**Workshop 1 lab**

## Install Unity3D

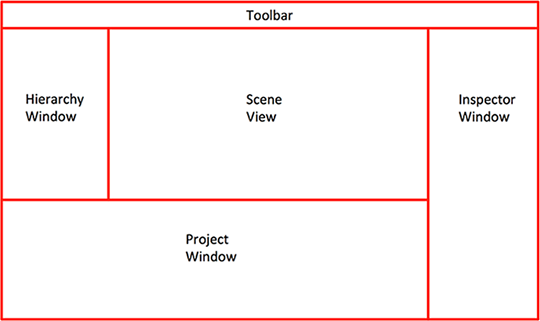
Follow the steps below to download and install the latest Unity.

* + - Go to <https://unity3d.com/unity>
    - Click on Personal
    - Scroll down and click on Try Personal
    - Click on Download Unity Hub
    - Install the latest Unity using Unity Hub

Useful links: <https://docs.unity3d.com/Manual/GettingStartedUnityHub.html> <https://docs.unity3d.com/Manual/GettingStartedInstallingHub.html>

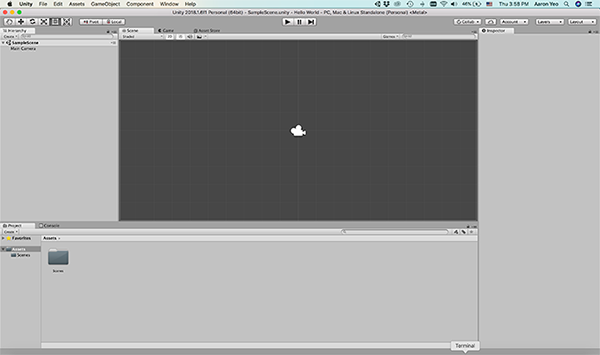
## Unity3D (IDE) Interface Overview

The Unity Integrate Development Environment (IDE) consists of 5 major components: The Project Window, The Scene View, The Hierarchy Window, The Inspector Window and The Toolbar. The diagram shows the default layout of Unity IDE.



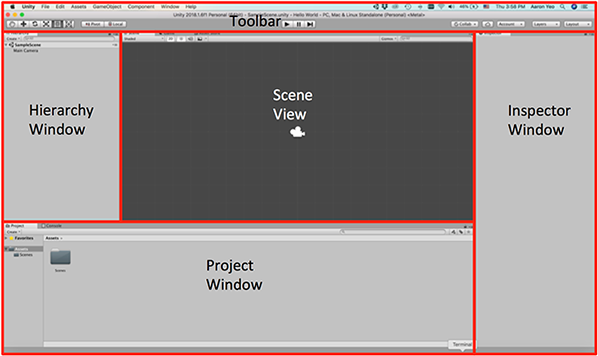
Unity3D IDE Layout

This is what you will see when you first set up a new Unity project.



Unity3D IDE Initial Layout

Note the placement of each window type in the IDE.



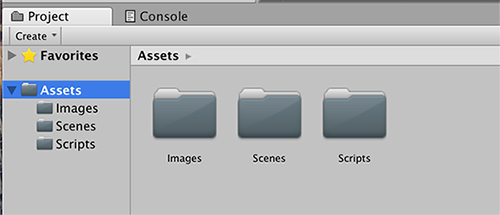
Unity3D IDE Initial Layout

The Unity IDE is made up of tabbed windows that can be rearranged, grouped and docked. You may rearrange each window according to your preference.

Next is a brief description of each window:

### **The Project Window**

The Project Window displays all the assets (e.g. codes, images) in the current project.

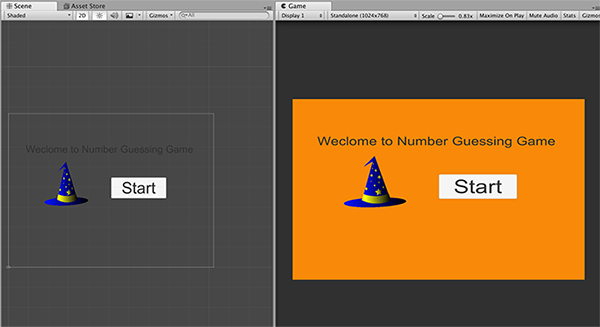


Unity3D Project Window

Useful link: <https://docs.unity3d.com/Manual/ProjectView.html>

**The Scene View**

The Scene View allows the developer to navigate and edit the game view visually. Depending on the project types (2D or 3D), the scene can be viewed in either 2D or 3D.



