“Before Dark” Final Progress Report

Published by: React Gaming

5/1/2020

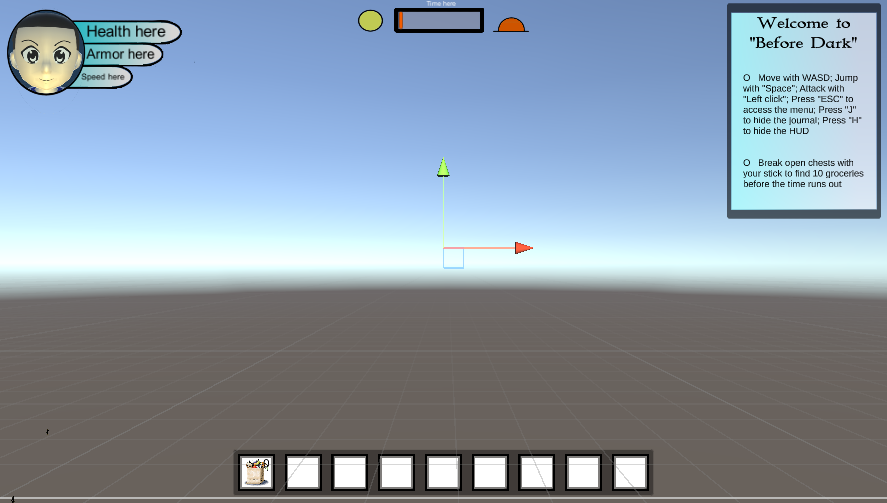
**Final Update**

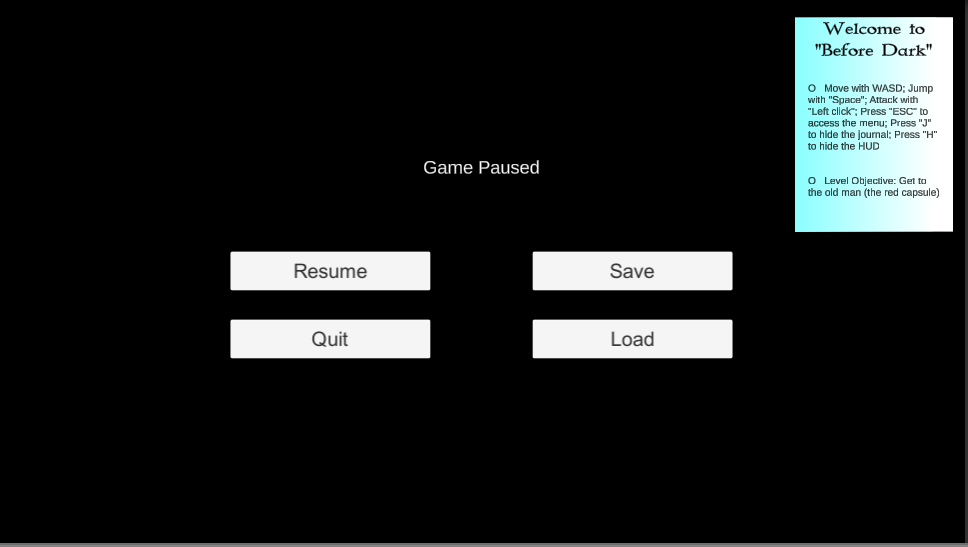
* Michael Gee: group leader; manages the group; makes sure tasks and goals are completed on time; schedules the group meetings; programs player and character movement; updates the OPPM
* Leopold Frilot: programmer and documenter; tracks everything the group does; manages the tech behind the group including GitHub and Discord; programs game logistics such as saving and scene management
* Bram Metz: environmental artist; designs the levels and general feel of the game as well as the entities that inhabit it such as enemies, loot, and structures
* Crouchet: programmer and web designer; manages everything about the website; designs most of the mechanics and second-to-second gameplay that the player will be experiencing
* Angel Martinez: character artist; creates character models and animations to go with them; implements these into the game

**Gameplay**

    The current version of “Before Dark” has a very simple gameplay loop. Pablo starts at the beginning of the level and is given a task to find all the groceries(10) that he left scattered around the park. Pablo moves with WASD, jumps with Spacebar, and attacks with Left Click. Using these options, Pablo should attack chests to open them and retrieve the groceries inside. Meanwhile, goblins will try to attack Pablo and must be attacked to destroy them. These chests may also hold armor pieces that passively buff Pablo’s stats. All collected items show up in the inventory bar. If Pablo runs out of health, he is simply transported back to the beginning of the level. The only way to lose the game is to run out of time (the sun goes down). Once Pablo has collected 10 groceries, the boss will spawn and will be able to be attacked. Once the boss is “killed” Pablo will move on to the next level and repeat the process. Once Pablo kills the Boss in the new level, the game will end as Pablo is able to return home on time.

**UI Design**

The UI is designed to be as simple and readable as possible. The top-left section of the HUD contains Pablo’s stats. Health is the amount of hitpoints Pablo can lose before being sent back to the start of the level. Armor reduces incoming damage by 5% \* the amount of armor. Speed is the current speed of Pablo. All three stats may be increased by items found randomly in chests. The upper middle of the HUD shows the time left to complete the level. The yellow circle on the left symbolizes daylight(5:00 or so). The orange half-circle on the right symbolizes the setting sun(7:00 or so). Once the orange bar reaches the sunset, the game is over. The rightimage is the “journal” which shows useful information to Pablo such as the controls and the current game objective. This may be hidden as well as the whole HUD.  


