Unity3D Lecture Note 1

Game Engine

A **game engine** is a software framework d esigned for the creation and development of video games.

Video game developers use them to create games for video game consoles, mobile de vices and personal computers.

The core functionality:

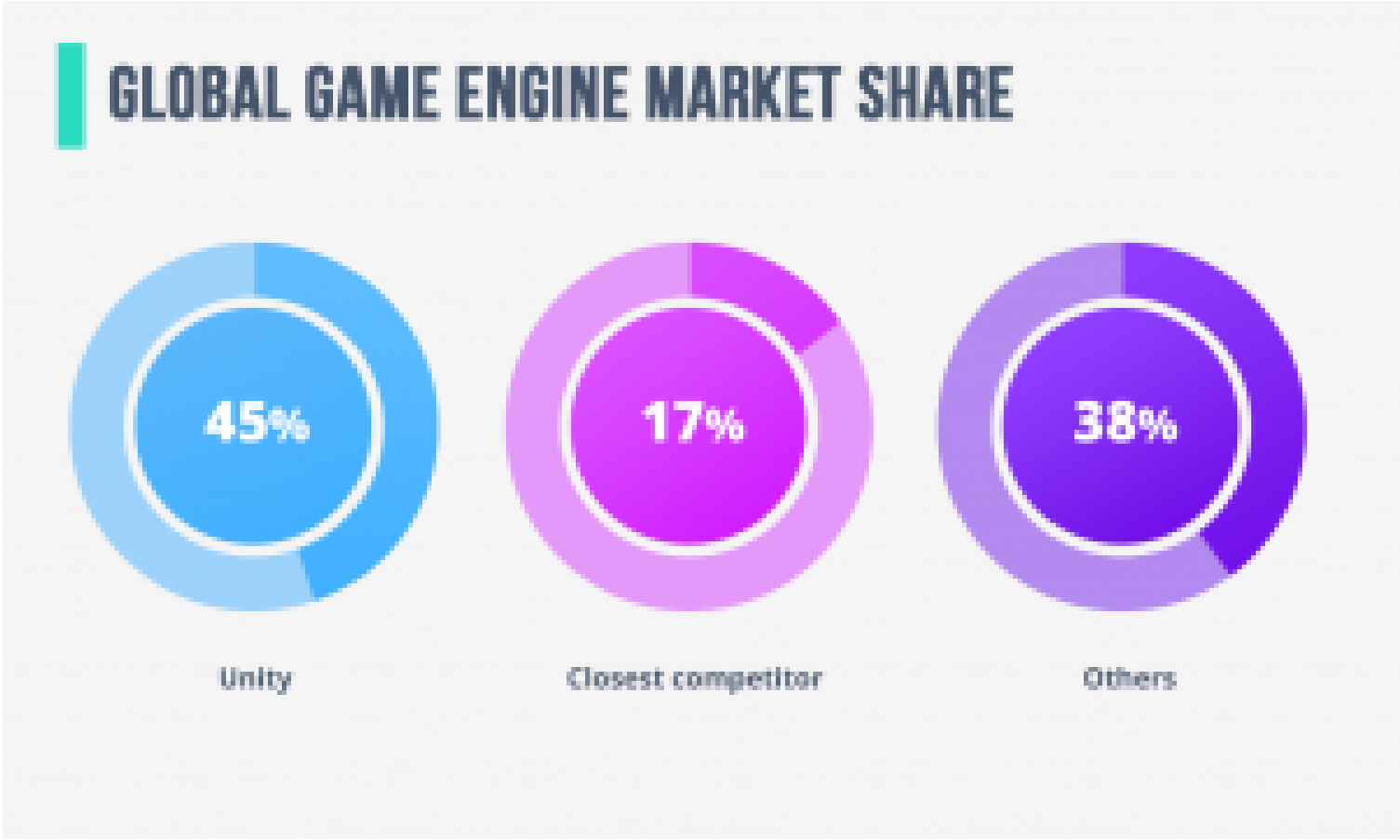
- rendering engine (“renderer”) for 2D or 3D graphics,
- physics engine
- collision detection
- sound
- scripting
- animation
- artificial intelligence
- networking
- Etc


unity
GameMaker
Studio
Construct 2
UNREAL
ENGINE

Engine	Programming ?	2D or 3D?	Available for	Exports to
Construct 2	No	2D (All Genres)	Windows	Desktop, Consoles, Mobile, Web
GameMaker: Studio	No	2D (All Genres)	Windows	Desktop, Consoles, Mobile, Web
Unity	Yes	2D + 3D (All Genres)	Windows, Mac	Desktop, Consoles, Mobile, Web

Unreal Engine	Yes	3D (All Genres)	Windows, Mac	Desktop, Consoles, Mobile
Clickteam Fusion	No	2D (All Genres)	Windows	Desktop, Mobile, Web
Stencyl	No	2D (All Genres)	Windows, Mac, Linux	Desktop, Mobile, Web
PlayCanvas	Yes	2D + 3D (All Genres)	Windows	Desktop, Mobile, Web
Torque 2D/3D	Yes	2D + 3D (All Genres)	Windows	Desktop, Consoles, Mobile
CraftStudio	No	2D + 3D (All Genres)	Windows, Mac, Linux	Desktop

2019년 Game Engine Market Share



iClosets competitors (Unreal Engine, GameMaker, and CryEngine)

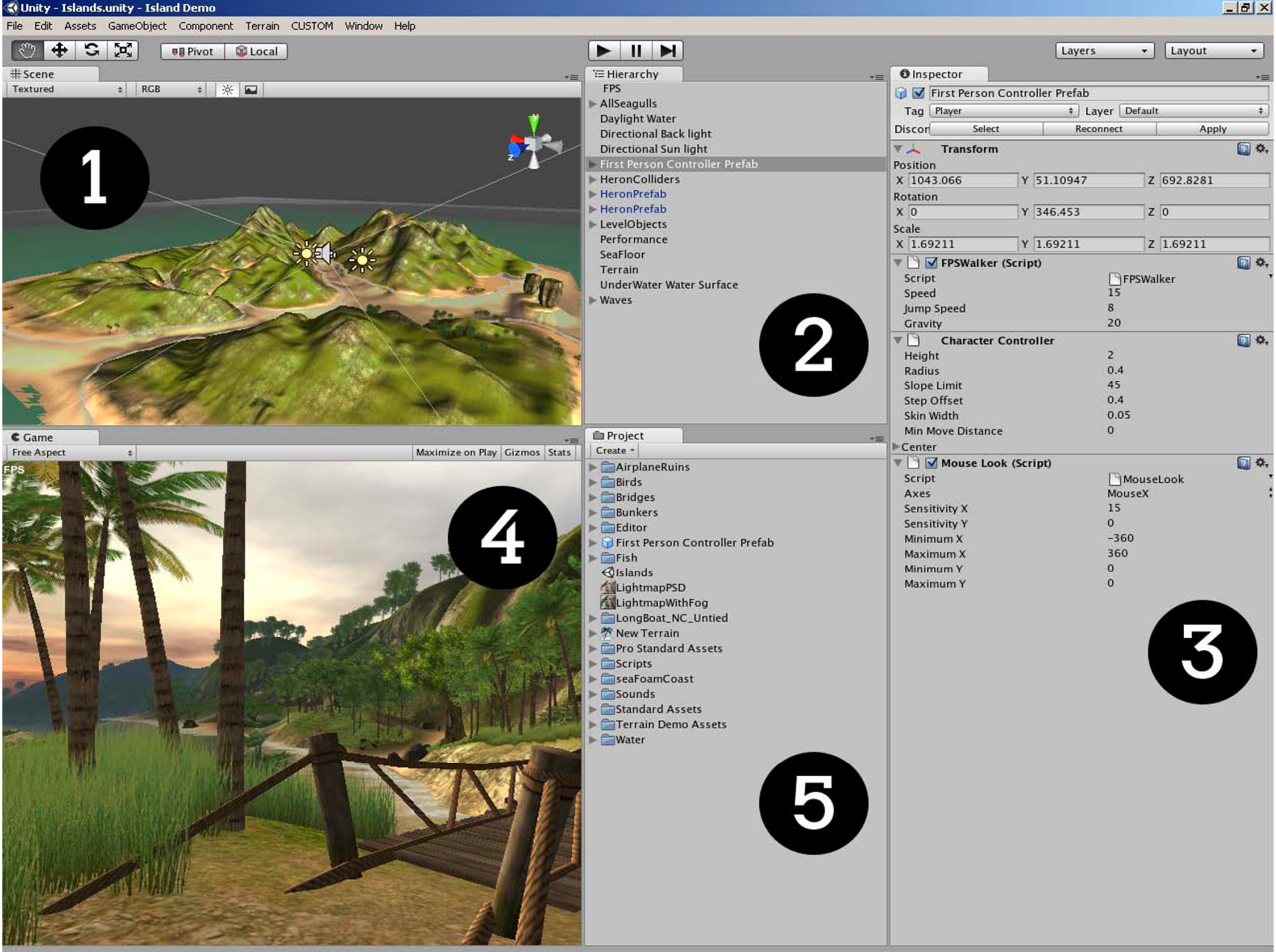
Install Unity3D

www.unity3d.com

Use Personal Edition for free


Unity - Game engine, to
https://unity3d.com/kr/unity
게임 창조. 유저와의 연결. 성공이라는 열매
Unity를 도입해보세요. 큰 수고를 들이지 않고도 고도로 최적화되고 아름다운 2D 또는 3D 게임을 창조하여, 클릭 한 번으로 다양한 플랫폼에 배포할 수 있습니다. 또한, Unity에 통합된 각종 서비스를 통해 개발 속도를 올리고 게임을 최적화하며, 사용자와 연결하고 성공을 거머쥘 수 있습니다.
Personal
무료
Plus
$35/월
Pro
$125/월
기업
맞춤형 설치
Unity를 내 손에
업계를 선도하는 멀티플랫폼 지원
멀티플랫폼 게임 엔진으로 업계 전반에 널리 알려진 Unity와 함께라면 더 적은 수고로 더 많은 플랫폼에 작품을 선보일 수 있습니다. 모바일, VR, 데스크톱, 웹, 콘솔 및 TV 등 다양한 플랫폼을 지원하는 Unity의 원클릭 배포 기능을 만끽해보세요.
iOS
PS4
PS VITA
3DS
Gear VR
tvOS
fireoss
3:19 PM
11/20/2016

Unity3D User Interface


Unity - Islands.unity - Island Demo
File Edit Assets GameObject Component Terrain CUSTOM Window Help
Pivot Local
Layers Layout
Scene
Textured RGB
1
Hierarchy
FPS
AllSeagulls
Daylight Water
Directional Back light
Directional Sun light
First Person Controller Prefab
HeronColliders
HeronPrefab
HeronPrefab
LevelObjects
Performance
SeaFloor
Terrain
UnderWater Water Surface
Waves
2
Inspector
First Person Controller Prefab
Tag Player Layer Default
Discor Select Reconnect Apply
Transform
Position
X 1043.066 Y 51.10947 Z 692.8281
Rotation
X 0 Y 346.453 Z 0
Scale
X 1.69211 Y 1.69211 Z 1.69211
FPSWalker (Script)
Script FPSWalker
Speed 15
Jump Speed 8
Gravity 20
Character Controller
Height 2
Radius 0.4
Slope Limit 45
Step Offset 0.4
Skin Width 0.05
Min Move Distance 0
Center
Mouse Look (Script)
Script MouseLook
Axes MouseX
Sensitivity X 15
Sensitivity Y 0
Minimum X -360
Maximum X 360
Minimum Y 0
Maximum Y 0
3
Game
Free Aspect
Maximize on Play Gizmos Stats
FPS
4
Project
Create
AirplaneRuins
Birds
Bridges
Bunkers
Editor
First Person Controller Prefab
Fish
Islands
LightmapPSD
LightmapWithFog
LongBoat_NC_Untied
New Terrain
Pro Standard Assets
Scripts
seaFoamCoast
Sounds
Standard Assets
Terrain Demo Assets
Water
5

GUI Windows

- **Scene** [1]—where the game is constructed
- **Hierarchy** [2]—a list of GameObjects in the scene
- **Inspector** [3]—settings for currently selected asset/object

- **Game** [4]—the preview window, active only in play mode
- **Project** [5]—a list of your project's assets, acts as a library

Unity 3D Terms

Assets

- building blocks of all Unity projects

- graphics (textures), models, sound files. The assets you use to create the scenes are stored in a folder called Assets

Scenes - scenes are individual levels, are as of game content. Scenes can be loaded on demand.

Game Objects

- The assets used in the scene become game objects (script name) .

- All game objects have at least one component, which is the Transform component.

Components

- groups of properties of an game object..

- Attach components to an object to add parts of the game engine, e.g a physics component, or a script component

Scripts

- components used to add, extend or modify behavior of game objects.

- Unity uses a Behavior class to facilitate the use of custom behaviours.

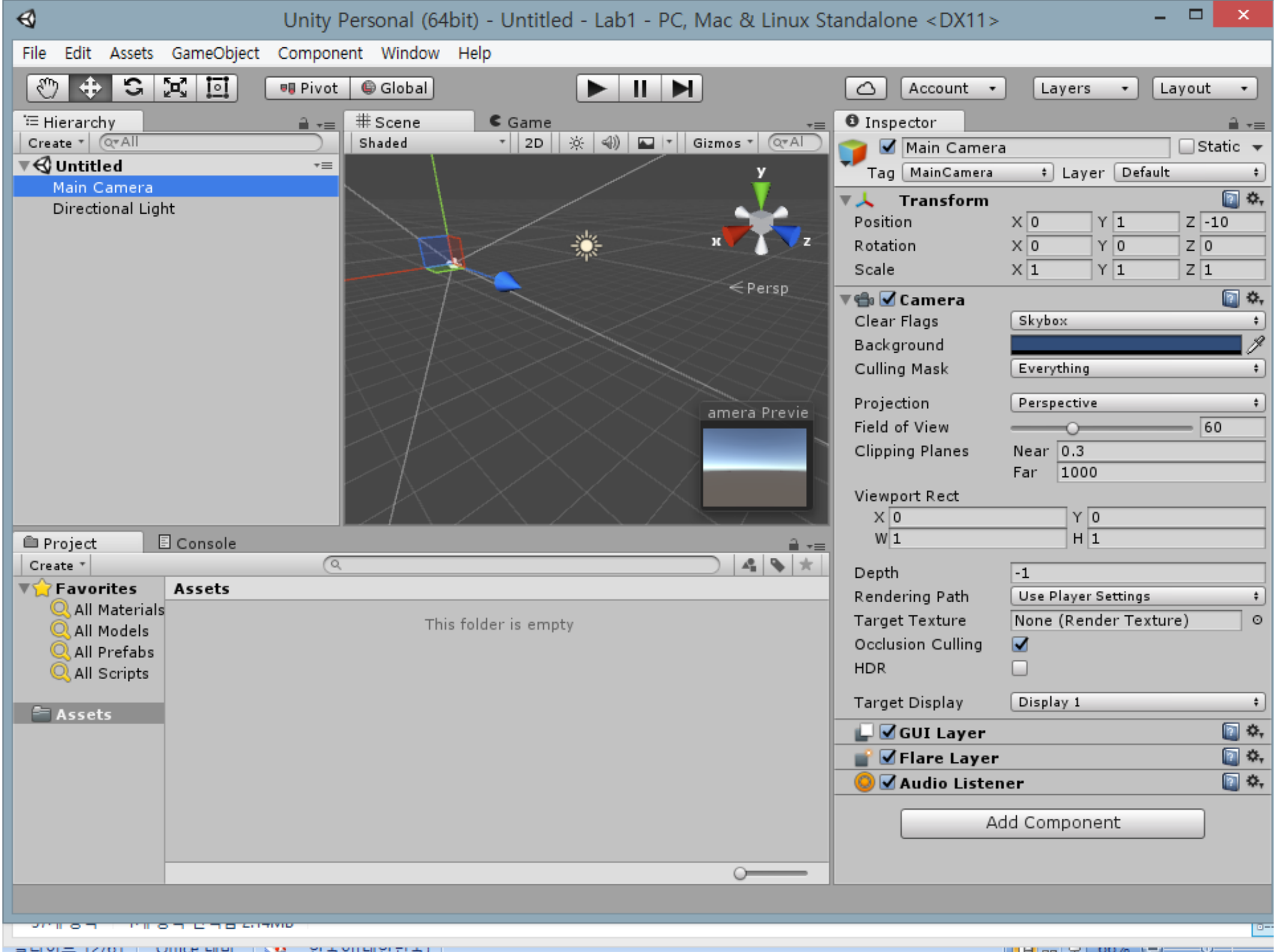
Prefabs

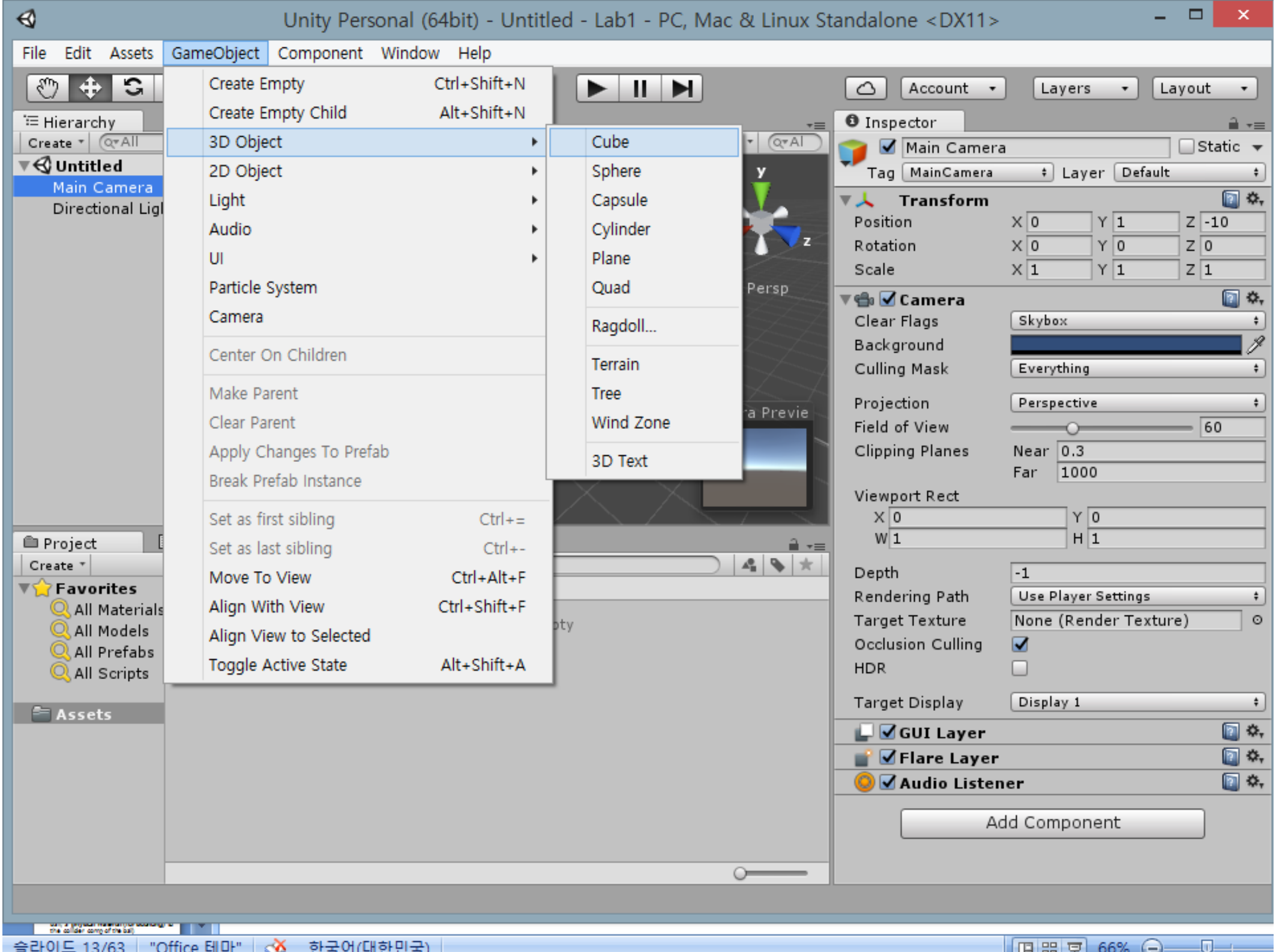
- prefabricated game objects with stored associated components and configuration.

