FIT 2097\_ A2

Link: <https://gitlab.com/yyeu0005/fit2097_a1.git>

Readme document has been updated, demonstration video and a copy of this document are pushed to the git (In the document file).

Program Design

Diagram, schematic

Description automatically generated

Character

Same as last version (A1) nothing new has added.

[This class manages the player’s health, food, and water level. Each of these variables has its own function for increase, decrease and get. I make those functions and the variable public to allow other classes, such as the Pickup class and the Trap class to get and make changes to those variables. The countdown, startTime, and reset timer variables are used to make the health, food, and water attributes drop every second. Those bool values: playing, pause, dead, and won are the status of the payer. The startClicked, isPause, checkDead and checkWon functions are created to check the player states and react. The CallMyTrace and ProcessTraceHit functions allow the player to interact with the interactable object. The CanPickup Boolean is created to ensure the player used Trace to interact with the object first before seeing the ‘how to interact’ message. The holdingItem Boolean is to check if the player picked up a key. ] ---- From A1

Enemy

The enemy character is 10% slower than the player. This is can be achieved by MaxWalkSpeed in character movement times 0.9.

EnemyAI Controller

Player health is drained when the enemy touching the player. As they seem could never touch each other, thus in Tick would I make it to calculate the distance between the player and the enemy character. And if the distance is lower than a certain number, the health starts to get drain from the player. And the Boolean underAttack becomes true.

Fuse

A simple class works the same as the Key class in A1.

PowerBox

The class has a hasFuse Boolean to check if the player has inserted a fuse into it. The Switch class would check this Boolean to activate itself.

Switch

As new game machines like fuse place to bridge and binary code switch are added to the game, there are new attributes added to the Switch class. needFuse, needKey Booleans are added to define which types of switch they are. Unlike the switch with the key. Switch class would look into the player inventory to see if they have a fuse but to cast the PowerBox class for the fuse. Switches would be able to switch off for the binary code switch.

The change of material is achieved by using the SetMaterial function and assigned two different materials.

NewBridge

I would make a new class for the bridge, as the old Bridge makes itself invisible when the switch is not activated but this assignment needs to see different material applies to it.

The bridge would be down once the switch is activated. The bridge would be placed vertically in the game world. The angle variable is set to 90 degrees so when it is down the player can cross over the flat bridge. The down bridge action has animation show it is not rotated 90 degrees to the right place in an instant. I create a doOnce variable to ensure the bridge is not going to rotate itself eternally. BCLiftBridge function is created for the binary code switch. I make the bridge teleport back to the start point, allowing the player to know that the binary code is not correct immediately.