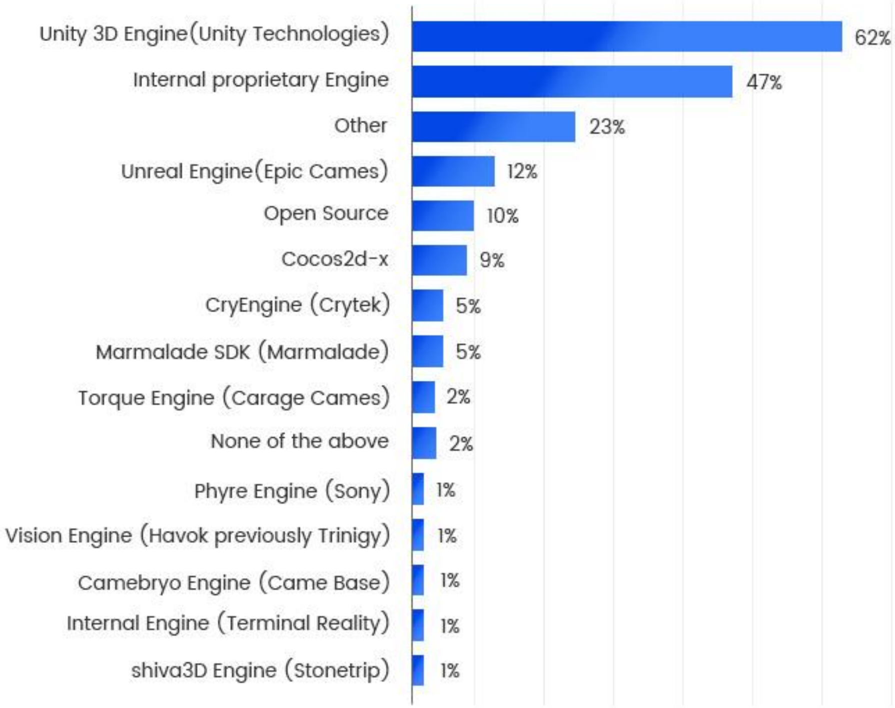
# A Case Study: Unity

## Computer Graphics

Yu-Ting Wu

# Unity Overview

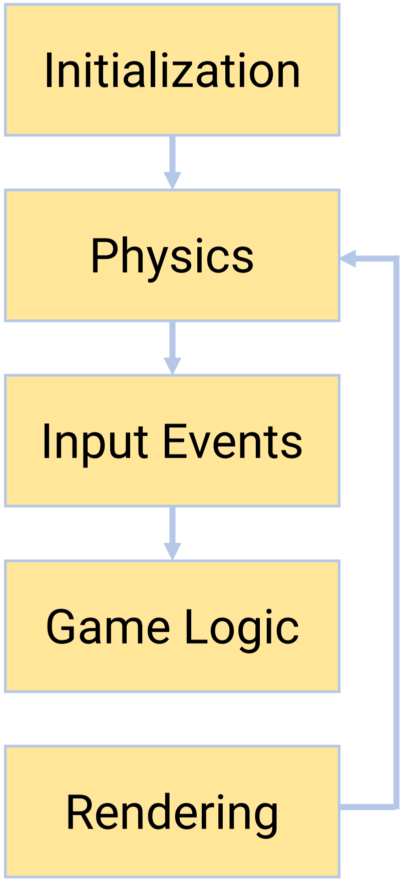
- The most widely used game engine (especially for mobile games) today
- Easier to jump in



statista

# Unity Overview (cont.)

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



Initialization

Awake, Start, OnEnable

Physics

FixedUpdate,
AnimationUpdate, PhysicsUpdate,
OnTriggerXXX, OnCollisionXXX

Input Events

OnMouseXXX

Game Logic

Update, Coroutine, LateUpdate

Rendering

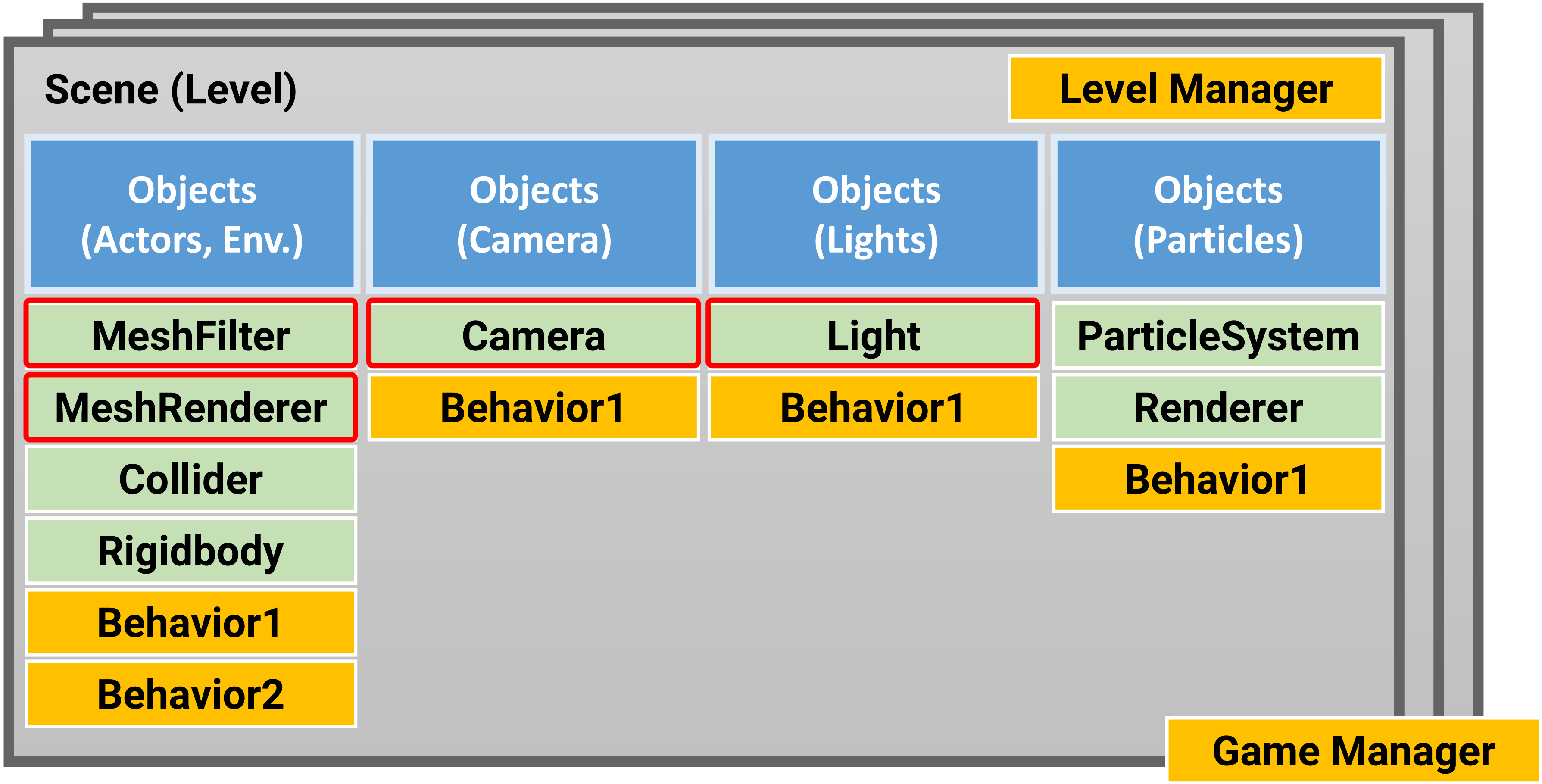
OnWillRenderObject,
OnPreRender, OnRenderObject, OnPostRender, OnRenderImage,
OnGUI