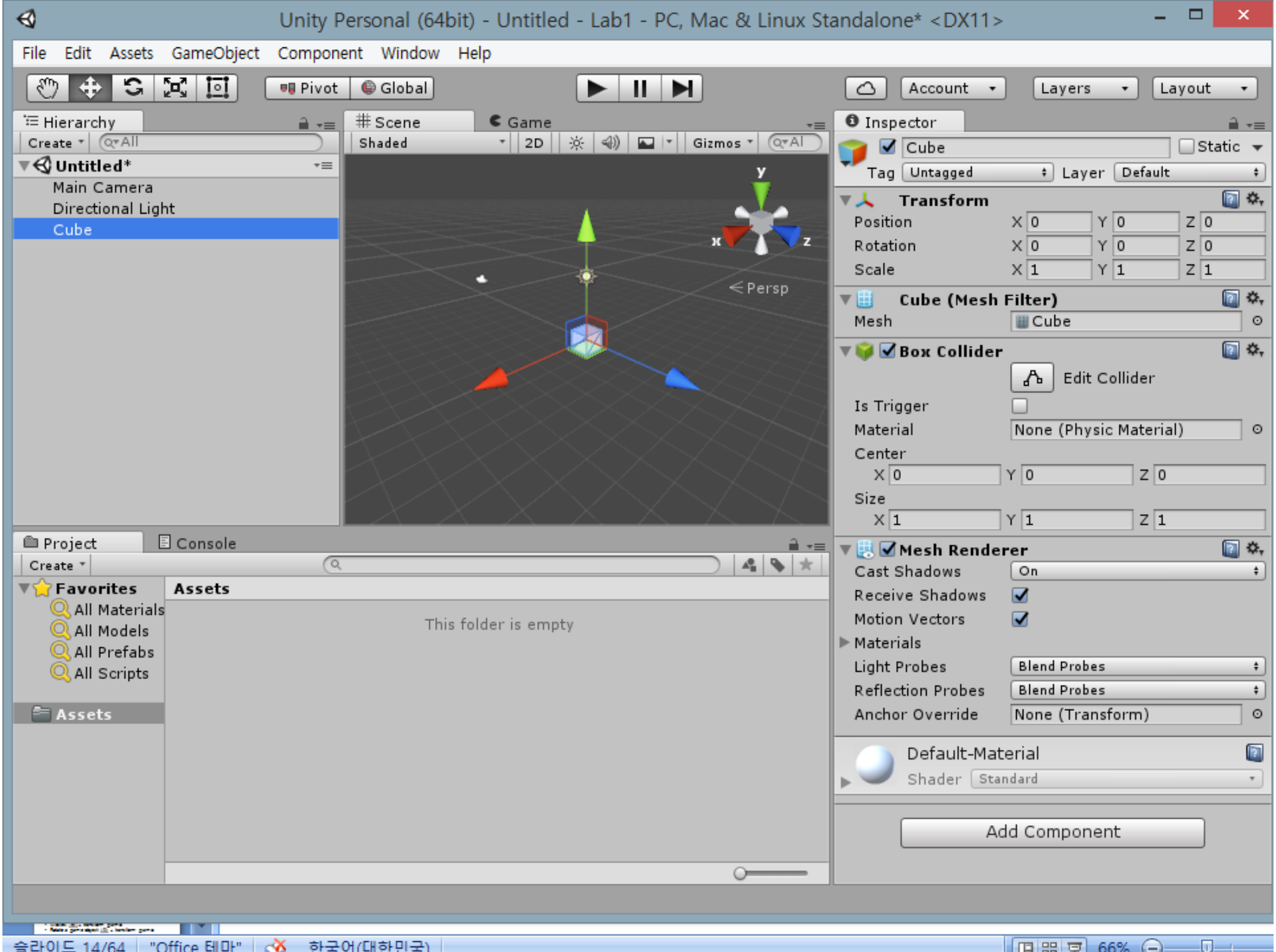
- Prefabs allow functional game objects to be reused in scenes (spawned) or imported into other projects as external assets.

Lab 1

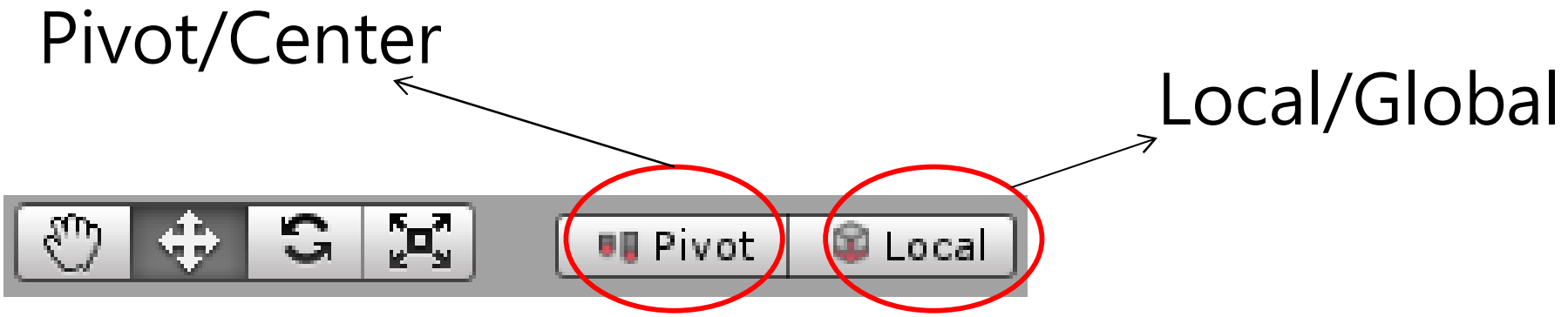
- Exercise: Create a sample 3D project
- GUI Windows
  - Project, Hierarchy, Scene, Game, Inspector
- Game Object & Attribute & Component & Asset
  - A **game object** has a lot of attributes, which are grouped into **components**.
  - An **asset** can be the **values of attributes** belonging to components


Unity Personal (64bit) - Untitled - Lab1 - PC, Mac & Linux Standalone <DX11>
File Edit Assets GameObject Component Window Help
Pivot Global
Account Layers Layout
Hierarchy
Create All
Untitled
Main Camera
Directional Light
Scene Game
Shaded 2D Gizmos All
y x z
Persp
amera Previe
Project Console
Create
Favorites
All Materials
All Models
All Prefabs
All Scripts
Assets
Assets
This folder is empty
Inspector
Main Camera Static
Tag MainCamera Layer Default
Transform
Position X 0 Y 1 Z -10
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Camera
Clear Flags Skybox
Background
Culling Mask Everything
Projection Perspective
Field of View 60
Clipping Planes Near 0.3 Far 1000
Viewport Rect X 0 Y 0 W 1 H 1
Depth -1
Rendering Path Use Player Settings
Target Texture None (Render Texture)
Occlusion Culling
HDR
Target Display Display 1
GUI Layer
Flare Layer
Audio Listener
Add Component


Unity Personal (64bit) - Untitled - Lab1 - PC, Mac & Linux Standalone <DX11>
File Edit Assets GameObject Component Window Help
Create Empty Ctrl+Shift+N
Create Empty Child Alt+Shift+N
3D Object
2D Object
Light
Audio
UI
Particle System
Camera
Center On Children
Make Parent
Clear Parent
Apply Changes To Prefab
Break Prefab Instance
Set as first sibling Ctrl+=
Set as last sibling Ctrl+-
Move To View Ctrl+Alt+F
Align With View Ctrl+Shift+F
Align View to Selected
Toggle Active State Alt+Shift+A
Cube
Sphere
Capsule
Cylinder
Plane
Quad
Ragdoll...
Terrain
Tree
Wind Zone
3D Text
Account
Layers
Layout
Hierarchy
Create
Untitled
Main Camera
Directional Ligl
Project
Favorites
All Materials
All Models
All Prefabs
All Scripts
Assets
Persp
Inspector
Main Camera
Static
Tag MainCamera
Layer Default
Transform
Position X 0 Y 1 Z -10
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Camera
Clear Flags Skybox
Background
Culling Mask Everything
Projection Perspective
Field of View 60
Clipping Planes Near 0.3 Far 1000
Viewport Rect X 0 Y 0 W 1 H 1
Depth -1
Rendering Path Use Player Settings
Target Texture None (Render Texture)
Occlusion Culling
HDR
Target Display Display 1
GUI Layer
Flare Layer
Audio Listener
Add Component


Unity Personal (64bit) - Untitled - Lab1 - PC, Mac & Linux Standalone* <DX11>
File Edit Assets GameObject Component Window Help
Pivot Global
Account Layers Layout
Hierarchy
Create All
Untitled*
Main Camera
Directional Light
Cube
Scene Game
Shaded 2D Gizmos All
y x z
Persp
Inspector
Cube Static
Tag Untagged Layer Default
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Cube (Mesh Filter)
Mesh Cube
Box Collider
Edit Collider
Is Trigger
Material None (Physic Material)
Center X 0 Y 0 Z 0
Size X 1 Y 1 Z 1
Mesh Renderer
Cast Shadows On
Receive Shadows
Motion Vectors
Materials
Light Probes Blend Probes
Reflection Probes Blend Probes
Anchor Override None (Transform)
Default-Material
Shader Standard
Add Component
Project Console
Create
Favorites
All Materials
All Models
All Prefabs
All Scripts
Assets
Assets
This folder is empty

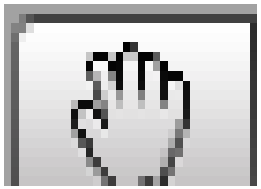
Lab 2



Exercise: Try the buttons above.

Change “viewing direction”

in the scene:



Mouse의 scroll wheel

Mouse의 오른쪽 버튼을 누른 채 move

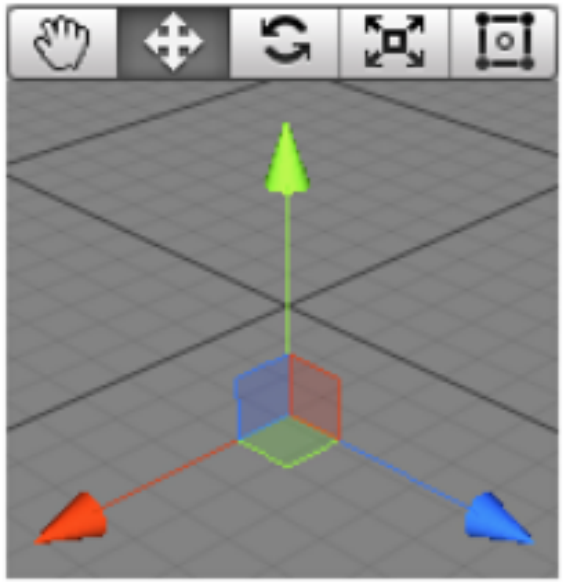
F key: 선택된 game object가 화면 중앙에 위치

Game Object 조작

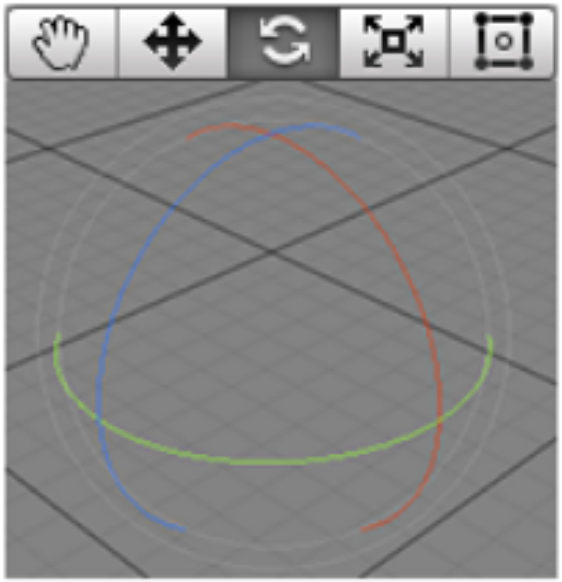
-Moves:  + transform gizmo

-Rotate:  + transform gizmo

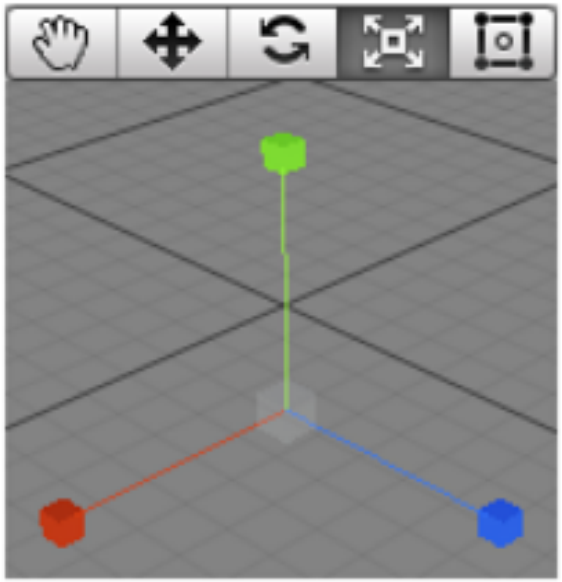
-Scale:  + transform gizmo



Translate (W)



Rotate (E)



Scale (R)

:

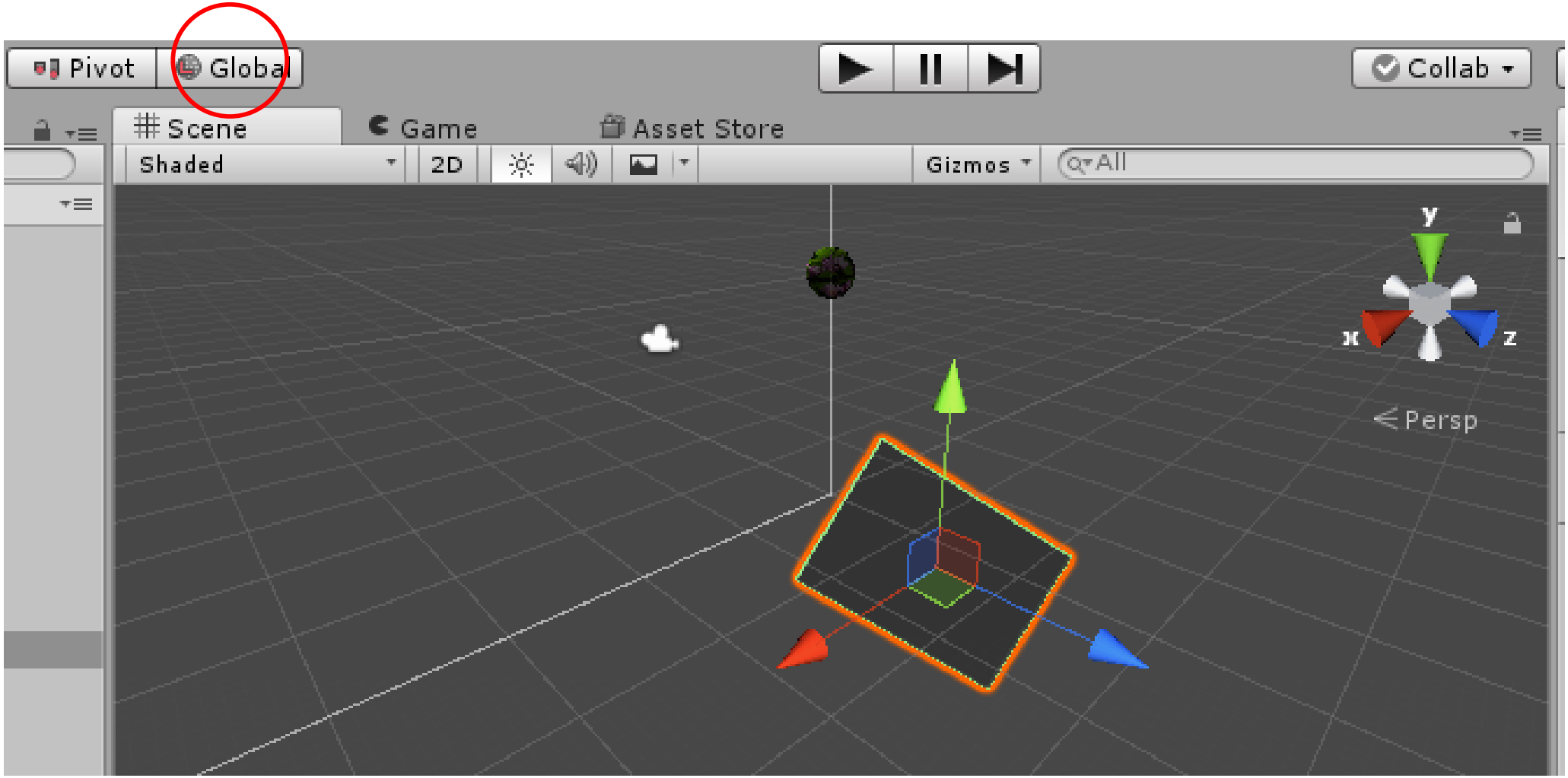
Pivot points control how objects rotate and scale, and also represent the exact locations of objects in space.
How to change the pivot point of an object :
Using a script or a modeling tool.

변환 기즈모(Transform Gizmo)의 위치:

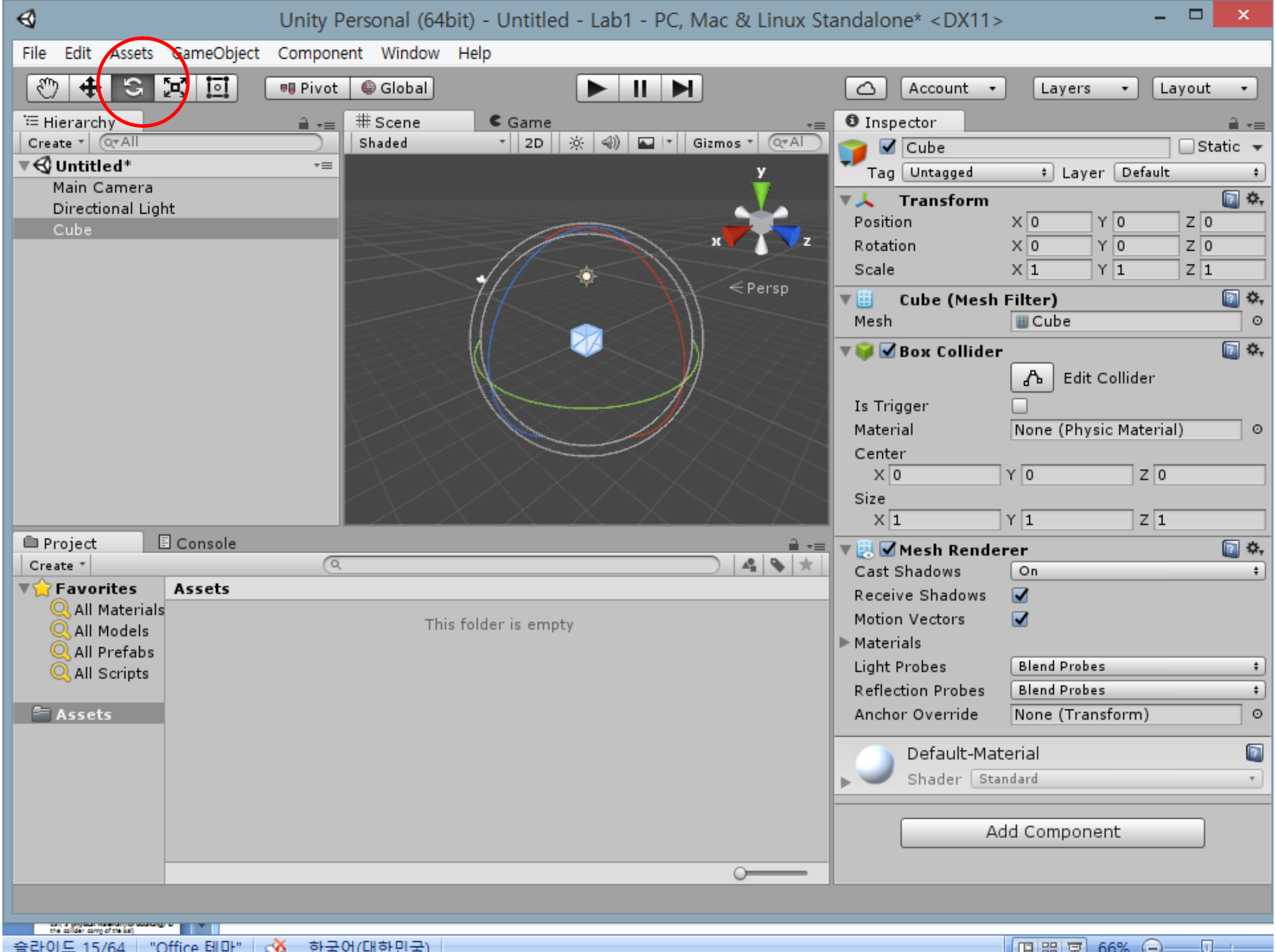
- Center는 오브젝트의 바운드의 중심에 기즈모를 배치.
- Pivot은 메쉬의 피벗 점에 기즈모를 배치.

