Final Report

Software Engineering
Professor Do Young Park

Team Members:
Emiz Intriago, Gurkiran Kaur, Wardah Aziz, Miranda Figueras
Team Leader:
Andrew Stephens

Table of Contents

Video Presentation: https://youtu.be/Bnr04HVrNYM

1. Introduction

The implementation of “Into the Woods” has progressed steadily throughout development. Early in the semester, we discussed the specifics of the game our team would strive to accomplish. We decided to model some aspects of our game on traditional 2D games, like “Mario Brothers”, while implementing some of the features of “Ori and the Blind Forest”. We wanted to add the adventure aspect to our 2D platformer by including the necessity of the player to search the levels in an attempt to find multiple Key collectibles to progress into the next level. After deciding on the major details of the game, we decided to divide our work to best suit our group members' prior skills. Three members of our group focused mainly on writing the assigned documentation and creating the associated diagrams, these members were Kiran, Wardah, and Miranda. One member of our group, Emiz, focused mainly on the game aesthetics creating the background images for the menu and levels, character designs, sprites, platforms, and editing documentation. Our team leader, Andrew, took on the task of completing the source code for our project as well as heading our group, editing documentation, and creating diagrams.

In our proposal, we suggested that we would be able to create a multi-level, 2D platform game, which we have since developed. We have also included extra features, such as giving the user the ability to change the game’s frame rate, window type, and enable or disable the audio. We have also been able to implement a save game feature so that if a player completes a level, they can save their game progress, close the game, and later reopen it to continue with the next level. Currently, all of the main functions of our game have been implemented and are functional.

Our game opens to the Main Menu screen, from where the player can select to start a new game from the New Game menu, continue from a previously started game from the Continue Game menu, or change the settings from the Options menu. When starting a new game, Level 1 begins and the user can openly roam through the game level, looking for the Key collectibles and attempting to avoid the Spike deterrents. We have been able to include additional features we had not originally planned for in order to improve the appearance of our game, including animation for the avatar when killed during gameplay as well as sound on specific trigger actions, an elastic camera, full scalability across many screen sizes, and much more.

2. Design changes

2.1 Controls Settings

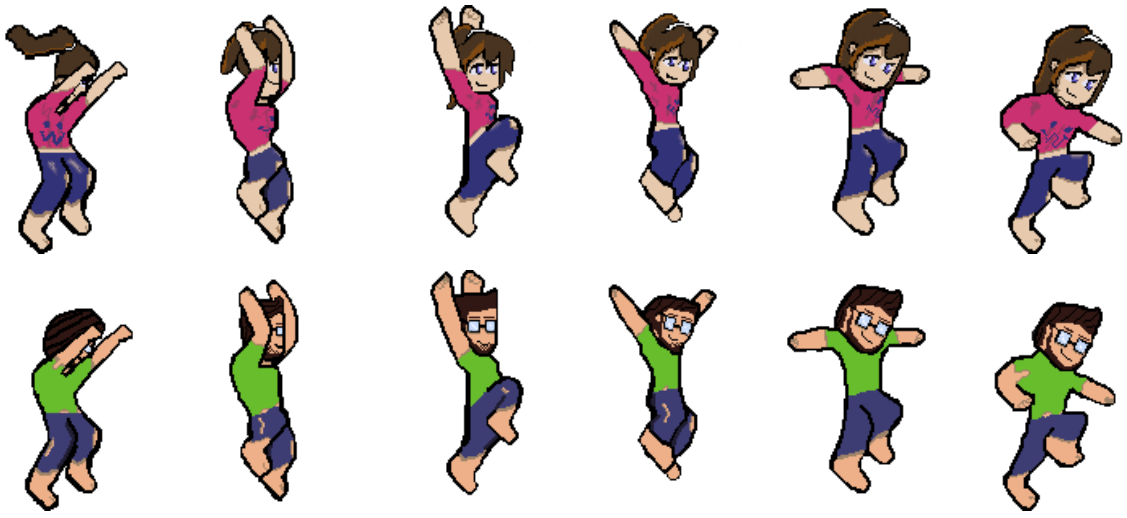
We have been able to implement all of the important features we described in our proposal document. While writing the Analysis Document, we had decided that we wanted the user to have the ability to change the key controls used to control the avatar, but were not able to implement this due to time constraints. Our team decided to instead set the key controls ‘A’ to move left, ‘D’ to move right, and the ‘Space Bar’ is used to make the avatar jump. By not making the key controls adjustable, the game is simpler for the user as the Help page can explain the key controls.

