Bosque Treasure Hunt

A 3D online game by: Naomi Burhoe

Introduction

When I started this project, I had no notion of the challenges that I would face. I did not know that the idea for the project would completely change as I continued developing nor did I know that I would spend days upon days editing graphics and controls to make the experience more interesting for the user. As it is, the game that finally emerged from this process is very different from the one that I had in my head when I first began. That being said, the game that emerged is actually pretty complicated, at least for a first time Unity developer such as myself. in addition, the web application which I created took some time due to issues arising from not understanding how to use the different frameworks available to me at the time. By looking up tutorials, one of the most in depth ones being Unity's "Create with Code" course, and reading documentation, such as for the Microsoft Azure portal, I was able to emerge from this journey a more confident and creative person and a better software developer.

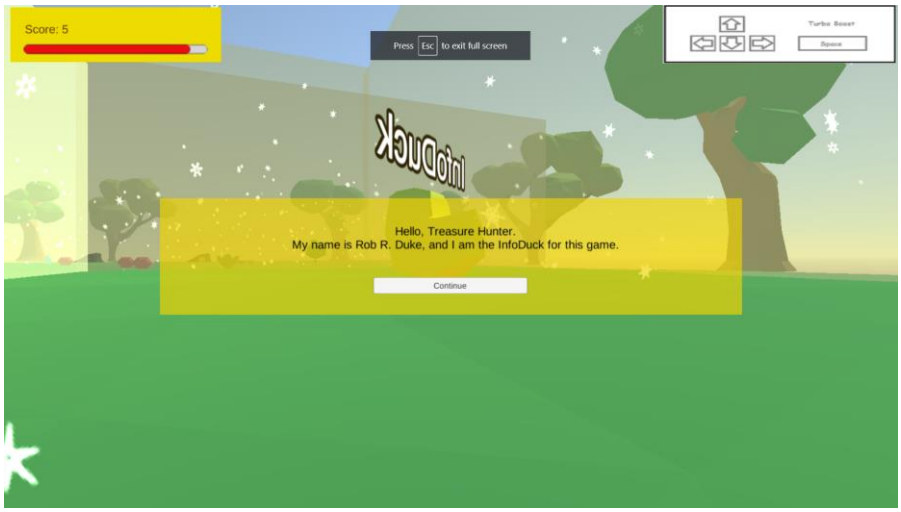
Architecture & Design

Overview of game

The first part of the project is an online 3D game, called Bosque Treasure Hunt, which was designed in Unity and built using the WebGL framework offered by Unity. In order to design the game, I chose to consult the Unity learning website so that I could grow more comfortable with the overall development environment and so that I could learn how to implement the different aspects of a game with code. Game features include an interesting user interface, standard player controls, RPG and dialogue elements, integration of various assets for look and feel of the game, score updates, game management elements for start and finish, and fabrication of new enemies and treasures to be found. When users first encounter the game screen, they will find themselves presented with a menu that allows them to either play the game

or exit the application. If they so choose, they can put the game into full-screen mode for a more interesting interactive experience.

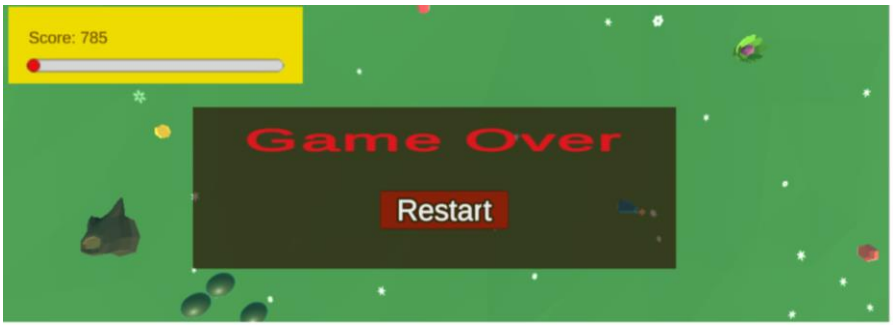
Users must be warned, else they might die prematurely due to the incessant attacks of spherical enemies, called Brownies. These creatures can spawn anywhere and will grow in number whenever all other Brownies have been destroyed or whenever all current treasures have been picked up. If new users wish to look for a safe zone, there is one that is only accessible when they interact with (run into) a helpful NPC, called Rob R. Duke the InfoDuck. This NPC gives helpful advice to new users about the dangers of the Brownies as well as information about game mechanics related to potions and collecting treasures.