# Unity Overview (cont.)

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

Custom

Built-in



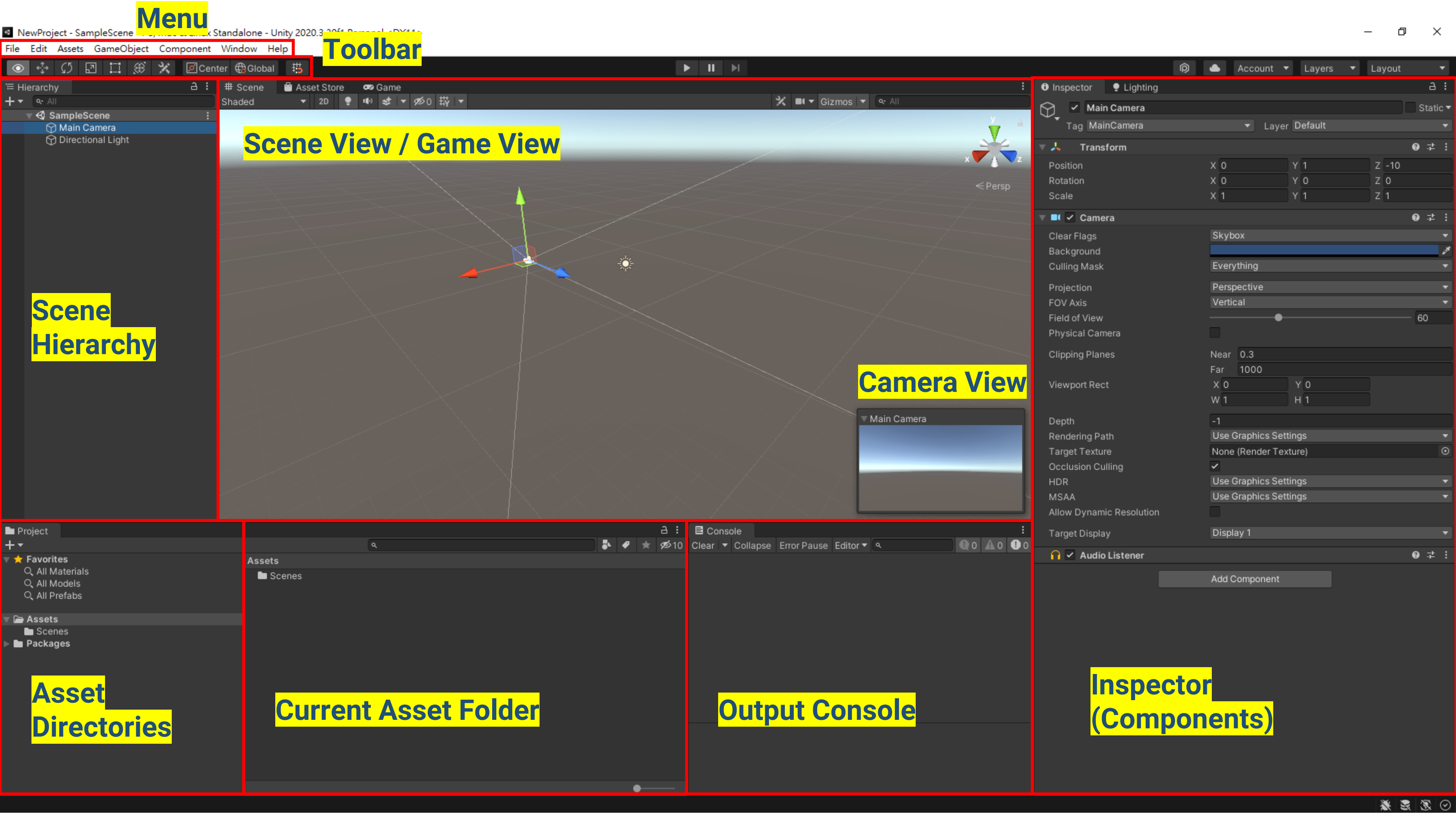
Scene (Level)

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 Editor



Menu

Toolbar

Scene Hierarchy

Scene View / Game View

Camera View

Asset Directories

Current Asset Folder

Output Console

Inspector (Components)

Main Camera

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

FOV Axis: Vertical

Field of View: 60

Physical Camera: □

Clipping Planes: Near 0.3, Far 1000

Viewport Rect: X 0 Y 0, W 1 H 1

Depth: -1

Rendering Path: Use Graphics Settings

Target Texture: None (Render Texture)

Occlusion Culling: ✓

HDR: Use Graphics Settings

MSAA: Use Graphics Settings

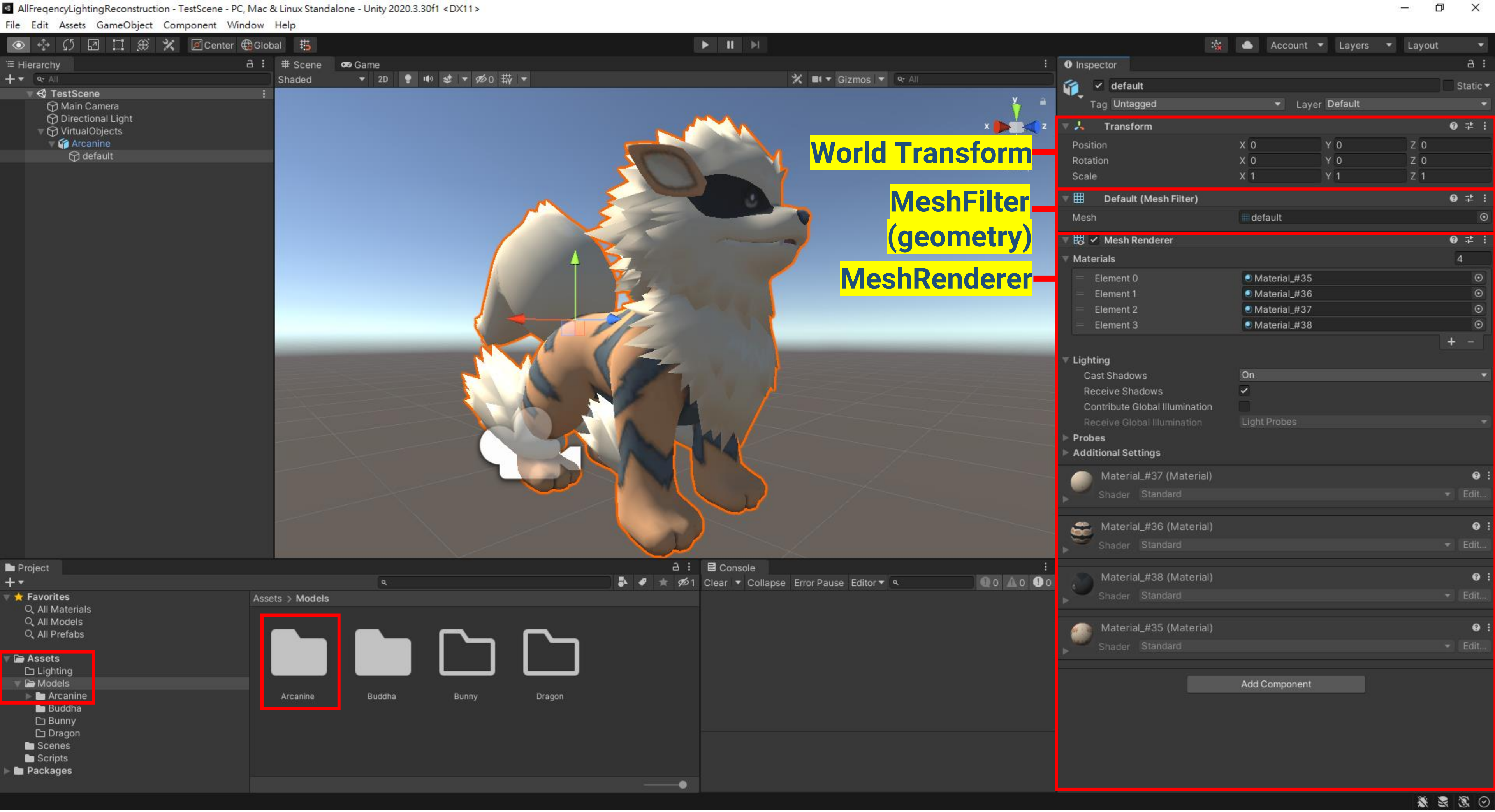
Allow Dynamic Resolution: □

Target Display: Display 1

Audio Listener

Add Component

# Unity Editor



World Transform

MeshFilter (geometry)

