Mathinator

Version <1.0>

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 08.05.2017 | 1.0 | Fill the Document with basic information | Sascha Hug |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 4

1.1 Purpose 4

1.2 Scope 4

1.3 Intended Audience 4

1.4 References 4

2. Evaluation Mission and Test Motivation 4

2.1 Background 4

2.2 Evaluation Mission 4

2.3 Test Motivators 4

3. Target Test Items 5

4. Outline of Planned Tests 5

4.1 Outline of Test Inclusions 5

4.2 Outline of Other Candidates for Potential Inclusion 5

4.3 Outline of Test Exclusions 5

5. Test Approach 5

5.1.1 Data and Database Integrity Testing 5

5.1.2 Function Testing 6

5.1.3 User Interface Testing 6

6. Entry and Exit Criteria 6

6.1 Test Plan 6

6.1.1 Test Plan Entry Criteria 6

6.1.2 Test Plan Exit Criteria 6

7. Deliverables 6

7.1 Test Evaluation Summaries 6

7.2 Perceived Quality Reports 6

7.3 Incident Logs and Change Requests 7

7.4 Smoke Test Suite and Supporting Test Scripts 7

7.5 Additional Work Products 7

7.5.1 Detailed Test Results 7

7.5.2 Additional Automated Functional Test Scripts 7

7.5.3 Test Guidelines 7

7.5.4 Traceability Matrices 7

8. Testing Workflow 7

9. Environmental Needs 7

9.1 Base System Hardware 7

9.2 Base Software Elements in the Test Environment 7

9.3 Productivity and Support Tools 7

10. Responsibilities, Staffing, and Training Needs 8

10.1 People and Roles 8

10.2 Staffing and Training Needs 10

11. Iteration Milestones 10

12. Risks, Dependencies, Assumptions, and Constraints 10

13. Management Process and Procedures 11

13.1 Measuring and Assessing the Extent of Testing 11

13.2 Assessing the Deliverables of this Test Plan 11

13.3 Problem Reporting, Escalation, and Issue Resolution 11

13.4 Traceability Strategies 11

13.5 Approval and Signoff 11

# Introduction

## Purpose

The purpose of the Iteration Test Plan is to gather all of the information necessary to plan and control the test effort for a given iteration. 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 Mathinator supports the following objectives:

* Model
* View
* Presenter

## Scope

Unit Test with Junit

* Testing all buttons and text fields
* Testing the functionality of the calculator
* Database connection

## Intended Audience

Students

## References

[GitHub](https://github.com/SaschaHug/Mathinator)

[Blog](https://mathinator.tobiaslamm.de/)

# Evaluation Mission and Test Motivation

Testing needs to be done to guarantee that the software is stable and furthermore stays stable over the development of new features. After releasing patches and / or fixing bugs testing is also necessary to guarantee the best and most stable experience for the user.

## Background

Software Testing is necessary because we all make mistakes. Some of those mistakes are unimportant, but some of them are expensive or dangerous. We need to check everything and anything we produce because things can always go wrong – humans make mistakes all the time.

## Evaluation Mission

Testing is done to provide stable software. And we try to reach this goal by following these points.

* find as many bugs as possible
* find important problems, assess perceived quality risks
* advise about perceived project risks
* certify to a standard
* verify a specification (requirements, design or claims)
* advise about product quality, satisfy stakeholders
* advise about testing
* fulfill process mandates
* and so forth

## Test Motivators

* Reduce technical risks.
* Functional and no-functional requirements
* Design elements
* Ensure q high quality software

# Target Test Items

The listing below identifies those test items⎯software, hardware, and supporting product elements ⎯that have been identified as targets for testing. This list represents what items will be tested.

* Model (Logic)
* View (Design)
* Presenter (GUI)

# Outline of Planned Tests

Testing with Junit

(Alternative Espresso)

## Outline of Test Inclusions

* [Provide a high level outline of the major testing planned for the current iteration. Note what will be included in the plan and record what will explicitly **not** be included in the section titled Outline of Test Exclusions.]

## Outline of Other Candidates for Potential Inclusion

* [Separately outline test areas you suspect might be useful to investigate and evaluate, but that have not been sufficiently researched to know if they are important to pursue.]

## Outline of Test Exclusions

* [Provide a high level outline of the potential tests that might have been conducted but that have been **explicitly excluded** from this plan. If a type of test will not be implemented and executed, indicate this in a sentence stating the test will not be implemented or executed and stating the justification, such as:
* “These tests do not help achieve the evaluation mission.”
* Take a Picture -> “There are insufficient resources to conduct these tests.”
* “These tests are unnecessary due to the testing conducted by xxxx.”
* As a heuristic, if you think it would be reasonable for one of your audience members to expect a certain aspect of testing to be included that you will not or cannot address, you should note it’s exclusion: If the team agrees the exclusion is obvious, you probably don’t need to list it.]