After the initial encounter with the InfoDuck, users will find that a new button has appeared in their status/action bar at the top of the page. This button allows the user to ask the

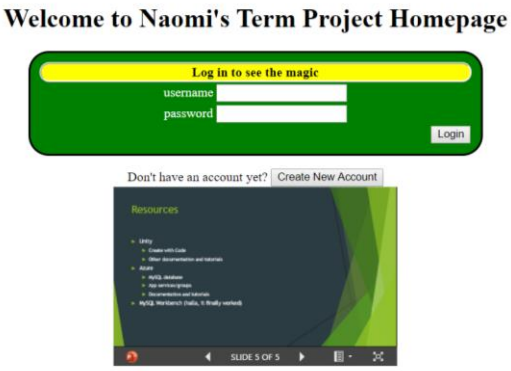
InfoDuck about information related to potions and treasures, or just to say hi. The status/action bar at the top of the screen allows the user to view current points, health levels, and helpful tips and hints for gameplay. As a note, controls for the player are W and S (or the up and down arrow keys) to go forwards and backwards and A and D (or the left and right arrow keys) to rotate the camera. If users press the space bar, they find that their player moves with just a bit more speed.

Points for the game are accrued when users collect treasures, or attack Brownies with strength potions equipped (by colliding with them). The various types of treasures include diamonds, emeralds, rubies, and coins. Potion types include variants for strength and health. When the user loses all of their health as a result of being attacked by Brownies, the game ends. At this screen, users can make note of their current high scores and restart the game. Upon hitting the restart button, the game menu reverts to the original screen and users can decide whether to play again or to quit the game.

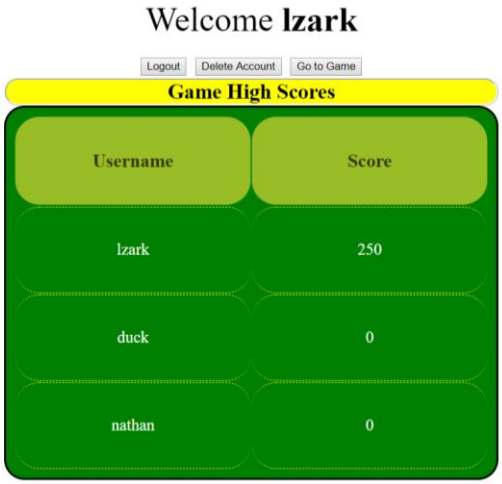


Overview of Website

The second part of the project involves the integration of the game into a website associated with a MySQL database. When one initially navigates to the website, they will find themselves presented with a login screen. From here, they can create new accounts, access project documents, or sign into the user homepage.



If the user elects to sign into their specific homepage, they will need to make sure that all form fields are filled out and that the information they are entering for username and password correspond to what they originally used to sign in. (Possible Fix: Add a “Forgot Password” element for processing data another way). Provided the data they enter is correct, they will be presented with a user homepage, as shown below. From here, users can navigate to the game screen or delete their account.



The game screen is the actual hosting mechanism for the Unity game which was generated previously. It has two primary features: the Unity game and a form for updating user high scores. As a note, the “update score” form is not usable during gameplay. This is because all

resources for the “game screen” webpage are being diverted to the game, rather than to the form that was created. If the user wants to ensure that their updated high score reflects the value they saw in the game, they should make a note of the highest score they achieved during gameplay. In addition, if users choose not to update their scores, they will need to navigate to the login screen for the website in order access user-sensitive data. (Possible fix: store current data in a different stream).

Database Design

For this project, I used one MySQL database to store user information, which includes usernames, passwords, and high scores. In order to access this data, I included a series of checks and balances into php that only parsed the server when certain requirements were met. If the user wished to create an account on the website, they had to make sure that they were using a unique username and that they also entered a password on the form. If either of these conditions was not met, then the user would be presented with an error message and presented with a button that returns them to the login homepage. The same logic was applied to other operations involving the database, as well, to ensure that users did not enter incorrect data or accidently change or delete important user-related information prematurely.

Conclusions and Thoughts for the Futures

While the initial logic for the project definitely changed as I continued developing it, the overall idea did not. I knew that I wanted to create a project that integrated ISP and HCI elements in an elegant and intriguing manner. The outcome of this process was everything I wanted it to be and more. Somehow or another (through days upon days of reading documentation, completing tutorials, and struggling to find the right assets and prefabs), I was able to design a relatively complicated project which included multiple levels of configuration and server-side processing. I am quite proud of what I was able to accomplish this semester and I think it would be something that I would like to continue working on in the future.

As such, I do have some ideas which I would like to implement at a later time. As one can see in the previous portions of this report, I made note of different elements of the project that I would want to change whenever I have time.

Here's a list of the changes I would want to make to the project:

1. Expand upon RPG elements and create a more adventure-driven schema for gameplay
2. Sync MySQL database directly to game so that users do not have to manually update high scores.
3. Add prefab to player in game so that it's not just a sphere move around a world
4. Cleanup controls and adjust user status/action panel
5. Add in mouse controls
6. On the website, I'd like to add a feature that saves data across the website so users don't have to manually input login information for every action they complete
7. I'd also like to reconfigure the gameplay webpage so that the update score feature doesn't look so weird when formatted with the WebGL window.
8. Add a "forgot password" feature so that users can update their password should they forget what they entered previously.