DHBW

Die verrückte Sightseeingtour
Metrik-Analyse

Version 1.1

Revision History

Date	Version	Description	Author
6/18/2013	1.1	2 Metriken analysiert	Janina Schilling, Christiane Helmchen, Yvonne Meininger

Table of Contents

Metrik-Analyse

1. Screenshots von der Analyse mit Sonar

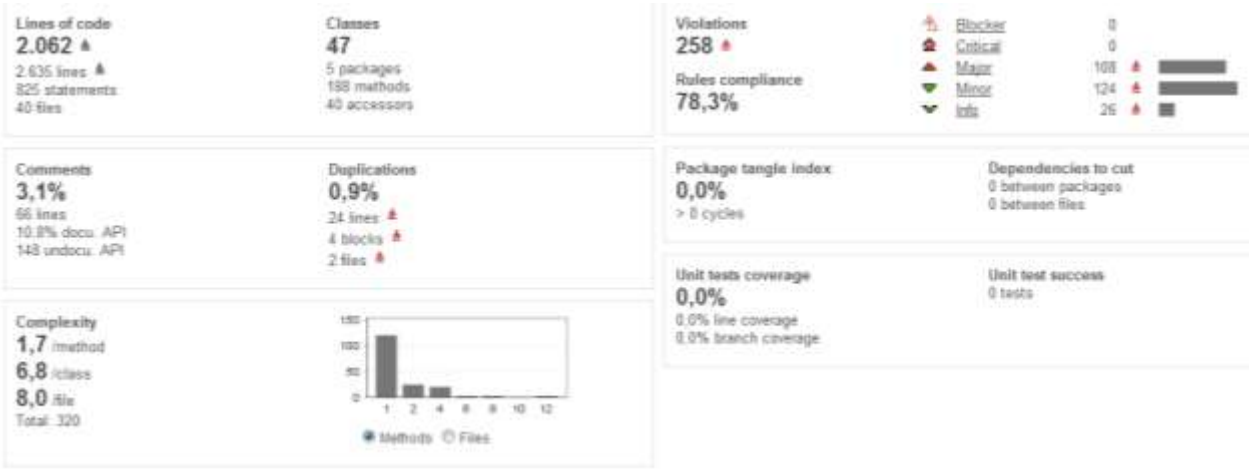


Abbildung 1: Dashboard

Most violated resources

SpielerActivity	0	0	15	0	0
Spielbrett	0	0	3	31	0
Spiel	0	0	11	4	0
Spielplatte	0	0	11	0	0
SpielView	0	0	7	10	6

Abbildung 2: Ressourcen mit den meisten Regelverstößen

Abbildung 3: Word-Cloud zeigt alle Klassen (je größer desto öfter referenziert)

2. Erläuterungen der Metriken

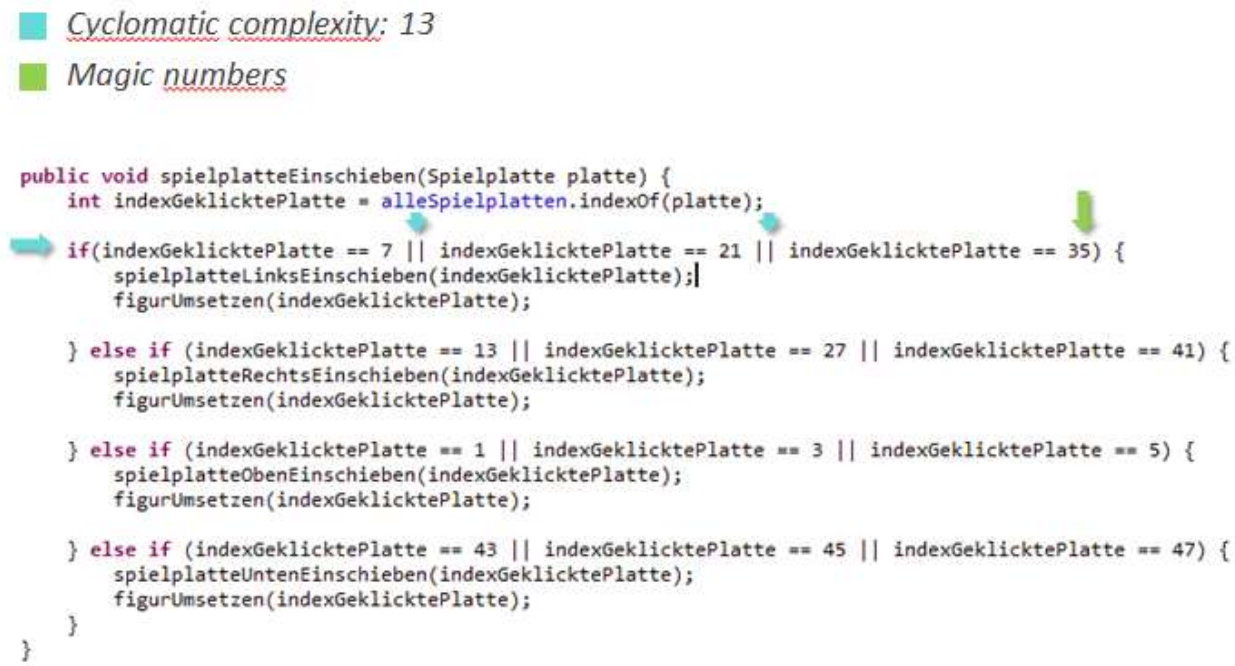
Englische Erläuterungen zu den von Sonar gemessenen Metriken zu finden unter
http://docs.codehaus.org/display/SONAR/Metric+definitions

