

**Submitted By:**Ashna Wasif (2791)  
Nimra Asif (2813)  
Nimra Ali (2828)

**Class**: BSCS 7B (Morning)

**Subject**: Computer Graphics

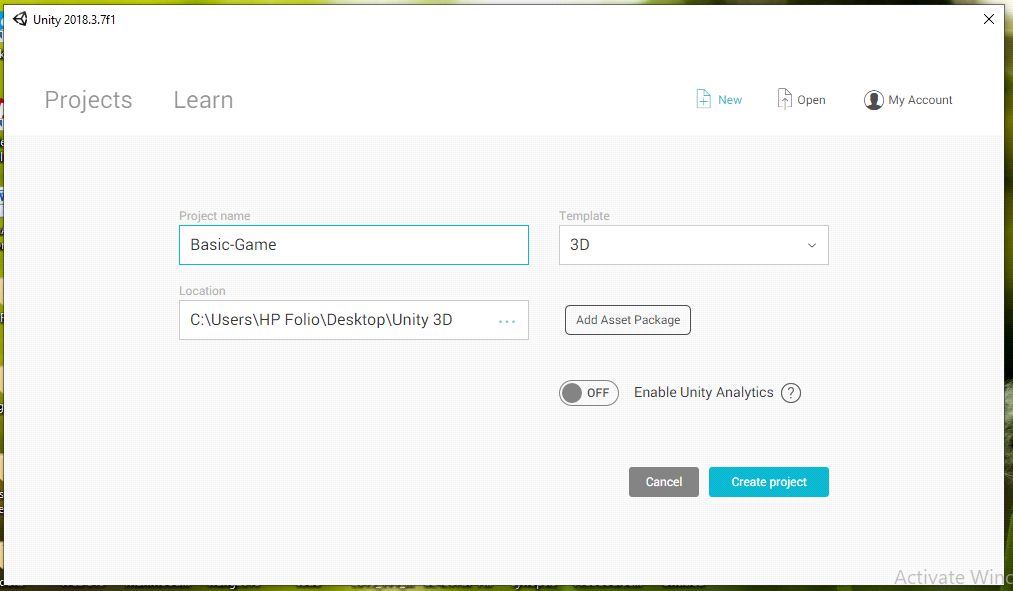
**Submitted To:** Mam Sabina Irum

**Date:** 2nd May, 2025

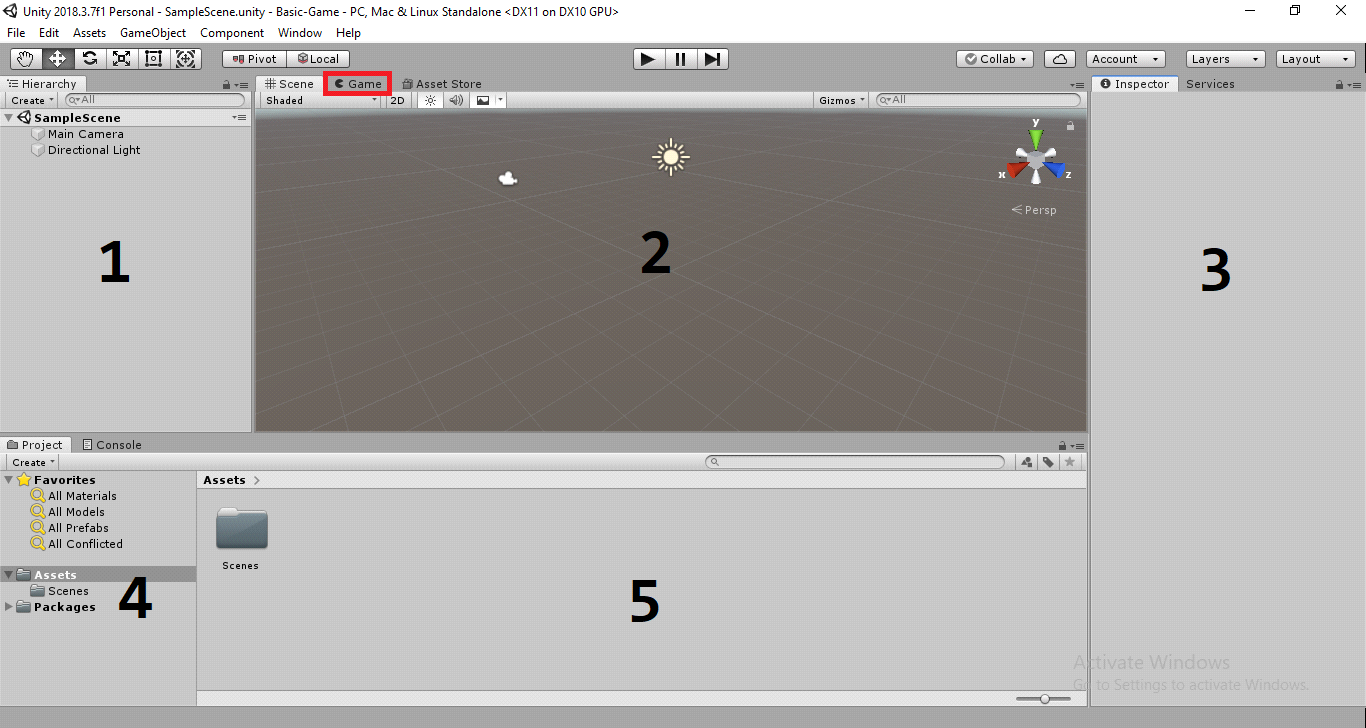
**Unity Assignment 3**

* **Step 1:**

Open Unity and create a new project. The Following screen will appear.



Name the project “Basic-Game” Make sure that the Template is 3D and leave all other setting as is. Click the Create Project Button and the following screen will appear. Wait for this window to finish installation.