# Test Approach

● Functional Tests

● Unit Tests, automatically after gradle build

● Testing with end user

### Data and Database Integrity Testing

|  |  |
| --- | --- |
| Technique Objective: | Testing the SQLite Database |
| Technique: | * View History * Delete Entry |
| Oracles: | * If all calculation results are correctly stored or deleted from the Database and viewed in history the tests are successful |
| Required Tools: | * Junit * database SQL |
| Success Criteria: | * the user can see their solved terms in history |

### Function Testing

|  |  |
| --- | --- |
| Technique Objective: | * Testing the calculator and the OCR-Framework |
| Technique: | * Take a Picture * Do manual calculator |
| Oracles: | * If all terms are done and solved correctly |
| Required Tools: | * Framework to recognize terms * selfmade calculator |
| Success Criteria: | * the user can solve their terms |

### User Interface Testing

|  |  |
| --- | --- |
| Technique Objective: | * Testing the GUI |
| Technique: | * Show Tour on first start * Menu buttons * Welcome screen |
| Oracles: | * If the user can navigate through the GUI |
| Required Tools: | * n/a |
| Success Criteria: | * A working surface |

# Entry and Exit Criteria

## Test Plan

### Test Plan Entry Criteria

* Database connection have to consist and some sample data

### Test Plan Exit Criteria

* Recognize and solve a term by taking a picture

# Deliverables

* Screenshots
* Video Capture of the running test

## Test Evaluation Summaries

* An evaluation can / should / will be produced after a new test was implemented / finished.

## Perceived Quality Reports

* n/a

## Incident Logs and Change Requests

* Junit

## Smoke Test Suite and Supporting Test Scripts

* n/a

## Additional Work Products

* n/a

### Detailed Test Results

* n/a

### Additional Automated Functional Test Scripts

* n/a

### Test Guidelines

* n/a

### Traceability Matrices

* n/a

# Testing Workflow

* Tests will be executed on every build
* Test results are coming soon.

# Environmental Needs

* A running Computer with Android Studio or a smartphone with Android version 5.0 (SDK 21)

## Base System Hardware

| **System Resources** | | |
| --- | --- | --- |
| **Resource** | **Quantity** | **Name and Type** |
| Database Server |  | SQLite |
| Client Test device | 1 | Huawei P8 lite |
| Test Development PCs | 3 | Sascha.Tobias,Tim |

## 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 | 5.0 or higher | Operating System |
| SQLite | 3.9.2 | Database system |
| Android Studio | Most recent | IDE |

## 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 | Junit | Vendor | 4.12 |
| Project Management | Youtrack | Vendor | Most recent |

# Responsibilities, Staffing, and Training Needs

## People and Roles

This table shows the staffing assumptions for the test effort.

| **Human Resources** | | |
| --- | --- | --- |
| **Role** | **Minimum Resources Recommended**  **(number of full-time roles allocated)** | **Specific Responsibilities or Comments** |
| Test Manager | 1 | Provides management oversight.  Responsibilities include:   * planning and logistics * agree mission * identify motivators * acquire appropriate resources * present management reporting * advocate the interests of test * evaluate effectiveness of test effort |
| Test Analyst | 2 | Identifies and defines the specific tests to be conducted.  Responsibilities include:   * identify test ideas * define test details * determine test results * document change requests * evaluate product quality |
| Test Designer | 2 | Defines the technical approach to the implementation of the test effort.  Responsibilities include:   * define test approach * define test automation architecture * verify test techniques * define testability elements * structure test implementation |
| Tester | 3 | Implements and executes the tests.  Responsibilities include:   * implement tests and test suites * execute test suites * log results * analyze and recover from test failures * document incidents |
| Database Administrator, Database Manager | 1 | Ensures test data (database) environment and assets are managed and maintained.  Responsibilities include:   * support the administration of test data and test beds (database). |
| Designer | 2 | Identifies and defines the operations, attributes, and associations of the test classes.  Responsibilities include:   * defines the test classes required to support testability requirements as defined by the test team |
| Implementer | 2 | Implements and unit tests the test classes and test packages.  Responsibilities include:   * creates the test components required to support testability requirements as defined by the designer |

## Staffing and Training Needs

This section outlines how to approach staffing and training the test roles for the project.

# Iteration Milestones

* [Identify the key schedule milestones that set the context for the Testing effort. Avoid repeating too much detail that is documented elsewhere in plans that address the entire project.]

| **Milestone** | **Planned Start Date** | **Actual Start Date** | **Planned End Date** | **Actual End Date** |
| --- | --- | --- | --- | --- |
| Iteration Plan agreed |  |  |  |  |
| Architecture baselined |  |  |  |  |
| User Interface baselined |  |  |  |  |
| First Build delivered to test |  |  |  |  |
| First Build accepted into test |  |  |  |  |

# Risks, Dependencies, Assumptions, and Constraints

| **Risk** | **Mitigation Strategy** | **Contingency (Risk is realized)** |
| --- | --- | --- |
| Test data proves to be inadequate. | <Customer> will ensure a full set of suitable and protected test data is available.  <Tester> will indicate what is required and will verify the suitability of test data. | * Redefine test data * Review Test Plan and modify * components (that is, scripts) * Consider Load Test Failure |
| Database requires refresh. | <System Admin> will endeavor to ensure the Database is regularly refreshed as required by <Tester>. | * Restore data and restart * Clear Database |

| **Dependency between** | **Potential Impact of Dependency** | **Owners** |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

| **Assumption to be proven** | **Impact of Assumption being incorrect** | **Owners** |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

| **Constraint on** | **Impact Constraint has on test effort** | **Owners** |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

# Management Process and Procedures

* n/a

## Measuring and Assessing the Extent of Testing

* n/a

## Assessing the Deliverables of this Test Plan

* n/a

## Problem Reporting, Escalation, and Issue Resolution

* Problems and bugs will be escalated as issues in our YouTrack Project Management Tool
* Our Project Manager will try to work on a solution with the Team

## Traceability Strategies

* Tracking via YouTrack Issues and GitHub commits and pushes

## Approval and Signoff

* n/a