**Dung Eon : A Dungeon Crawler**

**Test Plan**

**COP 4331, Spring, 2016**

Team Name: Bobby Sandwich

Team Members:

●     Jacob Crandall

●     Robert Bland

●     Jacob Jiskoot

Modification history:

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Who | Comment |
| V1.0 | 04/16/16 | Jacob Crandall | Initial Test Results |
| ... |  |  |  |

**Contents of this Document**

Description of Test Environment

Results of Individual Test Cases

Conclusion

**Description of Test Environment**

Actual testing did not differ from planned testing in anyway.

**Results of Individual Test Cases**

|  |  |
| --- | --- |
| Test Objective | Test the menu |
| Tested Requirements | Menureq, Playreq |
| Test Description | Click on all the buttons on the main menu screen and see if they work |
| Test Conditions | See Test Environment |
| Expected Results | The menu should load first and the buttons should work as expected on the menu. |
| Pass/Fail | Pass |

|  |  |
| --- | --- |
| Test Objective | Trying out every option when moving |
| Tested Requirements | MapReq, ItemReq, ControlReq, INTR4 |
| Test Description | When moving the character we should be able to:   * Move to target location * Move to target location and pick up item   + Can either Equip Item   + Or Use item * Move to load new level * Possibly move and attack the enemy |
| Test Conditions | See Test Environment |
| Expected Results | All of the possible options when moving should be working |
| Pass / Fail | Pass |

|  |  |
| --- | --- |
| Test Objective | Test out the sound of the game |
| Tested Requirements | SoundReq |
| Test Description | We will test the game to ensure that the sound is working, so doing every possible action that has sound related to it must be done |
| Test Conditions | See Test Environment |
| Expected Results | That sound will play whenever we do an action that has sound file related to it |
| Pass/Fail | Failed – Not yet implemented and therefore not yet tested. To be released in a future build. |

|  |  |
| --- | --- |
| Test Objective | Testing level tiles |
| Tested Requirements | INTR1, INTR2 |
| Test Description | We will test the level tiles itself to make sure it can properly display items and characters in them and that they can be removed/added to the tiles as well. |
| Test Conditions | See Test Environment |
| Expected Results | We expect the tiles to display what is on the tiles and we expect the characters (which includes enemies) to be able to move from a tile to another tile |
| Pass/Fail | Pass |

|  |  |
| --- | --- |
| Test Objective | Interaction between characters and objects |
| Tested Requirements | INTR3 |
| Test Description | Characters should have the ability to interact with other characters stats |
| Test Conditions | See Testing Environment |
| Expected Results | Characters should be able to remove or add onto other characters stats when interacting with them |
| Pass/Fail | Pass |

|  |  |
| --- | --- |
| Test Objective | Enemy type characters detecting user |
| Tested Requirements | INTR5 |
| Test Description | Enemies should engage the user when the user approaches them. Range for engaging depends on enemy type and will be tested at their max range accordingly. |
| Test Conditions | See Testing Environment |
| Expected Results | The enemy should approach the user and try to fight him. |
| Pass/Fail | Pass – With caveat – the enemy’s approach range is not wide – we may want to keep an eye on this just if the user finds the current range to be a good experience, but with the requirement it works correctly |

|  |  |
| --- | --- |
| Test Objective | Incorrect input does nothing |
| Tested Requirements | HF1 |
| Test Description | We will be testing to make sure that the user does no action on an incorrect input/touch or going to a tile that is out of bounds will result in the user doing nothing. |
| Test Conditions | See Testing Environment |
| Expected Results | When we do an incorrect input we expect nothing in return or nothing to happen when trying to go out of bounds. |
| Pass/Fail | Pass |

|  |  |
| --- | --- |
| Test Objective | Intuitive game design |
| Tested Requirements | HF2 |
| Test Description | We will get other users to test our game to make sure it is intuitive enough without a tutorial. |
| Test Conditions | See Testing Environment |
| Expected Results | Users should be able to play our game to completion without needing us to guide or help them. |
| Pass/Fail | Fail – We pass MOST requirements here. But we do not handle death in an intuitive enough way (no death dialog). So we will add this in a future build |

|  |  |
| --- | --- |
| Test Objective | Testing out the game |
| Tested Requirements | DARLVL |
| Test Description | We want to make sure the game is winnable, has enemies and usable items, so we will start the game, hit play and try to win. |
| Test Conditions | See Testing Environment |
| Expected Results | When we start the game we expect there to be a path to the next floor, enemies along the way and items. This doesn’t mean that every tile has items and enemies, but there should be enemies somewhere in the level. |
| Pass/Fail | Pass |