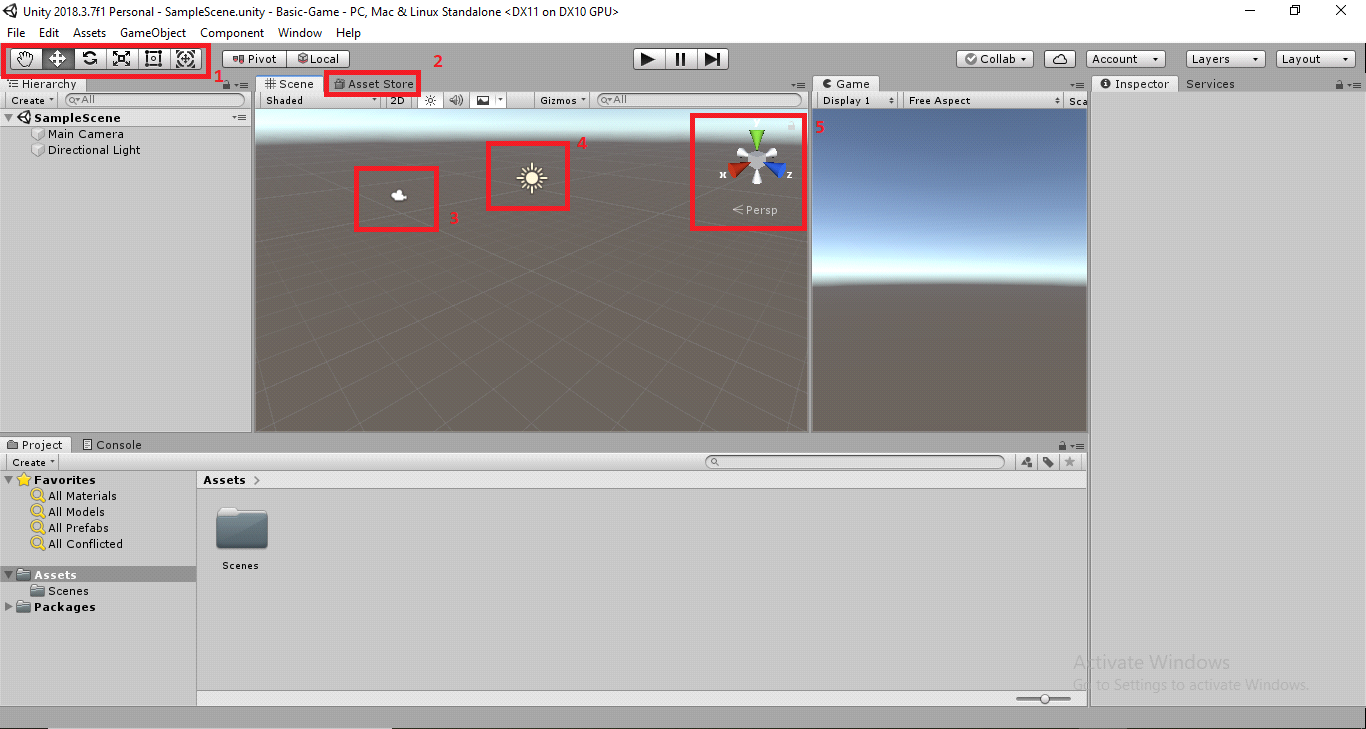


This is the basic Unity Interface. The Main panels are as follows:

* Hierarchy Panel: This panel contains all the Objects etc. in the game scene and the hierarchy of these objects. i.e. the way these objects are linked with each other.
* Scene Panel: This is the main panel where the game is designed. It contains the Camera and the light source by default.
* Inspector Panel: Whenever any object is selected in the scene or the hierarchy panel its properties are displayed in the Inspector panel.
* Projects Panel: The assets panel contains the downloaded assets, materials, prefabs etc used in the project.
* Assets Panels: This panel contains the saved scenes, prefabs, scripts materials etc used in the scene.

The Game tab contains the view of the game from the Cameras perspective. this is how the game will appear once it is exported. Hold the “Game” tab and drag it into the scene area on the right side. This will change the layout as below.

The main elements are as follows:

* The transformation toolbar. This toolbar is used to move, scale and rotate the objects.
* The Asset store is used to download the 2d and 3d Assets.
* The default camera.
* The default Light Source.
* The Gizmo is used to look at the 3d scene from different directions i.e. axis.

To navigate the scene the following control are used.

Pan / Scan: Hold the scene with the middle button of the mouse to rotate it. Alt + Middle mouse button can move the scene too.

Zoom-in/out: The middle mouse button i.e. wheel is scrolled to zoom in and out of the scene. Alt + Right Click can zoom too.

Rotate: The right mouse button is used to rotate the scene (W,A,S,D) to move around the scene like a first person shooter game. Alt +left Click also rotates the camera.

**Creating an Object**

Create an empty object by clicking on the Game Object Menu and then Create Empty. The empty object will appear in the hierarchy panel and the scene. The “Add Component” button in the inspector allows the empty object to have. Delete the empty Object and create a cube as follows:

Game Object -> 3d Object -> Cube.

