Senior Project Defense

By Derrick Kamphaus

*Web Browser -* Educational Video Game in Unity

Contents

[Statement of Purpose 1](#_Toc37858383)

[Background 1](#_Toc37858384)

[Research 2](#_Toc37858385)

[Hardware & Software 3](#_Toc37858386)

[Software used: 3](#_Toc37858387)

[Hardware: 3](#_Toc37858388)

[Project Screenshots and Explanation 4](#_Toc37858389)

[Main Menu 7](#_Toc37858390)

[Level Select 8](#_Toc37858391)

[Levels 9](#_Toc37858392)

[Unity 11](#_Toc37858393)

[Introduction: 12](#_Toc37858394)

[References: 12](#_Toc37858395)

[Test Items: 12](#_Toc37858396)

[Features to be Tested: 12](#_Toc37858397)

[Features Not to Be Tested: 13](#_Toc37858398)

[Approach 14](#_Toc37858399)

[Item Pass/Fail Criteria 14](#_Toc37858400)

[Test Deliverables 16](#_Toc37858401)

[Test environment 16](#_Toc37858402)

[Schedule 17](#_Toc37858403)

[Training Needs: 17](#_Toc37858404)

[Responsibilities: 17](#_Toc37858405)

[Risks: 17](#_Toc37858406)

[Assumptions and Dependencies: 17](#_Toc37858407)

[Usability Test-Plan 17](#_Toc37858408)

[Introduction: 18](#_Toc37858409)

[References: 18](#_Toc37858410)

[Test environment 18](#_Toc37858411)

[Materials Required: 18](#_Toc37858412)

[Playability Features to be Tested: 18](#_Toc37858413)

[Process of Testing 19](#_Toc37858414)

[Initial Setup 20](#_Toc37858415)

[Instructions for testing: 20](#_Toc37858416)

[Requirements 23](#_Toc37858417)

[Purpose 23](#_Toc37858418)

[Features 23](#_Toc37858419)

[Functional Requirements 23](#_Toc37858420)

[Performance Requirements 24](#_Toc37858421)

[Usability and Humanity Requirements 24](#_Toc37858422)

[Test Result Document 26](#_Toc37858423)

[1.) Levels 26](#_Toc37858424)

[Results: 26](#_Toc37858425)

[2.) Save and Load Progress 26](#_Toc37858426)

[Results: 26](#_Toc37858427)

[3.) Player Profile (Stretch Goal) 26](#_Toc37858428)

[Results: 26](#_Toc37858429)

[4.) Menu Display and Function 26](#_Toc37858430)

[Results: 26](#_Toc37858431)

[5.) Dialog System Display 27](#_Toc37858432)

[6.) Overworld to link levels 27](#_Toc37858433)

[8.) Resolution Uniformity 27](#_Toc37858434)

[9.) Cross Platform Compatibility 28](#_Toc37858435)

[10.) Conducive to learning 28](#_Toc37858436)

[Results: 28](#_Toc37858437)

[11.) Ease of User for target Audience 28](#_Toc37858438)

[Results: 28](#_Toc37858439)

[12.) Content Interchangeability 28](#_Toc37858440)

[13.) Editor for Levels (Stretch Goal) 28](#_Toc37858441)

[Results: 28](#_Toc37858442)

[14.) Localization Capability 28](#_Toc37858443)

[Results: 28](#_Toc37858444)

[Results of Player Testing 29](#_Toc37858445)

[Players 29](#_Toc37858446)

[Feedback 29](#_Toc37858447)

[Summary of Testing 29](#_Toc37858448)

[Challenges of Development 30](#_Toc37858449)

[Future Enhancements 32](#_Toc37858450)

[PowerPoint Slides 33](#_Toc37858451)

[References 36](#_Toc37858452)

# Statement of Purpose

The project I designed is a 2D educational video game. I created the game with Unity and created the scripts using C# programming language. The game, called *Web Browser*, is about a spider looking for collectables to put in his web. He gets through to the end by answering questions about his current environment. The project’s goal was to create a game that was both engaging and educational for children. I wanted to encourage learning about science and building problem solving skills. The project helped me pursue my personal goal of becoming a video game developer. Another reason for making a video game, was to show my ability to solve problems in a unique way.

# Background

Everyone has different styles of learning that work for them, such as visual or auditory. Some kids have trouble connecting in a classroom environment or reading books may not be engaging for them. This could cause children to miss out on something they may actually be interested in. It is important to teach using all instructional types so students of all learning styles can understand. This is a great benefit of educational video games as games combine all of the learning styles into an interactive platform. According to a case study, students that played educational games outperformed other students and were more likely to participate in class. (Ascione, n.d.)

*Web Browser* focuses on science and problem-solving skills. Players learn about physics and biology among other things. An educational game allows children to spend their spare time productively learning and developing. Children often play video games on smartphones and consoles. Since they are immersed in technology and entertainment, a form of entertainment that doubles as an educational experience allows children to become more engaged in learning and have doors opened to new interests they might otherwise miss. I have always had a passion for video games, and I want to show others how they can be helpful.

# Research

During the course of this project, I conducted considerable research to learn how to use Unity. Some of the research consisted of Google searches to figure out how to accomplish specific tasks, but I performed most of the research within Unity’s official documentation (Unity, 2019) as well as users on forums having similar questions or issues. For some portions of the project, I watched video tutorials on how to accomplish some of the tasks. I got a lot of helpful information, particularly from Brakeys’ YouTube channel (Thirslund, 2019). I learned how to create and animate a dialog system that I used as a Question and Answer system and a Save system. Through the course of the project, I learned a lot about Unity itself as well as generally working with other third-party libraries and APIs using a programming language, in this case C#. It was very unique as I was accustomed to programming in C# with Visual Studio using Windows API.

# Hardware & Software

## Software used:

* Unity 20192020.3.1 (Latest build) – Note version at start was current build at time, and upgrades were done as time went on. Unity acted as the main IDE for developing the game, managing assets and piecing together with scripting
* Visual Studio Professional 2019- IDE used for scripting all C# code that was utilized by the game created in Unity. Visual Studio was used as the primary means of debugging code related issues.
* Inno Setup Compiler- Installer creation program, Unity has no built-in package or installation software. This was used to create an installer for the game. It still can run from the standalone exe along with required files, without installation
* Signtool – a self-signing tool included in Windows 10 SDK. This allows the application to be signed and reduces the number of warnings to run the application.

## Hardware:

* Game was developed and tested on a Desktop PC with intel i5 8thgen CPU, 16 GB ram m3 SSD hard drive and Nvidia GTX 760 Graphics card.
* Some testing was performed on MSI GF63 laptop with added SSD.

# Project Screenshots and Explanation

**Main Menu**

When the application loads, the player is brought to the main menu (see Figure 1). From this menu the player can start the game or enter the Settings. The Settings Menu (see Figure 2) has the option to change the volume of the music and to reset the game data. These settings are remembered when the application is ended. Clicking “Reset Game Data” will cause a confirmation to appear, warning the player that their previous save data will be deleted. After clicking yes, the data will be removed, and starting the game will create a new save file. From the main menu, clicking on the start button will bring the player to the overworld.

**Overworld**

Upon entering the Overworld (see Figure 3), the player will see the available levels. When there is no existing data, the player will see only the garden level while the others will be hidden. Once a level is completed, the level is scored and given a star rating, which appears in the Overworld. The object that was collected at the end of the level will also be visible (see Figure 4). The scoring system works by calculating the number of questions in the level multiplied by the points per question variable. I currently have this set to 100 points, but it can be set on a per level basis. For example, in the garden level a perfect score is 500 points because there are five questions, each worth a maximum of 100 points. The score is then adjusted for incorrect attempts. Each incorrect attempt subtracts 25 points from the score. This value can also be set per level. Only three attempts will be counted, so that no question can give 0 or negative points. For instance, a player who got every question right on their second attempt would receive a score of 375 on level 1. The star rating is calculated with a ratio of player score vs total possible score. A score of 90% or higher awards three stars, at least 60% awards two stars, and less than 60% awards only one star. Levels can be repeated to get higher scores and more stars. All the data for the levels is managed by a gamemanager object. It holds the save file and all the stats for the levels. When the player loads a level by clicking on the button, the level is loaded, the gamemanager is transferred into the level, and then the Overworld is closed.

