

**SOKOBAN TEST STRATEGY DOCUMENT**

**Project Name:** Gameokoban

**Project Number:** 1

**Prepared For:** Retro Games Ltd

**Prepared By:** Chris Dworczyk

**Creation:** 05/3/2018

**Last Updated:** 07/3/2018

**Date Submitted:** 08/3/2018

**Class:** HND Software Development

**Version Number:** 01

**Security Classification:** Low

# Document Control

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision No** | **Revision Date** | **Description of Change** | **Author** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Document Detail

|  |  |
| --- | --- |
| **Title** | SOKOBAN TEST STRATEGY DOCUMENT |
| **Version** | V1 |
| **Date** | 03/3/2018 |
| **Electronic File Name** | TestPlan(Chris Dworczyk) |
| **Electronic File Location** |  |
| **Author** | Chris Dworczyk |
| **Contributors** |  |

# Referenced Documentation

|  |  |  |
| --- | --- | --- |
| **Ref** | **Document Name** | **Electronic File location** |
| Functional validation | DataValidationTestL | F:\HND\Sokoban Game\Documents To Submit\Testing\Functional |
| Event validation | EventValidationTest | F:\HND\Sokoban Game\Documents To Submit\Testing\Functional |
| Navigation validaton | CodeRectifications | F:\HND\Sokoban Game\Documents To Submit\Testing\Functional |

# Team Members and Roles

|  |  |
| --- | --- |
| **Resource Name** | **Role** |
| N/A | Project Manager |
| N/A | Programmer |
| Chris Dworczyk | Tester |

# Diary/Log of Errors

|  |  |  |
| --- | --- | --- |
| **Date** | **Error Description** | **Action** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Table of Contents

[Document Control 2](#_Toc371420101)

[Revision History 2](#_Toc371420102)

[Document Detail 2](#_Toc371420103)

[Referenced Documentation 2](#_Toc371420104)

[Team Members and Roles 2](#_Toc371420105)

[Diary/Log of Errors 2](#_Toc371420106)

[1.0 Introduction 4](#_Toc371420107)

[2.0 Objectives 4](#_Toc371420108)

[3.0 Scope 4](#_Toc371420109)

[4.0 Test Items 4](#_Toc371420110)

[5.0 Testing Methods 4](#_Toc371420111)

[6.0 Testing Strategy 4](#_Toc371420112)

[7.0 Testing Levels 5](#_Toc371420113)

[7.1.4 Acceptance Testing 5](#_Toc371420114)

[9.0 Exit Criteria 5](#_Toc371420115)

[10.0 Environmental Needs 5](#_Toc371420116)

[11.0 Staffing and Training 6](#_Toc371420117)

[12.0 Test Schedule 6](#_Toc371420118)

[13.0 Testing Deliverables 6](#_Toc371420119)

[15.0 Features not to be Tested 6](#_Toc371420120)

[16.0 Risks and Contingencies 7](#_Toc371420121)

[17.0 Approvals 7](#_Toc371420122)

[Appendices 8](#_Toc371420123)

# Introduction

I have created a 2D Sokoban game for Retro Games Ltd in which you try and get all the crates on buttons to progress to the next level. The game has some tough logic in it like wall detection which will needed to be tested thoroughly to ensure customer satisfaction.

# Objectives

I will use test logs to document all the features of the game to ensure its functionality.

# Scope

In this plan I will cover the entire navigational path in the application to make sure they all go to the correct pages. I will take a look at all the event driven controls like buttons so they all call the correct functions. Finally I will look at the functionality of the program as a while and tests all the necessary features to deliver a fully working game.

*This section describes what is being tested, such as all the functions of a specific product, its existing interfaces, integration of all functions. Test planning must be performed throughout the System Development Life Cycle. Give samples of what documentation you intend to use.*

# **Test Items**

*Navigational Log*

*Event validation log*

*Functional Validation log*

# Testing Methods

I will make use of the white box testing method as I understand my code and will be able to stop any glaring bugs once they occur. This method looks at the actual internal code instead of just the functionality of the program.

# Testing Strategy

Top down strategy will be used in this scenario as it will allow me to see if there are any major flaws at the top of the program. I will work my way down to the individual buttons and units which should help in determining the actual cause of any problem.

*In this section justify your testing strategy. There are two different strategies top down, bottom up. Below are some examples:*

*Documenting a test strategy gives you time to think about what exactly the testing has to achieve*

*Circulating a documented test strategy allows others to review, to cross reference sections and validate the content.*

*Documenting a strategy assists with the overall project planning.*

*Having a documented test strategy provides a point of reference both for you and others as the project progresses.*

*Having a documented strategy allows all stakeholders to have a say in the testing process.*

# 7.0 Testing Levels

*In this section describes the proposed levels of testing that you will undertake. Tests are frequently grouped by where they are added in the software development process, or by the level of specificity of the test.*

## Unit Testing

I will be testing each competent of the program like the buttons to make sure they perform correct functions.