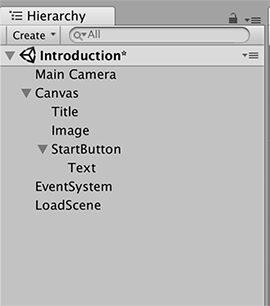
Unity3D Scene View

Useful link: <https://docs.unity3d.com/Manual/UsingTheSceneView.html> <https://docs.unity3d.com/Manual/GameView.html>

### **The Hierarchy Window**

All game objects available in the scene will be displayed in a hierarchical text structure in the Hierarchy Window.

It allows the developer to have a visual view of how game objects are linked together.



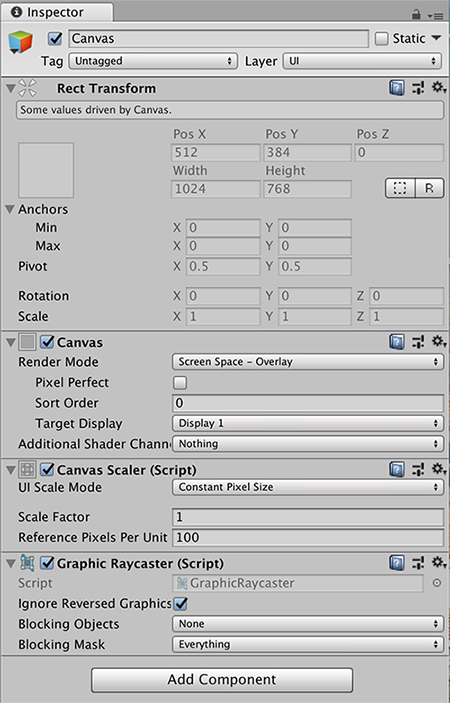
Unity3D Hierarchy Window

Useful link: <https://docs.unity3d.com/Manual/Hierarchy.html>

### **The Inspector Window**

The Inspector Window allows the developer to view and edit the properties of the currently selected game object.

As different types of game objects have different properties, the layout of the inspector will change according to the type of the selected game object.



Unity3D Inspector Window

Useful link: <https://docs.unity3d.com/Manual/UsingTheInspector.html>

### **The Toolbar**

The Toolbar is separated into three parts: left, centre and right.

To the left of Toolbar, the tools for manipulating the scene view and the game objects in the scene view are shown.

The centre of Toolbar contains buttons for play, pause and step control of the game.

To the right of the Toolbar, items such as Unity Cloud Services, Unity Account, Layer Visibility and Editor Layout menu are seen.

The position of toolbar is fixed and it cannot be rearranged.



Unity3D Toolbar

Useful link: <https://docs.unity3d.com/Manual/Toolbar.html>

# Activity 1: Creating a Simple Game

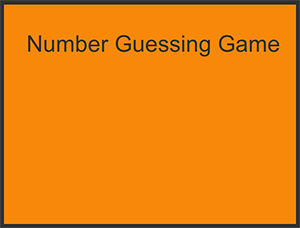
In this activity, we will create two simple 2D games; Number guessing game and Anagram game using Unity.

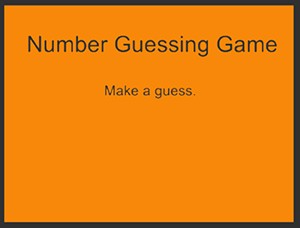
## Activity 1: Number Guessing Game

In this game, a secret number is randomly generated. The gamer is to guess the number. A hint (higher or lower) will be fed back to the gamer for every guess. The game ends when the gamer made the right guess.

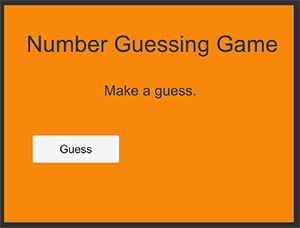
#### Procedure:

* + 1. Start Unity Hub
    2. New a project and name it Number Guessing Game. Under Template, select 2D.
    3. Under Project window, go to the assets->scene
    4. Rename the Sample Scene to NumberGuessingGame
    5. Change the game screen solution
       1. Go to Scene View-> Game panel.
       2. Change the game screen solution to “Standalone (1024x768)”.
    6. Add in the game title in the game scene canvas
       1. Go to Hierarchy window, right click on NumberGuessingGame.
       2. Select GameObject->UI->Text.
       3. Change the object name to “GameTitle”.
       4. Change the text to “Number Guessing Game”.
       5. Adjust the Text object on the canvas to make it visible.
       6. Click on Play in the Toolbar to test out.
       7. You should get something like below. (You may change the Main Camera background colour to change the background colour of the game canvas.)

