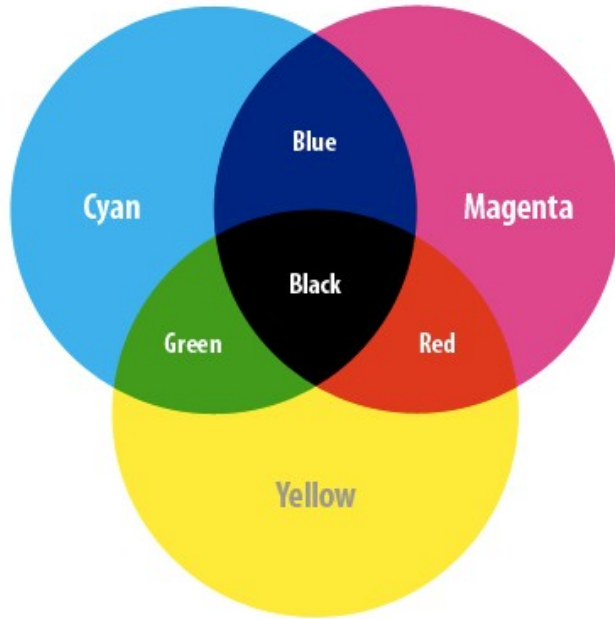


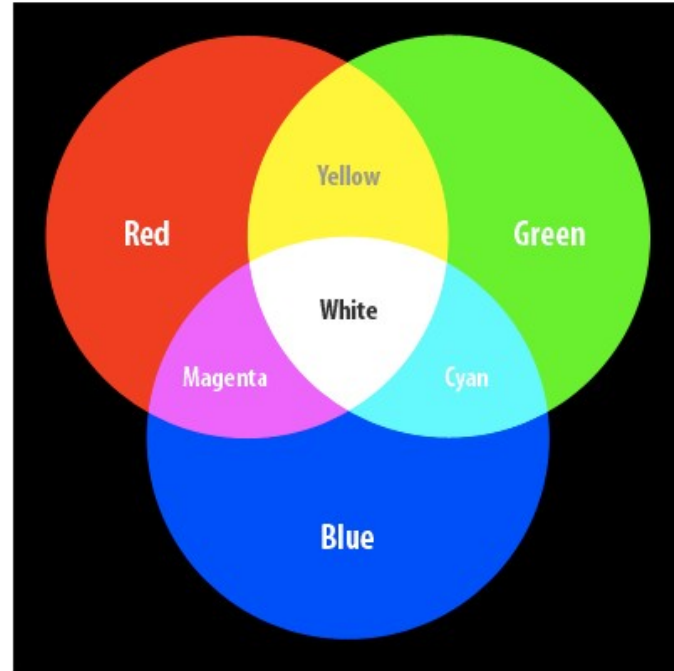
Color Systems

CMYK – The Subtractive System



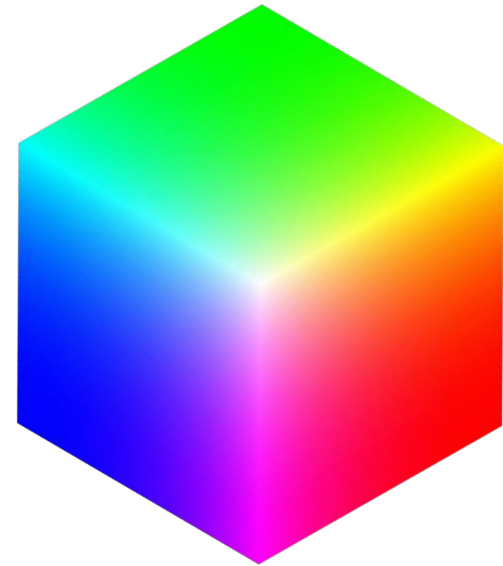
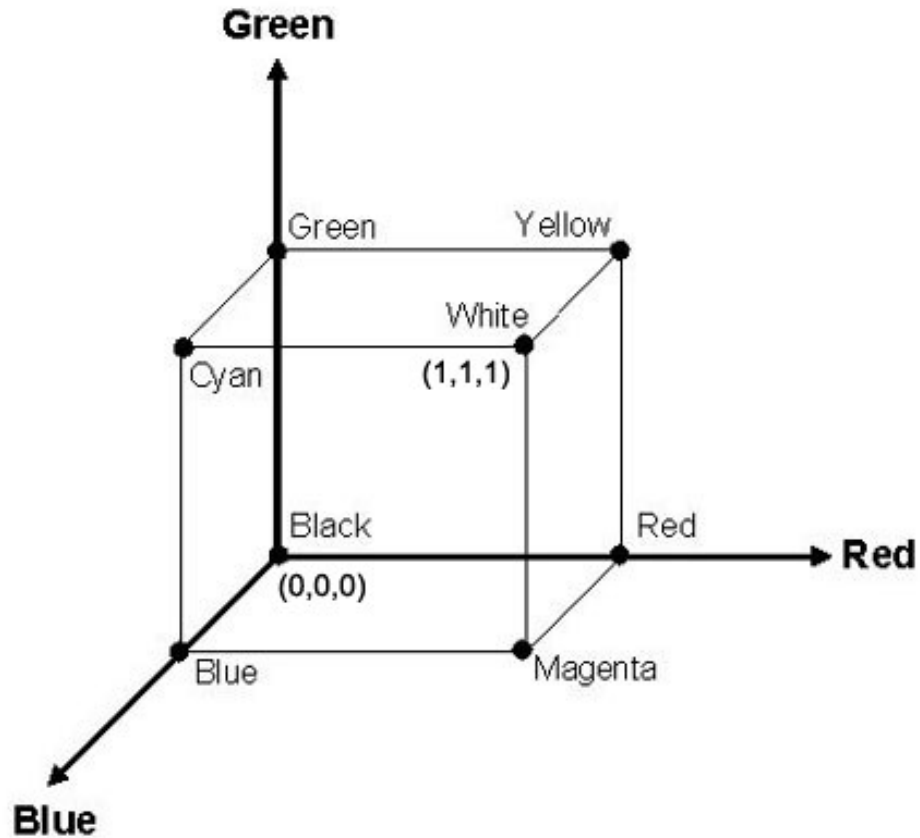
CMYK Color Model

RGB – The Additive System



RGB Color Model

Color Cube



RGBA Color

Representation of RGBA colors

Values for red, green, blue and alpha are floating point values with a range from 0 to 1 (ex. 0.3, 0.77)

Alpha component (a) defines transparency

alpha of 1 is completely opaque, alpha of zero is completely transparent

Black RGBA is (0, 0, 0, 1)

Blue RGBA is (0, 0, 1, 1)

Gray RGBA is (0.5, 0.5, 0.5, 1)

Clear Completely transparent. RGBA is (0, 0, 0, 0)

RGBA Color

Black RGBA is (0, 0, 0, 1)

Blue RGBA is (0, 0, 1, 1)

Gray RGBA is (0.5, 0.5, 0.5, 1)

Clear Completely transparent. RGBA is (0, 0, 0, 0)

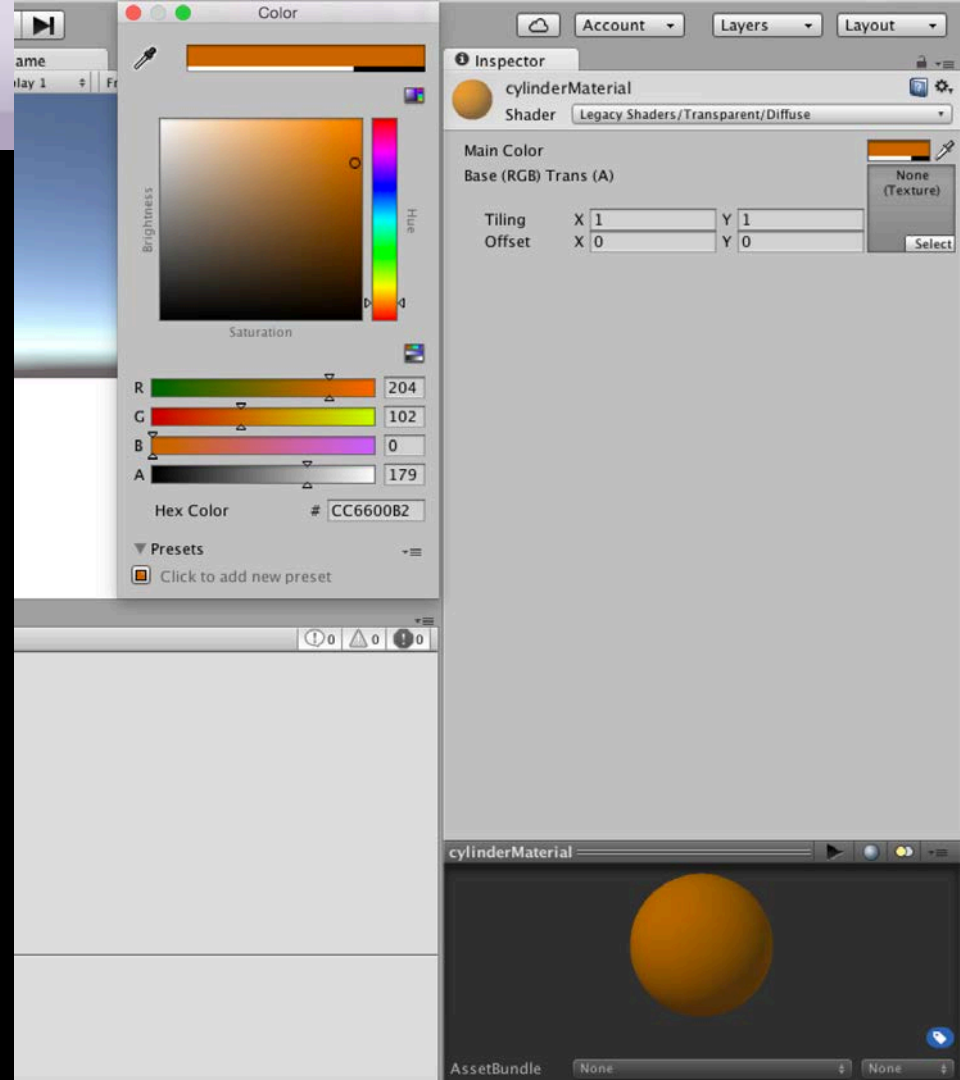
Magenta?

