Testplan

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 15/05/2016 | 1.0 | First version | Mehmet Ali Incekara |
| 23/05/2016 | 1.1 | Fill document | Mehmet Ali Incekara and Marvin Zerulla |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 5

1.1 Purpose 5

1.2 Scope 5

1.3 Intended Audience 5

1.4 Document Terminology and Acronyms 5

1.5 References 5

1.6 Document Structure 5

2. Evaluation Mission and Test Motivation 6

2.1 Background 6

2.2 Evaluation Mission 6

2.3 Test Motivators 6

3. Target Test Items 6

4. Outline of Planned Tests 6

4.1 Outline of Test Inclusions 6

4.2 Outline of Other Candidates for Potential Inclusion 6

4.3 Outline of Test Exclusions 6

5. Test Approach 7

5.1 Initial Test-Idea Catalogs and Other Reference Sources 7

5.2 Testing Techniques and Types 7

5.2.1 Data and Database Integrity Testing 7

5.2.2 Function Testing 7

5.2.3 Business Cycle Testing 7

5.2.4 User Interface Testing 7

5.2.5 Performance Profiling 7

5.2.6 Load Testing 8

5.2.7 Stress Testing 8

5.2.8 Volume Testing 8

5.2.9 Security and Access Control Testing 8

5.2.10 Failover and Recovery Testing 8

5.2.11 Configuration Testing 8

5.2.12 Installation Testing 8

6. Entry and Exit Criteria 9

6.1 Test Plan 9

6.1.1 Test Plan Entry Criteria 9

6.1.2 Test Plan Exit Criteria 9

6.1.3 Suspension and Resumption Criteria 9

6.2 Test Cycles 9

6.2.1 Test Cycle Entry Criteria 9

6.2.2 Test Cycle Exit Criteria 9

6.2.3 Test Cycle Abnormal Termination 9

7. Deliverables 9

7.1 Test Evaluation Summaries 9

7.2 Reporting on Test Coverage 9

7.3 Perceived Quality Reports 9

7.4 Incident Logs and Change Requests 9

7.5 Smoke Test Suite and Supporting Test Scripts 10

7.6 Additional Work Products 10

7.6.1 Detailed Test Results 10

7.6.2 Additional Automated Functional Test Scripts 10

7.6.3 Test Guidelines 10

7.6.4 Traceability Matrices 10

8. Testing Workflow 10

9. Environmental Needs 11

9.1 Base System Hardware 11

9.2 Base Software Elements in the Test Environment 11

9.3 Productivity and Support Tools 11

9.4 Test Environment Configurations 12

10. Responsibilities, Staffing, and Training Needs 12

10.1 People and Roles 12

10.2 Staffing and Training Needs 14

11. Iteration Milestones 14

12. Risks, Dependencies, Assumptions, and Constraints 15

13. Management Process and Procedures 16

13.1 Measuring and Assessing the Extent of Testing 16

13.2 Assessing the Deliverables of this Test Plan 16

13.3 Problem Reporting, Escalation, and Issue Resolution 16

13.4 Managing Test Cycles 16

13.5 Traceability Strategies 16

13.6 Approval and Signoff 17

# 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 supports the following objectives:

* The tests are aligned with the interface and functionality.
* The motivation is to develop an application with the fewest number of bugs.

## Scope

This document addresses the following types and levels of testing:

* JUnit tests
* Functional tests
* Blackbox TestFX tests
* JUnit like GUI tests with TestFX
* Installation test

## Intended Audience

n/a

## Document Terminology and Acronyms

tbd.

## References

|  |  |
| --- | --- |
| SikuliX – Directory | <https://github.com/nappydevelopment/docs/tree/master/sikulix> |
| Junit + GUI – Directory | <https://github.com/nappydevelopment/Nappy-the-ingenious/tree/master/src/test/java/nappydevelopment/nappyTheIngenious> |
| Codacy | <https://www.codacy.com/app/NappyDevelopment/Nappy-the-ingenious/dashboard> |
| Coveralls | <https://coveralls.io/github/nappydevelopment/Nappy-the-ingenious?branch=master> |
| SonarQube | <http://193.196.7.25/overview?id=5235> |
|  |  |
|  |  |
|  |  |

## Document Structure

n/a

# Evaluation Mission and Test Motivation

## Background

Testing supports all project members with feedback to their work. It ensures that use cases and functionalities are implemented in a correct manner and shows if changes infected the behaviour of the application in a bad way.

## Evaluation Mission

In general, our mission is to improve our design and code quality.

This contains to find as many bugs as possible, find quality risks and so forth.

