



Game Development with **Unity3D**

REVIEWED

By Saima Sipy at 5:52 am, Feb 24, 2022



Table of Contents

1. Introduction to Unity
2. Concepts and workflow
3. Live demo



Part 1

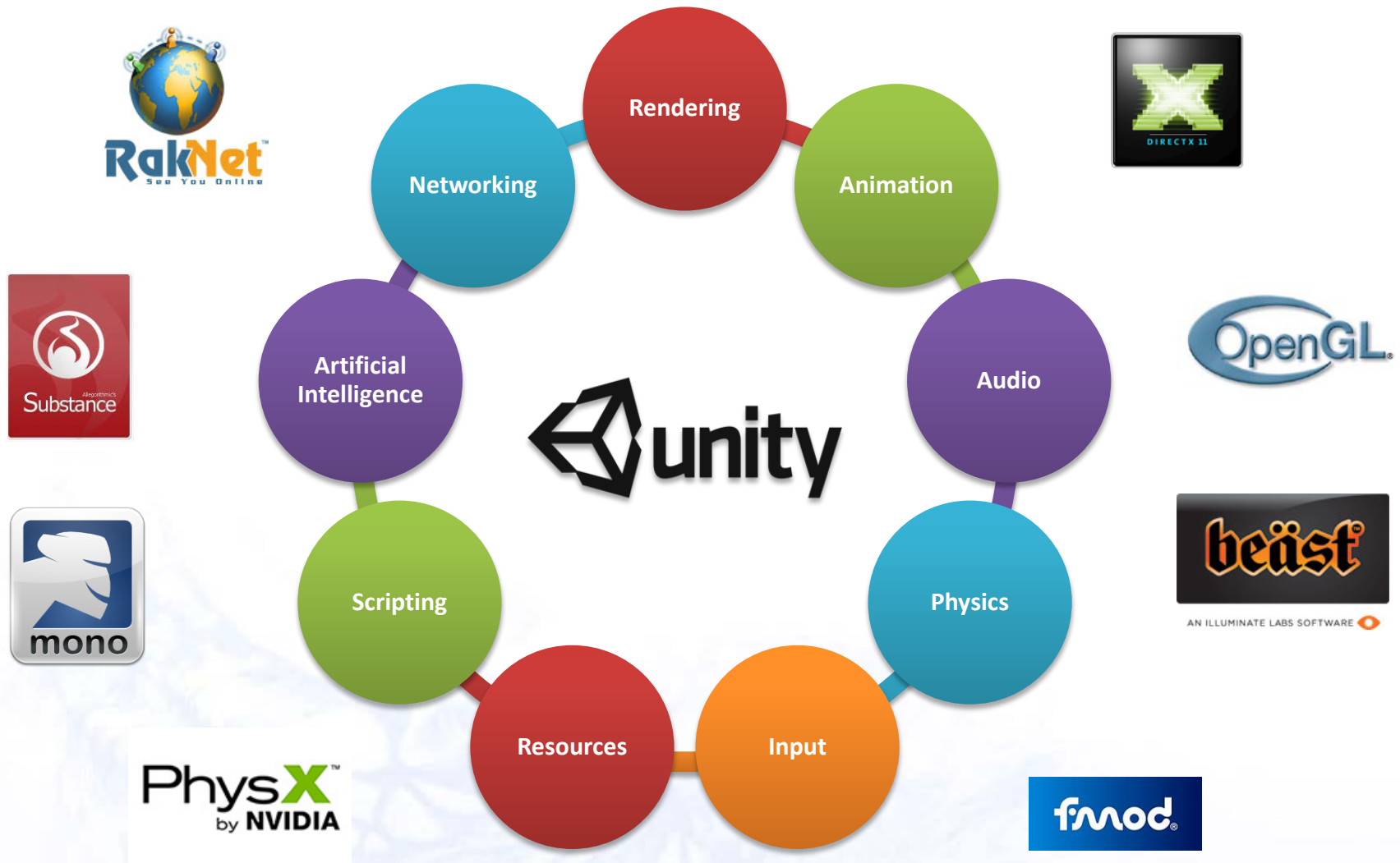
INTRODUCTION TO UNITY



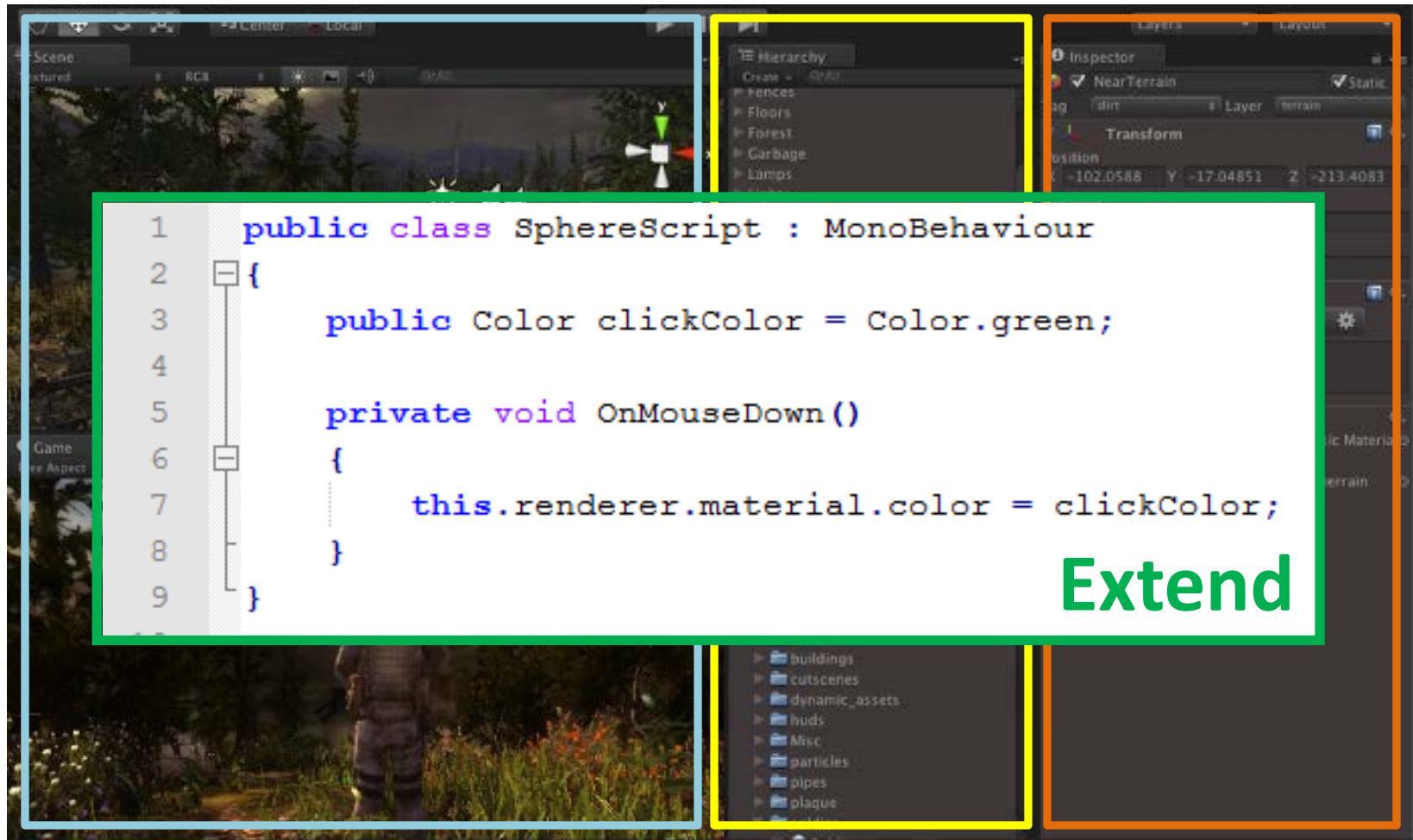
What is Unity?

