SE 3XA3: Test Plan
Space Pinball 2017

Team #17, Test Icicles
Andrew Bennett 1319879
Kyriakos Kyprianou 400025691
Teodor Tomescu 400038361

Contents

List of Tables

This document ...

1 General Information

1.1 Purpose

The purpose of this document is to provide a description of the testing methodologies used in creating Space Pinball 2017.

1.2 Overview of Document

This document contains information about our approaching to testing Space Pinball 2017. In this document you will find our testing plan, as well as specific information about the tests we conducted.

2 Plan

2.1 Software Description

Space Pinball 2017 is a software project built in the game engine Unity 3D. One of the most useful packages that comes with Unity is the Unity Test Tools suite. We will be using the Unity Test Tools Suite for all aspects of our program, which covers various testing methods including Unit Testing, Asset Testing, UI testing, and Dynamic Integration Testing. In addition to these, will be be running and logging tests within the Unity Graphics Editor and creating our own testing scripts.

Table 1: **Revision History**

Date	**Version**	**Notes**
Oct. 27th – Rev0.1	0.1	first draft
December 5th	1.1	Template adjusted to match Dr. Bokhari's. Gantt chart added to repo. Section 2 tests fixed.

2.2 Test Team

Andrew Bennett, Kyriakos Kyprianou, Teodor Tomescu

2.3 Automated Testing Approach

We will be focusing on dynamic testing to ensure the correctness of our game. More specifically we will incorporate white-box unit tests, integration tests, and user tests. Our unit tests will target the scoring mechanism mainly as we want it to be as accurate as possible. There will be various obstacles throughout the plane that the ball will be able to interact with, and our system will need to do these calculations accurately and on time in order to display this information to the user. Since gaining a high score will be the main end-game objective, it is critical that we test this sufficiently.

The integration testable code will consists of testing the game environment and the interaction between all game objects and the ball. We want to test that all surfaces have the correct texture and that all actions possible in the game function as expected.

The user tests will need to be manually tested by the group to ensure that the game maintains all the functionality that was originally intended. These tests will cover the GUI button capabilities and all user interactions with the game.

2.4 Testing Tools

The Unity game development platform has a built-in testing tool which has the ability to test code both in edit and play mode of the project. It uses the NUnit library; an open source library created to test .Net languages; to perform these test. Scripts will have to be written to test different attributes of the game, including different objects like the walls, the pinball, as well as functions of the game like gravity, bumper and flipper hit strength amongst other aspects to ensure the smoothness of the game.

2.5 Testing Schedule

See Gantt Chart in this directory

3 System Test Description

3.1 Tests for Functional Requirements

3.1.1 Shooting Mechanism Test

Type: Dynamic.
Initial State: The user has balls remaining and is ready to shoot a ball into play.
Input: User-input
Output: The ball is shot into the playing field.
How test will be performed: The shooting mechanism acts as a compressed string with force stored in it. The space button is the release for the mechanism, the longer the user holds the button the harder the ball will shoot into the playing field. Therefore there must be a test to ensure the mechanism remains immobile if no input is given. A minimum, maximum and several powers in between must also be tested to show the ball is sprung at different speeds based on how long the space key is help for.

3.1.2 Flipper Test

Type:Manual.
Initial State: The flippers are initially at rest at an approximate angle of 30-35 degrees below the horizon. A test to show the flippers will stay in place must be included.
Input:The user must use keys A for the left flipper and D for the right flipper to be activated and cause a reaction.
Output: The mentioned reaction from the flippers will be a quick motion up, to about 35-40 degrees above the horizon. This motion is intended to collide with the pinball causing it to shoot back up into the playing field in order to score more points. Various tests must be done here in order to ensure that the flippers will interact correctly with the ball if a collision takes place, as well as that the movement of the flippers is correct with or without a ball present.
How test will be performed: Manually via user input.

3.1.3 Menu Test

Type: Manual.
Initial State: The user has opened up the game and is in the main menu. This test will be a manual test completed by the group.
Input: The user presses the load game button. The user is then able to select one of his saved games to open up. The user selects a saved game instance.
Output: The user's saved game is loaded up. A countdown begins starting from 3 down to 1 and then the game is loaded up ready to be played.
How test will be performed: Manually via user input by clicking on the menu.

3.1.4 Scoring Keeping Test

Type: Dynamic.
Initial State:The ball is on course to hit a points bumper.
Input: The ball makes contact with the points bumper.
Output: At that exact frame of contact, points will be added to the top right hand corner of the information menu.
How test will be performed: Unity test tools will log collisions. We will manually verify that the correct number of collisions happened.

3.2 Tests for Nonfunctional Requirements

3.2.1 Look-and-Feel Requirements

Type: Manual
How test will be performed: We are striving to build our software that allows it to be:

- Apparently simple to use
- Approachable, so that people do not hesitate to use it
- Attractive to children, teens, and young adults
- Innovative and appearing to be state of the art
- Professional looking
- Exciting

To test each of these, subjective opinions will be collected from Space Pinball 2017's target demographic and these opinions will be used to evaluate our success in meeting each requirement.

3.2.2 Usability Requirements

Type: Manual

Space Pinball 2017 is a game that is designed to be intuitive and usable for any demographic.

Our goals for usability include:

- The game should be easy for anyone over the age of four to play and enjoy.
- Interface elements (e.g. menus) should be easy to understand
- The interface should be context sensitive and explain the purpose of the game.
- The interface actions and elements should be consistent.
- The screen layout and colour should be appealing.

Usability testing will be done by introducing real users to our software and recording any problems, questions, or positive experiences they encounter. The users who are testing the software will be asked to perform certain tasks or make efforts toward achieving certain objectives (try to earn 1000 points, try to hit the pinball through certain gates) and their progress will be recorded.

3.2.3 Performance Requirements

Type: Dynamic

Performance requirements are extremely important for Space Pinball 2017. While there are subjective components to performance requirements, the majority of our testing will look at measurable performance metrics like performance of the GPU, CPU, memory, rendering, and audio.

Unity 3D provides an incredibly useful tool called the Unity Profiler for evaluating, comparing, and recording system performance. Through the Unity Profiler, areas of our game that are particularly resource intensive will be documented and reexamined so that further updates can be made to improve performance.

3.3 Traceability Between Test Cases and Requirements

4 Tests for Proof of Concept

Testing to validate the proof of concept included a lot of user testing and manual assertion testing to verify that every component of the proof of concept was behaving as expected.
The Proof of Concept was functionally stable and served as a baseline during the constant improvement process. As new features were integrated into the existing proof of concept, tests were ran to ensure a consistently stable software.

5 Comparison to Existing Implementation

The existing implementation is now the completed final product and is tested with the tests in sections 3 and 4 of this document.

6 Unit Testing Plan

Unit testing were be an integral part of our development process for Space Pinball 2017. Unit testing was done both through the Unity Test Tool and through a custom testing suite for thorough coverage.

Some specific tests include:

- Ensuring user inputs have the desired output. Each button should function properly on its own, and will be tested to ensure that no unexpected outcomes result when atypical combinations of inputs are entered.
- Ensuring each game obstacle behaves as expected when collisions with the

pinball occur. These tests will include asset tests to verify the integrity of the 3D models used, as well as behavioural tests to ensure exceptional cases are managed.
- The pinball will be tested throughly with Unity Test Tools to verify predictable behaviour in edge-cases like high speed collisions, low speed collisions, and behaviour during the game-ready and game-over states.
- The user interface will undergo unit testing to ensure that it only accepts valid inputs from the user.