## Test Motivators

Tests reduce bugs in new features and in existing features. Also tests are good documentation and reduce the cost of work if something needs to be changed.

# 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.

* Desktop application “Nappy, the ingenious”
* Database connection

# Outline of Planned Tests

## Outline of Test Inclusions

Unit test for the game logic and GUI.

## Outline of Other Candidates for Potential Inclusion

Database and Wiki stress-test. It is unlikely that we add more Characters.

## Outline of Test Exclusions

Database (H2) itself

JavaFX library

Java System API (open Browser, Mail client, …)

# Test Approach

## Initial Test-Idea Catalogs and Other Reference Sources

n/a

## Testing Techniques and Types

### Data and Database Integrity Testing

n/a

### Function Testing

Customers can define SikuliX tests in plain text with screenshots and developers implement these in Java.

|  |  |
| --- | --- |
| Technique Objective: | Correct implementation of use-cases. |
| Technique: | Execute each use-case scenario’s individual use-case flows or functions and features, using valid and invalid data, to verify that:   the expected results occur when valid data is used |
| Oracles: | n/a |
| Required Tools: | SikuliX |
| Success Criteria: | SikuliX tests don’t fail |
| Special Considerations: | n/a |

### Business Cycle Testing

n/a

### User Interface Testing

User Interface (UI) testing verifies a user’s interaction with the software. The goal of UI testing is to ensure that the UI provides the user with the appropriate access and navigation through the functions of the target-of-test.

|  |  |
| --- | --- |
| Technique Objective: | Navigation through the target-of-test reflecting business functions and requirements, including window-to-window, field-to- field, and use of access methods (tab keys, mouse movements, accelerator keys).  Window objects and characteristics can be exercised–such as menus, size, position, state, and focus. |
| Technique: | Create or modify tests for each window to verify proper navigation and object states for each application window and object. |
| Oracles: | Maven Test or build fails. |
| Required Tools: | Java  Maven TestFX (loaded by maven automatically) JUnit (loaded by maven automatically) |
| Success Criteria: | Every Unit tests passes without assert fails or unexpected exceptions. |
| Special Considerations: | n/a |

### Performance Profiling

n/a

### Load Testing

n/a

### Stress Testing

n/a

### Volume Testing

n/a

### Security and Access Control Testing

n/a

### Failover and Recovery Testing

n/a

### Configuration Testing

n/a

### Installation Testing

|  |  |
| --- | --- |
| Technique Objective: | new installation: a new machine, never installed previously with Nappy, the ingenious  update: a machine previously installed Nappy, the ingenious, older version |
| Technique: | Install desktop application on a linux, mac or windows. |
| Oracles: | n/a |
| Required Tools: | Java 8 (JRE or JDK) |
| Success Criteria: | Installation Testing Document should be marked successfully |
| Special Considerations: | n/a |

# 

# Entry and Exit Criteria

## Test Plan

### Test Plan Entry Criteria

The entry of the Test Plan is to take immediately after the first version.

### Test Plan Exit Criteria

The Test Plan is completed once all tests have been successfully completed.

### Suspension and Resumption Criteria

The Test Plan cannot be canceled.

## Test Cycles

n/a

# Deliverables

## Test Evaluation Summaries

Our summary of our tests are always up-to-date in Travis:   
<https://travis-ci.org/nappydevelopment/Nappy-the-ingenious>

The summary will produce with every push on GitHub.

## Reporting on Test Coverage

Test Coverage is reported in:

* SonarQube: <http://193.196.7.25/overview?id=5235>
* Coveralls: <https://coveralls.io/github/nappydevelopment/Nappy-the-ingenious?branch=master>
* Codacy: <https://www.codacy.com/app/NappyDevelopment/Nappy-the-ingenious/dashboard>

## Perceived Quality Reports

[Provide a brief outline of both the form and content of the reports used to measure the perceived quality of the product, and indicate how frequently they will be produced. Give an indication about to the method and tools used to record, measure, and report on the perceived product quality. You might include some analysis of Incidents and Change Request over Test Coverage.]

