Die verrückte Sightseeing-Tour

Version 1.0

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 6/12/2013 | 1.0 | Testplan abgeschlossen | Christiane Helmchen, Yvonne Meininger, Janina Schilling |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 4

1.1 Purpose 4

1.2 Scope 4

1.3 Intended Audience 4

1.4 Document Structure 4

2. Evaluation Mission and Test Motivation 4

3. Target Test Items 4

4. Outline of Planned Tests 4

5. Test Approach 5

5.1 Testing Techniques and Types 5

5.1.1 Function Testing 5

5.1.2 Unit Testing 6

5.1.3 User Interface Testing 6

6. Entry and Exit Criteria 7

7. Deliverables 7

7.1 Test Evaluation Summaries 7

7.2 Reporting on Test Coverage 7

7.3 Perceived Quality Reports 7

7.4 Incident Logs and Change Requests 7

7.5 Smoke Test Suite and Supporting Test Scripts 7

7.6 Additional Work Products 7

8. Testing Workflow 8

9. Environmental Needs 9

9.1 Base System Hardware 9

9.2 Base Software Elements in the Test Environment 9

9.3 Productivity and Support Tools 9

9.4 Test Environment Configurations 9

10. Responsibilities, Staffing, and Training Needs 10

10.1 People and Roles 10

10.2 Staffing and Training Needs 10

11. Iteration Milestones 11

12. Risks, Dependencies, Assumptions, and Constraints 11

13. Management Process and Procedures 11

Test Plan

# Introduction

## Purpose

The purpose of the Test Plan is to gather all of the information necessary to plan and control the test effort for the project. It describes the approach to testing the software, and is the top-level plan generated and used by managers to direct the test effort.

This Test Plan for the DVST supports the following objectives:

* Outlines the testing approach that will be used.
* Identifies the required resources and provides an estimate of the test efforts.
* Lists the deliverable elements of the test project.

## Scope

* Unit und Functional Tests

## Intended Audience

* für den Kunden um zu zeigen wie die Qualität des Produktes gesichert wird

## Document Structure

* im Folgenden werden die verschiedenen genutzten Testarten, benötigte Ressourcen und Software sowie die definierten Zuständigkeiten innerhalb des Teams beschrieben

# Evaluation Mission and Test Motivation

* so viele Fehler und Probleme wie möglich identifizieren
* Qualität des Produktes sichern
* Versionskompatibilität (Tests auf verschiedenen Android Versionen)
* Gerätekompatibilität (Displayauflösung)
* Umsetzung der Requirements überprüfen

# Target Test Items

* Hauptbestandteil der Testitems sind die Modell-Klassen und die GUI

# Outline of Planned Tests

* not applicable

# Test Approach

## Testing Techniques and Types

### Function Testing

|  |  |
| --- | --- |
| Technique Objective: | * Test auf die vom Kunden gewünschten Funktionen, insbesondere die Funktionalität der Oberfläche |
| Technique: | * Nutzeraktionen (Mausklicks, Tastatureingaben) werden durch ein Tool emuliert und die danach vorherrschende Oberfläche mit einem definierten Soll-Zustand vergleichen |
| Oracles: | * Der Erfolg des Tests ist gegeben, wenn die grafische Oberfläche einen bestimmten Zustand eingenommen hat. Der Eintritt dieses Zustands kann festgestellt werden in dem man automatisiert auf die Existenz von GUI-Komponenten (Schaltflächen, Bilder, Eingabefelder) prüft |
| Required Tools: | * Calabash-Android (basiert auf Cucumber, Gherkin für Featurdefinitionen in der Domänensprache und Ruby für Step-Definitionen) * Android Emulator |
| Success Criteria: | * Alle Features müssen erfolgreich sein (Ausnahme siehe Special Considerations) |
| Special Considerations: | * Einige Benutzeraktionen können nicht mit der verwendeten Calabash-Version getestet werden (zum Beispiel: Popups oder Listenelemente) * Alle Feature-Definitionen wurden auf Deutsch geschrieben, da es die Muttersprache aller Stakeholder ist. |