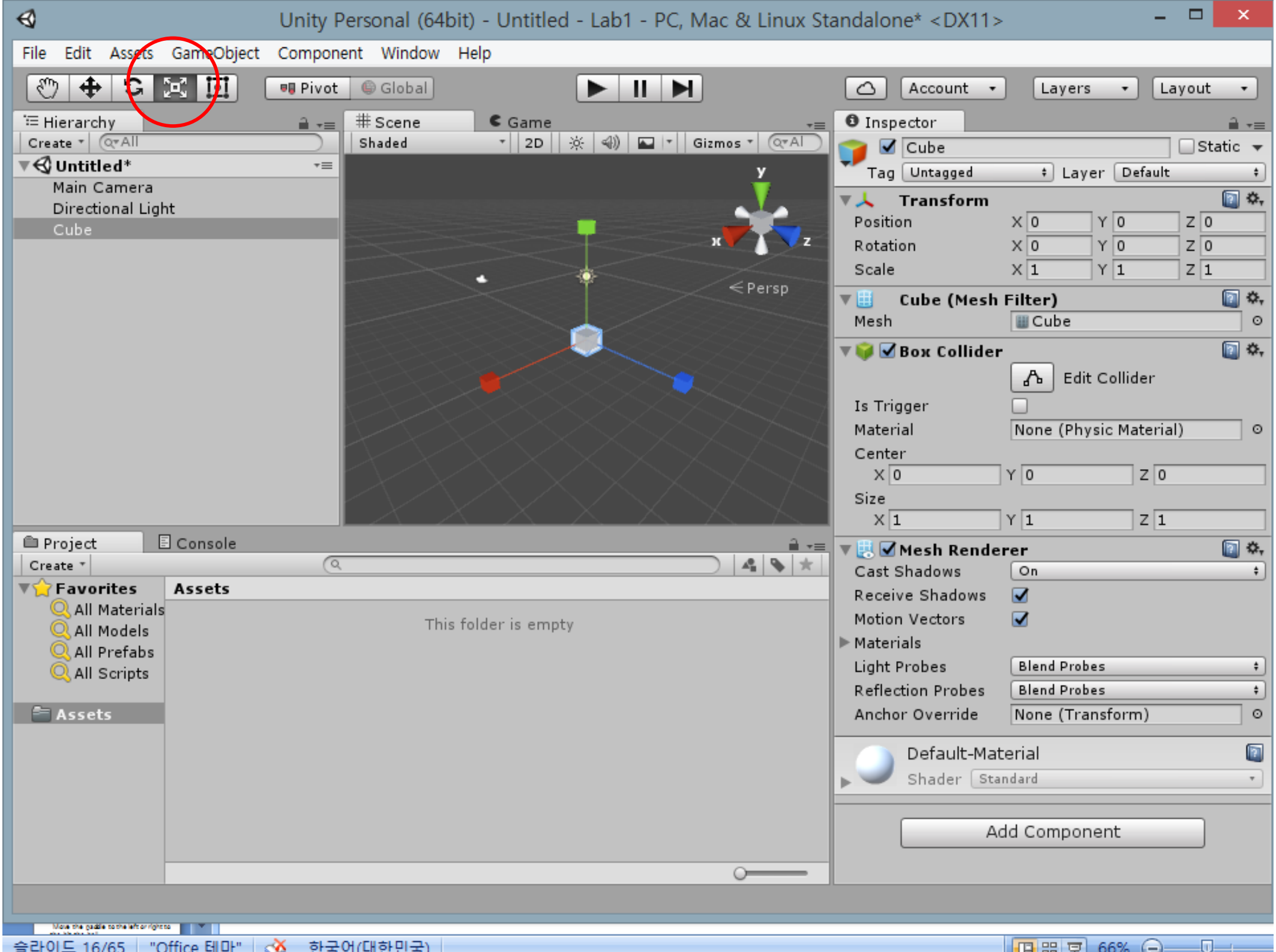
변환 기즈모(Transform Gizmo)의 방향:

- Local: 오브젝트의 회전에 대해 기즈모의 회전을 유지합니다.
- Global: 오브젝트의 회전에 대해 기즈모를 월드 공간의 방향으로 고정합니다.



평면이 회전되어 있지만, transform gizmo는 우측상단 좌표계와 정렬 유지


Unity Personal (64bit) - Untitled - Lab1 - PC, Mac & Linux Standalone* <DX11>
File Edit Assets GameObject Component Window Help
Pivot Global
Account Layers Layout
Hierarchy
Create All
Untitled*
Main Camera
Directional Light
Cube
Scene Game
Shaded 2D Gizmos All
y x z
Persp
Inspector
Cube Static
Tag Untagged Layer Default
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Cube (Mesh Filter)
Mesh Cube
Box Collider
Edit Collider
Is Trigger
Material None (Physic Material)
Center X 0 Y 0 Z 0
Size X 1 Y 1 Z 1
Mesh Renderer
Cast Shadows On
Receive Shadows
Motion Vectors
Materials
Light Probes Blend Probes
Reflection Probes Blend Probes
Anchor Override None (Transform)
Default-Material
Shader Standard
Add Component
Project Console
Create
Favorites
All Materials
All Models
All Prefabs
All Scripts
Assets
Assets
This folder is empty


Unity Personal (64bit) - Untitled - Lab1 - PC, Mac & Linux Standalone* <DX11>
File Edit Assets GameObject Component Window Help
Pivot Global
Account Layers Layout
Hierarchy
Create All
Untitled*
Main Camera
Directional Light
Cube
Scene Game
Shaded 2D Gizmos All
y x z
Persp
Inspector
Cube Static
Tag Untagged Layer Default
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Cube (Mesh Filter)
Mesh Cube
Box Collider
Edit Collider
Is Trigger
Material None (Physic Material)
Center X 0 Y 0 Z 0
Size X 1 Y 1 Z 1
Mesh Renderer
Cast Shadows On
Receive Shadows
Motion Vectors
Materials
Light Probes Blend Probes
Reflection Probes Blend Probes
Anchor Override None (Transform)
Default-Material
Shader Standard
Add Component
Project Console
Create
Favorites
All Materials
All Models
All Prefabs
All Scripts
Assets
Assets
This folder is empty

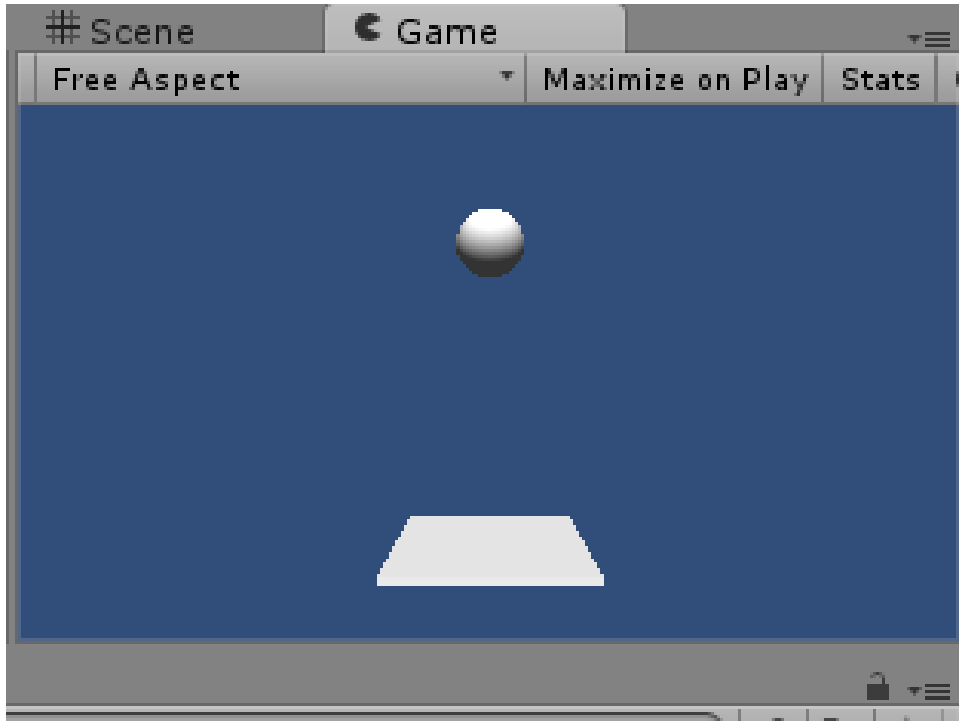
Lab 3. A Bouncy Ball

Exercise : **Make a ball bounce against a paddle**

- Create a ball (sphere) and a paddle (cube),
-Make the ball bounce back repeatedly. (add ‘rigidbody’ component (for falling) to the ball, create a physical material called ‘BouncyBall’ (for bouncing) and add it to the collider comp of the ball)

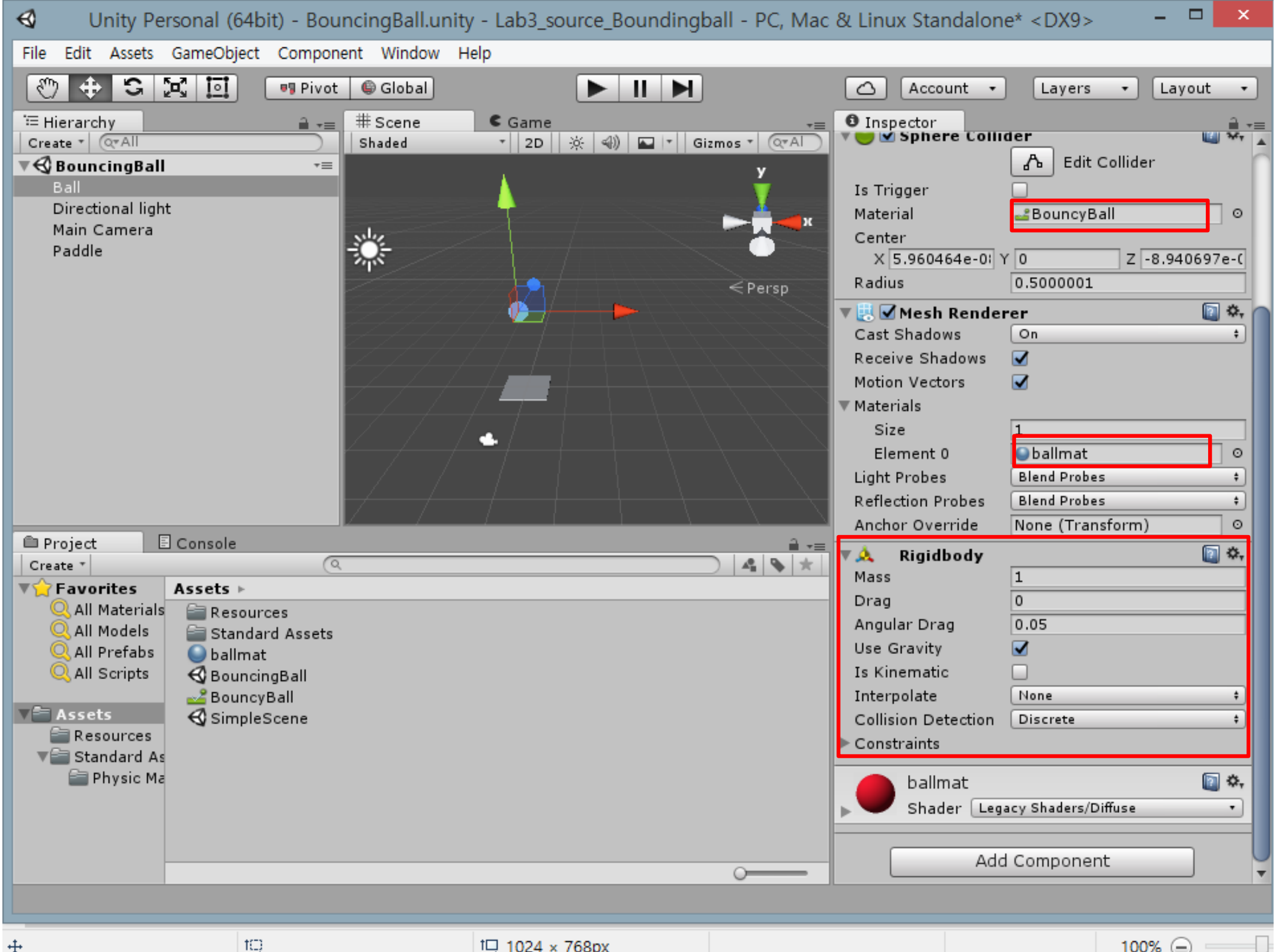
(For details, refer to “reference-book-chap03.pptx”)

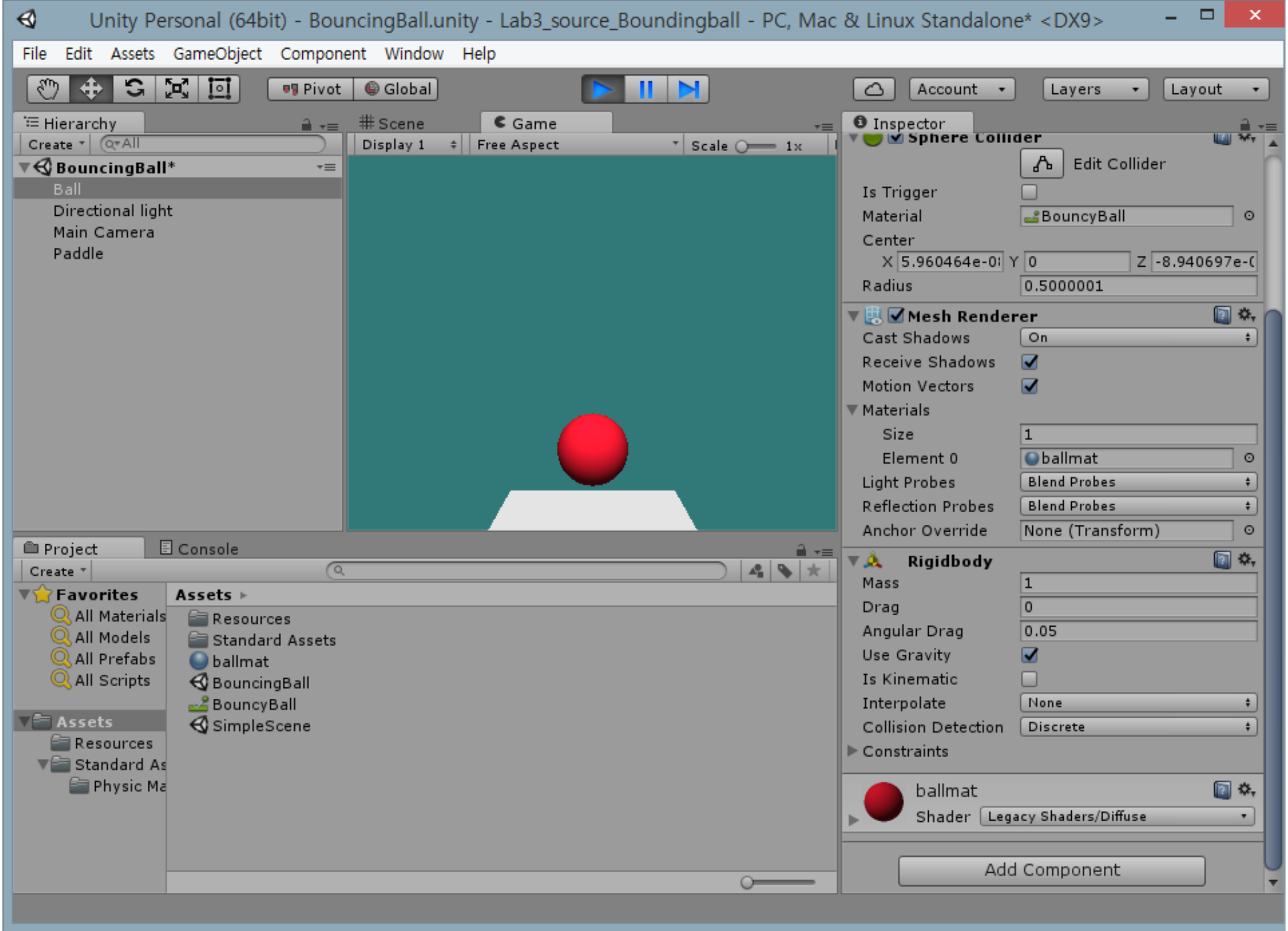
See Lab3_source_Boundingball project.



Exercise : **Add materials** to the ball and paddle.

- Make a material called ‘ballmaterial’, change the color or texture of the material, add it to the renderer of the ball.
- Do the same to the paddle.