**Level**

When a player enters a level, their character spawns at the beginning of the platform, which is set by the level editor (see Figure 7) The player controls a cartoon spider. It moves left and right with player input, which varies depending on the gaming platform. It is not bound specifically to keys, rather each platform defines commands which the Unity interprets as input. For the demo provided, a PC uses the W, A, S, and D keys or the arrow keys to move, and uses the spacebar to jump. The player moves in the entered direction by the factor of movement, which is set in the game editor. The player’s movement is inhibited by “collision”, utilized by Unity’s 2D physics system. Objects that have colliders prevent the player from moving past the object. If the character is on top of a collider, it is considered grounded and can move freely left and right. The spider has two colliders that track its collision points. The bottom collider is a circle which makes contact with the ground. The top of the spider is a rectangle which makes contact with solid objects behind, in front, or above the player. When the character jumps, upward force is applied. This force factor is also set by the editor. Gravity can also be applied to the character and can be edited in Unity. While the player is in the air, it is not considered to be grounded until the bottom collider makes contact with another object. The player is able to double jump, as long as required jumping conditions are met. If the top collider makes contact with any object while in midair, the jump will end causing the player to fall. This effect also prevents the player from climbing up a wall using the jump command. The player will not be able to jump again until the bottom collider makes contact with an object, thus grounding the player. Similarly, if the player falls, then they are no longer grounded. However, they can jump if they haven’t already made a jump attempt so long as the jump conditions are still met.

Insects with question marks are Question Givers (see Figure 5). The question appears once they are within assertion range. This value can be set per Question Giver in the editor. The player must either choose from the four potential answers or leave without answering the question by moving out of the Giver’s range. These questions are loaded from a text file and can be easily changed. When the player returns to the Giver, the question will be displayed again. If the player answers incorrectly, feedback in the dialog box will tell the player they were wrong. The player gets the answer wrong twice, the feedback suggests that they go back and look for a hint in the level. If there is a hint associated with the question, the hint object will generate an exclamation mark over it to make it easier to locate. Hint mechanics work similarly to the questions, in that entering the range of a hint will show the hint, then exiting the range causes the hint to disappear (see Figure 6). When the player answers correctly, the dialog will tell the player. After a few seconds, the question dialog goes away, the question mark disappears, and the Question Giver collider goes away, which allows the player to move on. After the question is answered, the score gained from the question is added to the level total.

When the player reaches the end of the level, they find an object to collect (see Figure 7). Gathering the collectable initiates the end of level sequence. Each level has its own levelmanager object which works similarly to the gamemanager. The two work together to generate save data. The player’s score is used to calculate award stars. Both the stars and the player’s score are passed into the levelmanager’s level data, and the level is marked complete. A sub routine then searches through the level data in the gamemanager to see if new levels required the completed level as a qualification. If found, then the prerequisite flag is set to true for that level. The data for the current level is transferred to the gamemanager object, which holds the information for all levels. The game manager then writes all the data to the disk. After data is saved, the overworld level is loaded into memory. The gamemanager object is moved from the current level into the overworld, then the current level is unloaded. The overworld displays the score and star rating for the completed level, and the next level will appear as unlocked and ready to play. Because the game data is saved to disk if the application is reloaded, the data will be retained.

**Unity**

Unity acts as the game creator and level editor. Different portions of the editor can be seen in Figure 8. The left menu shows all the objects within a scene, whether they are visible or not. The center of the screen shows the visual layout of all the objects. The Asset manager is below the scene view. All game assets are available here and can be dragged onto the scene to be added as an instance. This is how obstacles and new objects are added to the level. The other important part to Unity is the inspector. Here any object can have new components added to them. This allows things such as physics colliders, sprites, and animators to be added to the object. This is also how the scripts are added to the object. Scripts consist of a C# file that contains a class definition. Anytime a game object has a script attached to it, that becomes an instance of the C# class object. When an object has a script attached to it (see Figure 9) any public fields of the class are automatically populated in the inspector. This allows various settings to be set in the editor. This allows references to other assets to be set as well as setting important values, such as the player’s movement speed and scoring data as previously mentioned.

