The Lost Village

Overview

In this game, players can control a warrior with a sword and survive in a world full of zombies. In order to survive, it is necessary not only to avoid the attack of zombies, but also to maintain the physical strength required for action by eating. The attack of the zombies will become more and more fierce, and every day and night you survive will become a proof of your honor.

Game Description (e.g. levels, characters, game play)

WASD controls the direction, J to attack, ESC to open the menu, E to pick up items, T to open the backpack, left-click in the backpack to use the item, mouse middle-click to discard the item in the backpack and right-click to drop the item in the backpack. Players can earn points and supplies by killing zombies. Supplies include coin food and medicine. Damage taken during combat can be recovered by using medicines. Excessive hunger will reduce physical strength, but if the hunger is maintained above a certain value, the blood volume can be slowly restored. Hunger can be restored by eating. The game ends when the player dies, showing the score and survival time.

Implementation (e.g. game engine, scripts, algorithms, tools)

The game engine we used for this project was Unity. The game is comprised of 3 main scenes: the main menu, the game and the end game scene. The main menu scene is a simple 3 button menu which will either let the player start the game, change some settings or quit.

The main game scene is where the player controls a character sprite in 2D space. Scripts are responsible for the player being able to move (W/A/S/D keys) as well as attacking zombies and environmental objects with the J key. The way zombies spawn is via spawn points set on the map. Zombies can spawn at any of these set locations at a set interval (~10 seconds) as long as no other zombie is currently occupying that radius around that spawn point. Zombies are also essentially just game object prefabs that have a sprite and script attached to them. The attached script has all the traits of the zombie such as damage it does to the player, its default movement and player lock on.

In the main game scene, there is an inventory that the player can open with the “I” key. In the inventory, the player can use the mouse to click on the meat, potion, and apple to either restore their health or hunger. Pressing the “I” key again will close the inventory. The “E” and “R” keys will pick up and drop items respectively. The player can pick up the items dropped by zombies when killed or environmental objects that are hit and destroyed. The items dropped by the zombies are randomized by using the random function from 0-9. Depending on the number returned, the combination of items is dropped from the killed zombie. In addition to item drops, there is also a day/night cycle within the game controlled by a DayNight script. This script allows the cycle to have a certain length of time, smoothness, and also keeps track of the hour and day. This day/night cycle is further made possible by Unity’s 2D post processing in the 2D Rendering pipeline. The script adjusts the colors of post processing color adjustment, adjusting a weight between 0 and 1, to simulate a day night cycle. At night, the environment around the player is pitch black. To create some playability within the night, we also added a point light to the character to allow vision during the night.

In addition, another tool used was Unity's Universal RP pipeline (2D rendering pipeline). This allowed the project to be rendered and made in 2D. Thus enabling the use of shaders, normal maps, lighting, and specific layers which can be used to organize game objects throughout the scene and set specific rendering orders for lighting and object layering. This ultimately allowed us to create the day/night cycle, which is integral to the aesthetic of the game.

In the post game stats scene, it displays the amount of time the player survived as well as the number of zombies killed for that run. It grabs the timer and kill count from an earlier script where it is displayed in the main game scene as well. From this scene, players have the same options as the main menu scene. They can either restart the game and play again, change volume, or quit the game.

Another tool used was Unity's Animation controller. By using Unity's Animation controller we were able to utilize the animation states for the main character and enemies to allow them to have smooth, correct, and flowing animations. In addition, the Animation controller also allowed us to create events at specific animation states of a character (idle, swinging, footstep). Doing so, specific SFX sounds can be played when the player makes a footstep, or swing. This is so that the timing of the animation and the timing of the sound match up perfectly.

Lastly, some other tools that were used to complete the project is Aseprite. Aseprite is a drawing tool to draw 8-bit, 16-bit, or other forms of pixel images. This tool was used to create the objects in the environment, the map, trees, and rocks in the main game scene. It was also used to create a lot of the assets used for crafting (though, they are unused unfortunately).

