Games Programming 1

Aisha Saleem

(S1320089)

Third Year Computer Games Design

I confirm that the code in this file (other than provided or authorised) is all of my own work and has not been submitted elsewhere in fulfilment of this or any other award.

Reference Section

There was no references made outside of the resources that the lecturer of the module had given us, all the materials had been given by him. Some of the images were made by myself in Clip Studio 5 and I had found the music on a royalty free website.

Code Explanation

To start the program off the windowOGL class would handle the viewport by resizing the window and the rendering of the window. As that class is working cWNDManager deals with incoming messages and how the interpret the messages if the window is active. Another class, the cTexture class takes the image and transforms it into a texture that can be used inside the game, it can get the height and width of an image and it can return the image as well.

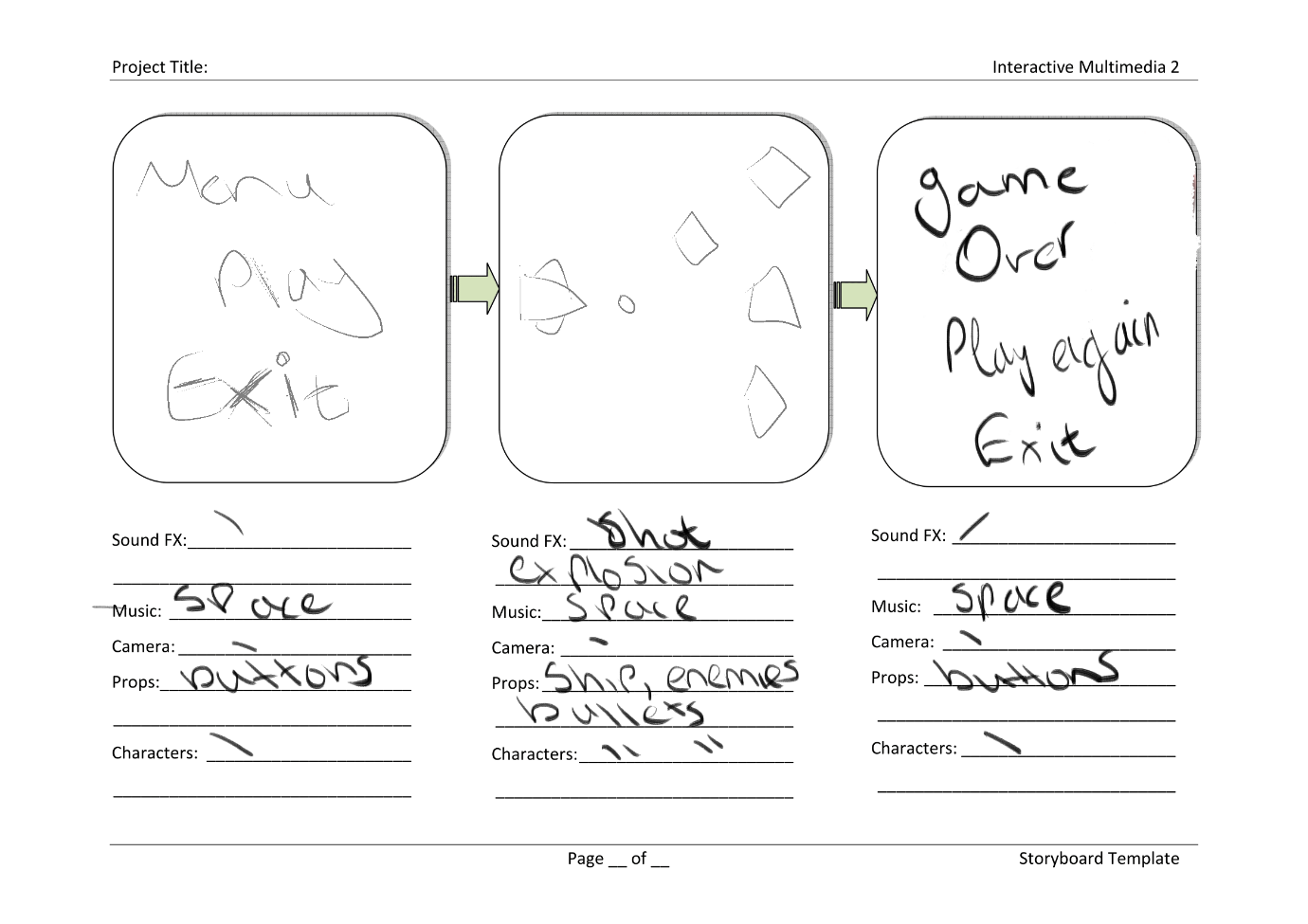
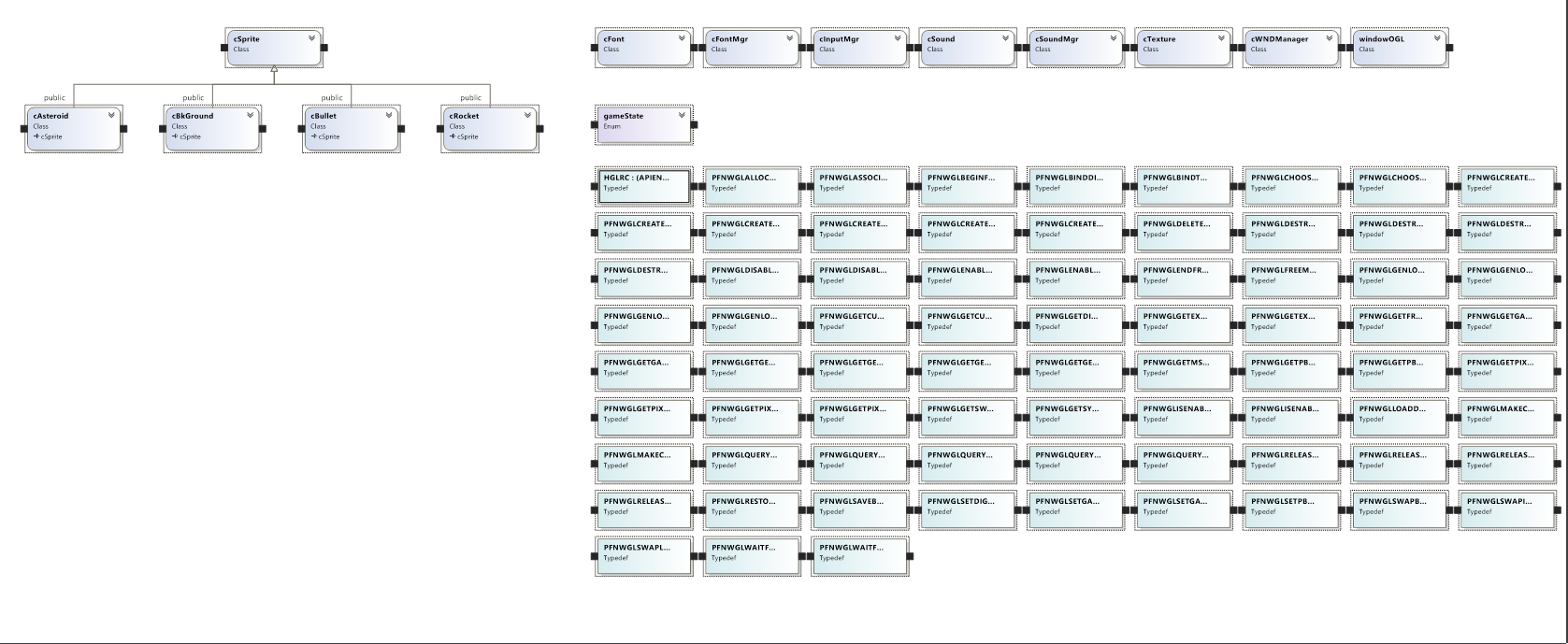
Another class that was used was to handle the bullets that were fired by the rocket. It can get and return the bullet velocity, setting a bounding box and making the bullet faster over time is just some of the few things it can do, another thing is that it has a render like other classes. The cRocket class is like the bullet in the way it can handle the velocity of the rocket and collisions along with the bounding box, it also includes the coliision between the bullets and the asteroids using something called iterators and making them inactive if the asteroids and bullets hit. The cAsteroid class is exactly like cBullet class.

The classes that were mentioned in the above class are all a part of the cSprite class, you could say they were “children” of it, although there was some differences. First of all the textures would be that to get and set the texture it would need to centre it and then halve its width and height. It also sets the coordinates of it and allows the input and sound managers to be attached to sprite.

There are three managers in the code, sounds, input and a font manager, the sound manager would add something called wav files and it would either be returned or deleted. The input manager clears buffers, keys pressed and bools to see if there are mouse buttons are being pressed, after that it updates the keydown and keypressed when a key is pressed and gets the mouse position on the screen. The font manager adds font from a file and it returns so that it can be used by iterators and it can be deleted the same way.

There is a cSound class which is similar to the sound manager class, this class sets everything for the sounds, like its frequency, buffer, length and bitrate. The sounds loads, becomes available for playing, plays the sound and it can be stopped/cleaned up.

The main class has everything it needs to make the game run, it adds all of the managers attaches each part to a correct part and then it creates a window based on the classes that have been mentioned at the beginning. After this had ran it would then create the asteroids at random areas and it would load the texture for them. Once the game has ended the program returns to zero to indicate success and it will close.

Class Diagram

Storyboards