Unity Personal (64bit) - BouncingBall.unity - Lab3_source_Boundingball - PC, Mac & Linux Standalone* <DX9>
File Edit Assets GameObject Component Window Help
Pivot Global
Account Layers Layout
Hierarchy
Create All
BouncingBall
Ball
Directional light
Main Camera
Paddle
Scene Game
Shaded 2D Gizmos All
Persp
Inspector
Sphere Collider
Edit Collider
Is Trigger
Material BouncyBall
Center
X 5.960464e-0 Y 0 Z -8.940697e-(
Radius 0.5000001
Mesh Renderer
Cast Shadows On
Receive Shadows
Motion Vectors
Materials
Size 1
Element 0 ballmat
Light Probes Blend Probes
Reflection Probes Blend Probes
Anchor Override None (Transform)
Rigidbody
Mass 1
Drag 0
Angular Drag 0.05
Use Gravity
Is Kinematic
Interpolate None
Collision Detection Discrete
Constraints
ballmat
Shader Legacy Shaders/Diffuse
Add Component
Project Console
Create
Favorites
All Materials
All Models
All Prefabs
All Scripts
Assets
Resources
Standard Assets
ballmat
BouncingBall
BouncyBall
SimpleScene
Assets
Resources
Standard As
Physic Ma
1024 x 768px
100%


Unity Personal (64bit) - BouncingBall.unity - Lab3_source_Boundingball - PC, Mac & Linux Standalone* <DX9>
File Edit Assets GameObject Component Window Help
Pivot Global
Account Layers Layout
Hierarchy
Create All
BouncingBall*
Ball
Directional light
Main Camera
Paddle
Scene Game
Display 1 Free Aspect Scale 1x
Inspector
Sphere Collider
Edit Collider
Is Trigger
Material BouncyBall
Center
X 5.960464e-0 Y 0 Z -8.940697e-(
Radius 0.5000001
Mesh Renderer
Cast Shadows On
Receive Shadows
Motion Vectors
Materials
Size 1
Element 0 ballmat
Light Probes Blend Probes
Reflection Probes Blend Probes
Anchor Override None (Transform)
Rigidbody
Mass 1
Drag 0
Angular Drag 0.05
Use Gravity
Is Kinematic
Interpolate None
Collision Detection Discrete
Constraints
ballmat
Shader Legacy Shaders/Diffuse
Add Component
Project Console
Create
Favorites
All Materials
All Models
All Prefabs
All Scripts
Assets
Resources
Standard Assets
Physic Ma
Assets
Resources
Standard Assets
ballmat
BouncingBall
BouncyBall
SimpleScene

Lab 4. Add C# scripts to count bouncing collisions

Exercise: Add C# script to count the bouncing collision

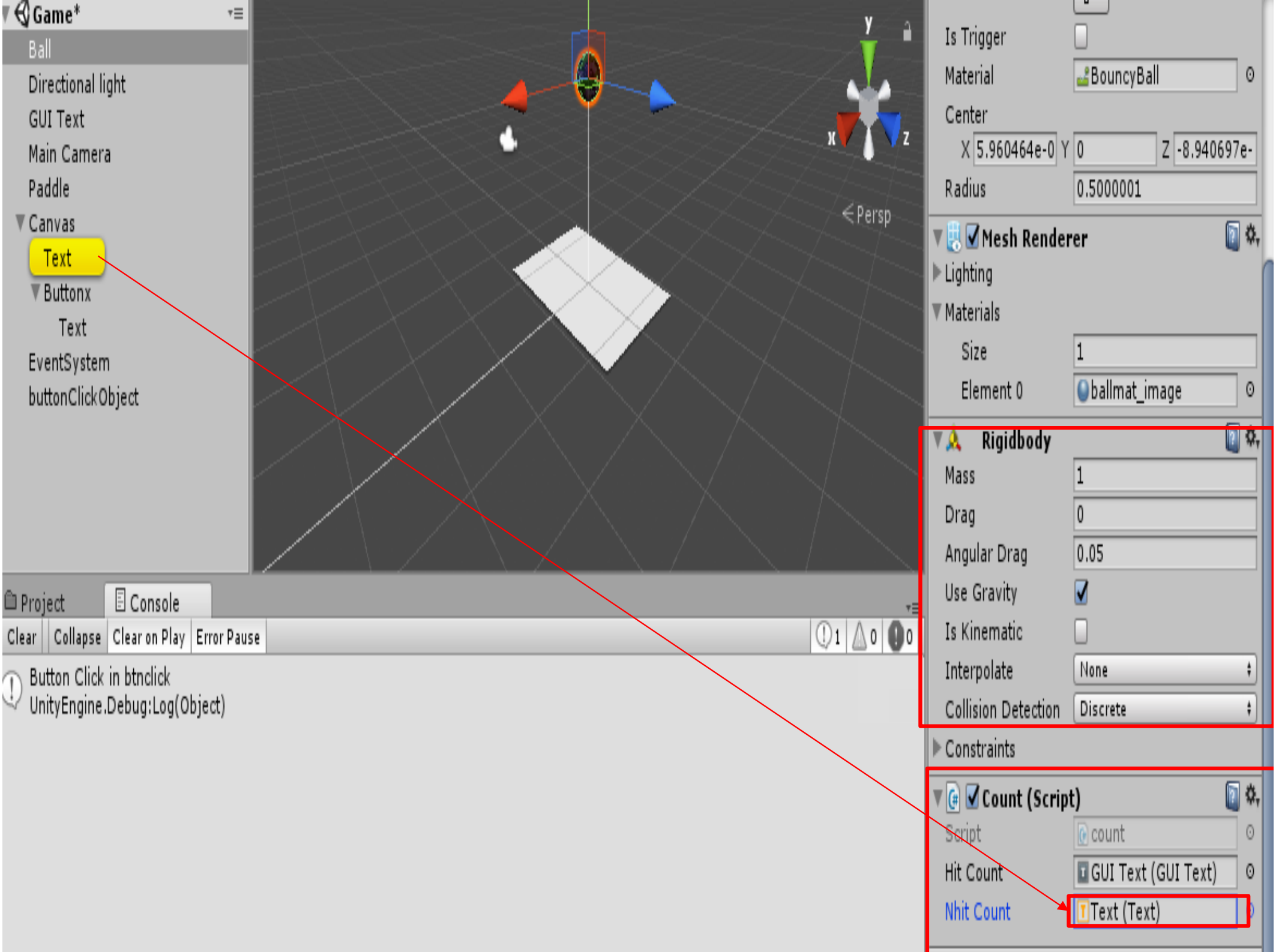
- Create C# script(named “count.cs”), and add it to the ball
- Create C# script(named “move.cs”), and add it to the paddle
- Create C# script(named “btnclick.cs”), and add it to an empty game object (named “buttonClickObject”)

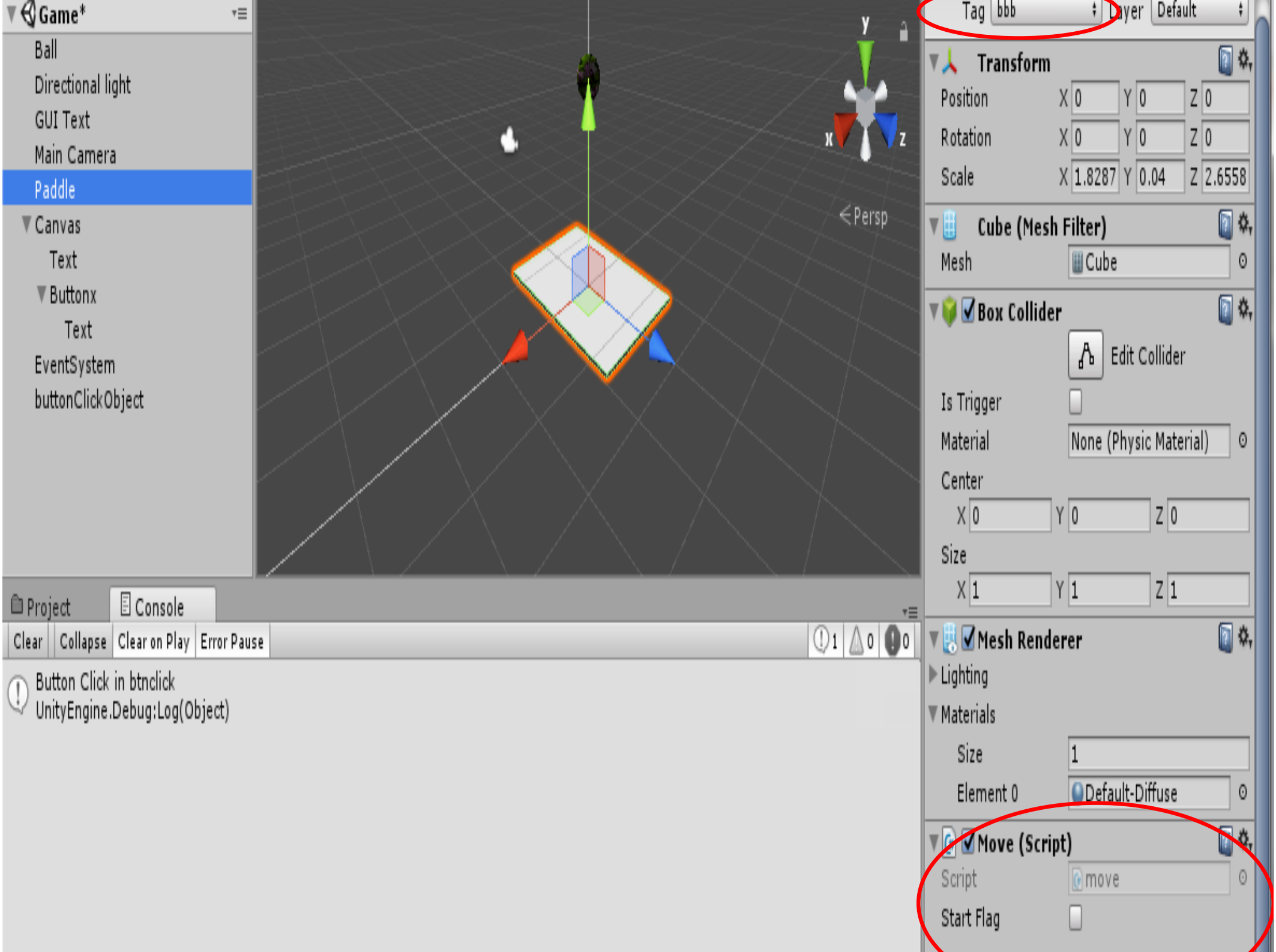
- Create a button UI object.
- Create an Onclick event and link to the empty game object buttonClickObject.

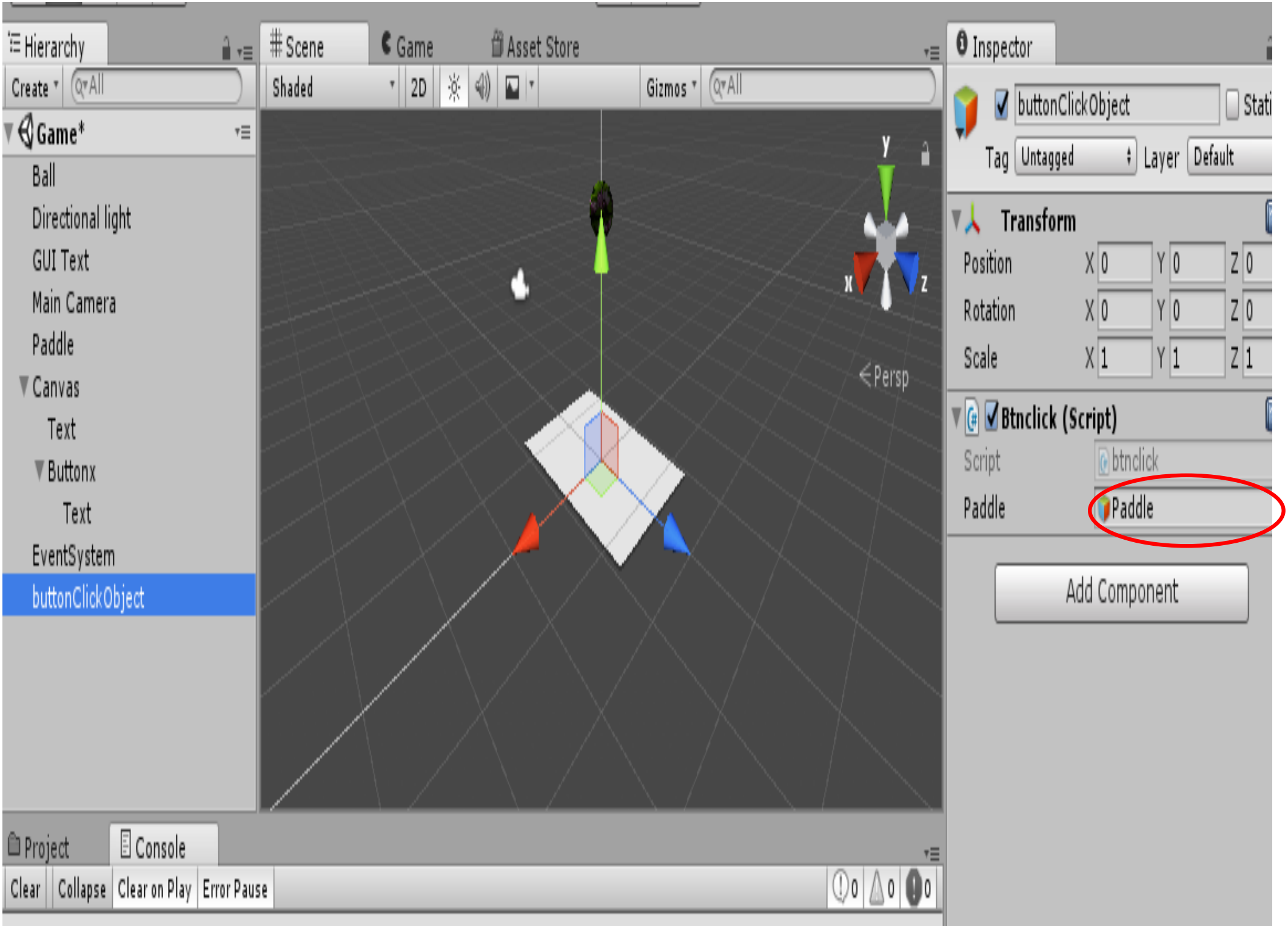
- Add tag “bbb” to the paddle game object in the inspector window.
- In count.cs,

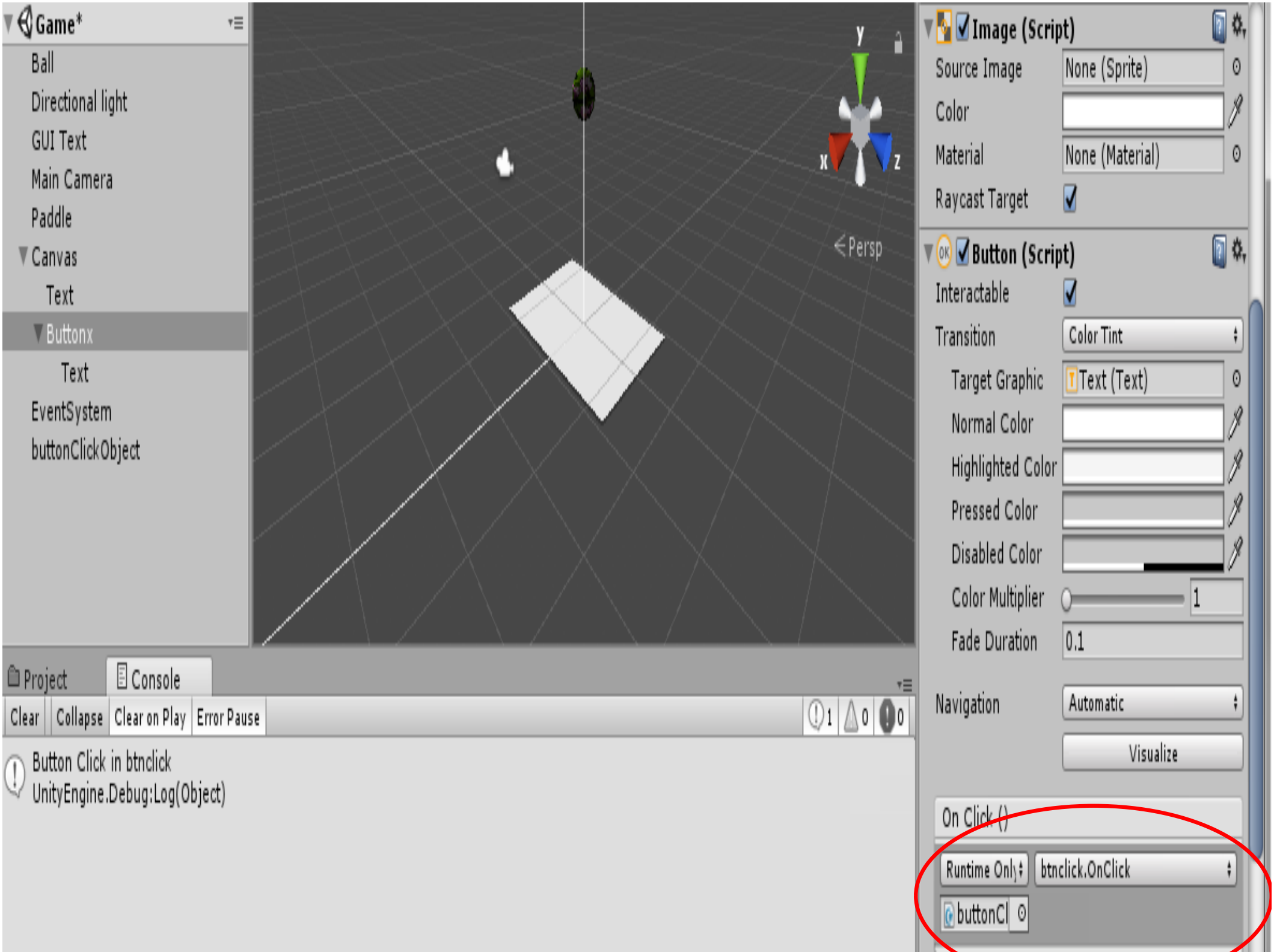
```
void  OnCollisionEnter(Collision col) {
 if(col.gameObject.tag == "bbb") {
    .... }
 void Update () {
    nhitCount.text = numHits.ToString ();
... }
```

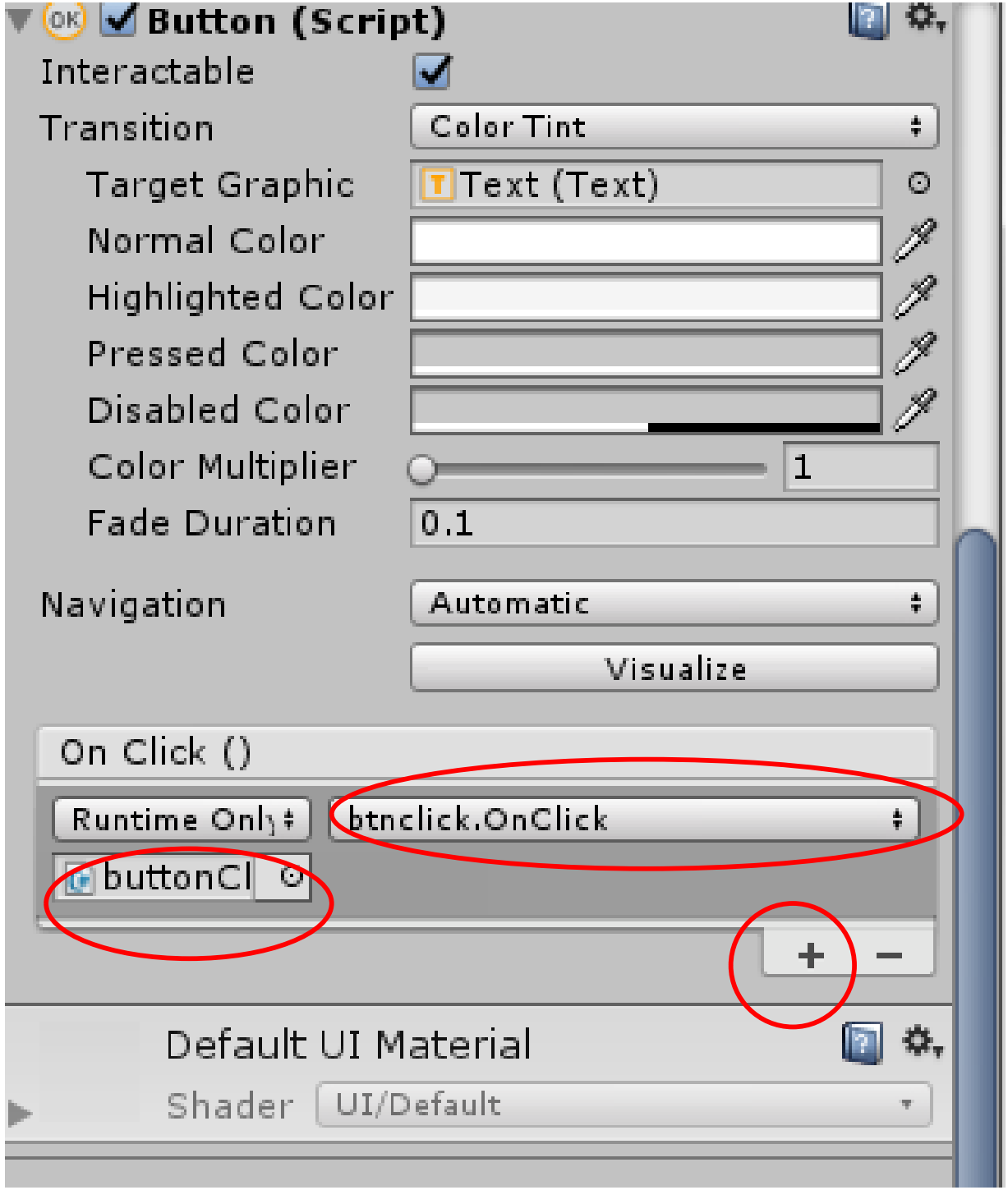
- See Lab4_Count_bouncing_ball project


Game*
Ball
Directional light
GUI Text
Main Camera
Paddle
Canvas
Text
Buttonx
Text
EventSystem
buttonClickObject
Persp
Is Trigger
Material
BouncyBall
Center
X 5.960464e-0
Y 0
Z -8.940697e-
Radius
0.5000001
Mesh Renderer
Lighting
Materials
Size
1
Element 0
ballmat_image
Rigidbody
Mass
1
Drag
0
Angular Drag
0.05
Use Gravity
Is Kinematic
Interpolate
None
Collision Detection
Discrete
Constraints
Count (Script)
Script
count
Hit Count
GUI Text (GUI Text)
Nhit Count
Text (Text)
Project
Console
Clear
Collapse
Clear on Play
Error Pause
Button Click in btnclick
UnityEngine.Debug:Log(Object)