2.2 Sound Design

We were able to add multiple additional features to our game than what we had originally envisioned. We were able to add sounds to the game, which we believe makes the game more enjoyable. There is a sound associated with the steps that the avatar takes, as well as a sound for picking up a key and opening the door at the finale of a level.

2.3 Sprite Animations

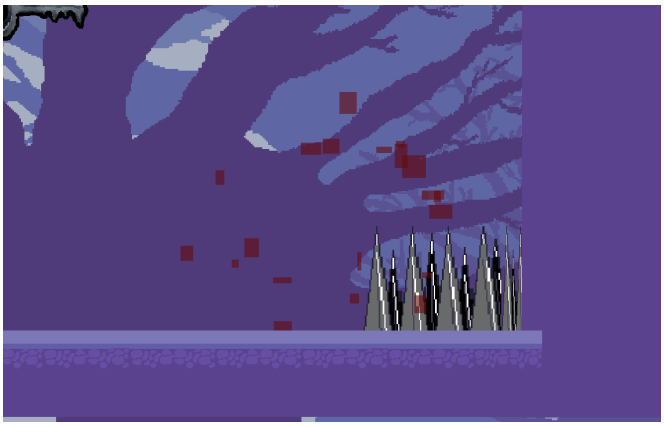
We also included character avatars which are designed with sprite sheets so that they look more realistic and mimic natural movements. We drew sprite sheets to represent the avatar idle, running, and jumping.



Sprite sheets for jumping animation for Melynn and Teo.

2.4 Death Animation and Transition

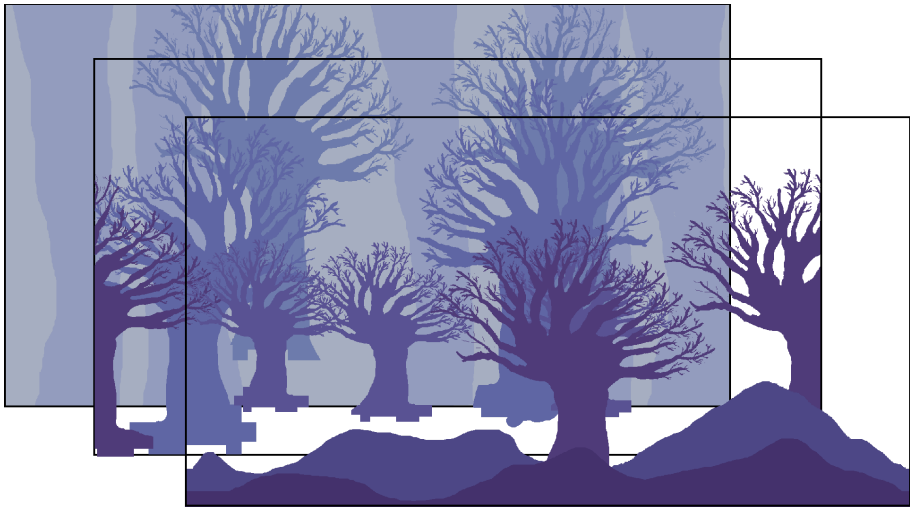
We were able to implement animation for the character Avatar death which happens when the avatar is in contact with the spikes. If the Avatar makes contact with the Spikes, the Avatar dies and the avatar will explode into small squares before the game level resets.



Example of the avatar death animation.