MeshRenderer

AllFreqencyLightingReconstruction - TestScene - PC, Mac & Linux Standalone - Unity 2020.3.30f1 <DX11>

Hierarchy:
- TestScene
  - Main Camera
  - Directional Light
  - VirtualObjects
    - Arcanine
      - default

Inspector: default — Tag: Untagged — Layer: Default

Transform

| | X | Y | Z |
|---|---|---|---|
| Position | 0 | 0 | 0 |
| Rotation | 0 | 0 | 0 |
| Scale | 1 | 1 | 1 |

Default (Mesh Filter)
Mesh: default

Mesh Renderer
Materials: 4

| Element | Material |
|---|---|
| Element 0 | Material_#35 |
| Element 1 | Material_#36 |
| Element 2 | Material_#37 |
| Element 3 | Material_#38 |

Lighting
- Cast Shadows: On
- Receive Shadows: ☑
- Contribute Global Illumination: □
- Receive Global Illumination: Light Probes

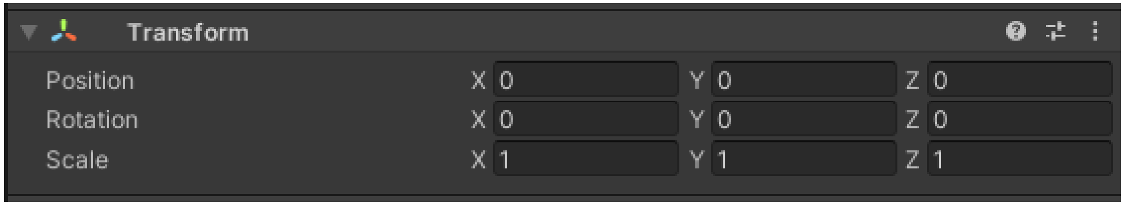
Material_#37 (Material) — Shader: Standard
Material_#36 (Material) — Shader: Standard
Material_#38 (Material) — Shader: Standard
Material_#35 (Material) — Shader: Standard

Project:
- Favorites
  - All Materials
  - All Models
  - All Prefabs
- Assets
  - Lighting
  - Models
    - Arcanine
    - Buddha
    - Bunny
    - Dragon
  - Scenes
  - Scripts
- Packages

Assets > Models: Arcanine, Buddha, Bunny, Dragon

# Geometry Data in Unity

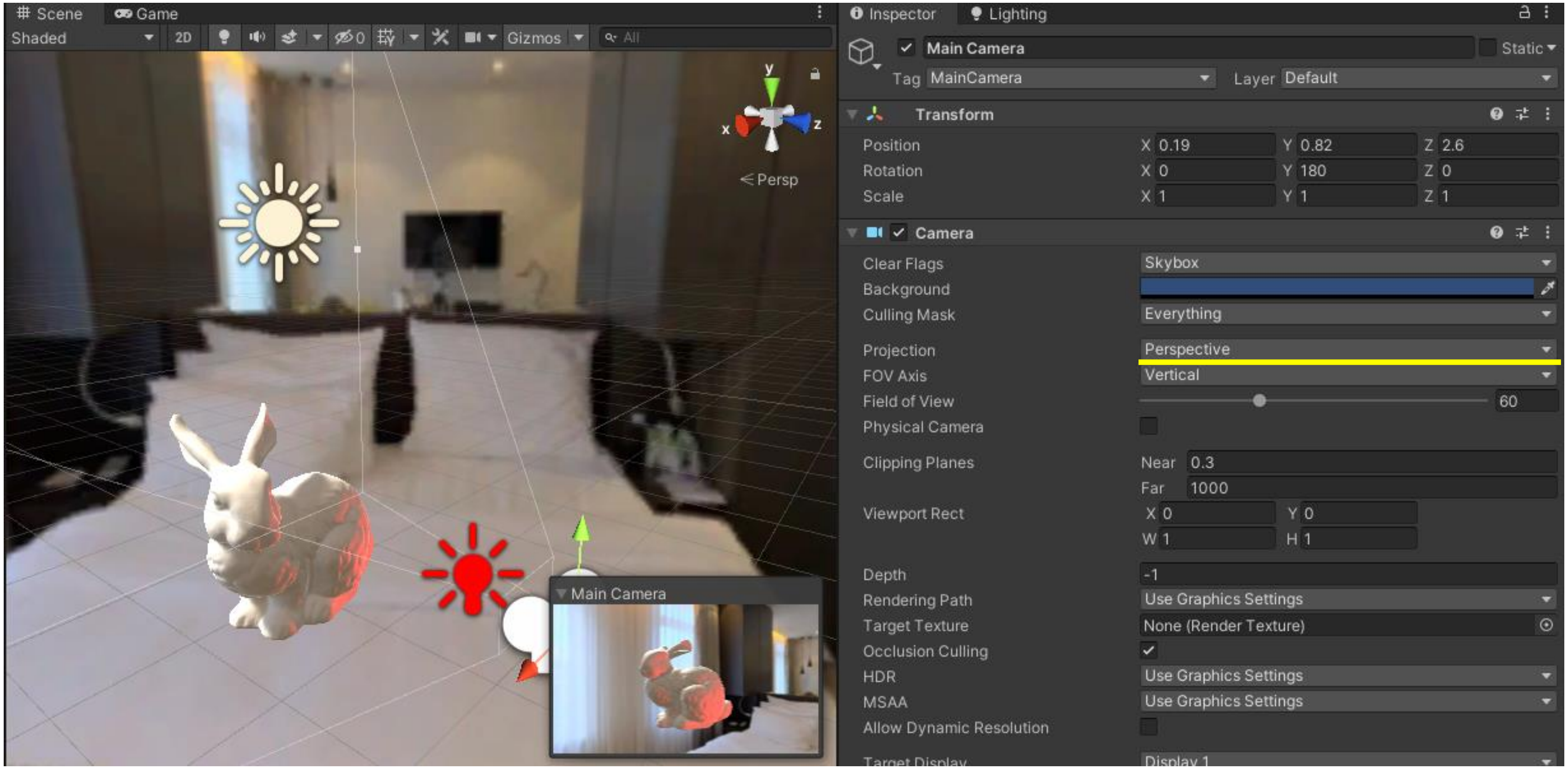
- Geometry data in Object Space is described in a **MeshFilter** component
  - **Mesh**
    - vertexBufferTarget / indexBufferTarget
    - vertices (position) / normals / uv(12345678) / tangents
    - triangles (indices)
    - subMeshCount
    - …
- An object is placed in the virtual world by a **World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



| | | | |
|---|---|---|---|
| Position | X 0 | Y 0 | Z 0 |
| Rotation | X 0 | Y 0 | Z 0 |
| Scale | X 1 | Y 1 | Z 1 |