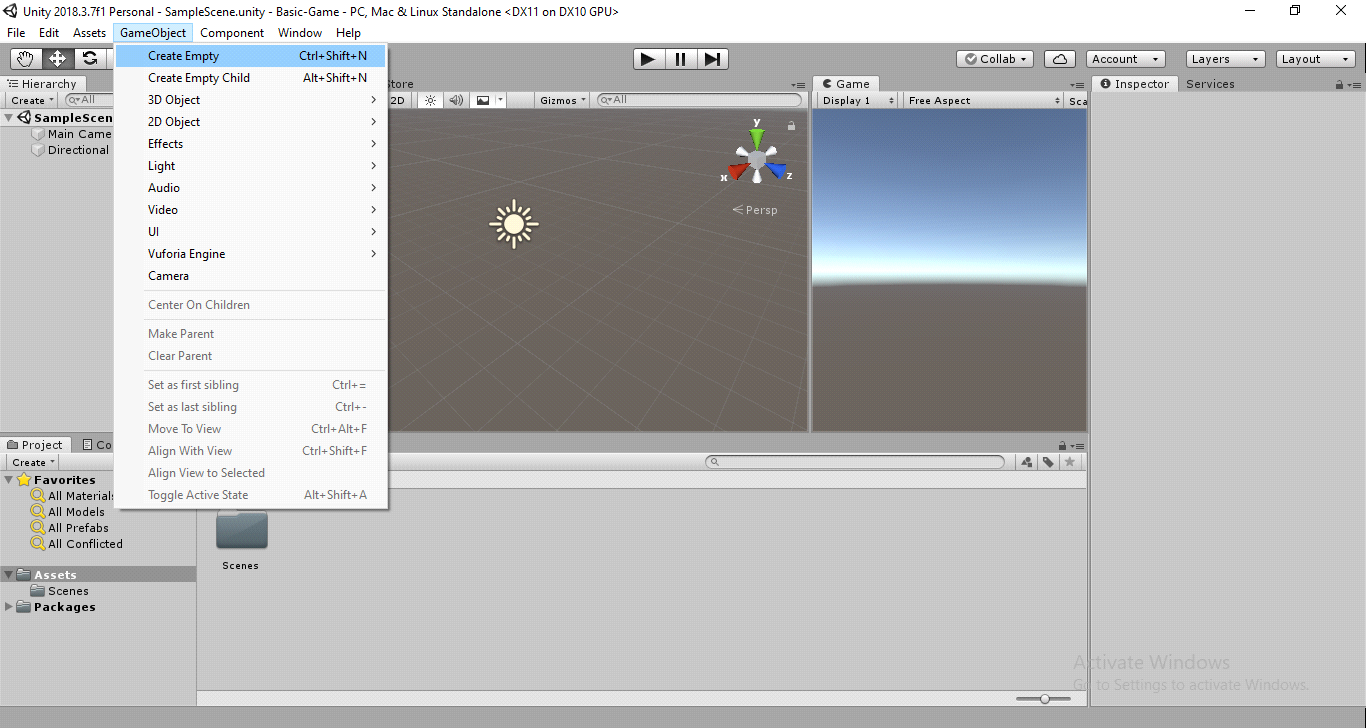
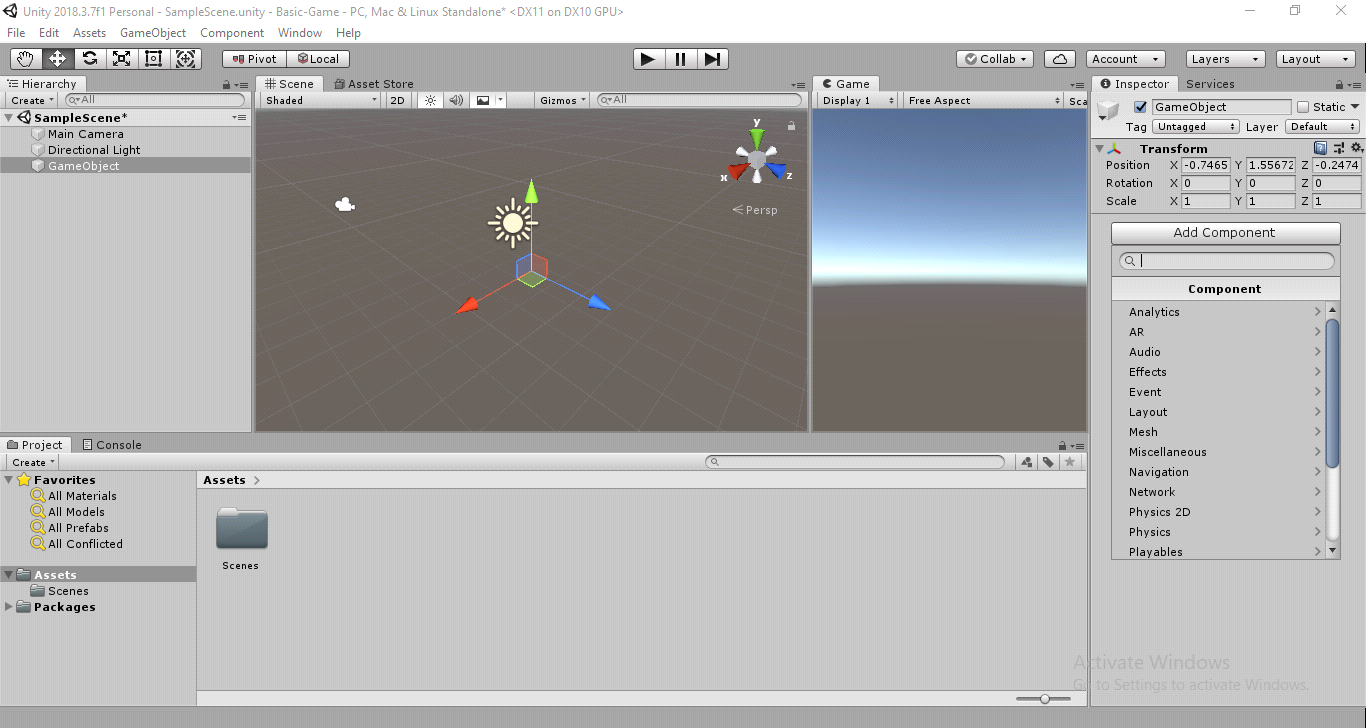
The Cube will look like the one in fig 3 below.

