Shader Beginner


k79k06k02k@gmail.com

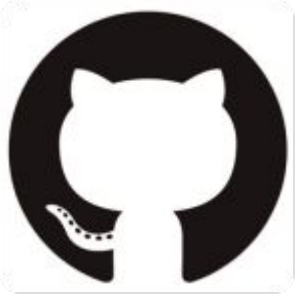

https://www.facebook.com/k79k06k02k

k79k06k02k

About Me

Website
ARKAI Studio

Facebook Page
@ARKAIStudio



GitHub
k79k06k02k

Outline

- CPU to GPU
- Programmable Graphics Pipeline
- Direct3D, OpenGL Render API
- Shader Language: GLSL, HLSL, Cg
- Material, Shader, Texture
- Unity Shader Code Struct
- Unity Shader Category
- Implementation (Vertex and Fragment Shader)
- Q&A

Shader

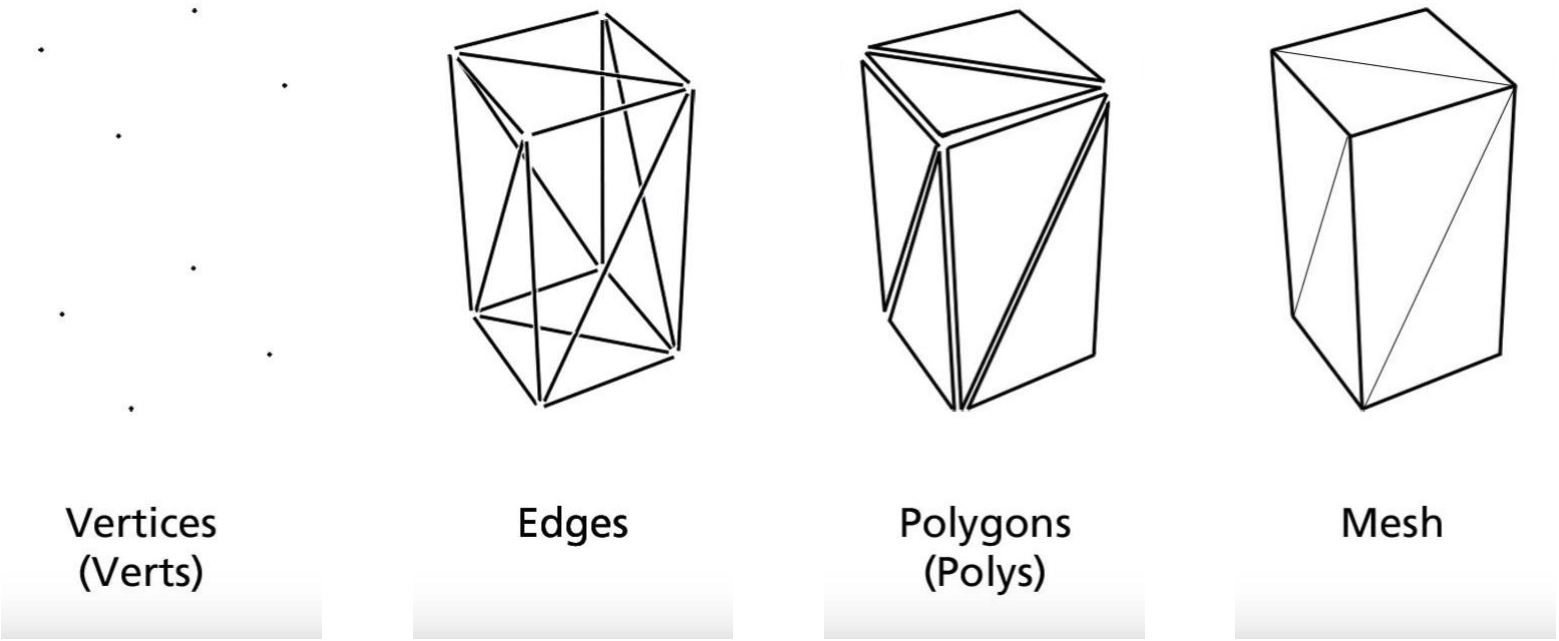
Vertex
Data、
Texture、
Variable...

Input => Shader => Output

Math、Code

Render Result

Mesh

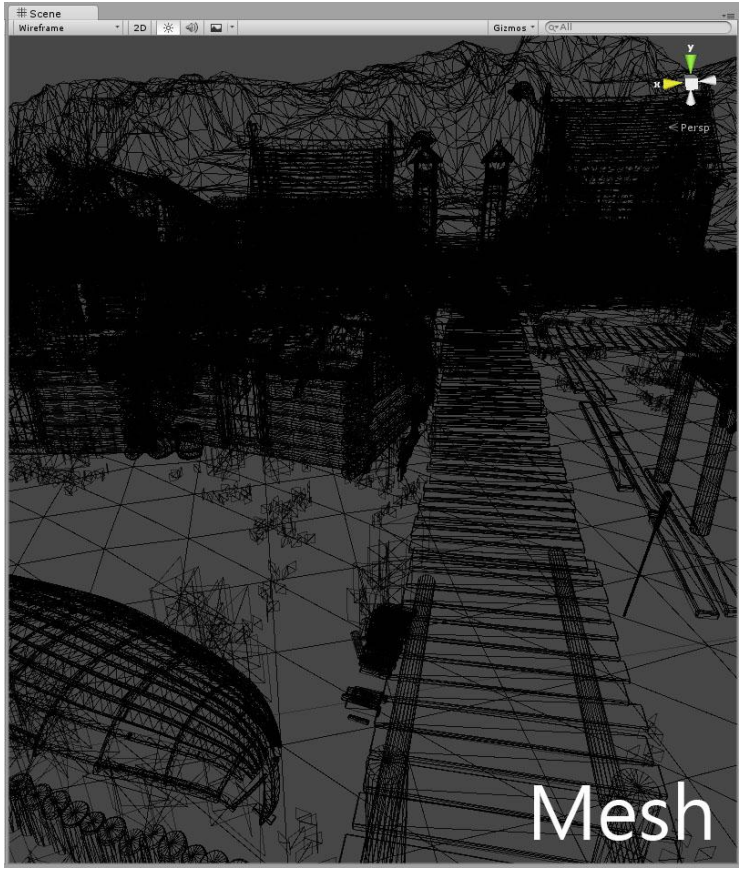


Reference:

Unity - Meshes

Wikipedia - Polygon mesh

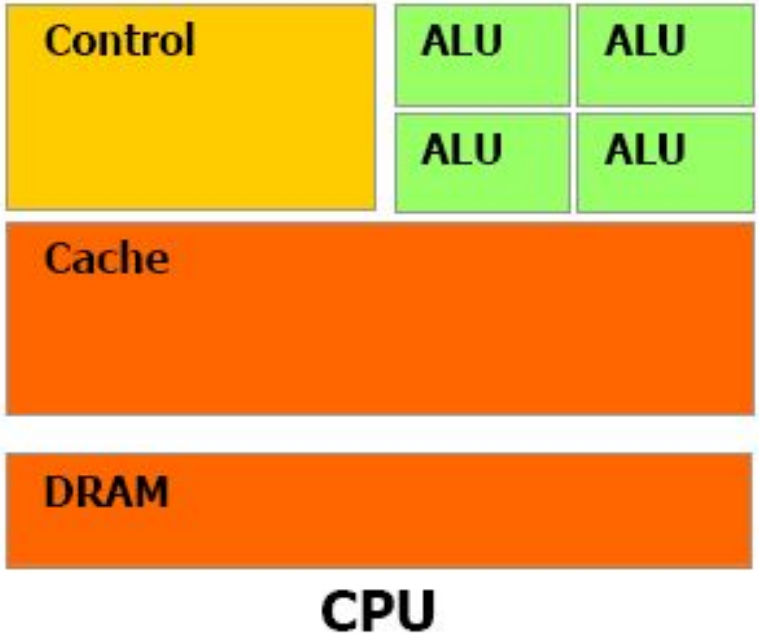
Mesh Render

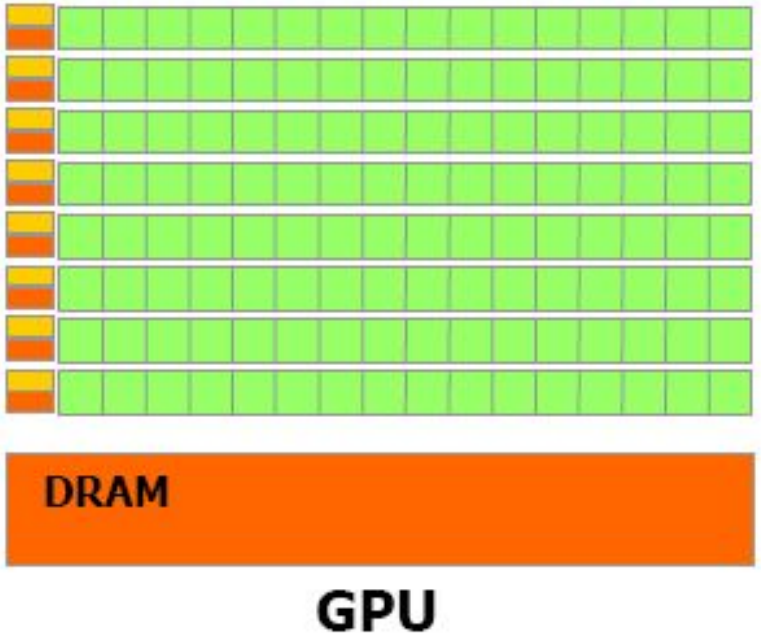

Scene
Wireframe
2D
Gizmos
Persp
Mesh


Game
Free Aspect
Maximize on Play
Mute audio
Stats
Gizmos
Render

CPU, GPU

CPU, GPU Compare



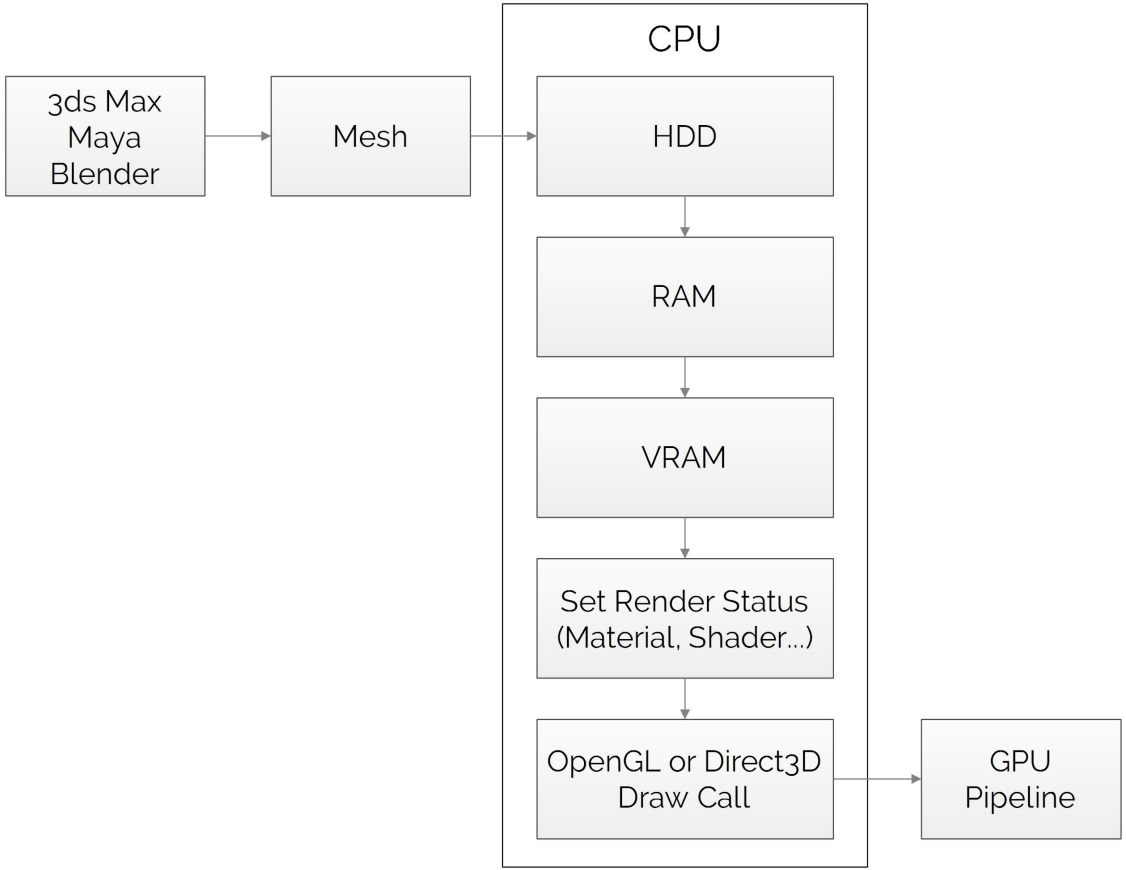


Reference:

[CUDA C Programming Guide]

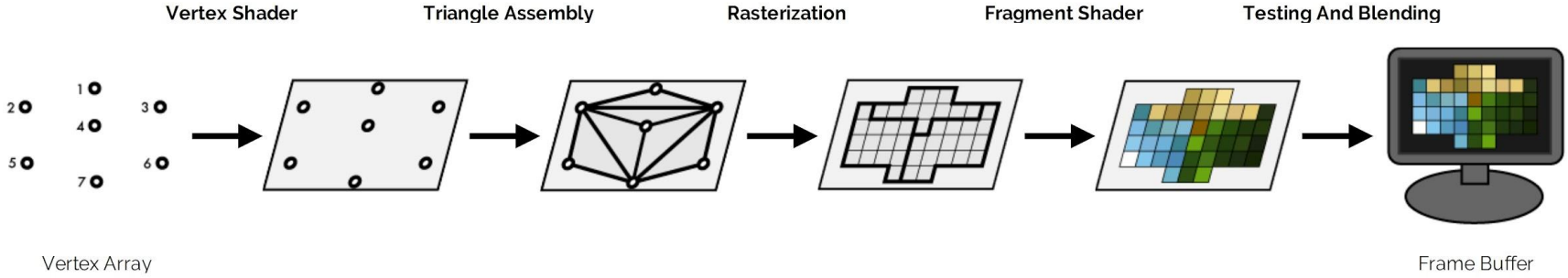
[CPU和GPU的设计区别]

CPU to GPU


3ds Max
Maya
Blender
Mesh
CPU
HDD
RAM
VRAM
Set Render Status
(Material, Shader...)
OpenGL or Direct3D
Draw Call
GPU
Pipeline

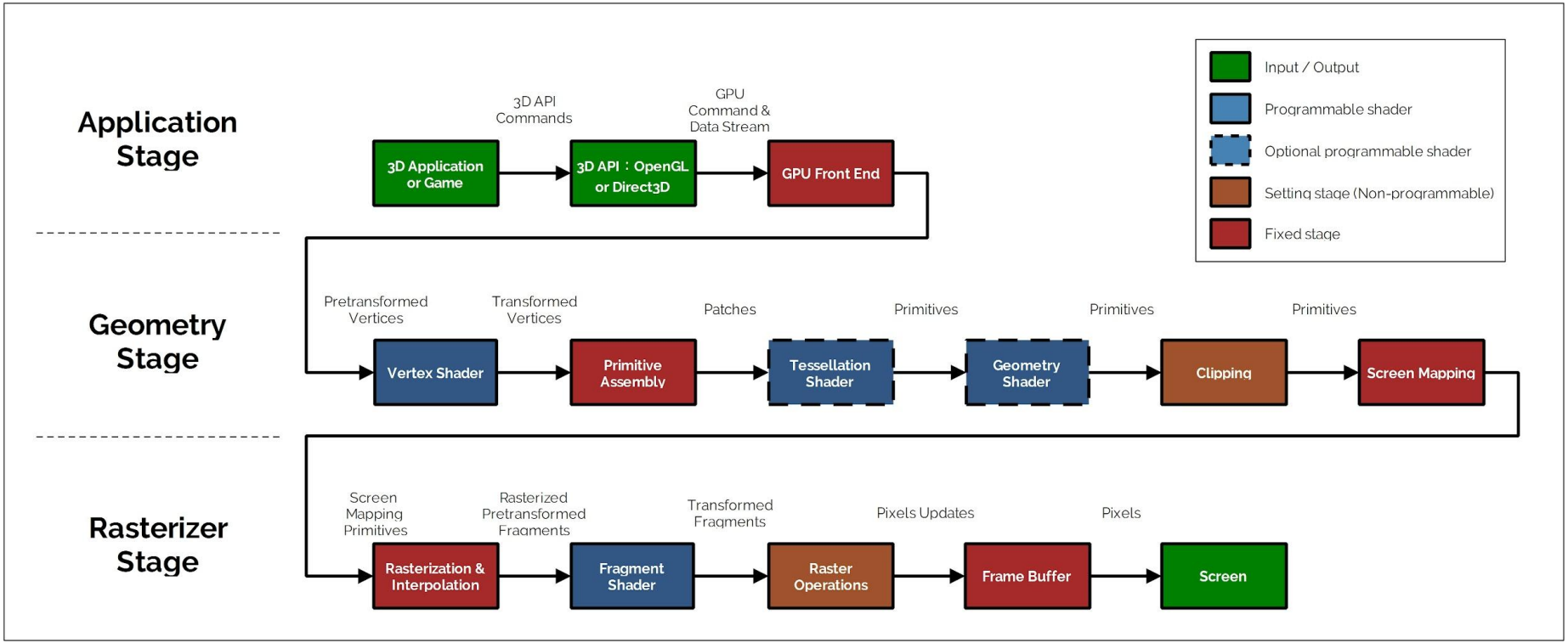
Graphics Pipeline

Simple Graphics Pipeline

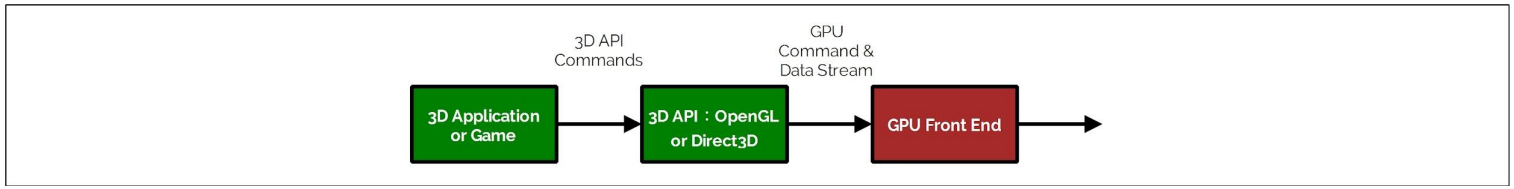


Reference:
joe's blog - OpenGL. Chapter 1

Programmable Graphics Pipeline

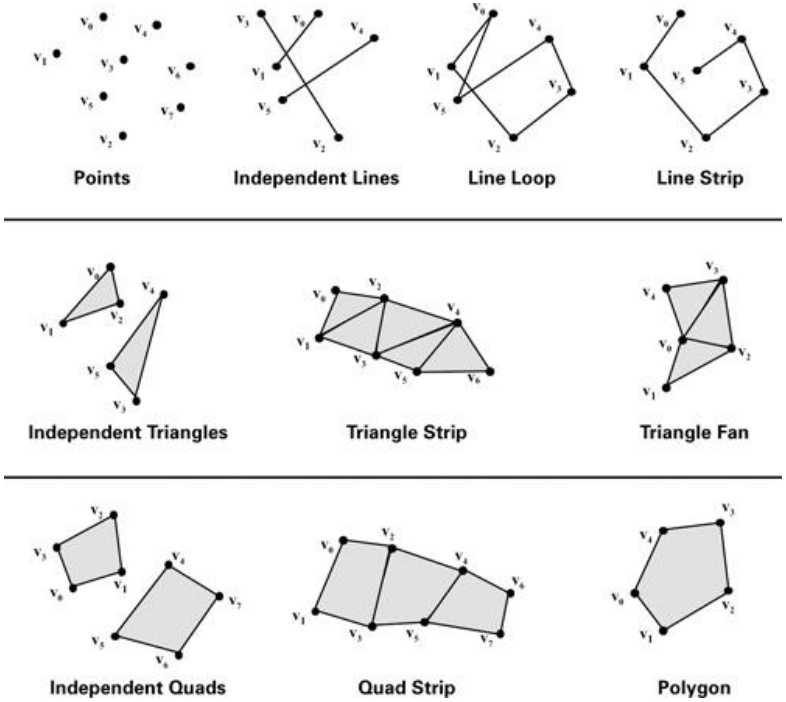


Application Stage



Vertex Data

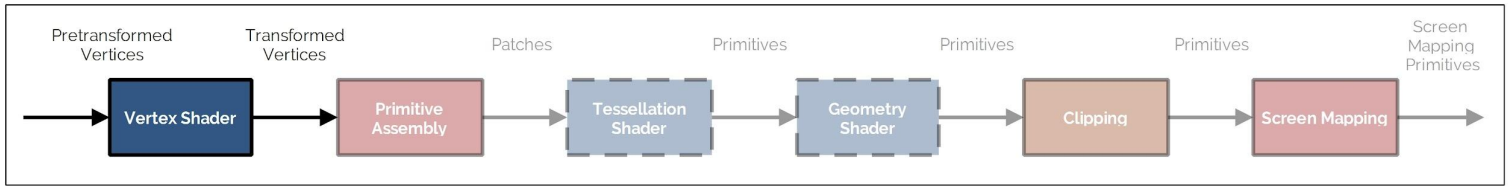
- Vertex position
- Vertex normal vector
- Vertex color
- Texture coordinate (UV)
- ....