## Incident Logs and Change Requests

n/a

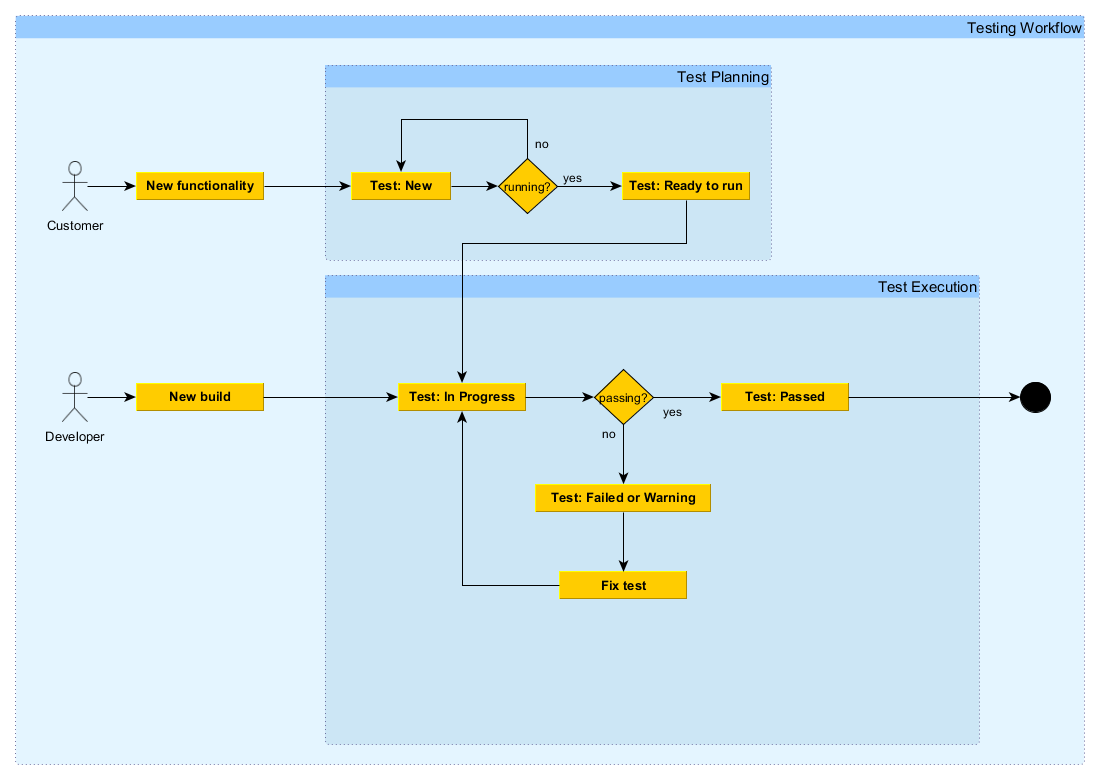
## Smoke Test Suite and Supporting Test Scripts

n/a

## Additional Work Products

n/a

# Testing Workflow



# Environmental Needs

## Base System Hardware

The following table sets forth the system resources for the test effort presented in this *Test Plan*.

| **System Resources** | | |
| --- | --- | --- |
| **Resource** | **Quantity** | **Name and Type** |
| Database Server |  |  |
| —Network or Subnet |  | TBD |
| —Server Name |  | TBD |
| —Database Name |  | TBD |
| Client Test PCs |  |  |
| —Include special configuration requirements |  | TBD |
| Test Repository |  |  |
| —Network or Subnet |  | TBD |
| —Server Name |  | TBD |
| Test Development PCs |  | TBD |

## 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** |
| --- | --- | --- |
| NT Workstation |  | Operating System |
| Windows 2000 |  | Operating System |
| Internet Explorer |  | Internet Browser |
| Netscape Navigator |  | Internet Browser |
| MS Outlook |  | eMail Client software |
| Network Associates McAfee Virus Checker |  | Virus Detection and Recovery Software |

## 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 |  |  |  |
| Defect Tracking |  |  |  |
| ASQ Tool for functional testing |  |  |  |
| ASQ Tool for performance testing |  |  |  |
| Test Coverage Monitor or Profiler |  |  |  |
| Project Management |  |  |  |
| DBMS tools |  |  |  |

## Test Environment Configurations

The following Test Environment Configurations needs to be provided and supported for this project.

| **Configuration Name** | **Description** | **Implemented in Physical Configuration** |
| --- | --- | --- |
| Average user configuration |  |  |
| Minimal configuration supported |  |  |
| Visually and mobility challenged |  |  |
| International Double Byte OS |  |  |
| Network installation (not client) |  |  |

# 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 | Mehmet Ali Incekara  Marvin Zerulla | Provides management oversight.  Responsibilities include:   * planning * agree mission * identify motivators * present management reporting |
| Test Analyst | Mehmet Ali Incekara  Marvin Zerulla | Identifies and defines the specific tests to be conducted.  Responsibilities include:   * define test details * evaluate effectiveness of test effort * document change requests * evaluate product quality |
| Test Designer | Marvin Zerulla | 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 | Marvin Zerulla  Mehmet Ali Incekara  Marc Mahler  Manuel Bothner | Implements and executes the tests.  Responsibilities include:   * implement tests * execute test suites * analyze and recover from test failures |
| Implementer | Marvin Zerulla  Mehmet Ali Incekara  Marc Mahler  Manuel Bothner | 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.