2.5 Parallax Background

As an additional effect, we decided to use a Parallax Background effect in order to create more visual stimulation for the player. This is executed by layering multiple backgrounds on top of each other and having each layer move independently at differing rates based on their depth, which gives a 3-dimensional effect that makes the background more interesting.



Example of one Parallax Background composite.

2.6 Audio Settings

While we were able to implement all of the major features of our game, we made some choices in exactly how those features worked. As such, we were able to give the user the option to have the sound play or not but decided not to implement a sliding bar to adjust the volume of the game sounds due to the lack of MP3 audio support from proprietary Java 17 libraries.

2.7 In-Game Overlays

We were also able to implement the in-game overlay, where the Map Overlay can be viewed. While it is implemented, it currently shows numbers on the map to represent where the Keys are located in the Level. We had wanted the Map Overlay to show a key in the location of the Key collectibles. The Map Overlay is also not scaled properly around the avatar. These features are not fully implemented solely due to time constraints from unforeseen additional assignments taking up that time.

3. Lessons learned

3.1 Software development

Programming was carried out utilizing the IDE, Intellij IDEA. Throughout the semester, our sole developer, Andrew, worked tirelessly on the software to finalize all the code on time. A few problems arose during production which were ironed out before release.

3.1.1 File Path Inconsistencies in Build Types

There is an unsung file recognition routine that executes differently between the IDE build versus compiled JARs. This is due to classpath and system path differences. This was fixed via more intelligent resource location processing.

3.1.2 BufferedImage's Subimage Memory Leak

A niche logic error, the BufferedImage subimage method requires the flush method to be called after obtaining the subimage. This is due to the subimage retaining and compiling raster data on every new call. This leak was located due to render calls constantly requesting new subimage method calls. The frame rate would actually control the speed at which the memory leak grew. This issue was remedied by calling the flush method. By doing so, the previous raster memory was dropped, which fixed the problem.

3.2 Document Building

Throughout the process of creating "Into The Woods," we have created three different reports. First was the Analysis Document of the project, also known as a Software Requirements Specification (SRS), which was produced as a result of the analysis of the development of our game system. Next was the Design Document, which is a non-exposed internal document and acts as a means for communicating ideas between different parts of a development team during the design phase. It also assisted communication pertaining to reasoning and rationale of key design decisions, design procedures (for example, top-down decomposition of the system and specific architectural style used and UML diagrams). Lastly, the Final Report, discusses how the implementation went and whether or not any major changes to the design had to be made due to complications during the implementation. This report includes a User's Guide and goes through the process of explaining everything that is needed to install and use the software properly.

3.3 Graphic Design

As the game title explains, Into the Woods is a forest base game where the user gets to choose between two characters: Teo, a male character, and Melynn, a female character. They are supposed to look like they have been lost in the woods and are looking for a way out. They believe the floating keys that are scattered around the forest and cave may help them come closer to civilization. All sprites and backgrounds were created on the application called Aseprite. The main two art styles that are mostly used in the game are pixel and abstract art. Pixel art brings a huge homage to any game each of us has played in the past, like Super Mario, Metroid or Contra. Pixel art, in our opinion, is the easiest to animate and abstract art is mainly our graphical artist's preferred art style. The two levels that are created have a forest theme and a cave theme background. These backgrounds are created to be ominous and were assembled in such a way that gave the feeling of depth. We did this by having the colors of anything that looks far away be lighter than the colors that should be closer to the character's, darker. We also used a composition of three layers moving at separate incrementations to create a scrolling Parallax Background.

3.4 Game Design

We determined as a team that “Into the Woods” would be a two-dimensional platformer game. To make our distinctive we decided that the game would include level backtracking as well as traversal on both the right and left sides. Backtracking is an important feature of the game since it allows for the inclusion of puzzle mechanics which are nested within the standard platforming experience. Traversal to the next level requires that the player collect unique collectibles placed across the level. Throughout each level, we thought it would be a good idea to include collectables to give the user a purpose to travel throughout the game. These collectables would be known as Keys. They are concealed behind static obstacles. As the game design flourished as more things were created we thought it would be best to establish an options menu to provide the client with a choice to change their preferences such as with their preferred window size or audio preference.

