ins Repository gepusht wird.

Der Compiler Vorgang wird auf einem Server ausgeführt. In einer Übersicht wird angezeigt, ob ein Build erfolgreich war. Der CI Server führt auch andere nachfolgende Qualitätssicherungs-Tools aus.

Code Style



Um einen einheitlichen Stil
beizubehalten wird Checkstyle
verwendet.

Bewertet werden: Klammersetzung, Importreihenfolge, Leere Zeilen,
etc.

2020-04-23

└─ Qualitätssicherung
└─ Code Style
└─ Code Style

Code Style Beispiel

```
/**  
 * Adapter for the JavaParser, using the specified grammar version.  
 *  
 * @author Pieter_Van_Raemdonck - Application Engineers NV/SA - www.ae.be  
 * @author Andreas Dangel  
 *  
 * @deprecated This is internal API, use {@link LanguageVersionHandler#getParser(ParserOptions)}.  
 */  
@InternalApi  
@Deprecated  
public class JavaLanguageParser extends AbstractJavaParser {  
    private final int jdkVersion;  
    private final boolean preview;  
  
    public JavaLanguageParser(int jdkVersion, ParserOptions parserOptions) {  
        this(jdkVersion, false, parserOptions);  
    }  
  
    public JavaLanguageParser(int jdkVersion, boolean preview, ParserOptions parserOptions) {  
        super(parserOptions);  
        this.jdkVersion = jdkVersion;  
        this.preview = preview;  
    }  
}
```

2020-04-23

└─ Qualitätssicherung
 └─ Code Style
 └─ Code Style Beispiel

Code Style Beispiel

```
/**  
 * Adapter for the JavaParser, using the specified grammar version.  
 * @author Pieter_Van_Raemdonck - Application Engineers NV/SA - www.ae.be  
 * @author Andreas Dangel  
 *  
 * @deprecated This is internal API, use {@link LanguageVersionHandler#getParser(ParserOptions)}.  
 */  
@InternalApi  
@Deprecated  
public class JavaLanguageParser extends AbstractJavaParser {  
    private final int jdkVersion;  
    private final boolean preview;  
  
    public JavaLanguageParser(int jdkVersion, ParserOptions parserOptions) {  
        this(jdkVersion, false, parserOptions);  
    }  
  
    public JavaLanguageParser(int jdkVersion, boolean preview, ParserOptions parserOptions) {  
        super(parserOptions);  
        this.jdkVersion = jdkVersion;  
        this.preview = preview;  
    }  
}
```

Unit Tests

Um den Code zu testen werden Unit Tests angelegt. Units Tests testen die Funktionalität von Teilen des Sourcecodes. Dabei werden in einer Testfunktion die zu testende Funktionen aufgerufen und die Ergebnisse überprüft. Auch ganze Klassen und zusammenhängende Funktionsaufrufe können getestet werden.

2020-04-23

└ Qualitätssicherung
└ Unit Tests
└ Unit Tests

Unit Test Beispiel

```
@Test
public void testUnicodeSupport() {
    String code = "\\u0000" + "#include <iostream>\n" + "#include <string>\n" + "\n" + "// example\n" + "\n"
+ "int main()\n" + "{\n" + "    std::string text(\"ó\");\n" + "    std::cout << text;\n"
+ "    return 0;\n" + "}\n";
    Tokens tokens = parse(code);
    assertNotSame(TokenEntry.getEOF(), tokens.getTokens().get(0));
    assertEquals(24, tokens.size());
}
```

Dieser Unit Test überprüft beispielsweise ob der C++ Parser mit Unicode Zeichen zurechtkommt. Dazu wird C++ Code in Form eines Strings angelegt. Der Code enthält auch Unicode Zeichen. Dann wird überprüft ob der Parser den Code richtig geparst hat.

2020-04-23

└ Qualitätssicherung
└ Unit Tests
└ Unit Test Beispiel

Unit Test Beispiel

```
1 //...
2 #include <iostream>
3 #include <string>
4
5 int main()
6 {
7     std::string text("ó");
8     std::cout << text;
9 }
10
```

Dieser Unit Test überprüft beispielsweise ob der C++ Parser mit Unicode Zeichen zurechtkommt. Dazu wird C++ Code in Form eines Strings angelegt. Der Code enthält auch Unicode Zeichen. Dann wird überprüft ob der Parser den Code richtig geparst hat.



Die Code Coverage gibt an wie viele Zeilen des Codes durch die Unit Tests abgedeckt werden. Die Code Coverage wird automatisch mit den Tool Coveralls ermittelt.

Unit Test Coverage



Die Code Coverage gibt an wie viele Zeilen des Codes durch die Unit Tests abgedeckt werden. Die Code Coverage wird automatisch mit den Tool Coveralls ermittelt.

Aktuell ist 45% des Codes über Unit Test abgedeckt. Dabei haben die unterschiedlich Sprachen unterschiedlich viel Code Coverage. Java hat z. B. 81.71% coverage, JavaScript hat 27.72% und Swift nur 0.27%.



PMD & CPD



PMD nutzt die eigenen Features
um statische Analysen des Codes
durchzuführen.



Codacy



Codacy ist eine SaaS Anwendung
welche die Codequalität und
Sicherheit anhand statischer
Analysen bewertet.

SonarCloud

sonarcloud 

SonarCloud ist eine Cloud
basierte Lösung für
Softwarequalitätsanalysen,

Verlässlichkeit und Sicherheit,
Schwachstellen, schlecht
strukturierter Code und Fehler
werden automatisch gefunden.
Es wird für mehr als 20 Sprachen
angeboten.

SonarCloud



Bei der SonarCloud Analyse von PMD sieht man, dass 77 Sicherheitslücken bei der letzten Analyse aufgetreten sind.

2020-04-23
└─ Qualitätssicherung
└─ SonarCloud
└─ SonarCloud

Allerdings findet sich SonarCloud nur in Konfigurationsdateien nicht in der Dokumentation, daher wissen wir nicht ob SonarCloud aktiv verwendet wird. Das Bild wurde am 19.03.2020 17:00 Uhr aufgenommen.



Bei der SonarCloud Analyse von PMD sieht man, dass 77 Sicherheitslücken bei der letzten Analyse aufgetreten sind.