# Camera in Unity

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



☑ Main Camera □ Static

Tag MainCamera Layer Default

**Transform**

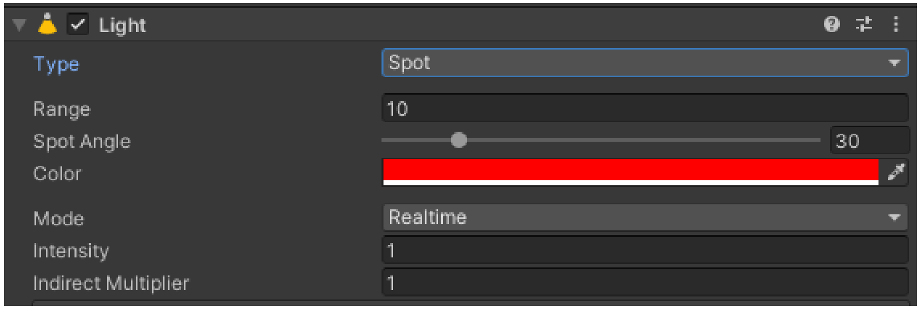
| | | | |
|---|---|---|---|
| Position | X 0.19 | Y 0.82 | Z 2.6 |
| Rotation | X 0 | Y 180 | Z 0 |
| Scale | X 1 | Y 1 | Z 1 |

☑ **Camera**

| | |
|---|---|
| Clear Flags | Skybox |
| Background | |
| Culling Mask | Everything |
| Projection | Perspective |
| FOV Axis | Vertical |
| Field of View | 60 |
| Physical Camera | □ |
| Clipping Planes | Near 0.3<br>Far 1000 |
| Viewport Rect | X 0 Y 0<br>W 1 H 1 |
| Depth | -1 |
| Rendering Path | Use Graphics Settings |
| Target Texture | None (Render Texture) |
| Occlusion Culling | ☑ |
| HDR | Use Graphics Settings |
| MSAA | Use Graphics Settings |
| Allow Dynamic Resolution | □ |
| Target Display | Display 1 |

# 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)

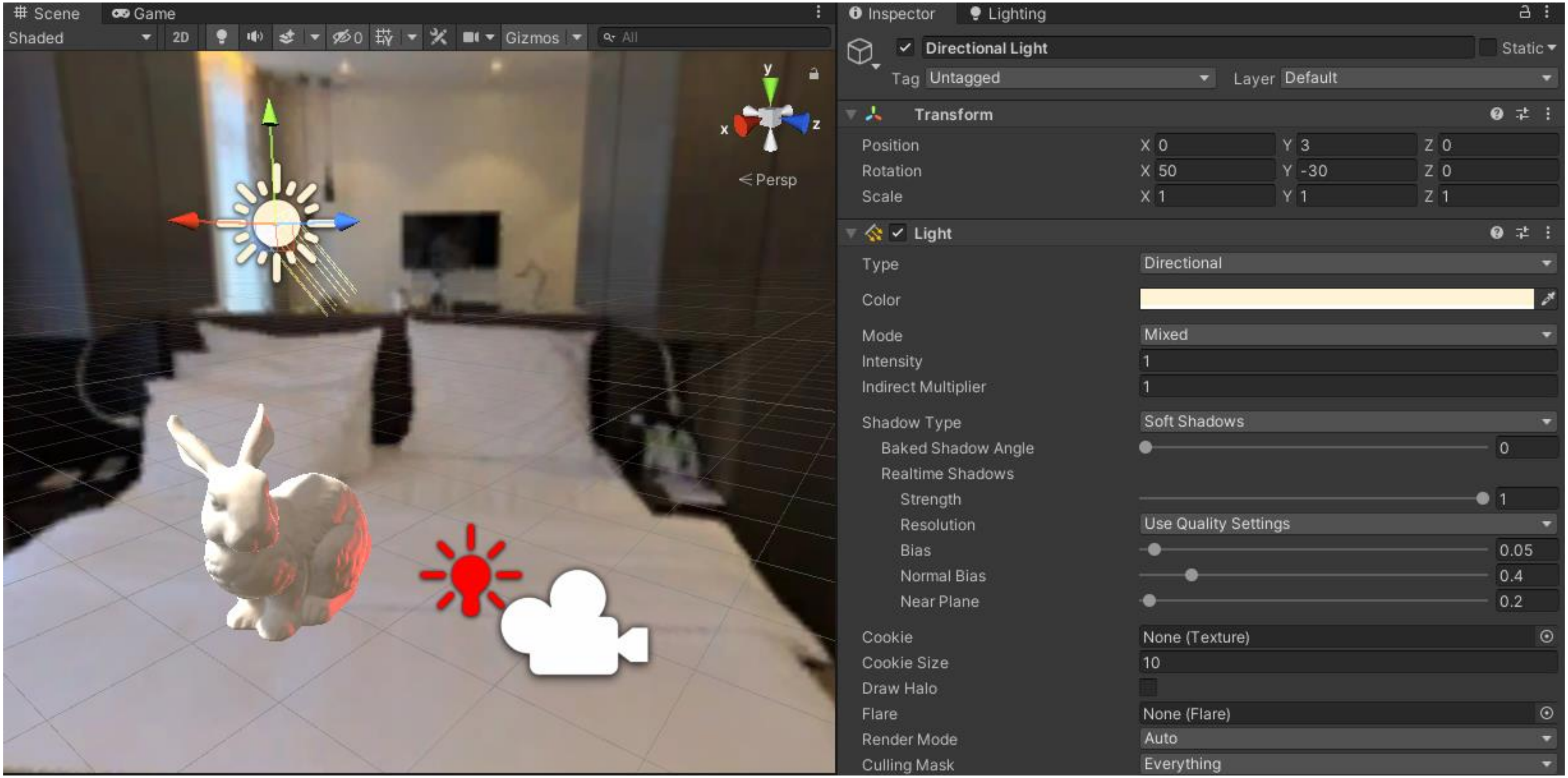


☑ Light

| | |
|---|---|
| Type | Spot |
| Range | 10 |
| Spot Angle | 30 |
| Color | 🟥 |
| Mode | Realtime |
| Intensity | 1 |
| Indirect Multiplier | 1 |

