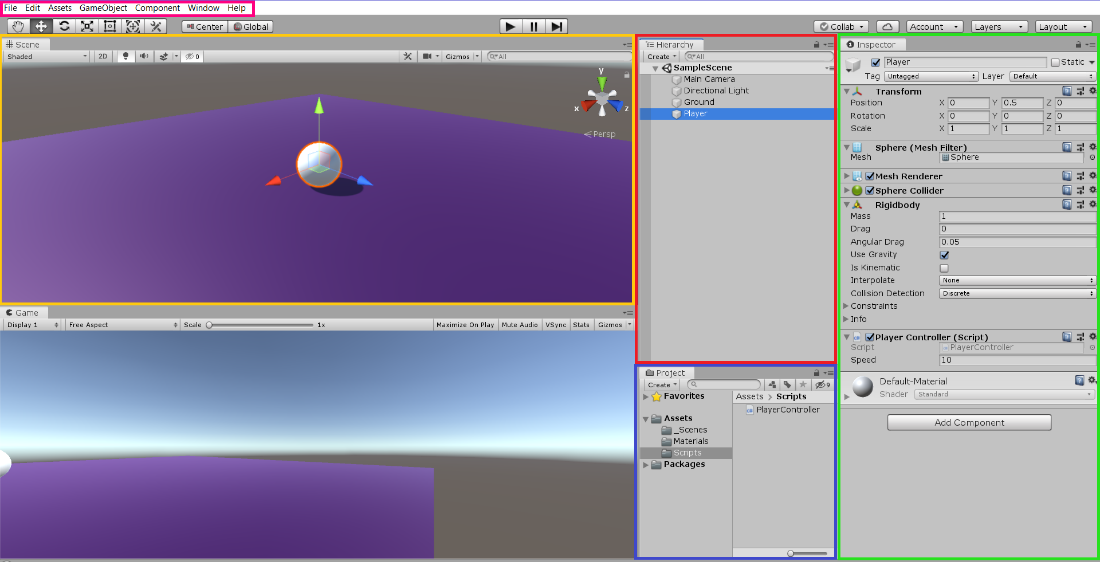
**Unity Roll-a-Ball Program**

Estimated time to completion: 2 hours

**Unity interface key:**



**Hierarchy view**

**Inspector view**

**Project view**

**Scene view**

**Menu toolbar**

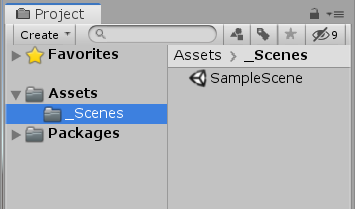
**Scripts located @ https://github.com/lcanner/RTFN**

(Leave this page open in an internet browser—you’ll refer to it frequently)

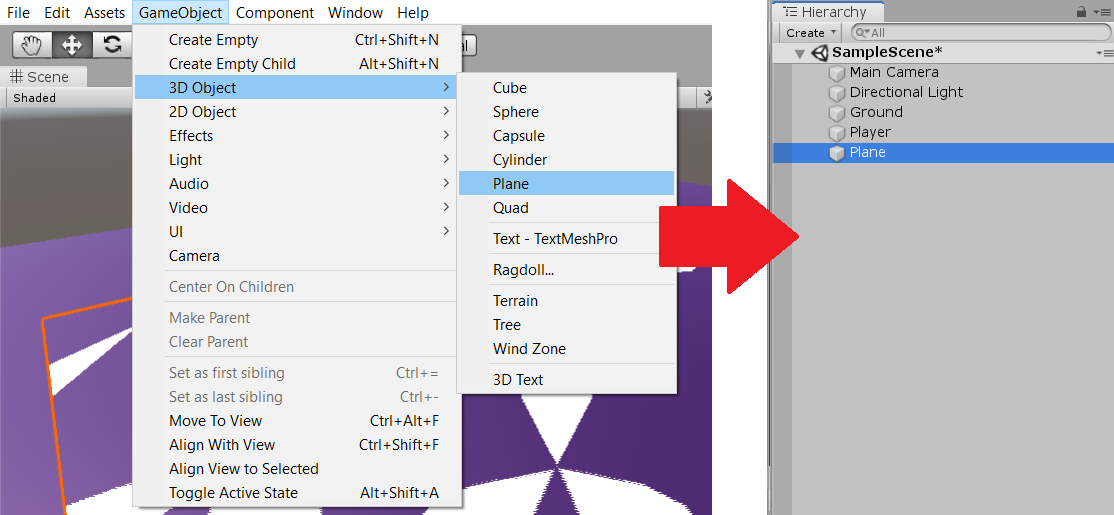
**Setting Up the Game**

**Creating a new Project and its Assets**

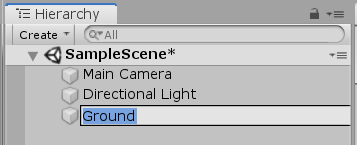
1. Create a new 3D project upon launching Unity, or from the Menu toolbar (File > New Project).
2. In the Project view, open the Assets folder and right-click on the Scenes folder. Add an underscore to the front of the folder so it reads “\_Scenes”. This will ensure that it stays at the top of the Assets folder as we add to it.



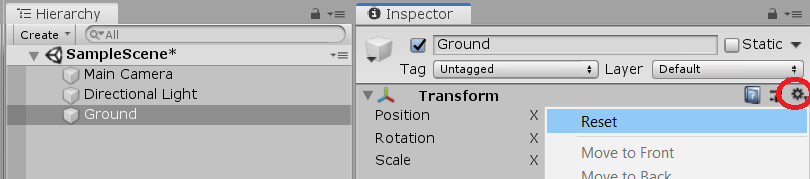
1. Use the Menu toolbar to create a 3D plane (GameObject > 3D object > Plane). Notice than a new Plane object is added to the Hierarchy on the right.



* 1. Right-click on the newly created Plane object in the Hierarchy and rename it “Ground”.



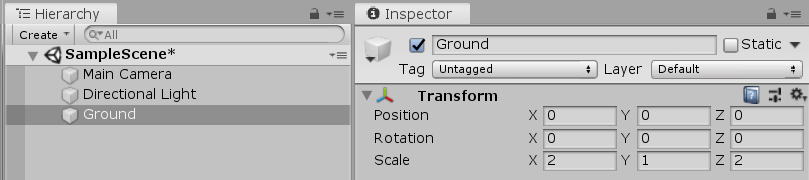
* 1. Reset the Ground object’s “transform stats” to origin by clicking the gear to the right of the Transform component in the Inspector and choosing “reset”. This will set the X, Y, and Z positions of the object to 0, moving it to the very center of the game world.



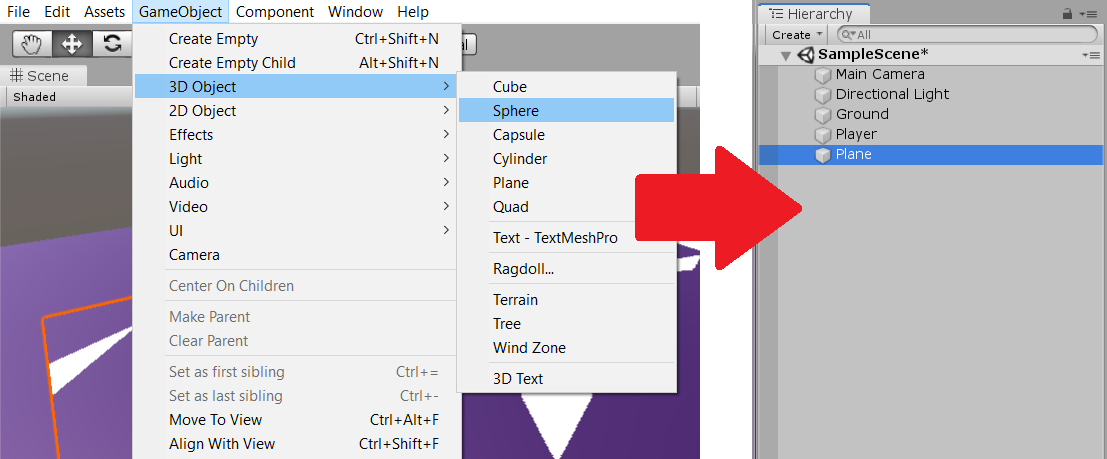
1. In the Scene view, click the “Gizmos” Menu at the top and uncheck “Show Grid” to hide measurement lines on objects—we won’t need them for this project.



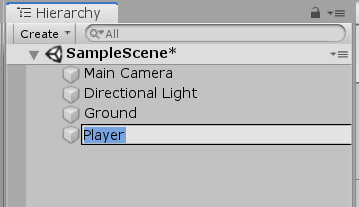
1. In the Inspector view, adjust the Transform scale of the Ground object: (X=2, Y=1, Z=2). This should double the size of the object; we leave the Y axis at 1 because there is no depth to a plane.



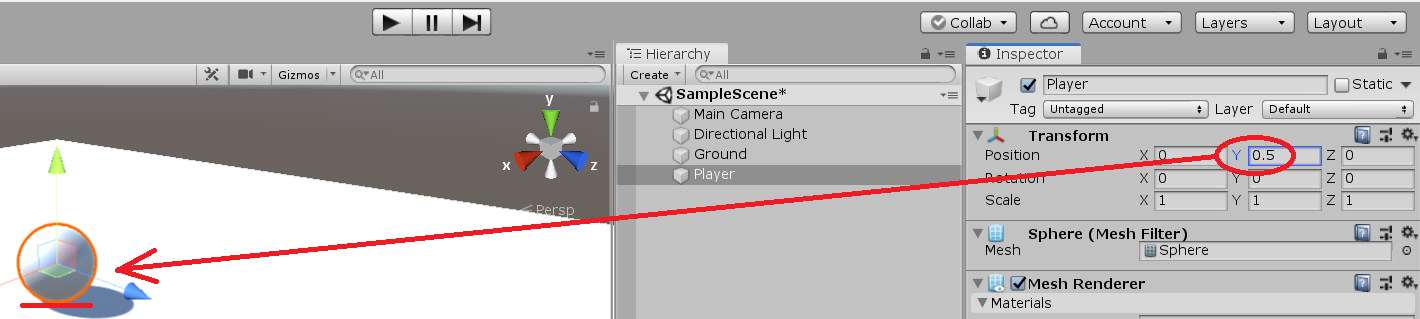
1. Create a new 3D sphere object using the Menu toolbar (Gameobject > 3D object > Sphere). Note that a new object called “Sphere” is added to the Hierarchy.



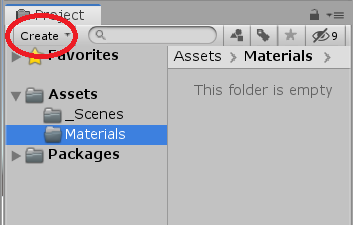
* 1. Right-click on the newly created Sphere object in the Hierarchy and rename it “Player”.



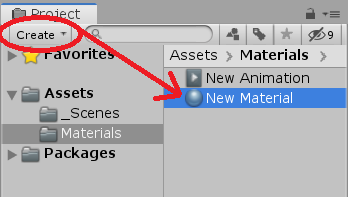
* 1. When the object is created, it should be at “origin”, resting at the center of the game world. In this case, the bottom half of the Player object is stuck beneath the plane. Select the Player object in the Hierarchy and adjust the Transform position in the Inspector to (Y=.5)



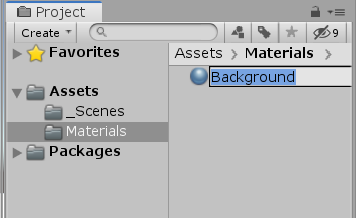
1. In the Project view, create a new folder and name it “materials”. This folder will hold skins and textures that can be applied to the objects we create.



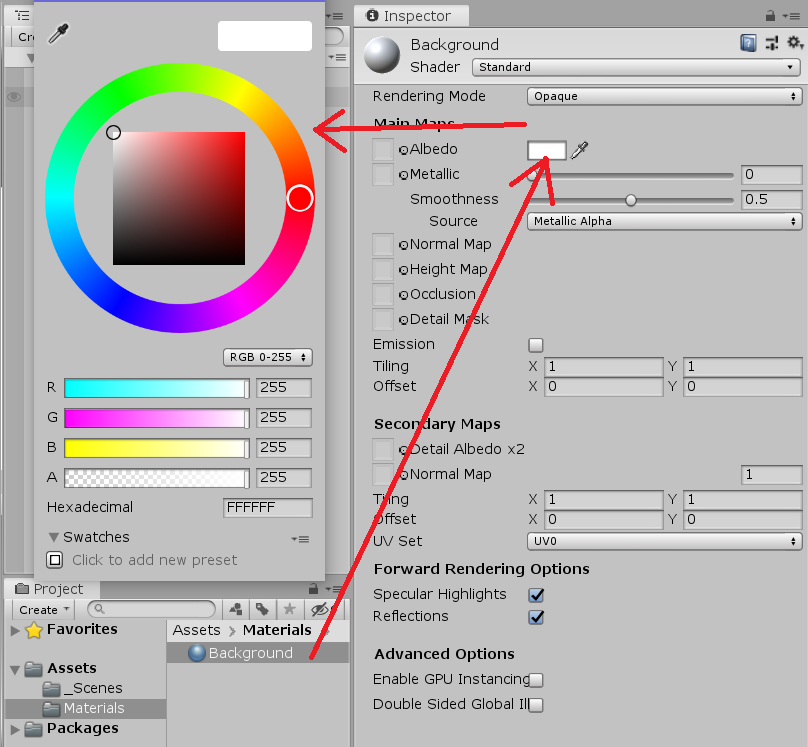
* 1. With the materials folder selected, click the Create Menu again and choose “material” to make a new material in the folder.