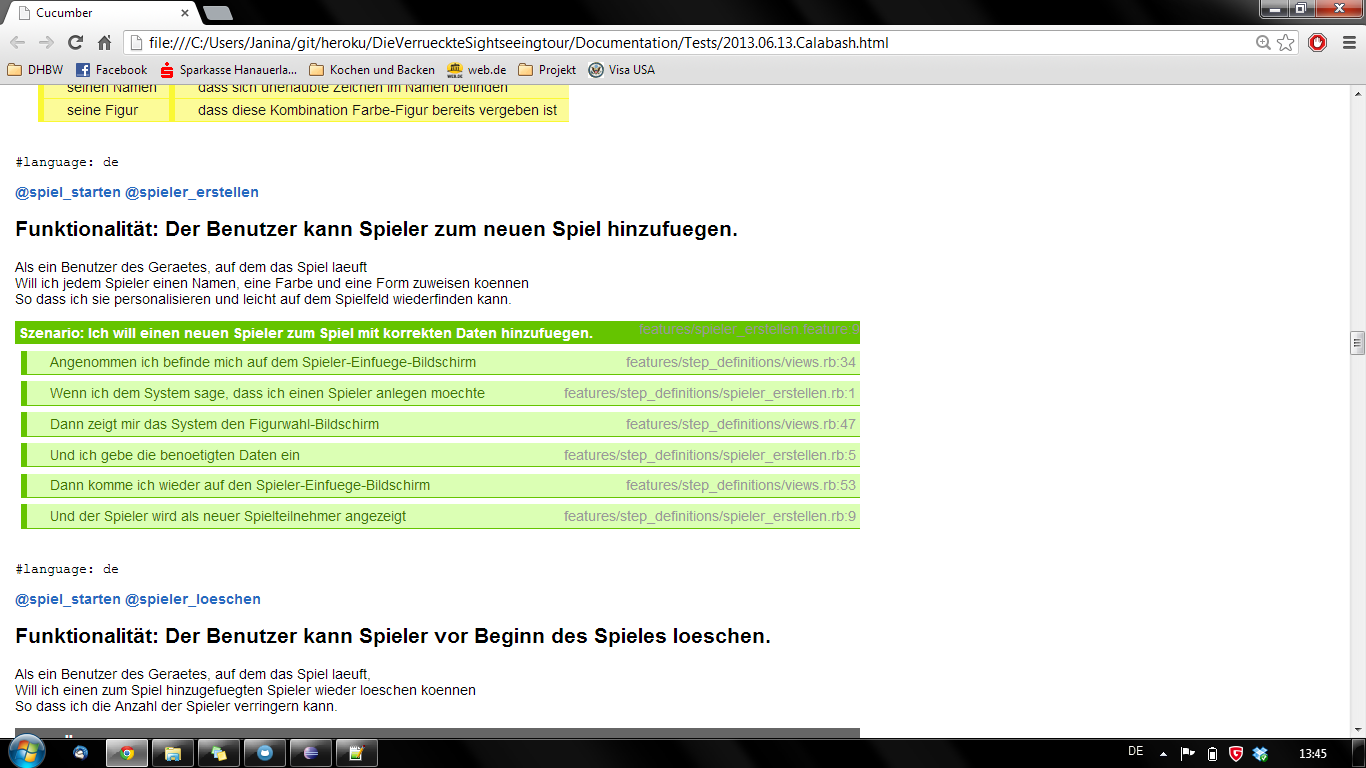


Abbildung 1: Screenshot des Calabash-Test-Logs von Use Case “Spieler erstellen”

### Unit Testing

|  |  |
| --- | --- |
| Technique Objective: | * Test der einzelnen Module (Methoden) |
| Technique: | * Methoden werden unabhängig voneinander in speziellen Testklassen ausgeführt und ihre Ergebnisse mit vordefinierten Angaben verglichen und damit auf Korrektheit geprüft |
| Oracles: | * Der Erfolg ist gegeben, wenn die aufgerufenen Assert-Methoden keine Exception werfen, enden und“ true“ zurückgeben. |
| Required Tools: | * JUnit |
| Success Criteria: | * Alle Unit-Tests müssen erfolgreich sein. |
| Special Considerations: | * In Android können keine Dialoge getestet werden (als Issue in der offiziellen Dokumentation von Android belegt). |

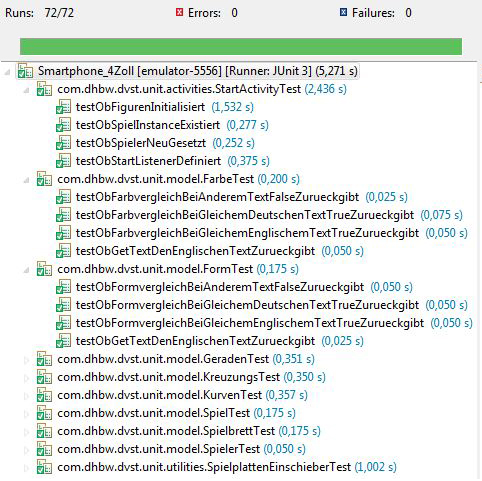


Abbildung 2: Screenshot der durchgelaufenen JUnit Tests

### User Interface Testing

Siehe Function Testing

# Entry and Exit Criteria

* Not applicable

# Deliverables

## Test Evaluation Summaries

* Not applicable

## Reporting on Test Coverage

* am Ende jedes Sprints werden alle Tests durchgeführt und die entsprechenden Log-Dateien erstellt
  + mit Ant und Emma werden Log-Dateien der Unit-Tests in Form von fertig formatiertem HTML erstellt
  + Calabash generiert einen Report in der Konsole, der manuell in eine Text-Datei kopiert wird