Yellow?

cyan?

RGBA Color

Unity Color Window



If statement

```
public class ifstatement : MonoBehaviour {  
    void Update() {  
        if (Input.GetKeyDown(KeyCode.C))  
        {  
            GetComponent<Renderer> ().material.color = Color.cyan;  
        }  
        if (Input.GetKeyDown(KeyCode.M))  
        {  
            GetComponent<Renderer>().material.color = Color.magenta;  
        }  
        if (Input.GetKeyDown(KeyCode.Y))  
        {  
            GetComponent<Renderer>().material.color = Color.yellow;  
        }  
    }  
}
```

Color script

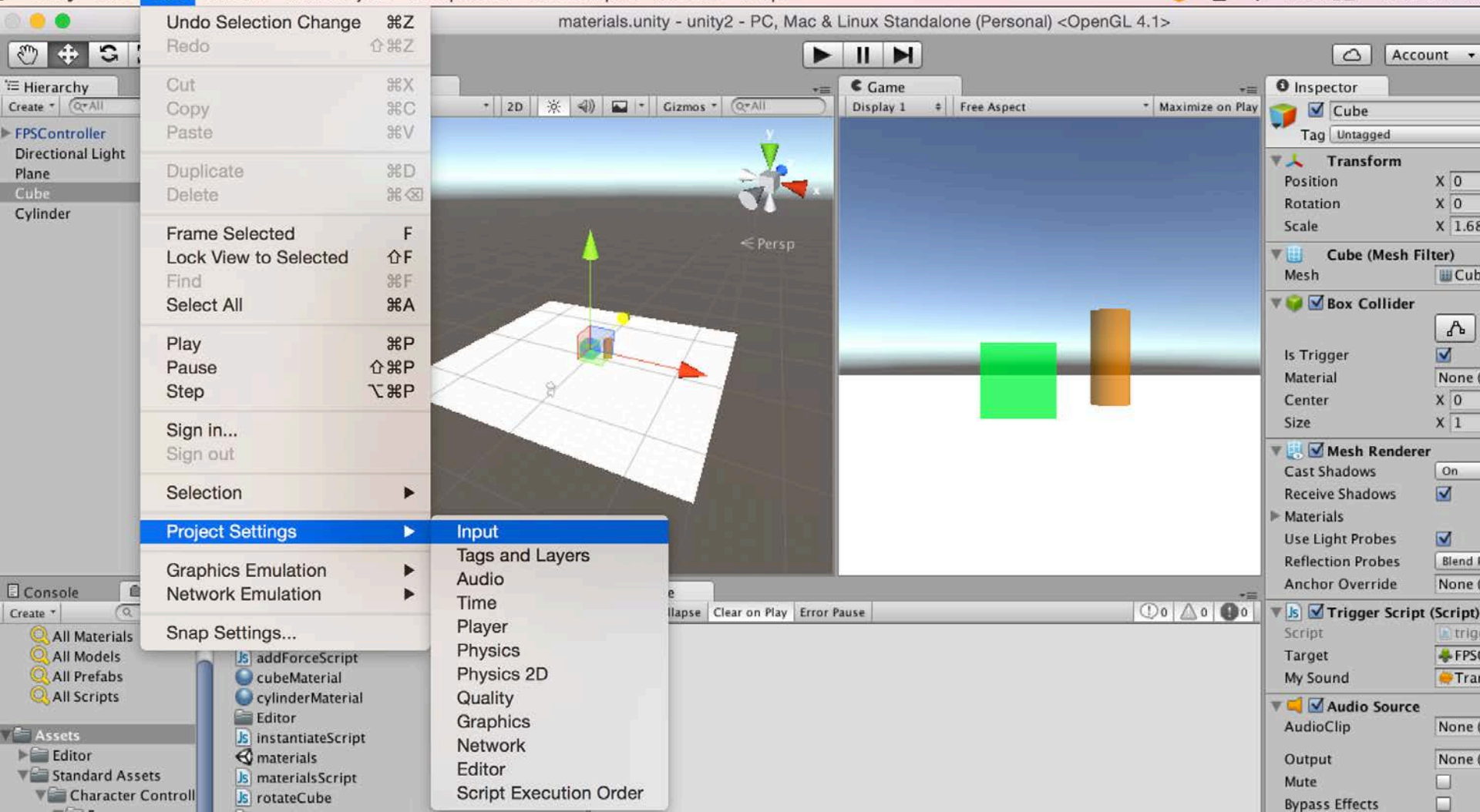
```
public class ifstatement : MonoBehaviour {  
    void Update() {  
        if (Input.GetKeyDown(KeyCode.C))  
        {  
            GetComponent<Renderer>().material.color=new Color(0.0f, 0.5f, 0.2f, 0.9f);  
        }  
        if (Input.GetKeyDown(KeyCode.M))  
        {  
            GetComponent<Renderer>().material.color = Color.magenta;  
        }  
        if (Input.GetKeyDown(KeyCode.Y))  
        {  
            GetComponent<Renderer>().material.color = Color.yellow;  
        }  
    }  
}
```

Interaction – Key and Button input

The input manager allows to name an input and specify a key or button for it.

You can access keys on the keyboard and mouse buttons through scripting interface.

Edit > Project Settings > Input



- Undo Selection Change ⌘Z
- Redo ⇧⌘Z
- Cut ⌘X
- Copy ⌘C
- Paste ⌘V
- Duplicate ⌘D
- Delete ⌘⌫

- Frame Selected F
- Lock View to Selected ⇧F
- Find ⌘F
- Select All ⌘A

- Play ⌘P
- Pause ⇧⌘P
- Step ⌘⇧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

- addForceScript
- cubeMaterial
- cylinderMaterial
- Editor
- instantiateScript
- materials
- materialsScript
- rotateCube

- All Materials
- All Models
- All Prefabs
- All Scripts

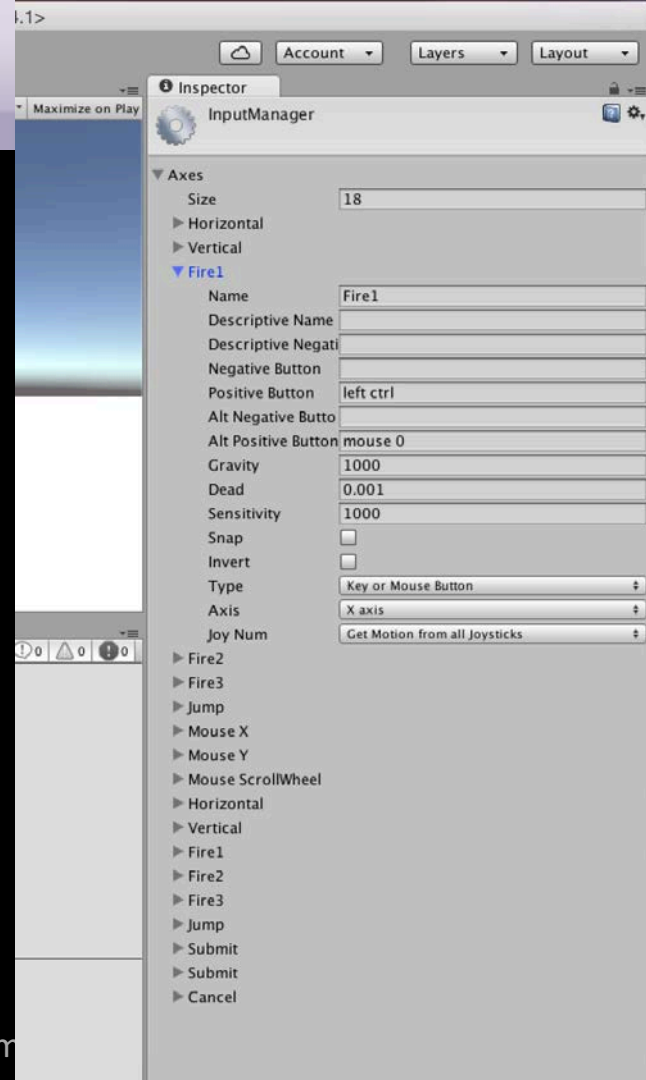
- Assets
- Editor
- Standard Assets
- Character Control

Interaction – Key and Button input

```
Input.GetButtonDown("Fire1")  
    Mouse button left  mouse 0
```

```
Input.GetButtonDown("Fire2")  
    Mouse button right  mouse 1
```

```
Input.GetKeyDown("Jump")  
    space key    space
```



Button Input script

```
public class ButtonInput : MonoBehaviour
{
    void Update()
    {
        if (Input.GetButtonDown("Fire1"))
        {
            GetComponent<Renderer>().material.color=new Color(0.0f, 0.5f, 0.2f, 0.9f);
        }
        if (Input.GetButtonDown("Fire2"))
        {
            GetComponent<Renderer>().material.color = Color.magenta;
        }
        if (Input.GetButtonDown("Jump"))
        {
            GetComponent<Renderer>().material.color = Color.yellow;
        }
    }
}
```

Unity Documentation - KeyCode

ImagePosition
IMECompositionMode
JointDriveMode
JointLimitState2D
JointProjectionMode
KeyCode
LegDof
LightmapBakeType
LightmapsMode
LightmapsModeLegacy
LightRenderMode
LightShadowCasterMode
LightShadows
LightType
LineAlignment
LineTextureMode
LocationServiceStatus
LODFadeMode
LogType

Properties

None	Not assigned (never returned as the result of a keystroke).
Backspace	The backspace key.
Delete	The forward delete key.
Tab	The tab key.
Clear	The Clear key.
Return	Return key.
Pause	Pause on PC machines.
Escape	Escape key.
Space	Space key.
Keypad0	Numeric keypad 0.
Keypad1	Numeric keypad 1.
Keypad2	Numeric keypad 2.
Keypad3	Numeric keypad 3.