Figure 1

  
Figure 2

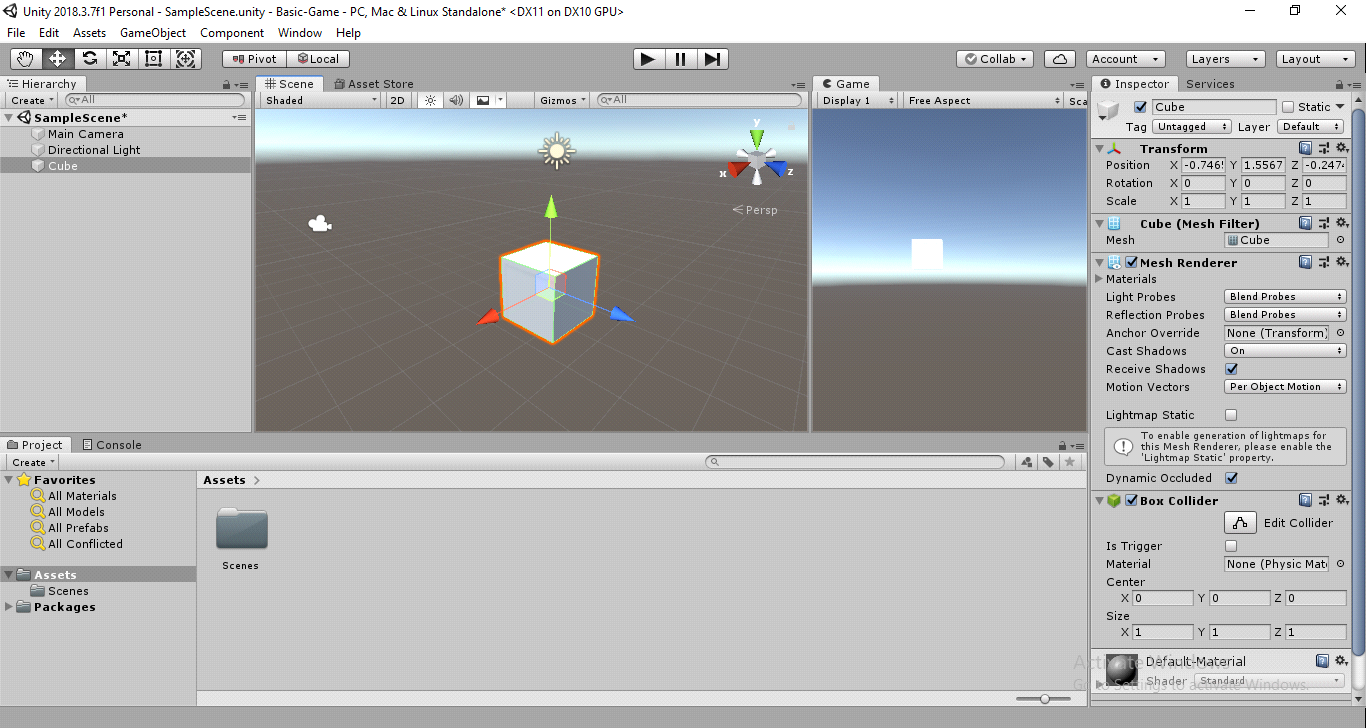
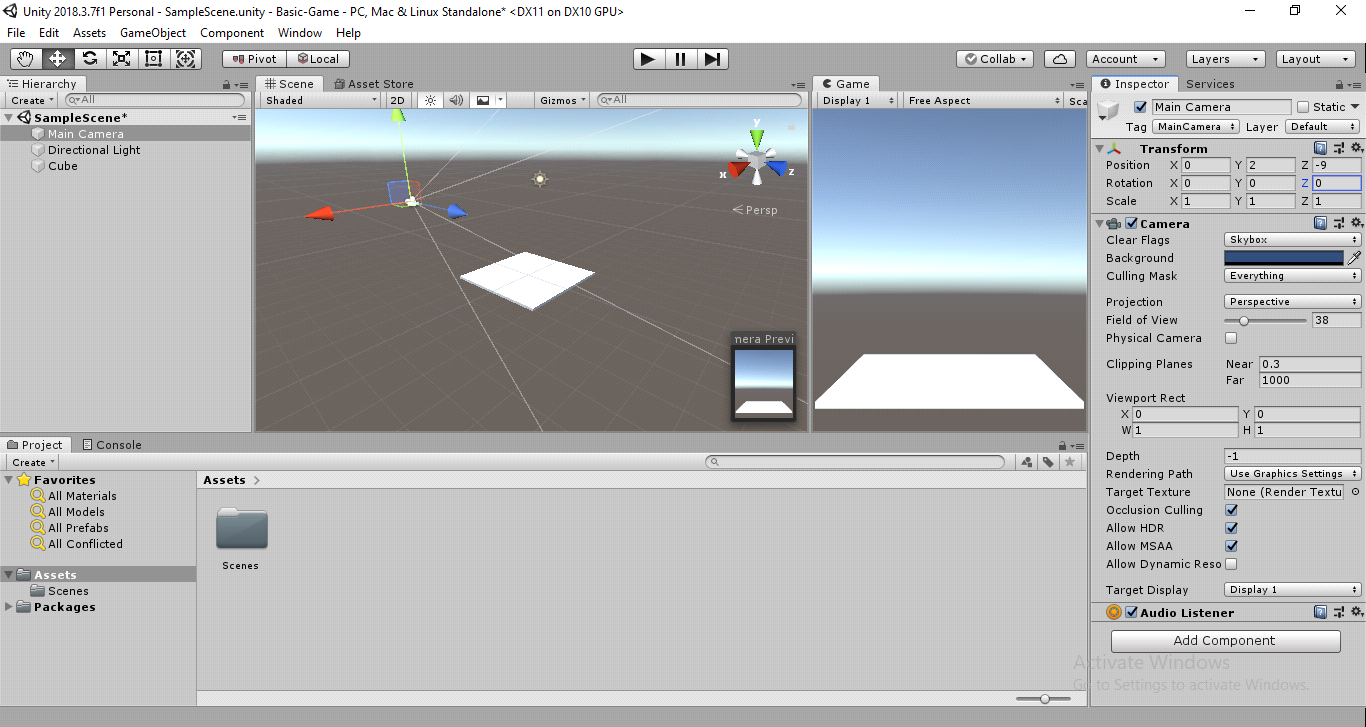


Figure 3

* **Step 2:**

With the cube selected add the following values in the transform panel’s Scale: **x:4, y:0.1, z:4**. The will flatten the cube to what is shown in the screenshot below. Now select the camera and enter the following values in the transform panel’s Position textboxes in the inspector **x:0, y:2, z:-9** and leave the scale settings to 1, 1, 1 and the rotations to 0, 0, 0. Doing this will change the scene and the games panel as below.