Game*
Ball
Directional light
GUI Text
Main Camera
Paddle
Canvas
Text
Buttonx
Text
EventSystem
buttonClickObject
Persp
Tag bbb
Layer Default
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1.8287 Y 0.04 Z 2.6558
Cube (Mesh Filter)
Mesh Cube
Box Collider
Edit Collider
Is Trigger
Material None (Physic Material)
Center
X 0 Y 0 Z 0
Size
X 1 Y 1 Z 1
Mesh Renderer
Lighting
Materials
Size 1
Element 0 Default-Diffuse
Move (Script)
Script move
Start Flag
Project
Console
Clear
Collapse
Clear on Play
Error Pause
1
0
0
Button Click in btnclick
UnityEngine.Debug:Log(Object)


Hierarchy
Create
All
Game*
Ball
Directional light
GUI Text
Main Camera
Paddle
Canvas
Text
Buttonx
Text
EventSystem
buttonClickObject
Scene
Game
Asset Store
Shaded
2D
Gizmos
All
y
x
z
Persp
Inspector
buttonClickObject
Stati
Tag
Untagged
Layer
Default
Transform
Position
Rotation
Scale
X 0
Y 0
Z 0
X 1
Y 1
Z 1
Btnclick (Script)
Script
btnclick
Paddle
Paddle
Add Component
Project
Console
Clear
Collapse
Clear on Play
Error Pause
0
0
0


Game*
Ball
Directional light
GUI Text
Main Camera
Paddle
Canvas
Text
Buttonx
Text
EventSystem
buttonClickObject
y
x
z
Persp
Image (Script)
Source Image
None (Sprite)
Color
Material
None (Material)
Raycast Target
Button (Script)
Interactable
Transition
Color Tint
Target Graphic
Text (Text)
Normal Color
Highlighted Color
Pressed Color
Disabled Color
Color Multiplier
1
Fade Duration
0.1
Navigation
Automatic
Visualize
On Click ()
Runtime Only
btnclick.OnClick
buttonCl
Project
Console
Clear
Collapse
Clear on Play
Error Pause
1
0
0
Button Click in btnclick
UnityEngine.Debug:Log(Object)


Button (Script)
Interactable
Transition
Color Tint
Target Graphic
Text (Text)
Normal Color
Highlighted Color
Pressed Color
Disabled Color
Color Multiplier
1
Fade Duration
0.1
Navigation
Automatic
Visualize
On Click ()
Runtime Only
btnclick.OnClick
buttonCl
+
−
Default UI Material
Shader
UI/Default


Rect Transform
center
middle
Pos X
Pos Y
Pos Z
-109
-308
0
Width
Height
160
30
R
Anchors
Min
X 0.5
Y 0.5
Max
X 0.5
Y 0.5
Pivot
X 0.5
Y 0.5
Rotation
X 0
Y 0
Z 0
Scale
X 1
Y 1
Z 1

프로퍼티:	역할:
(Anchors min, max가 어떤 점(기준점)을 정의할 때 사용) **Pos (X, Y, Z)**	Position of the rectangle's pivot point relative to the anchors.
Width/Height **----------**	Width and height of the rectangle. ----------
(Anchors min, max가 area를 정의할 때 사용) **Left, Top, Right, Bottom**	Positions of the rectangle's edges relative to their anchors. This can be thought of as padding inside the rectangle defined by the anchors. Shown in place of *Pos* and *Width/Height* when the anchors are separated (뒷장 참고)

Anchors	The anchor points for the lower left corner and the upper right corner of the area(rectangle) on the canvas. 단, min과 max가 같으면, 기준점을 정의하게 됨.
Min	The anchor point for the lower left corner of the rectangle defined as a fraction of the size of the parent rectangle. x=0,y=0 corresponds to anchoring to the lower left corner of the parent, while x=1,y=1 corresponds to anchoring to the upper right corner of the parent.
Max	The anchor point for the upper right corner of the rectangle defined as a fraction of the size of the parent rectangle. x=0,y=0 corresponds to anchoring to the lower left corner of the parent, while x=1,y=1 corresponds to anchoring to the upper right corner of the parent.

Pivot **(of UI object)** **(anchors min, max가 기준점을 정의할 때 사용됨)**	Location of the pivot point around which the rectangle rotates, defined as a fraction of the size of the rectangle itself. x=0,y=0 corresponds to the lower left corner while x=1,y=1 corresponds to the upper right corner.

count.cs (attached to Ball)

```
using UnityEngine;
using UnityEngine.UI;
using System.Collections;

public class count : MonoBehaviour {
        public GUIText hitCount; //old style
        public Text nhitCount;
        int numHits;
        float x, y;
```

```
// Use this for initialization
        void Start () {
                numHits = 0;
        }

        void OnCollisionEnter(Collision col) {
            if (col.gameObject.tag == "bbb") {
                        numHits++;
            }

        }
```

```
// Update is called once per frame
        void Update () {
          nhitCount.text = numHits.ToString ();
          //x = Input.mousePosition.x;
          //y = Input.mousePosition.y;



        }
}
```

move.cs (attached to Paddle)

```
using UnityEngine;
using System.Collections;
public class move : MonoBehaviour {
	float x, y;
	float smooth = (float)10;
	//float tiltAngle = (float)30.0;
	float rot = (float)0;
	public bool startFlag = false;

	GameObject hand; // =
GameObject.Find("Ball");
```

```
        /* void OnGUI() {
            if (GUI.Button(new Rect(10, 10, 50,
50), "Start"))
                //print("You clicked the button!");
                startFlag = !startFlag; //true;
        } */

    public void onClick()
    {
        startFlag = !startFlag; //true;
        Debug.Log("Button Click");
        //Application.Quit();
    }
```

```
void Start () {
        hand = GameObject.Find("Ball");
}
// Update is called once per frame
void Update () {
        //print("x:" + Input.mousePosition.x);
        //print("y:" + Input.mousePosition.y);
        if (!startFlag) {
                return;
        }
```

//paddle을 mouse 움직에 따라 움직임.

float halfW = Screen.width / 2; // store half of the screen width as a value in a variable called halfW

float halfH = Screen.height / 3;

```
            Vector3 tempvect = new
Vector3((Input.mousePosition.x - halfW) / halfW,
gameObject.transform.position.y,
(Input.mousePosition.y - halfH) / halfH);
                gameObject.transform.position =
tempvect;
            //ball을 회전
            rot += 5;
            var target = Quaternion.Euler (rot, 0, 0);
            hand.transform.rotation = target;
        } //Update()
}
```

btnclick.cs (attached to buttonClickObject)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class btnclick : MonoBehaviour {
    public GameObject paddle;
        // Use this for initialization
        void Start () {     }

        // Update is called once per frame
        void Update () {
        }
```

```
 public void OnClick()
    {
        Debug.Log("Button Click in
btnclick");
        move SN =
paddle.GetComponent<move>();
        SN.startFlag = !SN.startFlag;
    }
}
```

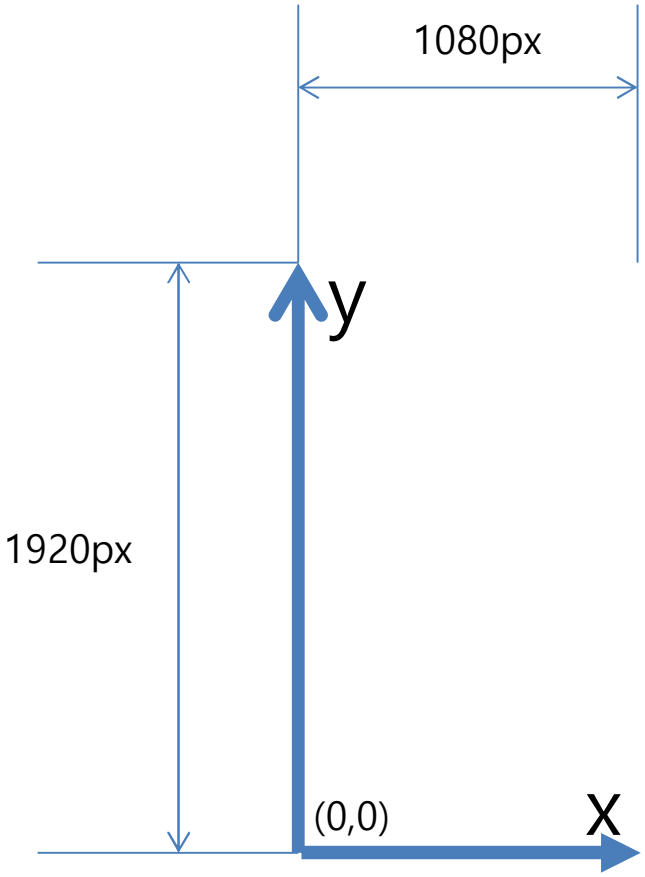
public - Show up in inspector and accessible by other scripts

[SerializeField] private - Show up in inspector, not accessible by other scripts

[HideInInspector] public - Doesn't show in inspector, accessible by other scripts

private - Doesn't show in inspector, not accessible by other scripts

1080x1920 의 경우



Screen 좌표계

Full-HD Resolution

Exercise: Add another ball

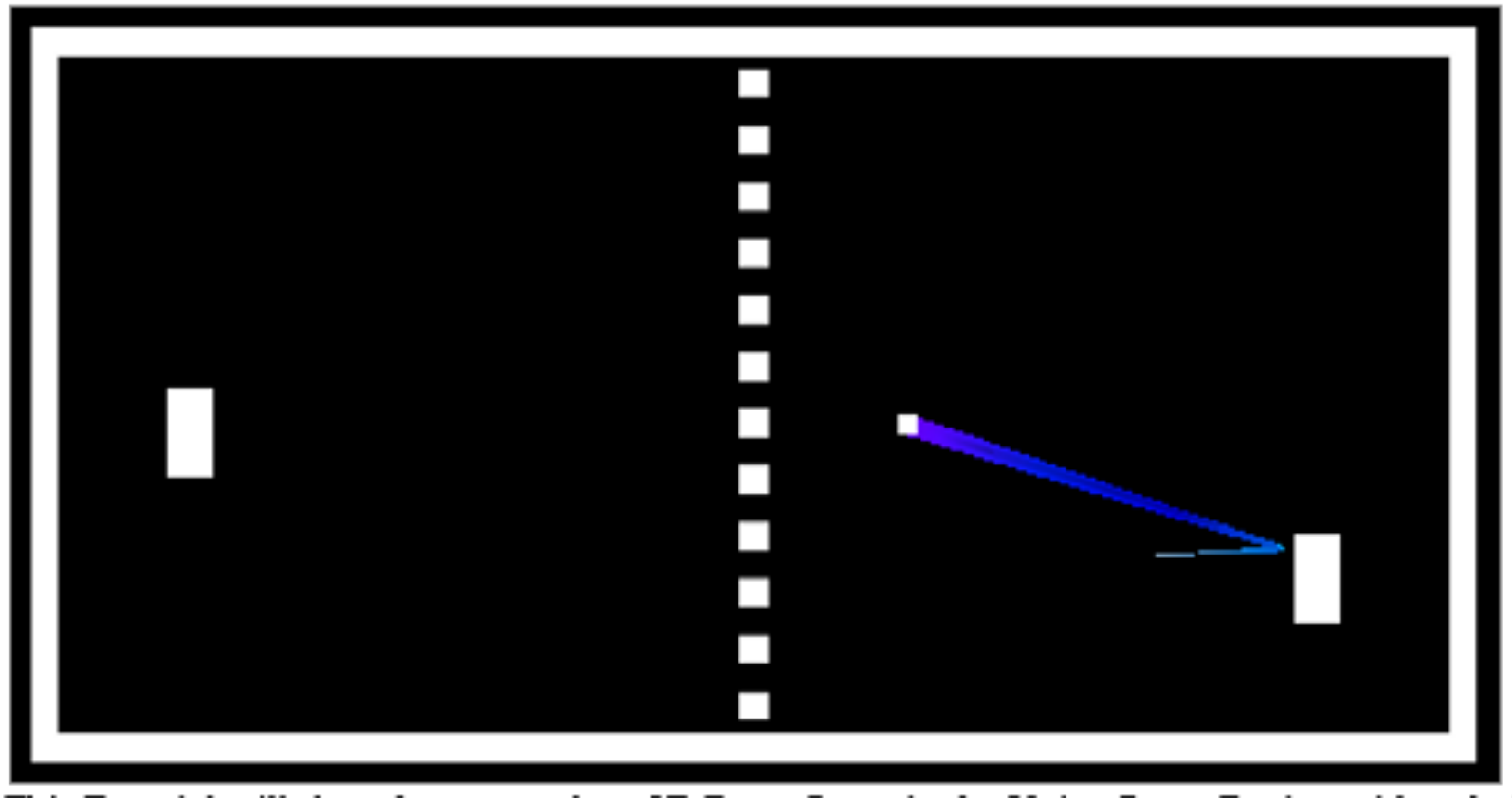
- Exercise: Modify Lab5_Count_bouncing_ball project as follows.
  - Add another ball, and make it bounce too.
  - Increment the count by 10 at every collision

Exercise: Handle mouse inputs

- Exercise: Modify Lab5_Count_bouncing_ball project as follows.
  - Make the initial heights of the two balls different. and spread them apart.
  - Move the paddle to the left or right in order to make it collide with by the ball.

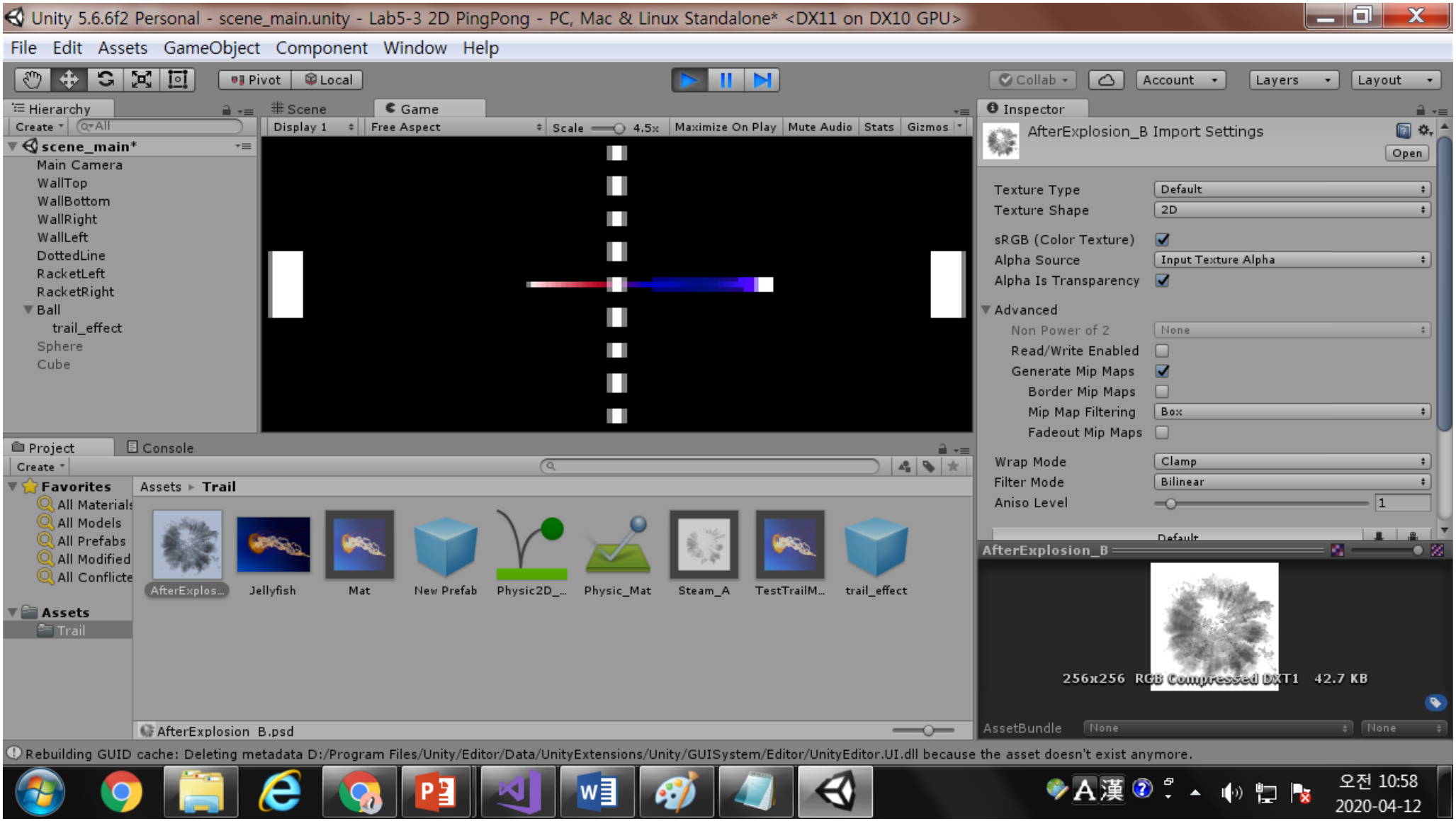
```
float x = Input.mousePosition.x;
float y = Input.mousePosition.y;
```

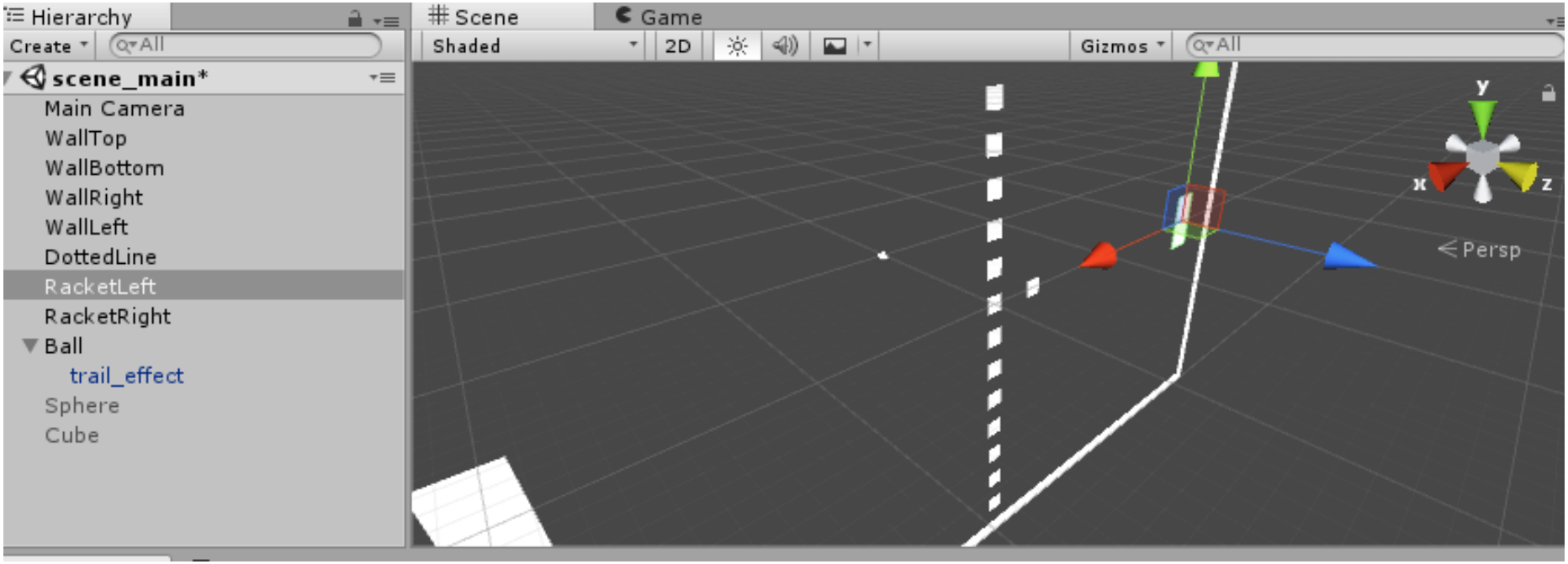
Lab5 A Simple PingPong Game



PingPong

- 2개의 paddle을 생성
- Ball을 상대방 paddle 로 쳐냄
- Paddle은 좌우로 움직일 수 있고, 각도를 조절해 상대방 측으로 공을 날려 보냄. 각도가 가파르면 고도가 낮고 길이는 짧게 비행하고, 각도가 완만하면 고도가 높고 길이는 길게 비행한다.


Unity 5.6.6f2 Personal - scene_main.unity - Lab5-3 2D PingPong - PC, Mac & Linux Standalone* <DX11 on DX10 GPU>
File Edit Assets GameObject Component Window Help
Pivot Local
Collab Account Layers Layout
Hierarchy
Create All
scene_main*
Main Camera
WallTop
WallBottom
WallRight
WallLeft
DottedLine
RacketLeft
RacketRight
Ball
trail_effect
Sphere
Cube
Scene Game
Display 1 Free Aspect Scale 4.5x Maximize On Play Mute Audio Stats Gizmos
Inspector
AfterExplosion_B Import Settings
Open
Texture Type Default
Texture Shape 2D
sRGB (Color Texture)
Alpha Source Input Texture Alpha
Alpha Is Transparency
Advanced
Non Power of 2 None
Read/Write Enabled
Generate Mip Maps
Border Mip Maps
Mip Map Filtering Box
Fadeout Mip Maps
Wrap Mode Clamp
Filter Mode Bilinear
Aniso Level 1
Default
Project Console
Create
Favorites
All Materials
All Models
All Prefabs
All Modified
All Conflicte
Assets
Trail
Assets ▸ Trail
AfterExplos...
Jellyfish
Mat
New Prefab
Physic2D_...
Physic_Mat
Steam_A
TestTrailM...
trail_effect
AfterExplosion_B.psd
AfterExplosion_B
256x256 RGB Compressed DXT1 42.7 KB
AssetBundle None None
Rebuilding GUID cache: Deleting metadata D:/Program Files/Unity/Editor/Data/UnityExtensions/Unity/GUISystem/Editor/UnityEditor.UI.dll because the asset doesn't exist anymore.
오전 10:58
2020-04-12



RacketLeft

RacketRight

Ball

