**Saif Sattar**

**S1313176**

**Computer Games (Software Development)**

**Year 3**

**Games Programming 2 Report**

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

S.Sattar

1. AudioSound .h and .cpp files

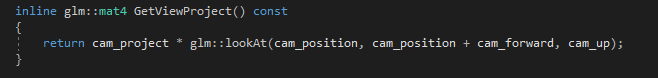
First of all in the header files two methods were created to handle data: isBigEnd () and convertInt (). convertInt () is used to simply convert a character array to an integer array as some of the sound data is stored as character data and Open AL uses integers as parameters.

Other methods were created in the header file for loading audio, deleting audio, playing audio, stopping audio and setting audio listener.

loadSoundWAV () is used to read in a WAV file in a way that allows for playback using Open AL. loadAudio () takes in a file from a pathway and load it and from there works out the type of audio data, sample rate, mono/stereo, etc then saves the data in a buffer ready for use. playAudio () plays a sound by calling the integer handler and the 3d position that the sound will play from whereas stopAudio () stops the sound. setAudioListener () sets the position of the audio listener and where the listener is looking.

1. Cam .h file

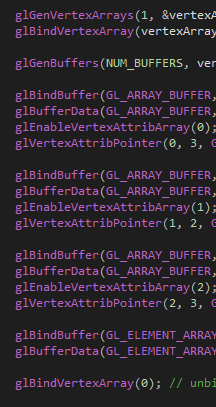
The method startCam () initialises the camera. This method takes in a vector 3 position, a float for the field of view, a float for the aspect ratio, a float for the near and clipping plane and a float for the far clipping plane.

A projection matrix was also created:

The private variables cam\_project, cam\_position, cam\_forward and cam\_up were also created.

1. Mesh .h and.cpp files

The methods that were created were drawing (), start (), loadingModel (), startModel () and updateSphereData ().

Open GL is used in the startModel () method.

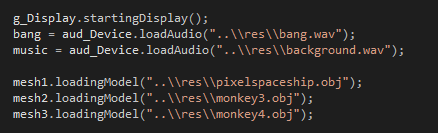
A vertex array is generated and stored in the vertex array object. The vertex array object is then bound using glBindVertexArray. glGenBuffers generates the buffers based on the array of data/buffers. Open GL is told what type of data the buffer is and passes the data. The data is moved to the GPU. At the end the vertex array object is unbound.

1. ScreenDisplay .h and .cpp files

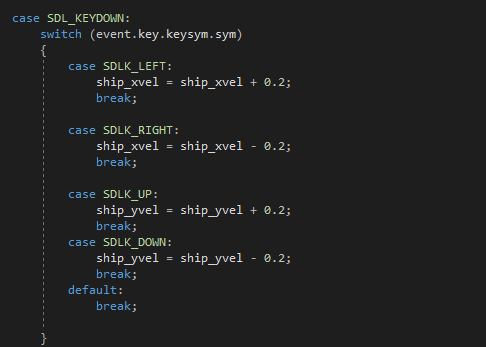
Methods are created for staring the display, swapping the buffer and clearing the display. Float variables to hold the screen width and screen height were also created.

1. Game .h and .cpp files

Methods were created for staring systems, processing inputs, the main game loop, the game drawing and playing audio. A Boolean was created for collision. The meshes were referenced for the three game objects as well as the camera being referenced. Integer variables were stored for the musing and bang sound effect. Float variables were stored for the ship’s (playable game object) x and y positions. This was for the extension portion of the project which was to allow control of model using user inputs.



Each mesh was assigned to their respective model. For the extension portion, user inputs were implemented.



Using SDL2, a switch statement is used on key down. Using SDLK\_LEFT, SDLK\_RIGHT, SDLK\_UP and SDLK\_DOWN, the player is able to move the model left, right, up and down to avoid the two moving objects.

References

* Space ship model - https://www.turbosquid.com/FullPreview/Index.cfm/ID/1063711
* Other models – GCU Learn