## Main Menu

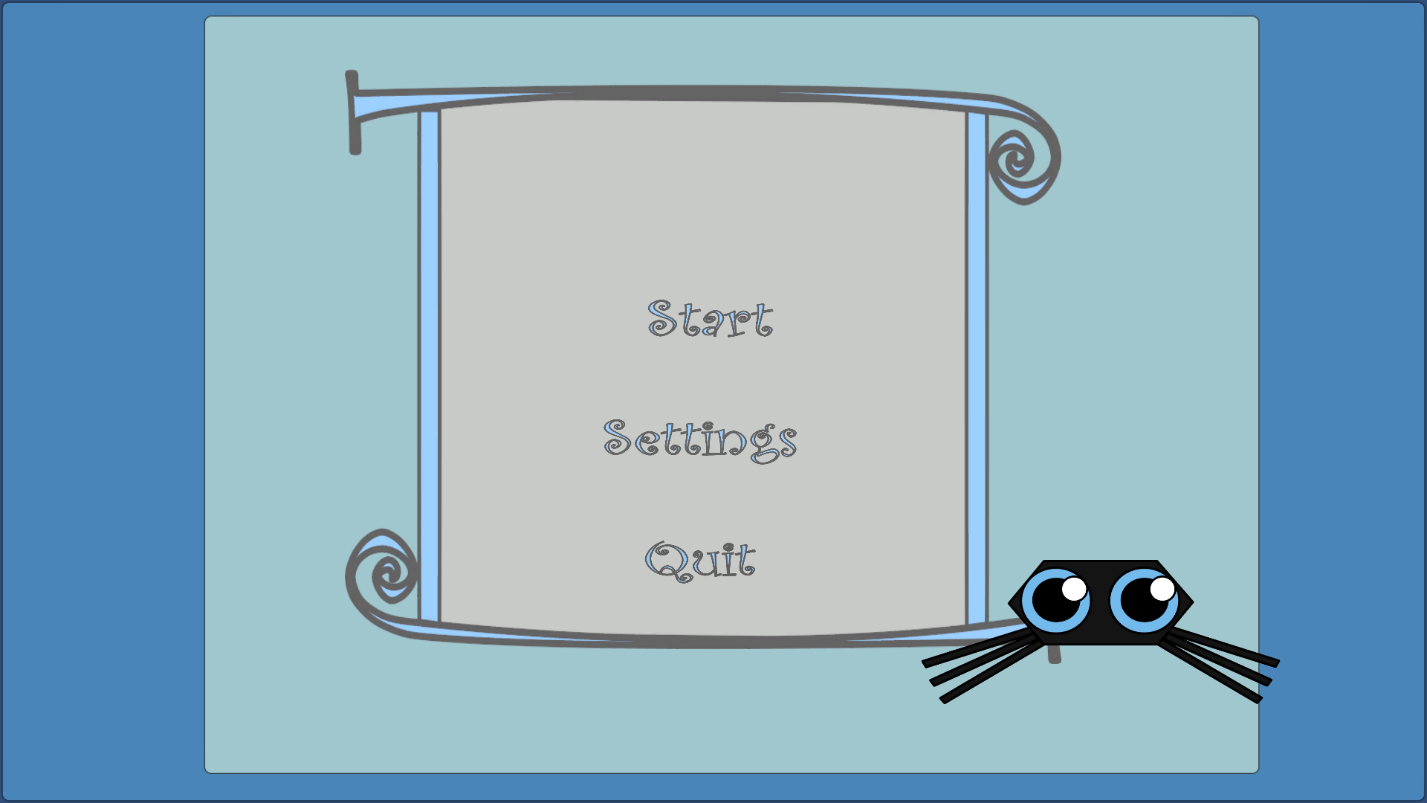


Figure - Main menu screen when game starts up

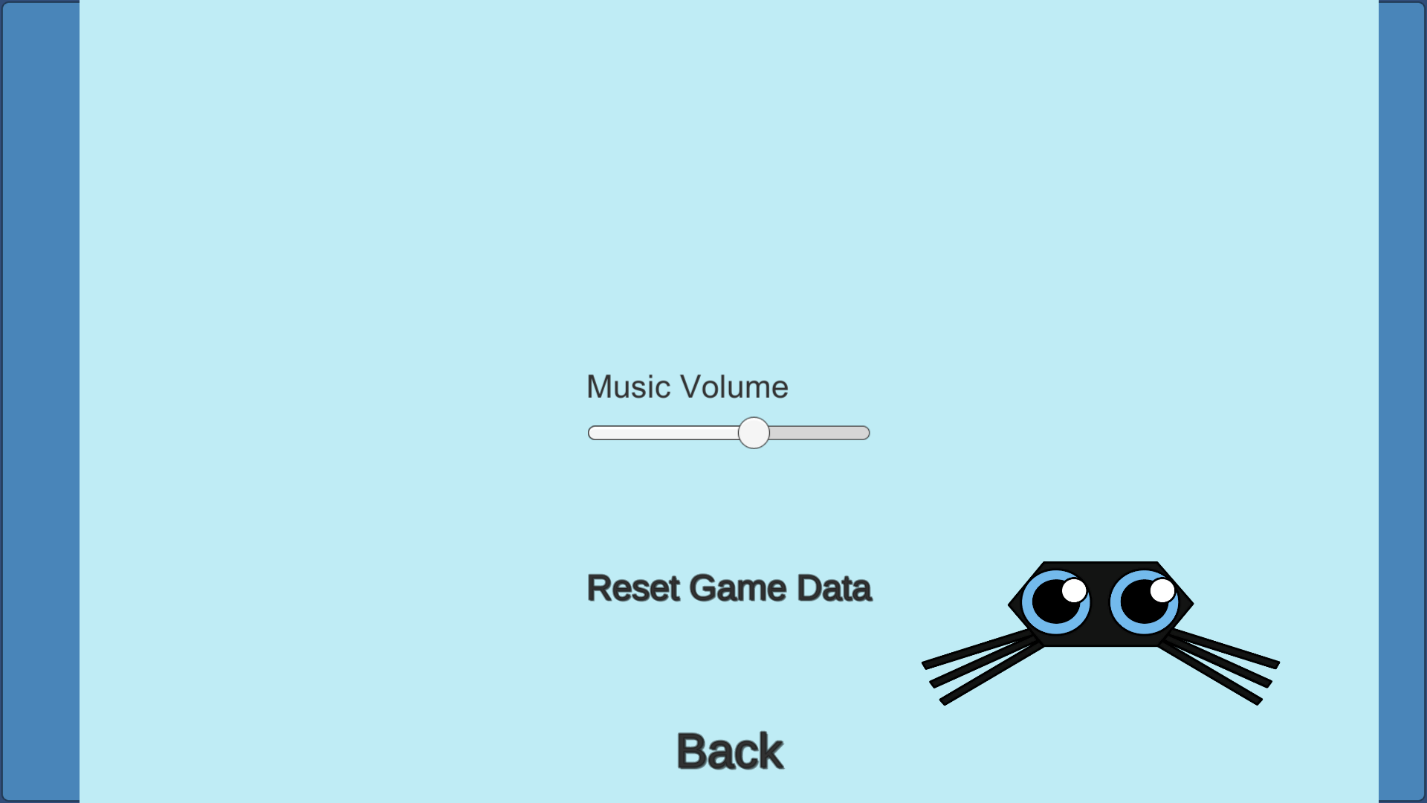


Figure - Settings menu from main menu

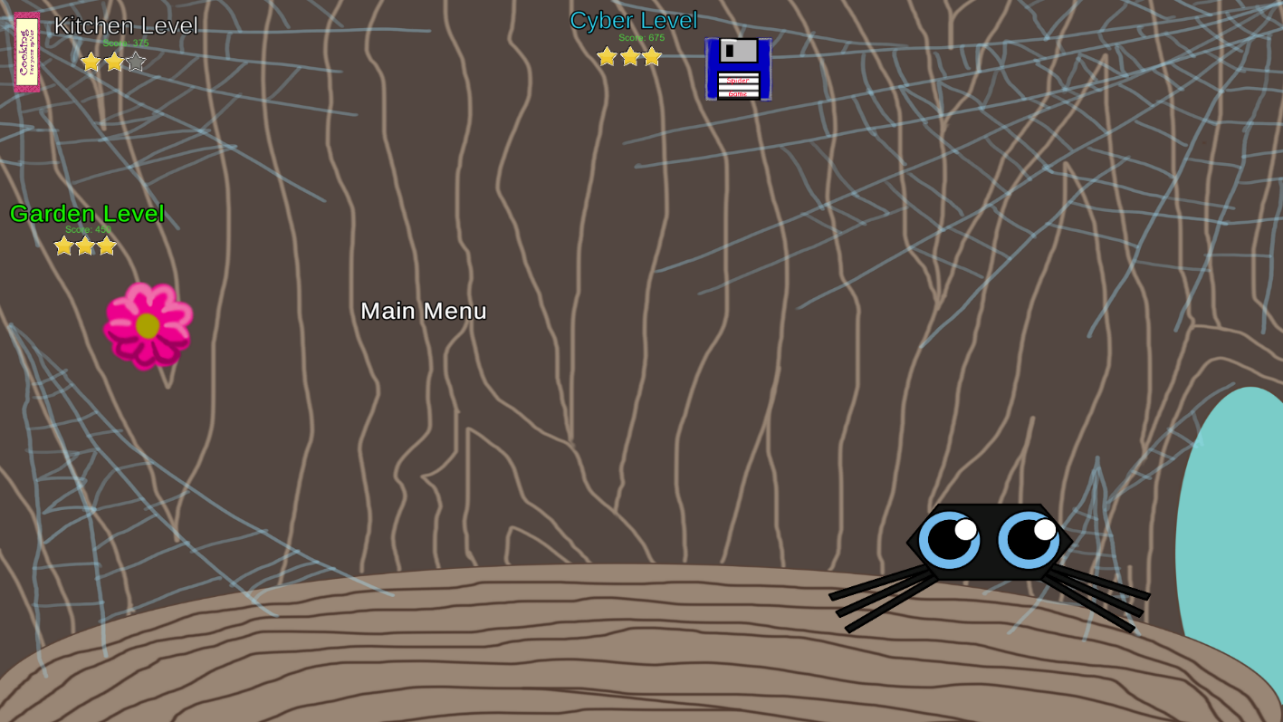
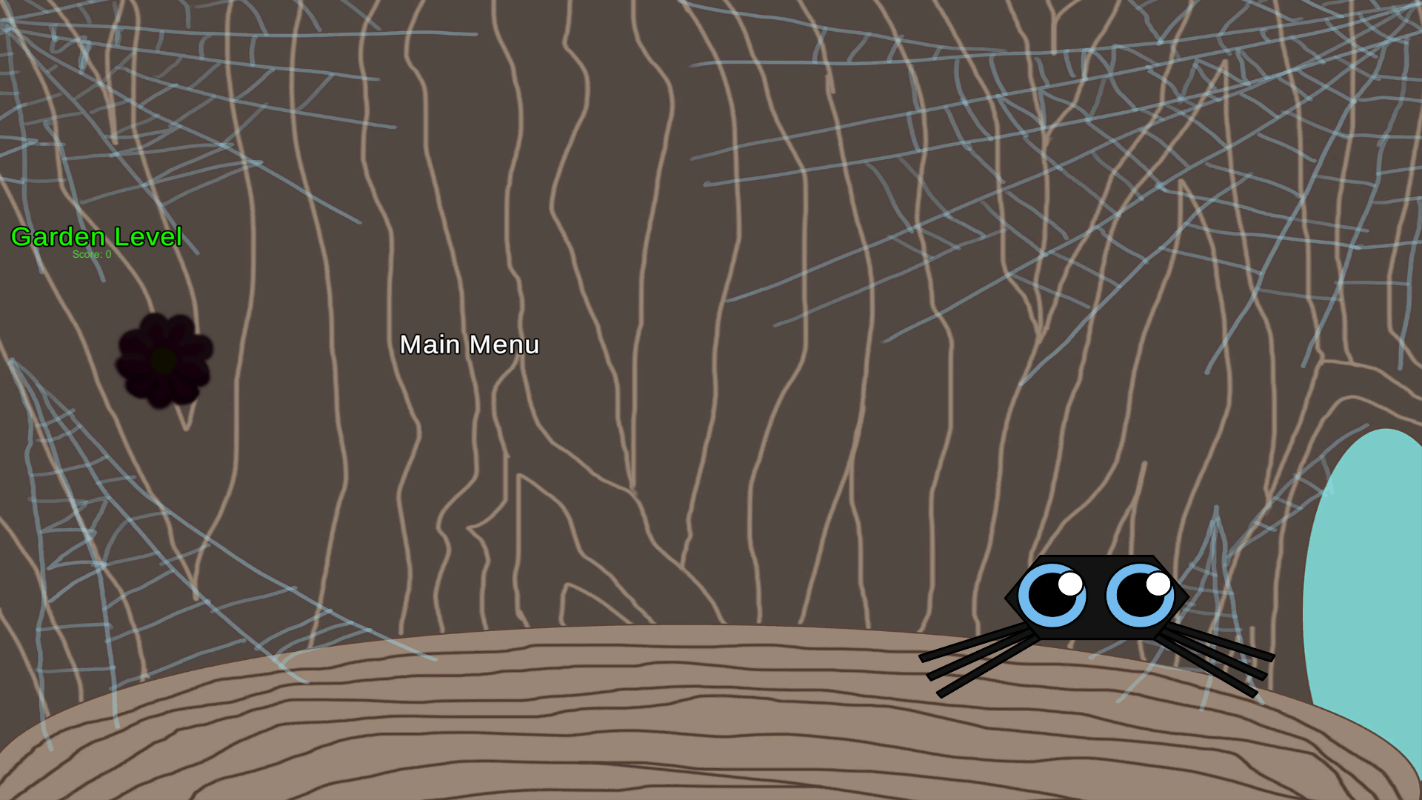
Level Select

Figure - Overworld where player selects levels. Initial view without completing all levels.