3. Analyse von einzelnen Metriken

3.1 Cyclomatic Complexity mit Sonar

Erläuterung:
Anzahl der verschiedenen Wege durch eine Methode
Je größer die Anzahl, desto unübersichtlicher sind die verschiedenen Endzustände und es ist schwer ersichtlich, welche Fälle abgedeckt werden.

Abbildung 4: Klassen, welche mindestens eine Methode mit einer hohen Cyclomatic Complexity enthält




```
public void spielplatteEinschieben(Spielplatte platte) {
    int indexGeklicktePlatte = alleSpielplatten.indexOf(platte);

    if(indexGeklicktePlatte == 7 || indexGeklicktePlatte == 21 || indexGeklicktePlatte == 35) {
        spielplatteLinksEinschieben(indexGeklicktePlatte);
        figurUmsetzen(indexGeklicktePlatte);

    } else if (indexGeklicktePlatte == 13 || indexGeklicktePlatte == 27 || indexGeklicktePlatte == 41) {
        spielplatteRechtsEinschieben(indexGeklicktePlatte);
        figurUmsetzen(indexGeklicktePlatte);

    } else if (indexGeklicktePlatte == 1 || indexGeklicktePlatte == 3 || indexGeklicktePlatte == 5) {
        spielplatteObenEinschieben(indexGeklicktePlatte);
        figurUmsetzen(indexGeklicktePlatte);

    } else if (indexGeklicktePlatte == 43 || indexGeklicktePlatte == 45 || indexGeklicktePlatte == 47) {
        spielplatteUntenEinschieben(indexGeklicktePlatte);
        figurUmsetzen(indexGeklicktePlatte);
    }
}
```

Abbildung 5: Klasse SpielplattenEinschieber mit der relevanten Methode

Interpretation:

- Problem: mehrere IF-Anweisungen mit jeweils mehreren mit OR verknüpften Bedingungen

Idee zur Verbesserung:

- innerhalb der IF-Bedingung eine Methode aufrufen, die die verschiedenen Optionen ermittelt und einen Boolean zurückliefert

3.2 Depth of Inheritance Tree mit Metrics-Tool für Eclipse (DIT)

Erläuterung:
Maximale Pfadlänge von der Wurzel bis zur betrachteten Klasse (Vererbungstiefe)

Abbildung 6

Mittel	Standardabweichung	Maximum	Klasse mit höchstem DIT
2,4	1,709	6	SpielerAnlegenActivity

Interpretation:

- recht hohe Pfadlänge durch weitere Vererbung der von Android zur Verfügung gestellten Klassen
  - wir haben selbst nur eine Vererbungsstufe eingebaut (SpielerAnlegenActivity erbt von SpielerActivity)
  - aber SpielerActivity muss von der Android-Klasse Activity erben
  - alle Oberklassen von Activity stellt Android automatisch

Idee zur Verbesserung:

- Activity muss verwendet werden, somit kaum Verbesserung möglich
- zusätzlich eingebaute Vererbungsstufe ist hilfreich zur Vermeidung von Coderedundanz, da die SpielerBearbeitenActivity ähnliche Funktionen hat (Problem: starke Abhängigkeit von der Android Activity)