Interaction – Key and Button input

Create new object Sphere, add rigidbody component

Create new Script AddForceScript.cs

Assign script to the sphere

Add Force

```
void Update () {  
    if (Input.GetKey("up"))  
    {  
        Debug.Log("up arrow key is held down");  
    }  
    if (Input.GetKey("down"))  
    {  
        Debug.Log("down arrow key is held down");  
    }  
    if (Input.GetKeyDown(KeyCode.Space))  
    {  
        GetComponent<Rigidbody>().AddForce(transform.forward * 200f);  
    }  
}
```

Add Force

Uncheck Use Gravity in the inspector
Test the scene

Add Force

Add useGravity to Rigidbody component in the script

```
GetComponent<Rigidbody>().AddForce(transform.forward * 500f);
```

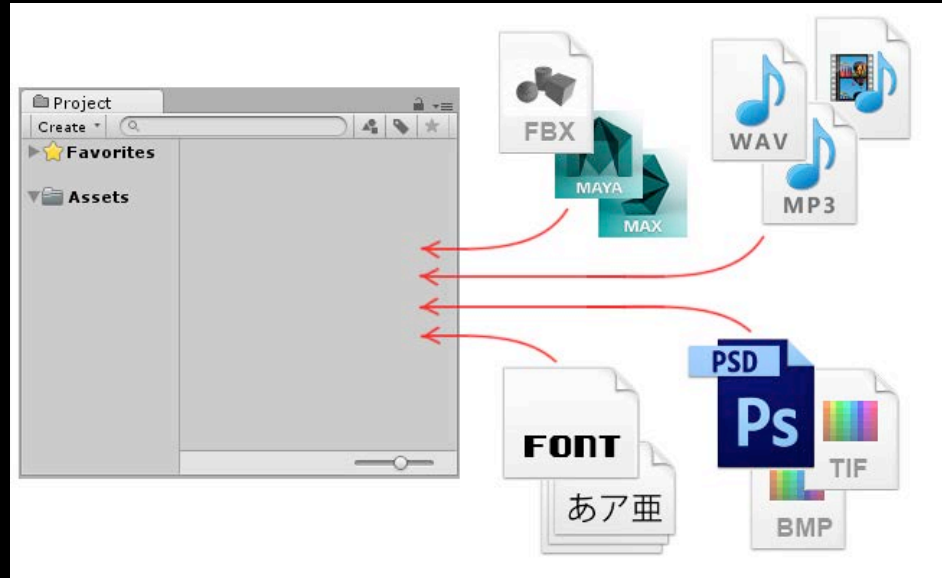
```
GetComponent<Rigidbody>().useGravity = true;
```

```
}
```


Project Assets

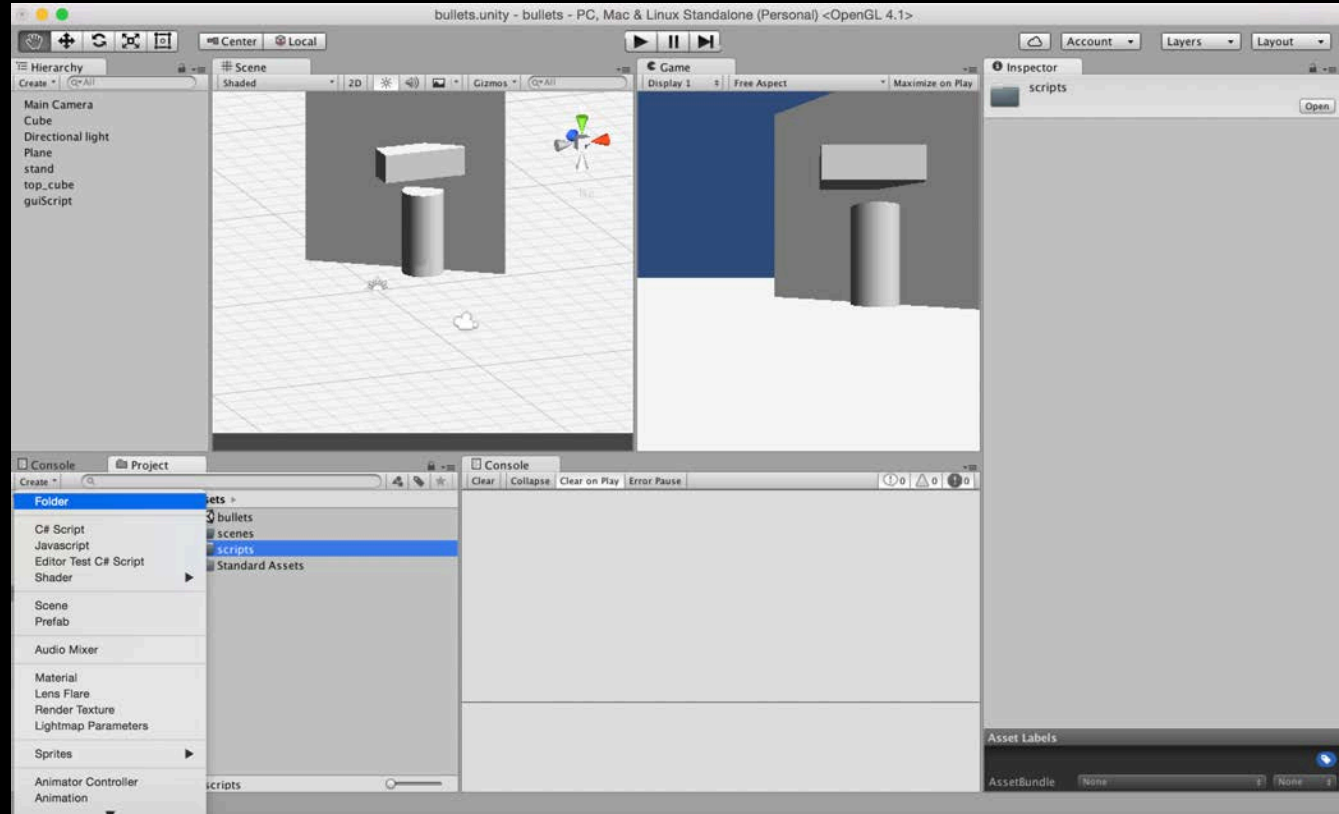
Assets are the models, textures, sounds and all other media files from which you make your 3D project

- Audio Files
- Materials
- Meshes
- Textures
- Prefabs
- Scripts
- Prefabs



Project Organization

Project > Create > Folder

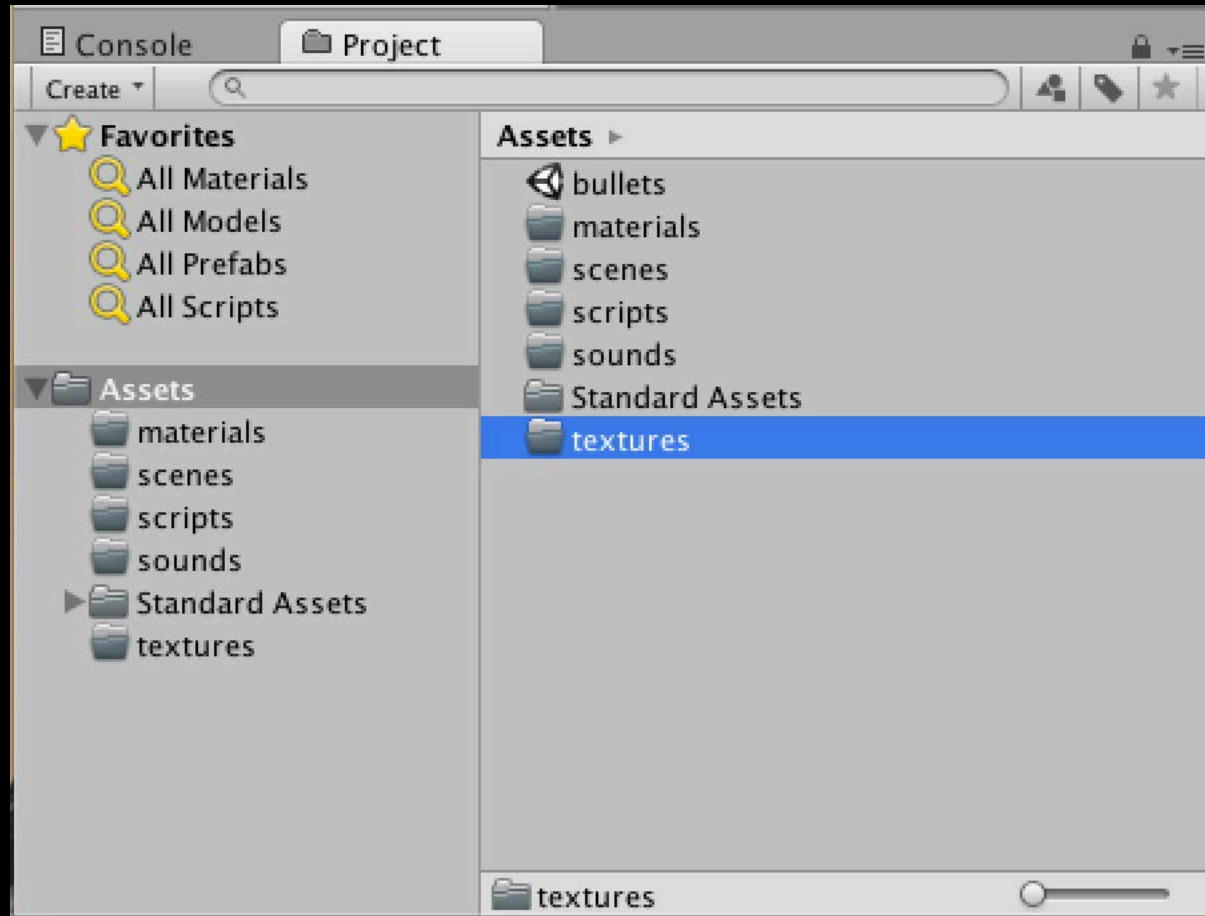


Project Organization

Assets > Folder >

Rename

- Materials
- Scripts
- Scenes
- Prefabs
- Textures
- Sounds



Project Organization

- Keep your assets organized
- Be consistent with naming. If you decide to use camel case for directory names and low letters for assets, stick to that convention.
- Use lowercase folder names
- Use camelCase naming convention if needed
- Sort all the assets in the corresponding folders