# Lights in Unity (cont.)

- Directional light



☑ Directional Light □ Static
Tag Untagged Layer Default

Transform

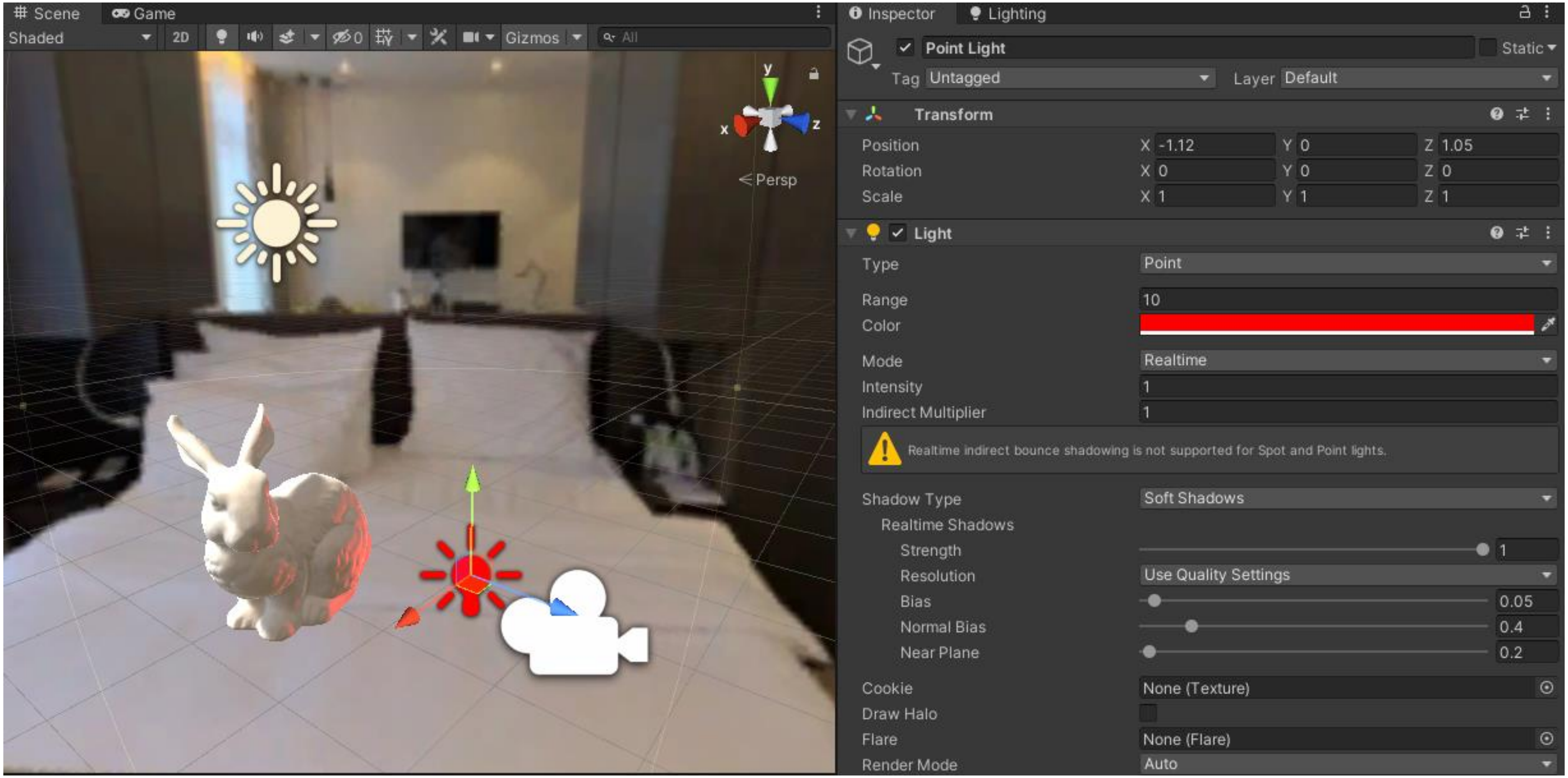
| | | | |
|---|---|---|---|
| Position | X 0 | Y 3 | Z 0 |
| Rotation | X 50 | Y -30 | Z 0 |
| Scale | X 1 | Y 1 | Z 1 |

☑ Light

| | |
|---|---|
| Type | Directional |
| Color | |
| Mode | Mixed |
| Intensity | 1 |
| Indirect Multiplier | 1 |
| Shadow Type | Soft Shadows |
| Baked Shadow Angle | 0 |
| Realtime Shadows | |
| Strength | 1 |
| Resolution | Use Quality Settings |
| Bias | 0.05 |
| Normal Bias | 0.4 |
| Near Plane | 0.2 |
| Cookie | None (Texture) |
| Cookie Size | 10 |
| Draw Halo | □ |
| Flare | None (Flare) |
| Render Mode | Auto |
| Culling Mask | Everything |

# Lights in Unity (cont.)

- Point light



| | |
|---|---|
| ☑ Point Light | |
| Tag | Untagged |
| Layer | Default |

**Transform**

| | X | Y | Z |
|---|---|---|---|
| Position | -1.12 | 0 | 1.05 |
| Rotation | 0 | 0 | 0 |
| Scale | 1 | 1 | 1 |

**☑ Light**

| | |
|---|---|
| Type | Point |
| Range | 10 |
| Color | 🟥 |
| Mode | Realtime |
| Intensity | 1 |
| Indirect Multiplier | 1 |

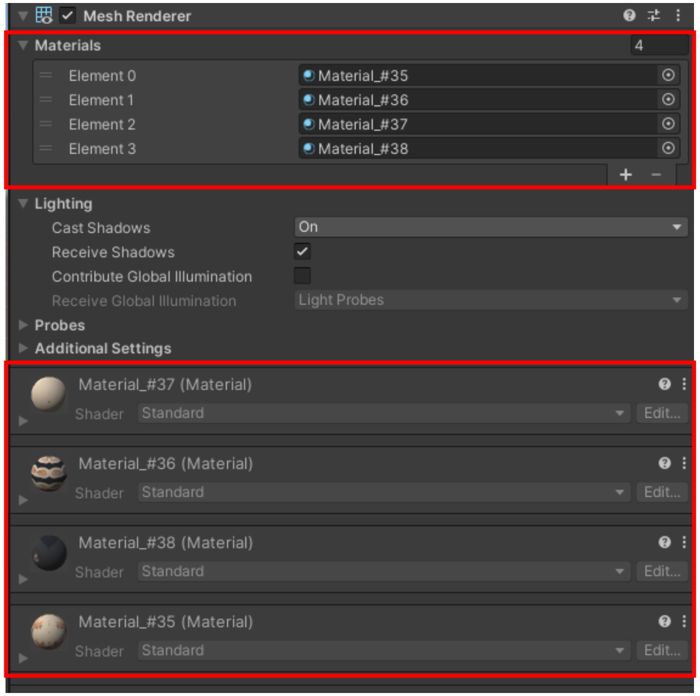
⚠ Realtime indirect bounce shadowing is not supported for Spot and Point lights.

| | |
|---|---|
| Shadow Type | Soft Shadows |
| Realtime Shadows | |
| Strength | 1 |
| Resolution | Use Quality Settings |
| Bias | 0.05 |
| Normal Bias | 0.4 |
| Near Plane | 0.2 |
| Cookie | None (Texture) |
| Draw Halo | □ |
| Flare | None (Flare) |
| Render Mode | Auto |

# 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



Mesh Renderer

Materials 4

| | |
|---|---|
| Element 0 | Material_#35 |
| Element 1 | Material_#36 |
| Element 2 | Material_#37 |
| Element 3 | Material_#38 |

Lighting

| | |
|---|---|
| Cast Shadows | On |
| Receive Shadows | ☑ |
| Contribute Global Illumination | □ |
| Receive Global Illumination | Light Probes |

Probes

Additional Settings

Material_#37 (Material)
Shader Standard Edit...

Material_#36 (Material)
Shader Standard Edit...