```
//MoveRacket.cs (attached to left & right rackets)
using UnityEngine;
using System.Collections;
public class MoveRacket : MonoBehaviour {
    public float speed = 30;
    public string axis = "Vertical";
    void FixedUpdate () {
        float v = Input.GetAxisRaw(axis);
        GetComponent<Rigidbody2D>().velocity =
new Vector2(0, v) * speed;
    }
}
```

```
//Ball.cs (attached to Ball)
using UnityEngine;
using System.Collections;

public class Ball : MonoBehaviour {
    public float speed = 30;
    void Start() {
        // Initial Velocity
        GetComponent<Rigidbody2D>().velocity = Vector2.right
* speed;
    }

    float hitFactor(Vector2 ballPos, Vector2 racketPos,
                    float racketHeight) {
        // ascii art:
        // ||  1 <- at the top of the racket
        // ||  0 <- at the middle of the racket
         // || -1 <- at the bottom of the racket
        return (ballPos.y - racketPos.y) / racketHeight;
    }
```

```
 void OnCollisionEnter2D(Collision2D col) {
        // Hit the left Racket?
        if (col.gameObject.name == "RacketLeft") {
            // Calculate hit Factor
            float y = hitFactor(transform.position,
                                col.transform.position,
                                col.collider.bounds.size.y);

            // Calculate direction, make length=1
via .normalized
            Vector2 dir = new Vector2(1, y).normalized;

            // Set Velocity with dir * speed
            GetComponent<Rigidbody2D>().velocity = dir
* speed;
        }
```

```
        //Hit the right Racket?
        if (col.gameObject.name == "RacketRight") {
            // Calculate hit Factor
            float y = hitFactor(transform.position,
                                col.transform.position,
                                col.collider.bounds.size.y);
            // Calculate direction
            Vector2 dir = new Vector2(-1, y).normalized;
            // Set Velocity with dir * speed
            GetComponent<Rigidbody2D>().velocity = dir *
speed;
        }
    }
}
```

Input Key

```
 if (Input.GetKeyDown(KeyCode.Space)) {
        Debug.Log("Space key was pressed.");
    }
    if (Input.GetKeyUp(KeyCode.Space)) {
    Debug.Log("Space key was released.");
    }
  }
}
```

Application.targetFrameRate (초당 프레임 수)

Update() FixedUpdate

FixedUpdate() : this function is called every **fixed frame rate**. (See Edit->Project Setting->Time)

- FixedUpdate should be used instead of Update when dealing with physics such as rigidbody.

Edit->Project Setting->Time

Fixed Timestep	물리적 계산이 나 *FixedUpdate()* events를 수행할 때 지시하는 프레임속도와 무관한 (framerate-independent) 구간.
Maximum Allowed Timestep	물리적 계산이 *FixedUpdate()* events는 이 값을 넘기지 않는 시간 내에서만 수행.
Time Scale	시간이 흐르는 속도. 이 값을 수정하여 총알 시간 효과(bullet-time effects)를 재현 할 수 있습니다. 1은 실시간을 의미하고, 0.5는 그 반의 빠르기, 2는 두 배의 빠르기를 의미.

Fixed Timestep: 모든 프레임이 스크린에 그려지기 전에, 유니티가 고정시간을 고정델타시간으로 당기고 그것이 현재시간에 다다를 때까지 연산(충돌 감지나 리지드바디 동작 같은 물리력 계산)을 수행. 타이머로 동작

Exercise: 3D Model Import

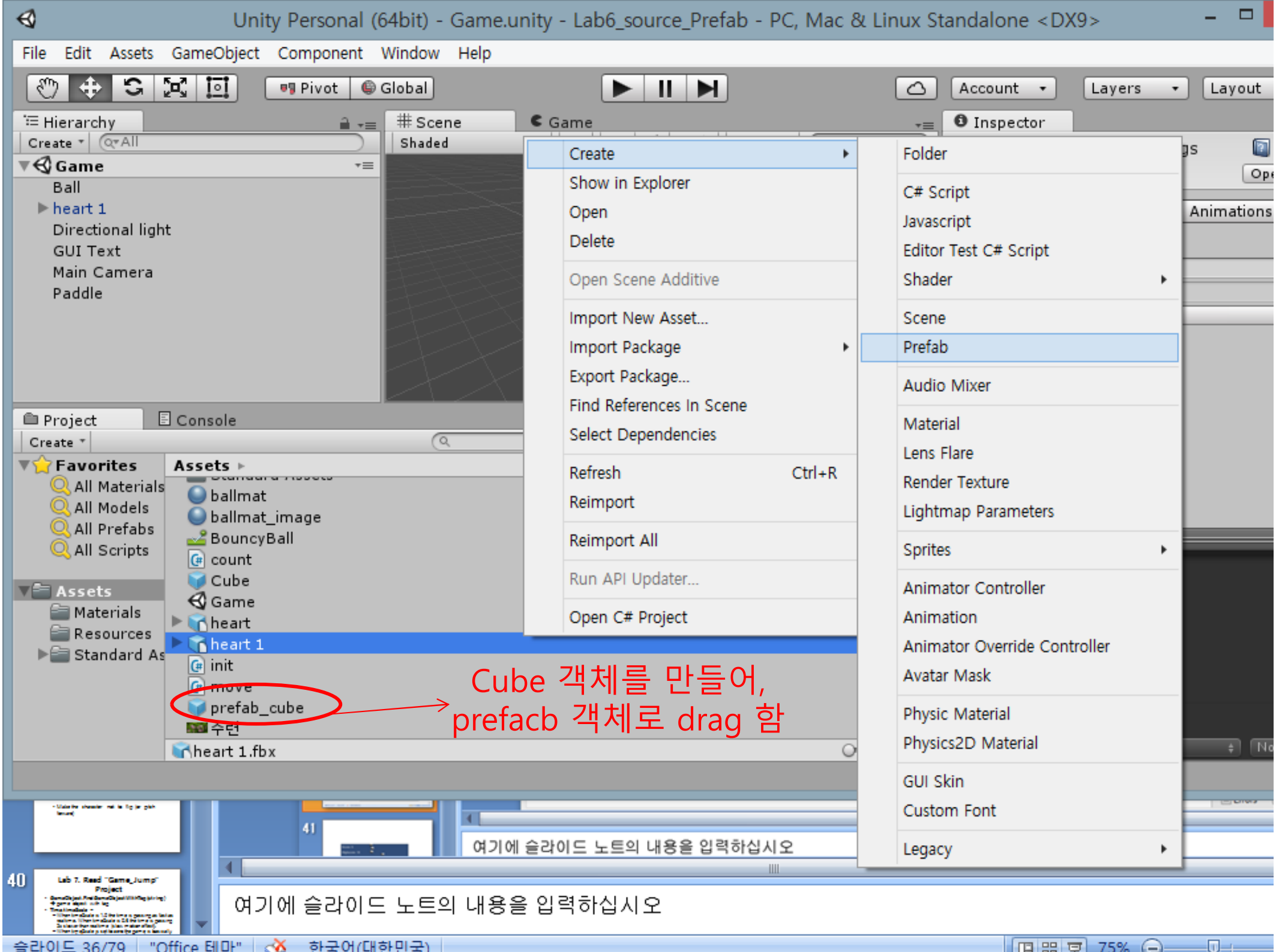
3D 앱에서 모델을 임포트 가능

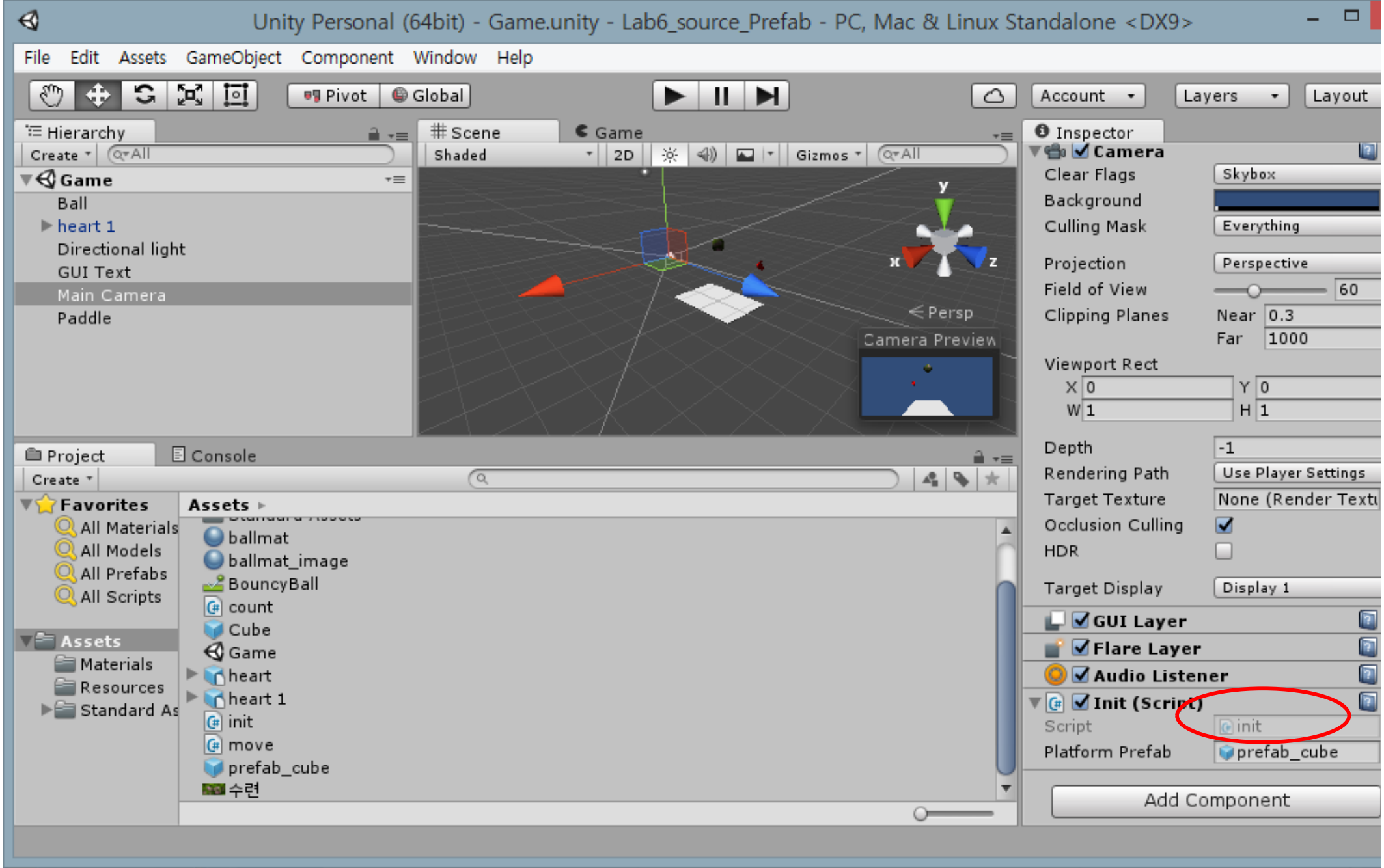
- 3D 모델 파일을 파일 브라우저에서 Unity 프로젝트 창으로 직접 드래그.
- 3D 모델 파일을 프로젝트의 Assets 폴더로 복사.

- .fbx, .dae, .3ds, .dxf, .obj, .skp 등
- **Max**, **Maya**, **Blender**, **Cinema4D**, **Modo**, **Lightwave**, **Cheetah3D** 와 같은 그래픽 소프트웨어에서 전용 파일:

Lab 6 Prefab

- Description:
  - Make a prefab containing a ball.
  - Modify Lab5_Count_bouncing_ball project to use the prefab for creating new balls.

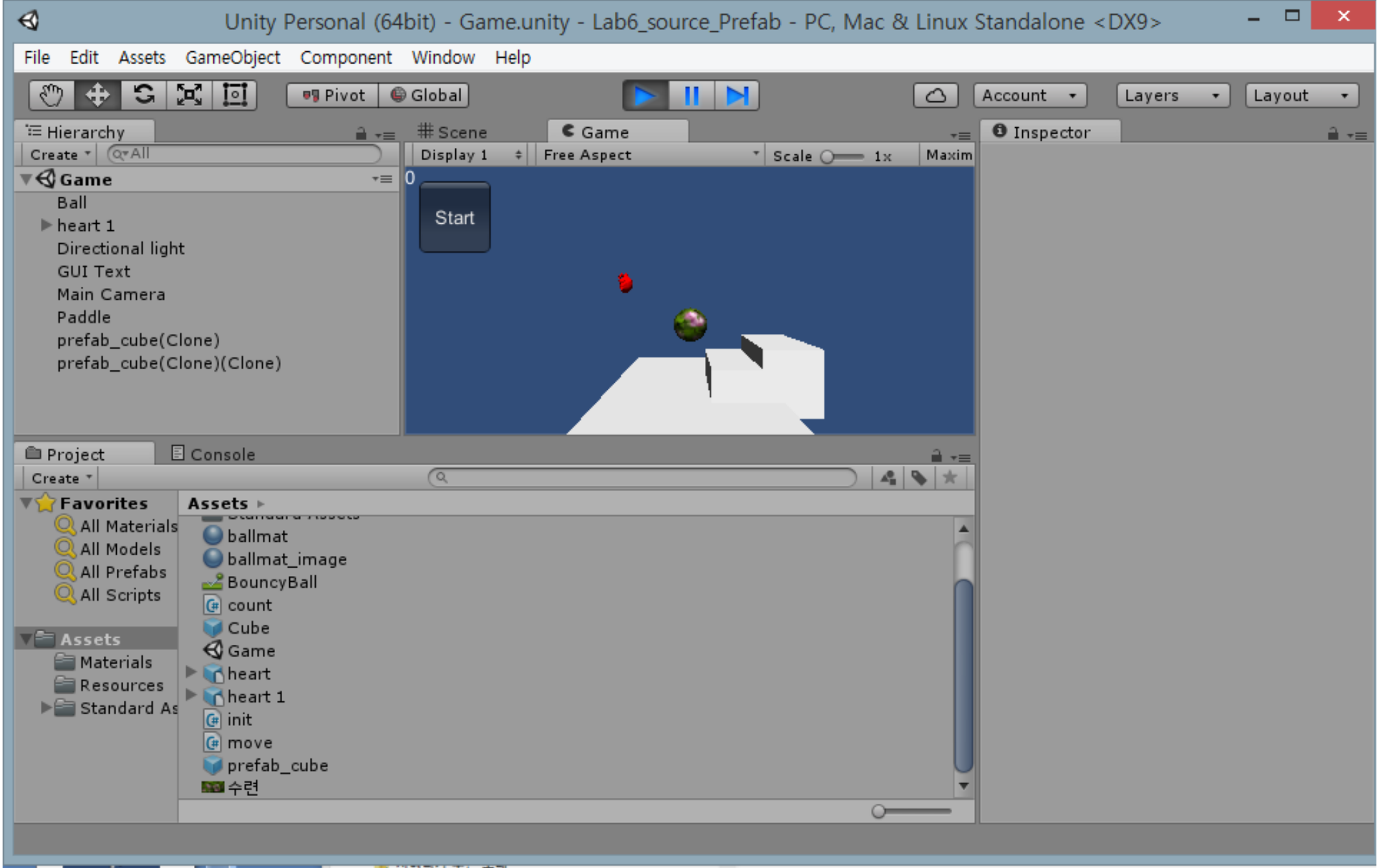

Unity Personal (64bit) - Game.unity - Lab6_source_Prefab - PC, Mac & Linux Standalone <DX9>
File Edit Assets GameObject Component Window Help
Pivot Global
Account Layers Layout
Hierarchy
Create All
Game
Ball
heart 1
Directional light
GUI Text
Main Camera
Paddle
Scene
Shaded
Game
Inspector
Project
Console
Create
Favorites
All Materials
All Models
All Prefabs
All Scripts
Assets
Materials
Resources
Standard As
Assets
ballmat
ballmat_image
BouncyBall
count
Cube
Game
heart
heart 1
init
prefab_cube
heart 1.fbx
Create
Show in Explorer
Open
Delete
Open Scene Additive
Import New Asset...
Import Package
Export Package...
Find References In Scene
Select Dependencies
Refresh Ctrl+R
Reimport
Reimport All
Run API Updater...
Open C# Project
Folder
C# Script
Javascript
Editor Test C# Script
Shader
Scene
Prefab
Audio Mixer
Material
Lens Flare
Render Texture
Lightmap Parameters
Sprites
Animator Controller
Animation
Animator Override Controller
Avatar Mask
Physic Material
Physics2D Material
GUI Skin
Custom Font
Legacy
Cube 객체를 만들어,
prefacb 객체로 drag 함
여기에 슬라이드 노트의 내용을 입력하십시오
40
여기에 슬라이드 노트의 내용을 입력하십시오
슬라이드 36/79
"Office 테마"
한국어(대한민국)
75%

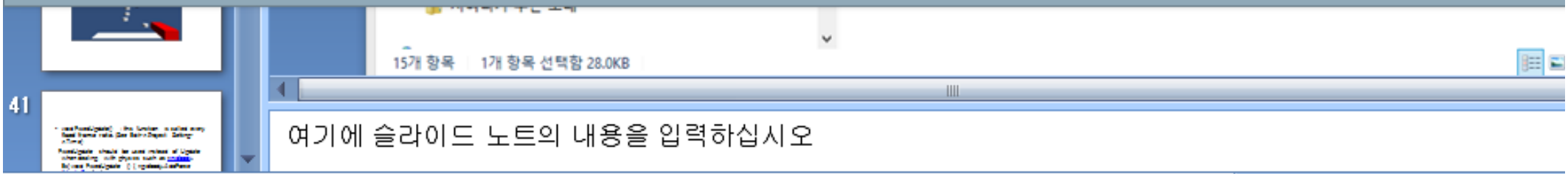

Unity Personal (64bit) - Game.unity - Lab6_source_Prefab - PC, Mac & Linux Standalone <DX9>
File Edit Assets GameObject Component Window Help
Pivot Global
Account Layers Layout
Hierarchy
Create All
Game
Ball
heart 1
Directional light
GUI Text
Main Camera
Paddle
Scene Game
Shaded 2D Gizmos All
Persp
Camera Preview
Inspector
Camera
Clear Flags Skybox
Background
Culling Mask Everything
Projection Perspective
Field of View 60
Clipping Planes Near 0.3
Far 1000
Viewport Rect
X 0 Y 0
W 1 H 1
Depth -1
Rendering Path Use Player Settings
Target Texture None (Render Textu
Occlusion Culling
HDR
Target Display Display 1
GUI Layer
Flare Layer
Audio Listener
Init (Script)
Script init
Platform Prefab prefab_cube
Add Component
Project Console
Create
Favorites
All Materials
All Models
All Prefabs
All Scripts
Assets
Materials
Resources
Standard As
Assets
ballmat
ballmat_image
BouncyBall
count
Cube
Game
heart
heart 1
init
move
prefab_cube
수련
41
여기에 슬라이드 노트의 내용을 입력하십시오
슬라이드 40/78
"Office 테마"
한국어(대한민국)
75%

init.cs

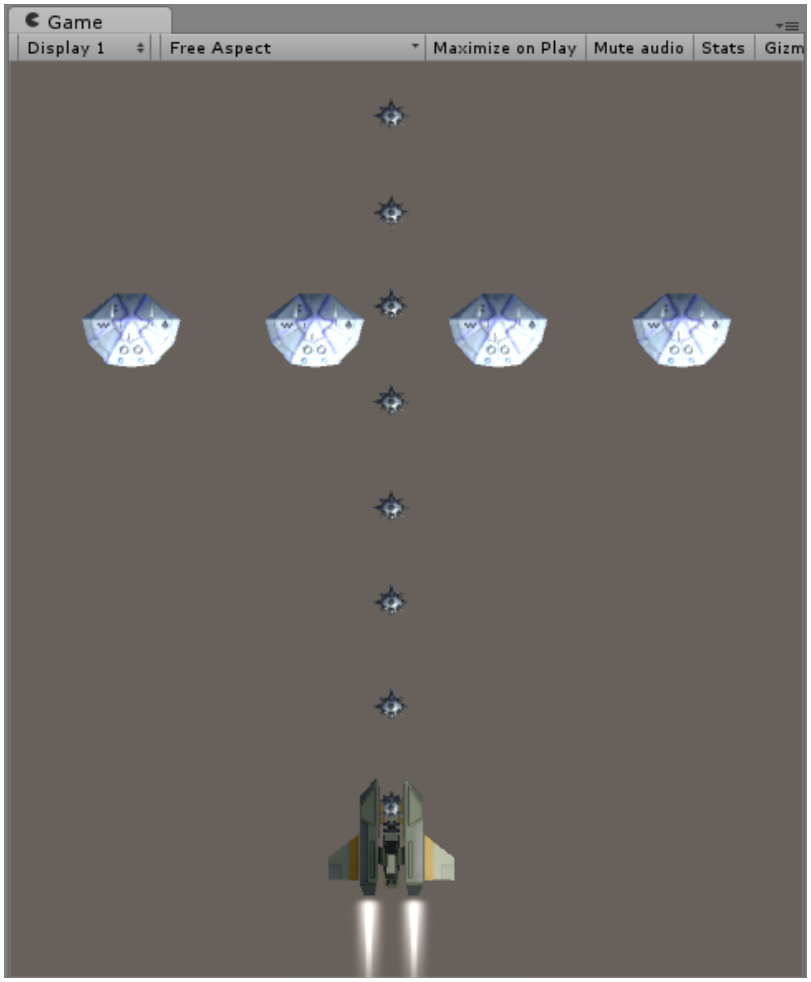
```
using UnityEngine;
using System.Collections;

public class init : MonoBehaviour {
    public GameObject platformPrefab;
    Vector3 pos = new Vector3((float)0.5, (fl
oat)0, (float)0.0);
    Vector3 pos2 = new Vector3((float)1, (fl
oat)0.2, (float)0.0);
    // Use this for initialization
```