Project Organization

- Do not store any asset files in the root directory. Use subdirectories whenever possible.
- Do not create any additional directories in the root directory, unless you really need to.
- Use 3rd-Party to store assets imported from the Asset Store. They usually have their own structure that shouldn't be altered.

Project Organization

3rd-Party

Animations

Audio

- Music

- SFX

Materials

Models

Plugins

Prefabs

Resources

Textures

Sandbox

Scenes

- Levels

- Other

Scripts

- Editor

Shaders

Unity Components

Components are the nuts & bolts of objects and behaviors in a game.

A GameObject is a container for many different Components.

Unity Components

Animation components

Asset components

Audio components

Physics components

The GameObject

Image effect scripts

Settings managers

Mesh components

Network group

Particle components

Rendering components

Transform component

UnityGUI group

Wizards

Unity Components

Audio components

Implement sound in Unity

- Audio Listener

Add this to a Camera to get 3D positional sound.

- Audio Source

Add this Component to a GameObject to make it play a sound

- Audio Effects

Unity Components

The Game Object

GameObjects are containers for all other Components

All objects in your scene are inherently GameObjects

They are containers that hold Components, which implement actual functionality. Ex. a Light is a Component which is attached to a GameObject.

Unity Components

Mesh components

3D Meshes are the main graphics primitive of Unity

Various components exist in Unity to render regular or skinned meshes, trails or 3D lines

Mesh Filter

Mesh Renderer

Skinned Mesh Renderer

Text Mesh

Unity Components

Particle components

Particle Systems are used to make effects

Smoke

Steam

Fire

Atmospheric effects

Particle systems in Unity work by using one or two textures (2D), and drawing them many times, creating a chaotic random effect

Unity Components

Particle components

Ellipsoid Particle Emitter

Line Renderer

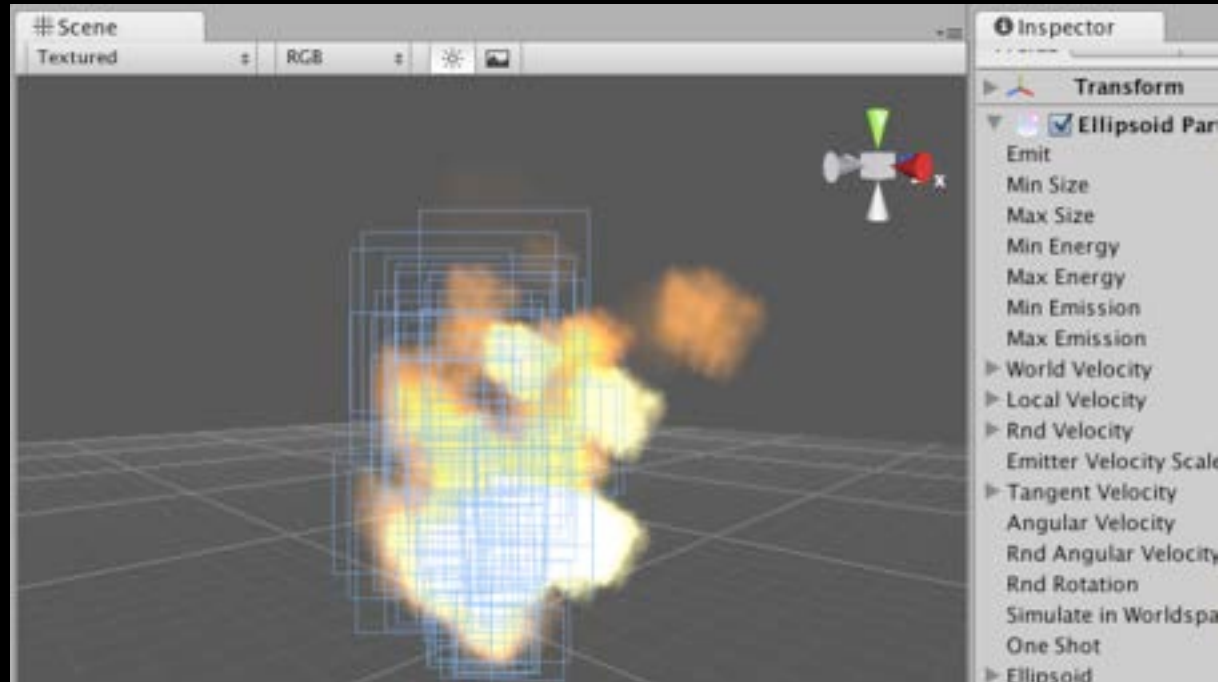
Mesh Particle Emitter

Particle Animator

Particle Renderer

Trail Renderer

Particle Collider



Unity Components

Particle components

Ellipsoid Particle Emitter

spawns particles inside a sphere

Use the Ellipsoid property to scale & stretch the sphere

Unity Components

Particle components

Line Renderer

draws a straight line between two or more points in 3D space

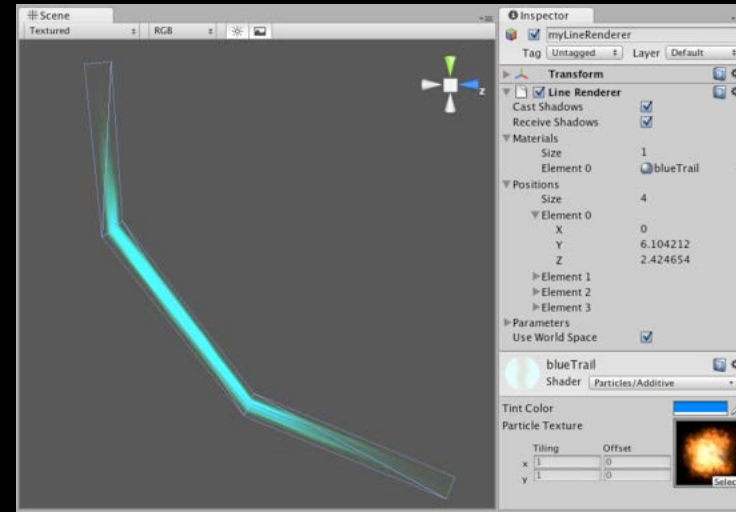
can be used to draw anything from a simple straight line, to a complex spiral