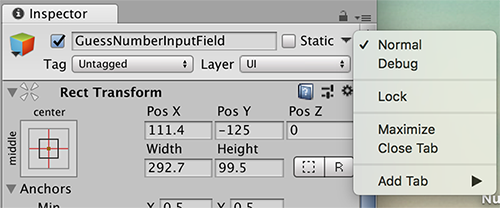


* + 1. Add in another UI Text object with name “Message” and default text “Make a guess”. You should get something like below.
    2. Add a button

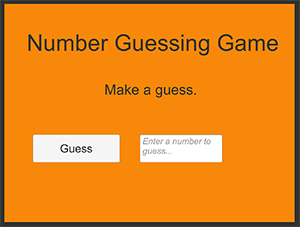
1. Right click on Canvas, select UI->Button.
2. Rename the button object name to “GuessButton”.
3. Change the text on the button to “Guess”.
4. Re-adjust the button object and text to a suitable size.
5. Click on Play in the Toolbar to test out.
6. You should get something like the below.



* + 1. Add an input field
       1. Right click on Canvas, select UI->Input Field.
       2. Rename the InputField name to “GuessNumberInputField”.
       3. Re-adjust the InputField object size to a suitable size.
       4. Change the default PlaceHolder text in the InputField to “Enter a number to guess…” and the text size to 30. (Re-adjust the InputField object size if necessary).
       5. Change the text size in the InputField to 30.
       6. In the Hierarchy window, click on “GuessNumberInputField”.
       7. Go to the Inspector window, on the extreme right side, click on the menu icon to show the menu, select Debug.



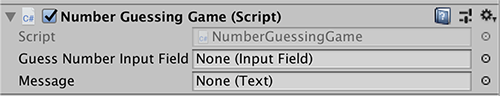
* + - 1. Scroll down to locate “Content Type” and select “Integer Number”. (This will only allow user to enter integer numbers into this input field).
      2. Click on Play in the Toolbar to test out.
      3. You should get something like the below.

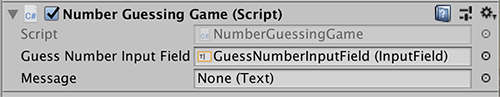


* + 1. Create the game scripts
       1. Go to Project window; create a new folder “Scripts” under “Assets” folder.
       2. In the Scripts folder, create a new C# Script.
       3. At the point when the C# Script is created, rename it to “NumberGuessingGame” immediately.
       4. Double click on this C# Script to open up Visual Studio.
       5. Add in the UnityEngine.UI namespace and the three fields as shown below.



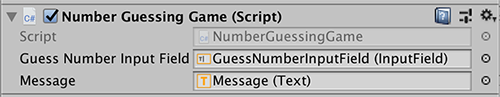
* + 1. Link the NumberGuessingGame C# Script to the game
       1. Go to Hierarchy window, right click on NumberGuessingGame, Select GameObject-> Create Empty.
       2. Rename the game object to “NumberGuessingGame”.
       3. Go to Project window, click on NumberGuessingGame C# Script and drag it to Hierarchy window and drop it on NumberGuessingGame game object.
       4. In the Hierarchy window, click on the NumberGuessingGame game object and you should get the following in the Inspector window under the section Number Guessing Game (Script).



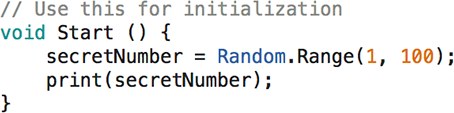
* + 1. Link the GuessNumberInputField to the NumberGuessingGame C# Script GuessNumberInputField
       1. On the Hierarchy window, select NumberGuessingGame game object. Make sure the Inspector window shows the properties of NumberGuessingGame game object.
       2. Click on GuessNumberInputField(without unselecting NumberGuessingGame game object and making sure that the Inspector window is still showing the property of NumberGuessingGame game object) and drag the GuessNumberInputField and drop it on the Inspector window Script input field. You should get something like this.

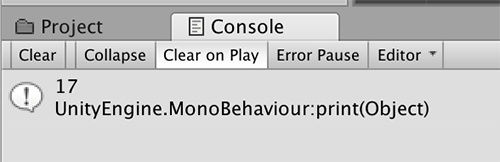
The C# Script Guess Number Input Field is now linked with GuessNumberInputField game object on the canvas.