Geometric Primitives

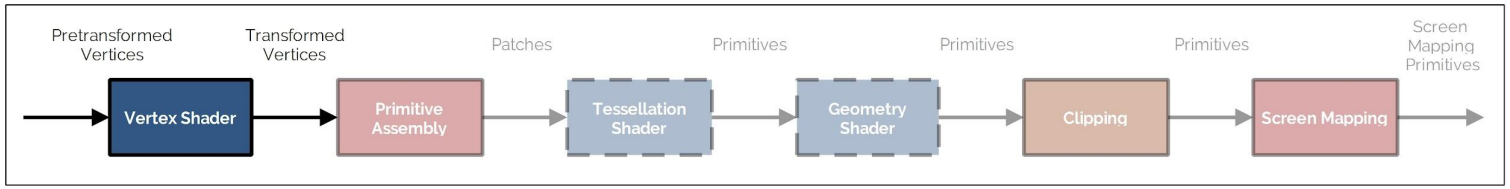
Reference:
Nvidia - The Cg Tutorial

Geometry Stage



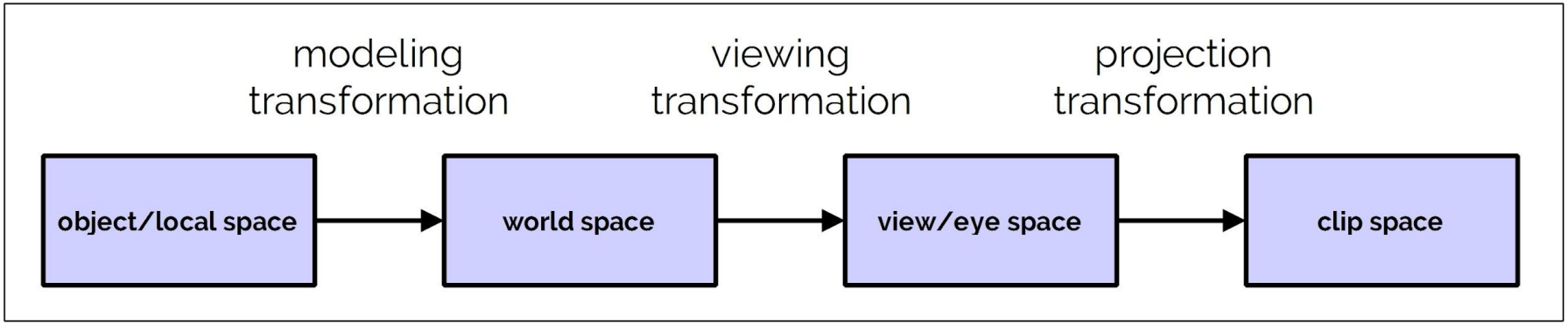
Vertex Shader

- Operates vertices
- Vertex transformations (Model-View-Projection)
- Texture coordinate transformations
- Per-vertex Lighting
- ....

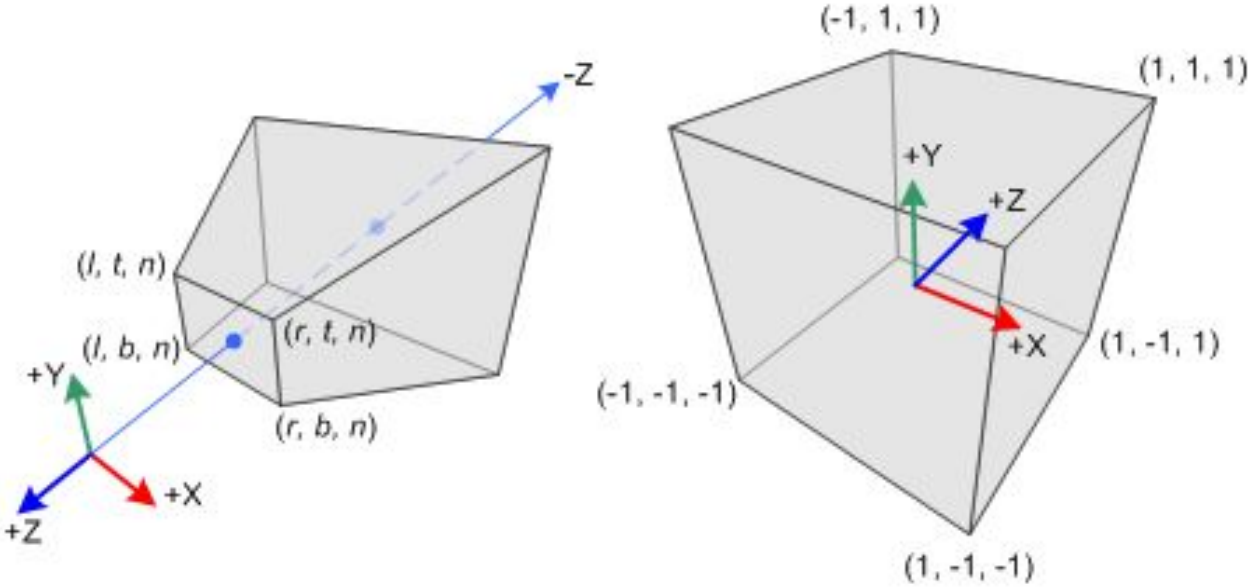


Vertex Shader

- Vertex transformations (Model-View-Projection)



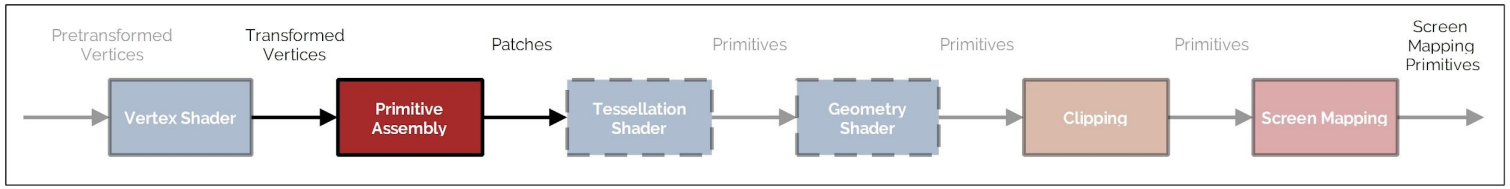
Perspective division



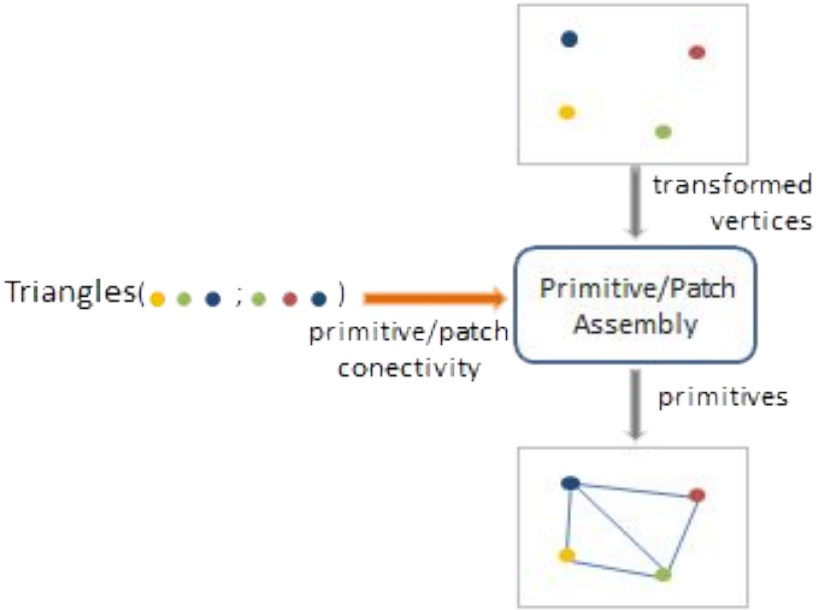
Perspective Frustum and Normalized Device Coordinates (NDC)

Reference:

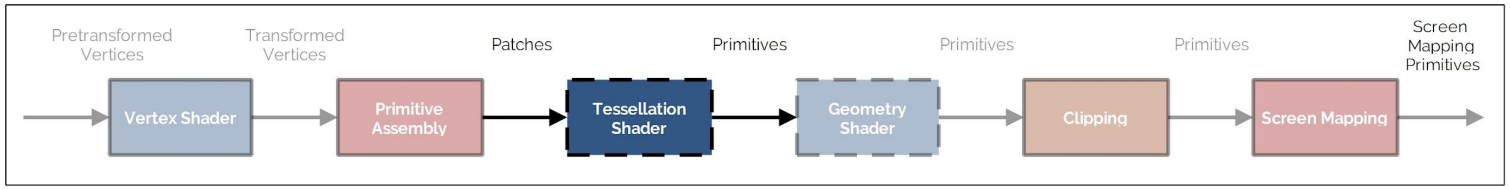
[OpenGL Projection Matrix]



Primitive Assembly



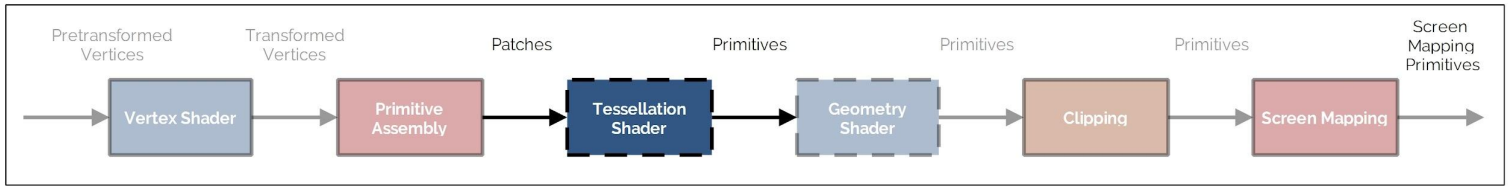
Reference:
Lighthouse3d - Primitive Assembly