renders billboard lines that have width

and can be textured

uses the same algorithm for line rendering

as the Trail Renderer

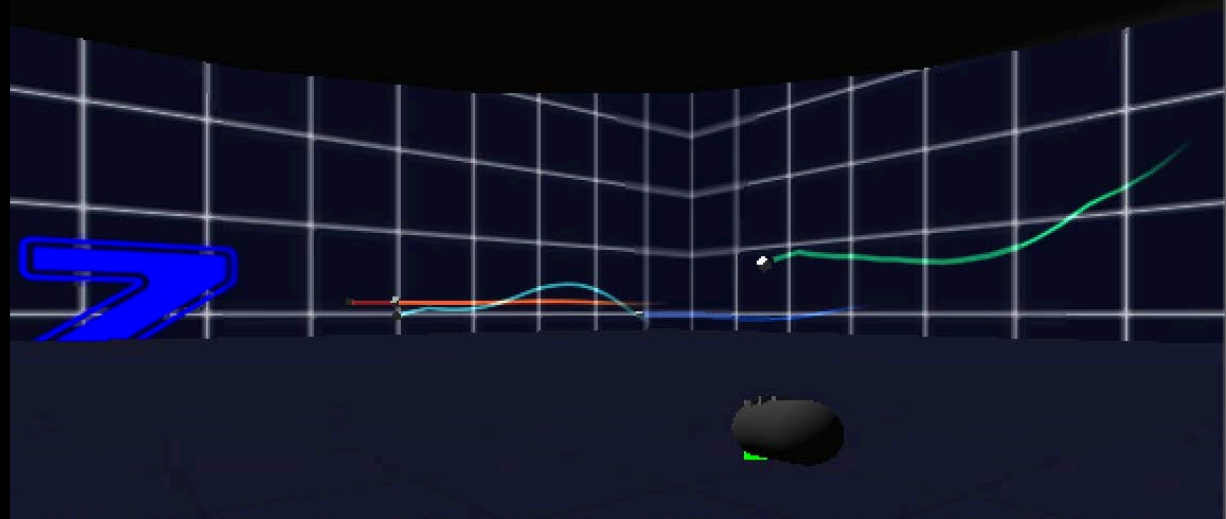


Unity Components

Particle components

Trail Renderer

makes trails behind moving objects in the scene



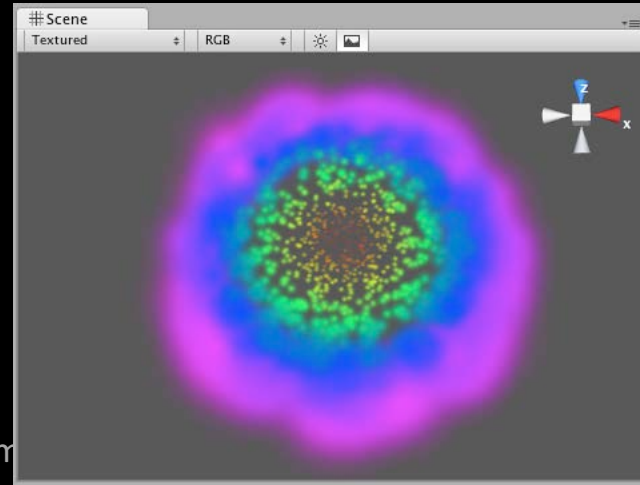
Unity Components

Particle components

Particle animator

Move particles over time

Used to apply wind, drag and color cycling to particle systems



Unity Components

Particle components

Mesh particle emitter

emits particles around a mesh

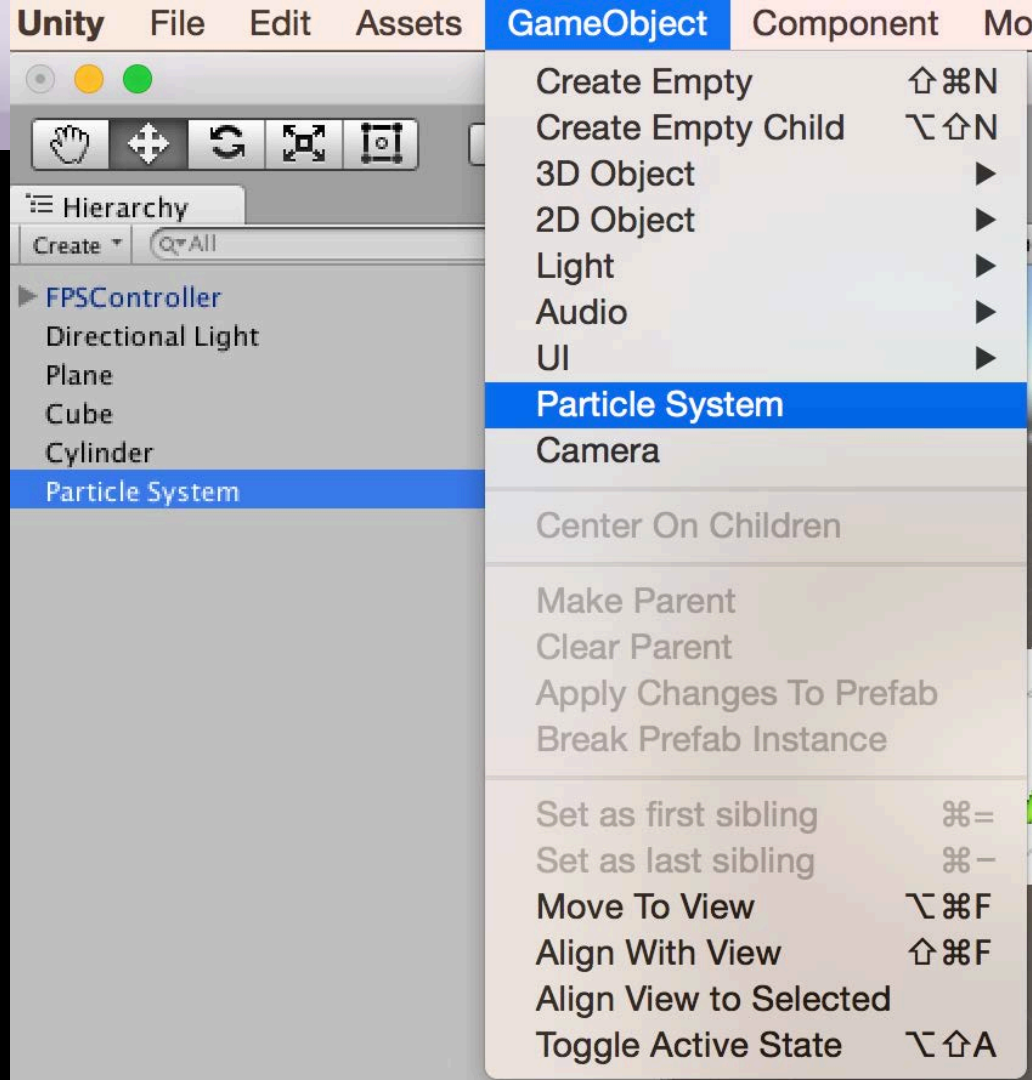
particles are spawned from the surface of the mesh



Particle component

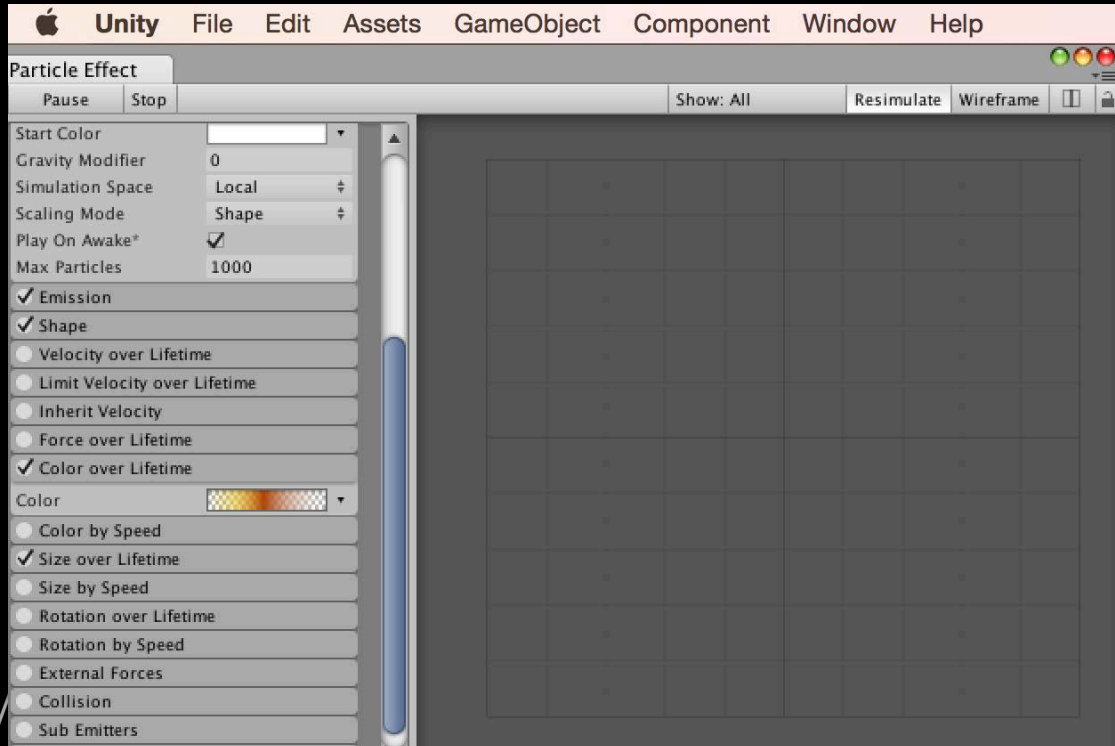
Open your Unity scene
GameObject > Particle System

Rename it "Confetti"



Particle component

Particle Effect window / Inspector
GameObject > Particle System



Particle Script - confetti

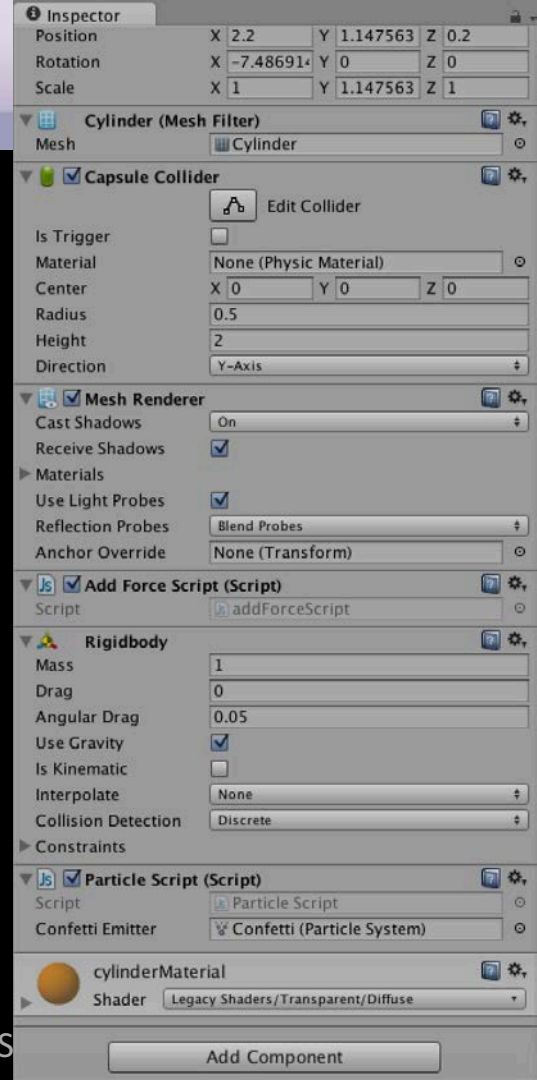
```
public class ParticleScript : MonoBehaviour { public ParticleSystem  
  
    confettiEmitter;  
    void Start () {  
    }  
  
    void Update () {  
  
        if (Input.GetButtonDown("Fire1"))  
        {  
            confettiEmitter.Emit(30);  
        }  
    }  
}
```