## JUnit Test Coverage Overview

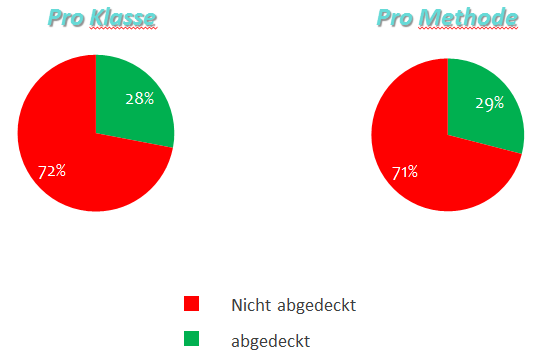


Abbildung 3: Analyse vom 19.06.2013

Der gewünschte Test Coverage von 20% (vgl. Iteration Milestones) wurde zu diesem Zeitpunkt erreicht. Jedoch bezieht sich der Test Coverage nur auf die JUnit Tests und somit auf die Java Klassen.

## Perceived Quality Reports

* Not applicable

## Incident Logs and Change Requests

* Not applicable

## Smoke Test Suite and Supporting Test Scripts

* Not applicable

## Additional Work Products

* Not applicable

# Testing Workflow

* Calabash-Tests am Ende der Implementierung jedes Use-Cases
* Unit-Tests nachdem eine Methode implementiert ist
* für jeden Use-Case sind beide Test-Arten fester Bestandteil der Projektplanung
* beide Test-Arten sind für den erfolgreichen Abschluss des Use-Cases erforderlich

# Environmental Needs

## Base System Hardware

| **System Resources** | | |
| --- | --- | --- |
| **Resource** | **Quantity** | **Name and Type** |
| Tablet | 1 | Samsung Tab 2 7.0 (Android OS) |
| Smartphone | 1 | Samsung Galaxy S3 (Android OS) |

## 

## Base Software Elements in the Test Environment

The following base software elements are required in the test environment for this Test Plan.

| **Software Element Name** | **Version** | **Type and Other Notes** |
| --- | --- | --- |
| Android Emulator |  | Virtuelle Maschine |
| JUnit (integriert in IDE) |  | Test-Framework für Unit-Tests |
| Emma |  | Test Coverage Tool |
| Ant |  | Library für Automatic Build |
| Calabash-Android |  | Automated Functional Test Tool |

## Productivity and Support Tools

The following tools will be employed to support the test process for this Test Plan.

| **Tool Category or Type** | **Tool Brand Name** | **Vendor or In-house** | **Version** |
| --- | --- | --- | --- |
| Test Management | JIRA |  |  |
| Defect Tracking | JUnit, Calabash |  |  |
| Project Management | JIRA, MS Project |  |  |

## Test Environment Configurations

* Not applicable

# Responsibilities, Staffing, and Training Needs

## People and Roles

This table shows the staffing assumptions for the test effort.

| **Human Resources** | | |
| --- | --- | --- |
| **Role** | **Resources Recommended** | **Specific Responsibilities or Comments** |
| Test Manager | Christiane Helmchen | Provides management oversight.  Responsibilities include:   * present management reporting * advocate the interests of test * evaluate effectiveness of test effort |
| Test Analyst | Christiane Helmchen | Identifies and defines the specific tests to be conducted.  Responsibilities include:   * identify test ideas * define test details * evaluate product quality |
| Tester and Implementer | Christiane Helmchen  Yvonne Meininger  Janina Schilling | Implements, unit tests the test classes and test packages and executes the tests.  Responsibilities include:   * creates the test components required to support testability requirements as defined * implement tests and test suites * execute test suites * log results * analyze and recover from test failures |

## Staffing and Training Needs

* not applicable

# Iteration Milestones

| **Milestone** | **Planned Start Date** | **Actual Start Date** | **Planned End Date** | **Actual End Date** |
| --- | --- | --- | --- | --- |
| Abgabe des ersten Calabash-Logs (Tests für ersten Use-Case) | 26.4.2013 | 26.4.2013 | 29.4.2013 | 29.4.2013 |
| Abgabe des Unit-Test Logs (Test Coverage: 20%) | 5.6.2013 | 11.6.2013 | 12.6.2013 | 12.6.2013 |
| Abgabe der gesamten Test-Logs (Test Coverage: 20%) | 20.6.2013 |  | 30.6.2013 |  |

# Risks, Dependencies, Assumptions, and Constraints

| **Risk** | **Mitigation Strategy** | **Contingency (Risk is realized)** |
| --- | --- | --- |
| mangelnde Zeit | Entscheidung für geringere Test Coverage | * Scope verringern |
| teilweise fehlende Testunterstützung durch Android (z.B. Popups) | Ausprobieren der nicht testbaren Funktionen | * Test Coverage verringern |
| teilweise fehlende Testunterstützung durch Calabash-Android | Ausprobieren der nicht testbaren Funktionen | * Test Coverage verringern |

# Management Process and Procedures

* not applicable