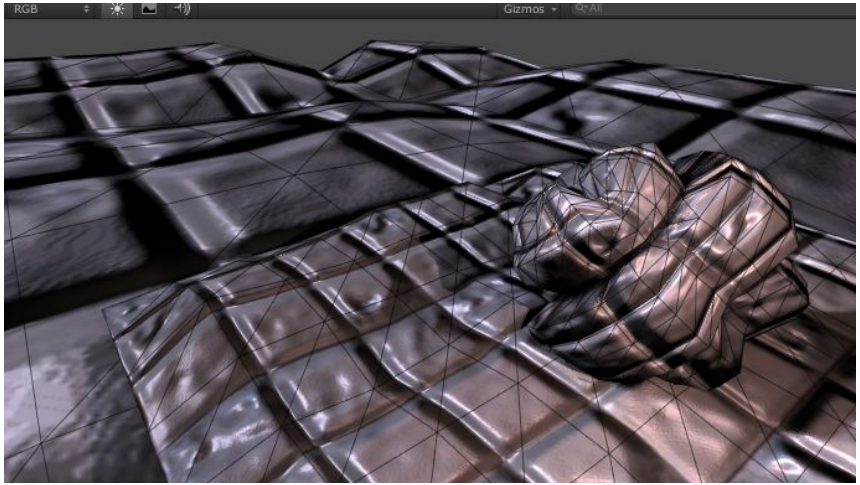
Tessellation Shader

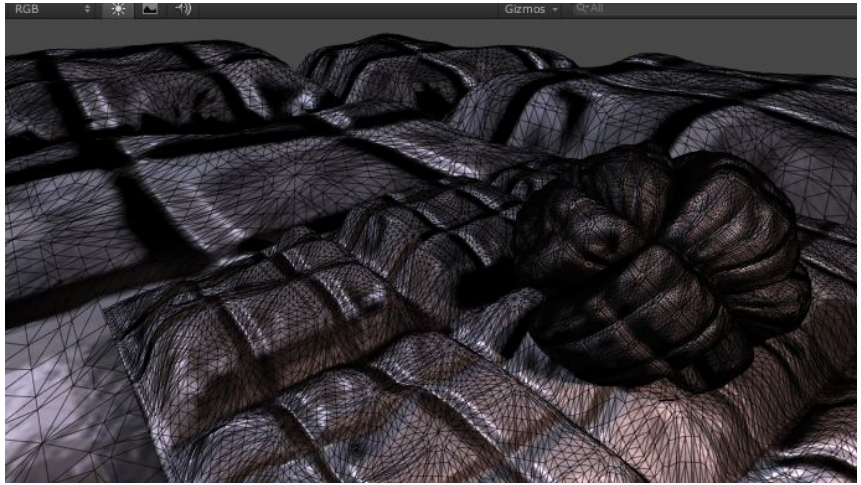
- Supported Direct3D 11、OpenGL 4、OpenGL ES 3.2
- Optional programmable shader
- Subdivided into smaller primitives
- ....

Reference:
[OpenGL - Tessellation]
[NVIDIA - DirectX 11 tessellation]



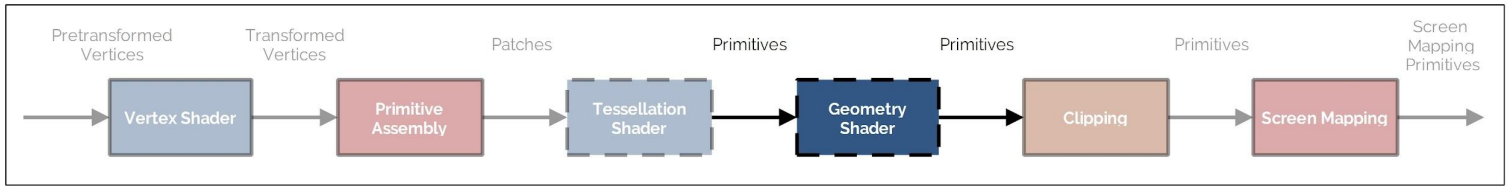
Tessellation Shader





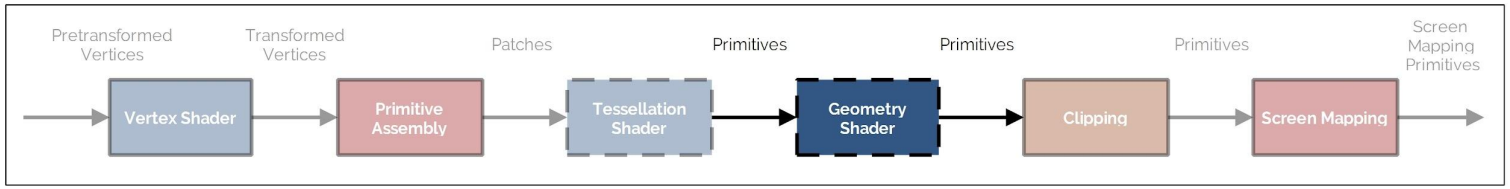
Reference:

[Surface Shaders with DX11 / OpenGL Core Tessellation]



Geometry Shader

- Supported Direct3D 10、OpenGL 3.2、OpenGL ES 3.2
- Optional programmable shader
- Add/Remove primitives
- Add/Remove vertices

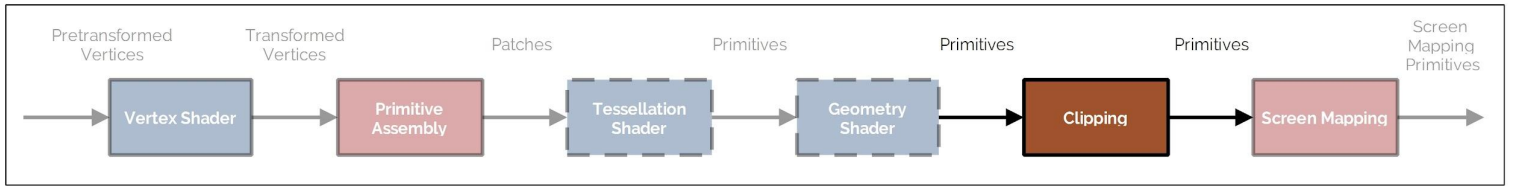


Geometry Shader

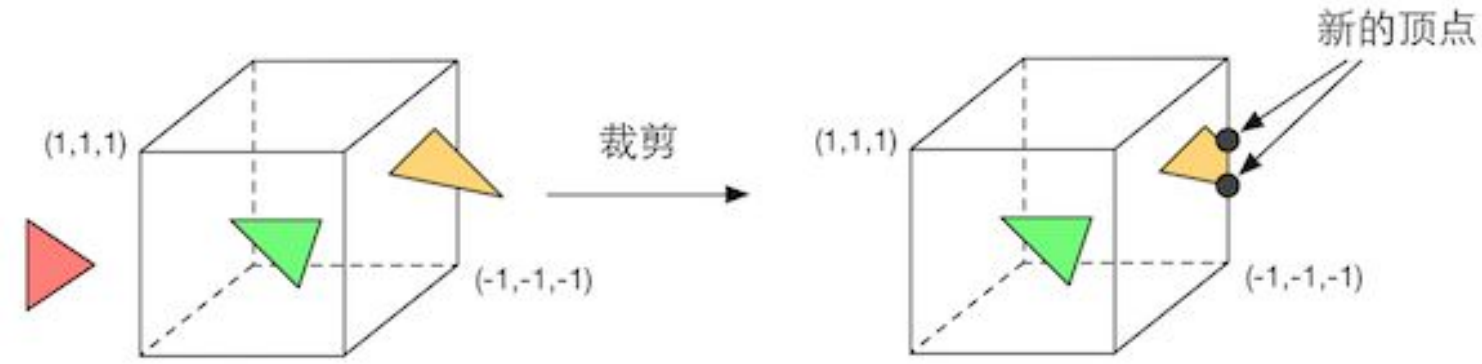
Reference:

Unity - Manual: DirectX 11 and OpenGL Core

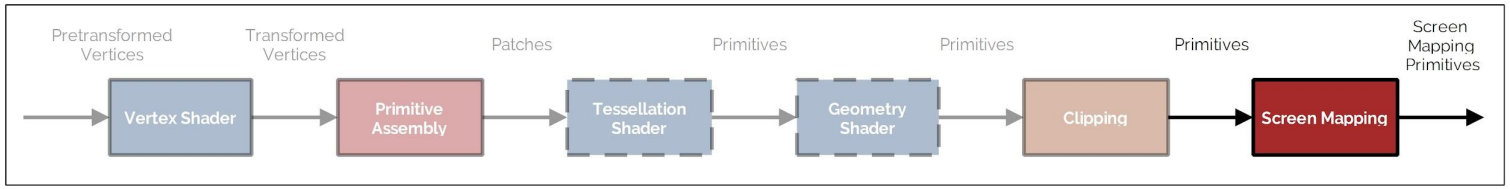
【风宇冲】Unity3D教程宝典之Shader篇：第二十七讲Geometry Shaders