Particle component

Assign “ParticleSystem” to a cylinder
(drag and drop)

Assign Confetti Particles to
Confetti Emitter variable in the Inspector
(drag and drop from Hierarchy)

Uncheck “Play on Awake” box in the Inspector
Uncheck “Looping” box
With selected Confetti object in Hierarchy





Center Local



Account Layers Layout

Hierarchy

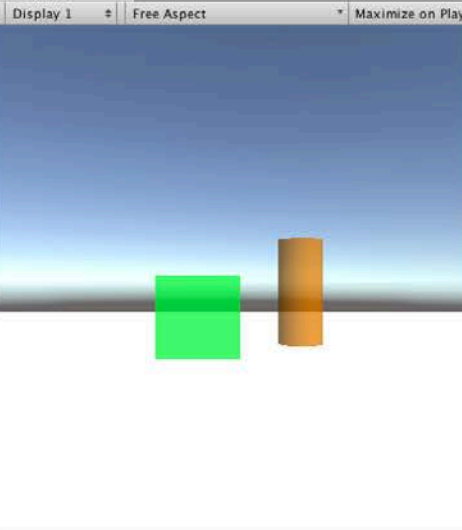
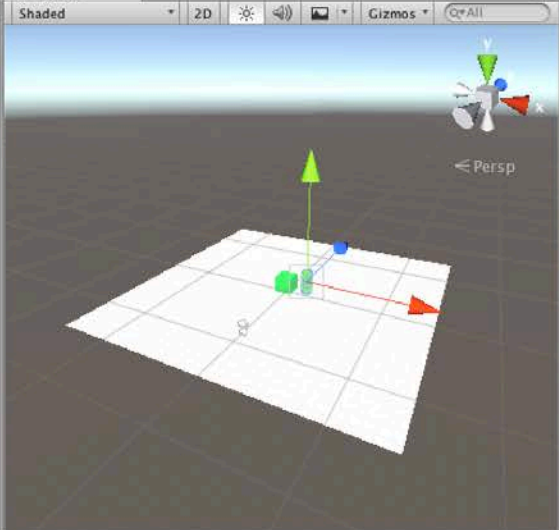
Scene

Game

Inspector

Create All

- FPSController
- Directional Light
- Plane
- Cube
- Cylinder
- Confetti



Position X 2.2 Y 1.5 Z 0.2
 Rotation X 0 Y 0 Z 0
 Scale X 1 Y 1.147563 Z 1

Cylinder (Mesh Filter)
 Mesh Cylinder

Capsule Collider
 Edit Collider
 Is Trigger
 Material None (Physics Material)
 Center X 0 Y 0 Z 0
 Radius 0.5
 Height 2
 Direction Y-Axis

Mesh Renderer
 Cast Shadows On
 Receive Shadows
 Materials
 Use Light Probes
 Reflection Probes Blend Probes
 Anchor Override None (Transform)

Add Force Script (Script)
 Script addForceScript

Rigidbody
 Mass 1
 Drag 0
 Angular Drag 0.05
 Use Gravity
 Is Kinematic
 Interpolate None
 Collision Detection Discrete

Particle Script (Script)
 Script Particle Script
 Confetti Emitter Confetti (Particle System)

cylinderMaterial
 Shader Legacy Shaders/Transparent/Diffuse

Console Project

Console

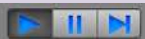
Create All

- All Prefabs
- All Scripts
- Assets
 - addForceScript
 - cubeMaterial
 - cylinderMaterial
 - Editor
 - instantiateScript
 - materials
 - materialsScript
 - Particle Script
 - particles
 - rotateCube
 - Standard Assets
 - Transition Out
 - triggerScript
 - Variables
 - varsandFunctions
- Editor
- Standard Assets
 - Character Controll
 - Sources
 - PrototypeCha
 - Scripts
 - Characters
 - FirstPersonChar
 - Audio
 - Prefabs
 - Scripts
 - RollerBall
 - ThirdPersonCha

Clear Collapse Clear on Play Error Pause



Center Local

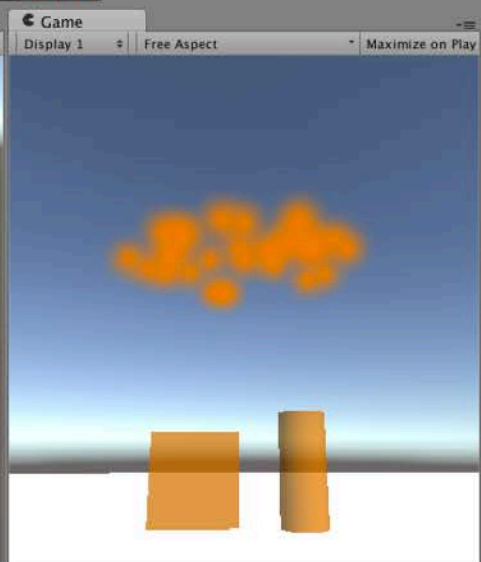
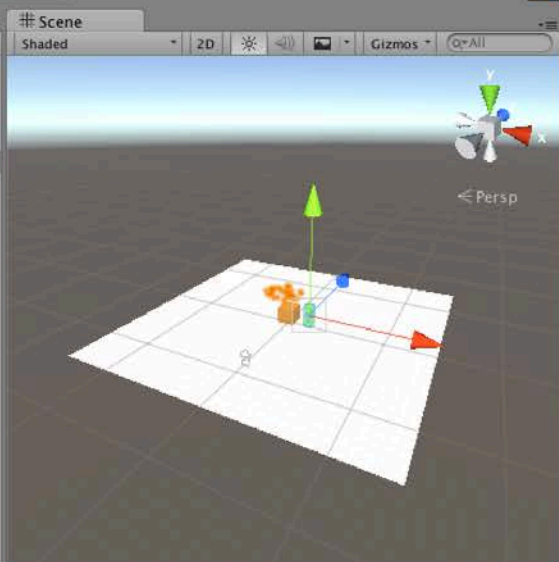


Account Layers Layout

Hierarchy

Create + Q*All

- FPSController
- Directional Light
- Plane
- Cube
- Cylinder
- Confetti



Inspector

Position X 2.2 Y 1.147563 Z 0.2
 Rotation X -7.48691 Y 0 Z 0
 Scale X 1 Y 1.147563 Z 1

Cylinder (Mesh Filter)

Mesh Cylinder

Capsule Collider

Is Trigger
 Material None (Physic Material)
 Center X 0 Y 0 Z 0
 Radius 0.5
 Height 2
 Direction Y-Axis

Mesh Renderer

Cast Shadows On
 Receive Shadows
 Materials
 Use Light Probes
 Reflection Probes Blend Probes
 Anchor Override None (Transform)

Add Force Script (Script)

Script addForceScript

Rigidbody

Mass 1
 Drag 0
 Angular Drag 0.05
 Use Gravity
 Is Kinematic
 Interpolate None
 Collision Detection Discrete

Constraints

Particle Script (Script)

Script Particle Script
 Confetti Emitter Confetti (Particle System)

cylinderMaterial

Shader Legacy Shaders/Transparent/Diffuse

Console

Project

Create +

All Prefabs
 All Scripts

Assets

- addForceScript
- cubeMaterial
- cylinderMaterial
- Editor
- InstantiateScript
- materials
- materialsScript
- Particle Script
- particles
- rotateCube
- Standard Assets
- Transition Out
- triggerScript
- Variables
- varsandFunctions

Console

Clear Collapse Clear on Play Error Pause

0 0 0

Particle component

Test the mouse button input and interaction

The particle should emit 30 particles on mouse click

Parenting

Parent-Child Relationship in Unity

The Child Game Object will inherit the behaviour of its parent
It will move, rotate, scale exactly as its Parent does.

Similar to Arm/Body relationship- whenever your body moves, your arm moves along with it

- This is a way of grouping objects together
- Children can have their own children and etc.
- Complex parent-child structure
- Represented in the Hierarchy window

Particle Script - confetti

Add another if statement to add force to push cylinder forward:

```
public ParticleSystem confettiEmitter;

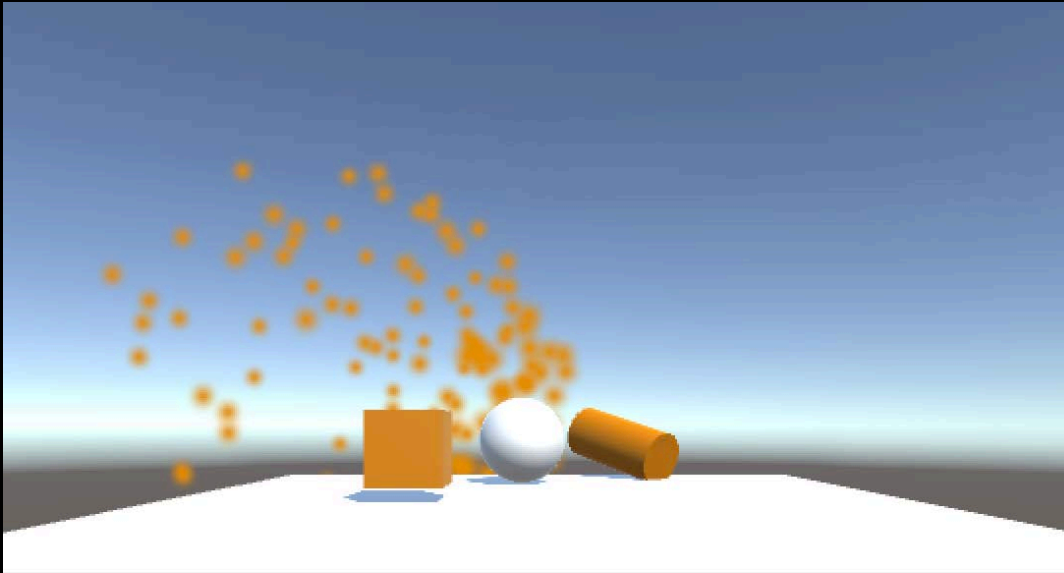
void Update () {

    if (Input.GetButtonDown("Fire1"))
    {
        confettiEmitter.Emit(30);
    }
    if (Input.GetKeyDown(KeyCode.Tab))
    {
        GetComponent<Rigidbody>().AddForce(transform.forward * 200f);
    }
}
```