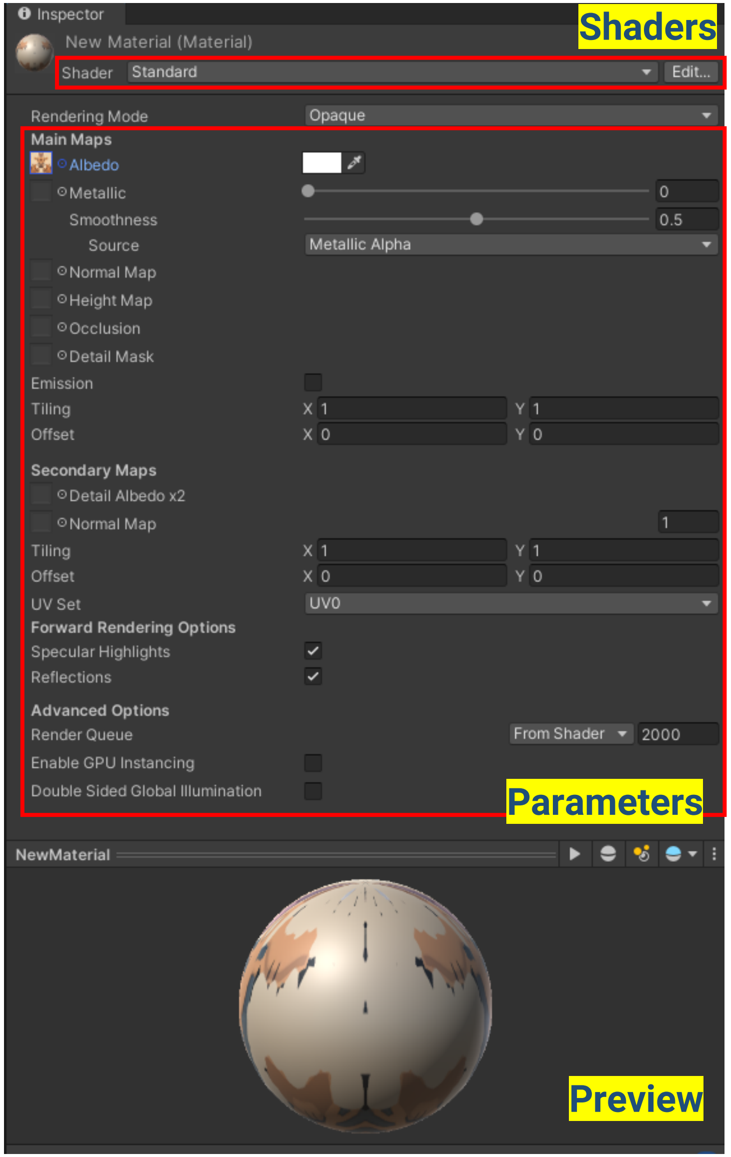
Material_#38 (Material)
Shader Standard Edit...

Material_#35 (Material)
Shader Standard Edit...

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



Shaders

Inspector

New Material (Material)

Shader: Standard | Edit...

Rendering Mode: Opaque

Main Maps
- Albedo
- Metallic: 0
- Smoothness: 0.5
- Source: Metallic Alpha
- Normal Map
- Height Map
- Occlusion
- Detail Mask
- Emission: □
- Tiling: X 1, Y 1
- Offset: X 0, Y 0

Secondary Maps
- Detail Albedo x2
- Normal Map: 1
- Tiling: X 1, Y 1
- Offset: X 0, Y 0
- UV Set: UV0

Forward Rendering Options
- Specular Highlights: ☑
- Reflections: ☑

Advanced Options
- Render Queue: From Shader, 2000
- Enable GPU Instancing: □
- Double Sided Global Illumination: □

