Unity API (UI Button, SceneManager, PlayerPrefs, Static Classes)

LAB # 8

Fall 2024

CSE-411L Intro to Game Development Lab

Submitted by: **Ali Asghar**

Registration No.: **21PWCSE2059**

Class Section: **A**

"On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work."

Submitted to:

Engr. Abdullah Hamid

Date:

25th December 2024

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

Objective:

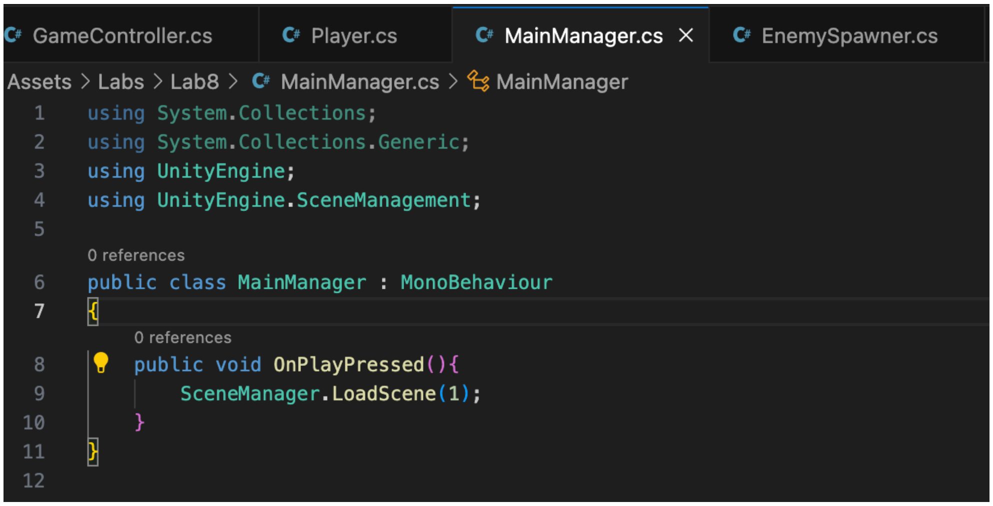
In this lab we further explored the Unity API.

Tasks:

1. Open or create a Unity scene.
2. Create a simple UI with a Play button. When pressed, the current scene should close, and a new scene (the game from the previous lab) should open.
3. In the previous lab's scene, add a new text box to the UI to display the high score.
4. For score management, create a new class called ScoreManager.cs. In this class:
    a. Define a static integer for the score.
    b. Add non-static references for highScore and score.
    c. Ensure this class manages both the score and the high score.
5. Include a static void function in the ScoreManager.cs class to be called when the game is over. This function should handle high score logic, updating the high score if the current score is higher.
6. In the game scene (from the previous lab), the Game Over panel should include
    a. two additional buttons:
    b. Exit: Returns to the scene with the Pl

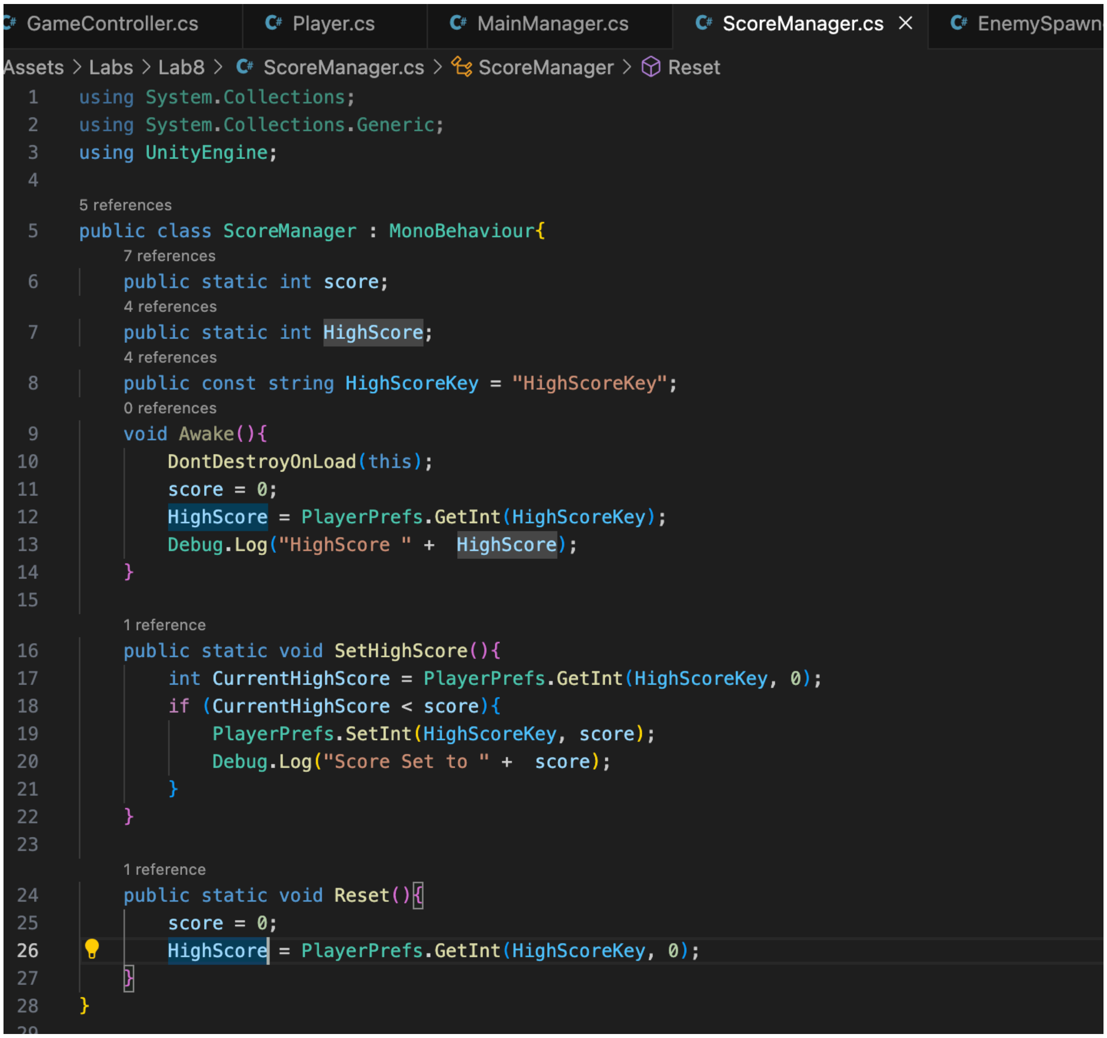
Code:

MainManager class



```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class MainManager : MonoBehaviour
{
    public void OnPlayPressed(){
        SceneManager.LoadScene(1);
    }
}
```

ScoreManager class

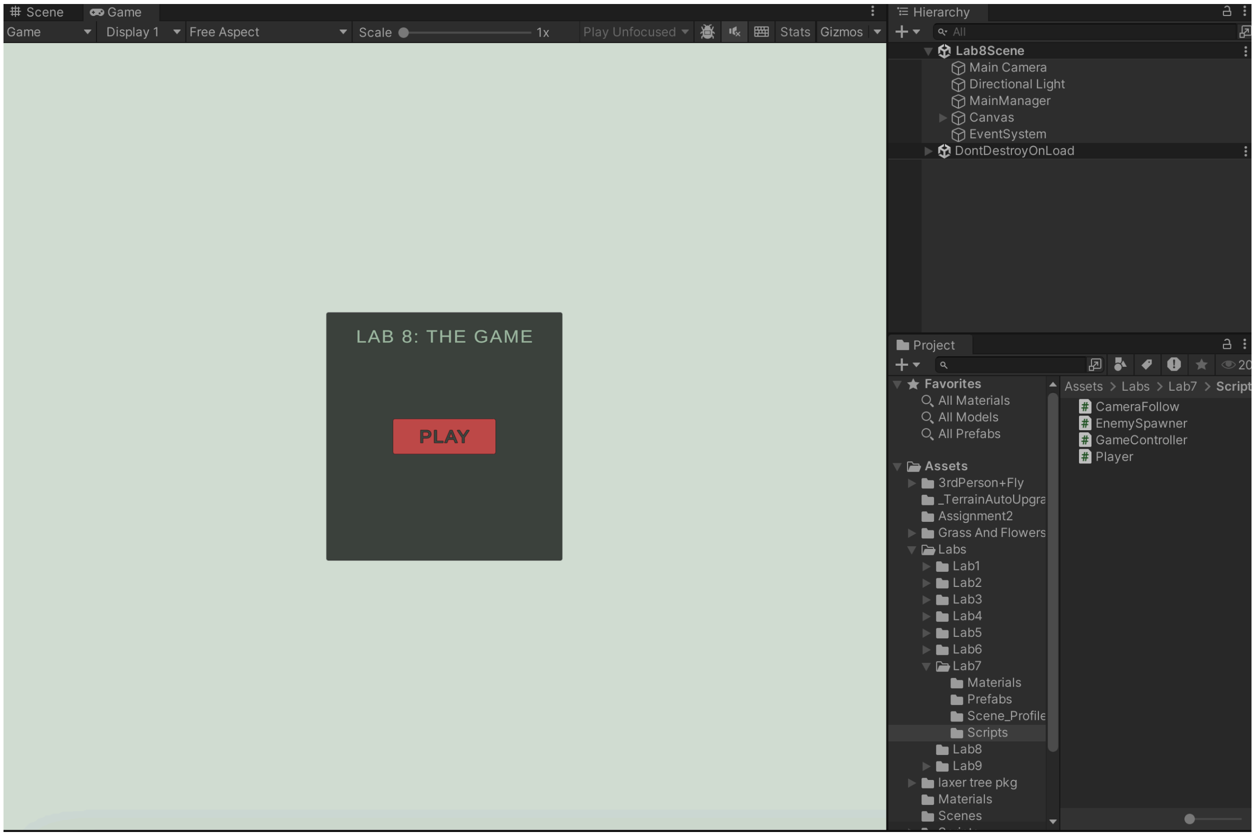


```csharp
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

5 references
public class ScoreManager : MonoBehaviour{
    7 references
    public static int score;
    4 references
    public static int HighScore;
    4 references
    public const string HighScoreKey = "HighScoreKey";
    0 references
    void Awake(){
        DontDestroyOnLoad(this);
        score = 0;
        HighScore = PlayerPrefs.GetInt(HighScoreKey);
        Debug.Log("HighScore " +  HighScore);
    }

    1 reference
    public static void SetHighScore(){
        int CurrentHighScore = PlayerPrefs.GetInt(HighScoreKey, 0);
        if (CurrentHighScore < score){
            PlayerPrefs.SetInt(HighScoreKey, score);
            Debug.Log("Score Set to " +  score);
        }
    }

    1 reference
    public static void Reset(){
        score = 0;
        HighScore = PlayerPrefs.GetInt(HighScoreKey, 0);
    }
}
```

GameController class

This script is the same as lab 7

Output:



The Gameplay scene is the same as lab 7