The pause menu has four major buttons: Resume, Quit, Save, and Load. Currently Saving and Loading only saves/loads Pablo’s position, health, armor, speed, number of chests left the the level, and the current time of game. Resume simply breaks the pause state and Quit quits the game entirely. Again, the goal here was simplicity and read-ablility. The journal also pops up here so that the user can be reminded of the instructions incase they hid it in the game and forgot how to reaccess the journal.****

**Changes and Trade-offs**

            Originally, we envisioned a team of 6 people working to make the best game in the class. However, after unforeseen circumstances we had to re-envision our original idea.

    When creating Pablo, we originally envisioned him being able to wield a variety of items and weapons found throughout the level. The entire process for creating the characters mesh, rig, textures, and animations were done in Blender, a free and open source 3D creation and animation software. During the animation process, it proved difficult to attach separate objects into his hands using the same action animations. Due to this, we had to limit him to only use one weapon by combining Pablo and his Stick Sword into one single mesh. This fixed the issue with his weapon sticking to the animation but meant that we could not allow him to hold different items anymore. For this, we had to insert a general “pickup” animation that simply adds the items to the inventory solely for quest progression purposes.

    Unfortunately setting up the old man was a lot more difficult than first imagined. There would need to be another model made, new animations, and rigging so that certain dialogue was displayed when Pablo has reached certain conditions. We elected to remove him from the game. The older brother also had this issue, but we decided to keep him at the beginning of the level and have him say the instructions to the player. His form is somewhat ethereal to indicate that he is not really a part of Pablo’s imaginary realm.

    While we got the sunset working, the mechanic of making the game harder as a result was not implemented. This is because the combat system is not nearly as complex as we originally intended. The height of the sun was supposed to change abilities, speed, properties, etc., but in the final product, the only thing it would be able to change was enemy damage. We elected to not include this, as that would need to be accompanied by items that Pablo would need to find in chests in order to make it remotely fair. It’s just not practical under the current scope of the project.

    While items were implemented which cause stat changes for Pablo, they do not visually change Pablo like originally intended. The hazardous terrain was also not implemented in time. The only enemies implemented were goblins and the shadow boss. We also only implemented two of the three levels, but this is because the ones that Bram designed and built are much more detailed than previously imagined.

**Software Design**

Some of the Finite State Machines we used dealt with character and npc animation. This involved our main character being able to transition states based on specific user inputs and increases/ decreases in movement speed. Some of these animations included: Pablo being able to idle, his stationary animation, run when his movement speed increases above a certain value, jump when the user presses the space-bar, and attack with the click of the left mouse button. Other finite state machines for the npc’s included enemies such as the goblins and a shadow boss, along with Pablo’s older brother which had similar transitions. The goblins had a stationary idle animation, once agrod by Pablo, the finite state machine would transition to a run state chasing the main character. The shadow boss includes an invisible state and an idle animation. Pablo’s older brother, who gives him his main quest, also idles and occasionally exits this animation to transition to an item pick up state, as if he sees something on the ground.

 • Give high-level explanation of any AI used: (explain goblin ai here)

**Audio Design**

    There are three main sources of audio. The first is background music which is a relaxing composition of Final Fantasy music put together by a YouTuber named “Donut.” The purpose is to give a fantastical, yet calm feel to the world through the audio. The second is a sound clip of birds chirping in Slovakia, recorded by a user named “bajko” and posted on freesound.org for others to use. The purpose of this is to bring the player back to reality with real-world sounds. The two background tracks sometimes harmonize, and sometime conflict with one another, symbolizing the conflicting worlds of Pablo’s imagination and the real world. If we were to expand on this further, different attack sounds would sound majestic and strong, but randomly we would add real-world stick swinging swooshes and Pablo saying ”HA!” to further drive this illusion home.

**Level Design**

**Model/Art Design**

    We wanted to base our game closer to the art design found in two popular games, *Breath of the Wild* and *Risk of Rain 2*. The art design features a simple yet pleasing to look at style. When making the models, we attempted to keep the overall poly-count relatively low and minimal for both aesthetic purposes as well as efficiency to run the game on lower end hardware. While the models themselves are quite simple, they deliver the proper message of a simplistic adventure from the mind of a simple child. For the texturing, we wanted the style to be closer to the look of anime, giving Pablo big, innocent, and adventurous eyes to reflect his personality. For the materials, we used toon shading on the older brother to enhance the effect of Pablo’s imagination. Each shading was mixed and matched based on the characteristics of the characters themselves. When UV-unwrapping the models for texturing, we needed to pay close attention to where we marked the seams in order to make them less noticeable to the user. Marking seams in the correct places reduces stretching in the mesh and the texture when animating the models, making it an important process. Once the models were rigged, weights needed to be added to each bone to ensure that the proper limbs were moving with their respective bones. Again, this helps to reduce the stretching of the model while certain body parts are moving. Finally, for animating each model, certain keyframes were prioritized to achieve the right set of motions for the animations. Not only does this speed up the process for animation, it also ensures that the right message was achieved for each action taken. Getting all of this set up in Blender first was a high priority as it makes exporting files over to Unity seamless with minimal setup on the programmer’s end.

- Credit any open-source audio, artwork and code used:

Camera controller and camera collision were implemented watching a YouTube video titled “Free 3rd Person Camera Setup & Camera Collision Tutorial” by Filmstorm. Character armatures were implemented using the MetaRig addon for Blender. Player movement was also implemented watching a YouTube series titled “Unity 3D Platformer – Learn to Make a 3D Action Platformer” by gamesplusjames. A UI item interface was added to the game via a tutorial video as well titled “Unity Inventory UI Tutorial” by Jayanam. The background music is a composition of Final Fantasy music put together by a Youtube creator by the name of “Donuts.” The bird sounds are open-source from freesound.org by a user named “bajko.” They were recorded in Koliba, Slovakia. All other script/code was made from scratch with the help of the Unity User Manual online.