Figure - Overworld after all levels have been completed.

## Levels

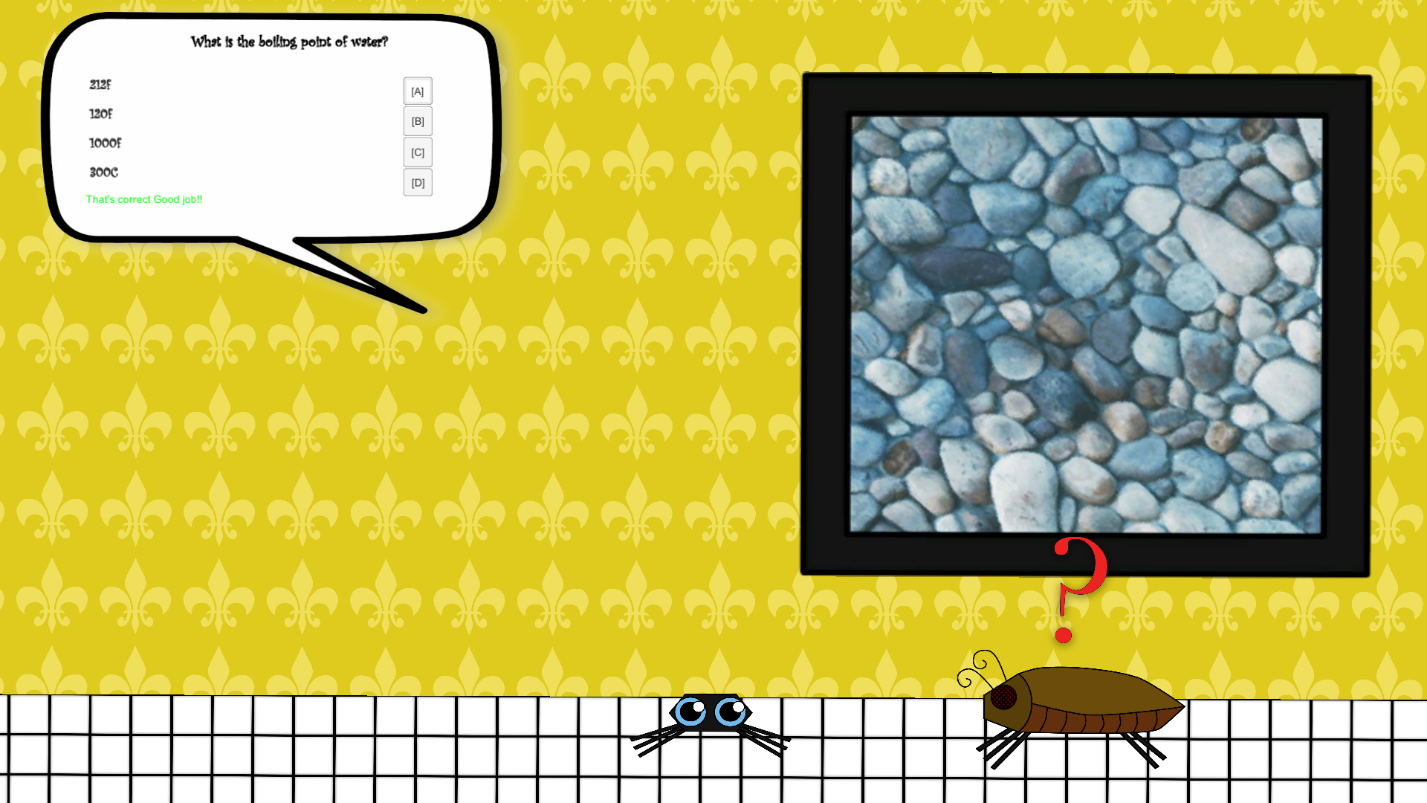


Figure 5 – Sample of level 2 showing an example of a question.