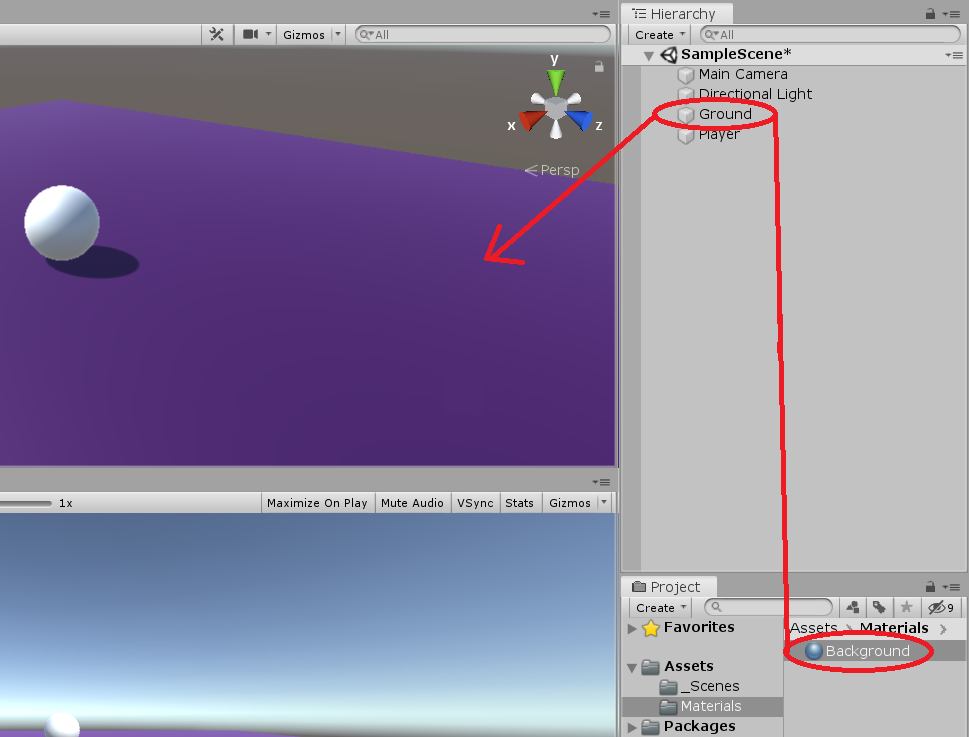
* 1. Right-click on the new material and rename it “Background”.



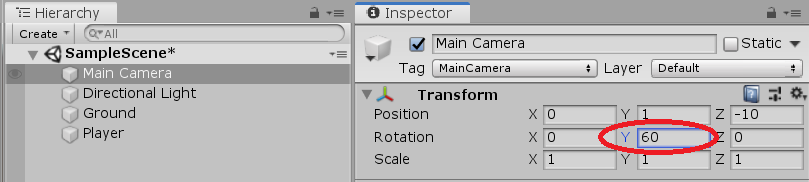
* 1. Select the Background material in the Project Menu. Look at the Inspector to the right and locate the “Albedo” window. This is the tool used to change the color of materials in Unity. Click the color window and choose a color to be used for the Ground plane.



* 1. Click+drag the Background material from the Project view to the Ground object in the Hierarchy. The Ground object should change colors to match the material applied.

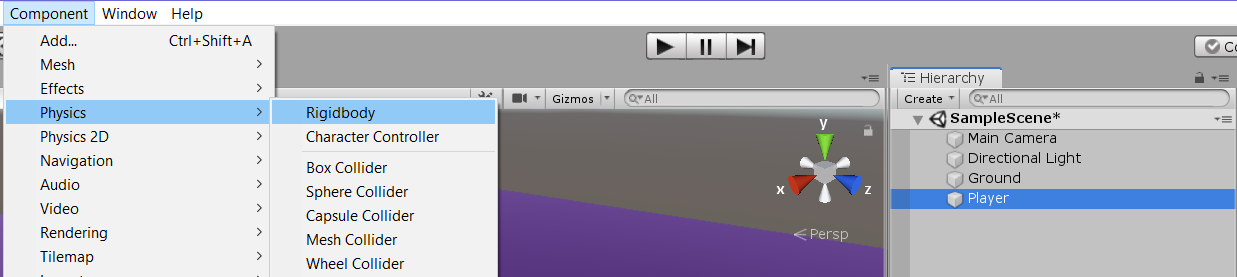


1. Select the Main Camera in the Hierarchy. In the Inspector, change the Transform rotation to (Y=60). This changes the direction of the light source to a more practical position for our game.

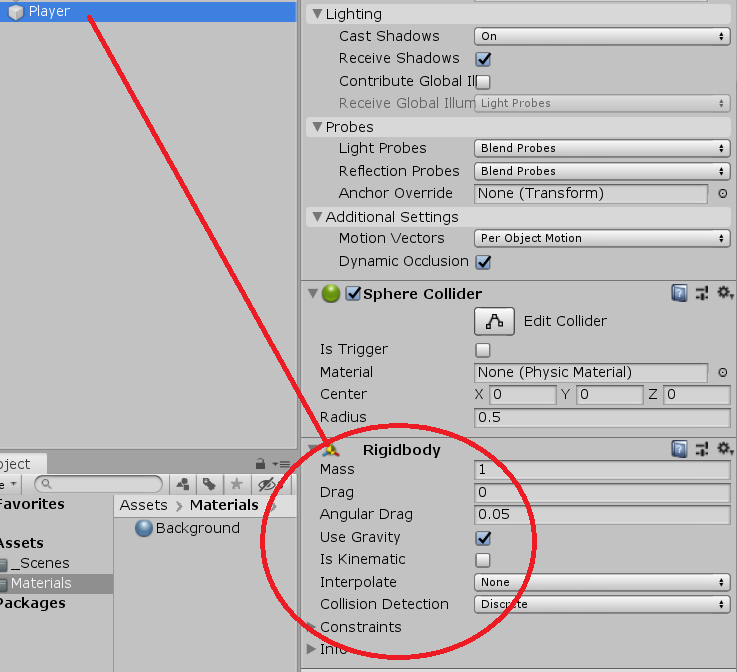


**Moving the Player**

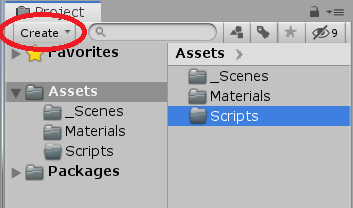
1. In order for our Player object to eventually be able to collect the Pick Up objects we’ll be creating, Unity needs to see that our Player object is solid and capable of colliding with other objects. To indicate this, select the Player object in the Hierarchy and use the Menu toolbar to create a Rigidbody (Component > Physics > Rigidbody).



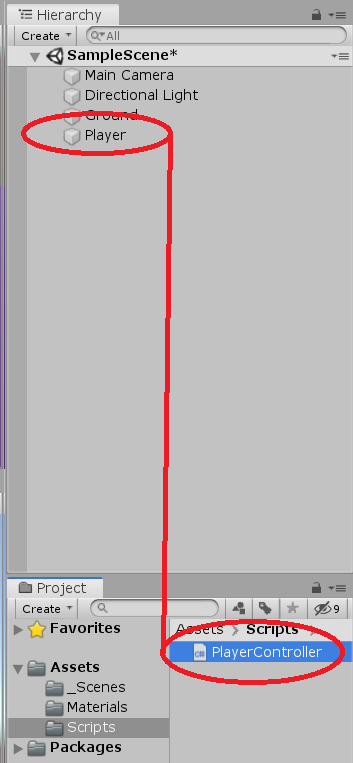
* 1. When the Rigidbody is created it will automatically be attached to the selected object—in this case, the Player object. Scroll through the Inspector and locate the Rigidbody component to ensure the physics component is successfully attached to the Player object.



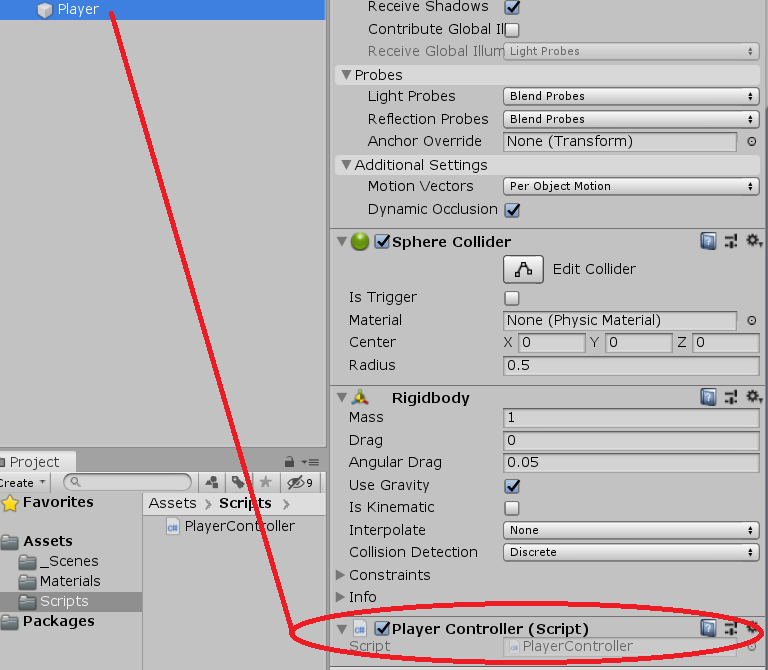
1. Select the Assets folder in the Project view. Use the Create button to make a new folder and name it “Scripts”. This is where we’ll be keeping the code we’ll need to make the game work.



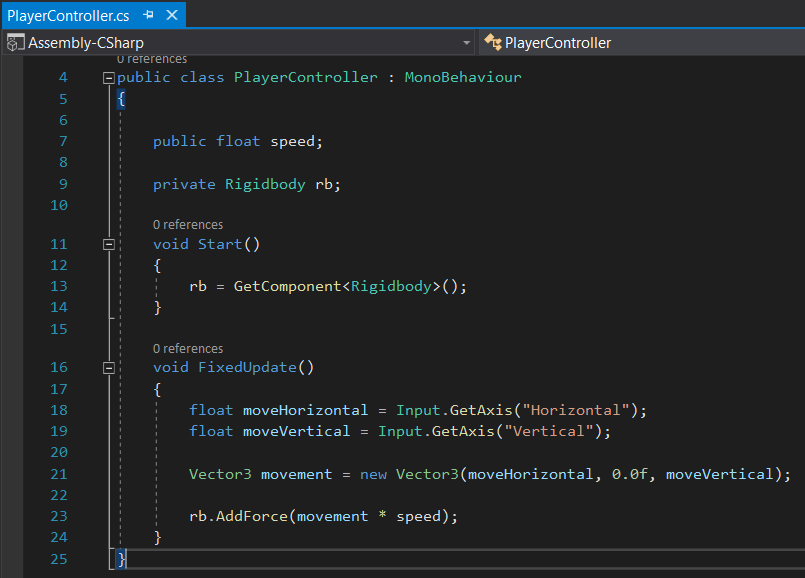
1. With the newly created Scripts folder selected, click the Create button again and choose “C# Script”. Name your script “PlayerController” (pay attention to the capital letters!)
   1. We’ll need to assign the PlayerController Script to the Player object. Do this by click+dragging the Script from the Project view to the Player object in the Hierarchy.



