Programming Project #4

Adding a Player Object

CpSc 4160/6160: Data Driven 2D Game Development

Computer Science Division, Clemson University

Brian Malloy, PhD

March 6, 2018

In order to receive credit for this assignment, your solution folder must be compressed and submitted to the web handin bucket by 8 AM, Friday, March 30^{th}, 2018. If you cannot make this deadline, you can receive 90% of the grade by submitting your assignment within three days of the due date.

For this assignment, you will begin to build a playable game with: a player object, HUD, AI, and collision detection.

1. Incorporate a player object into your animation (encapsulated); use asdw to control the player. I will provide 3 different designs and you should choose one of them: (1) Inherit the player from Drawable, (2) Inherit the player from MultiSprite, (3) use *composition*. The design choice is yours.

2. Collision detection and AI. Some of your sprites should be smart and react to the player when the player gets close. Use the observer pattern to enable the player to notify NPCs of its position.

3. Build a HUD. One approach is to use the SDL draw facility to make a HUD that is reconfigurable through modifications to your XML file. The Hud must be encapsulated in a class that you design and implement. The player should be able to toggle the HUD with F1. Your HUD should display information about how to move your player object so that we can test your game.

4. Your name printed clearly (font color/size) in lower left screen. You can nuke fps if you like.

5. **video**: submit a short video demonstrating your games features. You can either make the video with a capture program such as *simplescreenrecorder* or use F4.

6. In your README for this project, include a paragraph that describes the game level that you would like to build for your final project. Provide some details about actions in the game, your sprite source, how you will keep score, and how the game will conclude.

The course repository contains examples of how to implement player movement, collision detection with *Strategy Pattern*, scaling sprites, and using the *observer* pattern to implement AI. Study these examples. As you build your solution for this project, strive for proper C^{++}, and good object oriented principles. Your goal should be to write classes that "take care of themselves."

If you would like to work synergistically with a partner, this project can initiate your collaboration.

The Light at the End: Project #5 will entail incorporating projectiles and shooting, explosions, object pooling, sound, and music. The final project, Project #6, will require that your game reach a conclusion, and you incorporate more pizazz. You could incorporate a menu, a health meter that appears or disappears at strategic times in the game, *Painter's Algorithm*, etc.

For the remaining projects, you will use your tracker framework. Obviously, you can make any modifications or extensions to the tracker framework to accommodate the game that you intend to build.