```
void Start () {
        platformPrefab = ( GameObject) Instantiate
(platformPrefab, pos, Quaternion.identity);
        //platformPrefab.GetComponent<count>().

        platformPrefab = ( GameObject) Instantiate
(platformPrefab, pos2, Quaternion.identity);
    }

        // Update is called once per frame
        void Update () {
        }
  }
```


Unity Personal (64bit) - Game.unity - Lab6_source_Prefab - PC, Mac & Linux Standalone <DX9>
File Edit Assets GameObject Component Window Help
Pivot Global
Account Layers Layout
Hierarchy
Create All
Game
Ball
heart 1
Directional light
GUI Text
Main Camera
Paddle
prefab_cube(Clone)
prefab_cube(Clone)(Clone)
Scene
Game
Display 1 Free Aspect Scale 1x Maxim
0
Start
Inspector
Project Console
Create
Favorites
All Materials
All Models
All Prefabs
All Scripts
Assets
Materials
Resources
Standard As
Assets
ballmat
ballmat_image
BouncyBall
count
Cube
Game
heart
heart 1
init
move
prefab_cube
수련


15개 항목 1개 항목 선택함 28.0KB
41
여기에 슬라이드 노트의 내용을 입력하십시오
슬라이드 36/77 "Office 테마" 한국어(대한민국) 75%

Example Use of Prefab



Dynamically create bullets

```
public class BulletLauncher : MonoBehaviour {
    public GameObject bullet;
    public float fps = 1;

    // Use this for initialization
    void Start () {
        StartCoroutine(FireBullet());
    }
    IEnumerator FireBullet() {
        while(true) {
            yield return new WaitForSeconds(1 /fps);
            Instantiate(bullet, transform.position, Quaterni
on.identity);
        }
    }
}
```

Lab 7. Learn Various Concepts of C# Script

Exercise: Understand “Lab7_Game_Jump” project and modify it

GameObject.FindGameObjectWithTag(string) ➔ game object with tag

GameObject.FindGameObjectWithTag(string) ➔ game object with tag

Time.timeScale =

When timeScale is 1.0 the time is passing as fast as realtime. When timeScale is 0.5 the time is passing 2x slower than realtime (slow motion effect).

When timeScale is set to zero the game is basically paused if all your functions are frame rate independent.

✓ **Time.deltaTime** Time to last frame

- interval between consecutive frames. the time in seconds it took to complete the last frame (Read Only). Cf) Application.targetFrameRate = 100 or -1(meanining "as fast as possible"

Refer to the following code:

```
void FixedUpdate ()
    {
        //Debug.Log ("Hello", gameObject);
        bool jetpackActive =
    Input.GetButton("Fire1");


        //Rigidbody ball =
    GetComponent<Rigidbody>();
        if (jetpackActive)
        {

    GetComponent<Rigidbody>().AddForce(0, 1, 0,
            ForceMode.Impulse);
        }
    }
```


Score: 6
Highscore: 18

- void FixedUpdate() : this function is called every fixed frame rate. (See Edit->Project Setting->Time)

  FixedUpdate should be used instead of Update when dealing with physics such as rigidbody.
  Ex) void FixedUpdate ()
  { rigidbody.AddForce (Vector3.up); }

자세히 설명하면: Sequence of Update functions

- FixedUpdate(): FixedUpdate 는 종종 Update 보다 더 자주 호출됨. 프레임 속도가 낮은 경우 프레임당 여러 번 호출될 수 있으며 프레임 속도가 높은 경우 프레임 사이에 호출되지 않을 수 있음. 모든 물리 계산 및 업데이트는 FixedUpdate 후 즉시 발생 함. FixedUpdate 는 프레임 속도와 관계없이 신뢰할 수 있는 타이머에서 호출되기 때문임.

- Update(): Update 는 프레임당 한 번 호출됨. 프레임 업데이트를 위한 주요 작업 함수임.
- LateUpdate(): LateUpdate 는 Update 가 끝난 후 프레임당 한 번 호출됨. Update 에서 수행된 모든 계산은 LateUpdate 가 시작할 때 완료됨. LateUpdate 는 일반적으로 다음과 같이 3인칭 카메라에 사용합니다. 캐릭터를 움직이고 Update 로 방향을 바꾸게 하는 경우 LateUpdate 에서 모든 카메라 움직임과 로테이션 계산을 수행할 수 있음.

- Mathf.Lerp(A, B, T) Interpolates between a and b by t. t is clamped between 0 and 1. ex) Mathf.Lerp(0, 3, 0.5) => 1.5

Unity의 prefab 시스템을 이용하면 게임 오브젝트를 생성, 설정 및 저장할 수 있으며, 해당 게임 오브젝트의 모든 컴포넌트, 프로퍼티값, 자식 게임 오브젝트를 재사용 가능한 에셋으로 만들 수 있다.

prefab asset은 scene에 새로운 prefab instance를 만들기위한 템플릿 역할을 한다.

Prefab Instantiation

1) public Transform *platformPrefab*;

   (Transform) Instantiate (*platformPrefab, pos*, Quaternion.identity);

   ➔ instantiate a game object from *platformPrefab* at *pos* with Quaternion.identity rotation

2) GameObject go = Instantiate(Resources.Load("*FooPrefab*")) as GameObject;

PlayerPrefs.GetInit(*"name"*, *default_value*);

PlayerPrefs.SetInit(*"name"*, *value*);

Set player-specific value under the *name*

Application.LoadLevel(Application.loadedLevel) ➔ load the last loaded level again

Application.LoadLevel*(“scene-name”)*

Level name = scene name

Raycast(origin: Vector3, direction: Vector3, distance: float = Mathf.Infinit)

bool True when the ray intersects any collider, otherwise false.

Ex) Physics.Raycast(transform.position, -Vector3.up, 1.0f)

OnTriggerEnter() { … }

- Trigger 는 Collider 콤포넌트의 “Is Trigger” 항목을 체크하면 Collider 가 Trigger 로써 작동
- Collider 로 설정된 오브젝트는 다른 오브젝트와 충돌했을 때 서로 관통하지 못하지만 Trigger 로써 작동하면 Collider 가 설정되지 않은 것과 같은 효과가 있어서 다른 오브젝트가 관통할 수 있고 관통한 오브젝트에 대한 정보도 얻을 수 있음.

- 즉, Trigger 는 충돌탐지는 가능하지만 다른 오브젝트가 지나가는 것을 막지는 않는다는 점에서 Collider와는 기능상 차이를 보임.
- Trigger 는 무언가를 뚫고 지나가야 하는 총알과 같은 성질을 지닌 오브젝트를 구현할 때 유용하게 사용.

Exercise: Modify Lab7_Game_Jump project

Lab7_Game_Jump project in two ways:

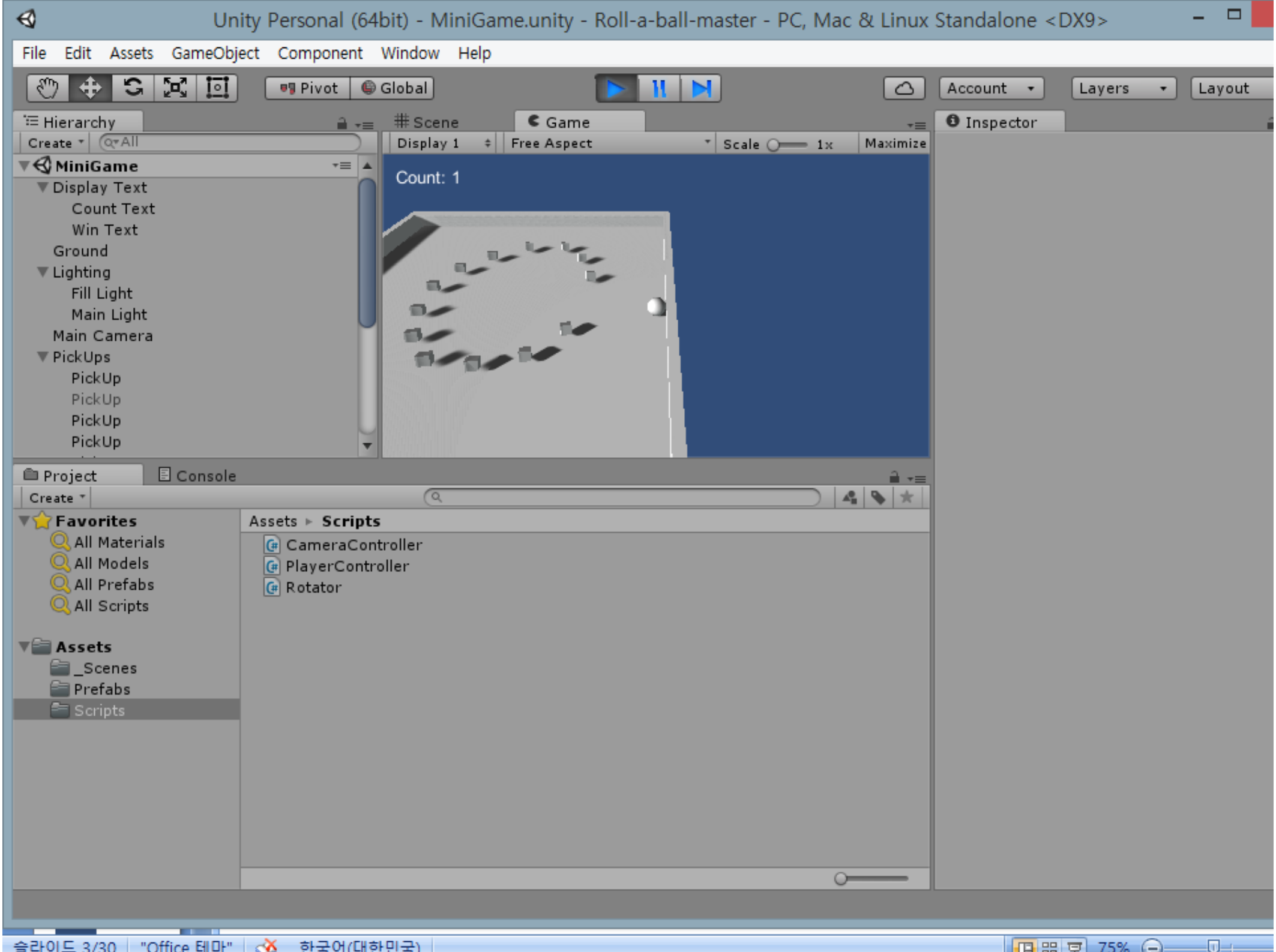
- Replace the cylinder by a box as a platform
- Make the character not to flip (or pitch forward)

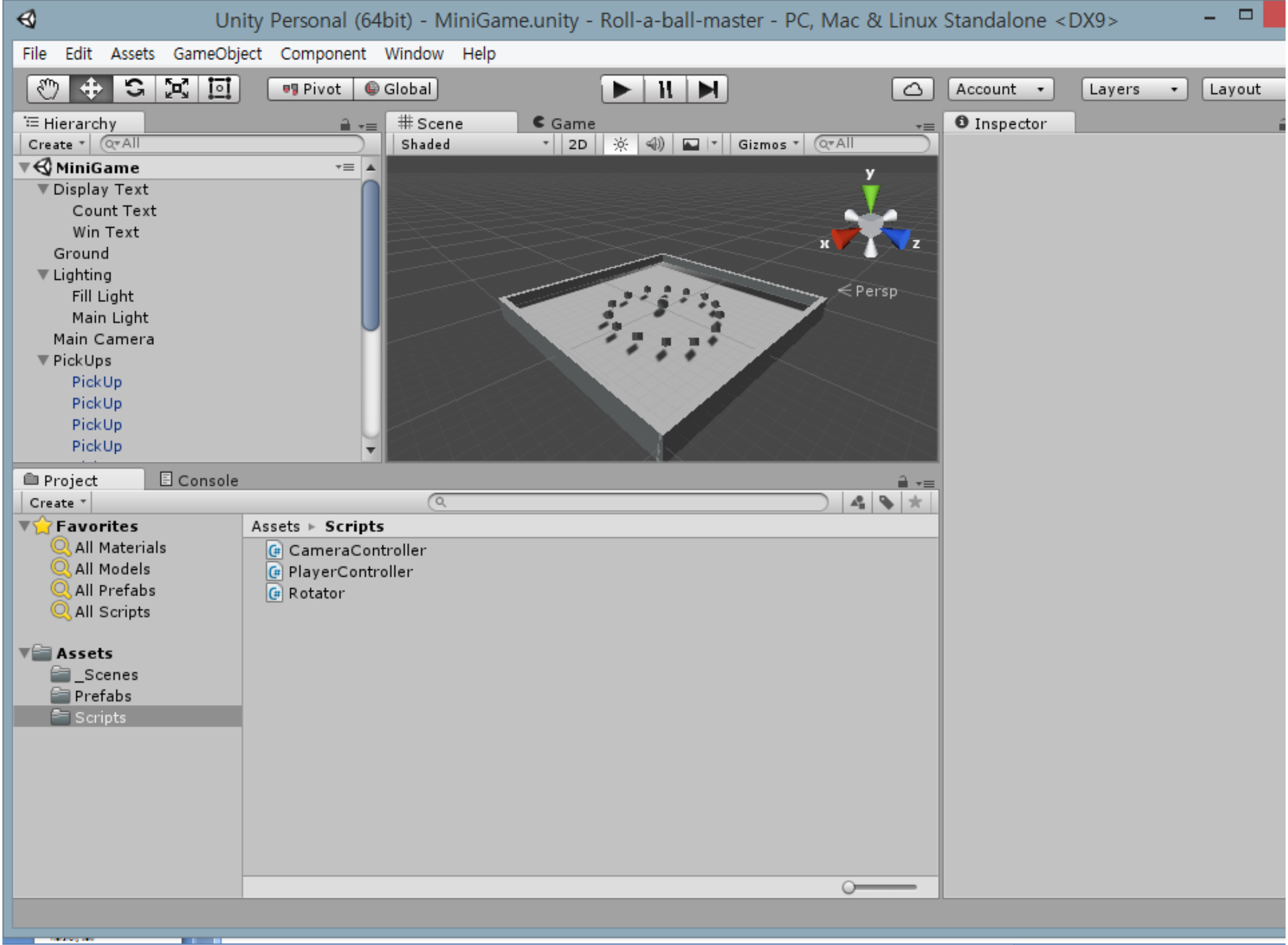
Lab 8. Modify Roll-a-ball project

- Exercise:

  - Understand Lab8_Roll-a-ball project.

  - Modify it as follows:
    - add a texture to the ball.
    - add a time bar, which expires after 3 minutes.
    - use coroutine.


Unity Personal (64bit) - MiniGame.unity - Roll-a-ball-master - PC, Mac & Linux Standalone <DX9>
File Edit Assets GameObject Component Window Help
Pivot Global
Account Layers Layout
Hierarchy
Create All
MiniGame
Display Text
Count Text
Win Text
Ground
Lighting
Fill Light
Main Light
Main Camera
PickUps
PickUp
PickUp
PickUp
PickUp
Scene Game
Display 1 Free Aspect Scale 1x Maximize
Count: 1
Inspector
Project Console
Create
Favorites
All Materials
All Models
All Prefabs
All Scripts
Assets
_Scenes
Prefabs
Scripts
Assets ▸ Scripts
CameraController
PlayerController
Rotator

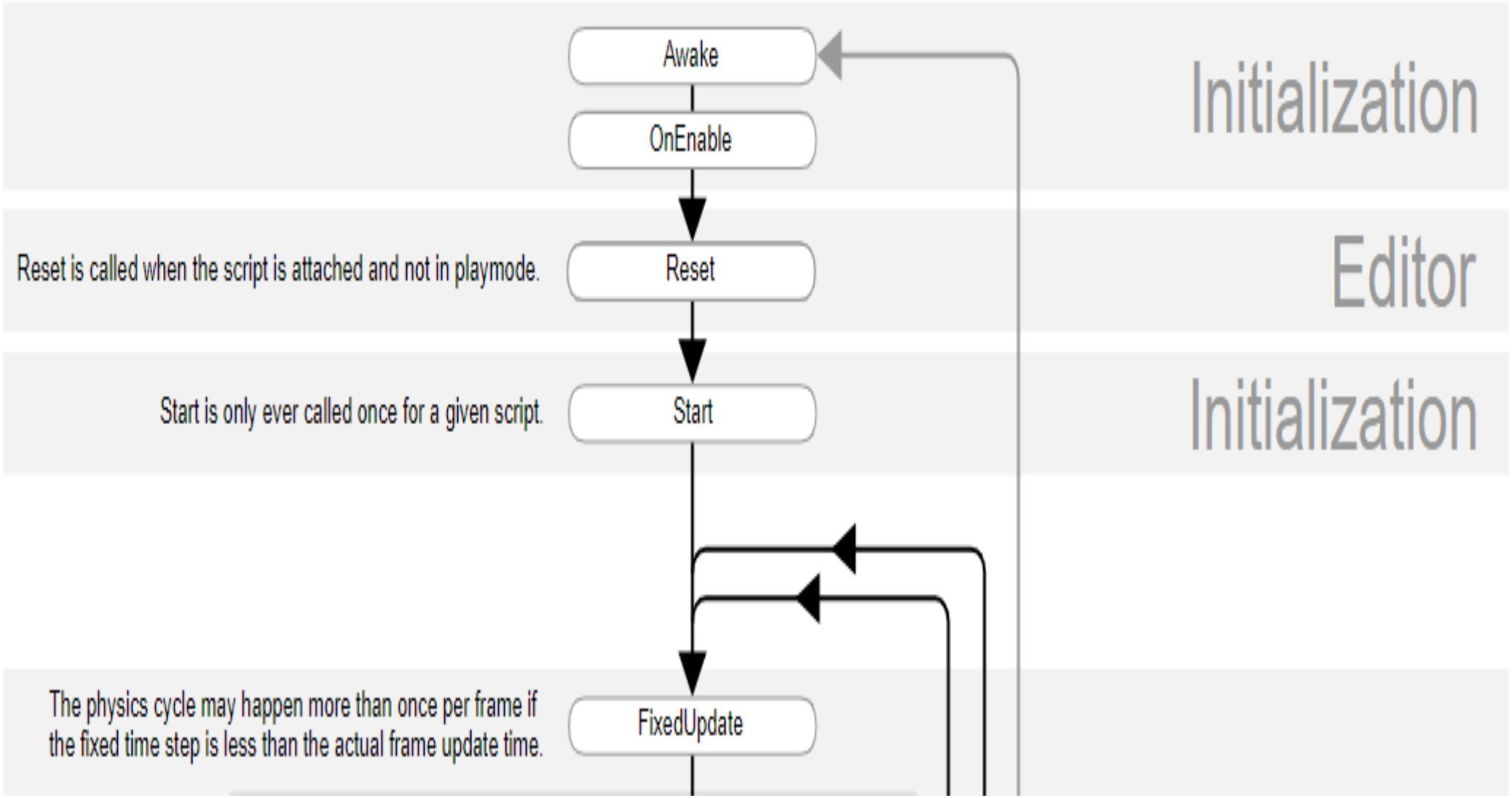

Unity Personal (64bit) - MiniGame.unity - Roll-a-ball-master - PC, Mac & Linux Standalone <DX9>
File Edit Assets GameObject Component Window Help
Pivot Global
Account Layers Layout
Hierarchy
Create All
MiniGame
Display Text
Count Text
Win Text
Ground
Lighting
Fill Light
Main Light
Main Camera
PickUps
PickUp
PickUp
PickUp
PickUp
Scene Game
Shaded 2D Gizmos All
y x z
Persp
Inspector
Project Console
Create
Favorites
All Materials
All Models
All Prefabs
All Scripts
Assets
_Scenes
Prefabs
Scripts
Assets ▸ Scripts
CameraController
PlayerController
Rotator

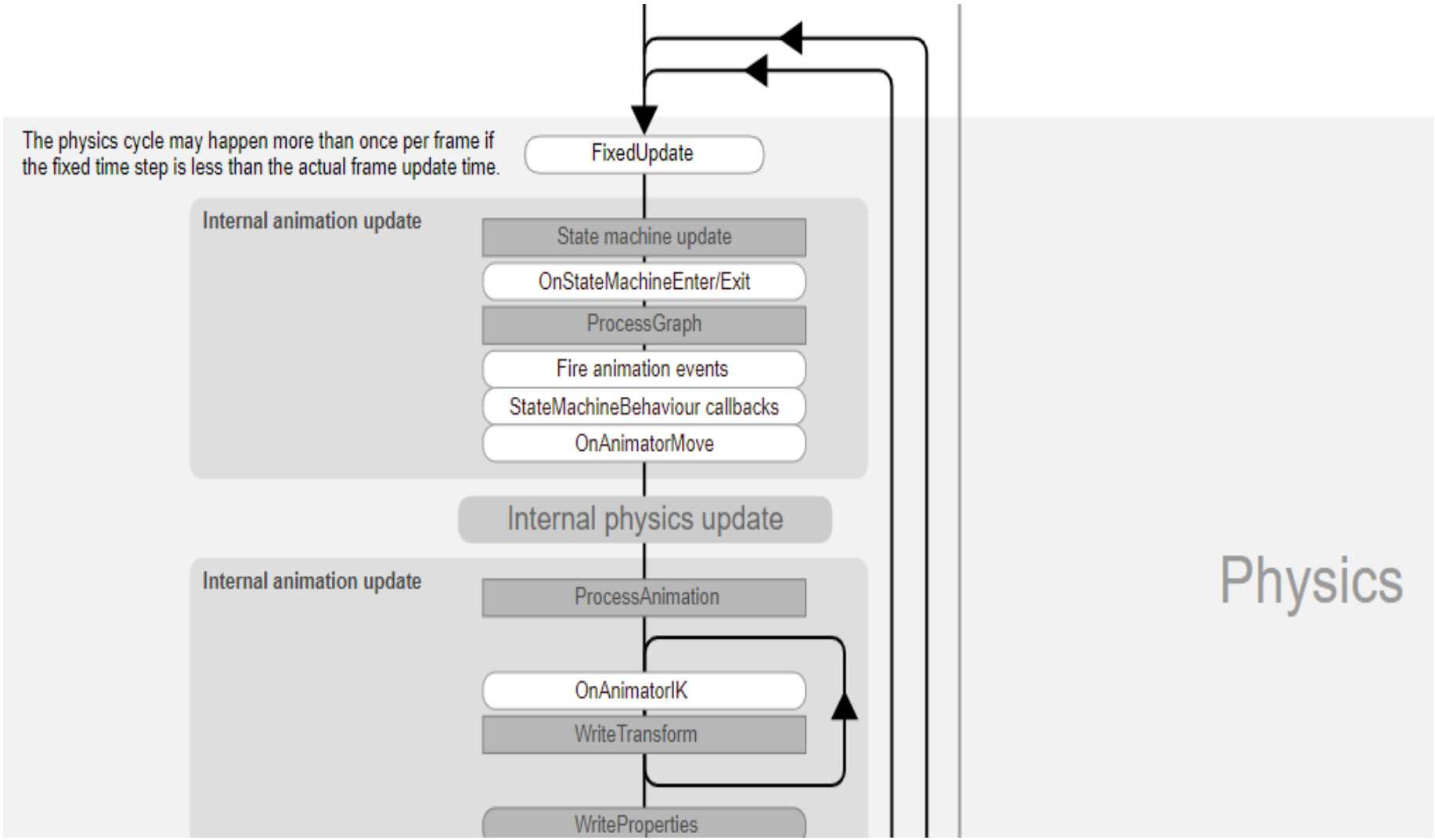
Order of Execution for Event Functions

https://docs.unity3d.com/Manual/ExecutionOrder.html

Awake() vs Start()

- Awake는 딱 한번 호출 됨. Script instance가 맨처음 초기화되는 과정에서 호출. Start()보다 먼저 호출. 여려 script 들간에 어떤 Awake()가 먼저 호출될 지는 보장안 함.
- Start()는 Script instance가 enable되면 한번 호출. Update()보다 먼저 호출 ➔ 초기에 script를 disable 시켰다가, 프로그램상에서 적당한 시점에 enable 시킴으로써, Start()를 호출하게 할 수 있음


Awake
OnEnable
Initialization
Reset is called when the script is attached and not in playmode.
Reset
Editor
Start is only ever called once for a given script.
Start
Initialization
The physics cycle may happen more than once per frame if the fixed time step is less than the actual frame update time.
FixedUpdate


The physics cycle may happen more than once per frame if the fixed time step is less than the actual frame update time.
FixedUpdate
Internal animation update
State machine update
OnStateMachineEnter/Exit
ProcessGraph
Fire animation events
StateMachineBehaviour callbacks
OnAnimatorMove
Internal physics update
Internal animation update
ProcessAnimation
OnAnimatorIK
WriteTransform
WriteProperties
Physics

Lab9. Coroutine

Coroutine

- It is essentially a function declared with a return type of **IEnumerator** and with the **yield return statement** included somew here in the body.
- The **yield** return line is the point at whic h execution will pause and be resumed t he following frame. **➔여러 플레임들에 걸쳐 수행되어야 하는 태스크에 적합**

- To set a coroutine running, you need to use the **StartCoroutine** function.
- In fact any variable or parameter will be correctly preserved between yields.