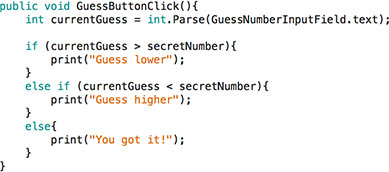
* + 1. Link the Message text to the NumberGuessingGame C# Script Message
       1. Perform similar steps as the previous and link the Message text to the NumberGuessingGame C# Script Message.
       2. You should get something like this.



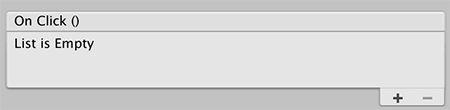
* + 1. Generate a secret number
       1. In the Start method of NumberGuessingGame C# Script, add in the following script.



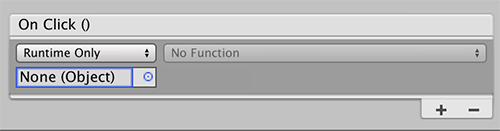
* + - 1. In the Scene View, Game panel, make sure “Maximize on Play” is disabled, click on Play in the Toolbar to test out.
      2. In the Project window, Console panel, you should get a random integer printed.
    1. Link the GuessButton to NumberGuessingGame C# Script
       1. Add in a new method to the NumberGuessingGame C# Script.



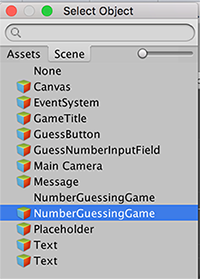
* + - 1. In the Hierarchy window, click on GuessButton, the Inspector window will shows the properties of GuessButton.
      2. In the Inspector window, scroll down and look for On Click ()



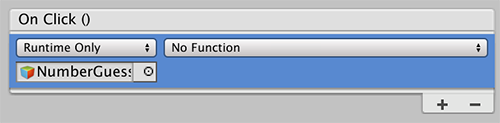
* + - 1. Click on + to add a listener.



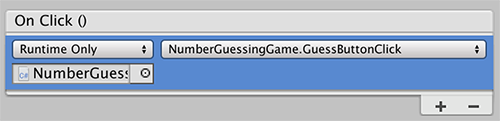
* + - 1. Click on blue circle besides None (Object).
      2. In the Select object window, select the Scene panel.
      3. Select NumberGuessingGame object.



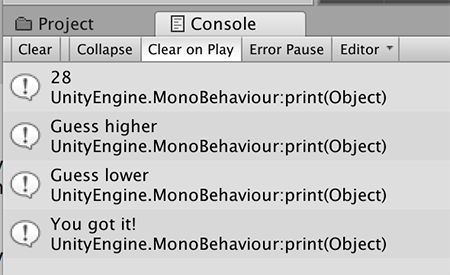
* + - 1. You should get something like this.



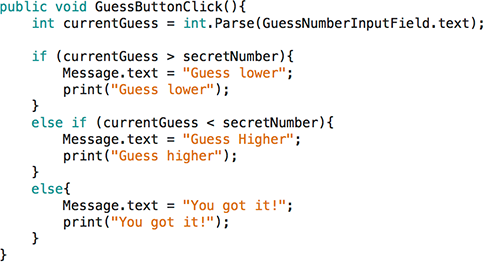
* + - 1. Click on No Function->Number Guessing Game->GuessButtonClick().
      2. You should get something like this.

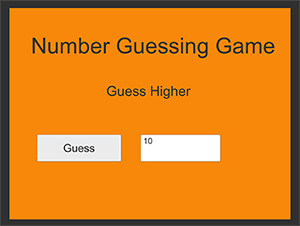


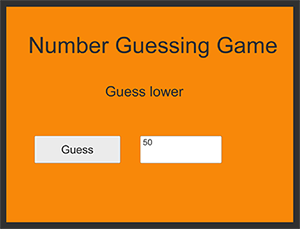
* + 1. Test run the game
       1. Click on Play in the Toolbar to test out.
       2. Look for the printed messages on the Console.
       3. You should get something like this.



* + 1. Update the message in the Message game object
       1. Change the GuessButtonClick() method.









* + 1. Create an Introduction Screen
       1. In the Project window, right click on Scene folder (or in the Scene folder), select Create->Scene.
       2. Rename the new Scene to “Introduction”.
       3. Design an Introduction scene for the game.
       4. Look for royalty some free images.
       5. Create an images subfolder in Project window under Assets folder.
       6. Drag the image to that folder.
       7. In the Hierarchy window, right click and select GameObject->UI-

>Image.