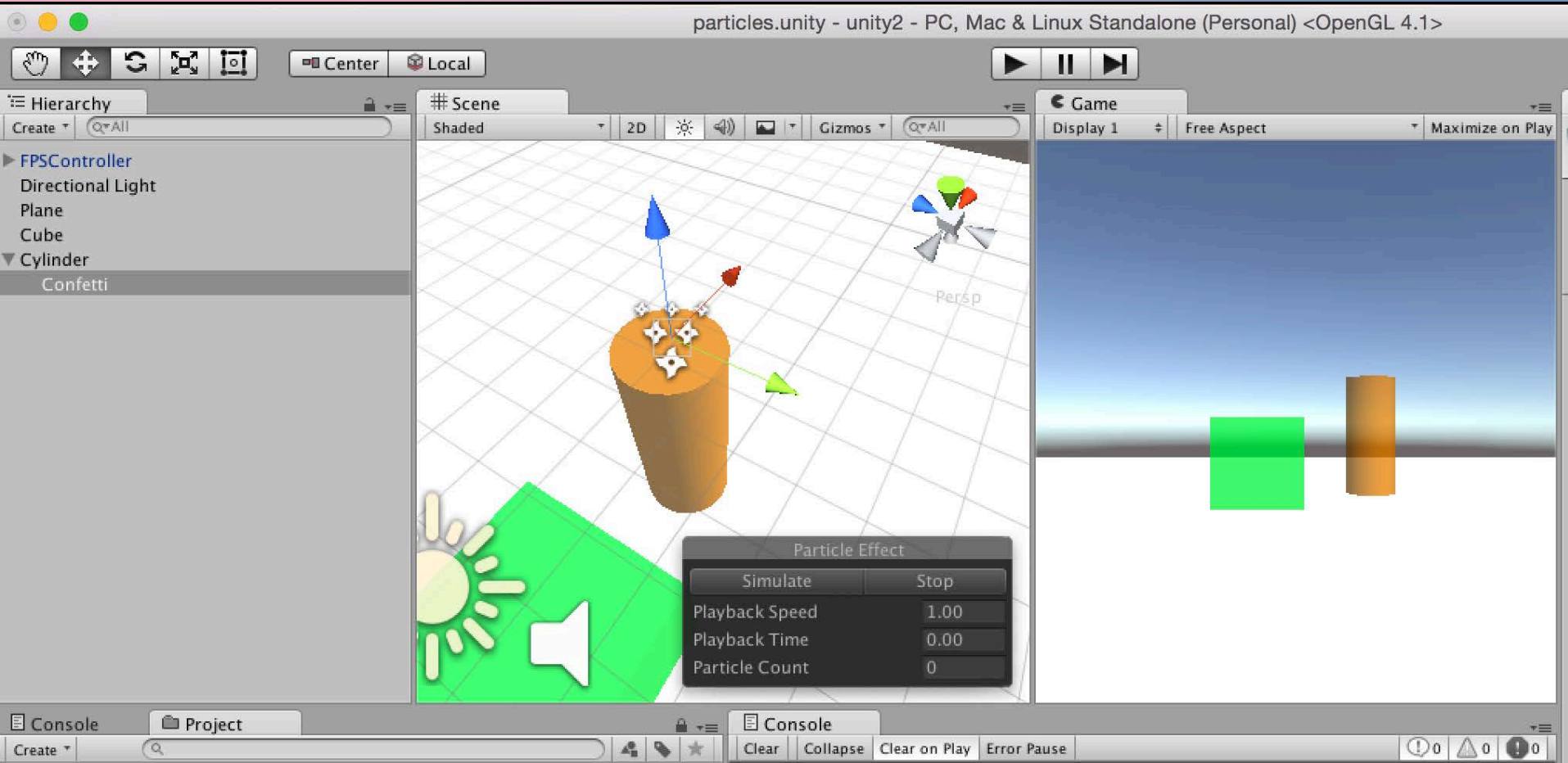
Parenting

Drag and drop Confetti particles inside the Cylinder object in the Hierarchy window

