FIT2097 – Assignment 1 Program Design Document

Prepared by Joshua Leung

StudentID: 28261526

AuthCate: jleu0001@student.monash.edu

UE4 Gameplay Framework Classes used:

Used to a certain extent but not modified in any way because they were existing in the template:

- Camera
    - Viewport functions
- GameState
    - SetPause

Used and modified extensively to achieve design goals and targets:

- Character (FIT2073Week3Character)
- GameMode (FIT2073Week3GameMode and MyFIT2073Week3GameMode(BP))
- HUD (Multiple HUD's, can be found in HUD folder)
- Input (Binded new Pause input to P, binded OnPause function when P is pressed in character)
- Actor (All interactable objects)

Self-created/Extensively modified classes and their base parent class:

(Note, there are some CPP classes created but not discussed here, those classes were created but never used, remained in the project so as to not break anything. Thus, they were not included here)

FIT2097Week3Character

- Base Parent Class: ACharacter

Logic, information and code encapsulation:

(Custom and modified variables and functions included only)

For this class, we would need to keep track of player health (to satisfy design goal), a Boolean value that we can alter at will through functions to set the pause state and event dispatchers for death, pause and win events.

To satisfy the design goals of the assignment, the player must be able to heal. This would be done by taking an int amount and adding it onto player health. This function would be binded to a broadcast event from HealthPickup to simulate the effect of healthpickup healing the player. The player class gets the heal amount information from making an initial collision and cast to bind the heal player function to the event. The player class then waits for execute interact and display information to be called and listens for the heal event to be dispatched.

To satisfy yet another design goal, the player must be able to take continuous damage. This would be done by GameMode binding the decrement health function to a timer that calls it every 2 seconds. This

would mean that every 2 seconds, the player loses 10 health. In this interaction, the Character class gets data from a Direct Link as we have stored a pointer to our character in our game mode.

The Character's DecrementHealth function has an event broadcaster which is called when the player's health reaches less than or equal to zero. When it is called it will broadcast the event to our gamemode. This could be achieved thanks to our Direct link. The broadcast will then call on lose game and display the level loss widget and enter the game dead state.

To "Collect" keys, the player makes an oncollision event happen with a keypickup object and triggers the execute interact and execute display information. This oncollision event alerts the key class that it must broadcast the give key event in their execute interact. Leading to the vertical doors now having the ability to be opened.

To open the AutoDoors, the player triggers a linetrace on the function onfire and ends up making a collision to a SwitchDoor. Through the linetrace code given in moodle, we alert the Switch door class that they need to execute their interaction and display information. This sends an event broadcast to call remote open in the Autodoors.

To pause the game, a new binding was created that binds the "P" key to Pause. In our setupplayerinput, we created an FInputActionBinding& toggle and binded the OnPause function call to the key press. We also set toggle to execute when paused. We did this so that we can call our OnPause function even when paused. This will allow the SetPause Boolean to switch values every function call and the event broadcast will send the setpause Boolean value to our gamemode and pass it to the pause game function allowing us to switch between pause states.

To win the game, the player must go through the level and collide with the end node. On collision, we get data by casting the other actor we collided with to an end node. If it is an end node, we broadcast the player win event which we have binded to call win game in our game mode thanks to our direct link.

FIT2097GameMode

- Base Parent Class: AGameModeBase

The GameMode contains references to created widgets, player controller, the characterclass and a FTimerHandle. The character class reference is used to have a direct link and allow events to be binded. The player controller reference is used to pause the game. The FTimerHandle is used to send data to the player class that decrease health is to be called every 2 seconds.

Interactable

- Base Parent Class: UInterface

Our base Interactable interface. Only has two functions declared which is where our Interact and DisplayInformation reside. It is used by our interactable objects to be able to be interacted with.

HealthPickup

- Base Parent Class: AActor

The HealthPickup class stores an int variable health set to 20 which is then broadcasted and sent to the character class when a collision occurs and calls the execute implementation and display information.

VerticalDoor

- Base Parent Class: AActor

The VerticalDoor contains an open Boolean value that determines whether the door is opened or closed and a hasKey Boolean value that determines if the door could be opened or not. It also contains a reference to a keypickup object that is blueprint exposed so we can directly link Vertical Doors and keys together. When it gets information from an event broadcast from a keypickup that calls the give key function, the door will then be able to be opened. This happens when it receives data that a linetrace collisionhas been fired from the character causing the execute interact and display information to be called. Causing the door to move up as set in the execute interact

KeyPickup

- Base Parent Class: AActor

The key is a pickup item similar to the health pickup but not having health. Instead it only has an event broadcaster and its interaction is to display information and broadcast the give key event to vertical door

AutoDoor

- Base Parent Class: AActor

The AutoDoor contains remote open function that opens the door when it is called. It also contains a reference to a switchdoor object that is blueprint exposed so we can directly link AutoDoors and Switches together. When it gets information from an event broadcast from a switch it calls the remote open function that leads the door to be opened. When it receives data that a linetrace it will display information.

SwitchDoor

- Base Parent Class: AActor

The SwitchDoor contains a remote open event broadcaster. This is called when it receives data that a linetrace collisionhas been fired from the character causing the execute interact and display information to be called. This causes the auto door to open.

EndNode

- Base Parent Class: AActor

EndNode simply has a mesh and displays information. It does not send or receives data. Instead it is used for a collision check that is then used to send information to game mode.

Widgets

- Base Parent Class: UUserWidget

The widgets used are containers for display element variables that are called in the game mode.