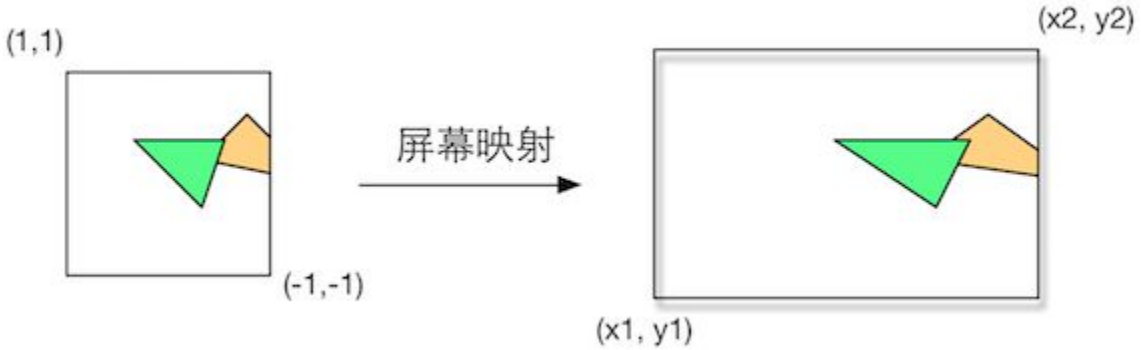
Clipping



Reference：
candycat1992/Unity_Shaders_Book:《Unity Shader入门精要》

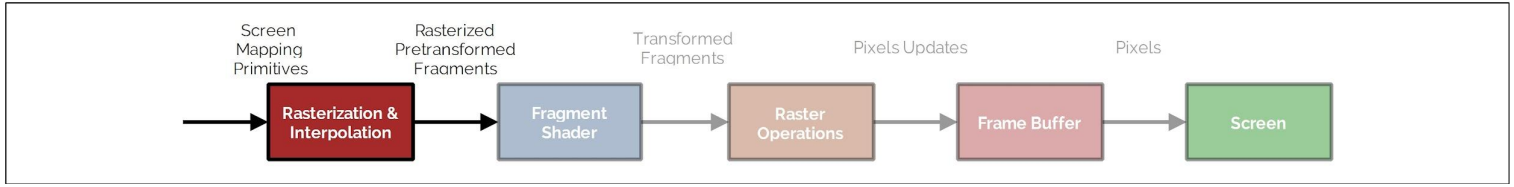


Screen Mapping



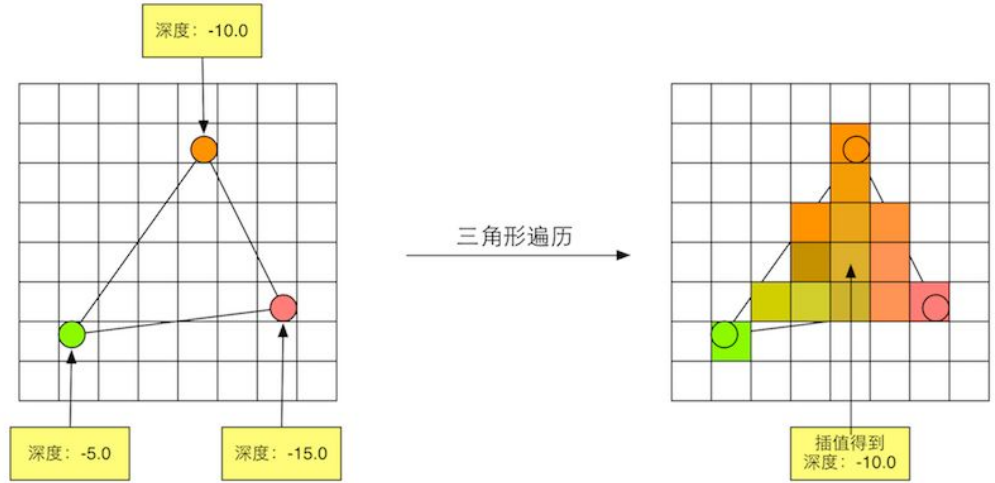
Reference：
candycat1992/Unity_Shaders_Book:《Unity Shader入门精要》

Rasterizer Stage

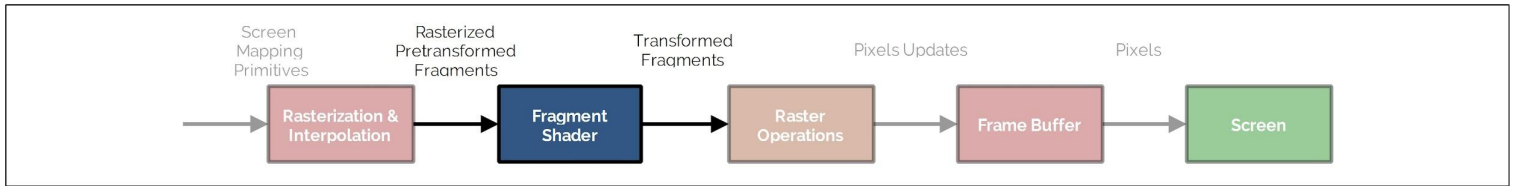


Rasterization & Interpolation

- Fragment Data
  - screen coordinate
  - color
  - depth
  - normal
  - texture coordinate (UV)
  - ...

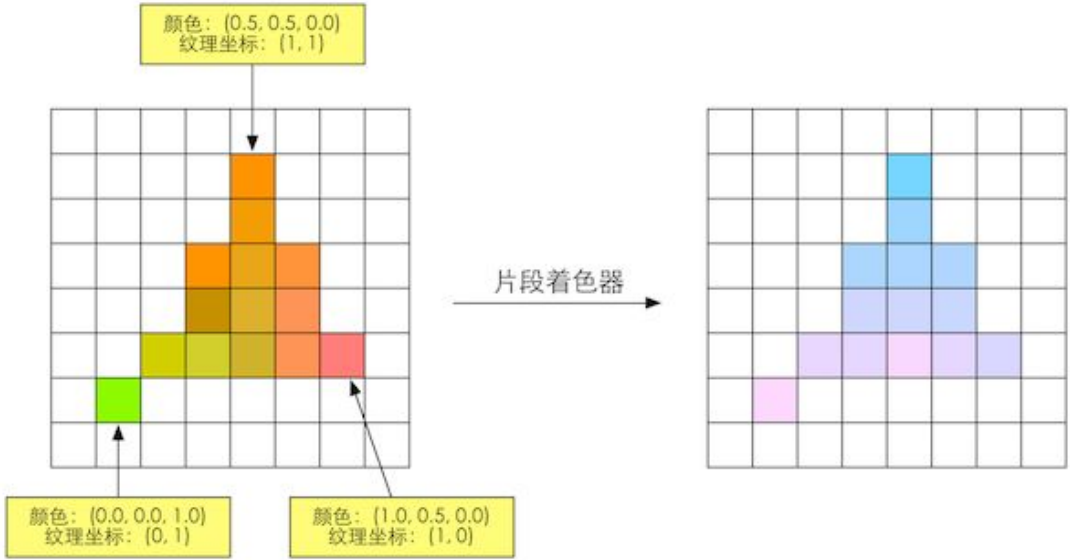


Reference：
candycat1992/Unity_Shaders_Book:《Unity Shader入门精要》

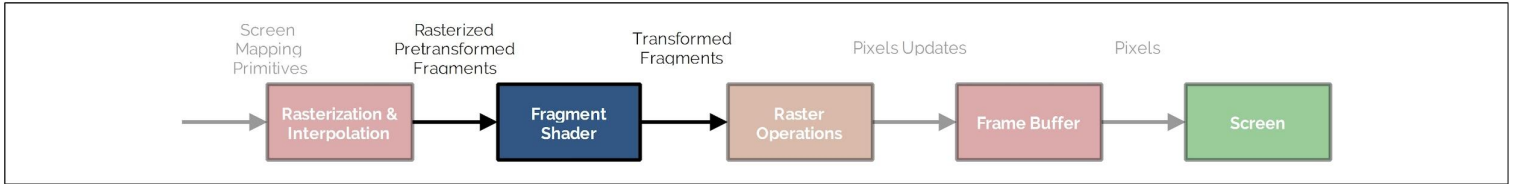


Fragment Shader

- Operates Fragment
- Texture mapping
- ...

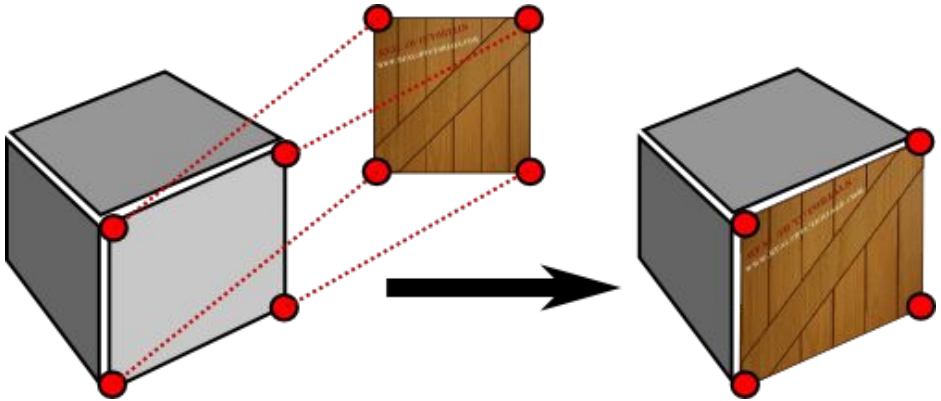


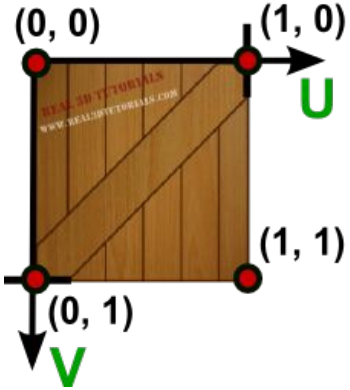
Reference：
candycat1992/Unity_Shaders_Book:《Unity Shader入门精要》

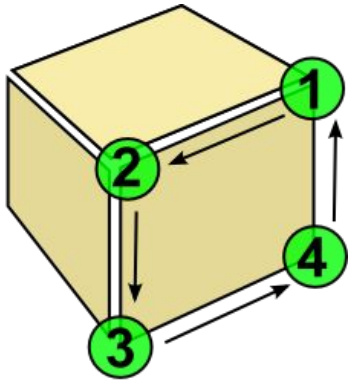


Fragment Shader

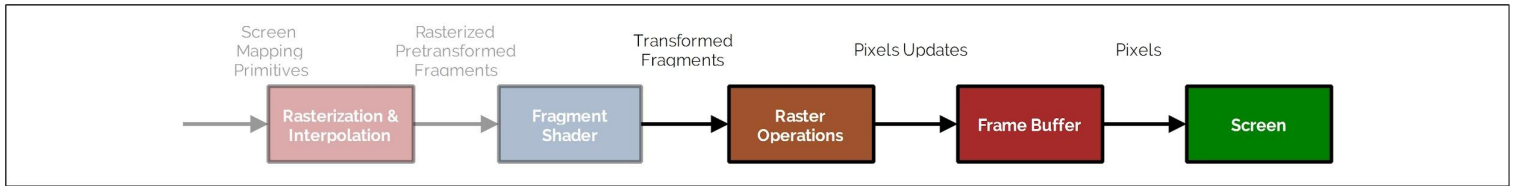
- Texture mapping





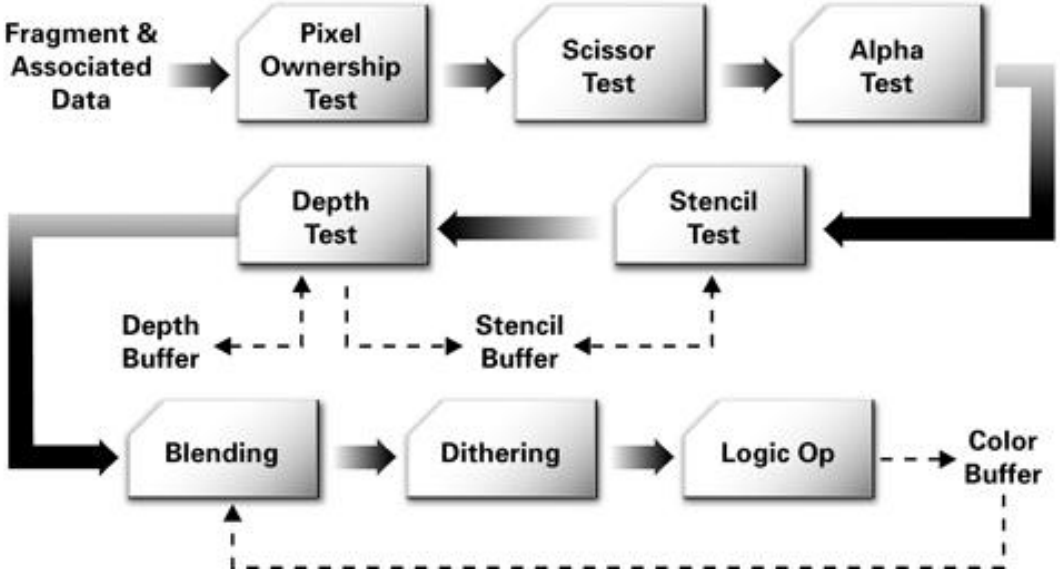


Reference:
Real 3D Tutorials: Tutorial 5 - Texture mapping



Raster Operations

- Early-Z
- Double Buffering
- ...