3.5 Team Management

Throughout development, the team leader kept to a strict meeting schedule. We held two meetings in person each week, three virtual meetings per week (regardless of vacation days and holidays), and we had four meetings during the week of Spring Break. We had also stayed in regular communication through an app called Discord where we shared all our files related to the project. It was very effective to have all the material gathered in one place as well as be able to comment on the reports and

hold meetings in the same place.

3.6 Project Management

Progress in the development cycle was routinely updated by the Team Leader. Therefore, changes were made available on a consistent basis to ensure appropriate communication between team members. Through the use of Git and the website Github, the project files were always at the fingertips for anyone to pull them. New commits to code were pushed daily or at the request of team members. Releases of alpha and beta test versions were often published to the team to allow for playtesting and open-ended feedback. This was extremely helpful for debugging across multiple systems and operating systems.

Individuals were routinely assigned to one or more sections of the project and were given instructions to complete them by a specific deadline. Deadlines were based on the requirements of all current deliverables and were balanced against any extraneous course load within this class and others which had been given to us within that particular period. This balance posed to be a challenge, but proved fruitful. After a member finished their section, the section would be placed under peer review by which other members would be able to make comments and help edit the section to make it better. In addition to the project's hard deadlines, we also had set soft deadlines for tasks which were to be due at a future date. This routine very much helped us with our individual time management, kept everyone aware of each team member’s tasks, and made sure everything was completed ahead of time. Team members were also often asked to temporarily switch off of their assigned task and onto another member’s task in order to assist them. Without such flexibility, there may have been delays in completing certain goals on time.

4. User's Guide

4.1 Installation

4.1.1 System Requirements

The system which is used to run "Into the Woods" must have Java JDK 17+, or Java 8+ installed. Some issues between running the proper JRE without the proper JDK has been noted by some users, but this seems to be a niche issue. The JDK will always work, however, so it is best to have that installed regardless.

4.1.2 Windows Systems

Once the jar file is downloaded, the file will be located in the Downloads folder. Double clicking on the .jar file will allow you to successfully run the game.

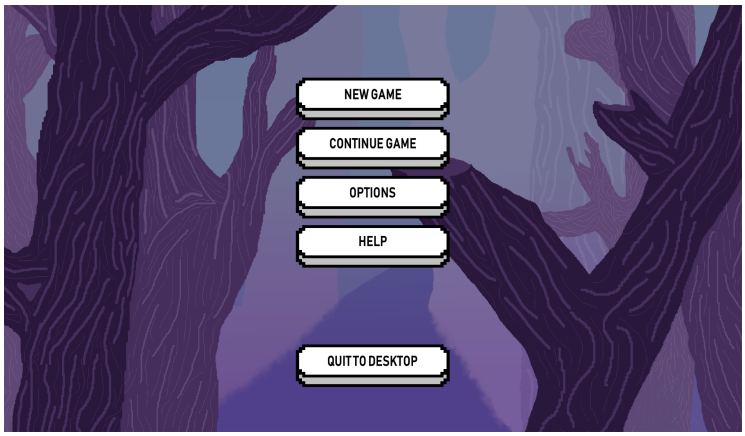
4.1.3 Mac Systems

Once the .jar file is downloaded, open the Finder app. Right click on the .jar and select 'Open'. Confirm that the software vendor is safe, if prompted. If these steps are not fruitful, compress the .jar, then right click and press 'Open'. Confirm that the software vendor is safe, if prompted. The game will then successfully run.

4.2 How to use

4.2.1 Main Menu

Upon starting the game, the user will be brought to the start menu and then subsequently the main menu of the game. There will be 5 menu options that will be displayed. Each Menu option serves a different purpose to the game.