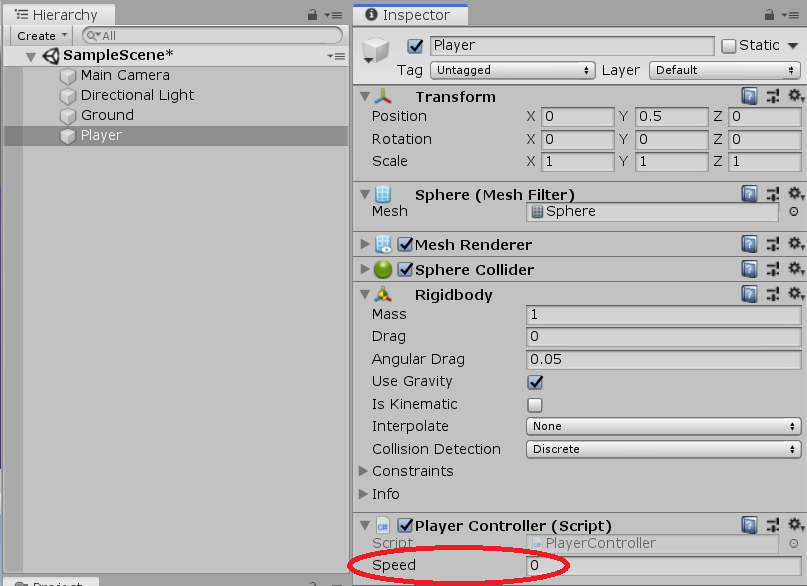
* 1. Confirm that the PlayerController script successfully attached to the Player object by selecting the Player object and locating the PlayerController component in the Inspector.



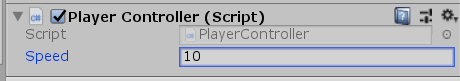
1. Double-click on the PlayerController script in the Project view to open it. Delete the placeholder text and replace it with the PlayerController code found @ https://github.com/lcanner/RTFN (PlayerController - Moving the Player)



1. With the PlayerController script updated, a new field called “Speed” will be created in the PlayerController component of the Player object in the Inspector.



* 1. Change the Speed from 0 to 10. This will enable the Player object to move when it receives input commands—in our case, these commands will eventually come from the arrow keys on our keyboard.



1. Press the Play button above the Scene and Hierarchy views to run a test version of the game we’ve created so far. You should be able to move the Player object by pressing the arrow keys!



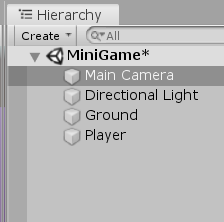
* 1. If not, verify that you updated the Speed field to 10 and double-check the code in the PlayerController script.

\*\*\*Note: Press the Play button again to exit the test run and get back to editor mode. If you make any changes in Unity while a test version of the game is running, the changes will not be saved.

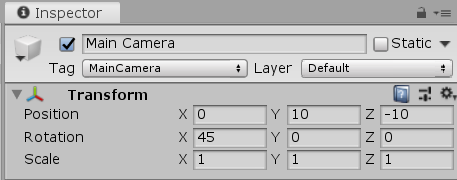
**Camera and Play Area**

**Moving the Camera**

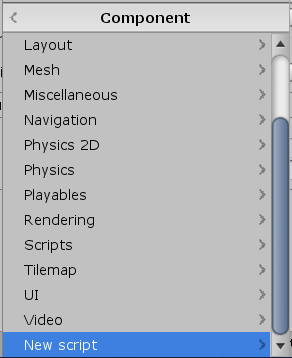
1. Select the Main Camera in the Hierarchy.



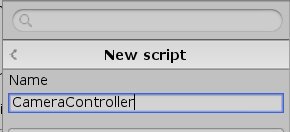
* 1. Set the Transform position of the Main Camera: (Y=10).
  2. Set the Transform rotation of the Main Camera: (X=-45).



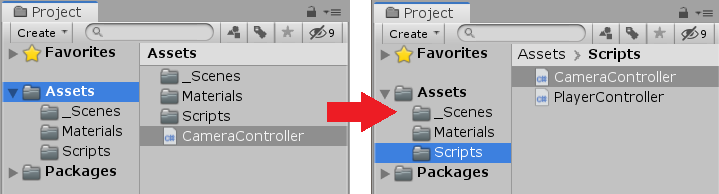
* 1. At the bottom of the Inspector, click “Add Component” and choose “new script” at the bottom of the list.



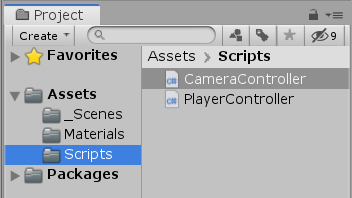
* 1. Name the new script “CameraController”.



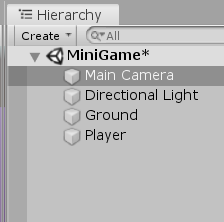
1. Look in the Project view under the Assets folder and locate the newly created CameraController script. Click+drag it to the Scripts folder.



1. Double-click on the CameraController script to open it for editing. Copy+paste the code @ <https://github.com/lcanner/RTFN> (CameraController - Moving the Camera).



1. Select the Main Camera in the Hierarchy.



* 1. Click+drag the Player object in the Hierarchy to the “Player” field of the Camera Controller component in the Inspector.



* 1. If done correctly, the field will now display 

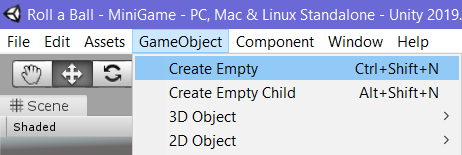
1. Test your game by clicking the Play arrow button above the Scene view.



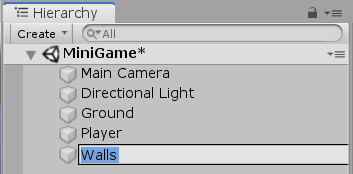
* 1. The camera should follow the ball as it moves, but it should not rotate as the ball spins. If this isn’t the case, end the test and double-check the Camera Controller script and verify the Player object is attached to the Camera Controller via the Player Field of the Camera Controller script component.

**Setting Up the Play Area**

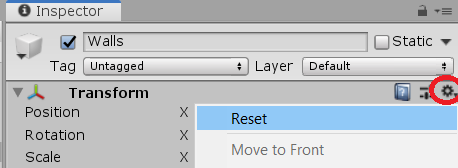
1. Create an Empty object using the Menu toolbar (Gameobject > Create Empty). An Empty object is used for grouping and organizing objects in the Hierarchy.



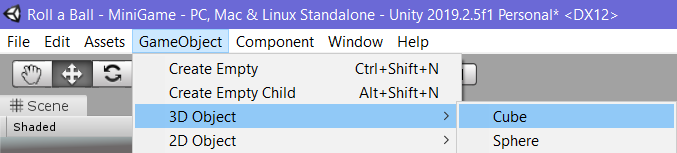
* 1. In the Heirarchy, right-click on the newly created Empty object and name it “Walls”.



* 1. Reset the Walls object’s transform stats to origin by clicking the gear to the right of the transform Menu in the Inspector and choosing “reset”.



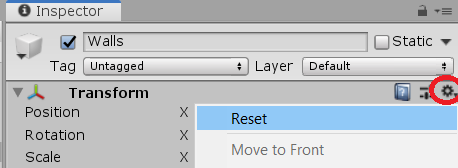
1. Create a 3D Cube object using the Menu toolbar (Gameobject > 3D object > Cube).



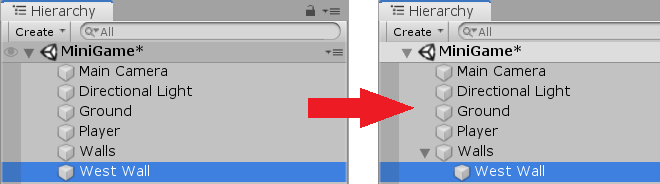
* 1. Name the Cube object “West Wall” in the Hierarchy.



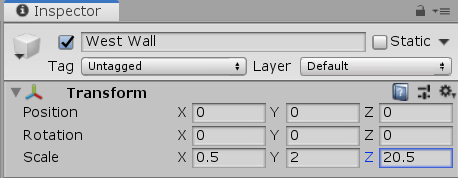
* 1. Reset the West Wall object to origin using the gear gizmo.



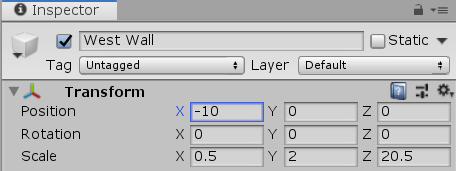
* 1. Click+drag the West Wall object to the Walls object in the Hierarchy to attach it. Walls in now the “Parent” of West Wall, and West Wall is now the “Child” of Walls.



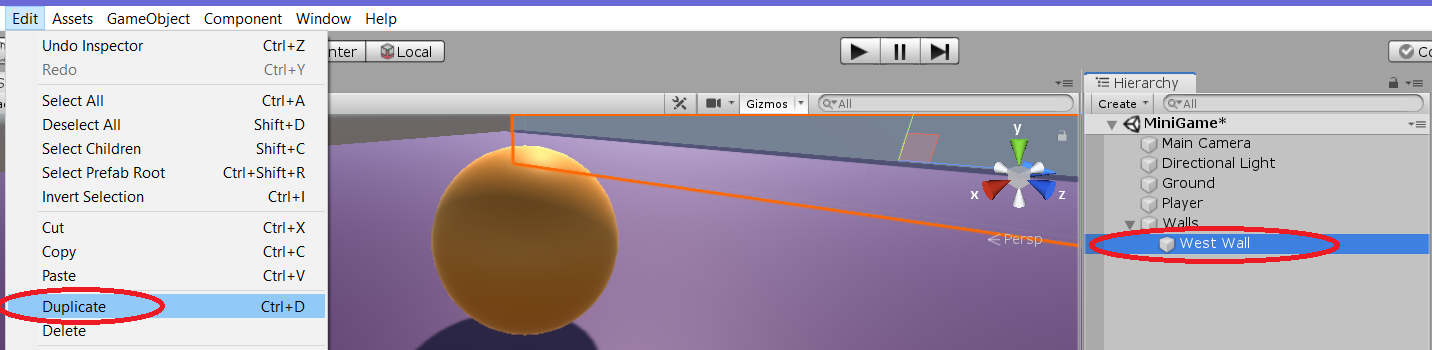
* 1. Change the Transform scale of the West Wall: (X=0.5, Y=2, Z=20.5)



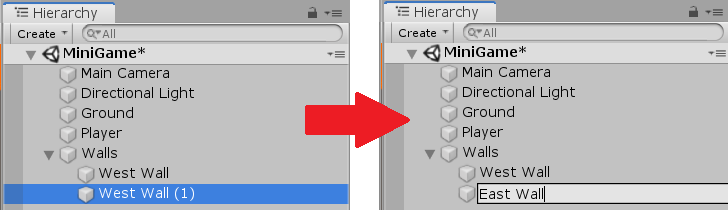
* 1. Change the Transform position of the West Wall: (x=-10).



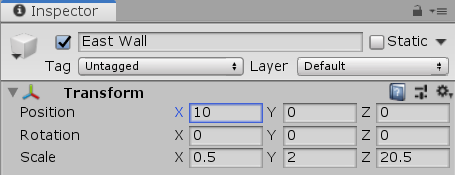
1. Select the West Wall in the Hierarchy. Use the Menu toolbar to duplicate the West Wall (Edit > Duplicate).



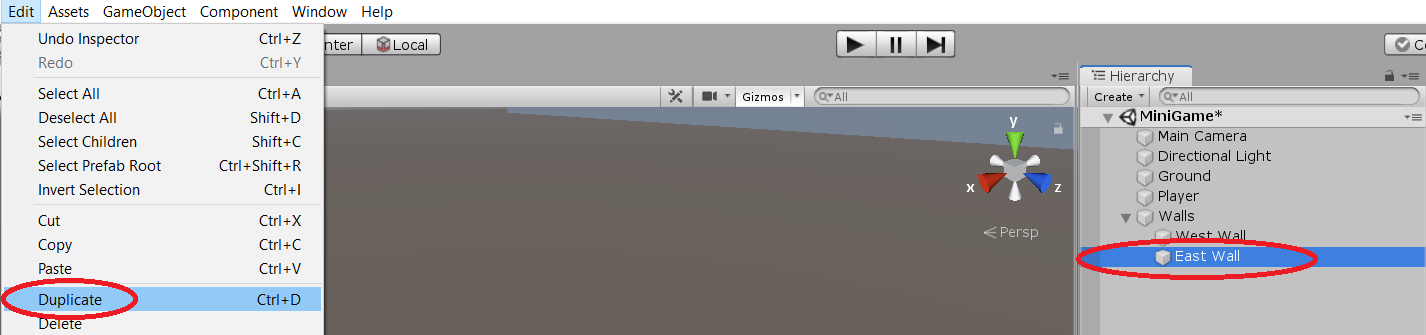
* 1. Right-click the duplicate object in the Hierarchy and rename it “East Wall”.



* 1. Change the Transform position of the East Wall: (X=10).



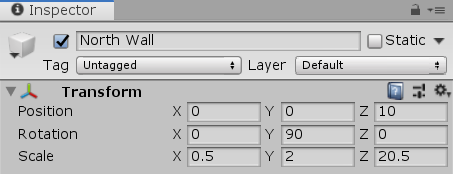
1. Duplicate the East Wall using the Menu toolbar (Edit > Duplicate).