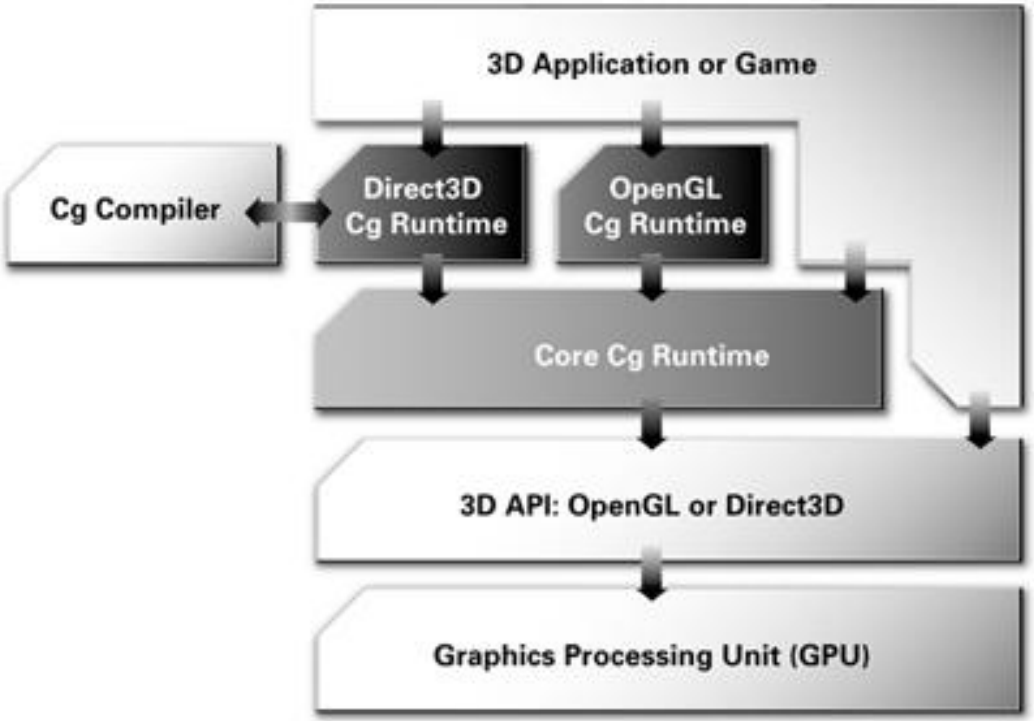
Reference:
Nvidia - The Cg Tutorial

Direct3D, OpenGL

Direct3D, OpenGL

Develop Company	3D API	Shading Language	Platform
SGI	OpenGL	GLSL (OpenGL Shading Language)	Windows、Linux、Mac、Mobile...
Microsoft	Direct3D	HLSL (High Level Shading Language)	Windows、Xbox360...
Microsoft + NVIDIA	OpenGL & Direct3D Top	Cg (C for Graphic)	Almost

