|  |
| --- |
| Elmer Lukas, Heidt Christina, Steiner Diego, Treichler Delia, Waltenspül Remo  18. Mai 2011 |

|  |
| --- |
| SE2 Projekt MRT |
| Codereview |
|  |

****

# Dokumentinformationen

## Änderungsgeschichte

|  |  |  |  |
| --- | --- | --- | --- |
| Datum | Version | Änderung | Autor |
| 04.05.2011 | 1.0 | Erste Version des Dokuments | WR |
| 06.05.2011 | 1.1 | Um Kapitel 3 und 4 ergänzt | TD |
| 08.05.2011 | 1.2 | Review für Ruby durchgeführt | EL |

## Inhaltsverzeichnis

[1 Dokumentinformationen 1](#_Toc293477605)

[1.1 Änderungsgeschichte 1](#_Toc293477606)

[1.2 Inhaltsverzeichnis 1](#_Toc293477607)

[2 Einführung 2](#_Toc293477608)

[2.1 Zweck 2](#_Toc293477609)

[2.2 Gültigkeitsbereich 2](#_Toc293477610)

[2.3 Definitionen und Abkürzungen 2](#_Toc293477611)

[2.4 Referenzen 2](#_Toc293477612)

[3 Eingesetzte Anwendungen, Plug-Ins 3](#_Toc293477613)

[3.1 Findbugs 3](#_Toc293477614)

[3.2 Checkstyle 3](#_Toc293477615)

[3.3 Eclipse Metrics 3](#_Toc293477616)

[4 Reviewprozess 4](#_Toc293477617)

[4.1 Ablauf 4](#_Toc293477618)

[4.2 Fokus bei Review 4](#_Toc293477619)

[5 Codereview 5](#_Toc293477620)

[5.1 Android 5](#_Toc293477621)

[5.1.1 Review 1 (04.05.2011) 5](#_Toc293477622)

[5.2 Rails 6](#_Toc293477623)

[5.2.1 Review 1 (08.05.2011) 6](#_Toc293477624)

# Einführung

## Zweck

Der Zweck dieses Dokuments ist die Protokollierung der durchgeführten Codereviews.

## Gültigkeitsbereich

Dieses Dokument ist während der gesamten Projektdauer gültig (21.02 bis 03.06.2011).

## Definitionen und Abkürzungen

Die Definitionen und Abkürzungen befinden sich in der ausgelagerten Datei doc/01\_Projektplan/glossar.docx.

## Referenzen

* Keine Referenzen vorhanden

# Eingesetzte Anwendungen, Plug-Ins

## Findbugs

Findbugs ist eine Open-Source Code-Analyse-Software, sie untersucht Java-Programme nach Fehlermustern. Es existiert ein Eclipse Plug-in für eine interaktive Nutzung.

Die Suche nach Fehlern geschieht ausschliesslich durch das Prüfen des Bytecodes, das Programm muss dazu nicht ausgeführt werden.

## Checkstyle

Checkstyle überprüft die Einhaltung von Coding Conventions bei der Erstellung von Java-Code. Convention-Verletzungen werden als Warnungen, nach Kategorie sortiert, angezeigt. Das Programm erlaubt einem, einen einheitlichen Coding Standard über alle Implementierungsarbeiten hinweg zu erreichen und so die Softwarequalität zu erhöhen.

## Eclipse Metrics

Das Eclipse Plugin Metrics berechnet verschiedenste Metriken bei der Überprüfung des Codes. Es gibt Warnungen für jede Metrik aus, bei der der Schwellwert überschritten ist.

Das Exportieren eines HTML-Reports ist möglich, er bietet eine gute Übersicht über alle Warnungen pro Metrik des überprüften Codes.

# Reviewprozess

## Ablauf

Ein Review wird mindestens zu zweit durgeführt. Ein Teilnehmer des Reviews ist die Person, welche den ausgewählten Code geschrieben hat.

Der Review-Vorgang besteht aus dem Durchschauen des Codes und der Anwendung der unter Kapitel 3 „Eingesetzte Anwendungen, Plug-Ins“ erwähnten Software-Produkten.

Während des Reviews wird eine Liste geführt, in der alle gefundenen Ungereimtheiten und Fehler dokumentiert werden.

Nach dem Review üben die in der Liste zugewiesenen Personen die Fehlerbehebung oder das Refactoring für die bestimmten Codeabschnitte aus.

## Fokus bei Review

* Code auf Verständnis überprüfen
* Code Smells
* Fehler finden
* Namenswahl (aussagekräftige Namen für Methoden, Klassen, Packages)
* Korrekte Abbildung von Architekturentscheidungen

# Codereview

## Android

### Review 1 (04.05.2011)

Review der Klasse LoginActivity

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Beschreibung | Schweregrad | Datum & Kürzel wenn behoben | Req. Ref. | Bemerkungen |
| 1 | Methode processLogin() ist zu umfangreich (smell Long Method) | Mittel | 04.05.2011 WR | - | Refactoring ExtractMethod |
| 2 | Ähnlicher Code wird mehrmals aufgerufen (smell Duplicated Code) | Mittel | 04.05.2011 WR | Ja | Refactoring ExtractMethod |
| 3 | Attributnamen zu wenig aussagekräftig | Leicht | 04.05.2011 WR | - | Refactoring RenameField |
| 4 | Name für Methode „checkPref“ zu wenig aussagekräftig | Leicht | 04.05.2011 WR | - | Refactoring RenameMethod |
| 5 | Zugriffsmodifikator für Methode switchToTimeEntryActivity() ist auf protected gesetzt | Mittel | 04.05.2011 WR | Ja | - |

#### Referenz ID 2

Die Objekte, welche die GUI Schaltflächen repräsentieren, werden bei jedem Zugriff neu geladen, anstatt diese direkt als Attribute der Klasse zu speichern.

#### Referenz ID 5

Die Methode switchToTimeEntryActivity() wird nur von der eigenen Klasse verwendet, dennoch steht der Zugriffsmodifikator auf protected. Dies ist unter Umständen ein Sicherheitsdefizit, welches auch unter dem Namen „Inappropriate Intimacy“ als Code Smell bekannt ist.

#### Verdikt

Das vorliegende Review wurde vom Team akzeptiert.

## Rails

### Review 1 (08.05.2011)

Review sämtlicher Models

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Beschreibung | Schweregrad | Datum & Kürzel wenn behoben | Req. Ref. | Bemerkungen |
| 1 | Kompliziertes Konstrukt anstelle von Polymorphismus verwendet (user => for\_select) | Mittel | 08.05.2011 EL | - |  |
| 2 | Auflösung Model in eine schriftliche Darstellung wird in View gemacht (Duplicated Code) | Mittel | 08.05.2011 EL | Ja | Extract Method |
| 3 |  | Leicht | 08.05.2011 EL | - |  |
| 4 |  | Leicht | 08.05.2011 EL | - |  |
| 5 |  | Mittel | 08.05.2011 EL | Ja | - |

#### Referenz ID 2

Statt einer Methode auf Modelebene (to\_s) um das Model als String darzustellen, wird bei jedem Gebrauch der String mit den passenden Attributen erstellt. Nach Konvention soll eine Methode erzeugt werden, die das Model als String repräsentieren kann (to\_s).

#### Verdikt

Das vorliegende Review wurde vom Team akzeptiert.