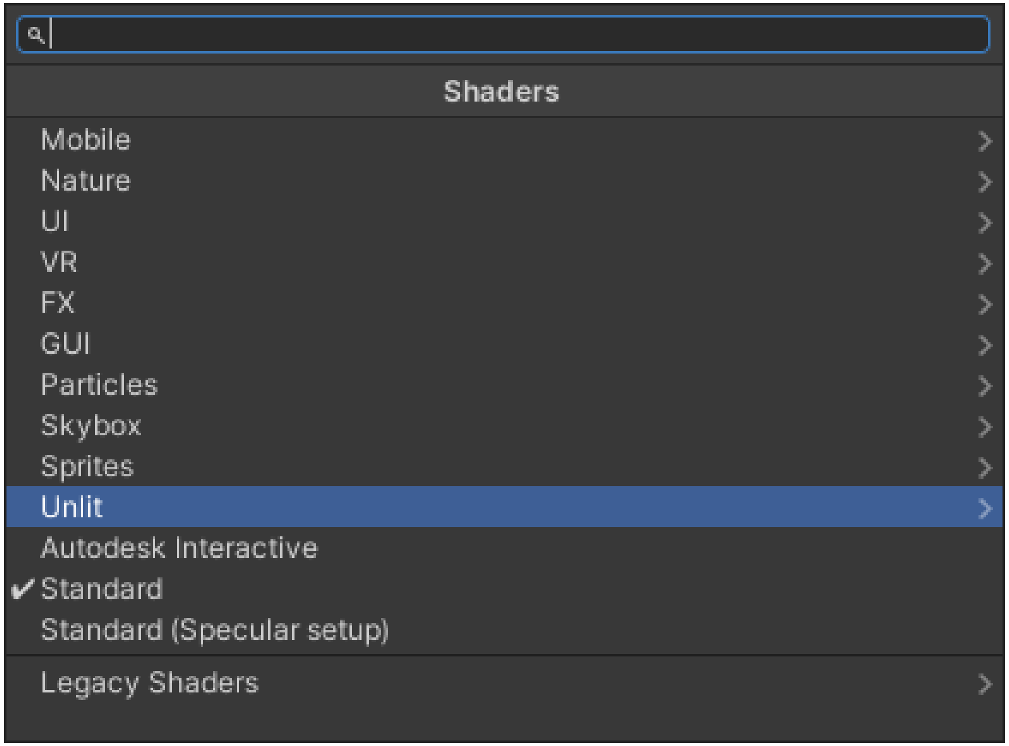
Parameters

NewMaterial

Preview

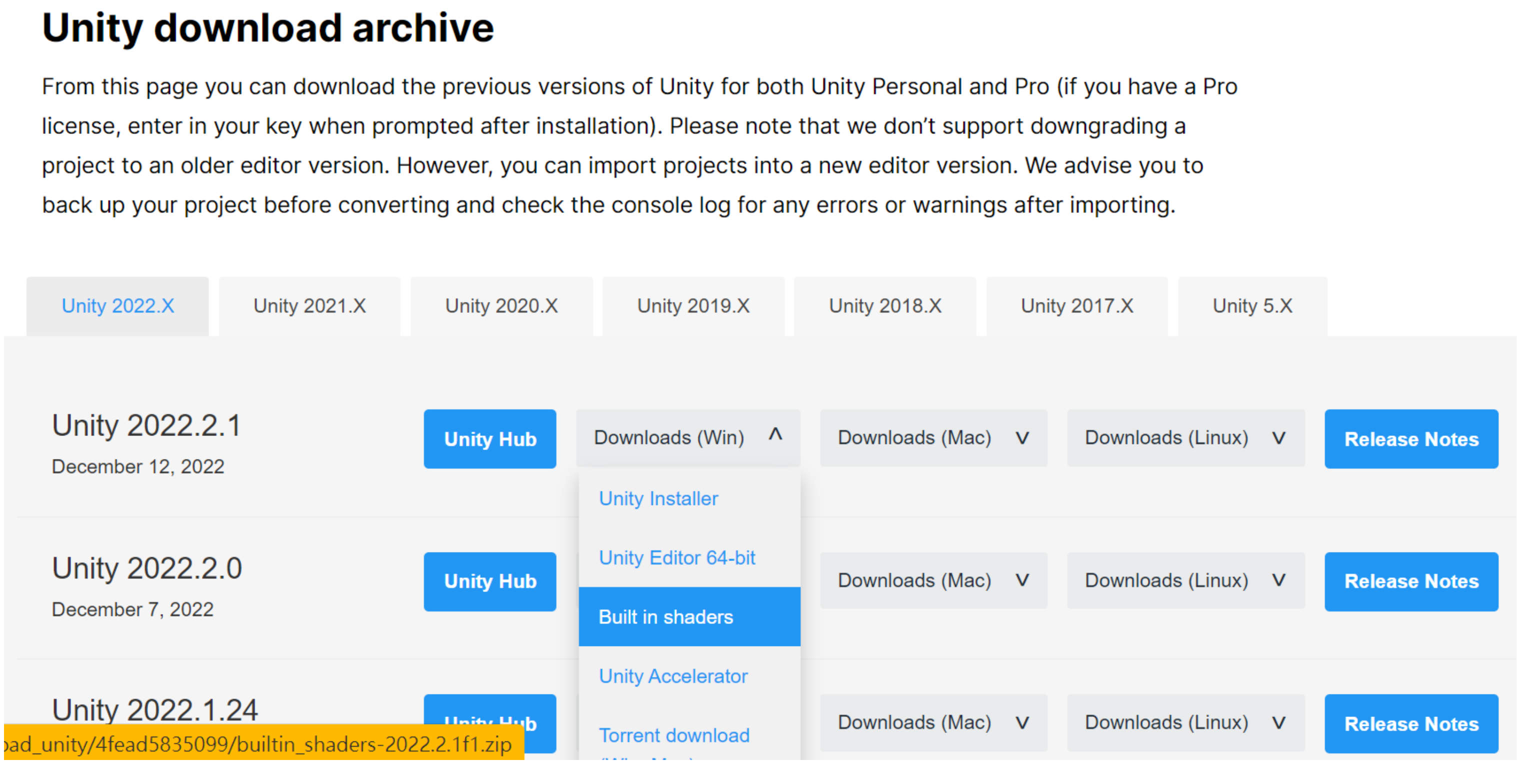
# Unity Built-in Shaders

- 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

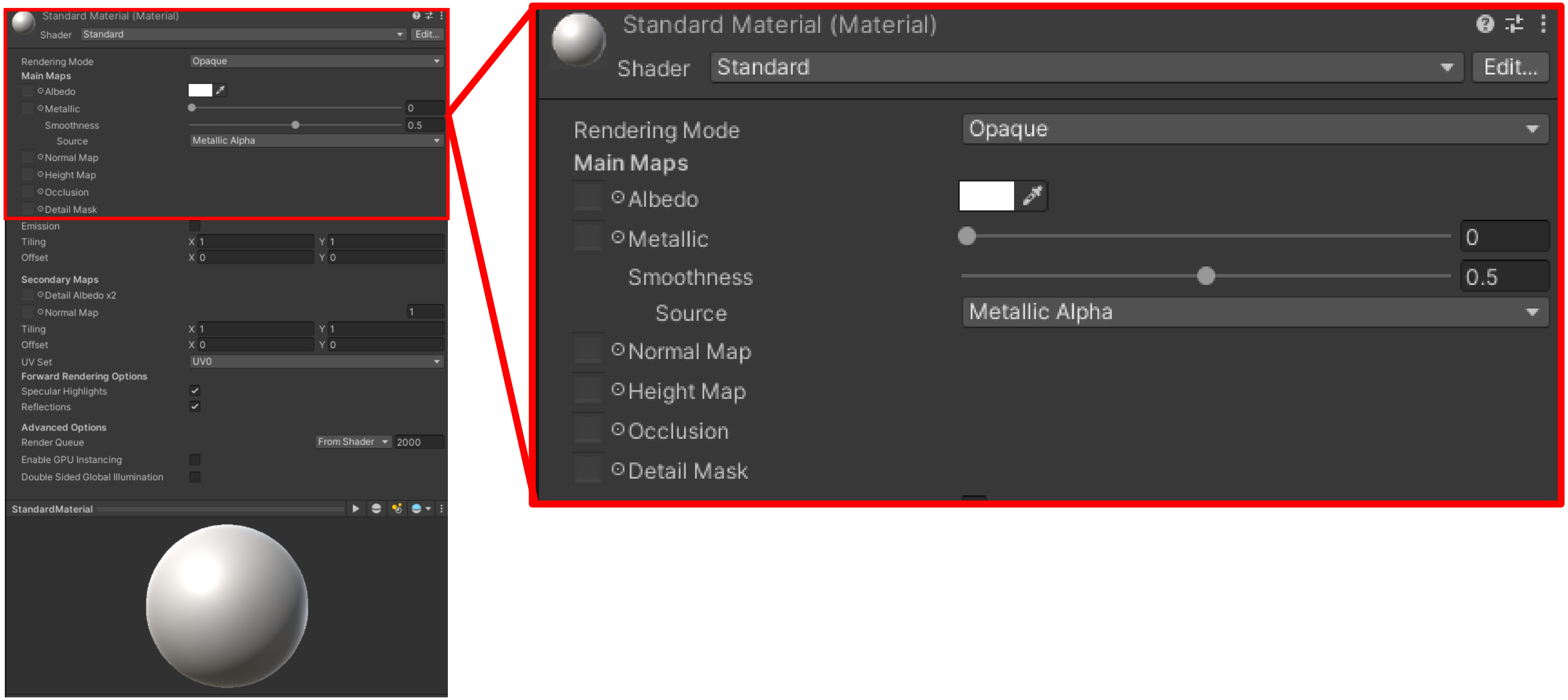


## Unity download archive

From this page you can download the previous versions of Unity for both Unity Personal and Pro (if you have a Pro license, enter in your key when prompted after installation). Please note that we don’t support downgrading a project to an older editor version. However, you can import projects into a new editor version. We advise you to back up your project before converting and check the console log for any errors or warnings after importing.

# 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



Standard Material (Material)

Shader: Standard (Edit...)

Rendering Mode: Opaque

Main Maps

- Albedo
- Metallic: 0
- Smoothness: 0.5
- Source: Metallic Alpha
- Normal Map
- Height Map
- Occlusion
- Detail Mask

Emission

Tiling: X 1, Y 1

Offset: X 0, Y 0

Secondary Maps

- Detail Albedo x2
- Normal Map: 1

Tiling: X 1, Y 1

Offset: X 0, Y 0

UV Set: UV0

Forward Rendering Options

- Specular Highlights ☑
- Reflections ☑

Advanced Options

- Render Queue: From Shader, 2000
- Enable GPU Instancing □
- Double Sided Global Illumination □

StandardMaterial

Standard Material (Material)

Shader: Standard (Edit...)

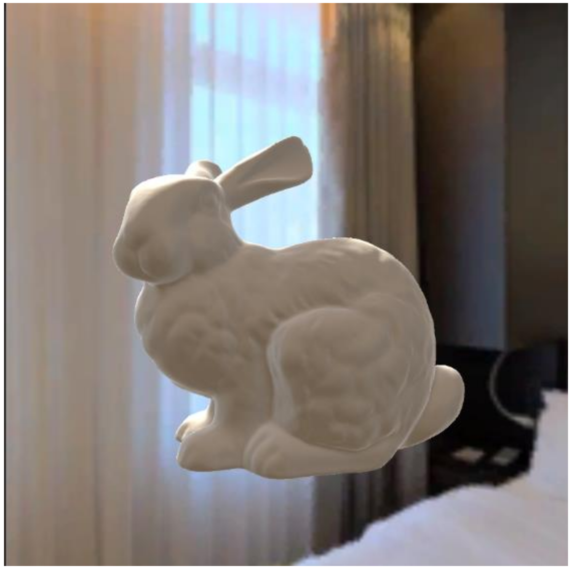
Rendering Mode: Opaque

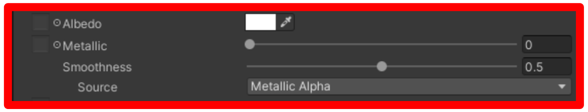
Main Maps

- Albedo
- Metallic: 0
- Smoothness: 0.5
- Source: Metallic Alpha
- Normal Map
- Height Map
- Occlusion
- Detail Mask