This refers to tests that verify the functionality of a specific section of code.

## Integration Testing I will be testing each of the buttons corresponding to each block of code to ensure they all work correctly.

## System Testing

The system as a whole will be tested to ensure that the correct outputs are displayed on the labels.

## *System testing tests a completely integrated system to verify that it meets its requirements.*

## **7.1.4** **Acceptance Testing** *At last the system is delivered to the user for Acceptance testing.*

## 7.1.5 Installation Testing *A installation test assures that the system is installed correctly and working at actual customer's hardware.*

## 7.1.6 Compatibility Testing *A common cause of software failure (real or perceived) is a lack of its* [*compatibility*](http://en.wikipedia.org/wiki/Computer_compatibility) *with other* [*application software*](http://en.wikipedia.org/wiki/Application_software)*,* [*operating systems*](http://en.wikipedia.org/wiki/Operating_system) *(or operating system* [*versions*](http://en.wikipedia.org/wiki/Software_versioning)*, old or new), or target environments that differ greatly from the original (such as a* [*terminal*](http://en.wikipedia.org/wiki/Computer_terminal) *or* [*GUI*](http://en.wikipedia.org/wiki/GUI) *application intended to be run on the* [*desktop*](http://en.wikipedia.org/wiki/Desktop_metaphor) *now being required to become a* [*web application*](http://en.wikipedia.org/wiki/Web_application)*, which must render in a* [*web browser*](http://en.wikipedia.org/wiki/Web_browser)*).*

## 8.0 Entry Criteria

*This section will provide information when testing starts.*

# 9.0 Exit Criteria

*This section will provide information when testing stops.*

# 10.0 Environmental Needs

*This section includes the hardware and software requirements you require to undertake the tests.*

|  |  |
| --- | --- |
| **Hardware** | **Software** |
| PC | Visual Studio |
|  |  |
|  |  |

# 11.0 Staffing and Training

*This section deals with specifying staff skills and training requirements to use the software.*

# 12.0 Test Schedule

*This section will show the estimated time required to do each testing task, include milestones.*

|  |  |  |  |
| --- | --- | --- | --- |
| **Task Name** | **Start** | **Milestone/Finish** | **Comments** |
| System Analysis | 22/08/2018 | 31/08/2018 | Approved |
| Getting Test Scenarios | 30/08/2018 | 10/09/2018 | Approved |
| Unit Testing | 05/09/2018 | 20/09/2018 | Approved |
| Meeting | 10/09/2018 | 11/09/2018 | Approved |
| Test Case | 29/09/2018 | 30/09/2018 | Approved |
| Acceptance Testing | 5/10/2018 | 10/10/2018 | Approved |

# 13.0 Testing Deliverables

*This should include the test document type, person responsible and date due*.

|  |  |  |
| --- | --- | --- |
| **Deliverable** | **For** | **Date/Milestone** |
| Test Plan | Project Manager | 22/10/2018 |
| Test Logs | Project Manager | 31/10/2018 |
|  |  |  |
|  |  |  |

## 14.0 Features to be Tested

Functional validation and event validation tests will be run on the main components and controls of the program for example: buttons perform the correct actions and the labels display the correct results.

* Button – Start: should open up the main gamepage window
* Button – Exit Results: exits the application
* Button – Load Level: display a select file prompt for user to select their level
* Label – Total Steps: should display the total steps taken
* Label – Current Level: should display the current level the player is on
* Player movement – The player should be able to move around the level but not inside walls/crates
* Crate pushing - the player should be able to push the crate around the level but no inside walls/crates

*Unit testing, also known as component testing, refers to tests that verify the functionality of a specific section of code, usually at the function level.*

[*Usability testing*](http://en.wikipedia.org/wiki/Usability_testing) *is needed to check if the user interface is easy to use and understand. It is concerned mainly with the use of the application*.

## *Functional testing refers to activities that verify a specific action or function of the code. These are usually found in the code requirements documentation.*

## 

15.0 Features not to be Tested  
Software security will not be tested

# 16.0 Risks and Contingencies

*This section will specify contingency plans for each delayed of test item and identify high risk items in testing process.*

# 17.0 Approvals

*This section must specify the names and titles of persons who must approve this plan.*

|  |  |  |
| --- | --- | --- |
| **Prepared By** | **Signature** | **Date** |
| Author  Job Title  Organisation | Chris Dworczyk  Software Tester  OOP Solutions Ltd | 08/03/18 |

|  |  |  |
| --- | --- | --- |
| **Accepted By** | **Signature** | **Date** |
| Project Manager  Job Title  Organisation |  |  |