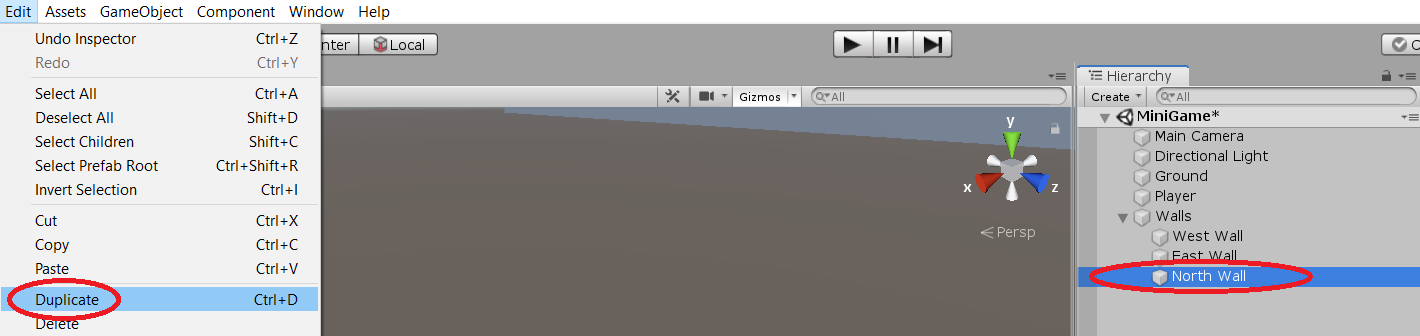
* 1. Rename the duplicate object “North Wall” in the Hierarchy.



* 1. Change the Transform rotation of the North Wall: (Y=90).
  2. Change the Transform position of the North Wall: (X=0 , Z=10).



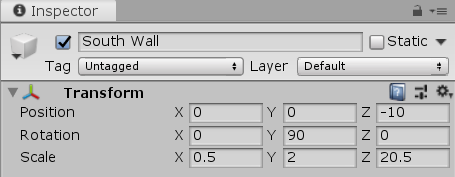
1. Duplicate the North Wall using the Menu toolbar (Edit > Duplicate).



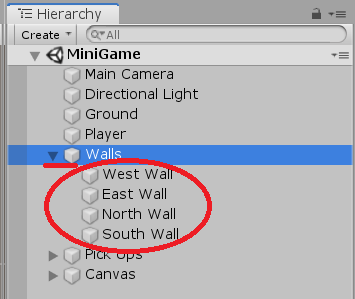
* 1. Rename the duplicate “South Wall” in the Hierarchy.



* 1. Change the Transform position of the South Wall: (Z=-10).



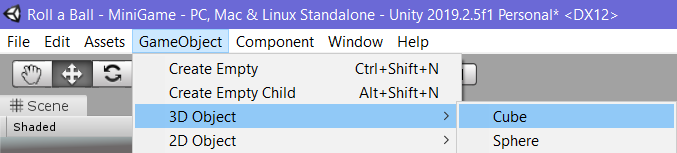
* 1. Double-check the Hierarchy to ensure all four Wall objects are grouped with the Walls empty object. If any of the Walls aren’t grouped, click+drag them to the Walls empty object to add them.



**Collecting, Scoring and Building the Game**

**Creating Collectable objects**

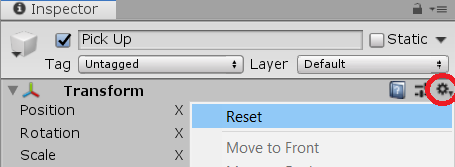
1. Create a 3D Cube object using the Menu toolbar (Gameobject > 3D object > Cube).



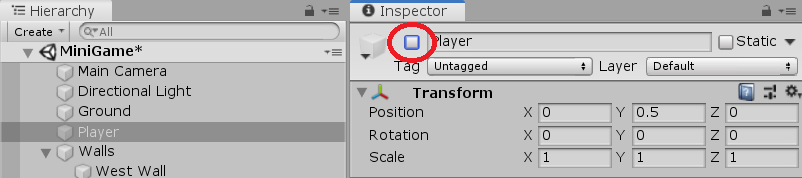
* 1. Name the newly created object “Pick Up” in the Hierarchy. If it was accidentally attached to the Walls empty object, click+drag it out to remove it from the group.



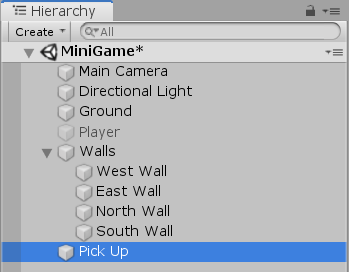
1. Reset the Pick Up object’s Transform stats to origin by clicking the gear to the right of the Transform component in the Inspector and choosing “Reset”.



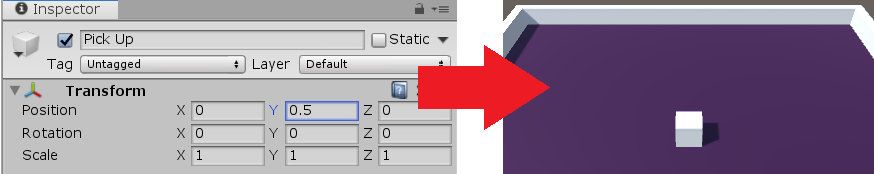
1. Select the Player object in the Hierarchy, and uncheck the “active” box next to the colored cube at the top of the Inspector. This will make the Player object disappear and give users a better view of the newly created Pick Up object.



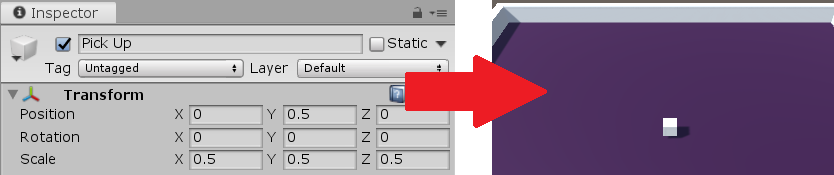
1. Select the Pick Up object in the Hierarchy.



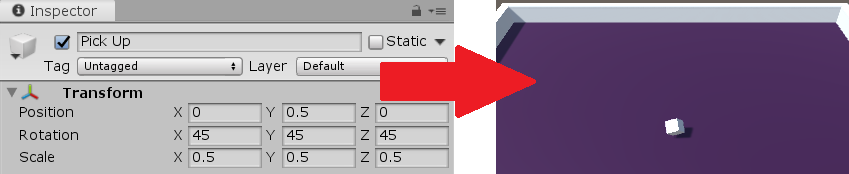
* 1. Align the Pick Up object flat on the Ground plane by changing the transform position: (Y=.5).



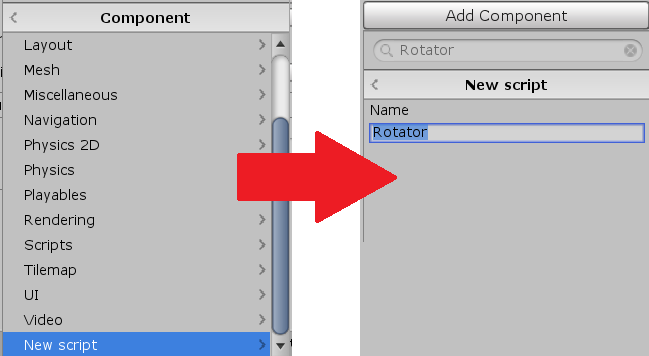
* 1. Change the Transform scale of the Pick Up object: (X=.5 , Y=.5 , Z=.5). This will shrink the Pick Up object so it appears to be floating above the plane surface.



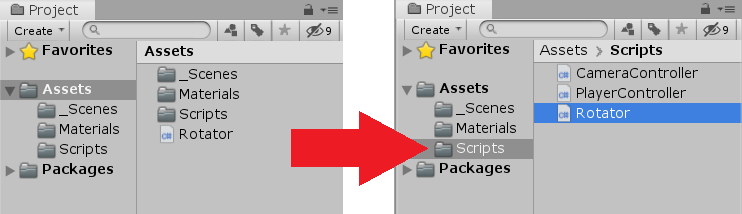
* 1. Tilt the Pick Up object on an angle by changing the Transform rotation: (X=45 , Y=45 , Z=45).



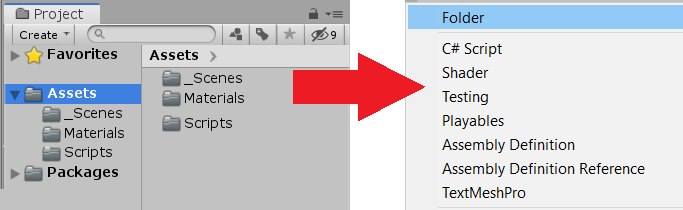
1. Verify that the Pick Up object is selected in the Hierarchy. Scroll to the bottom of the Inspector and click “Add Component.” Choose “New Script” and name the script “Rotator”.



1. In the Project view, locate the newly created Rotator script in the Assets folder and click+drag it to the Scripts folder. <Copy+paste> the script code found @ <https://github.com/lcanner/RTFN> (Rotator - Creating Collectible Objects)



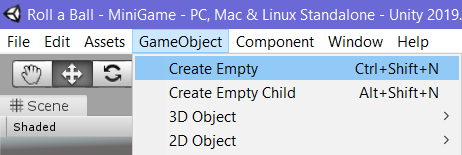
1. In the Project view, select the Assets folder and click “create” at the top. Choose “folder” and name the folder “Prefabs”.



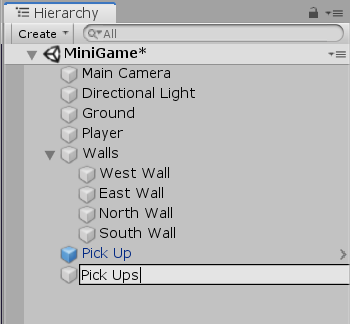
1. Click+drag the Pick Up object from the Hierarchy into the newly created Prefabs folder in the Project view.



1. Create a new Empty object using the Menu toolbar (Gameobject > Create Empty).



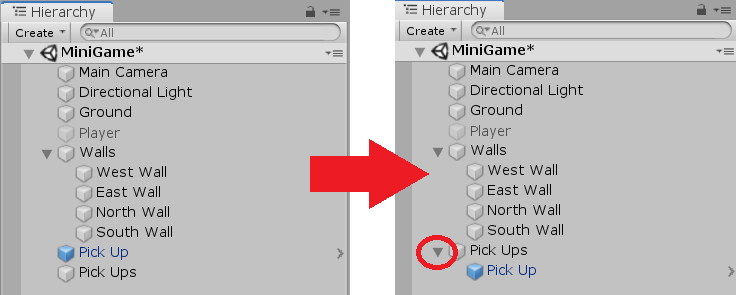
* 1. In the Hierarchy, name the newly created Empty object “Pick Ups”.



* 1. Reset the Pick Ups Transform stats to origin by clicking the gear to the right of the Transform Menu in the Inspector and clicking “reset”.

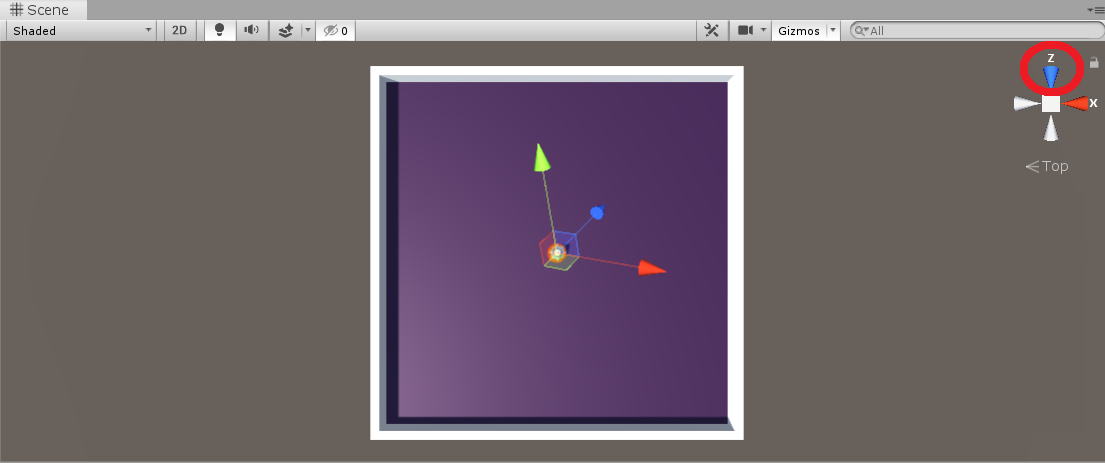


1. In the Hierarchy, click+drag the previously created Pick Up object into the newly created Pick Ups Empty object, like we did previously to group the Wall objects together.

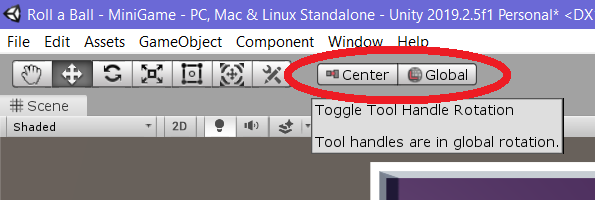


* 1. In the Hierarchy, click on the newly moved Pick Up object to select it.

