Test Log

# Test Cases

Every program has a been developed to behave in a particular way to meet the needs of the program requirement. Tests are developed to ensure the requirements are satisfied, and that the program behaves in the intended way. There are many different types of tests that we can develop to ensure correct behaviour, however we will be focusing on manual tests that verify that our project behaves as expected.

Types of tests to include:

* Build Tests  
  Tests that verify the projects build for the intended platform
* Acceptance Tests  
  Tests that verify an intended behaviour from a user/player perspective. These tests should not require any understanding of HOW the system behaves, they should focus on “when a user does x, y should happen / be observed”
* Functional Tests  
  Tese tests verify intended behaviour they may not be observed directly within the running application, but could be observed within the unity editor, they should verify that certain properties have been set correctly, components have been enabled/disabled correctly etc.

Example Tests Cases:

* Build Test: The project successfully compiles for windows
* Build Test: The project successfully compiles for android
* Build Test: The project successfully compiles for pc
* Acceptance test: The “Quit” button does not display on Mobile or Web builds
* Acceptance test: The “Virtual Joystick” only displays on Mobile builds
* Acceptance test: “Collecting a coind updates the displayed score on screen”
* Acceptance test: Enemy healthbar decreeses when shot
* Functional Tests: Highscore.txt file is created when a score entry is entered for first time
* Functional Tests: Highscore.txt file is deleted when scores are cleared
* Functional Tests: Clicking “some button” disables input for player by de-activating the “playerInput” script on the player object.

|  |  |  |
| --- | --- | --- |
| **Test Name** | 1. Windows build test | |
| **Test Type:** | Build Test | |
| **Test Platforms** | PC | |
| **Test Description:** | Making sure the project can successfully compile for windows. | |
| **Step** | **Details** | **Expected Result** |
| 1 |  |  |
| 2 |  |  |
| 3 | … insert more rows as required |  |

|  |  |  |
| --- | --- | --- |
| **Test Name** | 2. Android build test | |
| **Test Type:** | Build Test | |
| **Test Platforms** | Mobile | |
| **Test Description:** | Making sure the project can successfully compile for mobile. | |
| **Step** | **Details** | **Expected Result** |
| 1 |  |  |
| 2 |  |  |
| 3 | … insert more rows as required |  |

|  |  |  |
| --- | --- | --- |
| **Test Name** | 3. Web build test | |
| **Test Type:** | Build test | |
| **Test Platforms** | web | |
| **Test Description:** | Making sure the project can successfully compile for web | |
| **Step** | **Details** | **Expected Result** |
| 1 |  |  |
| 2 |  |  |
| 3 | … insert more rows as required |  |

|  |  |  |
| --- | --- | --- |
| **Test Name** | 4. <insert test name here> | |
| **Test Type:** | <is this an acceptance test of functional test> | |
| **Test Platforms** | (mobile | pc | web) | |
| **Test Description:** | <describe the test> | |
| **Step** | **Details** | **Expected Result** |
| 1 |  |  |
| 2 |  |  |
| 3 | … insert more rows as required |  |

# Test Log

When a new version of the software is developed (features added or refactored) The areas that had been changed should be tested to ensure that previously passing tests still pass, and that the new features / changes remain working as intended.

Most software will follow some kind of versioning scheme that will be recorded when the tests are run, for our case, we will assume to be running version 1.0.0.

|  |  |  |
| --- | --- | --- |
| **Date of Test** | <date of test> | |
| **Version** | 1.0.0 | |
| **Test Platform** | Platform tests are checked against | |
| **Test Case** | **Result (pass/fail)** | **Notes: for failing tests, describe which step failed, and what happened** |
| 1. <name of test> | Pass/Fail |  |
| 2. <name of test> | Pass/Fail |  |
| 3. <name of test> | Pass/Fail |  |
| 4. <name of test> | Pass/Fail |  |

Note: If you run your tests multiple times, make a new copy of the table

|  |  |  |
| --- | --- | --- |
| **Date of Test** | <date of test> | |
| **Version** | 1.0.0 | |
| **Test Case** | **Result (pass/fail)** | **Notes: for failing tests, describe which step failed, and what happened** |
| 1. <name of test> | Pass/Fail |  |
| 2. <name of test> | Pass/Fail |  |
| 3. <name of test> | Pass/Fail |  |
| 4. <name of test> | Pass/Fail |  |