Project Post Mortem

What tasks were accomplished?

All of the tasks that were higher priority were finished and were polished to our best abilities. The only task that was not optimized fully was the map boundaries. There are some bugs where parts of the character will be outside of the map, but this does not affect the gameplay in the slightest. It only affects the aesthetics of the game.

What planned tasks were not done?

The planned tasks that were not completed are background music and crafting. Background music was not really high on the priority list, and even if added would make no change to the overall gameplay of the game. A team member was going to implement an original BGM, however, that was not possible due to time constraints. For crafting, there was not enough time. Before we start working on crafting, we need inventory and item interaction. Since we were not able to work on it right away, it ended up being pushed back and not implemented.

How did scrum work for you?

The scrum worked decently well. The meeting technique worked well as it got us together to discuss what we needed to do and what was done. Although other than the day we met, we did not meet additional times. For the most part, it kept us on track until one of the team members (Alton) had a setback in one of the planned tasks.

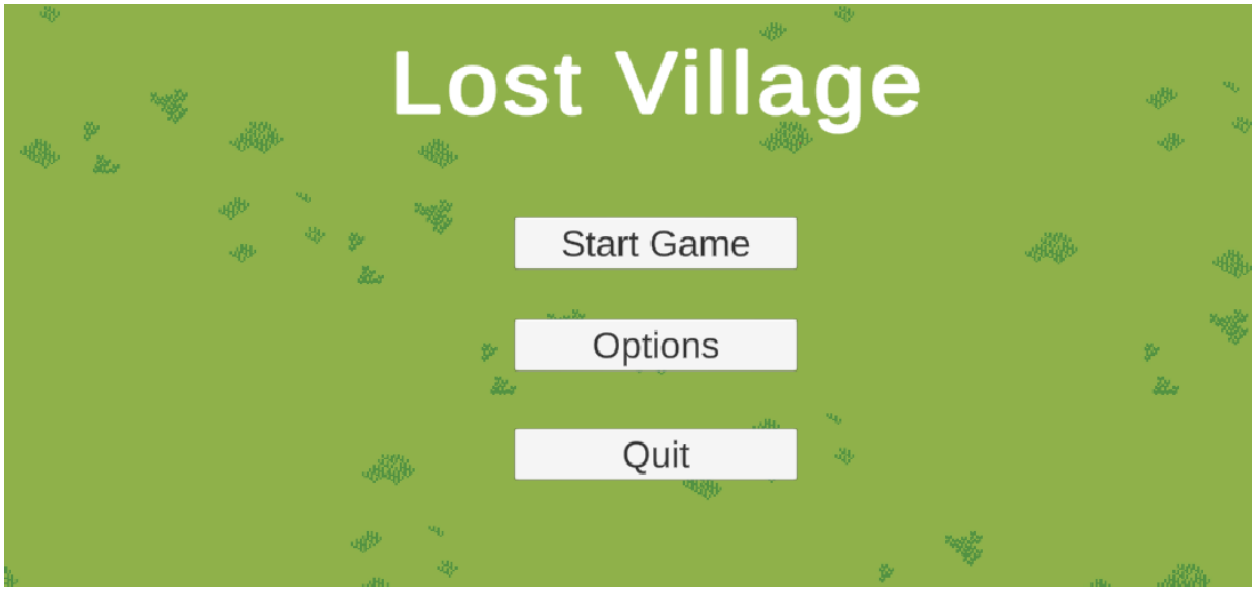
What would you do differently?

The only times we met as a whole group were during lab times. I feel like this limited how much work we were able to do. During the week we sent frequent messages to our chat

group about things we were working on and if others could test a feature but I feel like we would've benefited much more if we had an extra group meeting in the middle of the week before lab session in order to discuss plans and sprint progress. It would also be easier to gauge how everyone is doing if we meet an extra time per week.

Photos

Main menu



Starting area

Enemy

Loot and Light area when night time

Inventory

End Screen


Total kills: 1
Total Time Survived: 01:19:80
Start Game
Options
Quit