Game Engines Assignment 1 Report

By: Thomas Lindo, Jacob Rosengren, and Tyler Shamon

What We Did

We created a Dialogue Plugin for unity that allows the user to walk into an area, press ‘E’ and read dialogue, then click continue to move to the next line of dialogue. The dialogue box closes if they hit continue if there is no more lines of dialogue left or by simply moving out of the area.

Who Did What

Thomas - Got the base for dialogue management from a youtube video (<https://www.youtube.com/watch?v=_nRzoTzeyxU>), helped with various debugging errors, and helped the create the report.

Jacob - Programmed the plugin and helped to convert the c# version of the dialogue system to a plugin.

Tyler - Helped with debugging errors in C# (specifically errors with arrays which were solved by converting all arrays into lists) and Unity (found and helped solve several unset object reference to object errors) and helped write the report.

How It Works

The plugin itself was based off the simple logging plugin from week 2 of the tutorials and from the tutorial video listed above. The logging system was improved so that it now creates a separate directory for any logging files. The other 2 functions the plugin includes is that it can load dialogue from a text file and even save a string to text file to be used as dialogue. They way how they both do this is that they use the first argument, a\_ObjectName to define which folder the text should be loaded or saved, and then uses a\_Item to find the exact text file. a\_Value in SaveDialogue is the text that will be added to the document.

Now in the C# scripts, to load in all of the dialogue, we used foreach() to check and load all of the .txt files within the My\_Dialogue folder and store them in the NPCDialogue object which contains the names of the NPCs and the sentences.In order to make the dialogue appear when the player presses “e”, a text gameObject is created in Unity and attached to a canvas. In C#, a public Text object is used to manipulate the text on the canvas into displaying the dialogue that was stored. As for the continue button, a component is attached to the canvas which creates a button. In order to continue the dialogue, when you click the button, we advance the index of the list to display the next sentence (just like an array). Each text file makes up one item in the list (this was done so we can have multiple sentences as “one” line of dialogue), and when there are no lines left, the dialogue UI is destroyed.