Add another cube and make it a Rigid Body by pressing the “Add Component” button and typing rigid in it. The cube will now fall down when the scene is played. Press ctrl + d and duplicate the objects and stack the cubes on top of each other. Now when the scene is played, the cubes will all fall down. Create an empty object in the hierarchy panel and put all the cubes in it.

**Creating Complex objects**

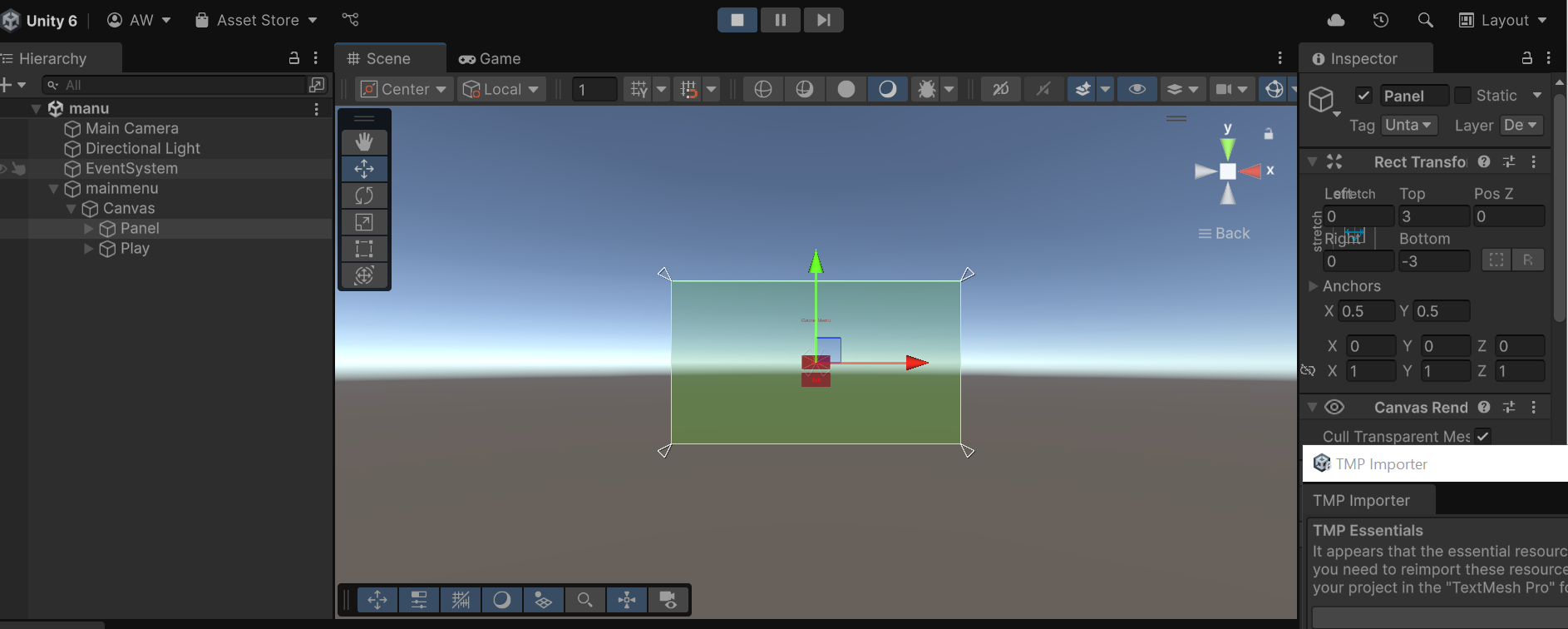
Add a sphere by right clicking in the hierarchy panel and clicking on 3d Object -> Sphere.

- Make the sphere a rigid body.

- Duplicate the sphere by selecting it and pressing ctrl +d.

- Put the duplicated sphere into the original sphere and remove the second spheres rigid body property by disabling it from the inspector. Adding two spheres inside another sphere and then rearranging them will result in something like the screenshot below. Note the arrangement of the spheres in the hierarchy panel. Make sure that only the parent sphere is a rigid body. this is how complex objects are created in unity using the primitives. You can create a caterpillar etc.

- Add color and material to the objects by right clicking inside the project tab and selecting Create -> material. It will create a ball icon in the assets panel. Select that icon and change the color in the inspector.