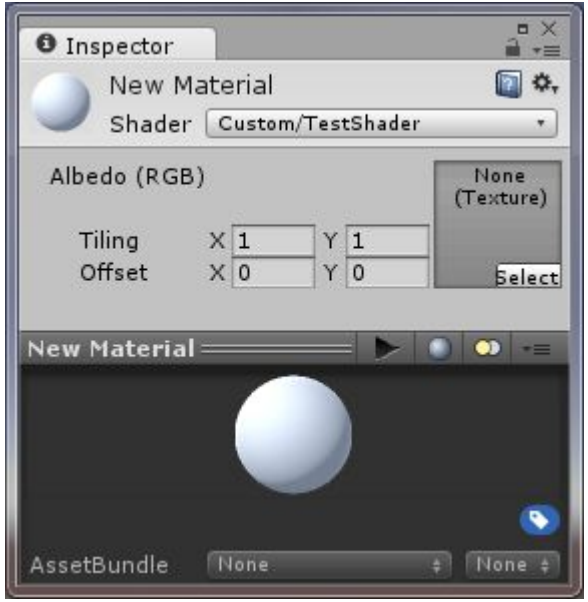
Cg Runtime Fits

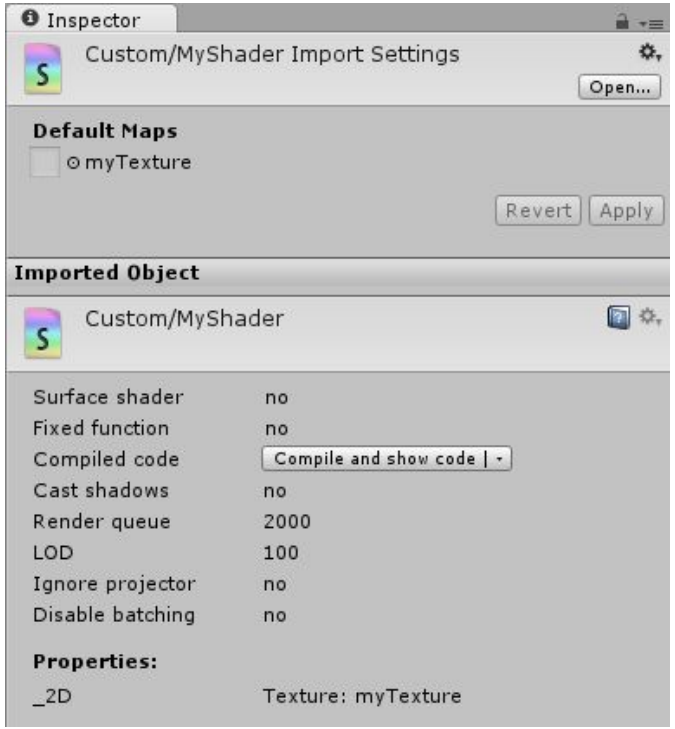


Reference:
Nvidia - The Cg Tutorial

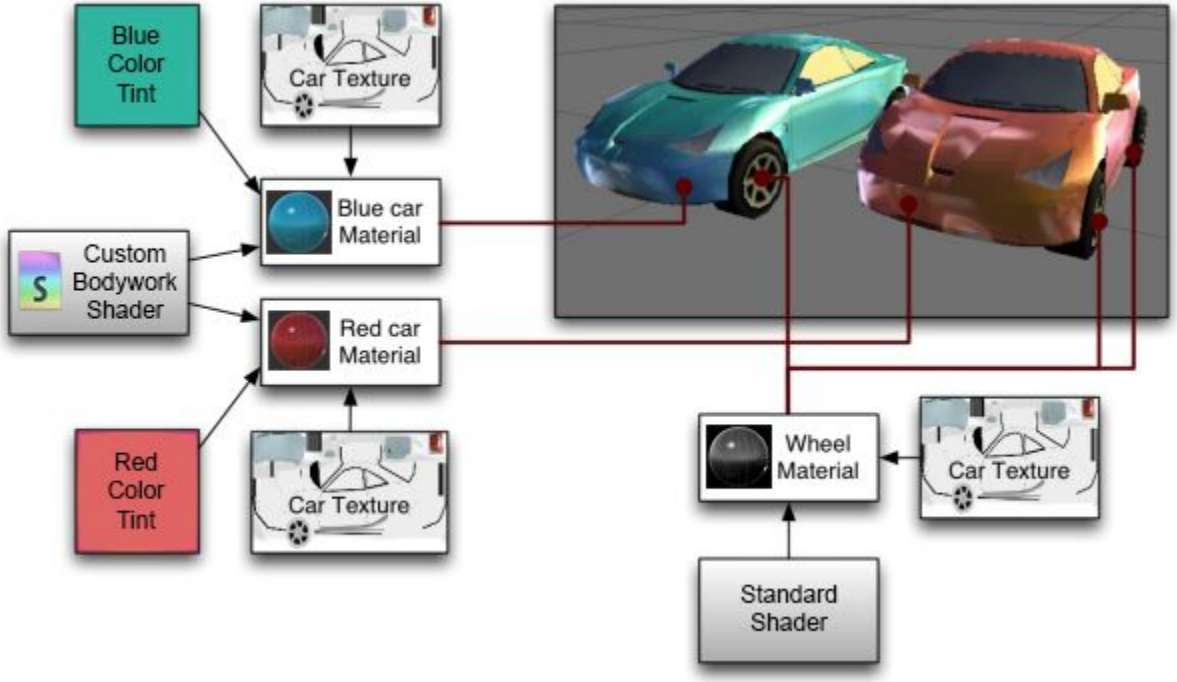
Material, Shader, Texture

Material, Shader, Texture





Material, Shader, Texture

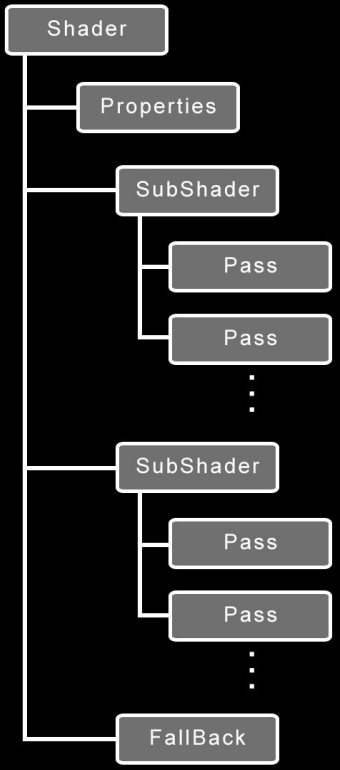


Reference:
Unity - Manual: Creating and Using Materials

Unity Shader Code Struct

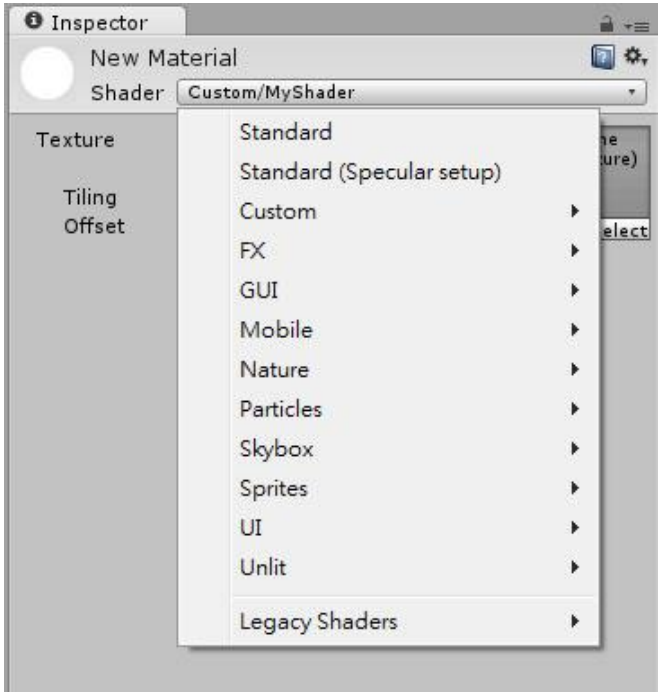
ShaderLab code struct

```
01  Shader "MyShader"
02  {
03    Properties
04    {
05    }
06
07    SubShader
08    {
09      Pass
10      {
11      }
12
13      Pass
14      {
15      }
16    }
17
18    SubShader
19    {
20      Pass
21      {
22      }
23
24      Pass
25      {
26      }
27    }
28
29    FallBack "Diffuse"
30  }
```



Shader Name

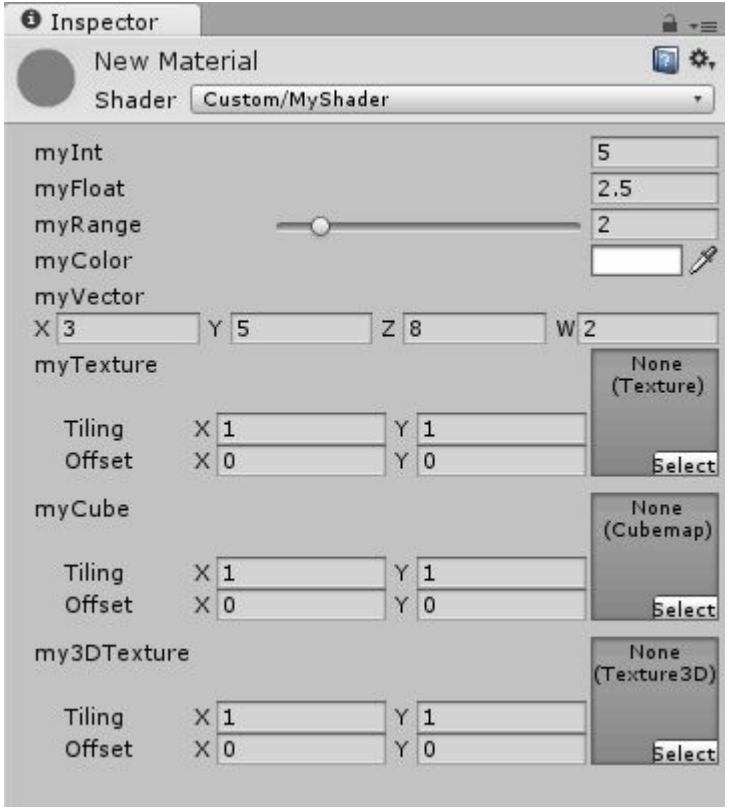
```
Shader "Custom/MyShader"
{
        ...
}
```



Shader Properties

_Name ("Display Name", type) = defaultValue[{options}]

```
Shader "Custom/MyShader"
{
	Properties
	{
		_Int ("myInt", Int) = 5
		_Float ("myFloat", Float) = 2.5
		_Range ("myRange", Range(1.5,5.5)) = 2.0
		_Color ("myColor", Color) = (1,1,1,1)
		_Vector ("myVector", Vector) = (3,5,8,2)
		_2D ("myTexture", 2D) = "white" {}
		_Cube ("myCube", Cube) = "" {}
		_3D("my3DTexture", 3D) = "" {}
	}
	...
}
```



Shader SubShader Tags

Subshader { [Tags] [LOD] [RenderSetup] Passdef [Passdef ...] }

Tags { "Queue" = "Geometry" "RenderType" = "Opaque" }

```
Shader "Custom/MyShader"
{
        Properties {...}
        SubShader
        {
                [Tags]
                [LOD]
                [RenderSetup]
                Pass {}
                ...
        }
}
```

Queue	渲染順序
RenderType	Shader分類，可用於 Shader Replacement 功能
DisableBatching	是否關閉批次處理，因可能在 local space 中針對 vertex 做動畫處理
ForceNoShadowCasting	是否會投射陰影
IgnoreProjector	是否受到 Projector 影響
CanUseSpriteAtlas	如果用於 Sprite 時設為 false，否則會與 SpritePacker 產生衝突不運作
PreviewType	inspector 上的 Material 預設模型，默認是圓形，可以改為 Plane 或是 SkyBox

Reference：

Unity - Manual: ShaderLab: SubShader Tags

Shader SubShader Tags Queue

```
Tags { "Queue" = "Geometry - 20" }

Tags { "Queue" = "Transparent + 100" }
```

order	Queue	Info
1000	Background	最早的渲染，用來渲染 Skybox或者背景
2000	Geometry	這是默認值，用來渲染非透明物體（普通情況下，場景中的 絕大多數物體應該是非透明的）
2450	AlphaTest	用來渲染經過 Alpha Test 的像素，單獨為 AlphaTest 設定一個 Queue 是在不透明體後渲染更高效
3000	Transparent	以深度值(Z)從後往前的順序渲染透明物體
4000	Overlay	用來渲染疊加的效果，是渲染的最後階段（比如鏡頭光暈等特效）

Reference：
Unity - Manual: ShaderLab: SubShader Tags

Shader SubShader Tags RenderType

Camera.RenderWithShader

Camera.SetReplacementShader

Queue	Info	Queue	Info
Opaque	不透明	TreeOpaque	樹木不透明
Transparent	透明	TreeTransparentCutout	樹木透明鏤空
TransparentCutout	透明鏤空	TreeBillboard	樹木布告牌
Background	背景	Grass	草地
Overlay	疊加	GrassBillboard	草地布告牌

Reference：

Unity - Manual: Rendering with Replaced Shaders

Billboard 和粒子 | OpenGL Tutorials

Shader SubShader LOD

Subshader { [Tags] [LOD] [RenderSetup] Passdef [Passdef ...] }

```
Shader "Custom/MyShader"
{
        Properties {...}
        SubShader
        {
                [Tags]
                [LOD]
                [RenderSetup]
                Pass {}
                ...
        }
}
```

100	VertexLit kind of shaders
150	Decal, Reflective VertexLit
200	Diffuse
250	Diffuse Detail, Reflective Bumped Unlit, Reflective Bumped VertexLit
300	Bumped, Specular
400	Bumped Specular
500	Parallax
600	Parallax Specular

Reference:

Unity - Manual: Shader Level of Detail

Unity - Manual: Normal Shader Family

Shader SubShader RenderSetup

Subshader { [Tags] [LOD] [RenderSetup] Passdef [Passdef ...] }

```
Shader "Custom/MyShader"
{
        Properties {...}
        SubShader
        {
                [Tags]
                [LOD]
                [RenderSetup]
                Pass {}
                ...
        }
}
```

Cull	*Back* | *Front* | *Off*
ZTest	*Less* | *Greater* | *LEqual* | *GEqual* |*Equal* | *NotEqual* | *Always*
ZWrite	*On* | *Off*
Blend	*SrcFactor DstFactor*
...	**...**

Shader SubShader Pass

Pass { [Name and Tags] [RenderSetup] }

```
Shader "Custom/MyShader"
{
        Properties {...}
        SubShader
        {
                [Tags]
                [LOD]
                [RenderSetup]
                UsePass {}
                GrabPass {}
                Pass {}
                ...
        }
}
```

UsePass	Name "MYNAME" UsePass "Custom/MyShader/MYNAME"
GrabPass	GrabPass {"Texture Name"}
LightMode	光照渲染方式 (Forward、Deferred、VertexLit)
RequireOptions	滿足要求選項的條件時才渲染，目前只有 SoftVegetation

Reference：

Unity - Manual: ShaderLab: Pass Tags

Unity - Manual: Rendering Paths

Unity - Manual: Graphics Command Buffers

Unity Shader Category

Fixed Function Shader

- Code
- 不可程式碼編輯 (選項式)
- 需完全使用 ShaderLab 設定命令
- 舊設備 (Direct 7.0、OpenGL 1.5、OpenGL ES 1.1)
- Unity 5.2+ 自動轉換成 Vertex / Fragment Shader
- 簡單效果 (貼圖顏色混合、簡單光照)
- ...

Reference：
Unity - Manual: Fixed function shaders

Surface Shader

- Code
- 可程式碼編輯 (CGPROGRAM...ENDCG)
- 編譯成多個 Pass
- 編譯成 Vertex / Fragment Shader
- 封裝了很多光影處理細節與光照模式
- ...

Reference：
Unity - Manual: Writing Shaders
Unity - Manual: Shader Compilation Target Levels

Vertex and Fragment Shader

- Code
- 可程式碼編輯 (CGPROGRAM...ENDCG)
- 最靈活彈性，但語法複雜
- 特別效果 (頂點位置改變、部分顏色混合)
- ...