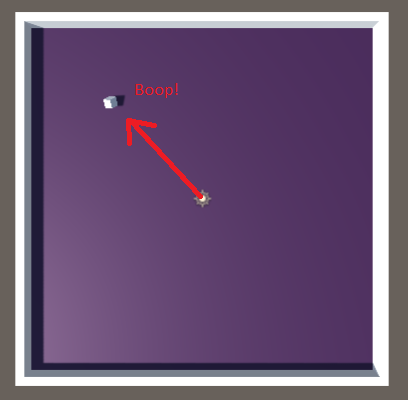
1. In the Scene view, click the Z button on the compass gizmo in the upper right corner. This will rotate the view to a top-down perspective. Zoom out until the entire Ground object can be seen.



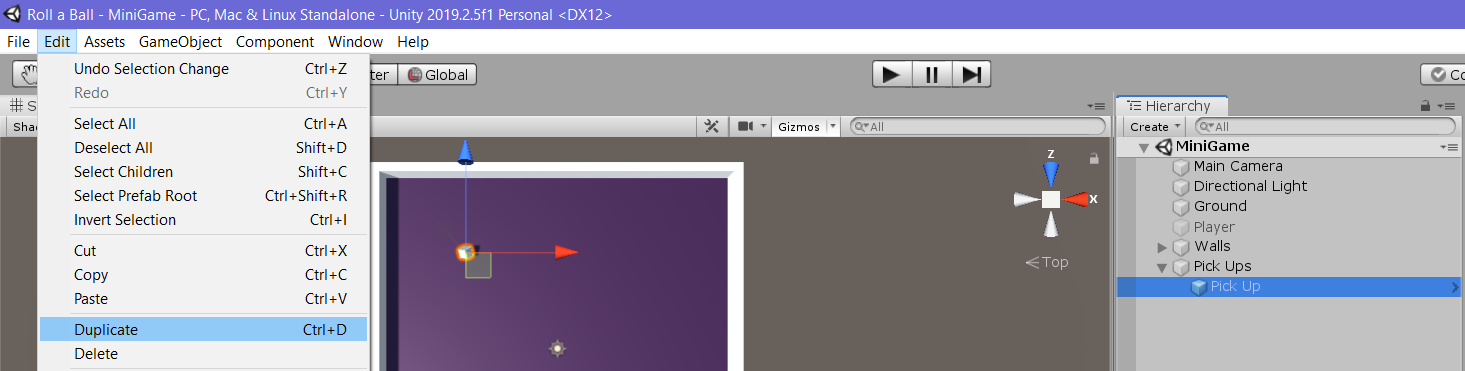
* 1. Change the Scene editor view from “local” to “global” using the tab at the top of the Scene view window.



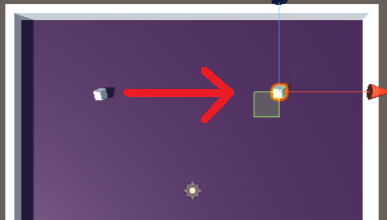
1. In the Scene view, click+drag the Pick Up cube to any desired location.



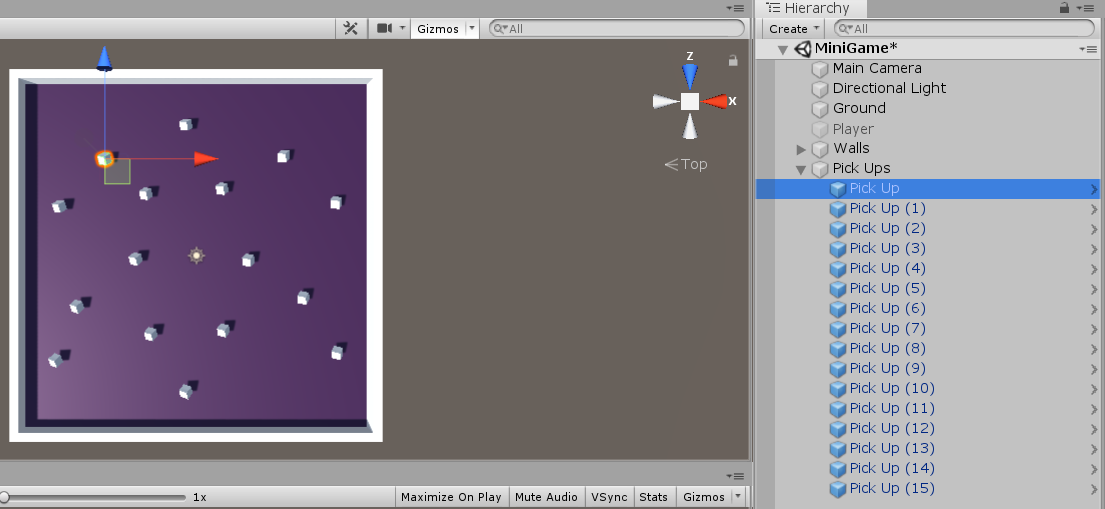
1. Create a duplicate of the Pick Up object using the Menu toolbar (Edit > Duplicate). The duplicate object will appear directly on top the original Pick Up object.



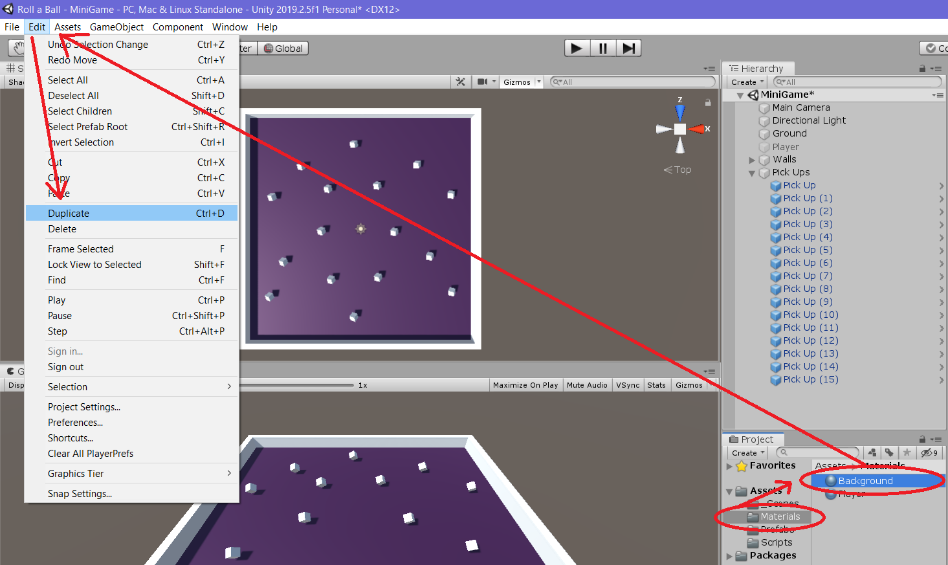
* 1. Click+drag the duplicate Pick Up object to another desired location.



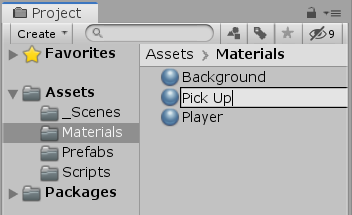
* 1. Note that the duplicate Pick Up object called “Pick Up (1)” has been added to the Pick Ups folder in the Hierarchy. Repeat the duplication and movement processes until you have as many Pick Up objects as you’d like.



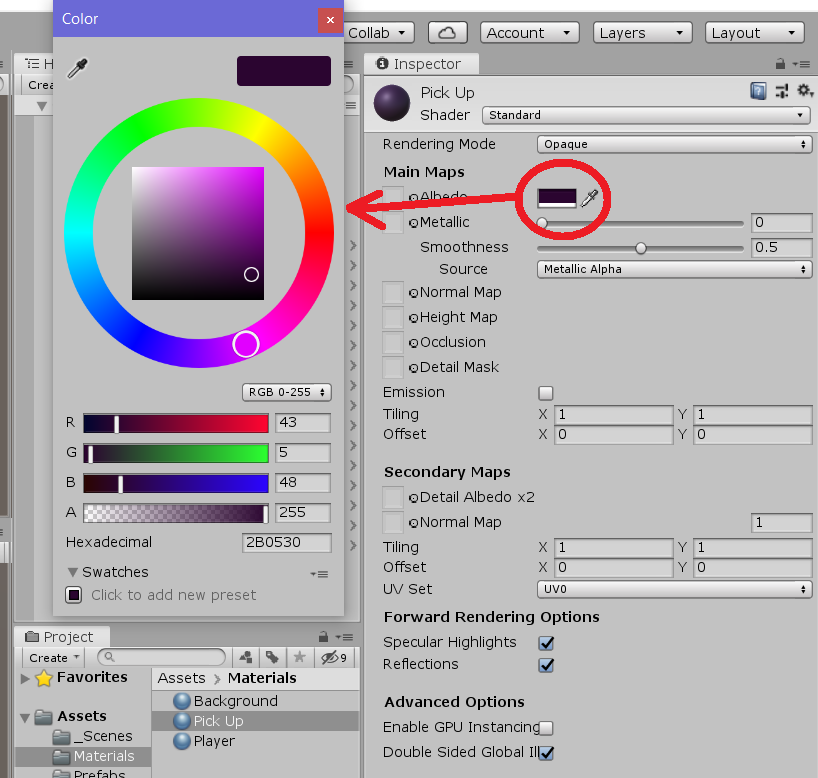
1. In the Project view, click on the materials folder and select the Background material. Use the Menu toolbar to duplicate it (Edit > Duplicate).



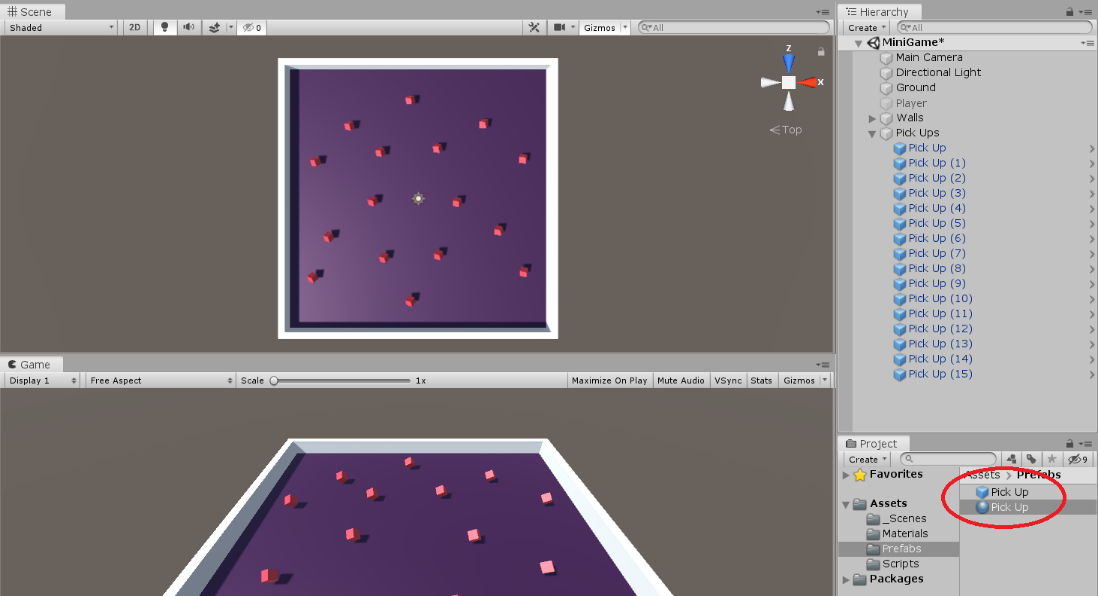
* 1. Name the newly duplicated material “Pick Up”. It should appear in the materials folder in the Project view. If it appears elsewhere in the Project view, click+drag it to the materials folder.