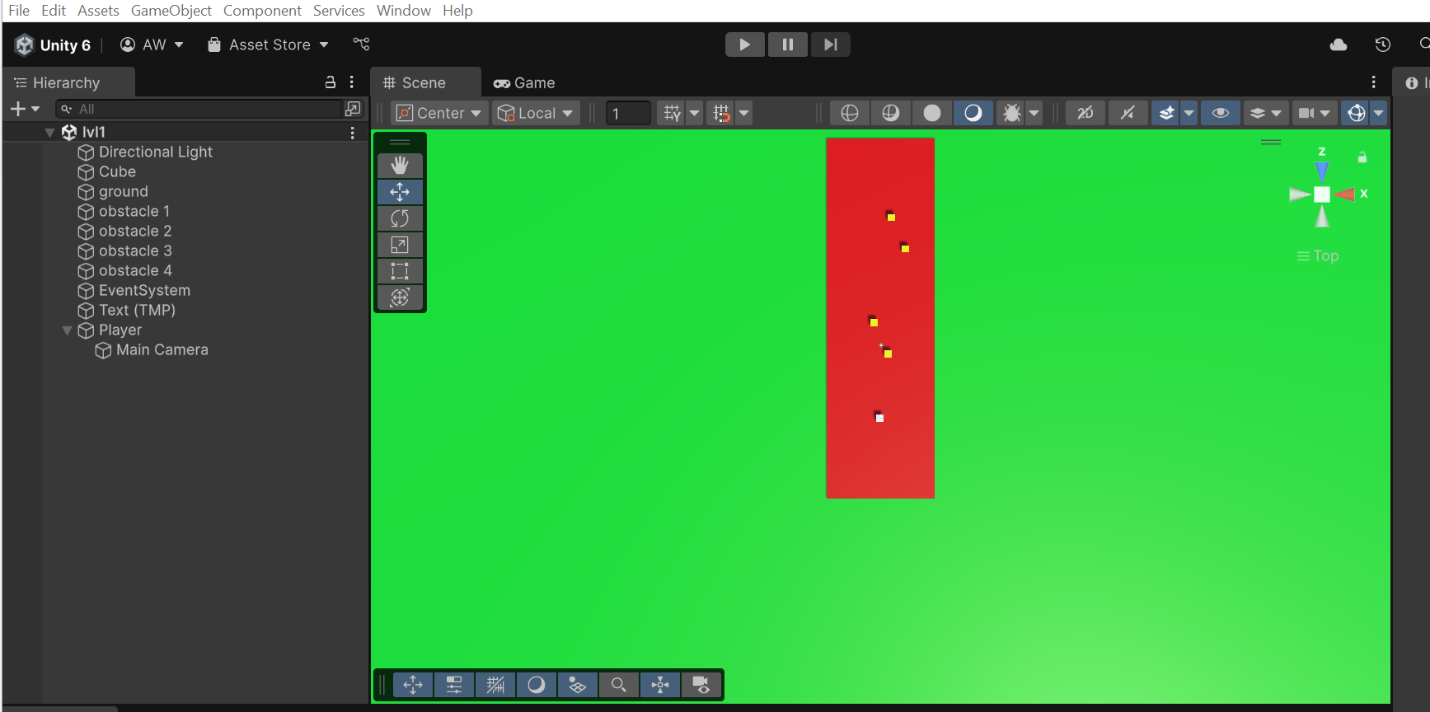
- Drag this material icon onto any object that you want to apply the material to. You can Create multiple materials to form a theme. To apply the material to multiple objects select them and in the mesh renderer drag the material onto the element 0 textbox.

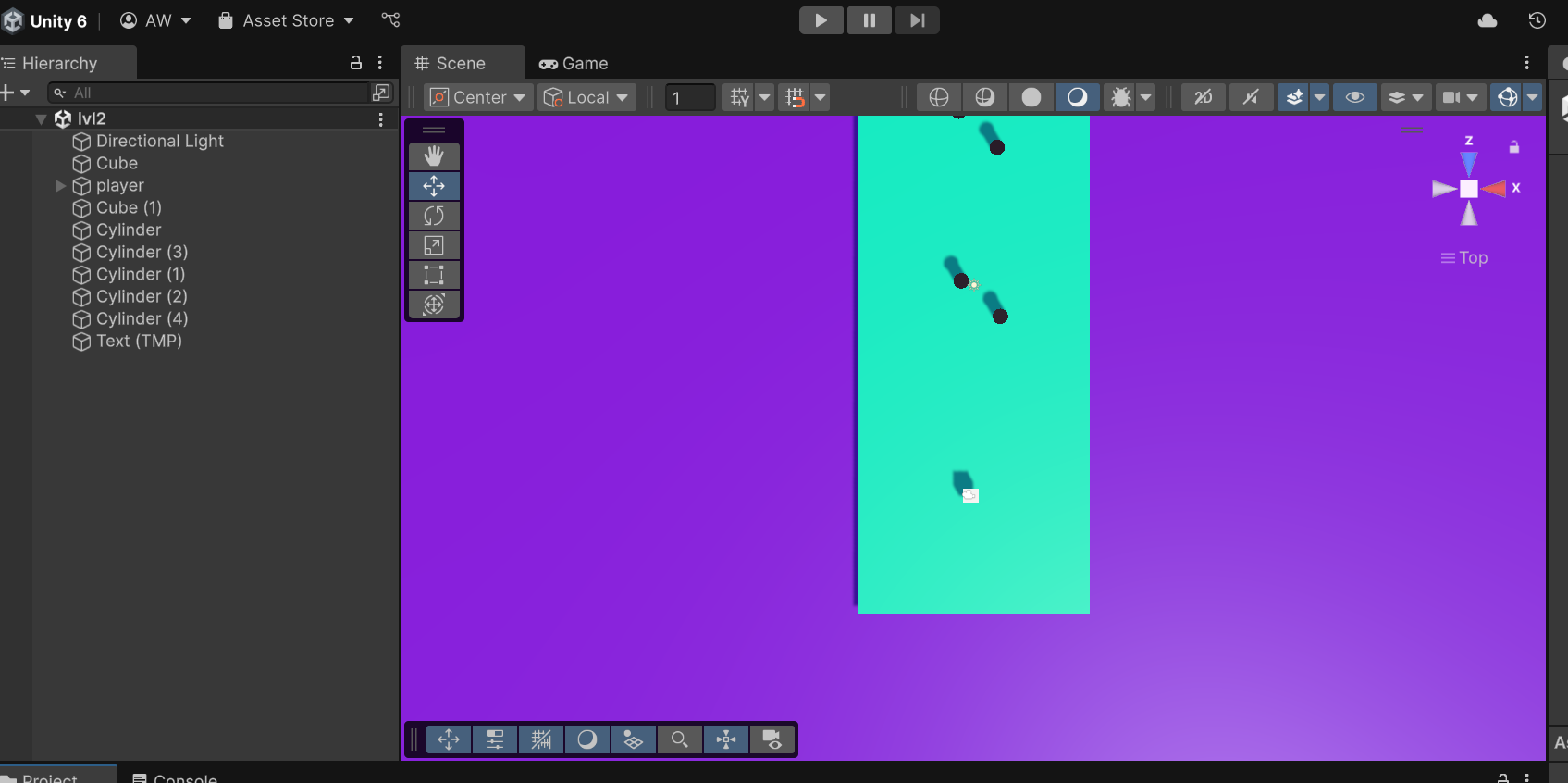
Save the scene and the project as much as possible.