- **Game engine** – system designed to help create video games
 - Easier & Faster
- **Visual editor** – see changes in real-time
 - Interactive & Rapid prototyping
- **Component-based** – functionality built out of smaller pieces
 - Modular & Extensible

What can Unity do for you?



What does Unity look like?



Unity games run everywhere



Unity understands you



Multiple programming languages

JavaScript

```
var explosion : Transform;

function OnCollisionEnter() {
    Destroy(gameObject);
    Instantiate(explosion, transform.position, transform.rotation);
}
```

C#

```
using UnityEngine;
using System.Collections;

public class Example : MonoBehaviour {
    public Transform explosion;

    void OnCollisionEnter() {
        Destroy(gameObject);
        Instantiate(explosion, transform.position, transform.rotation);
    }
}
```



Boo

```
import UnityEngine
import System.Collections

class Example(MonoBehaviour):

    public explosion as Transform

    def OnCollisionEnter():
        Destroy(gameObject)
        Instantiate(explosion, transform.position, transform.rotation)
```

What about 2D games?



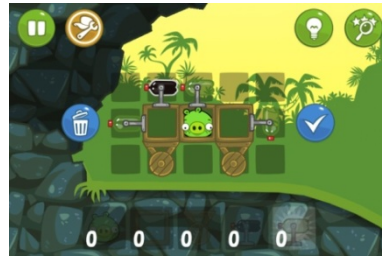
REALM

<http://www.therealmgame.com/>

Games created with Unity



Beat Sneak
Bandit



Bad Piggies



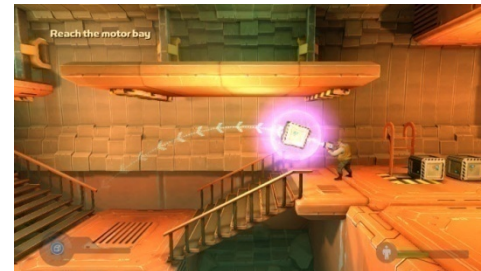
The Room



Temple
Run 2



Scrolls



Rochard



Thomas was Alone



CSR Racing



Dead Trigger 2

How to get Unity?

Unity Basic

- Has every essential features such as graphics, audio, physics, animation, networking, input, and scripting
- Free (with splashscreen)

Unity Pro

- Advanced graphics, audio, animation, and artificial Intelligence
- \$1.500+

Download from <http://unity3d.com>



Part 2

CONCEPTS AND WORKFLOW



Architecture

Game



```
graph TD; Game[Game] --- Assets[Assets]; Game --- Scenes[Scenes]; Assets --> Components[Components]; Scenes --- GO[Game Objects]; GO --- Components;
```

The diagram illustrates the architecture of a game. At the top is a blue box labeled 'Game'. Below it are two red boxes: 'Assets' on the left and 'Scenes' on the right. A large red arrow points from 'Assets' down and then right to a purple box labeled 'Components'. To the right of 'Assets' and below 'Scenes' is a green box labeled 'Game Objects'. Below 'Game Objects' is another purple box labeled 'Components'. The 'Components' box is shared by both 'Assets' and 'Game Objects'.

Assets

Scenes

Game Objects

Components

Editor Interface



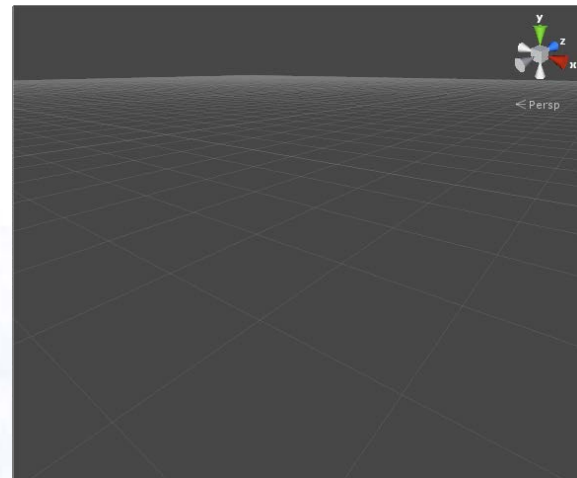
Assets

```
public class SphereScript : MonoBehaviour  
{  
    public Color clickColor = Color.green;  
  
    private void OnMouseDown()  
    {  
        this.renderer.material.color = clickColor;  
    }  
}
```



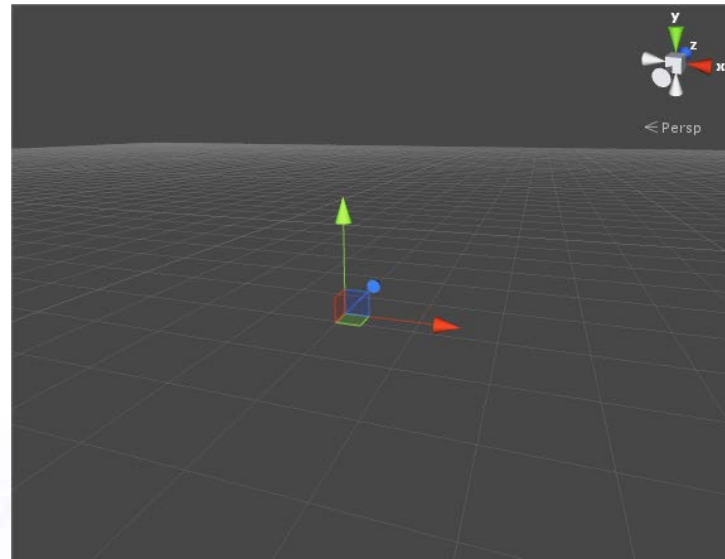
Scene

- Unity games are divided into **scenes**
- **Scenes** are empty spaces...
- ...that can be filled with **game objects**



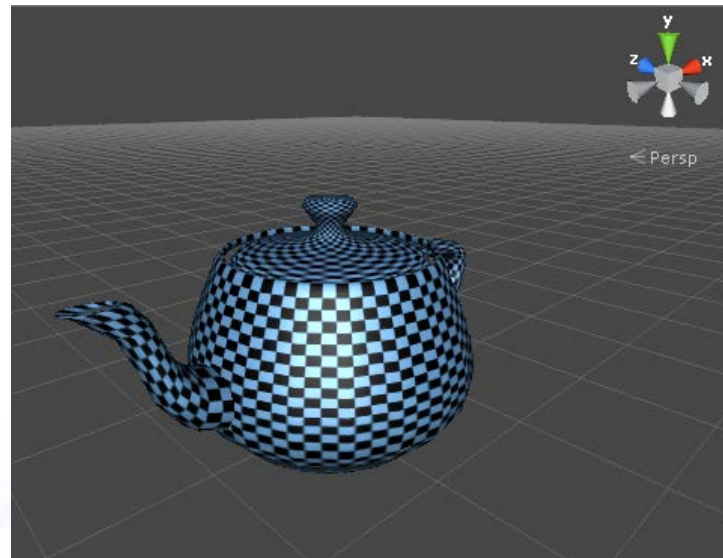
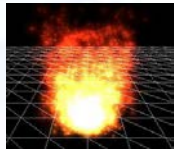
Game Object

- Everything inside a **scene** is a **game object**
- **Game objects** also start out empty and do nothing...



Game Object

- ...but by adding **components** to them they can become anything!



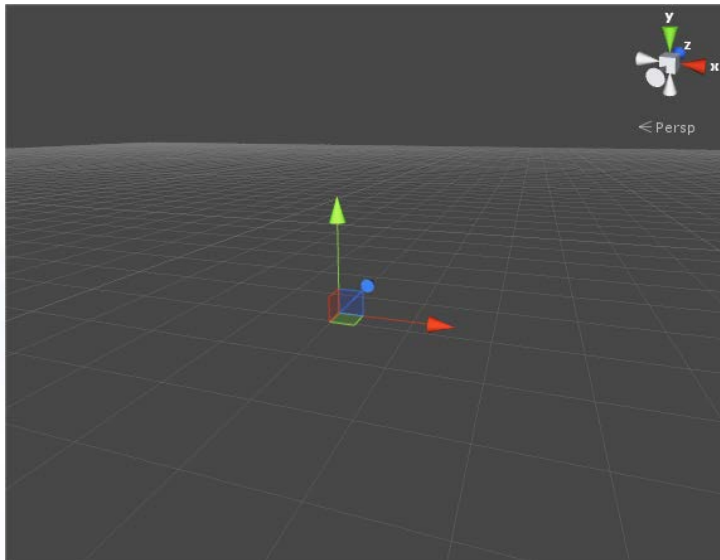
Component

- Each **component** adds a piece of functionality to the **game object**
- The combination of all **components** defines what the **game object** is

Let's see some examples!

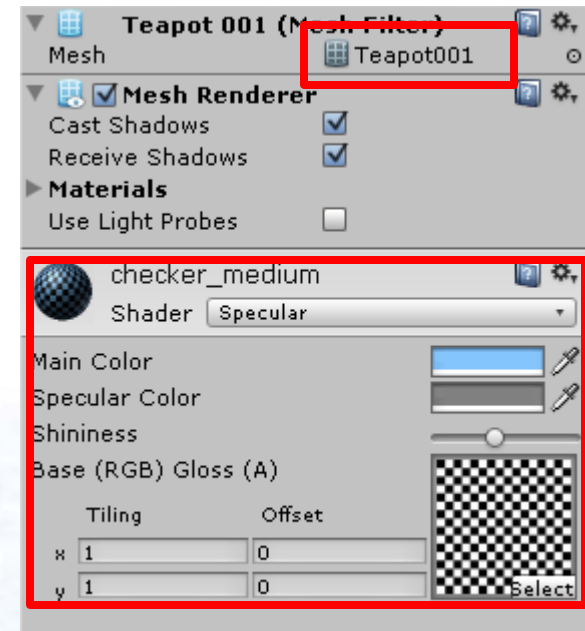
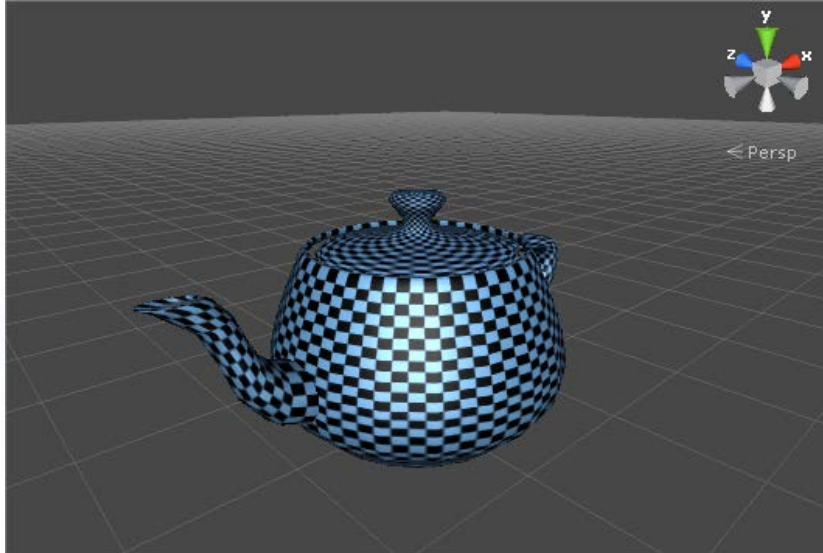
The Transform Component

- Where?
- Which direction?
- How large?



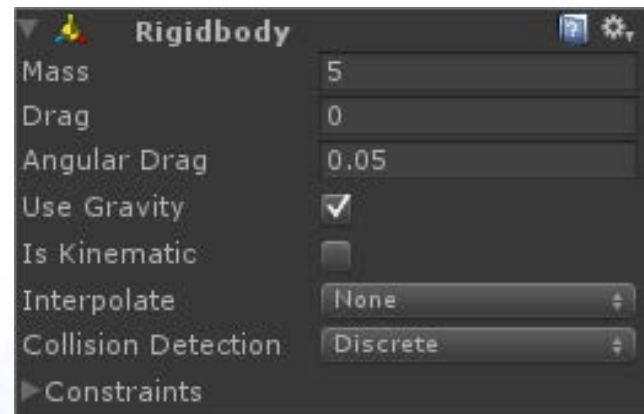
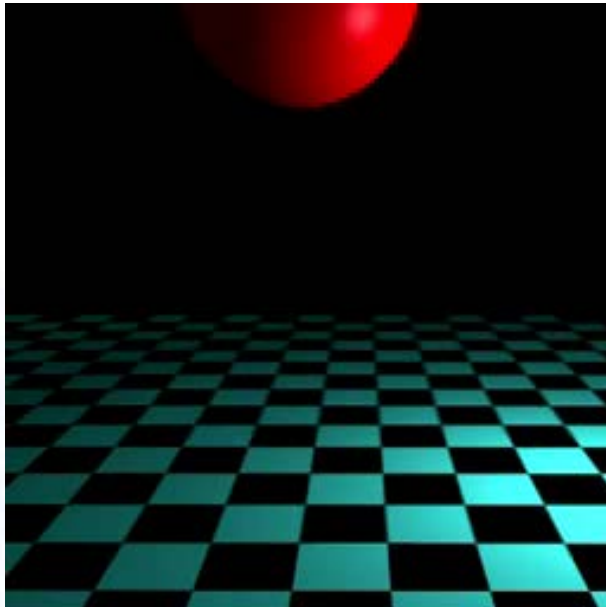
Rendering Components

- What to draw? (mesh filter)
- How to draw? (mesh renderer)



Physics Components

- Is solid? (collider)
- Moves? (rigid body)



The Script Component

- Adds custom behavior

```
1 public class SphereScript : MonoBehaviour
2 {
3     public Color clickColor = Color.green;
4
5     private void OnMouseDown()
6     {
7         this.renderer.material.color = clickColor;
8     }
9 }
```

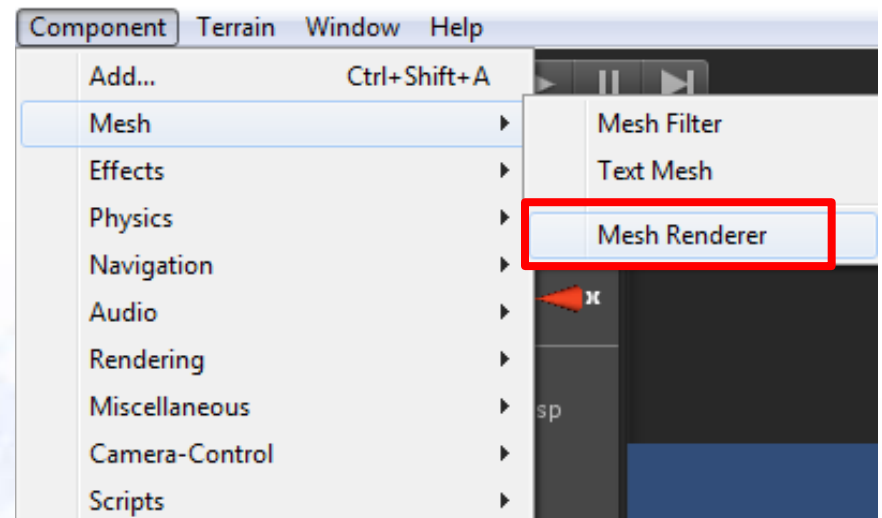
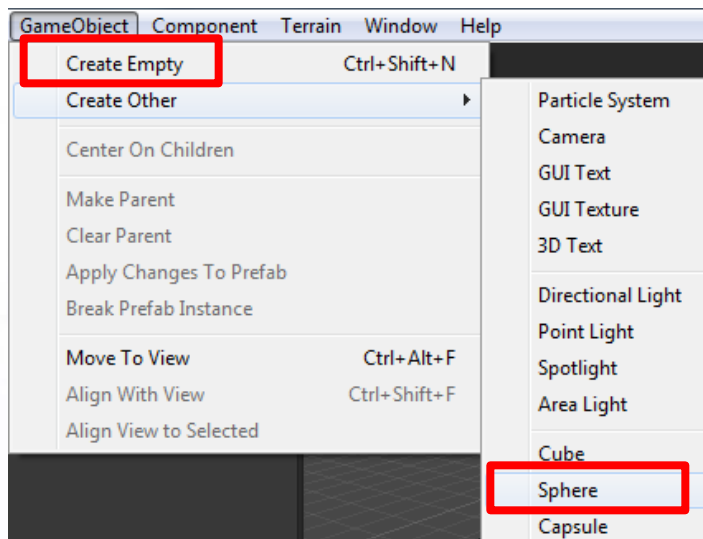


Other Components

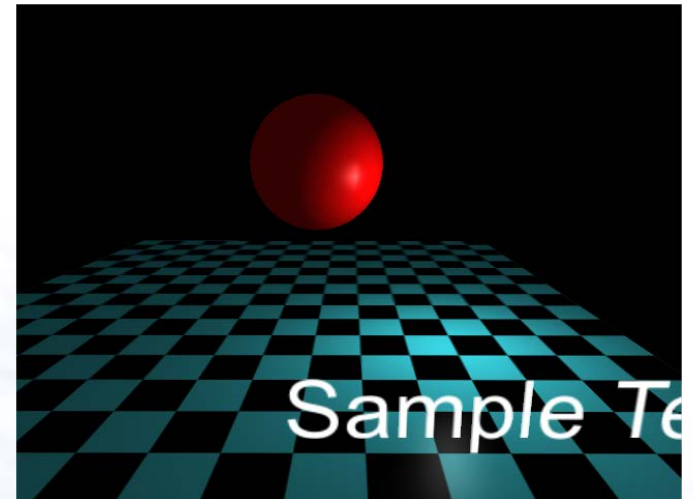
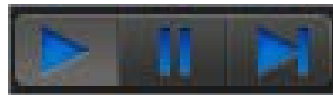
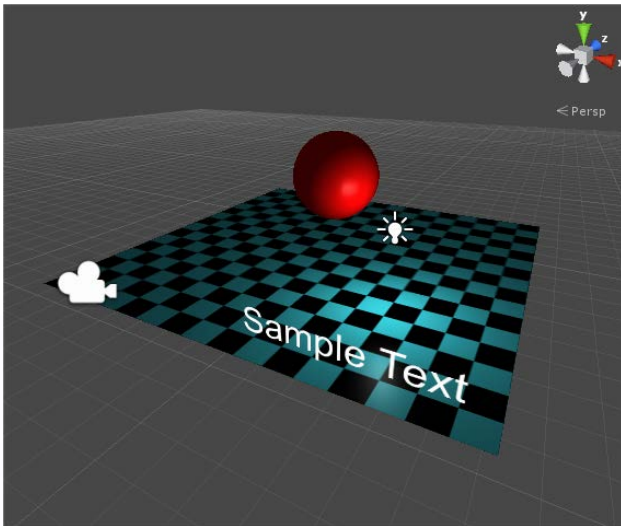
- Light
- Camera
- Text mesh
- Audio listener & source
- Particle system
- Skybox
- ...and many more.

How to create Game Objects

- Create an empty **game object** and manually add **components** to it
- Choose one of the default **game objects**



Game





Part 3

LIVE DEMO