[The way to approach staffing and training will vary from project to project. If this section is part of a Master Test Plan, you should indicate at what points in the project lifecycle different skills and numbers of staff are needed. If this is an Iteration Test Plan, you should focus mainly on where and what training might occur during the Iteration.

Give thought to your training needs, and plan to schedule this based on a Just-In-Time (JIT) approach—there is often a temptation to attend training too far in advance of its usage when the test team has apparent slack. Doing this introduces the risk of the training being forgotten by the time it's needed.

Look for opportunities to combine the purchase of productivity tools with training on those tools, and arrange with the vendor to delay delivery of the training until just before you need it. If you have enough headcount, consider having training delivered in a customized manner for you, possibly at your own site.

The test team often requires the support and skills of other team members not directly part of the test team. Make sure you arrange in your plan for appropriate availability of System Administrators, Database Administrators, and Developers who are required to enable the test effort.]

# Iteration Milestones

| **Milestone** | **Planned Start Date** | **Actual Start Date** | **Planned End Date** | **Actual End Date** |
| --- | --- | --- | --- | --- |
| Iteration Plan agreed |  |  |  |  |
| Iteration starts |  |  |  |  |
| Requirements baselined |  |  |  |  |
| Architecture baselined |  |  |  |  |
| User Interface baselined |  |  |  |  |
| First Build delivered to test |  |  |  |  |
| First Build accepted into test |  |  |  |  |
| First Build test cycle finishes |  |  |  |  |
| [Build Two will not be tested] |  |  |  |  |
| Third Build delivered to test |  |  |  |  |
| Third Build accepted into test |  |  |  |  |
| Third Build test cycle finishes |  |  |  |  |
| Fourth Build delivered to test |  |  |  |  |
| Fourth Build accepted into test |  |  |  |  |
| Iteration Assessment review |  |  |  |  |
| Iteration ends |  |  |  |  |

# Risks, Dependencies, Assumptions, and Constraints

| **Risk** | **Mitigation Strategy** | **Contingency (Risk is realized)** |
| --- | --- | --- |
| Prerequisite entry criteria is not met. | <Tester> will define the prerequisites that must be met before Load Testing can start.  <Customer> will endeavor to meet prerequisites indicated by <Tester>. | * Meet outstanding prerequisites * Consider Load Test Failure |
| 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 |

[List any dependencies identified during the development of this **Test Plan** that may affect its successful execution if those dependencies are not honored. Typically these dependencies relate to activities on the critical path that are prerequisites or post-requisites to one or more preceding (or subsequent) activities You should consider responsibilities you are relying on other teams or staff members external to the test effort completing, timing and dependencies of other planned tasks, the reliance on certain work products being produced.]

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

[List any assumptions made during the development of this **Test Plan** that may affect its successful execution if those assumptions are proven incorrect. Assumptions might relate to work you assume other teams are doing, expectations that certain aspects of the product or environment are stable, and so forth].

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

[List any constraints placed on the test effort that have had a negative effect on the way in which this **Test Plan** has been approached.]

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

# Management Process and Procedures

[Outline what processes and procedures are to be used when issues arise with the **Test Plan** and its enactment.]

## Measuring and Assessing the Extent of Testing

[Outline the measurement and assessment process to be used to track the extent of testing.]

## Assessing the Deliverables of this Test Plan

[Outline the assessment process for reviewing and accepting the deliverables of this **Test Plan**]

## Problem Reporting, Escalation, and Issue Resolution

[Define how process problems will be reported and escalated, and the process to be followed to achieve resolution.]

## Managing Test Cycles

[Outline the management control process for a test cycle.]

## Traceability Strategies

[Consider appropriate traceability strategies for:

* Coverage of Testing against Specifications — enables measurement the extent of testing
* Motivations for Testing — enables assessment of relevance of tests to help determine whether to maintain or retire tests
* Software Design Elements — enables tracking of subsequent design changes that would necessitate rerunning tests or retiring them
* Resulting Change Requests — enables the tests that discovered the need for the change to be identified and re-run to verify the change request has been completed successfully]

## Approval and Signoff

[Outline the approval process and list the job titles (and names of current incumbents) that initially must approve the plan, and sign off on the plans satisfactory execution.]