```
public class MyScript : MonoBehaviour
{
    void Start()
    {
        StartCoroutine(MyCoroutine());
    }

    IEnumerator MyCoroutine()
    {
        //This is a coroutine
    }
}
```

```
IEnumerator MyCoroutine()
{
    DoSomething();
    yield return 0; //Wait one frame, the 0 her
  e is only because we need to return an Ienu
  merable.
    DoSomethingElse();
}
```

```
• void Fade() {
   for (float f = 1f; f >= 0; f -= 0.1f) {
      Color c = renderer.material.color;
      c.a = f; //알파값(투명도) 변경 : 불투명
   ➔투명
      renderer.material.color = c;
    }
  }
```

이 함수는 하나의 프레임에서만 실행됩니다. 중간값은 시각적으로 알 수 없고, 오브젝트는 순간적으로 투명해집니다.

See the difference

```
• IEnumerator Fade() {
   for (float f = 1f; f >= 0; f -= 0.1f) {
      Color c = renderer.material.color;
      c.a = f;
      renderer.material.color = c;
       yield return null;
    }
  }
```

Fade가 시각적으로 알 수 있게 하기 위해서 알파는 몇 프레임에 걸쳐 렌더링되는 중간값으로 변경 한 후 점점 감소함

- By default, a coroutine is resumed on the frame after it yields but it is also possible to introduce a time delay using WaitForSeconds:

```
IEnumerator MyCoroutine()
{
    DoSomething();
    yield return new
  WaitForSeconds(2.0f);  //Wait 2 seconds
    DoSomethingElse();
}
```

- yield break; //for, while, if block을 빠져나올 때

- Start a coroutine
  - Ex) StartCoroutine(SpawnAfterSeconds( ));
  - Ex) StartCoroutine("SpawnAfterSeconds ");
- Stop a coroutine (강제로)
  - Ex) StopCoroutine("SpawnAfterSeconds ");

Start a Coroutine with Parameters

```
StartCoroutine("test", obj);
StartCoroutine(test2(2,2));

 IEnumerator test (object[] obj) {
                    Debug.Log ("obj=" + obj[0]);
                    yield return null;
 }

 IEnumerator test2 (int x, int y) {
                    Debug.Log ("x=" + x + "," + "y=" + y)
   ;
                    yield return null;
 }
```

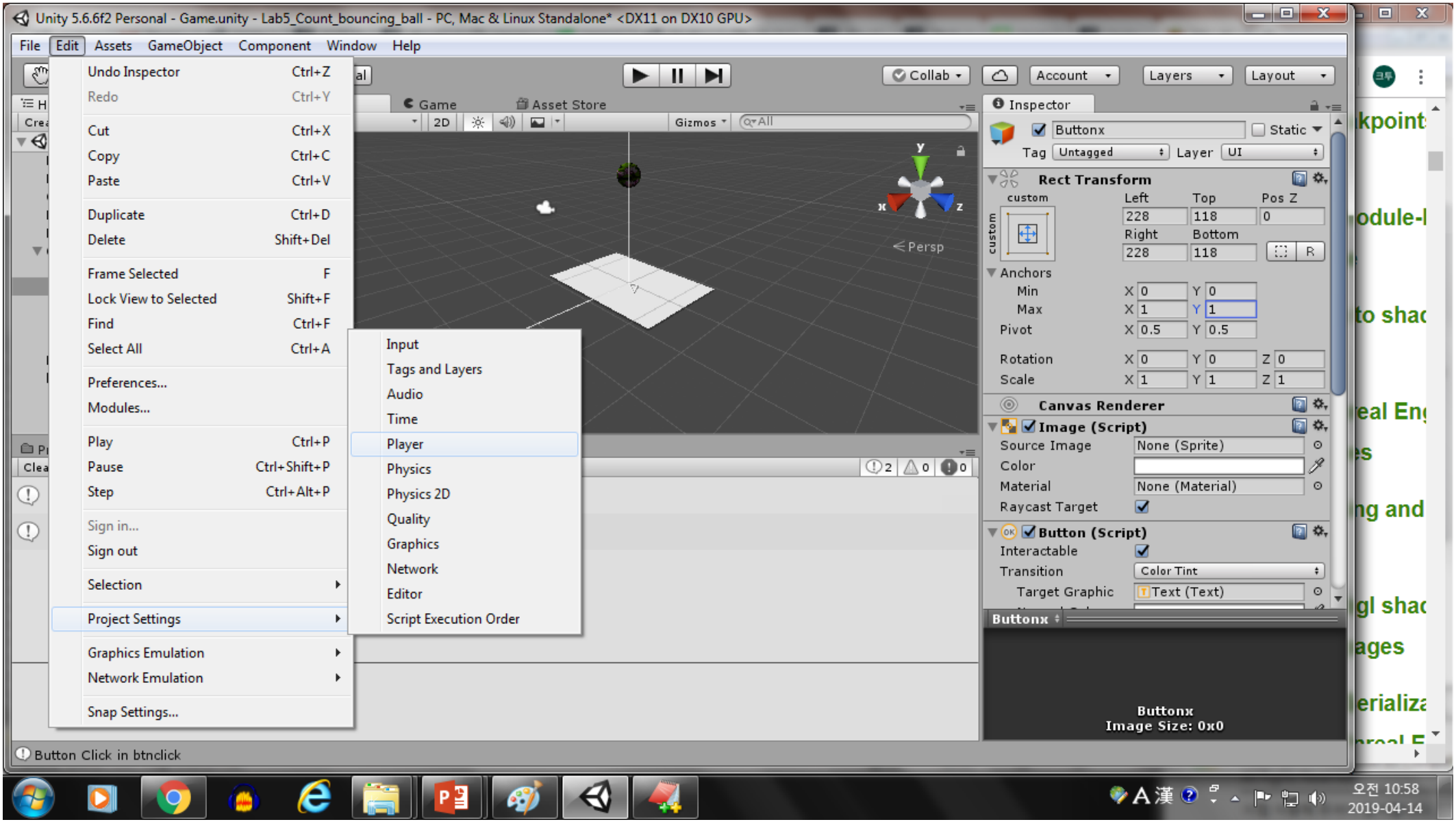
- 일반적인 코루틴 업데이트는 Update 함수가 반환된 후 실행. 코루틴은 주어진 Yield Instruction이 완료될 때까지 실행을 중단(양보)할 수 있는 함수.

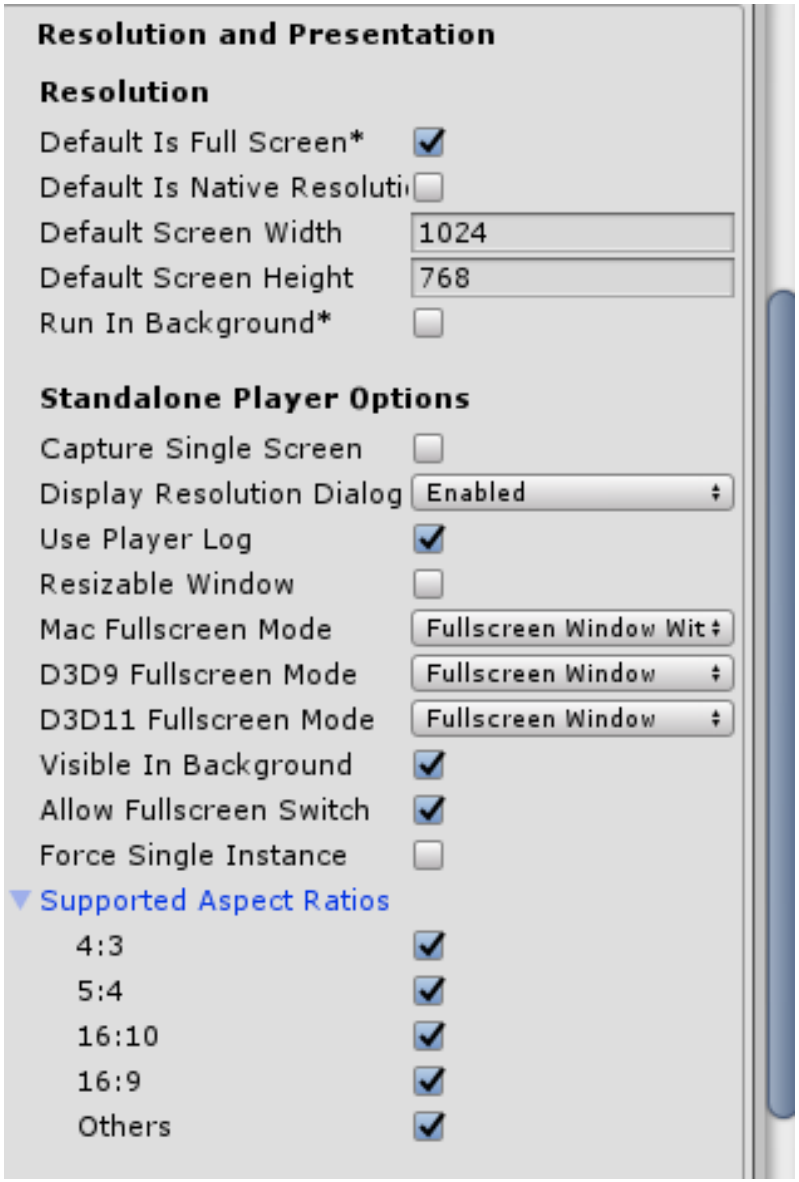
코루틴의 사용법:

- **yield 0** 코루틴은 모든 Update 함수가 다음 프레임에 호출된 후(실행후) 계속됩니다.
- **yield new WaitForSeconds(0.1f)** 지정한 시간이 지난 후, 모든 Update 함수가 프레임에 호출된 후 계속됩니다.
- **yield WaitForFixedUpdate()** 모든 FixedUpdate가 모든 스크립트에 호출된 후 계속됩니다.
- **yield ne WWW(url)** WWW 다운로드가 완료된 후 계속됩니다. Ex) yield new WWW(url)
- **yield StartCoroutine(**MyFunc **)** 코루틴을 연결하고 MyFunc 코루틴이 먼저 완료되기를 기다립니다.

주요 기능들

Managing Screen Resolution


Unity 5.6.6f2 Personal - Game.unity - Lab5_Count_bouncing_ball - PC, Mac & Linux Standalone* <DX11 on DX10 GPU>
File Edit Assets GameObject Component Window Help
Undo Inspector Ctrl+Z
Redo Ctrl+Y
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Duplicate Ctrl+D
Delete Shift+Del
Frame Selected F
Lock View to Selected Shift+F
Find Ctrl+F
Select All Ctrl+A
Preferences...
Modules...
Play Ctrl+P
Pause Ctrl+Shift+P
Step Ctrl+Alt+P
Sign in...
Sign out
Selection
Project Settings
Graphics Emulation
Network Emulation
Snap Settings...
Input
Tags and Layers
Audio
Time
Player
Physics
Physics 2D
Quality
Graphics
Network
Editor
Script Execution Order
Collab
Account
Layers
Layout
Game
Asset Store
Persp
Inspector
Buttonx
Static
Tag Untagged
Layer UI
Rect Transform
Left Top Pos Z
Right Bottom
Anchors
Min
Max
Pivot
Rotation
Scale
Canvas Renderer
Image (Script)
Source Image None (Sprite)
Color
Material None (Material)
Raycast Target
Button (Script)
Interactable
Transition Color Tint
Target Graphic Text (Text)
Buttonx
Image Size: 0x0
Button Click in btnclick


Resolution and Presentation
Resolution
Default Is Full Screen*
Default Is Native Resoluti
Default Screen Width
1024
Default Screen Height
768
Run In Background*
Standalone Player Options
Capture Single Screen
Display Resolution Dialog
Enabled
Use Player Log
Resizable Window
Mac Fullscreen Mode
Fullscreen Window Wit
D3D9 Fullscreen Mode
Fullscreen Window
D3D11 Fullscreen Mode
Fullscreen Window
Visible In Background
Allow Fullscreen Switch
Force Single Instance
Supported Aspect Ratios
4:3
5:4
16:10
16:9
Others

Choose the **fullscreen window mode** instead. In fullscreen window mode, your game will run inside a maximized borderless window.

- When you select a *Default Screen Width/Height* in player settings **greater** than the current display resolution, Unity will use the current display resolution (i.e ignores your setting).
- When you select a Default Screen Width/Height **smaller** than the current display resolution, Unity will use this smaller size to render the current frame in memory and then upscale it to fit the full-resolution window.

➔ script에서 Screen.width, Screen.height 확인

Unity Remote

- Use “Unity Remote” app to test this unity application.
  - IOS: Use WiFi to connect a PC and a device.
  - Android: Use a Cable to connect a PC and a device.