**Sinister Transistor**

**Test Plan**

**COP 4331, Spring 2016**

Team Name: <your team name here>

Team Members:

* Greg Kelso
* Mark Boutwell
* Joel Gardyasz

Modification history:

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Who | Comment |
| v0.0 | 03/25/16 | Greg | Template |
| V0.5 | 03/25/16 | Joel | Introduction and Description of Test Environment |
| V0.6 | 3/25/16 | Mark | Stopping Criteria |
| V1.0 | 3/25/16 | Greg | Description of Individual Test Cases |

**Contents of this Document**

Introduction:  
    Overall Objective for Software Test Activity  
    Reference Documents

Description of Test Environment

Overall Stopping Criteria

Description of Individual Test Cases

**Introduction**

**Overall Objective for Software Test Activity**

The software testing effort should determine if all aspects of the game and code are running as expected. The game should be thoroughly tested in order to make an attempt to catch all bugs or things that could go wrong during a showcase of the game. Also, during the software testing, a short video of the game working should be made in order to use if the game does not work correctly during the final showcase. This will act as a fail-safe for if something goes wrong and can be used to show the game in a working state.

**Reference Documents**

* Concept of Operations
* Project Plan
* Software Requirement Specifications

**Description of Test Environment**

The hardware environment that will be used to test will primarily be the personal computers of the developers. The software environment that will be used to run the tests will be the Unity game engine in which the software will be made. Most testing will be done in the development stages of the design, while different aspects of the game are being developed. Hopefully, the game reaches a state to where it can be exported and sent to many people for secondary testing and debugging. In this case the test environment would be the computers of the people playing the game.

The test environment will be the same environment that the software operates in. The testers will most likely be the developers of the software, however, there may be opportunities to have other users test the system. Having actual users test the game is an excellent way to test for bugs or flaws in the game. Users who did not create the game have no prior cognition on how the game is supposed to operate, therefore, will play how they think they game should be played. This in itself can lead to the game being stressed in ways that the developers may not have thought of which is an excellent debugging tool.

**Stopping Criteria**

A test will run until all steps of the test have been completed or a fatal bug has occurred that makes the test impossible to continue. All non-fatal bugs that occur will be logged and the test will proceed. Once a test has ended, bug fix assignments will be given to group members with a priority placed on fatal bugs. Testing will be rerun once all of the fatal bugs have been fixed.

Testing will be done individually for each of the possible use cases. Once the system has fulfilled all of the requirements and there are no known errors that do not have a well-defined workaround, then testing will be completed and the system will be ready to deliver.

**Description of Individual Test Cases**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Objective** | **Test Description** | **Test Conditions** | **Expected Results** |
| Test collisions between game objects | There will be multiple types of game objects in the game, and each will have different behaviors to execute when colliding with other game objects. All possible collision combinations will be tested. | There will be only one “mode” that the game will run in, and all collisions will be tested there. There will also be tests to ensure collisions work the same across different rooms. | Collisions behave as expected. The three likely behaviors that exist include walls that do not allow the two game objects to exist in the same space, enemies that damage the player, and items that disappear and modify a particular value for the player. |
| Test menu and GUI functionality | At any point in the game, the user should be able to access a menu and be able to quit the game. The game environment should pause when this occurs. All options from the menu screen should work as intended. | This well be tested during different phases of the game, and from different menu options. | The menu should always appear when the function is called, and all menu options should behave as intended. |
| Test health, damage, and item functionality. | Ensure that the health is deducted by the correct amount, the death mechanics work as intended for both players and enemies, and ensure that all items operate as intended. | These is only one game mode that these operations are available in, and they will be tested in all combinations possible. | Damage should mirror what the code describes, and there should be no bugs with item pickup and use. |