**Level 1:**

The same way, You can add as many levels as you want. Here’s the screenshot of level 1:

**Level 2:**

It can be shown as:



**Starting the game**

* Add a .cs file to add Logic:

|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class CameraController : MonoBehaviour  {      public float moveSpeed = 5f; // Speed of camera movement      void Update()      {          float moveX = Input.GetAxis("Horizontal"); // Left/right arrow or A/D          float moveZ = Input.GetAxis("Vertical");   // Up/down arrow or W/S          Vector3 movement = new Vector3(moveX, 0, moveZ) \* moveSpeed \* Time.deltaTime;          transform.Translate(movement, Space.World);      }  } |

|  |
| --- |
| [RequireComponent(typeof(Rigidbody))]  public class KeyboardControl : MonoBehaviour  {      [SerializeField]      Vector3 v3force = new Vector3(1f, 0, 0);      [SerializeField]      KeyCode KeyPositive = KeyCode.RightArrow;      [SerializeField]      KeyCode KeyNegative = KeyCode.LeftArrow;      private Rigidbody rb;      void Start()      {          rb = GetComponent<Rigidbody>();      }      void Update()      {          if (Input.GetKey(KeyPositive))          {              rb.velocity += v3force;          }          if (Input.GetKey(KeyNegative))          {              rb.velocity -= v3force;          }      }  } |

**Restart Level with Button**

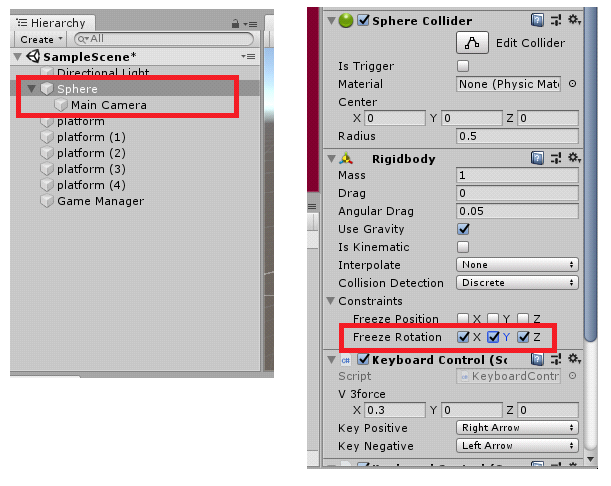
* To restart the level with a button. Create a new script and call is “**ButtonRestart**”. Double click the script and open it in Visual Studio etc. Add the following code to it.

|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  using UnityEngine.SceneManagement;  public class ButtonRestart : MonoBehaviour  {      [SerializedField]      KeyCode RestartKey;      // Start is called before the first frame update      void Start()      {        }      // Update is called once per frame      void Update()      {          if (Input.GetKey(RestartKey))              SceneManager.LoadScene(SceneManager.GetActiveScene().name);      }  } |

* After creating this script. Save it and then create an empty object in the scene and add the ‘**ButtonRestart’** script to it.
* This script adds a package that manages contains functions etc to manage the scene and then it calls the LoadScene() function and reloads the scene that we are already in.
* In Unity now the inspector will show a new field whenever the ‘Game Manager’ object is selected. Select the ‘R’ key to restart the level and play the scene.
* Now whenever the R key is pressed the scene will restart.

**Camera Following the Player**

* The simple way to make the camera follow the ball is to make it the sphere’s child. To do that just drag the camera and put it into the sphere. The camera will now follow the player but the problem is it will also follow its rotation to avoid the rotations can be disabled from the constraints panel but this is not an optimal solution.



* A better way to do this to add a script to follow the sphere to the camera. So fix the hierarchy of the camera and remove the freeze rotation constraints and go to the script folder, add a new script and call it “CopyPosition”. Double click and open the script in Visual Studio etc.
* Add the following code to the script.

|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class CopyPosition : MonoBehaviour  {      [SerializeField]      Transform TransTarget;      // Update is called once per frame      void Update()      {          transform.position = TransTarget.position;      }  } |

* In Unity drag the script onto the camera and in the **TransTarget** field drag and drop the sphere from the hierarchy panel. Now the **TransTarget** object targets the sphere and the transform object targets the camera.
* Now when the scene is played the camera will follow the sphere but the problem is it will be positioned inside the sphere.
* To fix this issue create a new empty object and place it exactly where the sphere is placed. Call this object “CameraRig”. Make the camera the child of the CameraRig and remove the script from the camera and apply it to the CameraRig object. Now the camera follows the player from a distance.
* Now build a complete level and move the ball around it.
* To load a new scene on collision the following script can be used.

|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  using UnityEngine.SceneManagement;  public class loadNewLevel : MonoBehaviour  {      [SerializeField]      string strTag;      [SerializeField]      string strScene;      private void OnCollisionEnter(Collision collision)      {          if (collision.collider.tag == strTag) {              SceneManager.LoadScene(strScene);          }      }  } |

**Restart Automatically on Falling**

* To restart automatically on falling create a large ground underneath the level first.
* Duplicate the script ‘ButtonRestart’ make the following modifications.