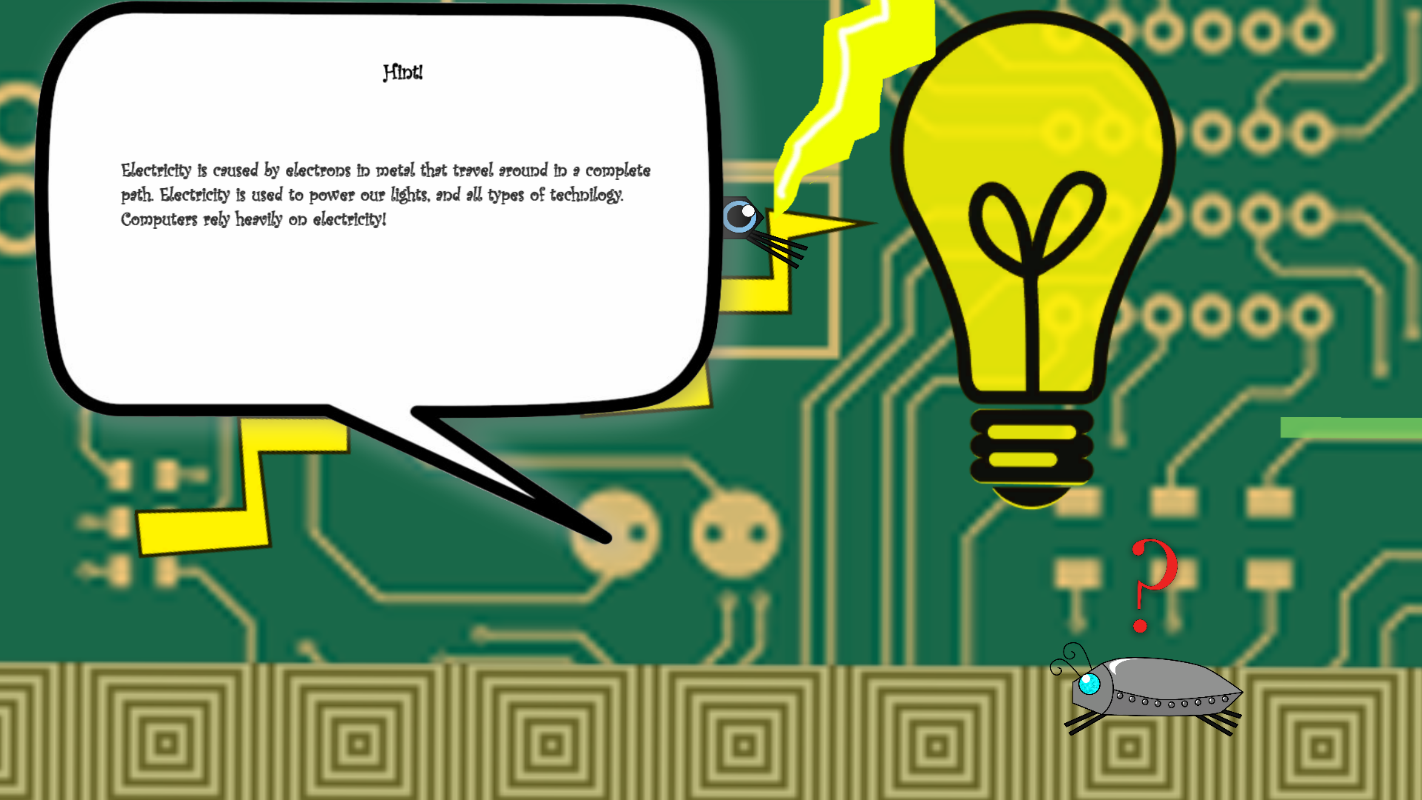


Figure - Sample of level 3 showing an example of a hint

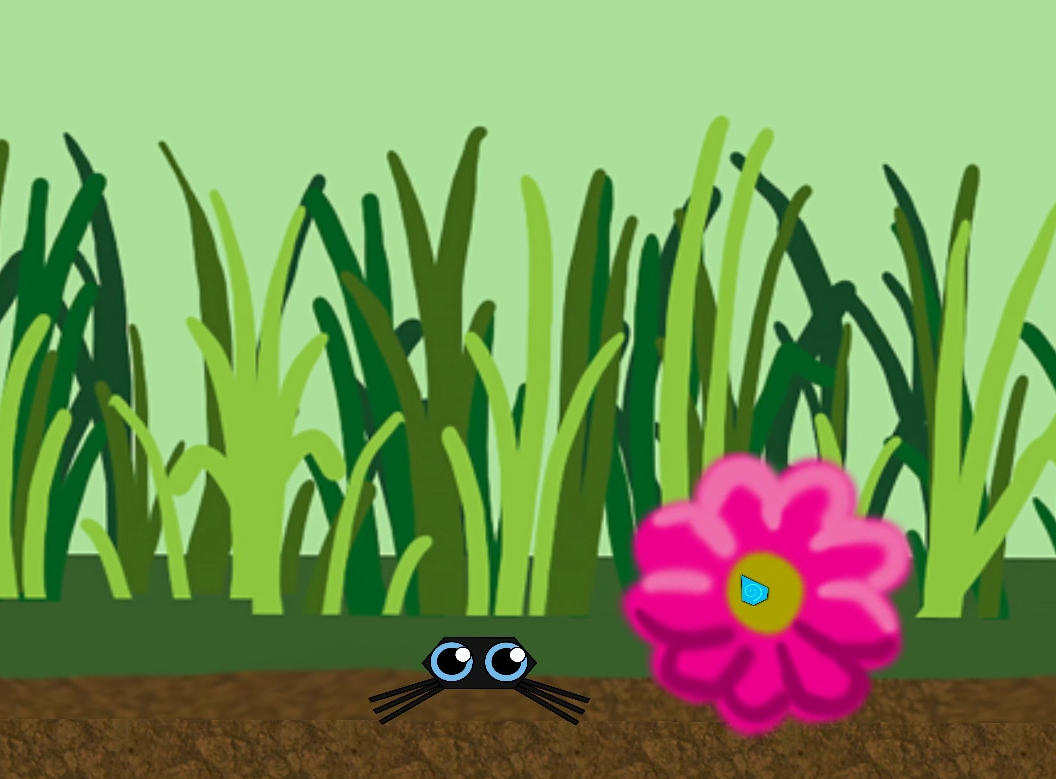


Figure 7 – Sample of end of the level when player collects the object it ends the level

## Unity

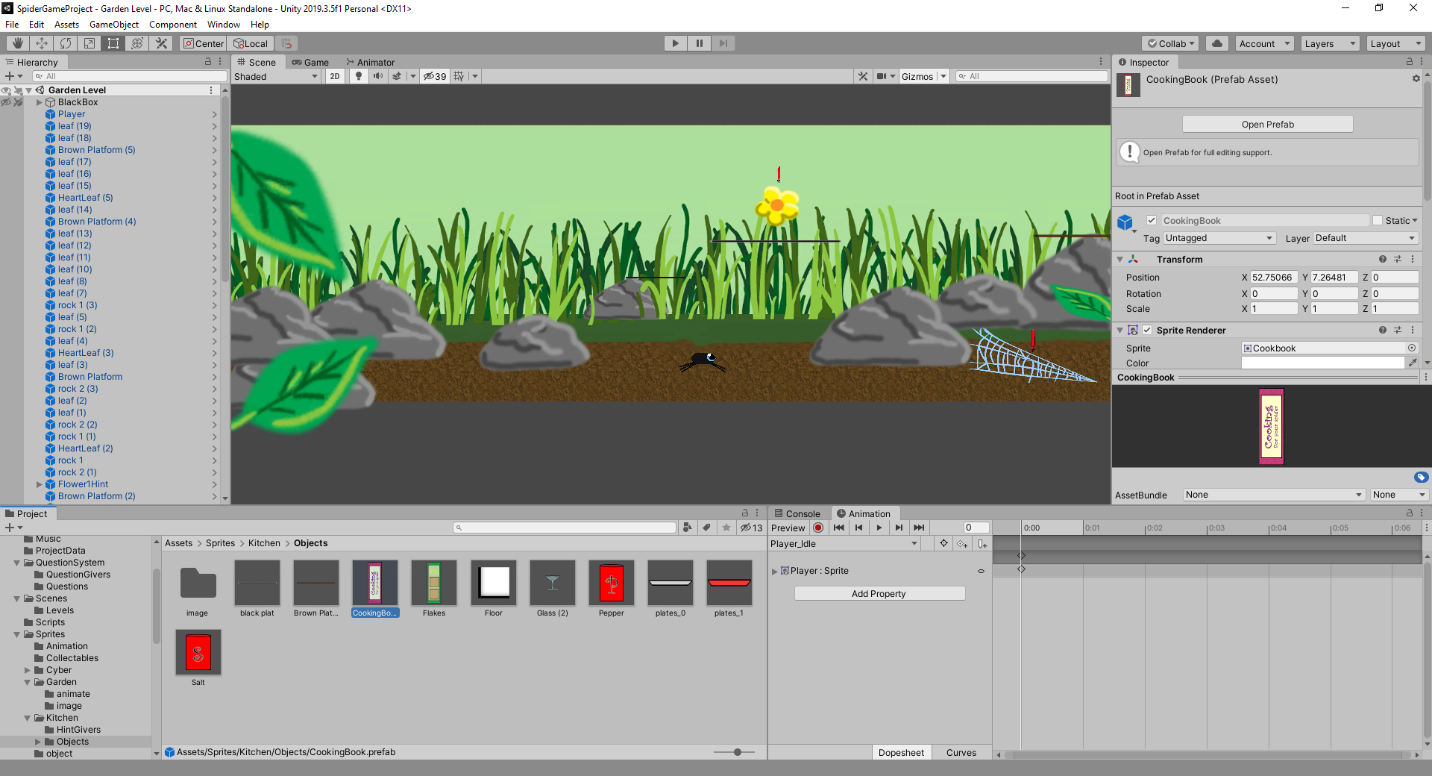


Figure 8- Unity IDE shown editing a level

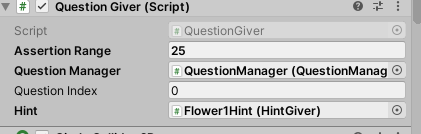


Figure 9 - Objects with attached scripts allow public fields to be adjusted from the editor.Master Test-Plan

## Introduction:

This test overview is designed to ensure the game is working properly as well accomplishes its goals to the intended target audience.

Goals & Objectives:

* Fully functional educational game that runs on platforms accessible to students.
* Goals: provide educational value to students who play it, to be engaging and educational
* Must be extensible so others can provide educational content

## References:

* [Requirements](#_Requirements)
* [Usability Test-Plan](#_Usability_Test-Plan)

## Test Items:

* Unity Version 2019.2.3f1 Personal – IDE for creating the game as well as the engine to run the tests for the game.
* Microsoft Visual Studio Professional 2019 Version 16.2.3 – Programming IDE for C# scripts used in unity engine. Used to test code specific items and debugging.

## Features to be Tested:

### Functionality Features:

* Level Accessible and Functioning:
  + Accessibility of the levels and ability to gain access to all intended areas, and stay within intended boundaries.
  + Questions being able to be answered correctly and correctly display on the screen.
  + All content desired content fits within screen boundary and fit to the display.
* Save and Load Progress
  + Progress of players is can be saved.
  + Score, collectables, and questions answered on current level can be saved.
  + Loading game will restore all progress and previous data will be displayed.
* Player Profile (Stretch Goal)
  + Can create new profiles, and select existing from list
  + All player profiles will still appear when game is reloaded
  + All progress is saved with the respective profile and can be loaded when game is reloaded and profile is selected.
* Menu Display:
  + Menus display during start of game and during pause. Players can edit settings and start game. Correct items display when buttons pressed.
* Dialog System Display
  + Must properly appear when character enters in range of an object or obstacle and questions will go away when answered.
* Overworld to link levels
  + Over World will link levels and spiders’ home. Show items collected. When enough items collected next level is unlocked.

### Performance Features:

* Program Stability
  + Game does not crash or freeze and has good performance on most common devices.
* Resolution Uniformity
  + Game elements fit properly on various resolutions and screen sizes. Text and UI remain readable.
* Cross Platform Compatibility
  + Properly function in the same compacity for all platforms tested.
  + Platforms to be tested:
    - Windows 8 and later
    - Android OS.
  + Test of Display Systems and Resolution hold across targeted platforms.

### Usability:

* Conducive to learning
  + The environment provided by the game and the delivery of the educational content aid
* Ease of user for target audience
  + Children need to be able to understand how to play.
  + Gameplay not over complicated.
* Content Interchangeability
  + All questions and puzzles should be easily changed by a third party to provide educational content.
  + Stretch Goal:
    - An easy to use editor for third party to create content for educators.
* Localization Capability
  + Have the ability to swap out the questions and game dialog with a different language to be extensible in the future.
* Editor for Levels – Stretch Goal
  + A functioning tool to allow third party to edit the levels and provide educational content

## Features Not to Be Tested:

* The Educational value of the questions themselves and content specific
  + The goal of the game is to provide a platform for a learning environment that education providers can upload content for students to use to learn while playing the game. The focus is how the game can facilitate the educational setting but not the educational content included in the game demo.
* Quality of the questions
  + The intention is not how well students can answer the questions themselves, as the intention is for others to supply the content, but how the questions themselves are presented and interfaced for the best delivery of the educational content.
* Stretch Goals that uncompleted
  + Stretch Goals listed are desired features, but will not be tested if they are not complete and implemented within the specified timeline.