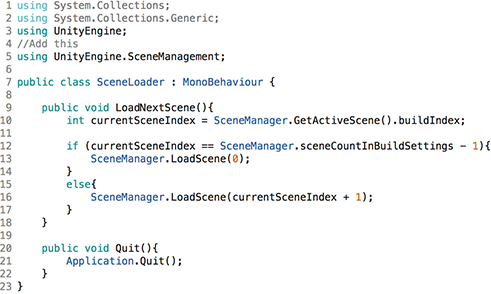
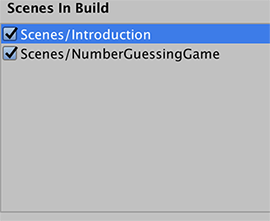
* + - 1. In the Image Property Inspector, select the Source Image.
      2. Adjust the image object.
      3. You should have something like this.



(Source: [https://pixabay.com/en/wizard-hat-sorcerer-hat-](https://pixabay.com/en/wizard-hat-sorcerer-hat-magic-3361668/) [magic-3361668/](https://pixabay.com/en/wizard-hat-sorcerer-hat-magic-3361668/))

* + 1. Create a start button
       1. Insert a new button object.
       2. Rename it to StartButton.
       3. Adjust the button to the right size.
       4. Change the text of the button to “Start”.
       5. Click on Play in the Toolbar to test out.
       6. You should have something like this.



* + 1. Create a SceneLoader C# Script
       1. In Project window, Scripts folder, create a SceneLoader C# Script.
       2. Delete the default Start and Update method.
       3. Add in the UnityEngine.SceneManagement namespace and LoadNextScene and Quit method as below:
    2. Arrange the scene sequence
       1. Go to Unity main menu, select File->Build Setting.
       2. In Build Setting, drag Introduction scene from the Project window to Scenes in Build.
       3. Arrange the Introduction scene before the NumberGuessingGame scene.
    3. Create a LoadScene object in Introduction scene
       1. Create an empty Game Object.
       2. Rename the empty Game Object to “LoadScene”.
       3. Link the SceneLoader script to LoadScene object.
       4. In the Property Inspector of LoadScene object, you should see the link as shown below.
    4. Test the program
       1. Run the game from the Introduction scene.
       2. Click on Start.
       3. The scene should change to NumberGuessingGame scene.
    5. Attempt the following tasks on your own

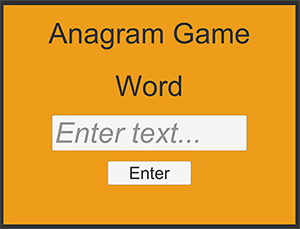
1. Create an Ending scene.
2. When the gamer is done with the game, show the Ending scene.
3. The gamer has the option to replay or exit the game.
4. Limit the number of guesses to 5 times. The gamer wins if the right guess was made within 5 tries.

## Activity 2: Anagram Game (attempt this on your own)

In this game, an English word is randomly generated. The gamer forms another word using letters from the word. For example, the letters in the word “care” can be rearrange to form the word “race”. You can get a list of common anagrams from: [http://](http://itools.subhashbose.com/wordfind/common-anagrams/) [itools.subhashbose.com/wordfind/common-anagrams/](http://itools.subhashbose.com/wordfind/common-anagrams/)

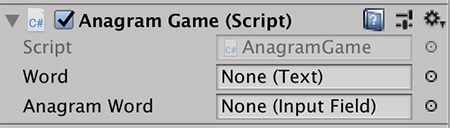
#### Procedure:

* + 1. Start Unity Hub
    2. New a project and name it Anagram Game Under Template, select 2D.
    3. Under Project window, go to the assets->scene
    4. Rename the Sample Scene to AnagramGame
    5. Change the game screen solution a. Go to Scene View-> Game panel. b. Change the game screen solution to “Standalone (1024x768)”.
    6. Add in the game title, word (to display the random word), input field and enter button in the game scene canvas You should get something like this:

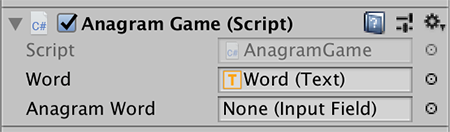
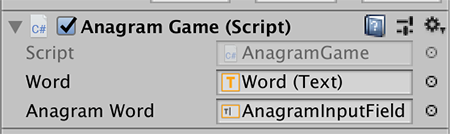