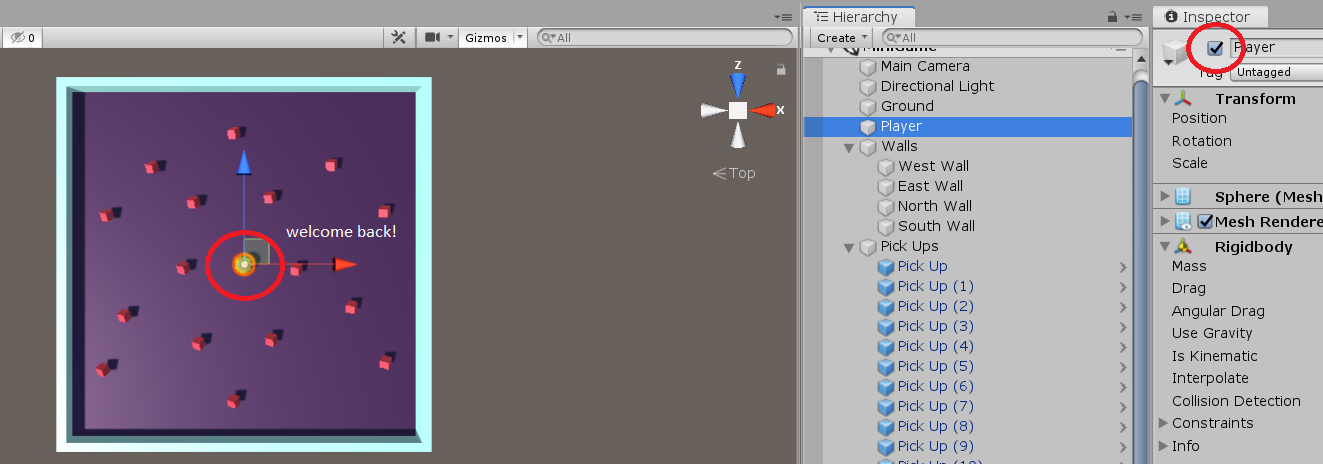
* 1. With the Pick Up material selected in the Project view, locate the color window in the Inspector and click on it. Use the color tools to select a color for your Pick Up objects.



* 1. In the Project view, click+drag the Pick Up material from the materials folder to the Prefabs folder. Then click+drag it to the Pick Up Prefab—this should change the color of all Pick Up objects in the Scene view. Click+drag the Pick Up material back to the materials folder to stay organized.

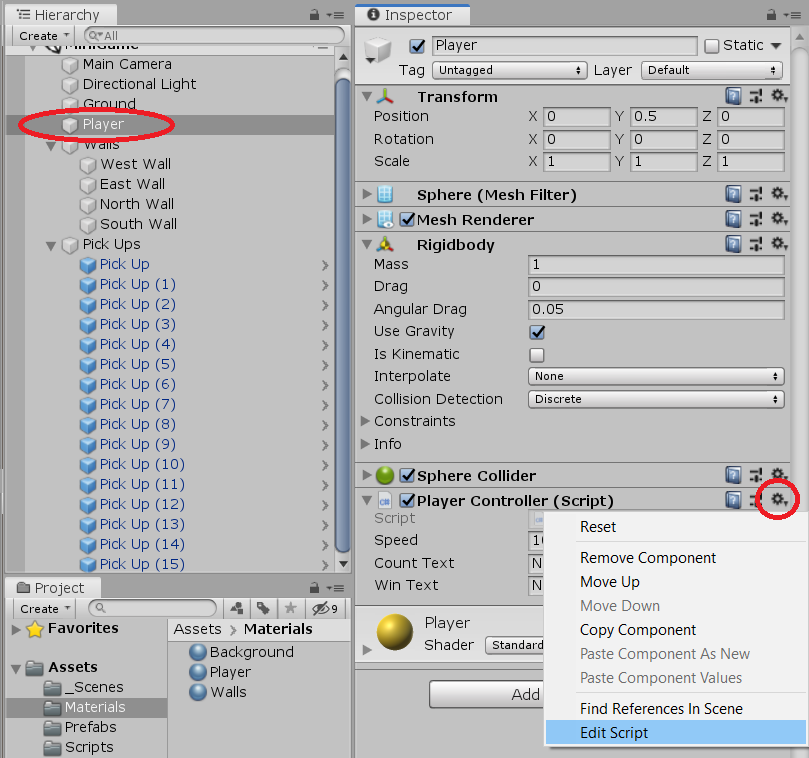


1. Select the Player object in the Hierarchy and check the “active” box at the top of the Inspector to place the Player object back into the game.

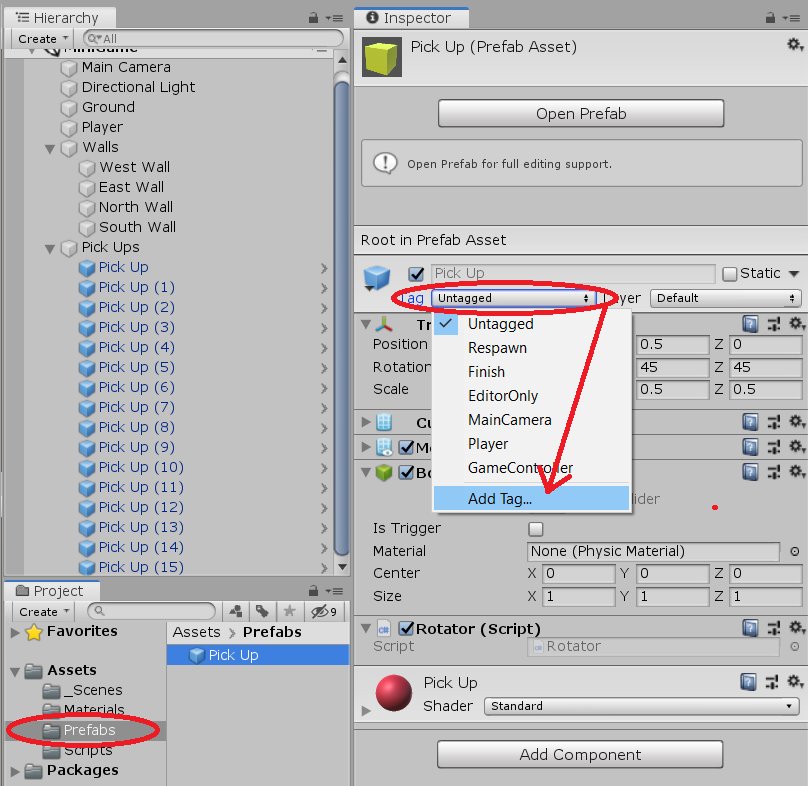


**Collecting the Pick Up objects**

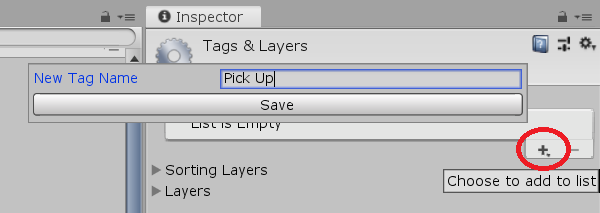
1. Select the Player object in the Hierarchy. In the Inspector, find the Player Controller script component and click the gear on the right. Select “Edit Script” to open the script editor. Copy+paste the code found @ <https://github.com/lcanner/RTFN> (PlayerController - Collecting the PickUp Objects).



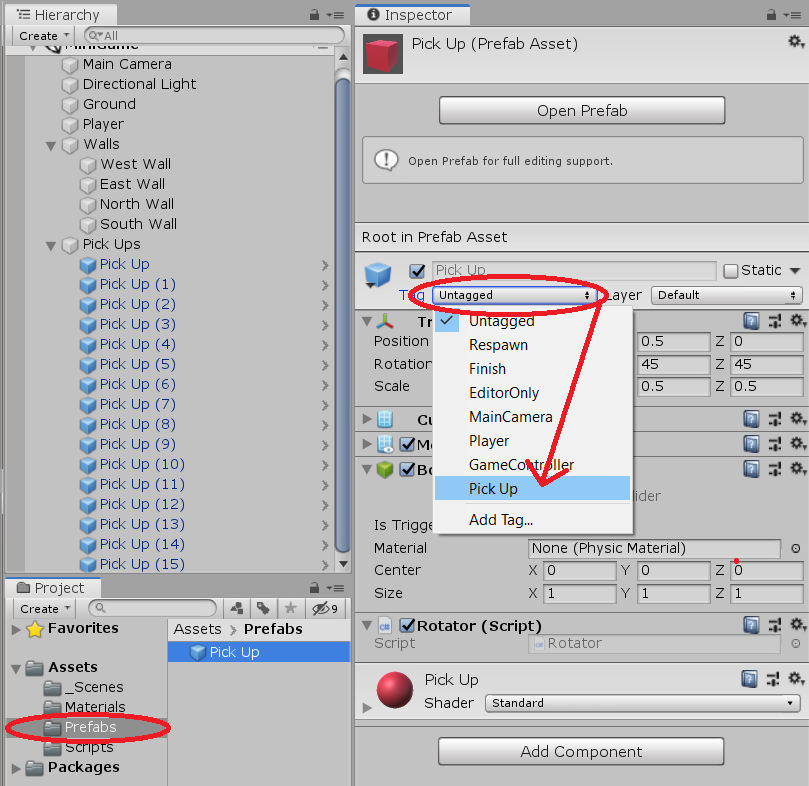
1. In the Project view, select the Pick Up Prefab and click the Tag Menu at the top of the Inspector. Choose “add tag”.



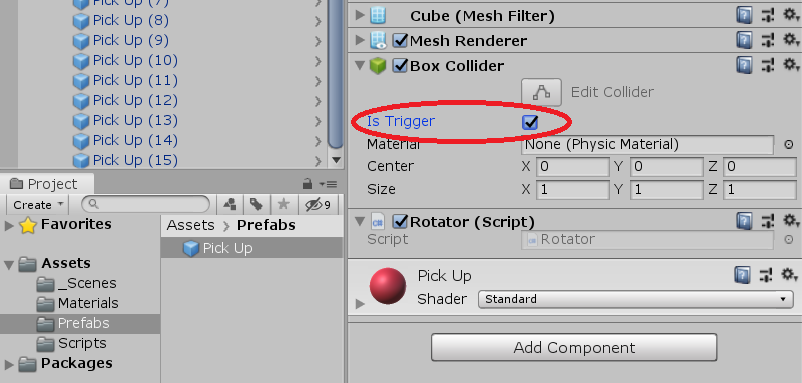
1. When the new tag Menu opens, click the “+” beneath “tags” to add a tag. Name the tag “Pick Up” (the tag is case sensitive).



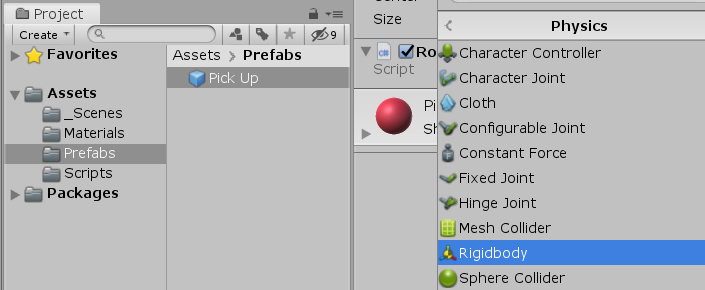
* 1. Select the tag drop down Menu in the Inspector again, and choose the newly created “Pick Up” tag.



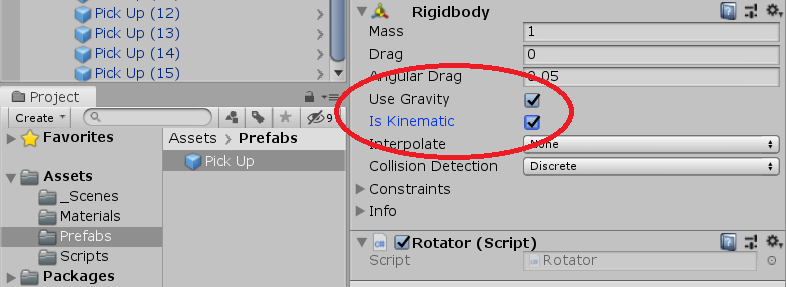
1. Select the Pick Up prefab in the Project view and look at the “box collider” component in the Inspector. Check the “is trigger” box. This will make Unity hold these objects in place when they experience a collision, and the impact will be noted internally. We can use this process to “trigger” an event happening whenever one of the Pick Up objects is touched by the Player object.



1. With the Pick Up prefab selected, click “Add Component” at the bottom of the Inspector, and choose “Physics” and then “Rigidbody”. The Rigidbody component is what makes 3D objects solid and allows them to experience collisions.