Test the game



Parenting



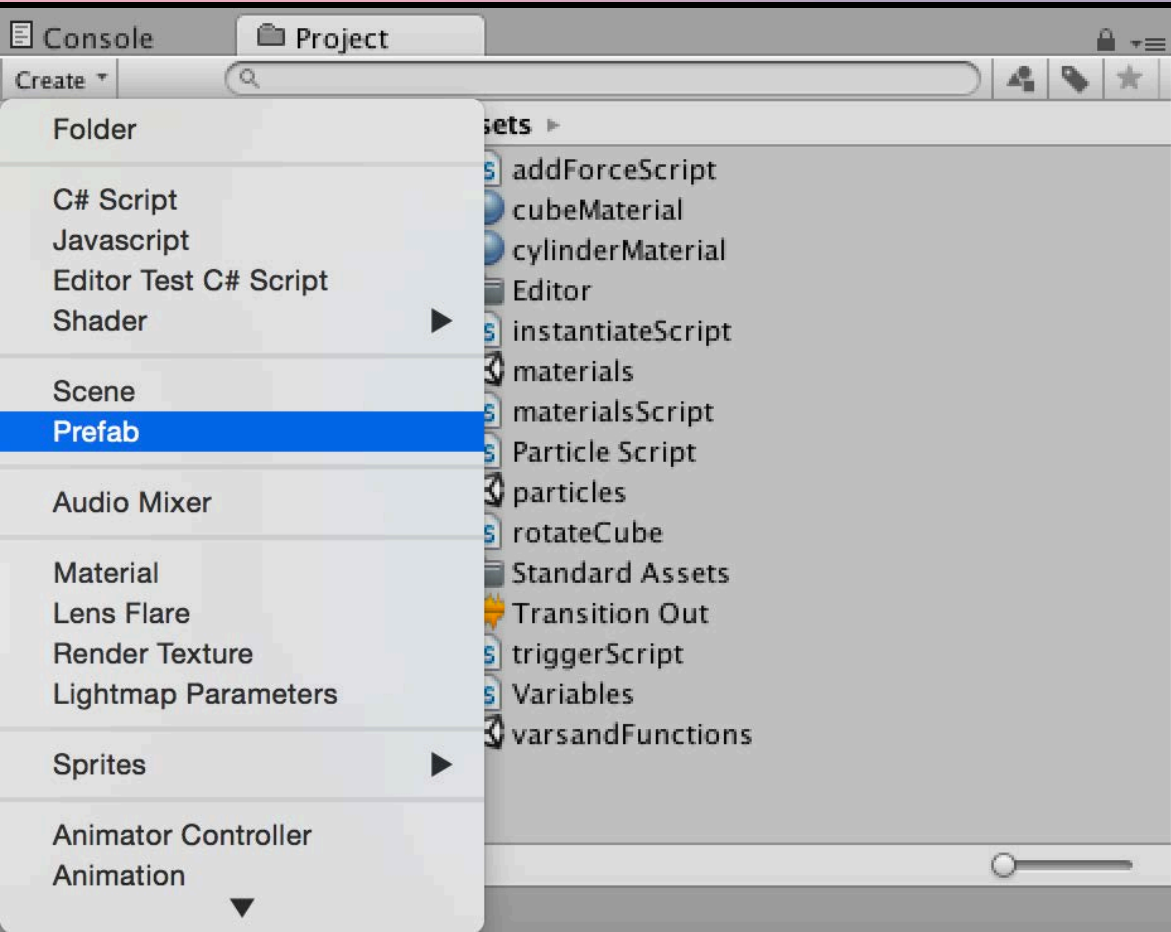
Prefabs

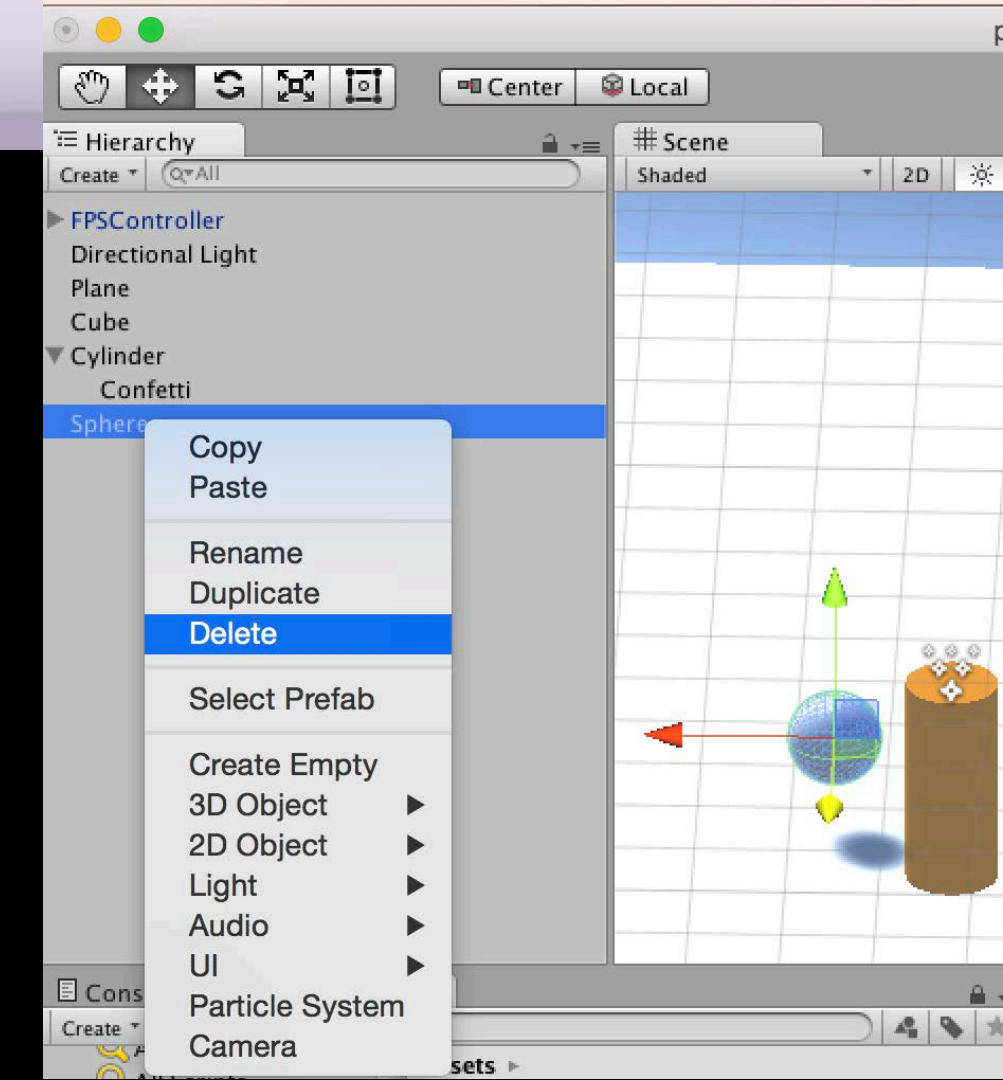
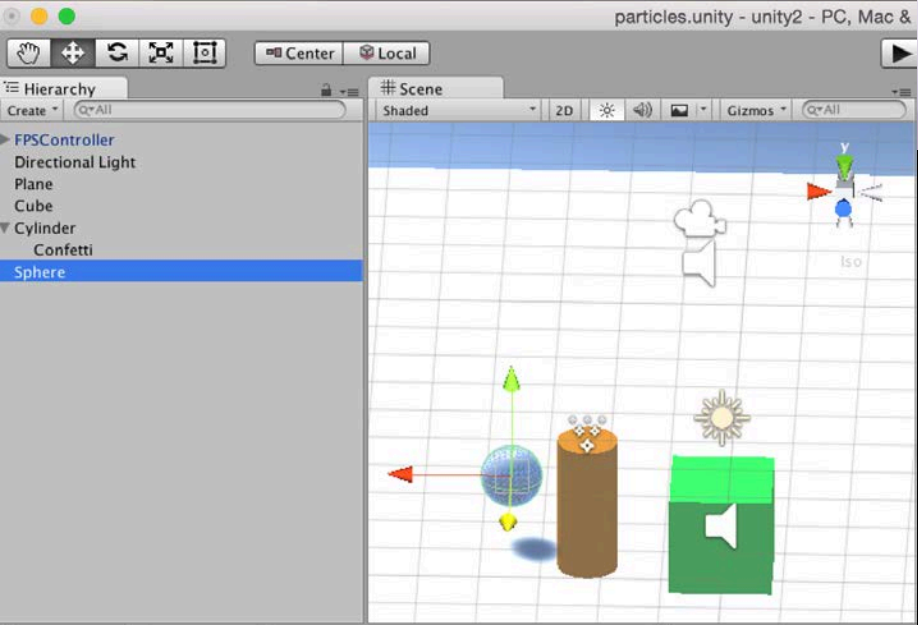
Prefabs and reusable assets

Create Prefab:

- Project > Create > Prefab
- Rename it “PrefabSphere”
- Game Object > 3D > Sphere
- Inspector : Add Rigidbody Component
- Drag and drop sphere into PrefabSphere
- Delete Sphere in the Hierarchy window / scene

Prefabs





Instantiate

Instantiate is to create objects during run-time (as the game is being played)

Instantiate function takes three parameters;

- (1) the object we want to create
- (2) the 3D position of the object
- (3) the rotation of the object

Instantiate

1. Which object to instantiate?

the best way is to expose a variable

We can state which object to instantiate by using drag and drop to assign a game object to this variable prefab

2 Where to instantiate it?

create the new game object to locate new prefab whenever the Fire2 button is pressed.

Prefabs

Create Instantiate script and assign it to the cylinder:

```
public class InstantiateScript : MonoBehaviour
{
    public Rigidbody spherePrefab;
    public Transform placetoStart;
```

Prefabs

```
void Update ()
{
  if(Input.GetButtonDown("Fire2"))
  {
    Rigidbody newInstance;

    newInstance = Instantiate (spherePrefab, placetoStart.position,
    placetoStart.rotation);

    newInstance.AddForce(placetoStart.forward * 50);
  }
}
```

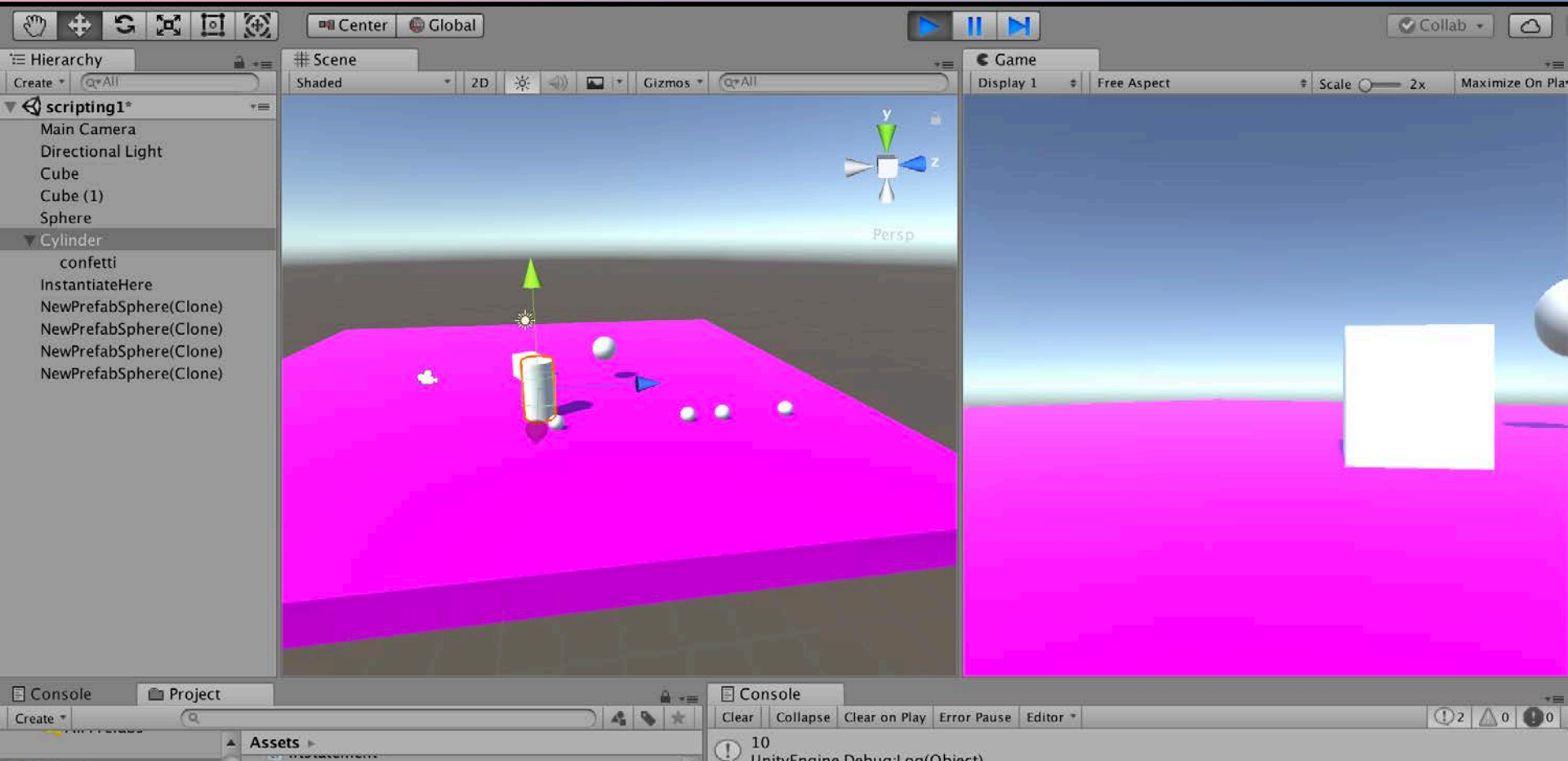
Prefabs

Create a new GameObject and assign it as a reference to Transform placetoStart in the inspector

Assign prefabShpere to spherePrefab in the Inspector

Test the game ad press mouse button 2 to instantiate spheres

Prefabs



Instantiate- Cleaning Instances

To clean generated prefabs from the scene, attach the following script “PrefabDestruction” to the prefabSphere

```
using UnityEngine;
using System.Collections;

public class PrefabDestruction : MonoBehaviour
{
    void Start()
    {
        Destroy (gameObject, 5.5f);
    }
}
```