A Case Study: Unity

Computer Graphics

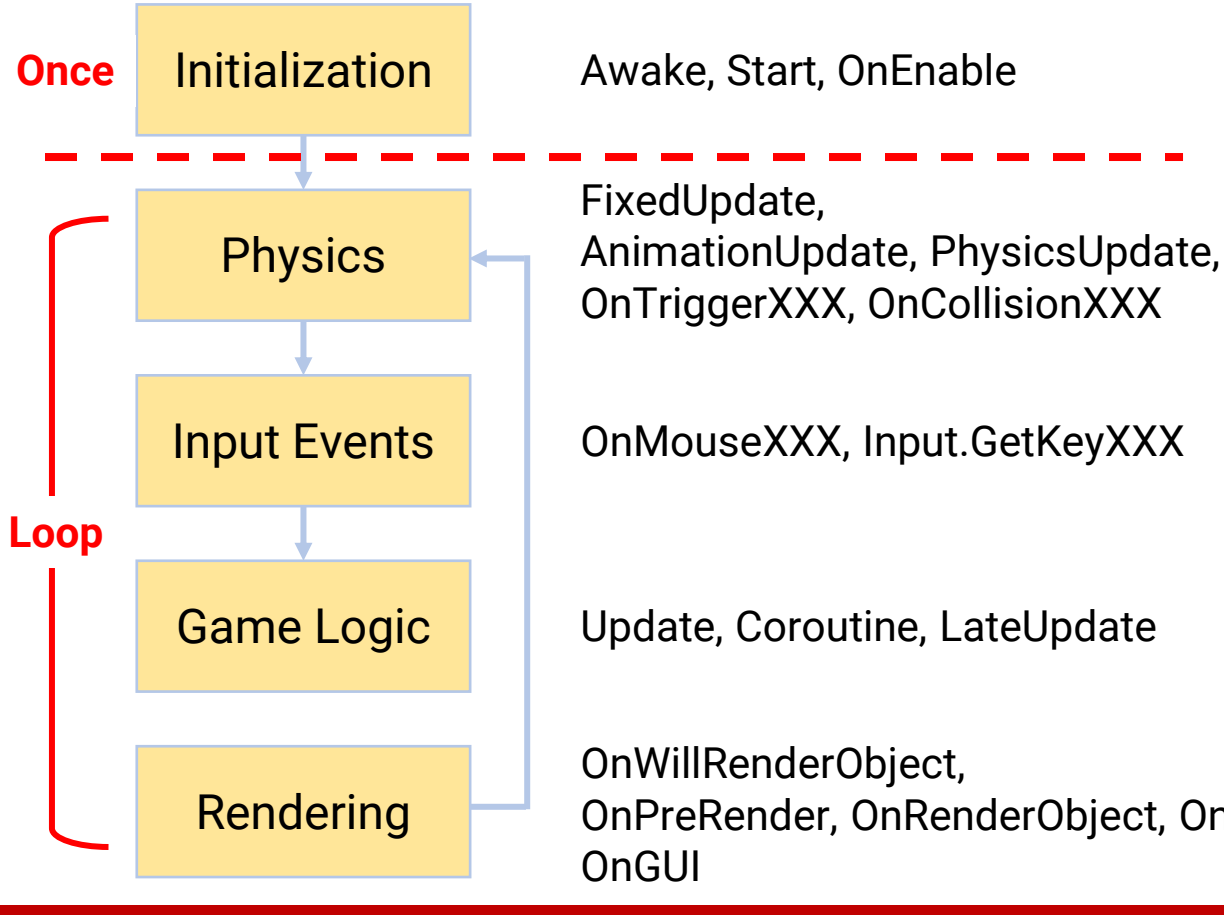
Yu-Ting Wu

Unity Overview

- The most widely used game engine (especially for mobile games) today
- Easier to jump in

Unity Overview (cont.)

- Unity event list order:
  https://docs.unity3d.com/Manual/ExecutionOrder.html



Unity Overview (cont.)

Custom

Built-in

- Component-based (**C# scripts**)

Scene (Level)

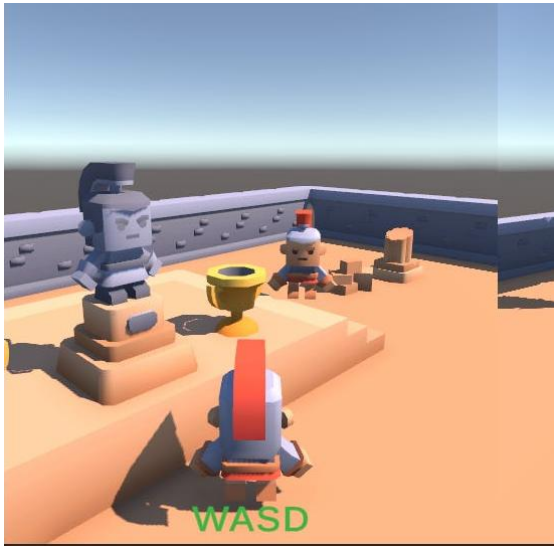
Level Manager

Objects (Actors, Env.)	Objects (Camera)	Objects (Lights)	Objects (Particles)
MeshFilter	Camera	Light	ParticleSystem
MeshRenderer	Behavior1	Behavior1	Renderer
Collider			Behavior1
Rigidbody			
Behavior1			
Behavior2			

Game Manager

Unity Overview (cont.)

- **Custom script**
  - Define the behavior of a game object



Custom

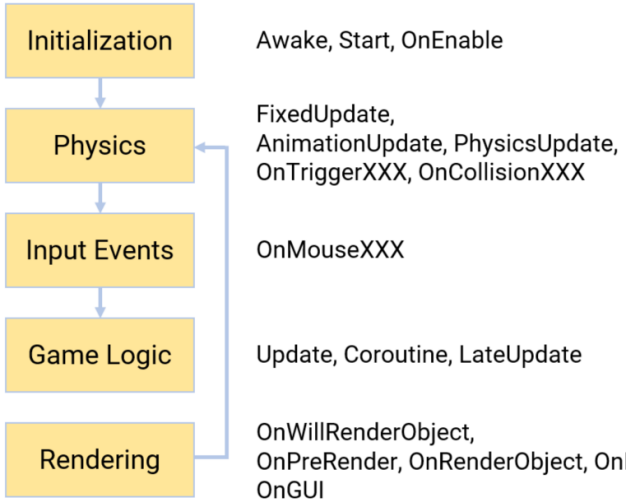
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CharacterMovement1 : MonoBehaviour
{

    public float upmove = 0.0f;
    public float downmove = 0.0f;

    void Update()
    {
        if (Input.GetKeyDown("e") == true){
            downmove -= 0.1f;
            transform.Translate(0.0f, downmove, 0.0f);
        }

        if (Input.GetKeyDown("r") == true){
            upmove += 0.1f;
            transform.Translate(0.0f, upmove, 0.0f);
        }
        transform.Translate(0.0f, 0.0f, Input.GetAxis("Vertical"));
        transform.Rotate(0.0f, Input.GetAxis("Horizontal"),  0.0f);
    }
}
```

float variable for upward movement on y axis

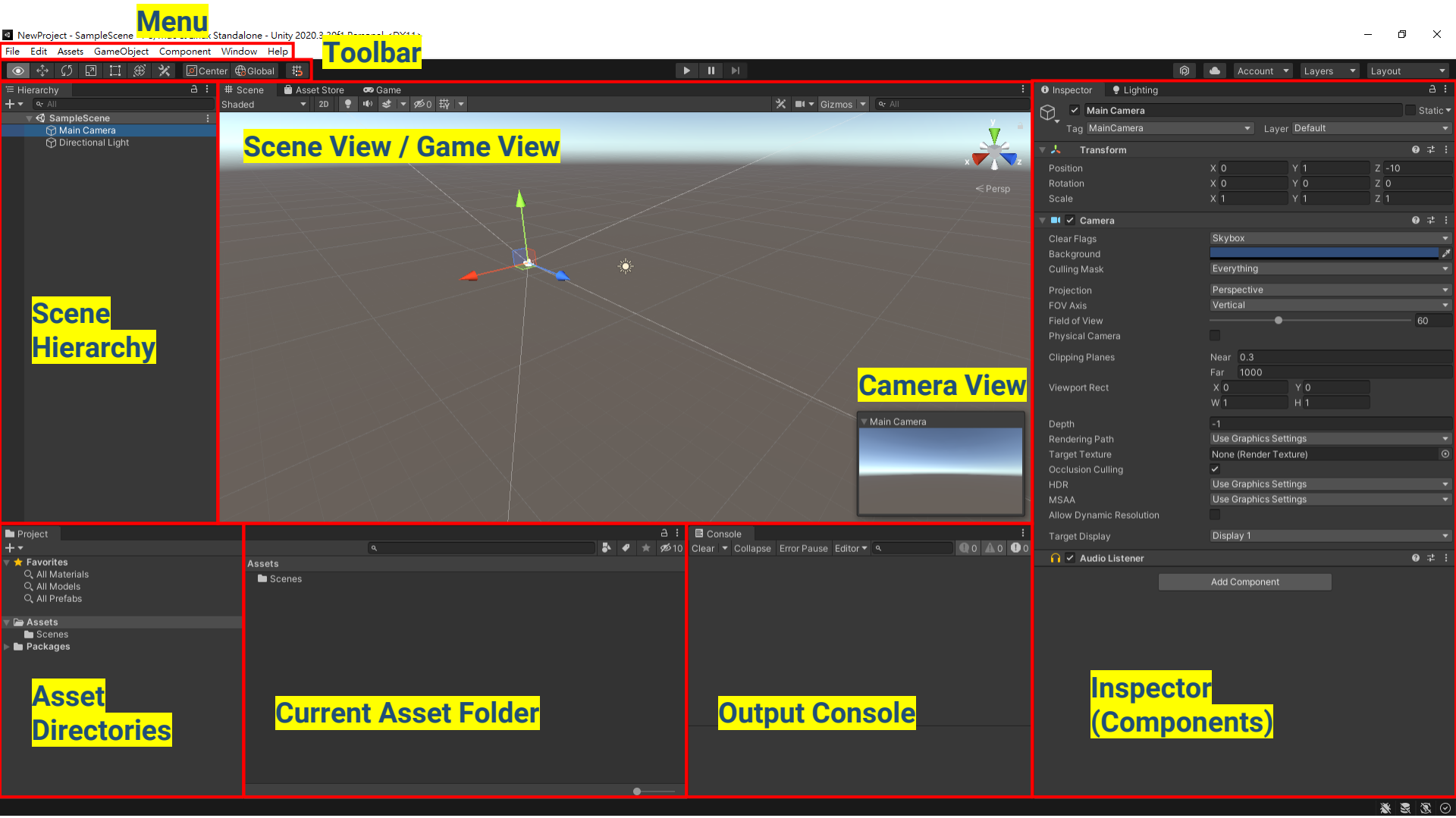
float variable for downward movement on y axis

implements movement on z axis

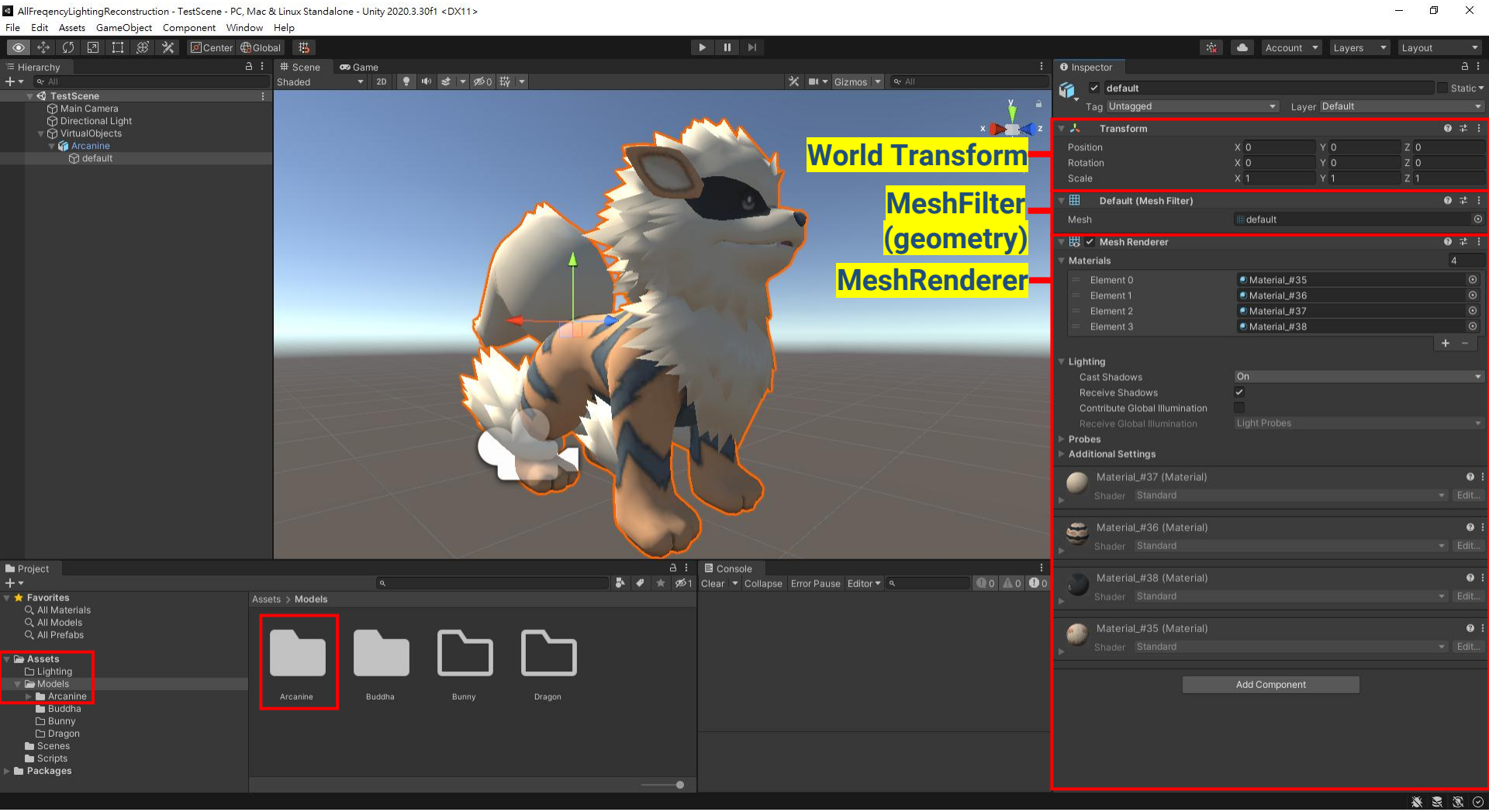
implements rotation on y axis



Unity Editor

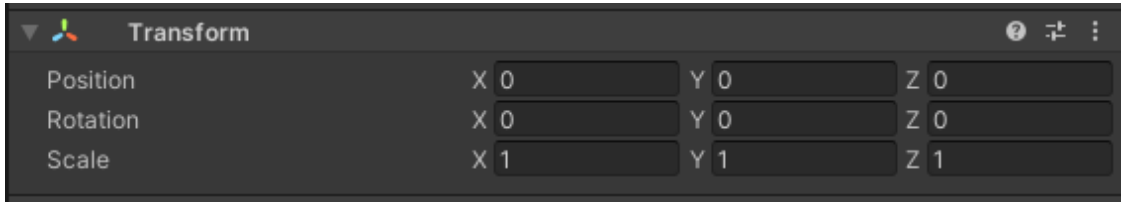


Unity Editor



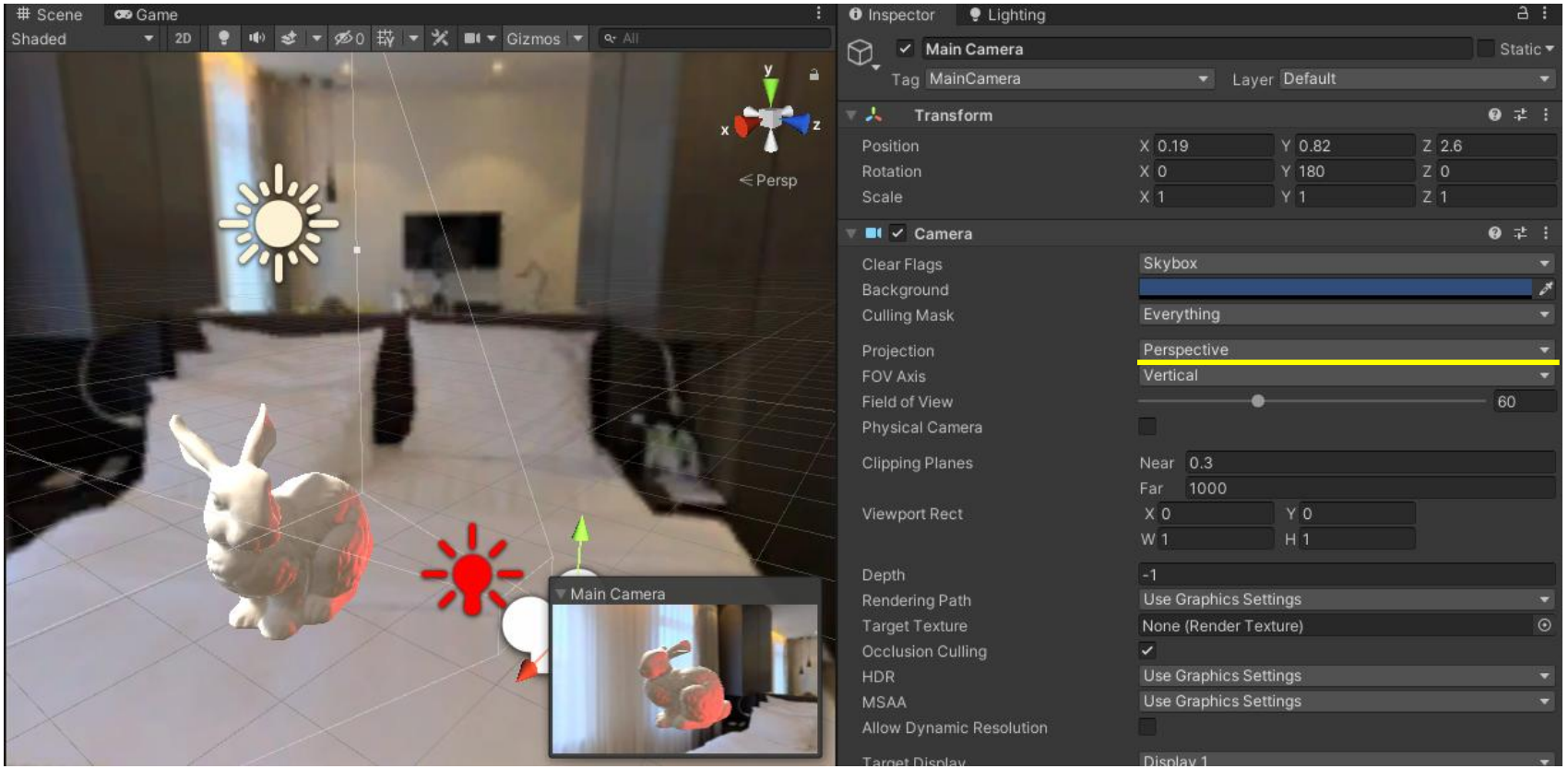
Geometry Data in Unity

- Geometry data in Object Space is described in a **MeshFilter** component
  - **Mesh**
    - vertexBufferTarget / indexBufferTarget
    - vertices (position) / normals / uv(12345678) / tangents
    - triangles (indices)
    - subMeshCount
    - …
- An object is placed in the virtual world by a **World Transform**, described by
  - Position (translation)
  - Rotation
  - Scale



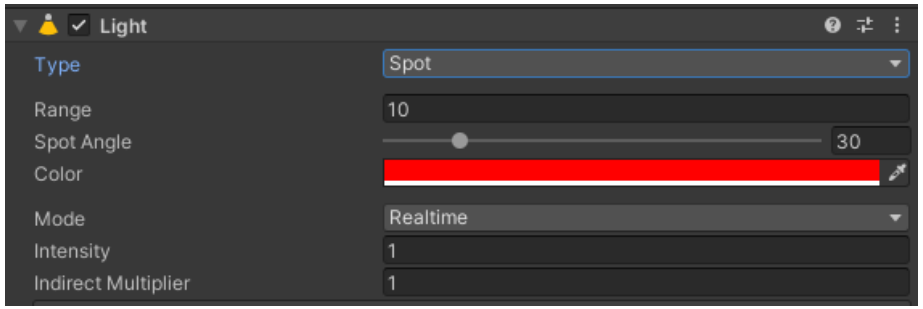
Camera in Unity

- An object that attaches a Camera component will become a camera



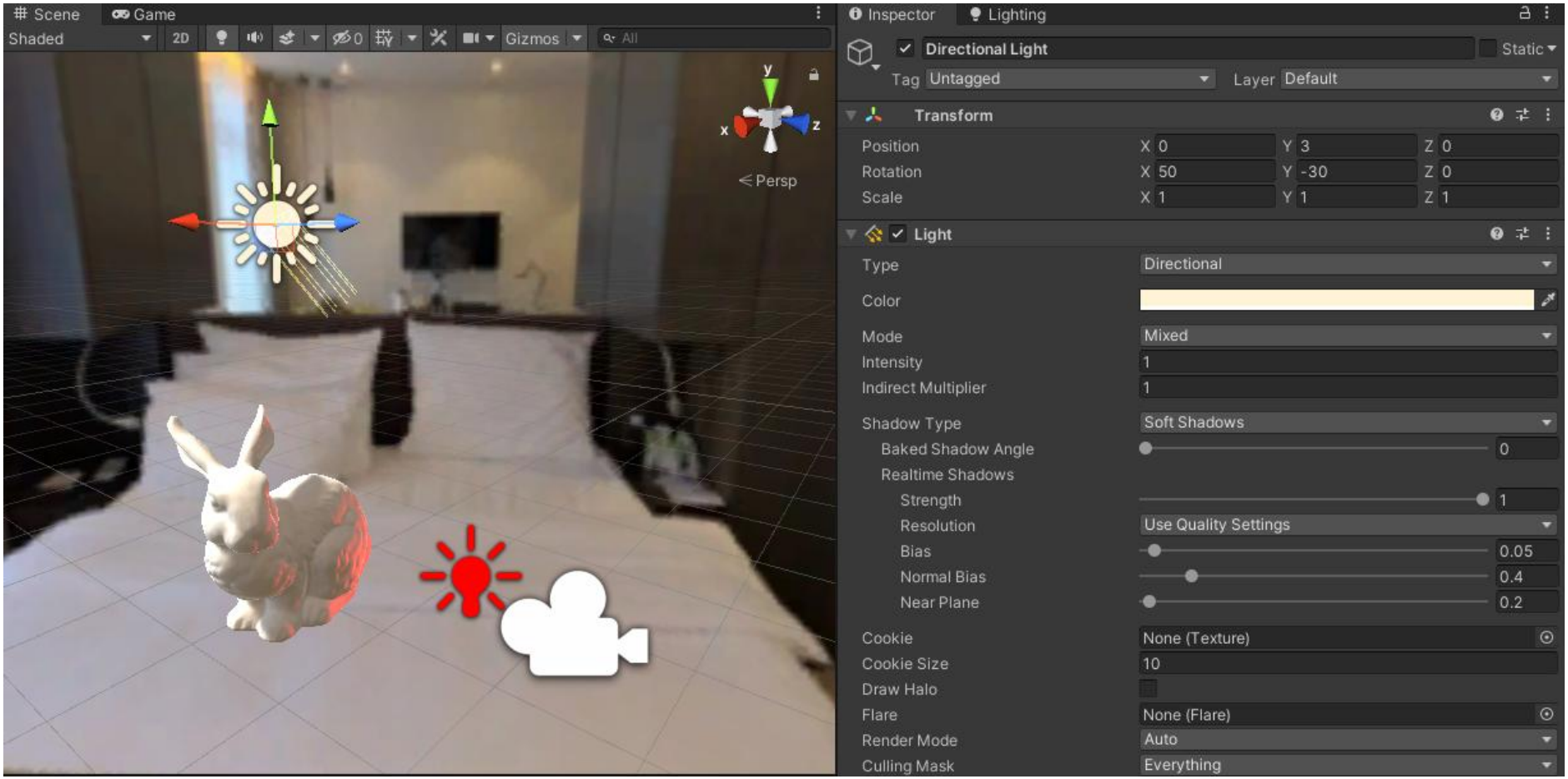
Lights in Unity

- An object that attaches a light component will become a light
- Unity supports several types of lights
  - Directional light
  - Point light
  - Spot light
  - Area light (bake only)
  - Environment light
    (using spherical harmonics)



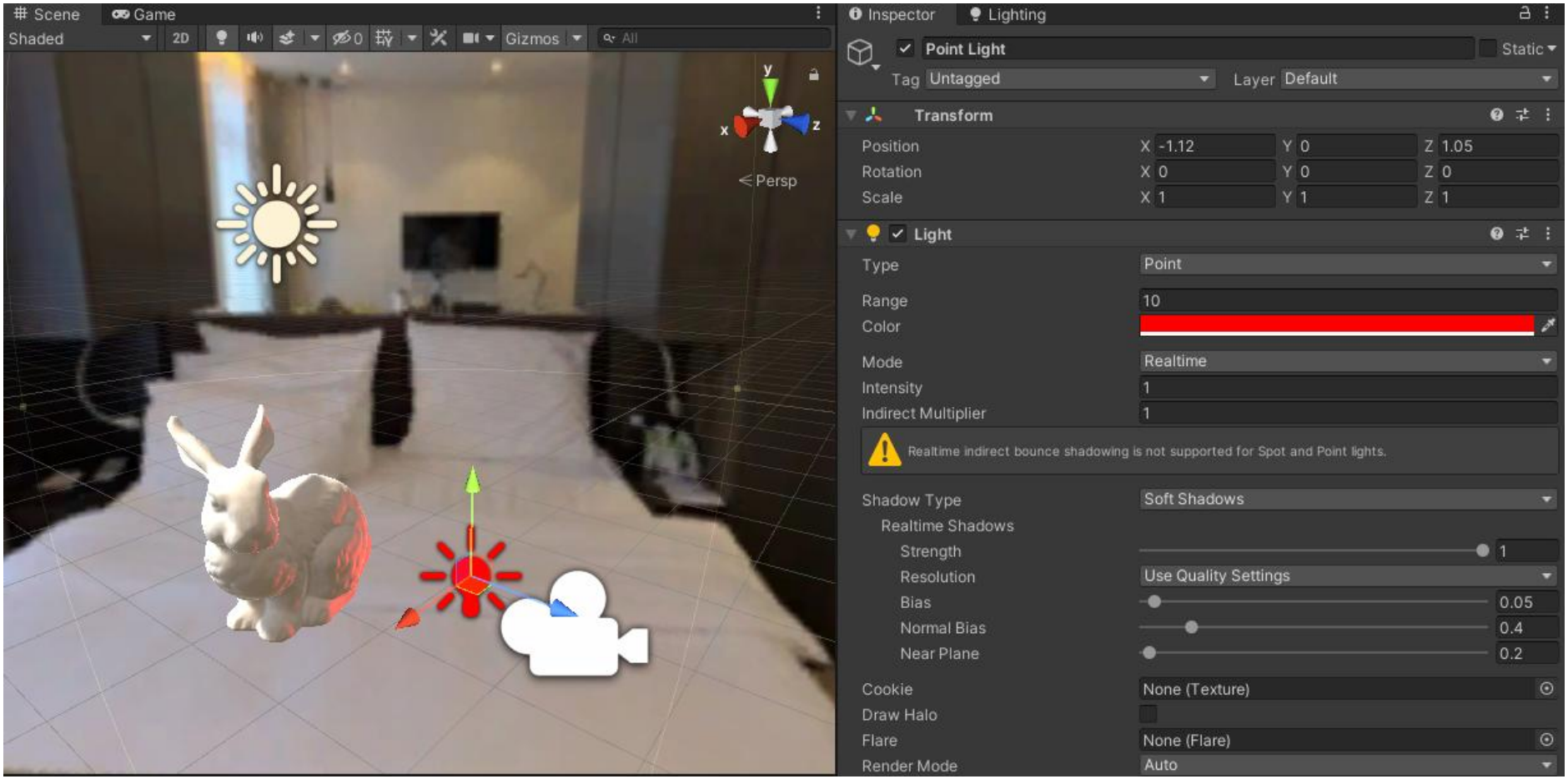
Lights in Unity (cont.)

- Directional light



Lights in Unity (cont.)

- Point light

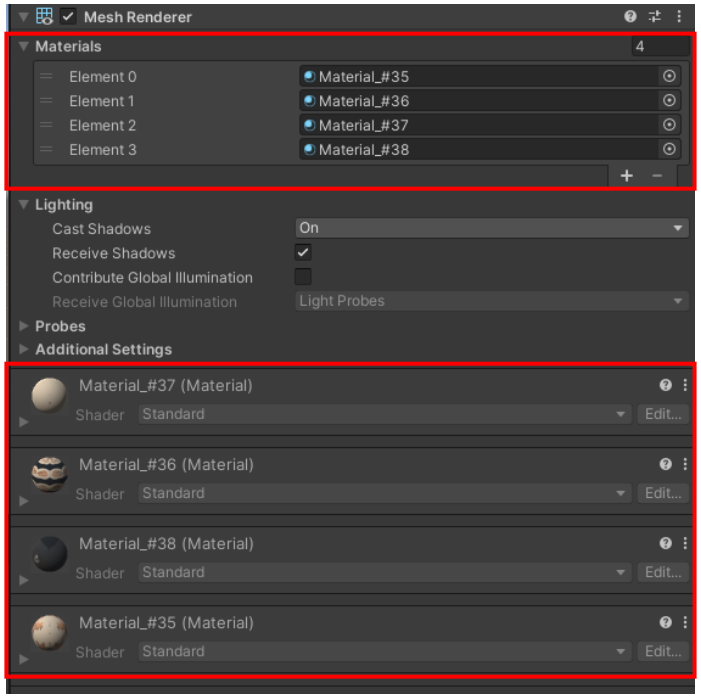


Unity MeshRenderer

- Rendering features are described in a **(Mesh)Renderer** component
  - **Materials**
    - The material of each subMesh
  - **Lighting**
    - Does the object cast/receive shadows?
  - **Probe**
    - Does the object shade with light probes

    (e.g., reflection cubemaps)

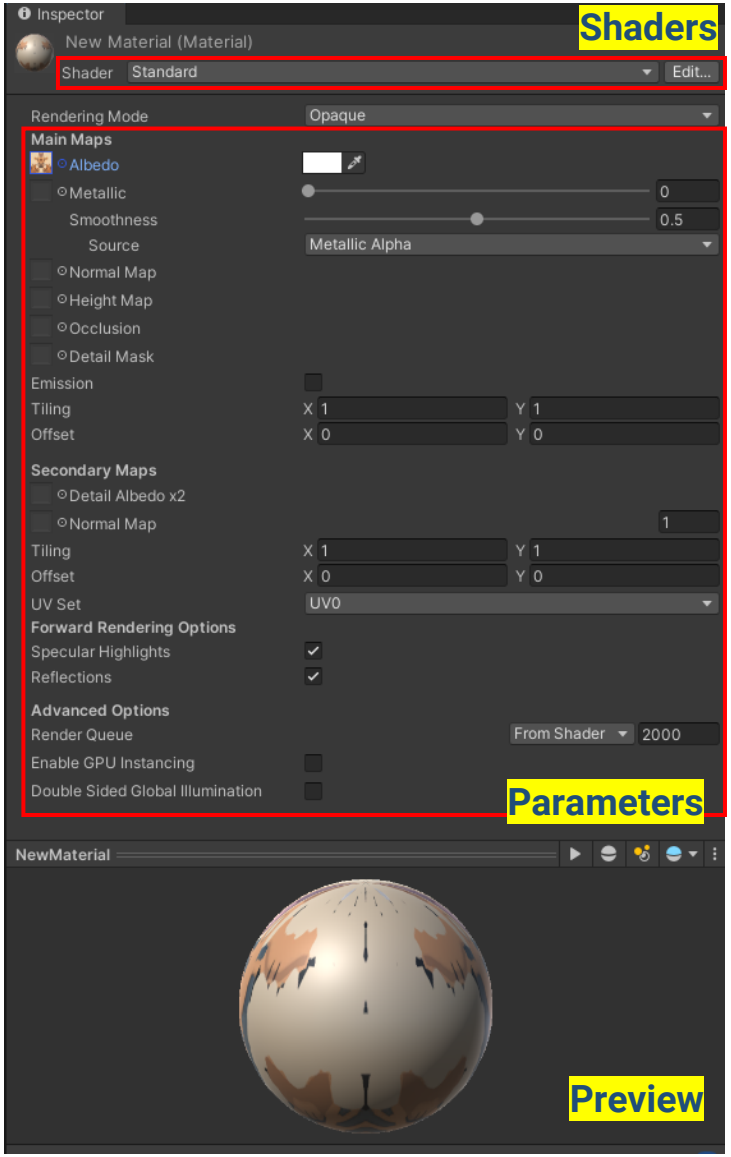
four subMeshes



four materials

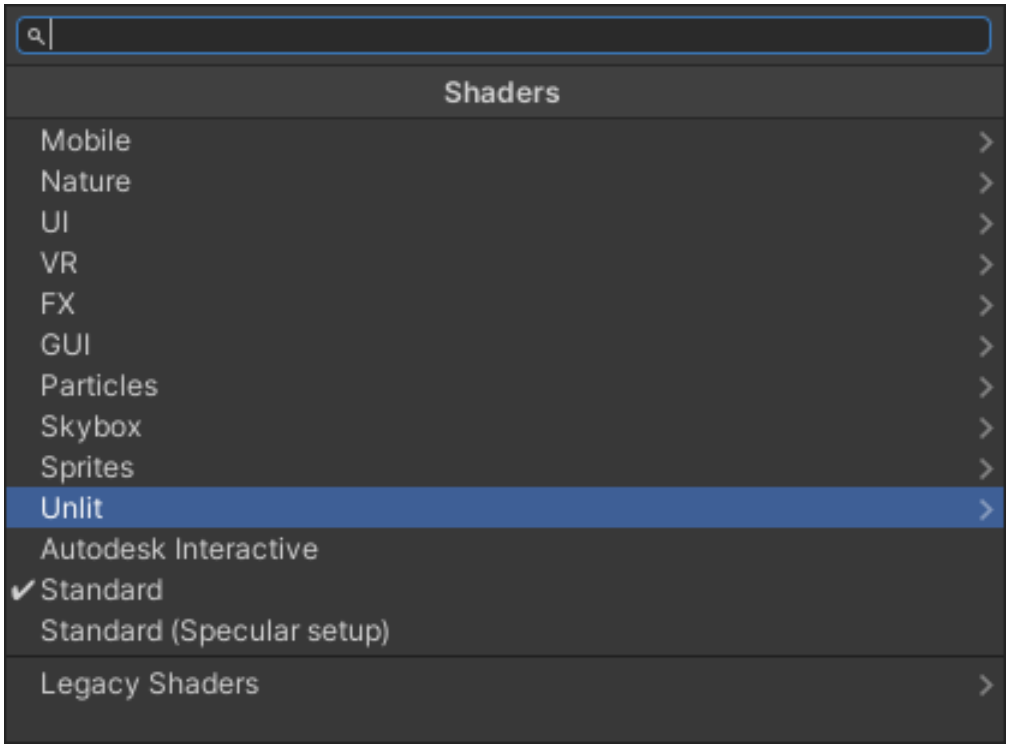
Unity Material

- **Material** = **Shader** + **Parameters**
  - A Unity shader file comprises at least a vertex shader and a fragment shader, and may include a geometry shader or tessellation shader
  - **Shader** defines the way (e.g., math) to transform objects and compute surface color
  - **Shader** also defines a set of parameters



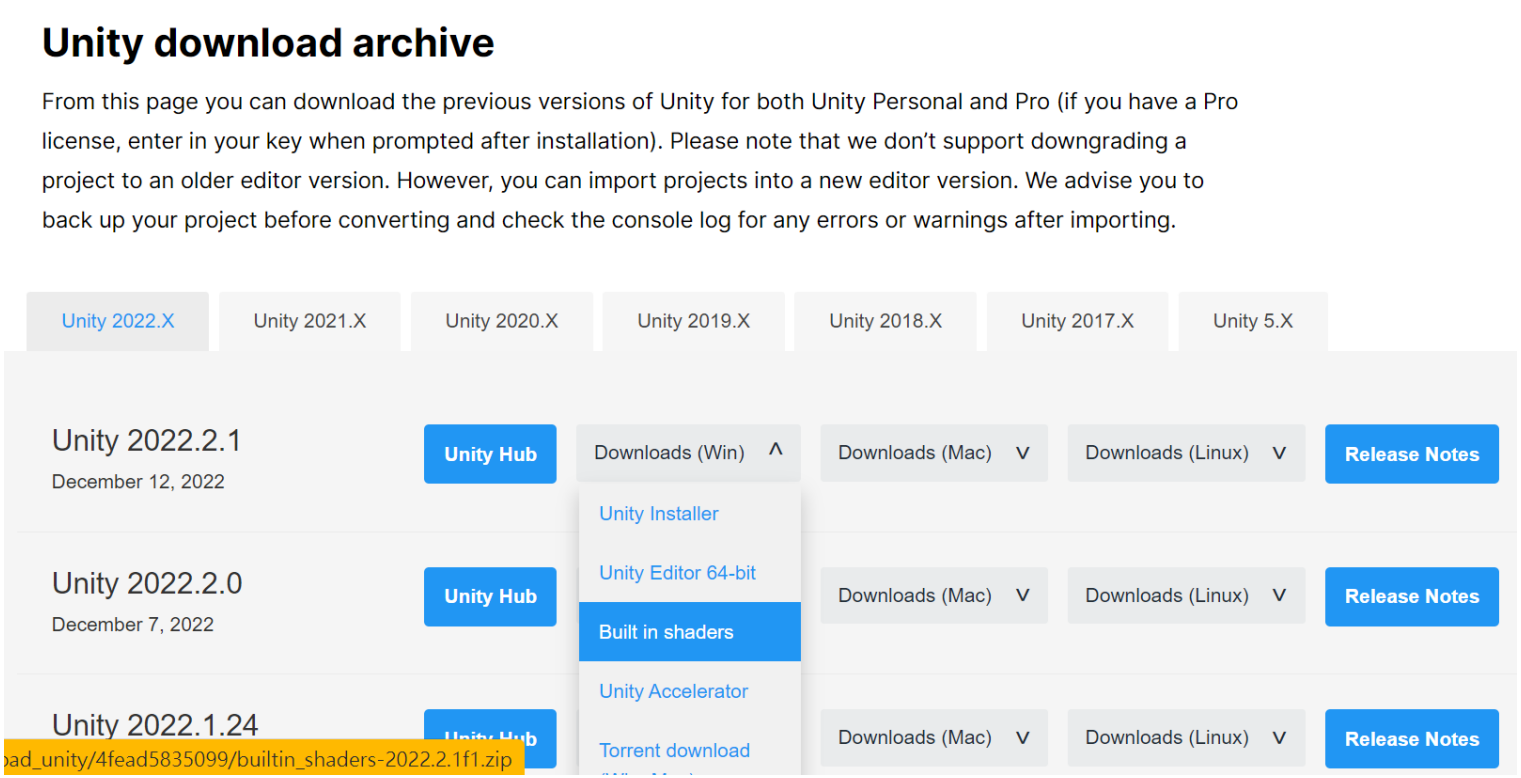
Unity Built-in Shaders

- Unity provides a bunch of built-in shaders
- Developers can also create their own shaders by writing shader code (NVIDIA Cg)



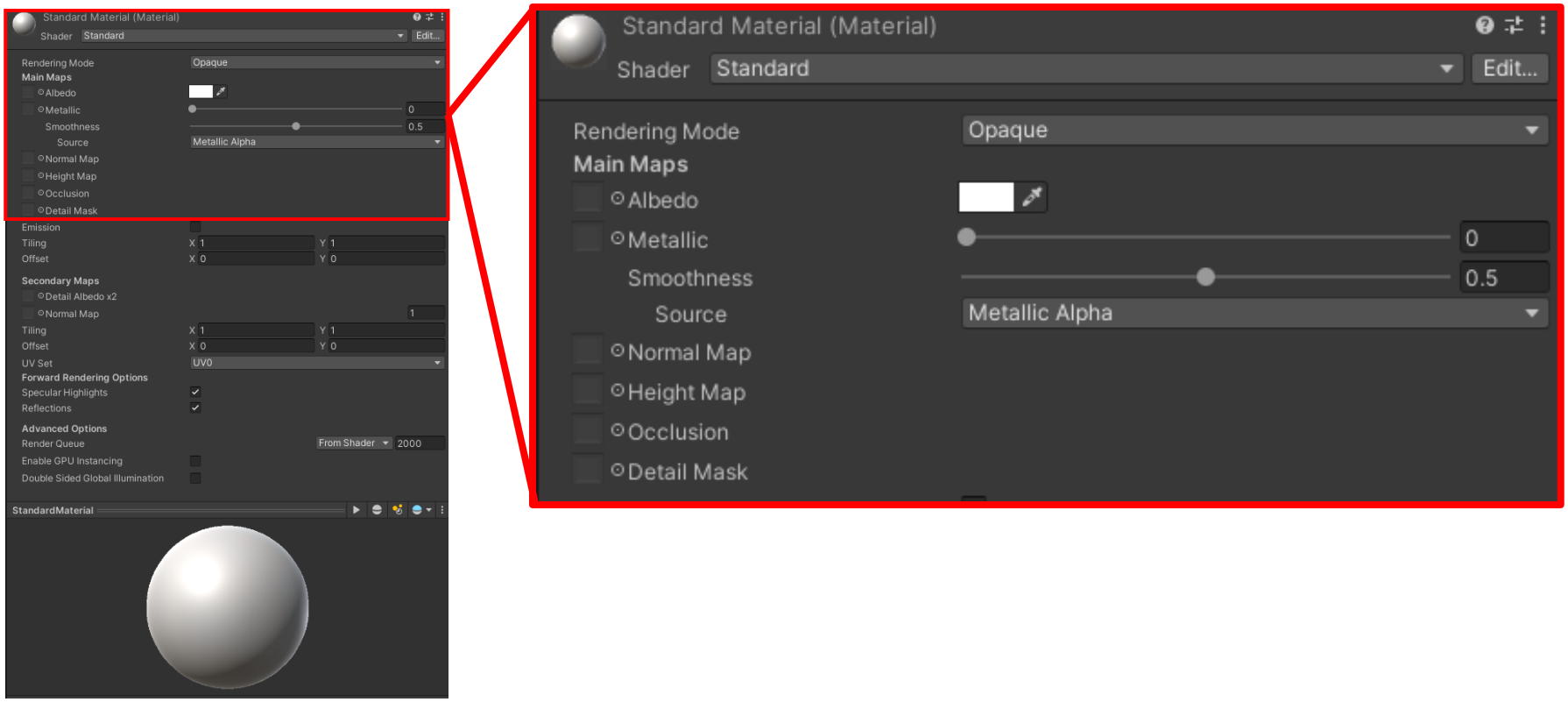
Unity Built-in Shaders (cont.)

- You can download the built-in shaders for reference https://unity.com/releases/editor/archive



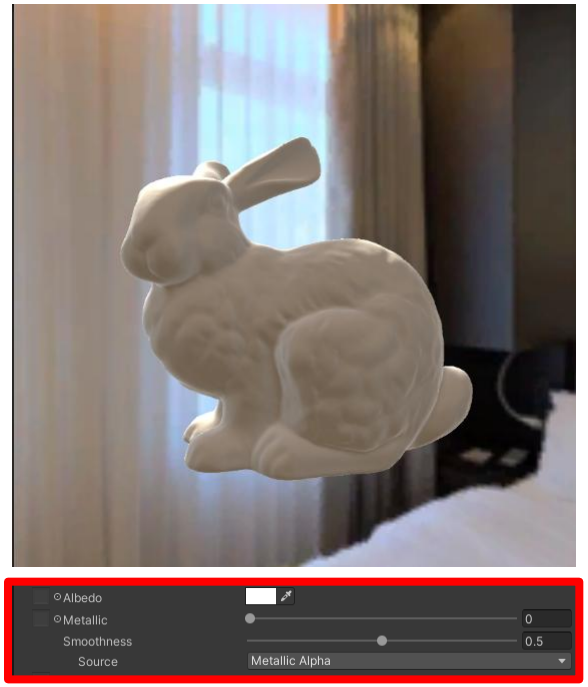
Unity Built-in Shaders (cont.)

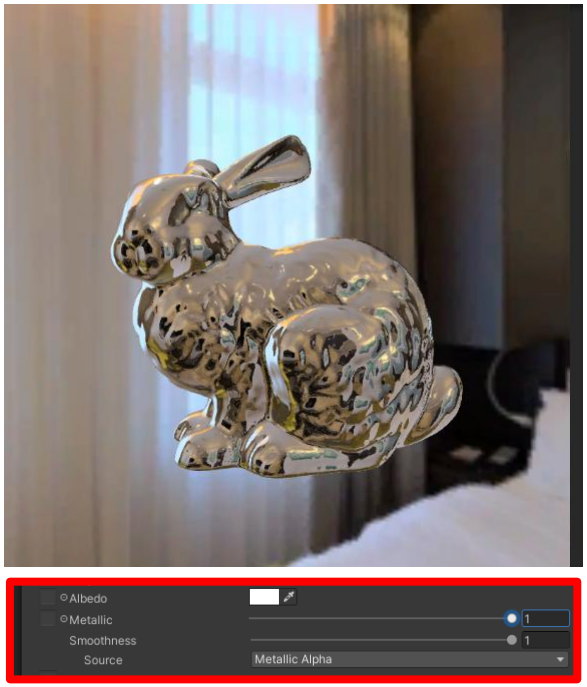
- **Standard shader**
  - You can use the Unity standard shader for most 3D objects
    - A variant of Disney's BRDF model

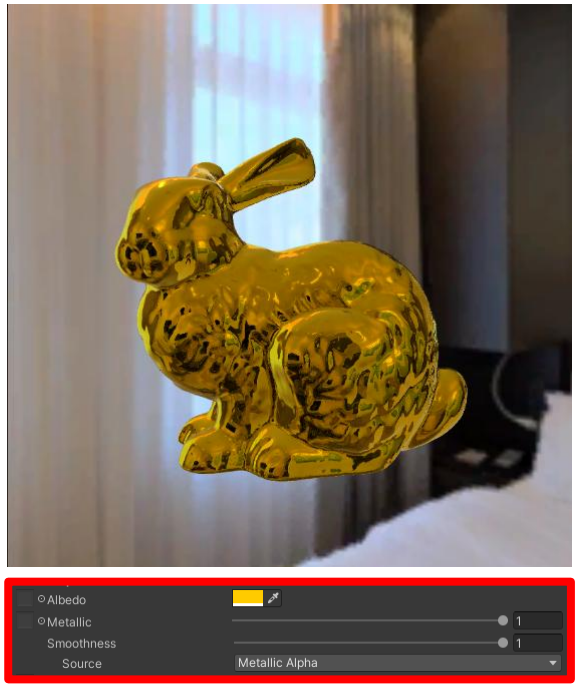


Unity Built-in Shaders (cont.)

- **Standard shader**
  - You can use the Unity standard shader for most 3D objects
    - A variant of Disney's BRDF model







Design of Unity's Rendering System

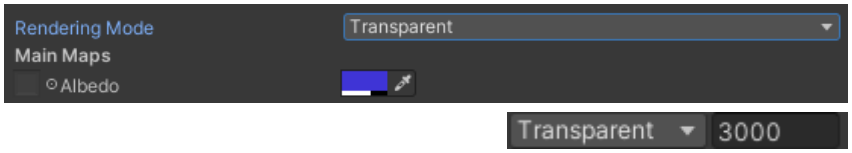
- How does Unity handle **various materials**? For example, with or without an **Albedo** texture

```
Shader "Standard"
{
    Properties
    {
        _Color("Color", Color) = (1,1,1,1)
        _MainTex("Albedo", 2D) = "white" {}
```

For materials that do not use an albedo texture
Unity will create a pure white one,
so the shader code can be unified

Design of Unity's Rendering System (cont.)

- How does Unity handle **transparency**?
  - By defining RenderQueue
    - Background (1000)
    - Geometry (2000)
    - AlphaTest (2450)
    - Transparent (3000)
    - Overlay




END OF PRESENTATION
ANY QUESTIONS?