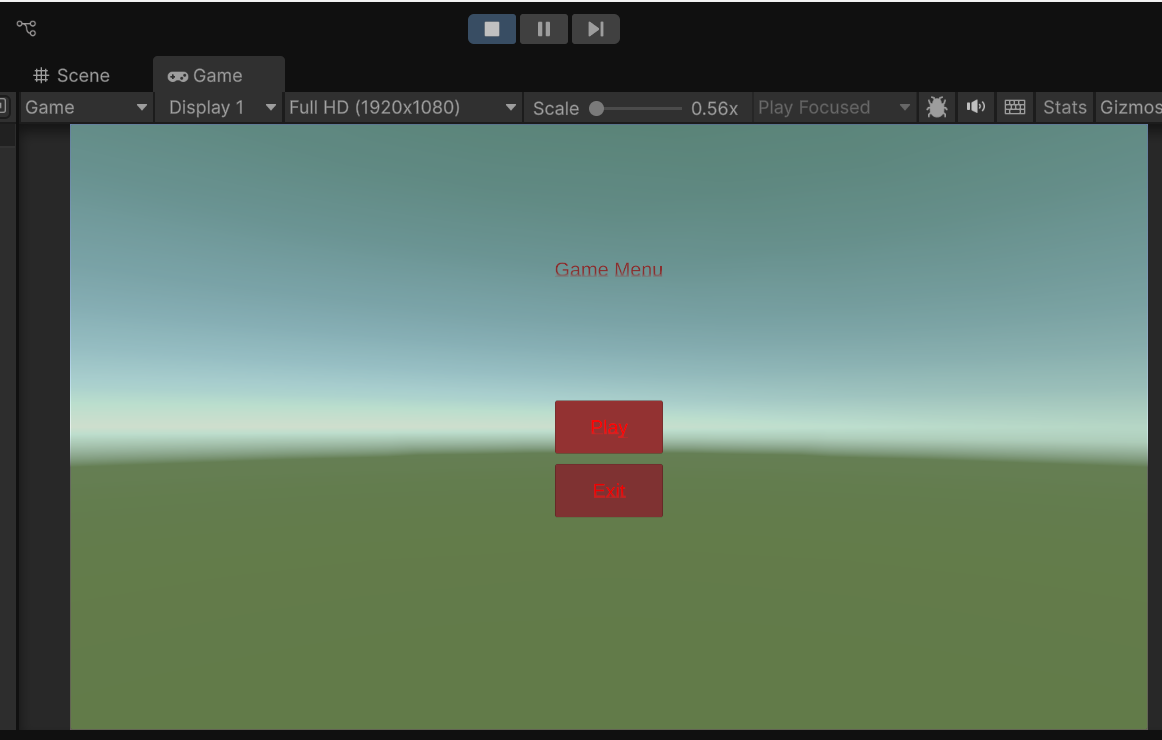
|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  using UnityEngine.SceneManagement;  public class CollisionRestart : MonoBehaviour  {      [SerializeField]      string StrTag;      // Update is called once per frame     private void OnCollisionEnter(Collision collision) {          if(collision.collider.tag == StrTag )                  SceneManager.LoadScene(SceneManager.GetActiveScene().name);     }    } |

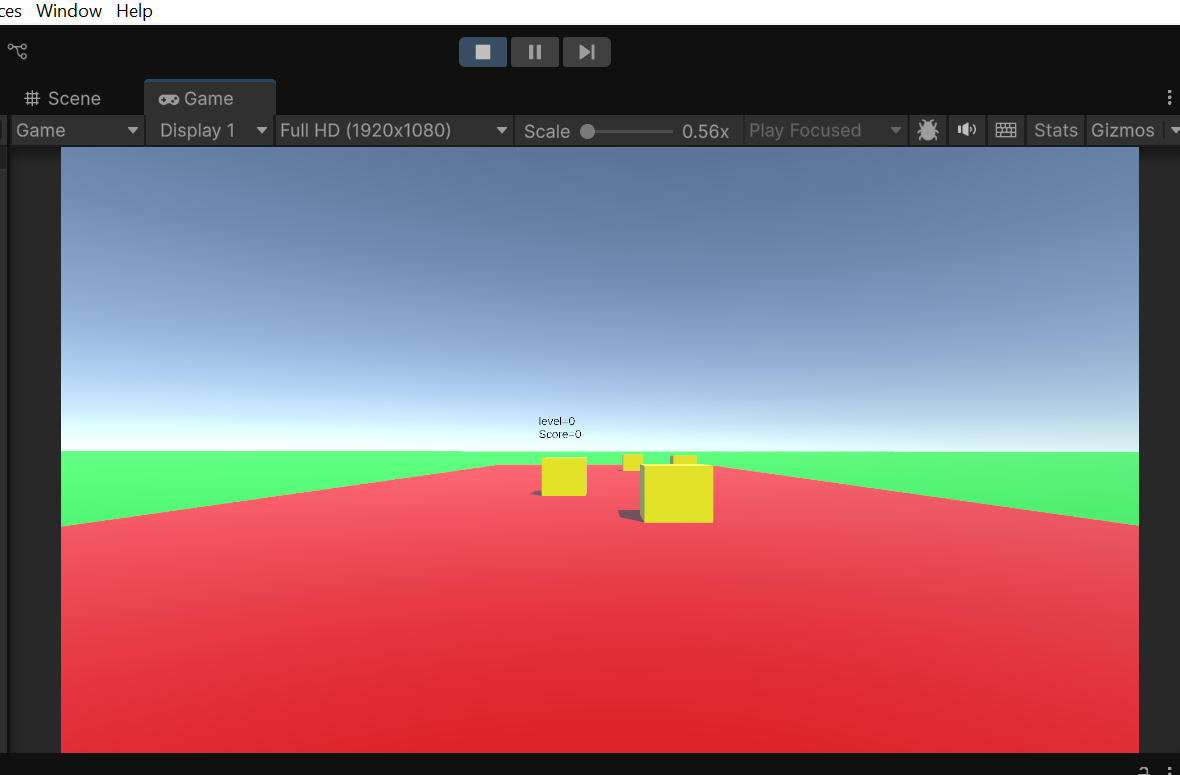
* This code will now look for ‘Player’ tag to reload the game. After this go to Unity. Select the sphere and give the sphere the ‘Player’tag.

**Export the File**

* To Export the file, go to File -> Build Settings
* Add the open scenes there and select the scene that was saved earlier.
* Click on the ‘Build and Run’ button and build then Run your game.

The final Screen Shots are:

**Main Menu:**

**Level 1:**

**Level 2:**