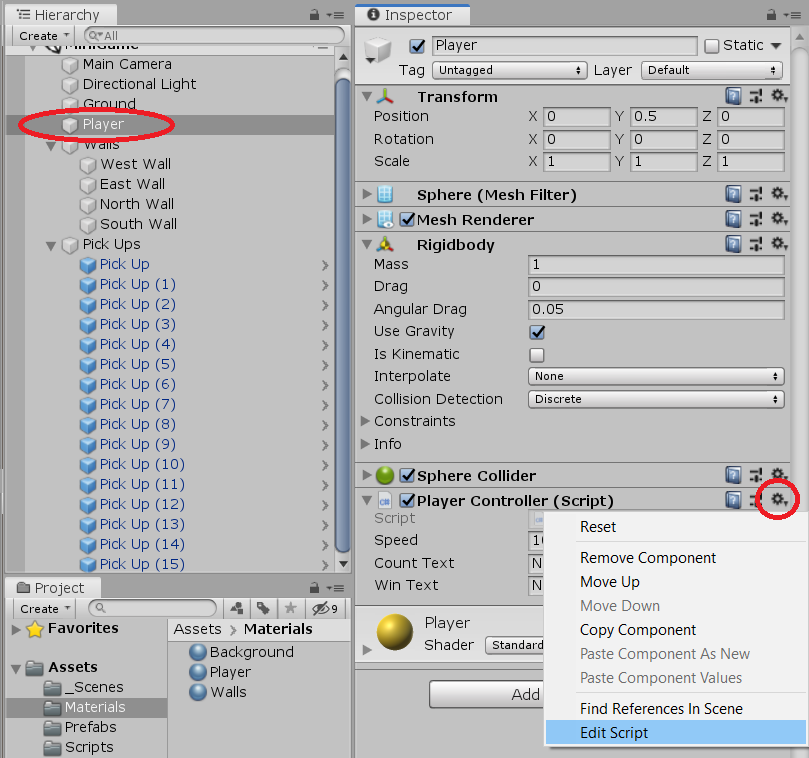


1. Check the “Use Gravity” and “Is Kinematic” boxes in the newly added Rigidbody component. Code in the Player Controller script says that when the Player object collides with a Pick Up object, the Pick Up object’s active state is set to “false”. This means that when a Pick Up is touched by the Player object, it will disappear—just like the Player object did when we deselected the “Is Active” box at the top of the Inspector.



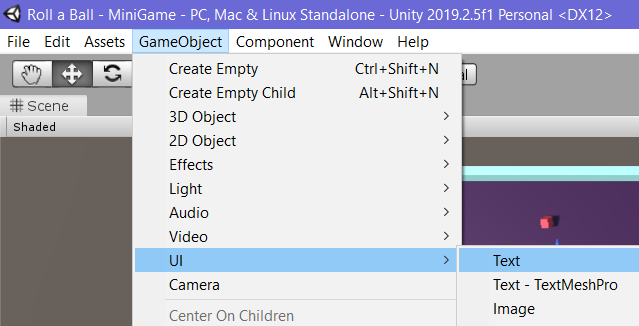
**Displaying the Score and Text**

1. In the Hierarchy, select the Player object. In the Inspector, locate the Player Controller script and click the gear to the right. Choose “edit script” to view the script text. <Copy+>paste the code found @ <https://github.com/lcanner/RTFN> (PlayerController - Displaying the Score and Text).

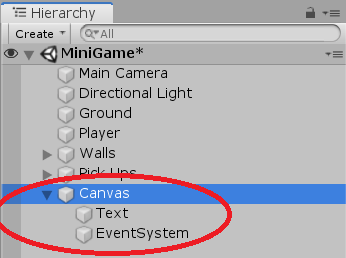


* 1. Locate the line of code “if (count >= 12)” in the PlayerController script. Change the 12 to match the number of Pick Up objects in your game.

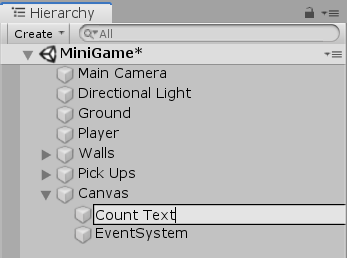
1. Create a new User Interface Text Element using the Menu toolbar (GameObject > UI > Text).



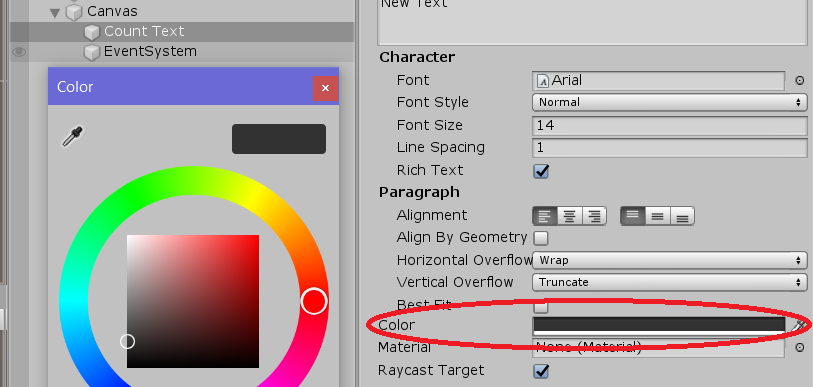
* 1. Notice that creating the UI element also creates a Folder called “Canvas” and an element called “EventSystem”. These are necessary for the User Interface to function. Make sure that both elements are attached to the Canvas folder. If they aren’t, click+drag them to the Canvas folder.



* 1. Name the newly created UI text element “Count Text”.



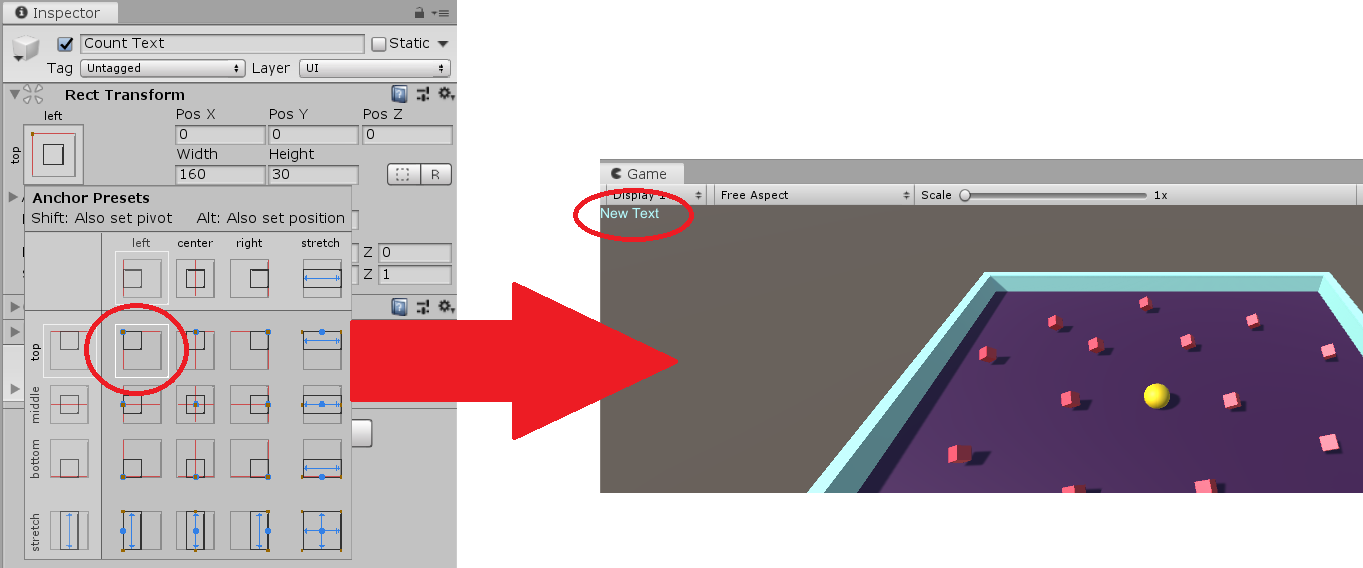
* 1. Select the Count Text Element in the Hierarchy. Change the color of the UI text by clicking the Color component at the bottom of the Inspector.



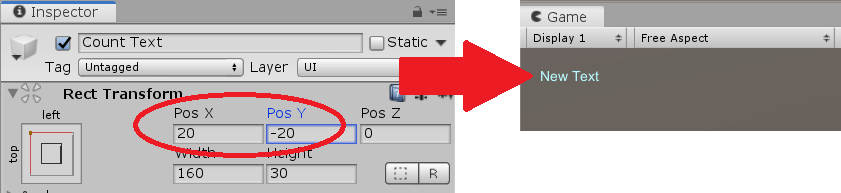
* 1. Open the Anchor Menu by clicking the square symbol in the “Rect Transform” component of the Inspector. The Anchor Menu is what’s used to position elements of the User Interface in different areas of the Game view.



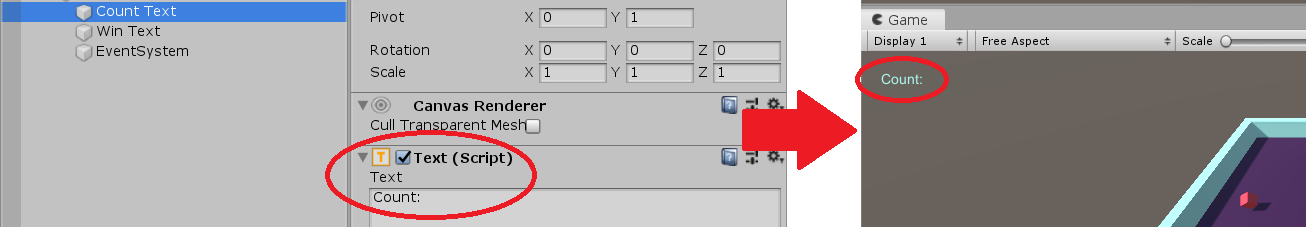
* 1. Hold the Shift and Alt keys on the keyboard and select the top-left alignment. Note that shift and alt set the pivot and the position of the text on the Canvas.



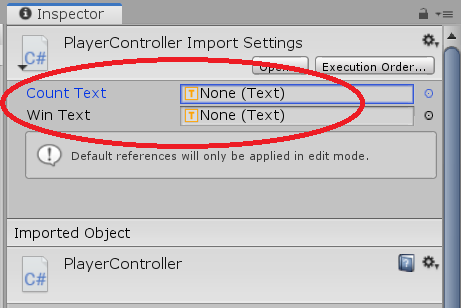
* 1. In the Inspector, change the Rect Transform positions of the Count Text element: X=20 , Y=-20). This will give the text more room in the upper left corner.



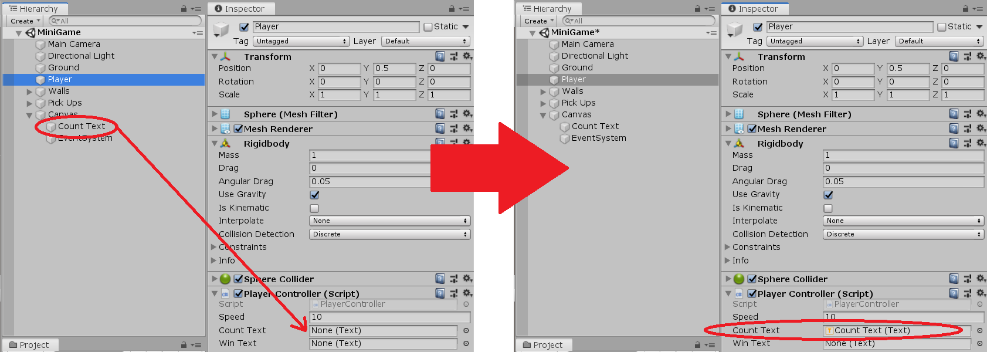
* 1. In the Text (Script) component of the Inspector, enter the text “Count:” This will change the displayed text in the corner. The PlayerController script is coded to automatically add a number at the end of the displayed text.



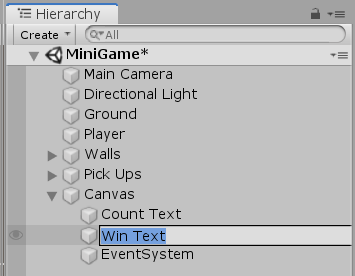
1. Open the PlayerController script in the Project view. [[PlayerController 5:35]](https://youtu.be/bFSLI2cmYYo?t=324) If written correctly, a new field called “Count Text” should appear at the top of the Player Controller Script in the Inspector.



1. Select the Player object in the Hierarchy. Click+drag the Count Text UI element from the Hierarchy and drop it in the newly created Count Text field of the Player Controller Script in the Inspector. If done correctly, the count displayed in the upper left of the game will go up by one each time an object is collected.



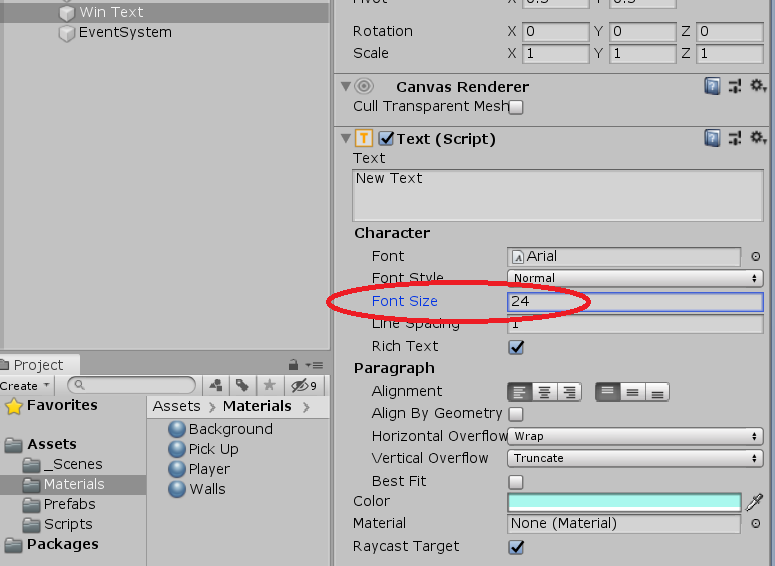
1. Create a new User Interface Text element (GameObject > UI > Text) and name it “Win Text”. Check to make sure it is in the Canvas folder in the Hierarchy—if it’s not, click+drag it to the Canvas folder.