# 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



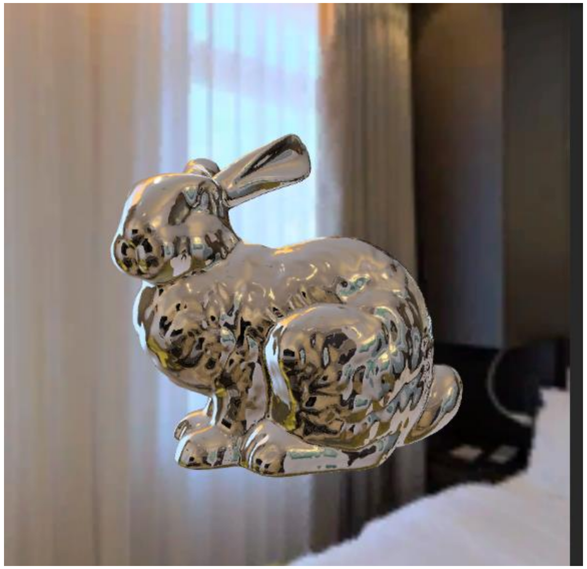


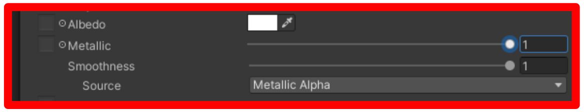
Albedo

Metallic 0

Smoothness 0.5

Source Metallic Alpha



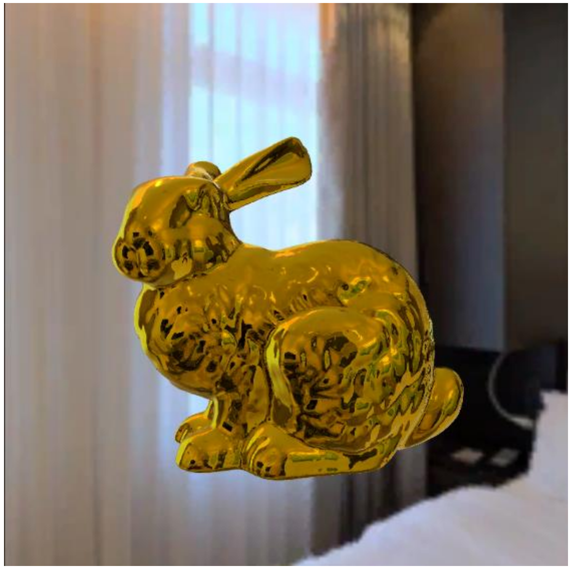


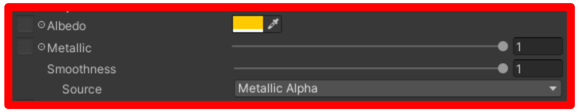
Albedo

Metallic 1

Smoothness 1

Source Metallic Alpha





Albedo

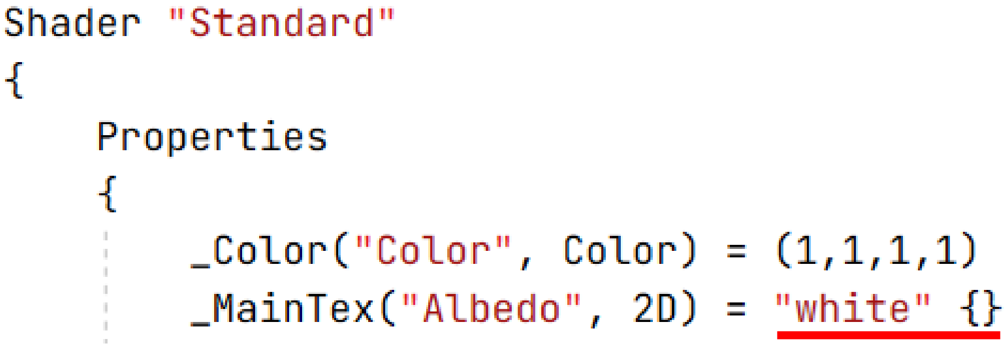
Metallic 1

Smoothness 1

Source Metallic Alpha

# Design of Unity’s Rendering System

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



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

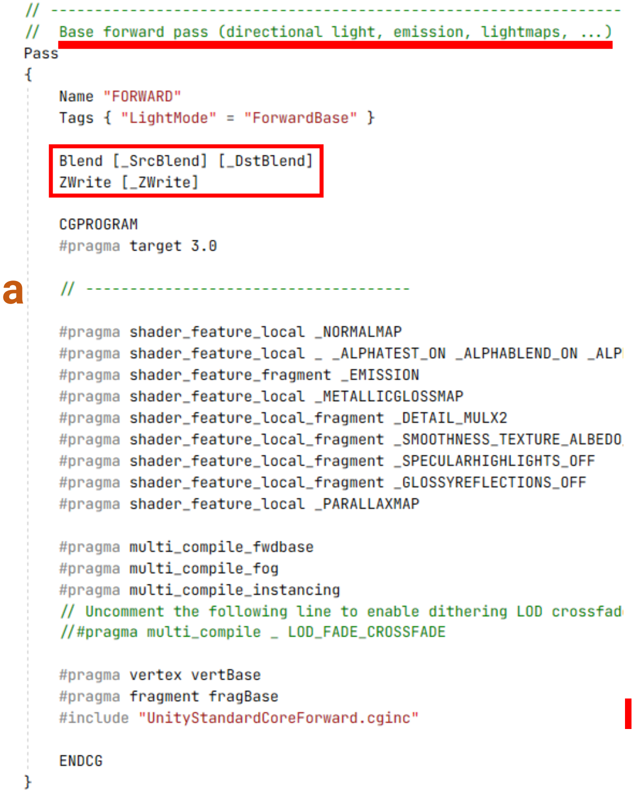
**For materials that do not use an albedo texture**
**Unity will create a pure white one,**
**so the shader code can be unified**

# Design of Unity’s Rendering System (cont.)

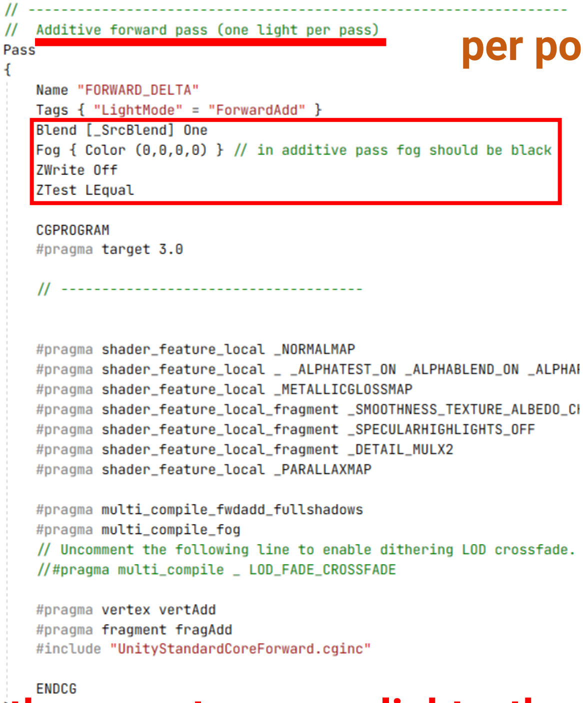
- How does Unity handle the **arbitrary number** and **various types** of **lights**?
  - By multiple rendering passes

**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
}
```



```
// ------------------------------------------------------------------
//  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
}
```

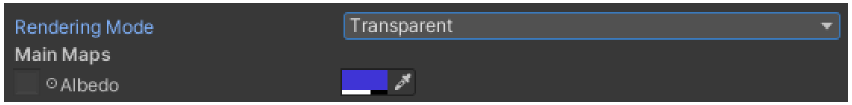
**per point (spot) light**
**ZWrite Off**
**ZTest LEqual**

**Blend**
**SrcAlpha One**

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

# Design of Unity’s Rendering System (cont.)

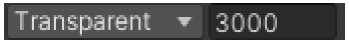
- How does Unity handle **transparency**?
  - By defining RenderQueue
    - Background (1000)
    - Geometry (2000)
    - AlphaTest (2450)
    - Transparent (3000)
    - Overlay



Rendering Mode: Transparent

Main Maps

Albedo



Transparent 3000

END OF PRESENTATION
ANY QUESTIONS?