Display of Main Menu

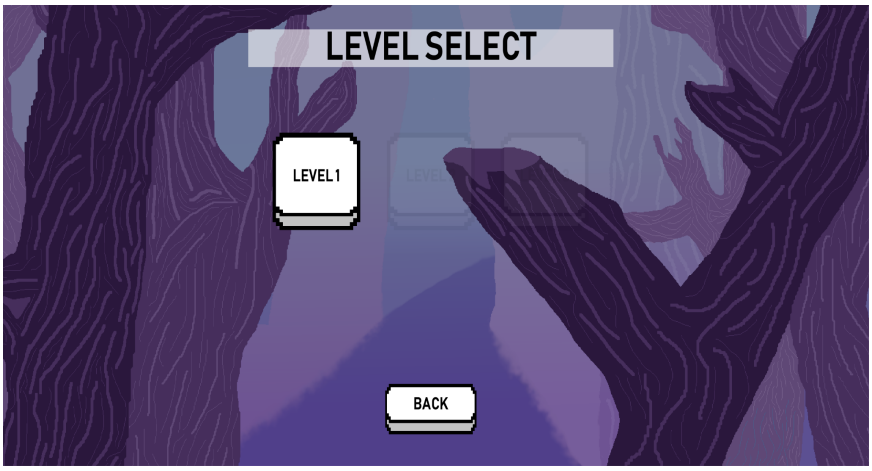
4.2.2 New Game

Once the user clicks the New Game option from the Main Menu they will be required to choose a character, Teo or Melynn. Once the Avatar is chosen the user will be able to start the game.

Display of Create New Game instances

4.2.3 Continue Game

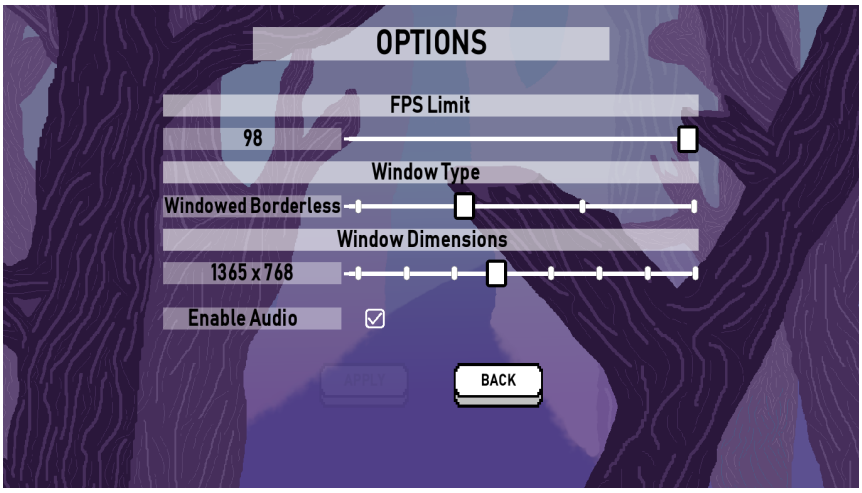
If the user has played and successfully finished and unlocked another level, the continue game option on the Main Menu will allow the user to play the saved/unlocked levels at any time.



Display of Continue Game Menu

4.2.4 Options

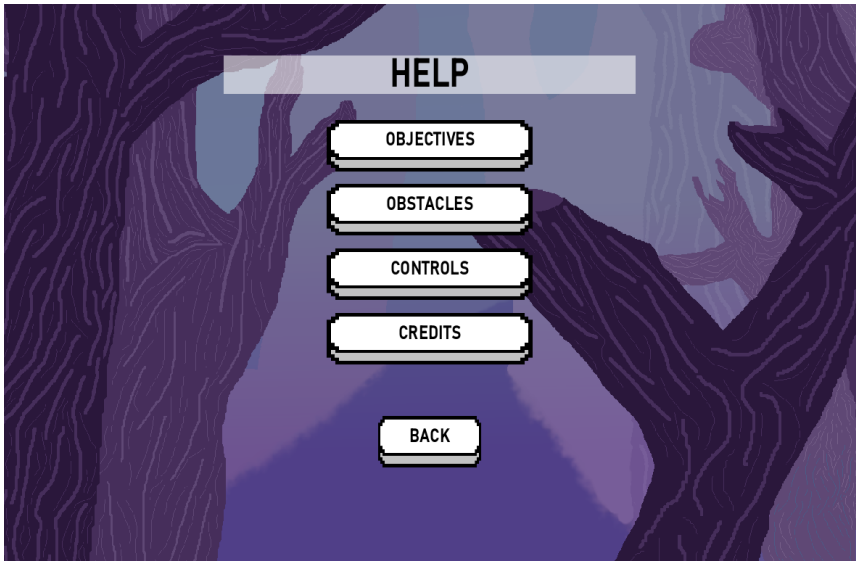
The Options screen displays the FPS Limit, Window Type, Window Dimensions and the option to enable or disable the game Audio. The User can change the settings to their desire at any time.



