Scenes

This lab tutorial introduces the reader to the Scenes implementation. Use your code from the previous lab.

1. Overview

Look at the **OnUpdateFrame()** and **OnRenderFrame()** methods in the **SceneManager** class. This is where all of the relevant stages of the game loop are executed. Notice that they simply call the delegate method that they currently point at. The methods pointed to by the delegates will be contained inside of a Scene class (e.g. MainMenuScene or GameScene).

The **SceneManager** class currently has a method called **StartMenu()** that creates a new instance of the **MainMenuScene** class – this is called from the **GameScene** when the 'M' key is pressed.

The **SceneManager** class also currently has a method called **StartNewGame()** that creates a new instance of the **GameScene** class – this is called from the **MainMenuScene** when the left mouse button is pressed.

These two methods were implemented as a prototype to see how to switch scenes – this needs to be improved.

2. Exercise 1

1. You may have noticed that the models render very quickly when the game starts for a second time. This is because the **ResourceManager** stores the assets in memory. If you were to create a new game level, the old assets would still consume memory. There is a **RemoveAllAssets()** method in the **ResourceManager** but it is not called.
2. Therefore add code to call the **RemoveAllAssets()** method in the **Close()** method of **GameScene**.

3. Exercise 2

The switching of Scenes is handled in the current active Scene, but the Scene requires knowledge of the required method to call in the **SceneManager**. If we want a new Scene then we need to add more methods to **SceneManager** etc.

Please attempt these exercises, but if you get stuck or you are confused then ask for help during the scheduled lab times.

Add the following functionality to your software (in the appropriate classes or create new classes).

1. Add a new method called **ChangeScene()** to **SceneManager** that can be called by each Scene that will be used to change scenes. It is proposed that the method protype look like the following.

   ```
   public void ChangeScene(SceneTypes sceneType)
   ```

   It is proposed that you create a new **enum** in **Scene** to identity each scene as below.

   ```
   enum SceneTypes
   {
       SCENE_NONE,
       SCENE_MAIN_MENU,
       SCENE_GAME
   }
   ```

2. In the **MainMenuScene** scene edit the mouse button method. Add code into this method to call the **ChangeScene()** method in the **SceneManager** to change the scene to the **GameScene**.
3. Add code to the keyboard method in **GameScene** to call the **ChangeScene()** method in the **SceneManager** when the ‘M’ key is pressed to change the scene to the **MainMenuScene**.

4. Exercise 3

1. Create a **GameOverScene** scene class that can be used to display a game over scene (you can copy the code from **MainMenuScene** and change the code appropriately).
2. When the game ends, the **GameScene** should start **GameOverScene**. Given that the 'game' does not end yet, simply use the 'M' key.
   Add appropriate code to handle this. The **GameOverScene** should start the **MainMenuScene** when any key is pressed.
   NOTE: Remember to add and remove any keyboard delegate in the constructor and **Close()** method and add a **SceneType** in **Scene**.