## Approach

* Testing will be broken into two categories.
  + Functionality and Performance- Features that are part of the core functionality of the game, that must work the intended way to ensure the game is usable and facilitates the intended goals.
  + User Playability – Features that are more subjective the target users and quality of life that may need to be changed or adjusted base off of feedback from the players.
    - See attached [Usability Test](#_Usability_Test-Plan) Documentation for more details
* Functionality and Performance Method of testing
  + Functionality will be tested on a pass, fail Criteria where specific parameters must be met to determine if they meet the requirements.
  + Automation:
    - Automation ranges from auto input, and premade testing levels that will run specific tasks automatically.
    - For functionality Features, automation will be used on case by case basis as there are many variables to account for and is subjective.
    - Performance will utilize automation to ensure uniformity in testing.
      * Stability: Will utilize stress testing by simulating high levels of input and content on game.
      * Cross Platform: Create Automated mode that provides preset input that should behave uniformly across platforms.
  + Features that are independent of device screen size or platform will be tested on a standard device.
* User Playability
  + Testing will be conducted in a controlled environment among target audience.
  + Process of testing will be uniform amongst each playthrough.
  + Documentation will be used with a standard list of questions and order of testing.
  + See attached [Usability Test](#_Usability_Test-Plan) Documentation for full feedback requirements, forms, and other details.

## Item Pass/Fail Criteria

1. Levels
   1. Pass Criteria: Must be able to bring character from beginning to end on each level.
   2. Fail Criteria: If player gets stuck, falls out of bounds, or otherwise unable to access the entire level.
   3. Recurrence: Test will be performed for each level, and criteria must be met for each case.
2. Save and load Progress
   1. Pass Criteria: Player saves their progress. When the game is closed and reopened all saved data will return. Player score, progress in level, and previous level scores and overall progress load.
   2. Fail Criteria: Any progress is not loaded when game is reopened and save file is loaded. File fails to load. Collectables lost, or current score not displayed.
3. Player Profile (Stretch Goal)
   1. Pass Criteria: New profile can be created. When game is closed, the profile remains. Profile is displayed from a list and can be selected. Saves made to profile will save with respective profile and load the correct data. At least 3 profiles are created and all previous criteria hold to be true.
   2. Fail Criteria: Profile is not created. When the game closes, profile is not present. Any profile whether newly created or previously created does not display in list. Any profile does not load the progress saved into it. Any profile loads data from another profile.
4. Menu Display and Function
   1. Pass Criteria: All items on menu do as it says. They display properly and all buttons work when pressed. Players are able to load the correct level that the menu indicates.
   2. Fail Criteria: Menu does not lead to intended location; any item does not respond when clicked.
5. Dialog System Display
   1. Pass Criteria: Questions will be answered with the correct answer then disappear opening the way forward. Wrong answers do not answer the question. All text is legible and appears on screens on all levels.
   2. Fail Criteria: Questions dialog does not disappear after being answered, or range has been left. Player unable to move after question is answered. Dialog does not fit in designated area, or text is not legible or visible. An answer cannot be clicked or does not work.
6. Overworld to link levels
   1. Pass Criteria: Levels must properly transition when they are selected. Criteria is met to unlock level, and provide proper access.
   2. Fail Criteria: Wrong level is loaded, level transitions without meeting requirement, level doesn’t load at all, unable to get to overworld upon finishing a level.
7. Program Stability
   1. Pass Criteria: No crashing, freezing, or lagging at any point of gameplay, to be tested across various performance machines.
   2. Fail Criteria: Game crashes, computer crashes (consistently and repeatable as a direct result of the game) or game lags.
   3. Recursion: Test will be performed on multiple devices, of varying capabilities low and high end. Must yield passing results on all devices.
8. Resolution Uniformity
   1. Pass Criteria: All previous tests involving Display to include Dialog System, Menu Display Hold true across all target resolutions and screen sizes. Content fully fits on screen, and text and UI remain legible.
   2. Fail Criteria: Any content does not fit fully on the screen that is meant to. U/I or text does not fully fit on screen, or cutoff. Any previous test involving display fails.
   3. Recursion: Test will be performed on multiple screen sizes, and set to multiple resolutions. Must yield passing results on all devices, all previous tests on display will be made again where applicable, and must hold true.
9. Cross Platform Compatibility (Requirement Not Met)
   1. Each Platform tested will have the same pass-fail criteria
   2. Pass Criteria: The game displays properly and controls function the same across all platforms
   3. Fail: The platform behaves differently than intended, game does not function the same as the standard test machine. Any resolution or other display requirement is not met
10. Conducive to learning
    1. Pass Criteria: Gameplay does not distract from the content, players are able to stay focused on the task and access the educational content. Players need to feel significant reward for playing.
    2. Fail: More than half of the players are distracted, do not want to play the game, do not feel like they learned anything.
11. Ease of user for target audience
    1. Pass Criteria: While playing there is minimal interaction with children to explain how the game works. It is straightforward for players and they are able to move towards the goal and not get lost or confused.
    2. Fail Criteria: More than half of the players requires significant explanation or assistance in order to play the game. Players are lost or confused.
12. Content Interchangeability
    1. Pass Criteria: Questions that are in text file load properly and display where desired. Puzzles can be modified.
    2. Fail Criteria: The data does not load at all from the text file, questions do not appear in the correct place, the answers to not match the questions, the features are unable to be changed without recompiling the game.
13. Editor for Levels (Stretch Goal)
    1. Pass Criteria: User is able to edit levels of the game and provide questions and content. When the game is loaded, levels edited by the user will be playable and load.
    2. Fail Criteria: Editor is not functioning properly, and crashes. Files created in editor cannot be loaded into the game. The editor freezes or crashes.
14. Description: Localization Capability (Requirement Not Met)
    1. Pass Criteria: Alternate text files are able to be swapped out and selected from the settings menu and work correctly. It not need be a different language but all text provided to game should be interchangeable and potentially be in a different language.
    2. Fail Criteria: Alternate text file does not load when selected or it displays when the primary text should be showing.

## Test Deliverables

* Test Plan
* Test Case
* Test Reports
* User Feedback Forms

## Test environment

* Standard Testing Computer: Laptop MSI Intel i5 CPU, 8GB RAM, 500GB SSD, OS: Windows 10, GPU: Nvidia GTX 1050.
* Most Recent build of project will be run on laptop

## Schedule

* Jan 15th test levels accessibility
  + Resolve issues if present by Jan 31st
* Jan 31st Test menu accessibility
  + Resolve issues if present by Feb10th
* Feb 10th test multiplatform
  + Resolve issues as necessary by March1st
* March 2nd – 10th Begin player testing trials with volunteers.
  + May need to evaluate if additional data is necessary. If feedback dictates a feature must be changed, have done by 15th then recommence testing with the new features
  + If necessary additional trials after 15th for changed features
* April 1st Compile all findings from testing
* April 10th All errors should be fixed and game should be about completely finished.

## Training Needs:

* Users may need to be trained on basic controls of the game during testing. If more training is needed during testing, in-game instructions will be considered to be added.

## Responsibilities:

* Developer: Derrick Kamphaus will be responsible for implementing the testing, and distributing for player testing.
* Players: responsible for providing feedback during game play. Fill out a questionnaire after playing through.

## Risks:

* During play testing players may feel like they have to keep playing or testing the entire time
  + Inform players at any point they are able to end testing at any point if they feel the need to.

## Assumptions and Dependencies:

* All features have been completed to full extend prior to testing
* Group play trials depend on having subjects to cooperate

Approvals:

* Project Advisor: Dr. Sean Hayes:

Signature

# Usability Test-Plan

Date

## Introduction:

This test plan is designed to supplement the Master Test Plan and provide additional details to the execution of the Usability Tests. It is intended to measure the goals mentioned in the master test plan and provide the requirements for implementing the testing.

Goals & Objectives of Usability Test:

* Determine if the requirements of the Master Test Plan for Usability are met
* Provide a means for user feedback to improve on certain design aspects of the game.
* Measure the success of the meeting the goals and provide a means to get data and feedback to reach goals.

## References:

* [Master Test-Plan](#_Testplan)
* [Requirements](#_Requirements)

## Test environment

* Standard Testing Computer: Laptop MSI Intel i5 CPU, 8GB RAM, 500GB SSD, OS: Windows 10, GPU: Nvidia GTX 1050.
* Most Recent build of project will be run on laptop

## Materials Required:

### Provided to Testers:

* Standard Testing Computer provided to tester, with latest stable build.
* One Player Feedback Form
* Pencil
* Mouse

### Provided to Administrator:

* Administrator Testing Instructions Sheet
* Administrator will be in charge of setting up Testers Materials and ensure they have access to it.