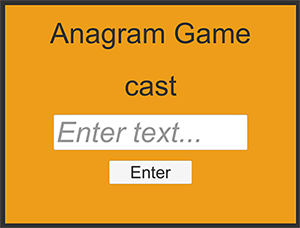
* + 1. Create the game scripts
       1. Go to Project window, create a new folder “Scripts” under “Assets” folder.
       2. In the Scripts folder, create a new C# Script.
       3. At the point when the C# Script is created, rename it to “AnagramGame” immediately.
       4. Double click on this C# Script to open up Visual Studio.
       5. Add in the UnityEngine.UI namespace and the C# codes as below:

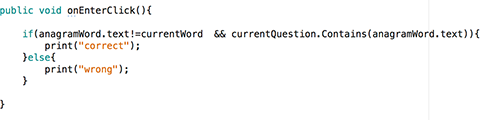
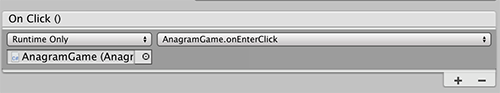


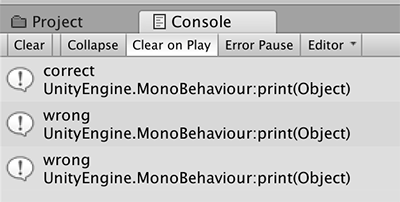
* + 1. Link the AnagramGame C# Script to the game
       1. Go to Hierarchy window, right click on AnagramGame, Select GameObject-> Create Empty.
       2. Rename the game object to “AnagramGame”.
       3. Go to Project window, click on AnagramGame C# Script and drag it to Hierarchy window and drop it on AnagramGame game object.
       4. In the Hierarchy window, click on the AnagramGame game object and you should get the following in the Inspector window under the section Anagram Game (Script).
    2. Link the Word text and Anagram Word input field to the AnangramGame C#

Script

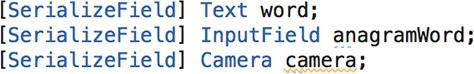
* + - 1. On the Hierarchy window, select AnagramGame game object. Make sure the Inspector window shows the properties of AnagramGame game object.
      2. Click on Word(without unselecting AnagramGame game object and making sure that the Inspector window is still showing the property of AnagramGame game object), drag the Word and drop it on the Inspector window Script text field. You should get something like this.
      3. Repeat that for Anagram Word input field. You should get something like this.
    1. Test runs the game. You should get a random word displayed every time you run the game. You should get something like this.



* + 1. Check if the anagram entered by the gamer is correct Add this method to AnagramGame C# script
    2. Link the Enter button to the method onEnterClick
       1. In the Inspector window of EnterButton properties, you should get something like this.
    3. Test out the game

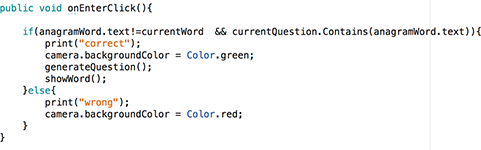


* + 1. Change the camera background colour for correct and wrong answer
       1. Open the AnagramGame C# script
          1. Add in a serialize field camera

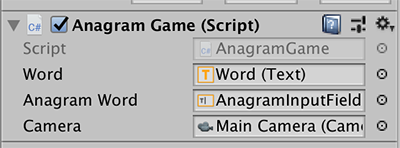


**Figure 1.45** Activity 2-10

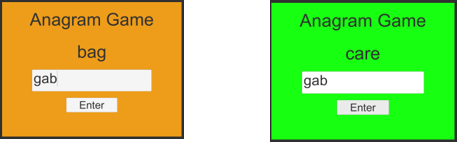
* + - * 1. Update the onEnterClick method



* + 1. Link the camera to AnagramGame game object
       1. You should get something like that.



* + 1. Test the game
       1. When a right anagram is entered, the background will turn green and a new word will be displayed.



* + - 1. When a wrong anagram is entered, the background will turn red.



* + 1. Attempt the following tasks on your own
       1. Create an Introduction with instruction and Ending scene.
       2. When the gamer is done with the game, show the Ending scene.
       3. The gamer has the option to replay or exit the game.
       4. Gamer loses if a wrong anagram is entered.
       5. Word will not repeat.
       6. Gamer wins when all correct anagrams are entered for all the words.
       7. Different game level – 3 letters to 4 letters to 5 letters.