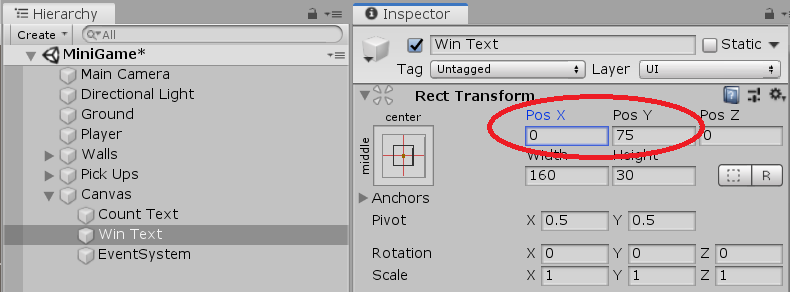
* 1. Use the color component in the Inspector to change the color of the text to match the color of the Count Text.



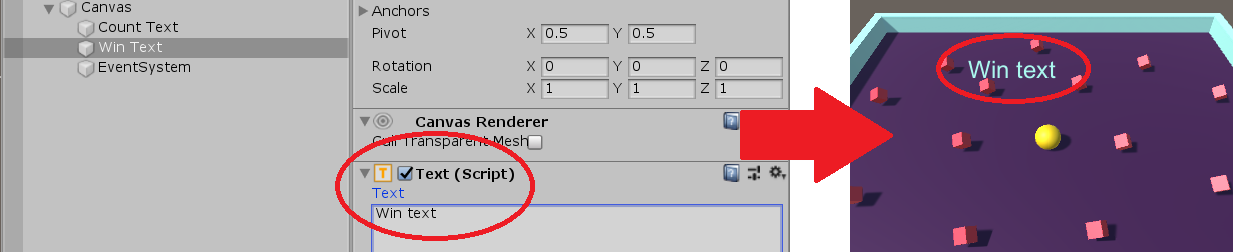
* 1. Increase the size of the Win Text using the “font size” field in the Inspector (recommended is size 24).



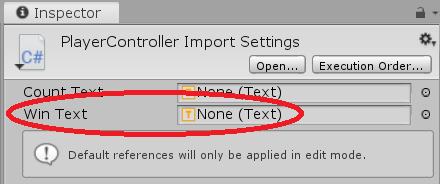
* 1. Change the Rect Transform position of the Win Text: (X=0 , Y=75)



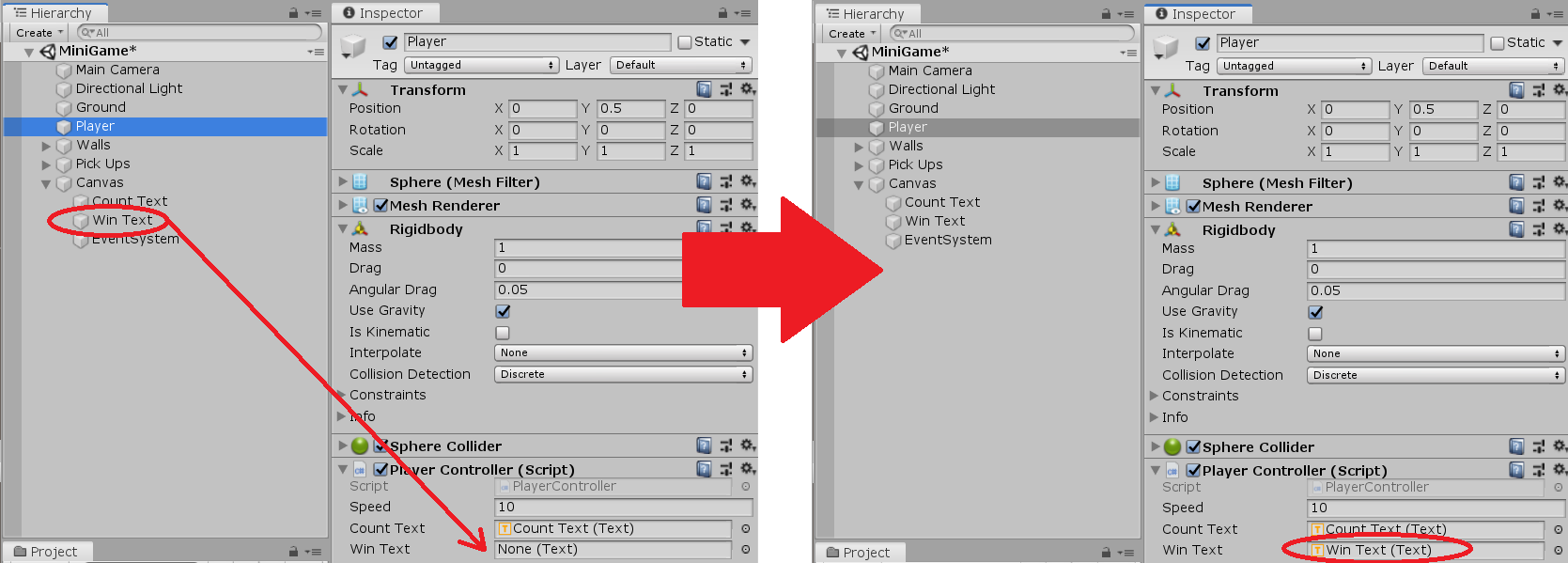
* 1. In the Text (Script) section of the Inspector, enter the phrase “Win Text” This will change the placeholder text in the corner. The PlayerController script is coded to display the real text when all of the Pick Up objects have been collected. This placeholder is used to gauge where on the screen the PlayerController script-generated text will appear.



1. Open the PlayerController script in the Project view [[PlayerController 9:40]](https://youtu.be/bFSLI2cmYYo?t=579) . If written correctly, a new field called “Win Text” should appear at the bottom of the Player Controller Script in the Inspector.

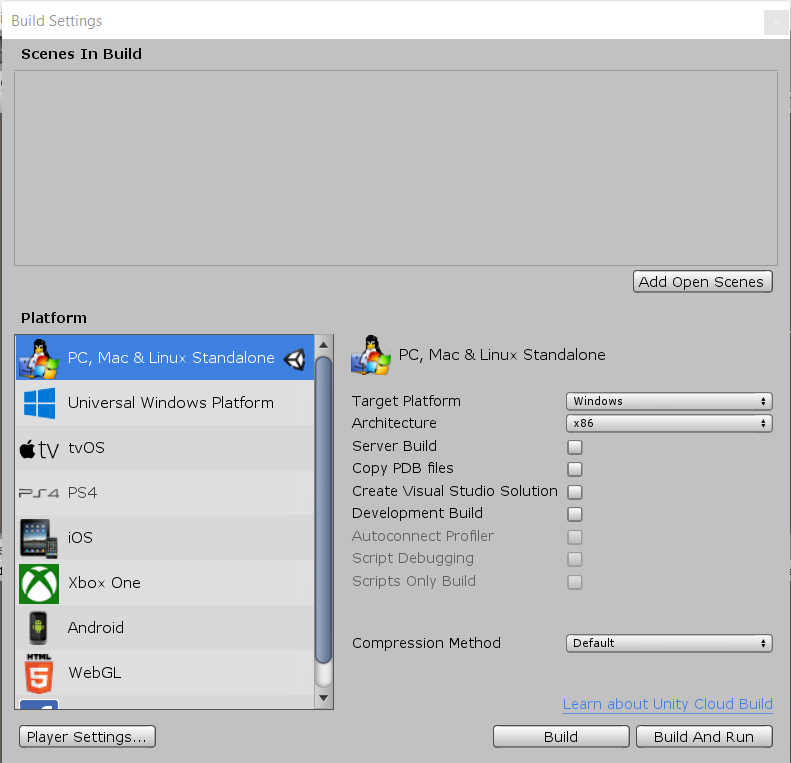


1. Select the Player object in the Hierarchy. Click+drag the Win Text UI element from the Hierarchy and drop it in the newly created Win Text field of the Player Controller Script in the Inspector. If done correctly, the text should appear upon collecting the final object.

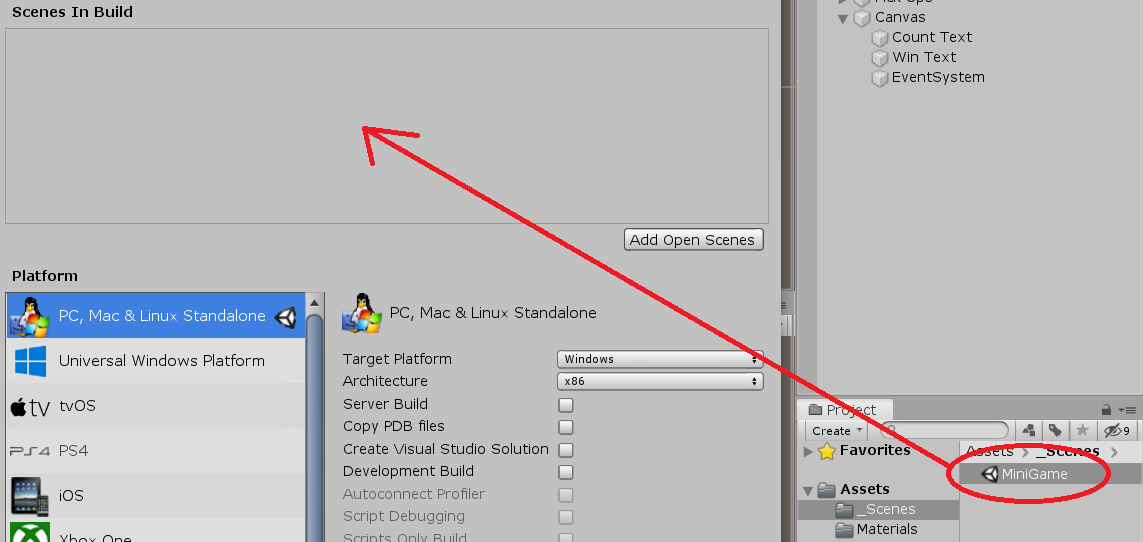


**Building a Game**

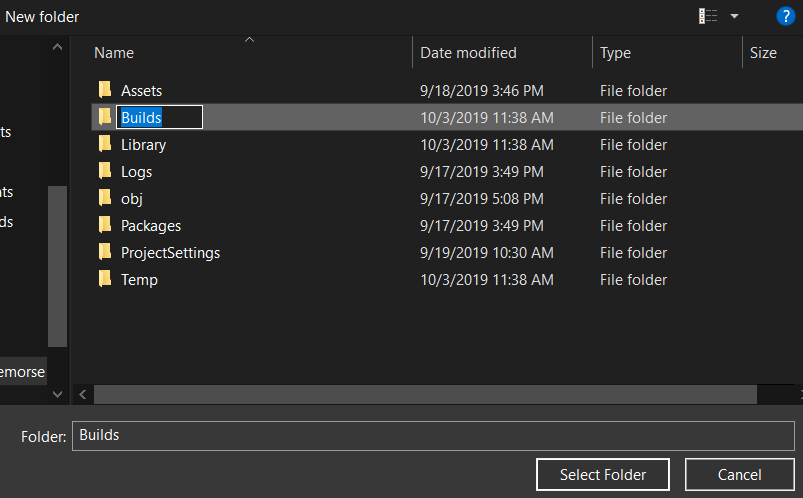
1. Open the “Build Settings” window (File > Build Settings).



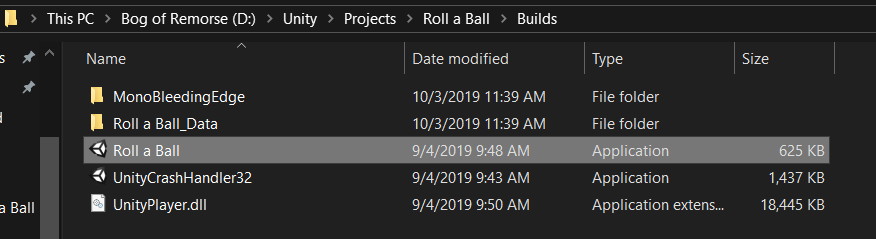
* 1. Select “PC, Mac & Linux Standalone” as the platform. Click+drag the MiniGame scene from the \_Scenes folder in the Project view to the “Scenes in Build” window. Click “Build” button.



* 1. Click the “new folder” button at the lower left of the build save window, and name the folder “Builds”.



* 1. Enter the name "Roll a Ball” in the “Save As field at the top of the window and click “save” in the lower right.



1. Navigate to the newly created Builds folder and open the newly created Roll a Ball application. There’s your game, you did it!!!