Display Of Options Menu

4.2.5 Help Menu

Once the user clicks on the Help Menu there will be 4 options, which are Objective, Obstacles, and Control, that will be available to browse through. All three options will help the User understand how to play the game. There is also the Credits option, which displays the contributors of the project.



Display Of the Help Menu

4.2.6 Objectives

Once the user selects the Objectives button there will be descriptions that will explain what needs to be done while playing the game.

Displays the Objectives Page after selecting the Help Menu Option

4.2.6.1 Objective Types

4.2.6.1.1 Key

The Key is the collectible that the Player must aim to obtain. There are a multitude of Keys within a Level which must all be collected to unlock the Door.

4.2.6.1.2 Door

The Door is the final goal for the Player. The Door's job is to transport the Player to the next Level. The Door will start 'Locked', then transition to 'Unlocked' once the player collects all Keys, and then again to 'Open' once the Player arrives at the 'Unlocked' Door.

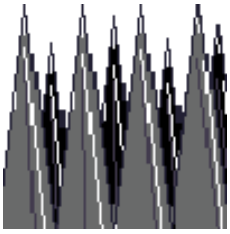
4.2.7 Obstacles

The Obstacles option on the Help Menu, explains what to avoid/use when playing the game. The Obstacle page shows that the game has Spikes, Gas Pockets, and Platforms with the descriptions of the purpose of each one of them.

Display of the Obstacles Page after Selecting the Help Menu Option

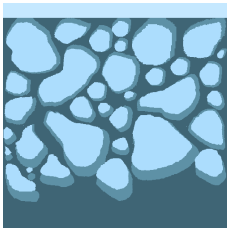
4.2.7.1 Obstacle Types

4.2.7.1.1 Spikes



The Spikes act as a hostile physical obstacle. Spikes will incur an immediate death upon touch, which will cause the player to restart the level and lose all current level progress.

4.2.7.1.2 Platform



The Platform acts as a physical boundary and supplies the Avatar with a surface for running against. The Avatar will also be able to use the Platform as a medium for wall climbing or wall jumping.

4.2.7.1.3 Air Pocket

The Air Pocket is a Trigger entity which acts to rocket the Avatar in a given direction upon entering its area.

4.2.8 Controls

Once the users selects the Controls Page they will see the controls that are used to play the game.

4.2.8.1 Directional Movement

4.2.8.1.1 Left

The 'A' Key will allow movement to the left. Moving against the right side of a Platform will extend the time that the Avatar stays grappled to that wall.

4.2.8.1.2 Right

The 'D' Key will allow movement to the right. Moving against the left side of a Platform will extend the time that the Avatar stays grappled to that wall.

4.2.8.2 Actions

4.2.8.2.1 Jump

The 'Space Bar' will allow the player to floor jump or wall jump.

4.2.8.2.2 Pause / Unpause

The 'Escape' Key will allow the player to pause or unpause the game at any time.

4.2.9 Pause Menu

While the Player is playing a Level, they can click on the “Pause” control Key which is ESC to get to the Pause Menu. The User will have multiple options in the Pause menu; they can resume the game, select Options Menu again to change any settings, or Select Help Menu to read how to play the game, and lastly they can go back to the Main Menu.

Display of the Pause Menu