|  |  |
| --- | --- |
| Test Objective | Character movement efficiency |
| Tested Requirements | DARDIJK |
| Test Description | We will make sure the characters fighting the user will take the shortest path to him to fight him. |
| Test Conditions | See Testing Environment |
| Expected Results | We expect the enemy characters to move the main character in the least amount of moves needed to engage in combat with him. |
| Pass/Fail | Pass with caveat – the enemy find area is so small there is no real shortest path as it is already there |

|  |  |
| --- | --- |
| Test Objective | Make sure that the user is regaining health when not in combat |
| Tested Requirements | DARHEAL |
| Test Description | During run-time inspect health variables to make sure that health is slowly being increased.  Also make sure that a visual change is happening.  This will be tested by entering combat - losing something health - and then exiting combat.  If the variable and visual reflects health gain the requirement is fulfilled. |
| Test Conditions | See testing conditions - Specifically this will be run once on a loaded level - during the main gameplay |
| Expected Results | The user will regain health overtime.  The health bar will slowly grow in size which will also reflect a change internally that health is increasing. |
| Pass/Fail | Pass |

|  |  |
| --- | --- |
| Test Objective | Make sure that we use accessible tools (Xamarin & CocosSharp) to ease future development for potential future developers |
| Tested Requirements | XamCoReq |
| Test Description | Check that our project was developed using the accessible tools of Xamarin and CocosSharp.  Do this by making sure that the project can be emulated from source code using a system that only has Xamarin and CocosSharp installed. |
| Test Conditions | On each individual team member’s development environments |
| Expected Results | That each individual team member only utilized Xamarin and CocosSharp as part of their development environment |
| Pass/Fail | Pass |

|  |  |
| --- | --- |
| Test Objective | To make sure that our compiled project does not take up too much room on a user’s phone |
| Tested Requirements | SpaceReq |
| Test Description | Check to make sure that our compiled project will be less than 10MB when stored on an android device. |
| Test Conditions | See testing conditions - specifically this matters most on the actual hardware, not emulator so we will be checking on the real android application |
| Expected Results | That our total and compiled project is less than 10MB when loaded onto a real android device |
| Pass/Fail | Pass |

|  |  |
| --- | --- |
| Test Objective | Make sure that our application is not overloading the user’s phone |
| Tested Requirements | ProcessReq |
| Test Description | We will use provided emulator tools that monitor things such as processor use  -  we can use this to see how much of the processor our application is commanding.  For now we are saying 70% or less is successful, but the general idea is that the application is responsive without slowing down other phone features. |
| Test Conditions | See Testing Conditions - Using the emulator at a an Android API level of 23 and using HAXM.  This will be tested in the main gameplay state - with a level loaded. |
| Expected Results | The processor monitoring tools will display less than 70% when our game is being run in its main gameplay state (on a level). |
| Pass/Fail | Pass |

|  |  |
| --- | --- |
| Test Objective | This tests that our program can handle all possible cases of user interaction without running into bugs |
| Tested Requirements | RelReq, BugReq, AOS |
| Test Description | This is impossible to extensively test in our given time.  We will test this by utilizing the system for a large amount of time to simulate no more than 3 bugs per month.  This testing time is a sample of all potential conditions we can have.  So we will go through each test above (except for human factor tests) and log the amount of bugs we encounter (not concerning the type).  To try to expose our system to multiple bugs that may have multiple causes we will try to utilize the system at variable times and positions (see test conditions). |
| Test Conditions | See Test Environment + Variable times of day (morning, noon night) and variable phone positions (horizontal / vertical) |
| Expected Results | The desired result is that no run-time bugs are encountered.  Each above test yields its expected results.  We will accept no failures of any test during this testing time.  If there is a known bug it produces an output that we have already defined.  If the above tests run successfully in our test conditions we will consider this test to have completed successfully.  Anything else is considered failure. |
| Pass/Fail | Pass |

**Conclusion**

What are your final conclusions from the tests? Is more testing required? What additional tests should be performed? Is the software ready to be delivered? Do you have any maintenance concerns? What remains to be done for the software?

We believe that the game is in a usable state currently. It is not anything we would like to release to a playing audience. However, it is something that we would show to other developers to collaborate on or to users for beta testing. The main improvements we could make are not necessarily in bugs but rather in user experience. We would like to add a few items and depth to the game (such as better death handling) that make the game a more enjoyable experience for users. It would be great from now on to continue partnering with users to create a better and better user experience.

We would like to also update our source code. Currently at the end of testing it is not a large code base. However, it has some artifacts from our learning experience. Because this is a learning experience the longer we work on the project the better we become. So it will be important for us to always be going back and cleaning up our code. That is our main maintenance concern. The power of CocosSharp makes it easy to keep our code base small so this is not a major concern of ours.