## Playability Features to be Tested:

* Conducive to learning
  + The environment provided by the game and the delivery of the educational content aid
* Ease of user for target audience
  + Children need to be able to understand how to play.
  + Gameplay not over complicated.

### Feedback Required:

* Players should provide feedback to verify that the requirements of the game are being met, and that is appropriate for the targeted audience.
* How the player felt playing the game
* How difficult was it to understand the game mechanics and controls?
* Did the objectives seem straightforward?
* Was the game interesting
* Did the player learn anything?

## Process of Testing

### Personnel Involved:

* Target Tester
* Administrator
* Developer if separate from Administrator

### Administrator:

* Either developer or an adult (over the age of 18)
* Understands how to run the game and able to answer questions about it.

### Target Tester:

* Student between the ages of 8 and 12

### Sample Size:

* At least 5 to 10
* More is preferable

### Requirements:

* Parental permission required for testing subject
* Student should have basic reading and writing skills at appropriate grade level.
* Privacy of test subject must be maintained. No names or contact information will be recorded from the test taker. Only information about the user’s Age, Grade, and Gender will be recorded for trend analysis purposes.

### Provisions:

* At any point the subject can choose to terminate testing.
* Subject must be made aware of this at the beginning of the testing.

### Sequence of Testing:

* Testing will be done with one subject at a time.
* Subject will be informed that they may request to leave the testing at any time if they feel the need to.
* Instructions for testing will be read to subject
* Provide time to answer any questions to the subject
* Provide subject materials to include feedback form and pencil and the testing computer.
* Subject will begin testing once the administrator runs the game.
* Player will play through the game naturally, and may ask questions if unable to understand what to do.
* Once player has finished playing through the game, player will fill out feedback form.

Attached on the following pages are the printable instructions for the Testing Administrator and survey forms for players.

Administrator Testing Instructions

## Initial Setup

* Before tester arrives ensure the testing materials are available. There should be available at a table the player feedback form, a pencil and the provided testing laptop.
* Turn on the testing Laptop
* Login to the Tester Profile on the laptop, there is no password.
* On Desktop will be a shortcut to *Web Browser*. Double click this will launch the game.
* Game is ready, and setup is complete testing with the subject will be ready.

## Instructions for testing:

* Bring in the test subject have them sit in front of the material
* Instruct them to not do anything until instructed to.
* Read the follow to them

“Thank you for coming here today to test the game. This test is completely voluntary, please know that if at any point you feel the need to leave for any reason you can. We will be testing a game called Web Browser it was created by Derrick Kamphaus. We will begin shortly; the test will include you playing through the game followed by a quick survey. When you are playing through at first try to do as much as you can on your own. If you feel like you cannot figure out what to do please ask for help. The goal is to see how easy it is to understand how to play the game. Do you have any questions”?

* After questions are answered, inform the subject they may begin when they are ready. Tell them to start the game.
* When the player as played through a few levels either after completion or they no longer want to play have them answer the survey.
* Provisions for survey: If the subject is younger, or has trouble with writing you can read the questions on the survey to them and write their responses.

*Web Browser* Game Survey

Thank you for taking the time to play **Web Browser**. Please answer the following questions to the best of your ability. Be honest, there are no right or wrong answers. If you need help please ask the grownup for help.

Please write Your

**Age**: \_\_\_\_\_\_\_\_\_\_\_\_\_ **Grade**: \_\_\_\_\_\_\_\_\_\_\_

For the questions below circle the answer that applies.

1. When you first started the game, how easy was it to get started playing?

Very Easy Easy Somewhat Easy Hard Very Hard

Please write what made it feel easy or hard to get started. If it was difficult what would make it easier?

­­

1. After you started playing, was it clear how to get to the next level from inside the spider’s home?

Very Clear Clear Sort of Confusing Confusing Very Confusing

Please write what made it feel clear or unclear to get the level started. If it was confusing to you what would help to make more sense?

­­

1. When you were playing the levels, was it easy to control the spider and get to where you wanted to go?

Very Easy Easy Somewhat Easy Hard Very Hard

Please write what made it difficult to control if you had trouble making the spider move. What could make it easier to control him?

1. When you were answering the questions in the game, was it easy to select the answer you wanted to the question?

Very Easy Easy Somewhat Easy Hard Very Hard

Please state what made it easy or hard to choose the answer you wanted to pick. If you did not know an answer did you feel like you were able figure out the answer with any hints in the level?

1. Did you learn anything while playing the game? Would you have fun at school if you played this game in class?
2. Overall did you like this game?

Loved it Liked it Not Sure about it Disliked it Really Disliked it

Please write why you liked or didn’t like the game. Be specific about what you liked and didn’t like.

1. Do you have any suggestions on what could make the game better?

# Requirements

## Purpose

* Educational video game aimed for children
* Provide educational environment that is engaging for children
* Aids learning with different learning styles

## Features

* Player can move around left and right and jump
* Player answers questions to move forward
* Player collects items to progress further
* Menu System
  + Save Progress
  + Adjust Display
* Overworld to connect levels, and show current progress

## Functional Requirements

1. Level Accessible and Functioning
   * Description: All levels must be fully accessible to intended areas and prohibit access to areas not intended by design. Questions presented in levels should be answerable with the correct answer and display properly on the screen. All the content will display properly in the bounds of the screen.
   * Fit Criterion: Player cannot with reasonable attempt leave boundaries, able to complete the level and gain access to all intended areas. Every level should yield the same results with testing.
2. Save and load Progress
   * Description: Player progress must be able to be saved at reasonable intervals. Once data is saved after game is closed, player should be able to restore progress upon reloading the game, and selecting the appropriate menu items.
   * Fit Criterion: Player saves progress through game interface. Player score, progress through level to include, questions answered, items collected, is saved to a file. Player closes game, then reopens it. When player loads the game, all data saved will return to the game. Inside level data is loaded if the level was not complete. In the overworld menu, all scores on previous levels as well as collectables will also be loaded.
3. Player Profiles (Stretch Goal)
   * Description: Player profiles in Menu system. Player creates new profile or selects theirs form list. The profile will hold all their data which they can save and load.
   * Fit Criterion: New Profile is created and added to list. When game is closed profile will remain. Profile is selected from list.
   * Dependencies: Requirement 2- Save and Load Progress. All saved data and progress associated with respective profile will behave as required by above requirement.
4. Menu Display and Function

* Description: The main menu should be accessible during the start of the game and during pause. The menu will allow players to edit settings and start the game.
  + Fit Criterion: All items on menu do as it says. They display properly and all buttons work when pressed. Players are able to load the correct level that the menu indicates.

1. Dialog System Display
   * Description: Must properly appear when character enters in range of an object or obstacle. Questions will go away when answered.
   * Fit Criterion: Questions will be answered with the correct answer then disappear opening the way forward. Wrong answers do not answer the question. All text is legible and appears on screens on all levels.
2. Overworld to link levels
   * Description: Over World will link levels and spiders’ home. Show items collected. When enough items collected next level is unlocked.
   * Fit Criterion: Levels must properly transition when they are selected. Criteria is met to unlock level, and provide proper access.

## Performance Requirements

1. Program Stability
   * Description: Game does not crash or freeze and has good performance on most common devices
   * Fit Criterion: No crashing or freezing at any point of gameplay, to be tested across various performance machines.
2. Resolution Uniformity
   * Description: Game elements properly fit on various resolutions and screen sizes.
   * Fit Criterion: Levels, Menus, and all UI uniformly displays in intended position across all screen sizes and resolutions. More content may be displayed on larger resolutions, but content will be scaled, and still appear clear and text and other UI is legible.
   * Dependencies: All above requirements involving display still hold across various resolutions and screen sizes.
3. Cross Platform Compatibility
   * Description: Game runs on standard devices Windows 8 and up, and Android OS.
   * Fit Criterion: Game functions basically the same on all intended platforms. All Display appears uniformly on each target platform. No other Functional requirement is not met on any of the platforms.
   * Dependencies: All performance and functional requirements regarding display should be met on other platforms.

## Usability and Humanity Requirements

1. Conducive to learning
   * Description: The environment provided by the game and the delivery of the educational content will aid in learning for the players. Focus is not on the value of the content, but how the game as a whole is able to deliver the content in an educational manner.
   * Fit Criterion: Gameplay does not distract from the content, players are able to stay focused on the task and access the educational content. Players need to feel significant reward for playing.
2. Ease of user for target audience
   * Description: Children need to be able to understand how to play. Gameplay cannot be so complex and cumbersome that a child would not be able to understand easily.
   * Fit Criterion: While playing there should be minimal interaction with children to explain how the game works. It should be intuitive and straightforward, players are able to move towards the goal and not get lost or confused.
3. Content Interchangeability
   * Description: All questions and other puzzles should be easily changed by a third party to provide educational content.
   * Fit Criterion: Questions that are in text file load properly and display where desired. Other content can easily be changed without any compiling the binary files.
4. Editor for Levels (Stretch Goal):

* Description: A easy to use editor for third party to create content for educators.
  + Fit Criterion: Editor is able to edit levels and content. Runs as executable requiring no recompiling binaries.

