Programming Assignment 2 - Report

Abel Weldaregay

ECE 407

Professor Yuzhong Shen

25 February 2020

Version 1

Introduction:

The goal of the assignment was to introduce us to MonoGame development and to have a deeper understanding of what is going on under the hood of 2D games. This helped have a better understanding of the control flows and implementation of MonoGame, which will make it easier to transition to Game Development on Unity.

Program Design:

The documentation generated for this project is included in the ./doc directory. The installation and compilation commands are the same as the original template provided. The game can be compiled and ran on Visual Studio 2019.

Results:

I began by reading the code and the provided documentation for each file to have a better understanding of the control flow. In completing task 2, I first began by generating a sprite font using the monogame pipeline, then I created SpriteFont object and loaded the generated font in LoadGameGraphics() method using the Content load manager. In completing task 3, I used a png picture downloaded from the internet. I added it to the pipeline by using the existing item functionality and loaded it into a 2DTexture and set the background of the game using the SpriteBatch. Sound effects were added using the SoundEffect class. The first step is to add it to the project using the MonoGame pipeline and then creating a SoundEffect class and loading the sound effect. Then creating a SoundEffectInstance which plays the sound effect. Adding background music was a similar process but using the Song object since it supports mp3 and wav files. Handling user inputs were done in a very similar pattern. The key thing to do when handling user input is to remember the old state so that it is executed only once. All of the user inputs are handled in a very similar format to the following example in figure 1.

```
KeyboardState newState = Keyboard.GetState();  // get the newest

// handle the input
if (oldState.IsKeyUp(Keys.P) && newState.IsKeyDown(Keys.P))
{

    // do something here
    // this will only be called when the key if first pressed
    this.paused = !this.paused;
    if (this.paused == true)
    {
        MediaPlayer.Pause();
    } else
    {
        MediaPlayer.Play(backgroundMusic);
    }
}
```

Figure 1: Handling User Input - Pause

The inputs are randomized by using the Random generator. The first thing that is done is creating a randomx and randomy array. This will hold two generated random x and random y values at the specified bounds. Then a random index is selected to pick from the randomx and randomy values. The code snippet for this is shown in Figure 2.

```
int[] randomx = new int[2];
int[] randomy = new int[2];
randomx[0] = rnd.Next(4, 6);
randomx[1] = rnd.Next(-6, -4);
int xIndex = rnd.Next(0, randomx.Length);
randomy[0] = rnd.Next(3, 5);
randomy[1] = rnd.Next(-5, -3);
int yIndex = rnd.Next(0, randomy.Length);
int x1 = randomx[xIndex] ;
int y1 = randomy[yIndex];
```

Figure 2: Generating Random Values

The name is displayed by using a simple SpriteBatch.DrawString method. The code snippet responsible for drawing the name to the screen is shown in Figure 3.

```
String message = "Ping Pong Customized by Abel Weldaregay";
spriteBatch.DrawString(nameFont, message, new Vector2(Window.ClientBounds.Width/2 - (message.Length*2), Window.Client
```

Figure 3: Drawing Name to Screen

I think calling the method CollisionOccurred() at the end of MoveBall() is a problem because it does not provide accurate collision detection. I think the CollisionOccurred() method should be called exactly when the ball hits the paddle instead of calling it all the way at the end. Calling it exactly when the ball hits the paddle will provide a more realistic user experience.

To address the issue of collision detection when the ball hits the paddle at high speeds is to implement more accurate collision detection. One way to implement more accurate collision detection is to implement speculative contacts.

One other improvement I did to the game is that i updated the frame rate from 30 to 15 to make the game more exciting. This also made the game more realistic.

Screenshots: