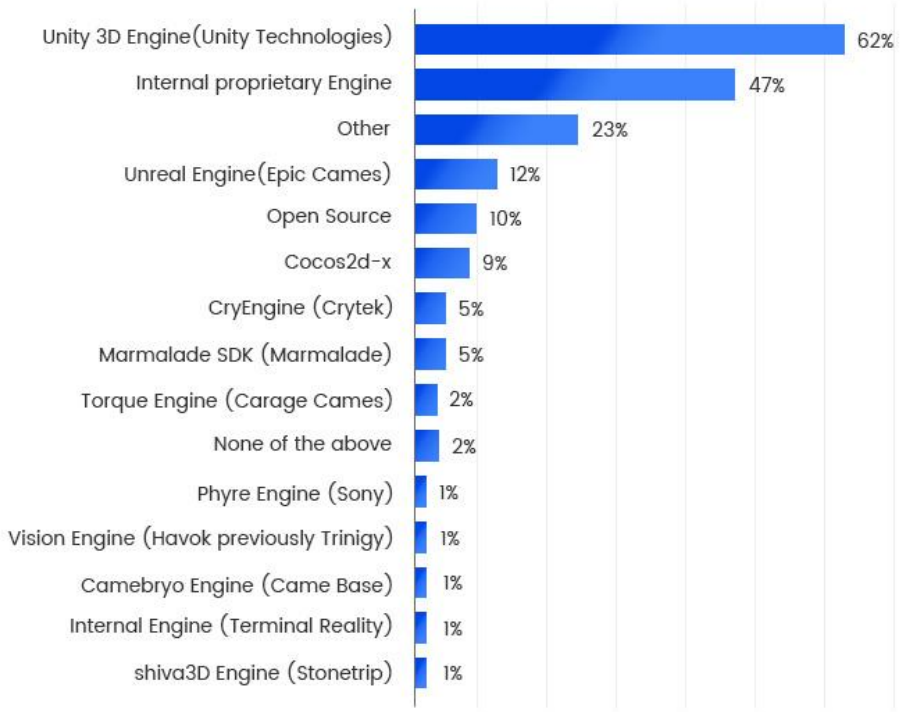
A Case Study: Unity

Computer Graphics

Yu-Ting Wu

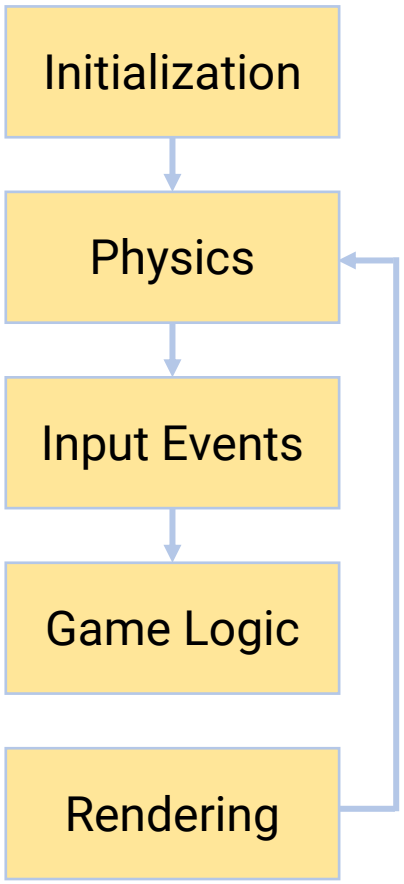
Unity Overview

- 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



Awake, Start, OnEnable

FixedUpdate,
AnimationUpdate, PhysicsUpdate,
OnTriggerXXX, OnCollisionXXX

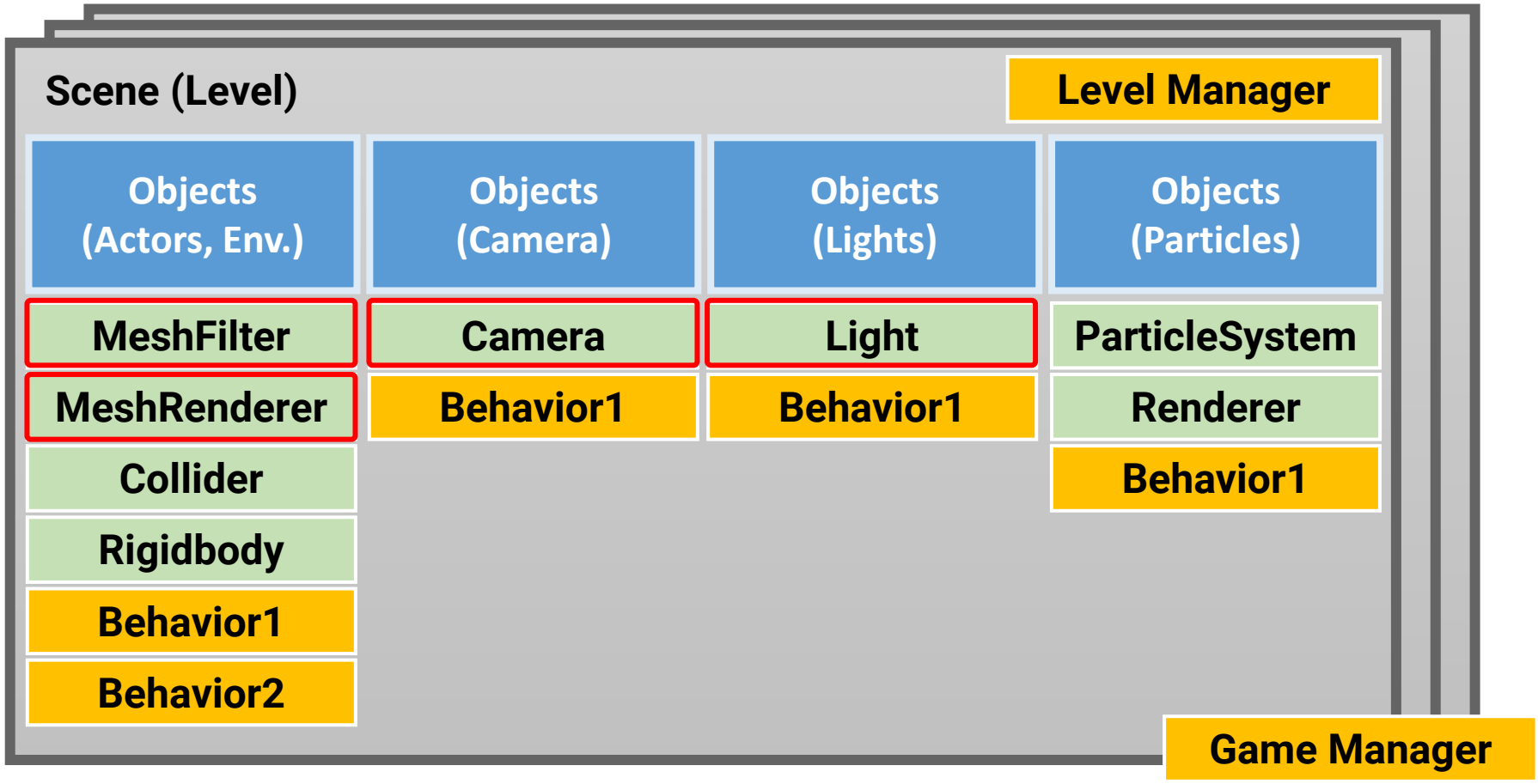
OnMouseXXX

Update, Coroutine, LateUpdate

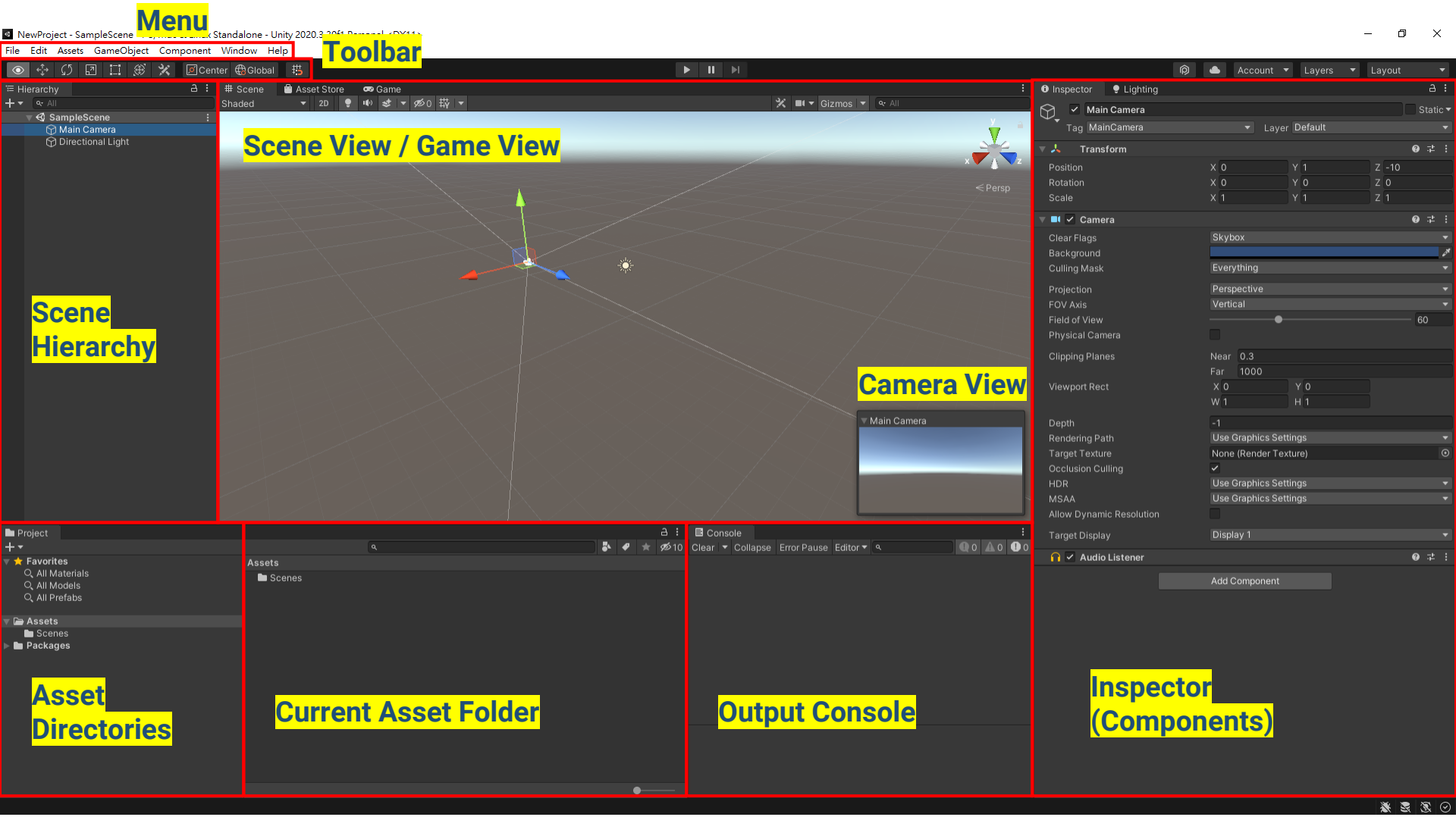
OnWillRenderObject,
OnPreRender, OnRenderObject, OnPostRender, OnRenderImage,
OnGUI

Unity Overview (cont.)

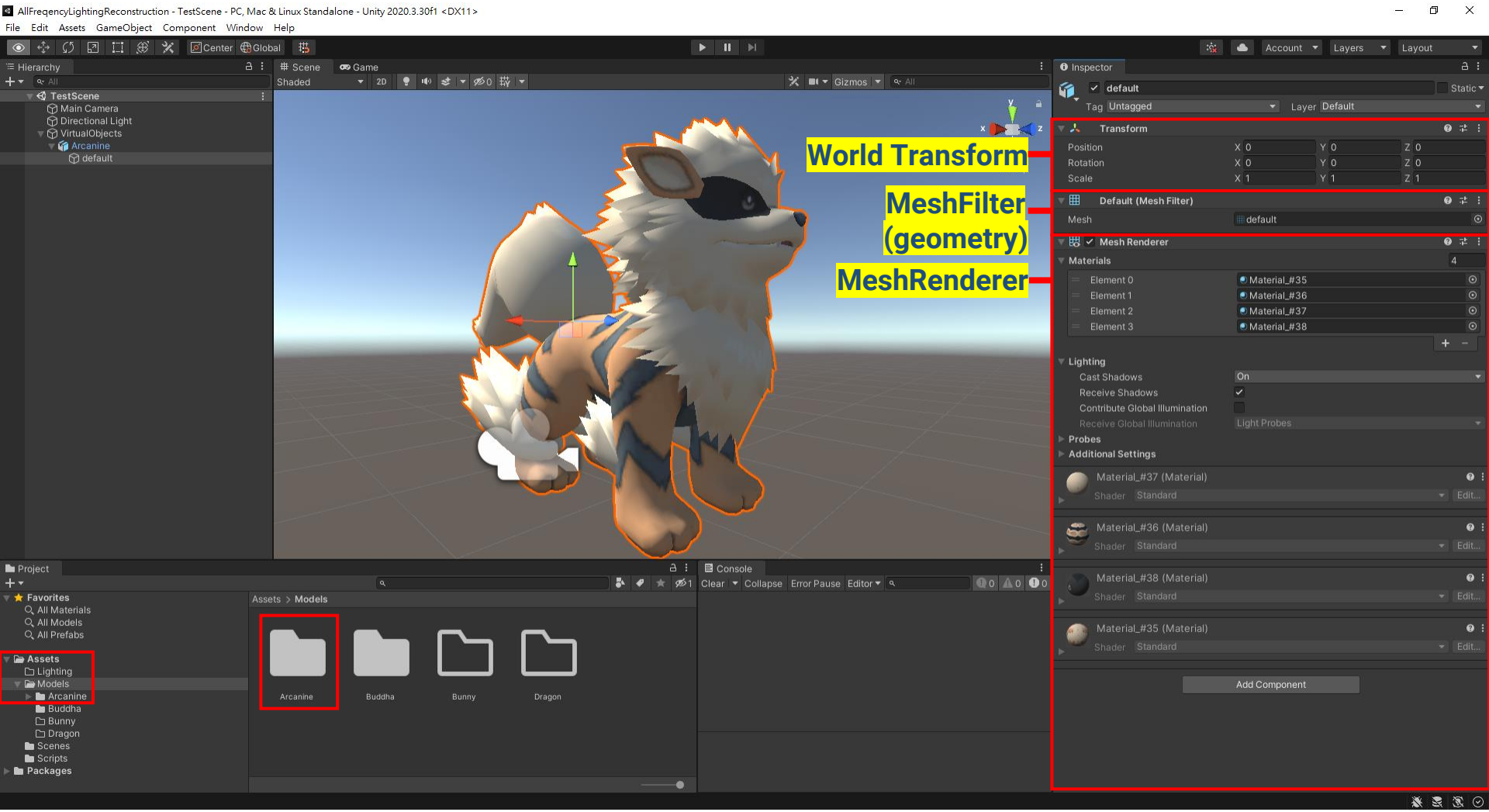
- Component-based (**C# scripts**)



Unity Editor

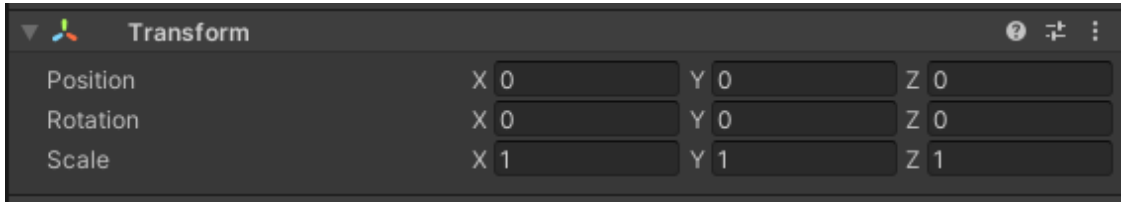


Unity Editor



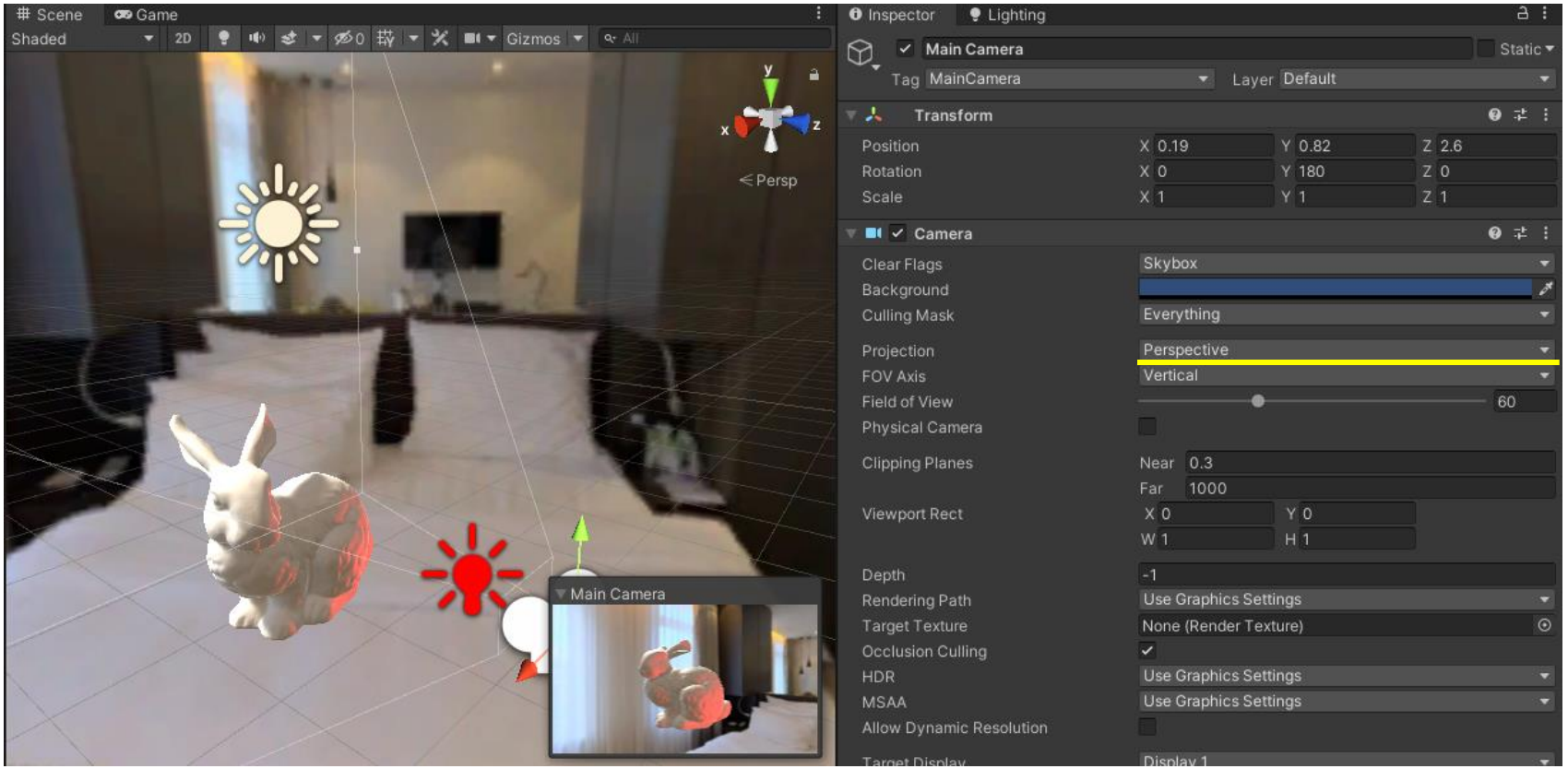
Geometry Data in Unity

- 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



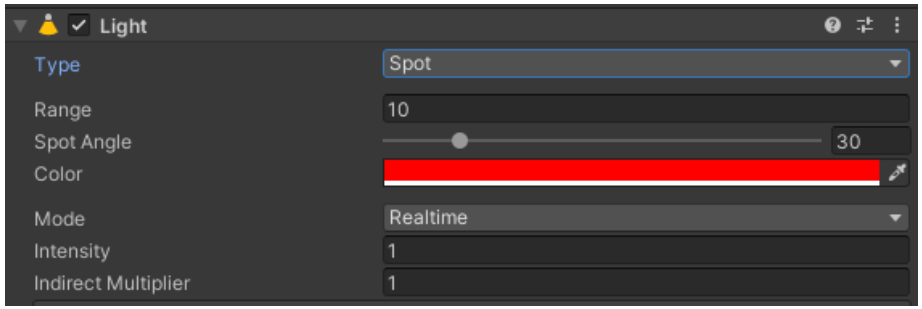
Camera in Unity

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



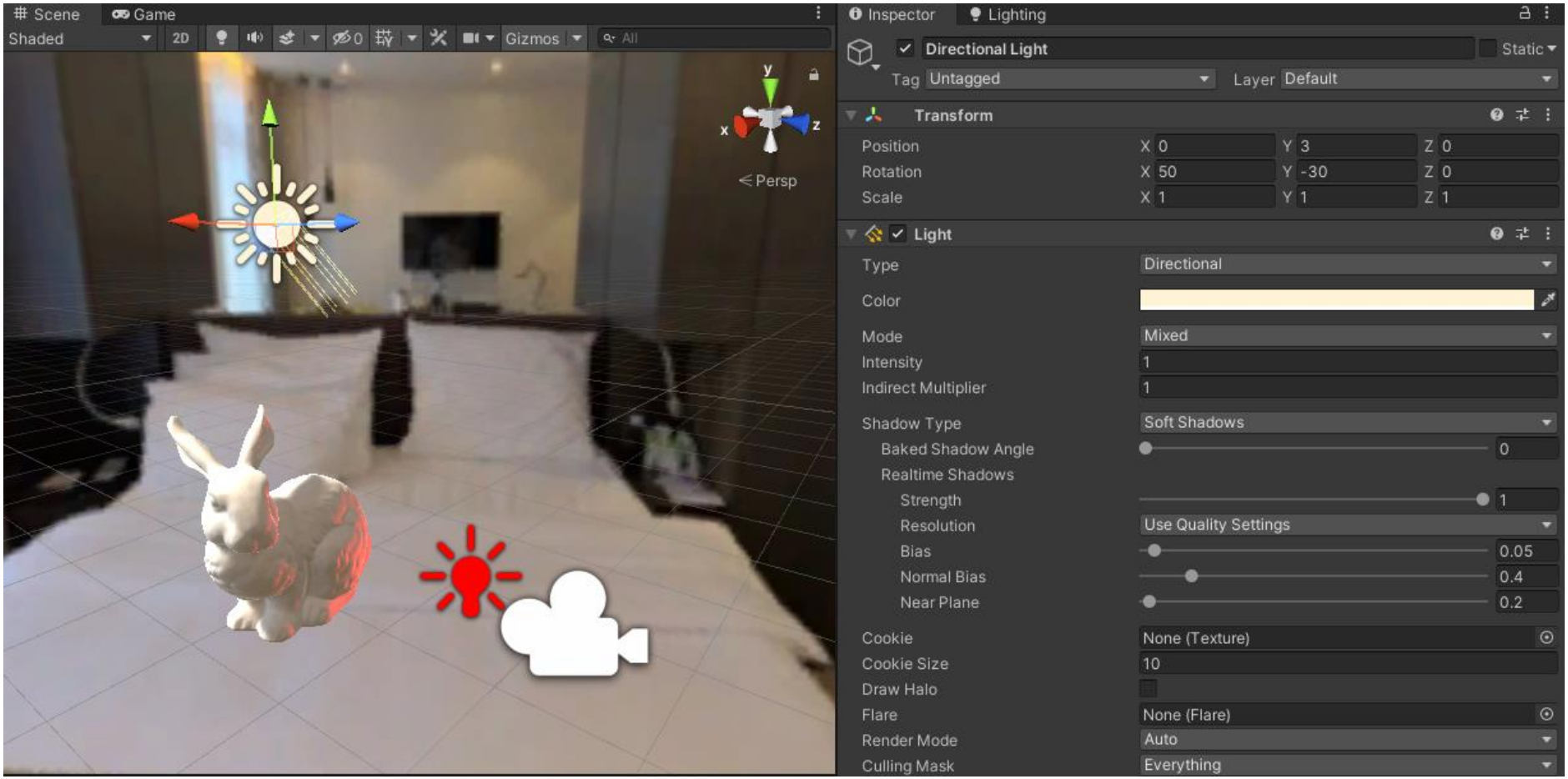
Lights in Unity

- 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)



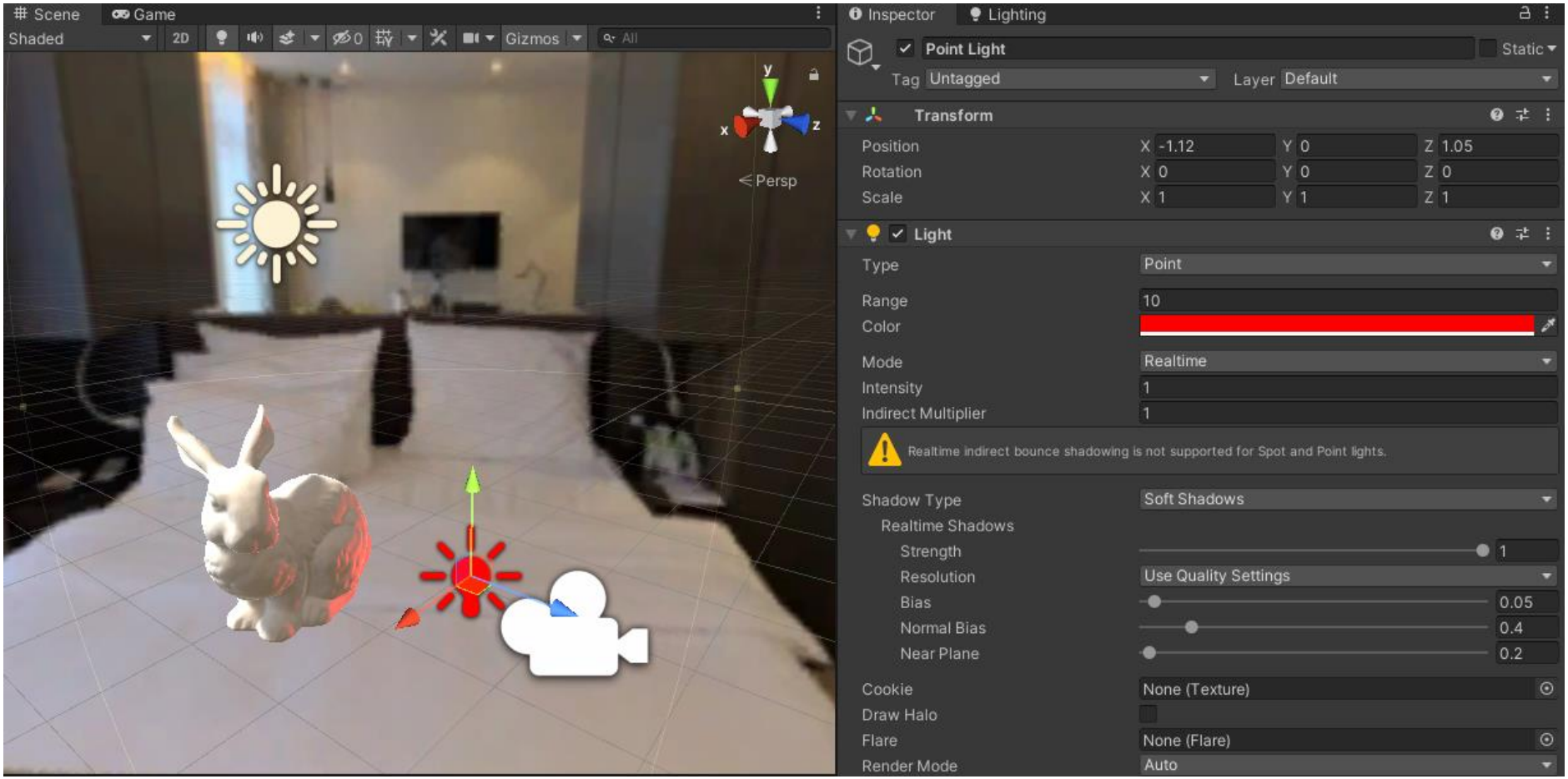
Lights in Unity (cont.)

- Directional light



Lights in Unity (cont.)

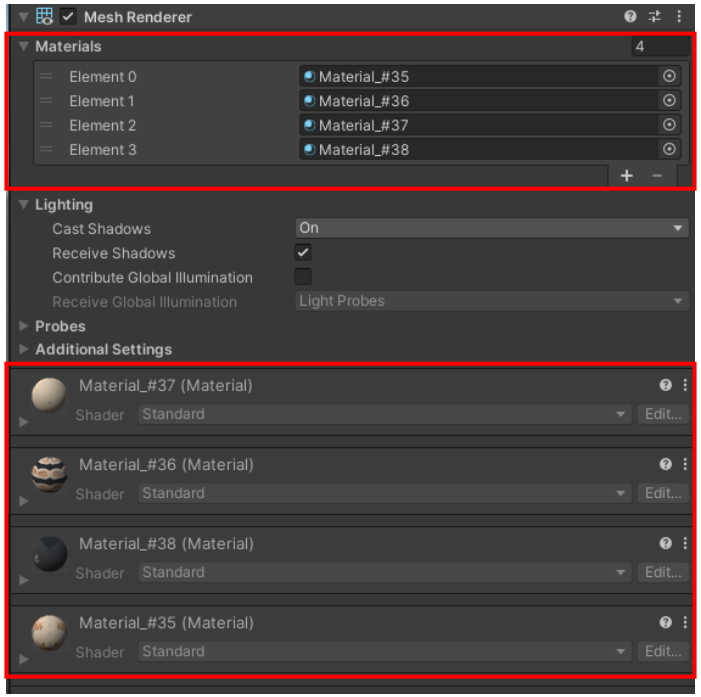
- Point light



Unity MeshRenderer

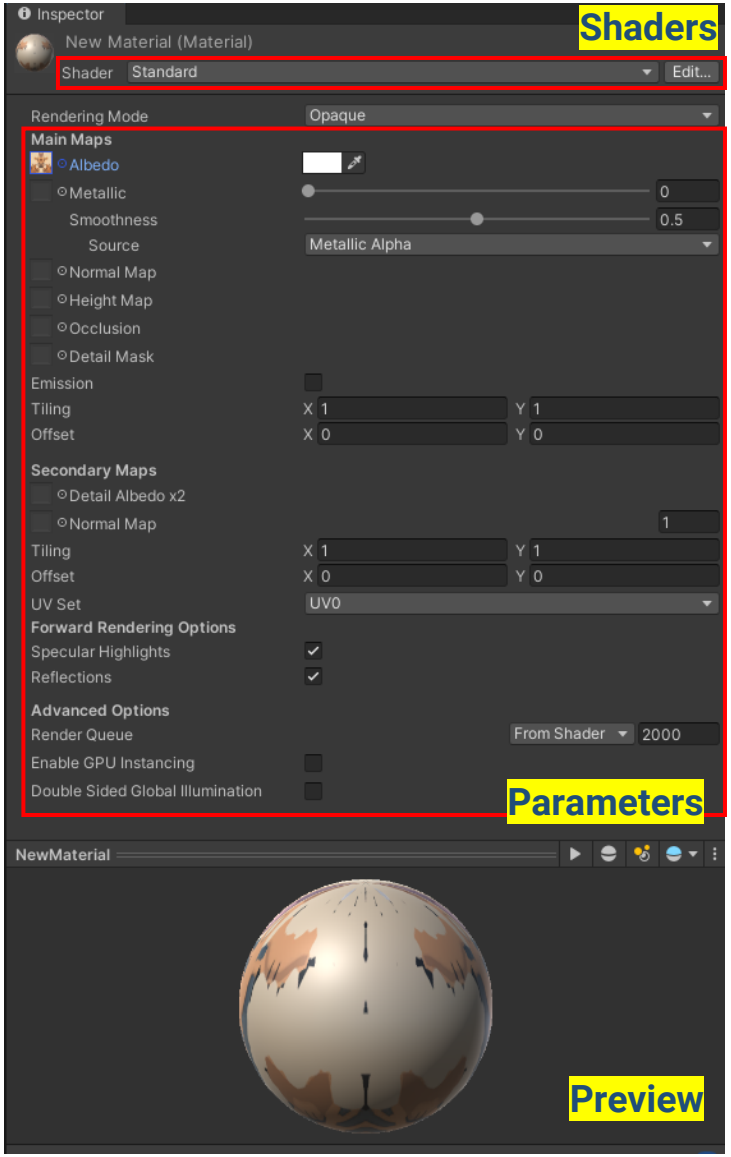
- Rendering features are described in a **(Mesh)Renderer** component
  - **Materials**
    - The material of each subMesh
  - **Lighting**
    - Does the object cast/receive shadows?
  - **Probe**
    - Does the object shade with light probes

    (e.g., reflection cubemaps)



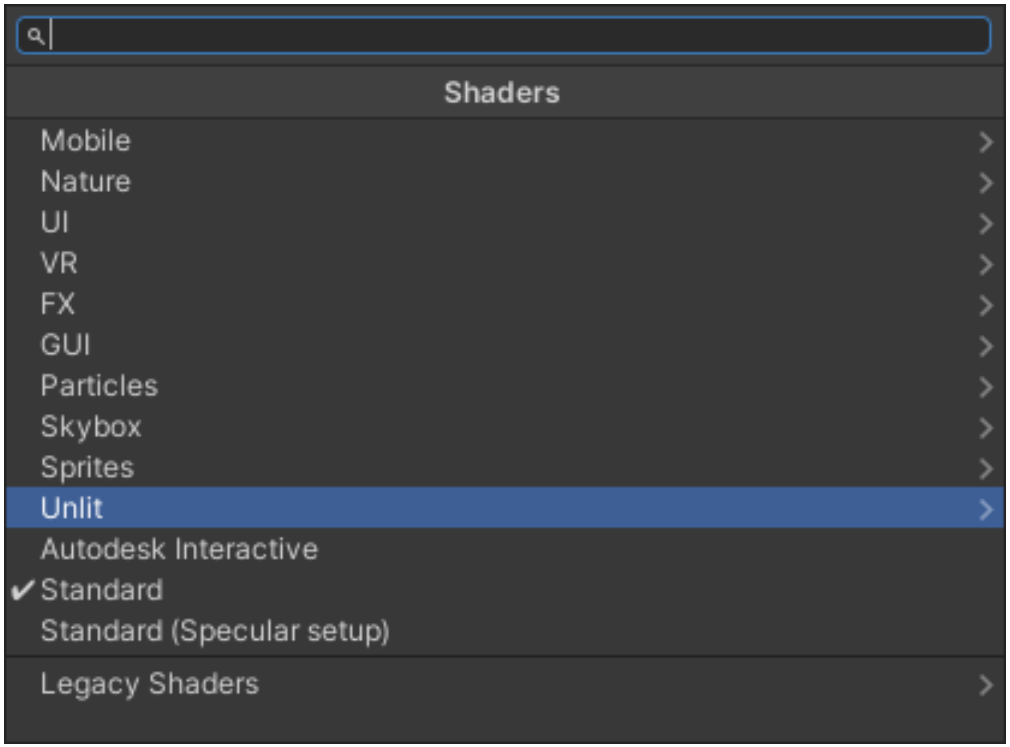
Unity Material

- **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



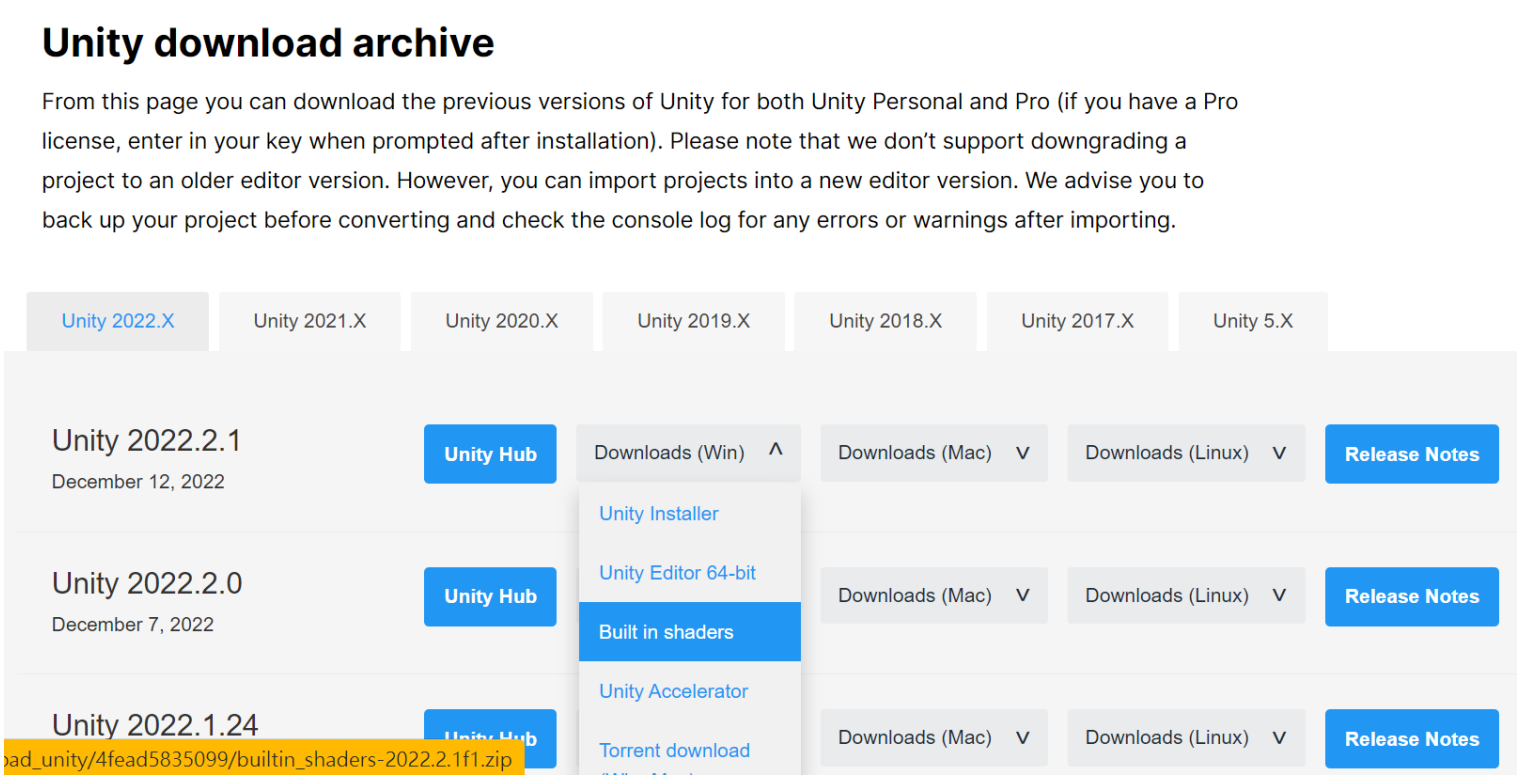
Unity Built-in Shaders

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



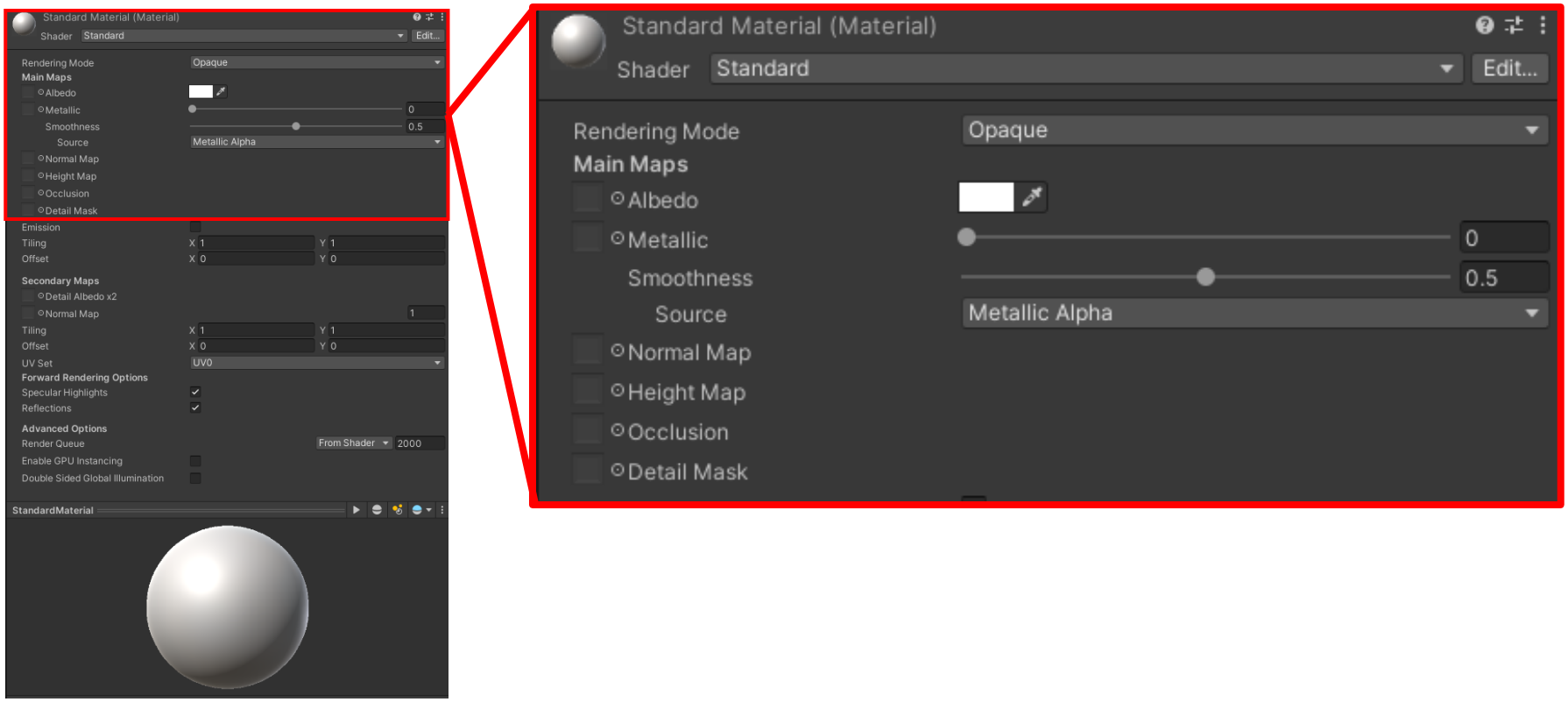
Unity Built-in Shaders (cont.)

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



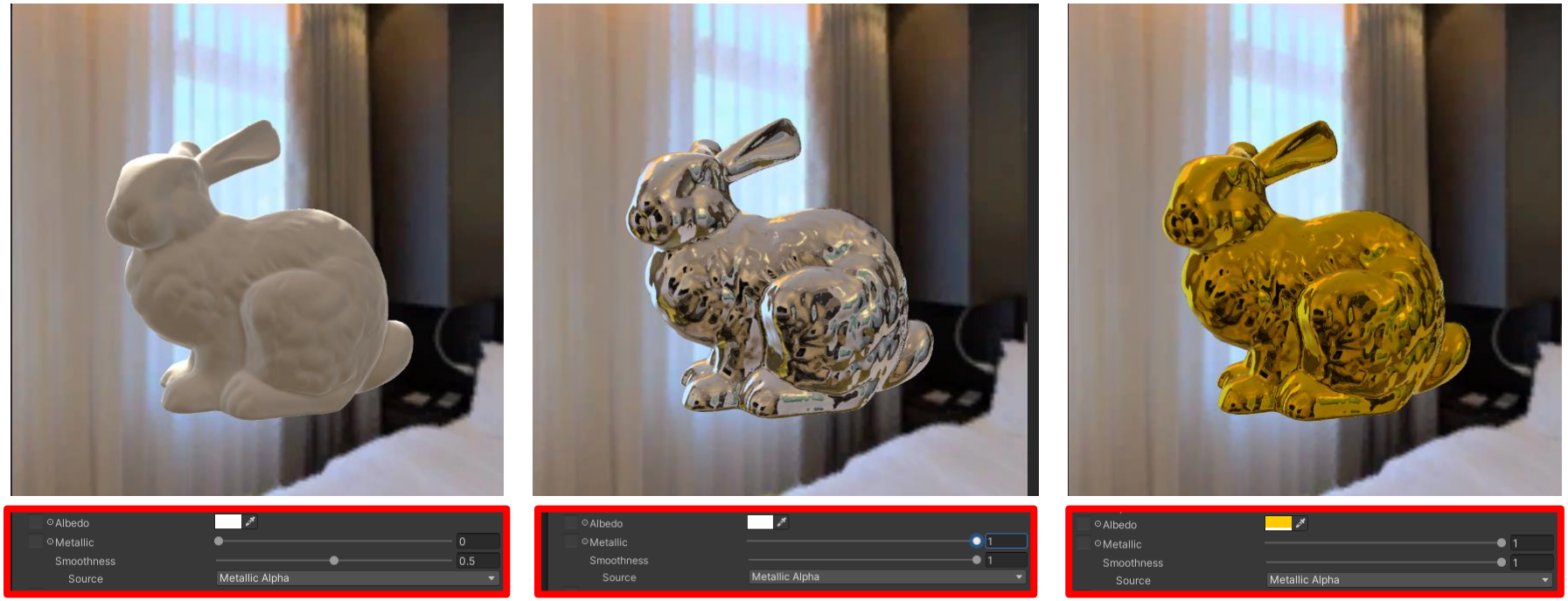
Unity Built-in Shaders (cont.)

- **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 the **arbitrary number** and **various types** of **lights**?
  - By multiple rendering passes

directional light
ZWrite
ZTest LEqual

Blend
SrcAlpha DstAlpha

```
// ------------------------------------------------------------------
//  Base forward pass (directional light, emission, lightmaps, ...)
Pass
{
    Name "FORWARD"
    Tags { "LightMode" = "ForwardBase" }

    Blend [_SrcBlend] [_DstBlend]
    ZWrite [_ZWrite]

    CGPROGRAM
    #pragma target 3.0

    // -------------------------------------

    #pragma shader_feature_local _NORMALMAP
    #pragma shader_feature_local _ _ALPHATEST_ON _ALPHABLEND_ON _ALP
    #pragma shader_feature_fragment _EMISSION
    #pragma shader_feature_local _METALLICGLOSSMAP
    #pragma shader_feature_local_fragment _DETAIL_MULX2
    #pragma shader_feature_local_fragment _SMOOTHNESS_TEXTURE_ALBEDO
    #pragma shader_feature_local_fragment _SPECULARHIGHLIGHTS_OFF
    #pragma shader_feature_local_fragment _GLOSSYREFLECTIONS_OFF
    #pragma shader_feature_local _PARALLAXMAP

    #pragma multi_compile_fwdbase
    #pragma multi_compile_fog
    #pragma multi_compile_instancing
    // Uncomment the following line to enable dithering LOD crossfad
    //#pragma multi_compile _ LOD_FADE_CROSSFADE

    #pragma vertex vertBase
    #pragma fragment fragBase
    #include "UnityStandardCoreForward.cginc"

    ENDCG
}
```

per point (spot) light
ZWrite Off
ZTest LEqual

Blend
SrcAlpha One

```
// ------------------------------------------------------------------
//  Additive forward pass (one light per pass)
Pass
{
    Name "FORWARD_DELTA"
    Tags { "LightMode" = "ForwardAdd" }
    Blend [_SrcBlend] One
    Fog { Color (0,0,0,0) } // in additive pass fog should be black
    ZWrite Off
    ZTest LEqual

    CGPROGRAM
    #pragma target 3.0

    // -------------------------------------


    #pragma shader_feature_local _NORMALMAP
    #pragma shader_feature_local _ _ALPHATEST_ON _ALPHABLEND_ON _ALPHAI
    #pragma shader_feature_local _METALLICGLOSSMAP
    #pragma shader_feature_local_fragment _SMOOTHNESS_TEXTURE_ALBEDO_CI
    #pragma shader_feature_local_fragment _SPECULARHIGHLIGHTS_OFF
    #pragma shader_feature_local_fragment _DETAIL_MULX2
    #pragma shader_feature_local _PARALLAXMAP

    #pragma multi_compile_fwdadd_fullshadows
    #pragma multi_compile_fog
    // Uncomment the following line to enable dithering LOD crossfade.
    //#pragma multi_compile _ LOD_FADE_CROSSFADE

    #pragma vertex vertAdd
    #pragma fragment fragAdd
    #include "UnityStandardCoreForward.cginc"

    ENDCG
}
```

If there are too many lights, the less important ones will be rendered by vertex lighting

Design of Unity's Rendering System (cont.)

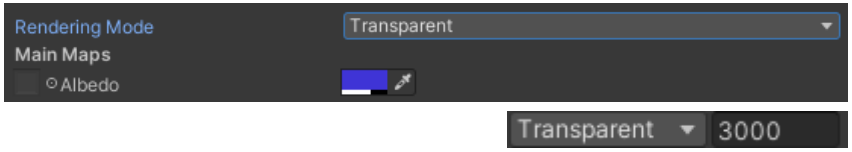
- How does Unity handle **various materials**? For example, with or without an **Albedo** texture

```
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?