1. Description: Localization Capability
   * Description: Have the ability to swap out the questions and game dialog with a different language to be extensible in the future.
   * Fit Criterion: Text files can be swapped out. It not need be a different language but all text provided to game should be interchangeable.
   * Dependencies: Requirement 11: Content Interchangeability.

# Test Result Document

## 1.) Levels

## Results:

Performed 4/10/2020

All criteria met. Completed all 3 levels, all areas were accessible, no areas that were meant to be off limits were accessible, levels were able to be completed. No fail criteria met.

## 2.) Save and Load Progress

## Results:

Performed: 3/17/2020

All pass criteria met. Completed each level and closed out of the game. Then loaded the game back up and the file remained. Deleted the save file, and then did one level again with new score. Both times the game properly saved. No fail criteria met.

## 3.) Player Profile (Stretch Goal)

## Results:

Stretch goal feature was not implemented.

## 4.) Menu Display and Function

## Results:

Performed 3/20/20

Went through the all the buttons on the main menu. They all worked, properly. Went to start the game, the game began. Once in the overworld, clicked on the menu button, which worked. Then went into each level. Paused the game then found an issue. Pause menu contains a save button that does nothing. All other buttons display and are pressed correctly. Levels are loaded correctly. Fail Criteria met, item Save does nothing when clicked. All other menu items worked properly.

### Corrective Action:

Save button, is an old feature that was removed, but left in place unintentionally. Remove save button from menu retest to ensure label is gone.

### Retest:

Performed 3/20/20

Pass Criteria met; corrective action successful.

## 5.) Dialog System Display

### Results:

Performed 3/20/2020

Answered all the questions on each level, and verified that the full question was visible. Used every possible answer button, which worked. All pass criteria were met, no difficultly reading text or seeing full dialog box. Test Passed.

### Retest:

Required to be verified for each resolution / platform

## 6.) Overworld to link levels

### Results:

Performed 3/17/2020

Attempted to load the second level, which did not load as expected due to not yet being unlocked. Loaded the first level successfully. Finished the first level, then verified the score appeared as expected as well as end of level item appearing in overworld. Then went to second level which was unlocked, did the same, verified the level had the proper items and scored unlocked afterwards. Test passed.

### Retest:

Will be required if any additional levels are added.

### 7.) Program Stability

### Results:

Tested the game on various hardware. Ran through each level checking for any lag with audio or visuals, as well as controls not interacting. No freezing on unresponsiveness occurred. Test passed

## 8.) Resolution Uniformity

### Recursive Test:

Tested resolutions: 1280 x 1024, 1366 x 768, 1600 x 900, 1920 x 1080.

Test involved checking all menus and levels, also ensuring the dialogs for questions appeared correctly.

### Results:

Dialog requirement not met, slightly too far left, full box not on screen

### Corrective Action:

Move dialog to the right, ensure displays correctly on larger displays.

## 9.) Cross Platform Compatibility

Requirement not met due to time constraints, and issues with Unity Multiplatform build failing

## 10.) Conducive to learning

## Results:

Test passed. Game was given to several players. They played through the game and filled out a survey. Based on response, players generally felt like they learned or could learn from the game.

## 11.) Ease of User for target Audience

## Results:

Test Passed. Game was given to several players, in the target age group players understood how to play the game and were able to complete it without too much difficulty.

## 12.) Content Interchangeability

### Results:

Tested multiple levels, verifying that the text files loaded properly with the question givers. All text displayed properly. When a text file was changed, without recompiling the game, the questions would change to reflect the file changes as expected. Test passed.

## 13.) Editor for Levels (Stretch Goal)

## Results:

Stretch goal feature was not implemented.

## 14.) Localization Capability

## Results:

Requirement not met due to time constraints

# Results of Player Testing

## Players

The player group consisted of family members and their children. The original intent was to include several players at the targeted age group around 8-12. Due to the current quarantine, the original intention was not able to be met. Players included children as well as adults. Though adults were not intended as audience, their input proved to be insightful and useful.

## Feedback

Initial feedback from the players from the first playthrough indicated that the games controls were difficult. After making some changes to the movement and jump mechanics, the game play improved greatly. One player suggested adding double jumping, something I had not intended to implement. After adding the double jump feature and fixing other issues with the mechanics, on a second playthrough the players all felt the game was easier to control and navigate. One of the children playing the game had several suggestions about alternate levels as well as making the questions mathematical. This feedback may be considered for future improvements.

# Summary of Testing

Testing was a vital portion of the project. It identified several issues with the project. The functionality testing identified several discrepancies with the visuals displaying correctly, and the presence of nonfunctioning menu items. Usability testing identified additional functionality issues that were overlooked or issues that occurred upon making other changes. The player testing identified that the game is on track to be educational, has adequate difficulty but is not too difficult for the target audience. The player feedback was very valuable. Several suggestions were taken, and after the players did a second run through, they all felt the changes were improvements.

# Challenges of Development

During the course of this project, I encountered plenty of challenges, many of which resulted directly from a lack of familiarity with a new development environment, Unity. While Unity is a very powerful tool for creating video games and other simulations, it can be challenging at first. Some of my biggest challenges were figuring out how to interact with the Unity environment within my C# code.

Non inclusive list:

* Initially there were many issues related to proper character animation. The player would not display the correct animation, which required some reworking and understanding of how Unity dealt with assets.
* Early on I had many issues with dealing with the camera, resulting in having to resort to online resources to learn how best to get the camera to stay onscreen (Unity, 2019).
* There were many quirks with the levels that I had to figure out. Often times it required use of breakpoints with Visual Studio, which proved an invaluable resource. This allowed me to troubleshoot why certain subs weren’t being called when expected as well as unexpected values.
* One of the biggest challenges was the Save system. I had to create various level and game handler scripts that handled the current level and the overworld. I wrote a save system using the help of [Brackys](https://www.youtube.com/watch?v=XOjd_qU2Ido)’ YouTube channel to learn out to use Save (Thirslund, 2019). When I first tested the system, it gave me lots of issues. Testing revealed the Save system itself was flawless but there were issues transferring data to and from scenes in Unity which the Save system heavily relied on. It was a very complicated process, but I fixed it with the help of Unity documentation from Google search (Unity, 2019).
* The Questions system was also a great challenge. After initially setting it up, there were several issues with scoring and questions being skipped or not properly working. After considerable investigation and tweaking, the issue was discovered to be from two causes. When a player entered a question within range, but while quickly moving out of range, the question would sometimes go away. Another issue occurred when the spider approached a question and the question was displayed already answered. The first issue was resolved by increasing the range required to make the question go away to a greater value than that required to activate it. The second issue was determined to be actually caused by the GUI buttons for the questions being unintentionally selected by the players movement controls and confirmed by the Jump key. The questions were being inadvertently answered, very quickly. After removing selection of buttons by the keys, this issue was resolved. After I fixed both issues , the score values returned to where they were expected to be, rather than being extremely high. The issues appeared to have been answering the questions multiple times and giving credit.
* The jumping and movement system proved to be very challenging especially during player testing. It was unsmooth overall, and there were glitches allowing the player to climb up a wall and potentially skip questions or exit the level boundaries. I rectified this issue by changing the player collision detector from one single rectangle to a circle at the base, and rectangle at the head. The control system would detect the players base circle collider being off an object and mark the player as no longer grounded. When the player is no longer on the ground, if their top collider hits a wall, it stops movement and any current jumps while restricting further jumping until the spider is once again grounded. After resolving this issue, players testing the game felt control had greatly improved, as a side effect of changing the bottom collider into a circle, movement smoothed out considerably.

# Future Enhancements

* Adding many more levels to the game
* Creating an easy to use editor that works with the post-compiled game to allow educators and other content creators to easily design the level with no knowledge of Unity or programming to be at the most educational and accurate for the questions and answers
* Puzzles to solve and other challenges in addition to questions during the levels
* Implementing localization, allowing multiple languages
* Multiple platforms - would like to get game to work on android and IOS tablets and Mac.
* Multiple profiles: multiple players could have profiles and multiple save files.

# PowerPoint Slides

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# References

Ascione, L. (n.d.). Retrieved from ESchoolNews: https://www.eschoolnews.com/2017/10/19/educational-games-teachers/2/

Thirslund, A. (2019). *Brakeys*. Retrieved from You Tube: https://www.youtube.com/user/Brackeys/

Unity. (2019). *Unity User Manual*. Retrieved from Unity: https://docs.unity3d.com/Manual/index.html