Reference：

Unity - Manual: Writing vertex and fragment shaders

Unity - Manual: Shader data types

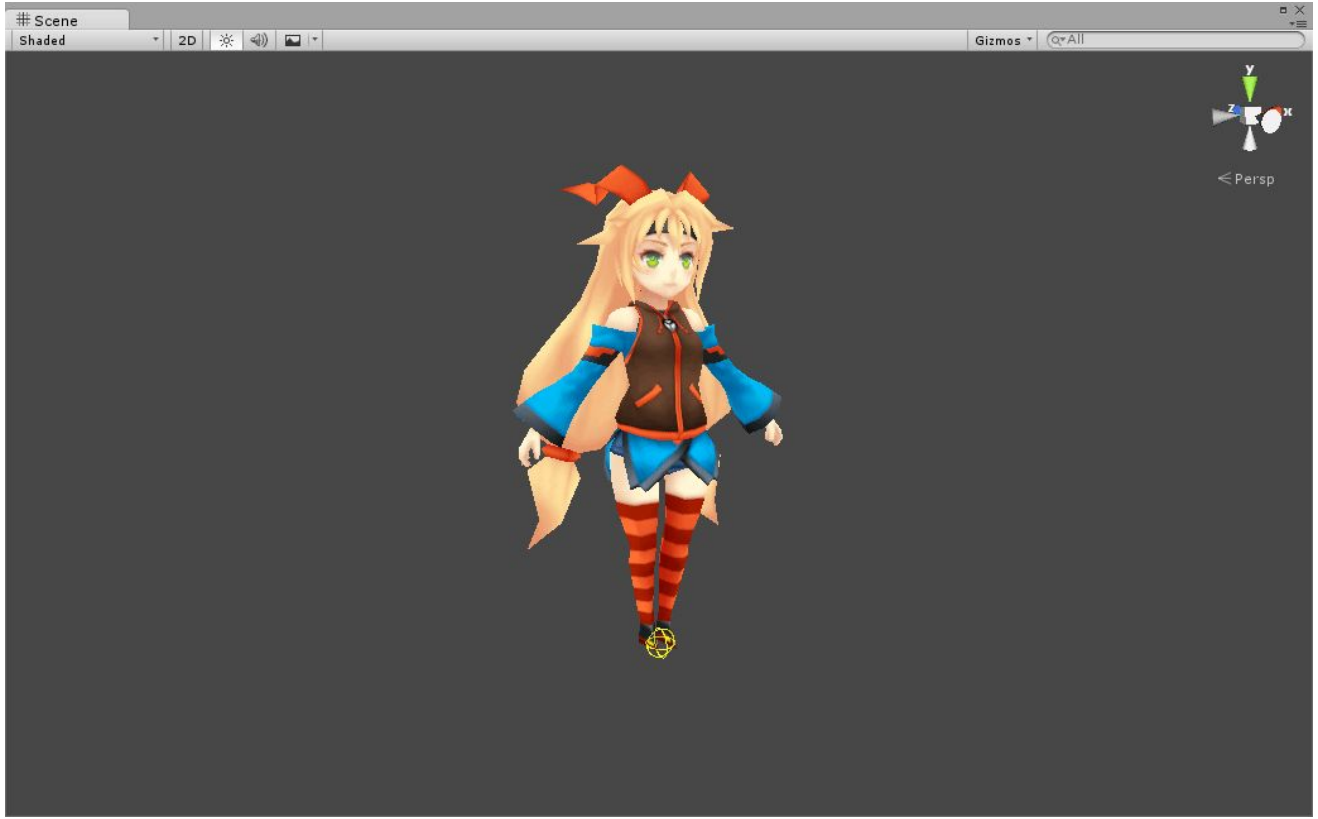
Semantics (Windows)

Implementation

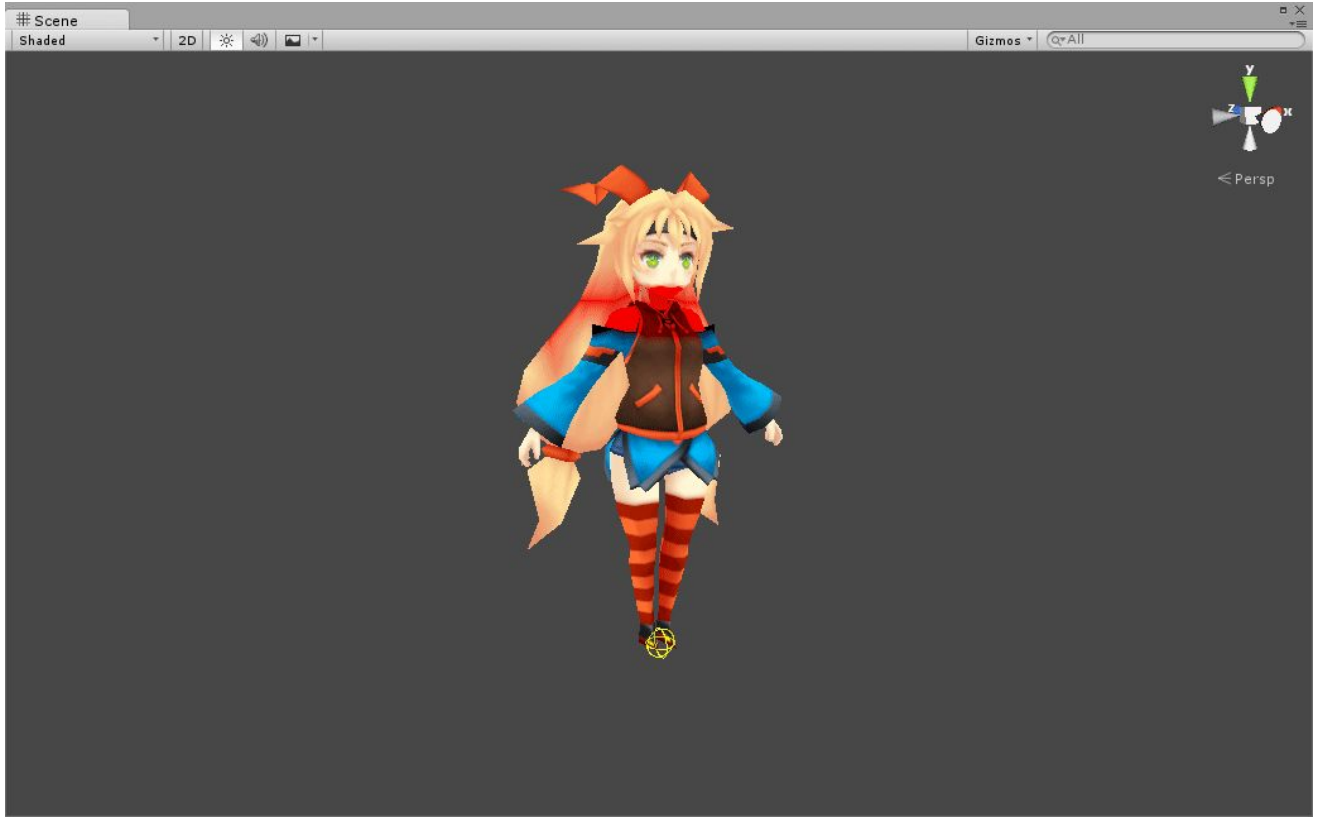
Vertex and Fragment Shader

Github

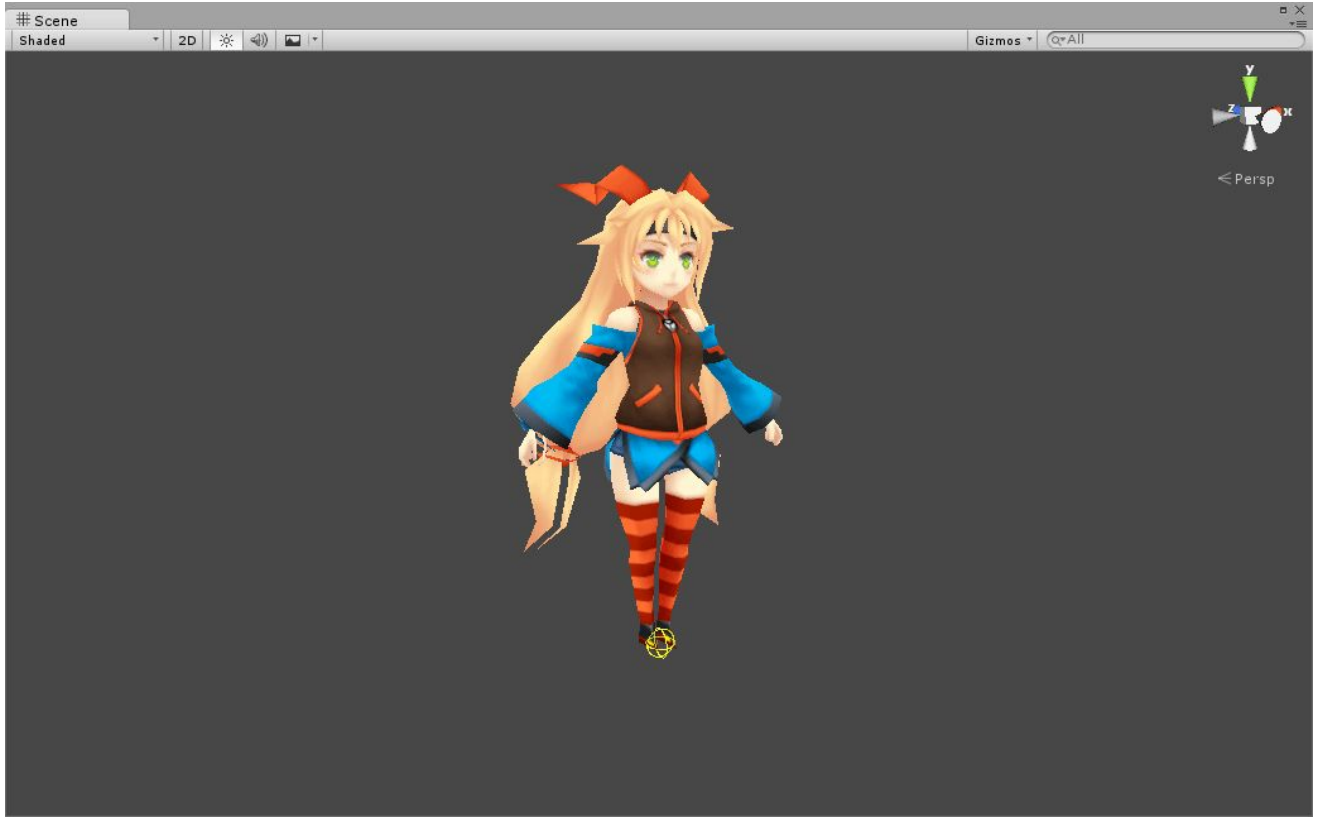
Direction Offset



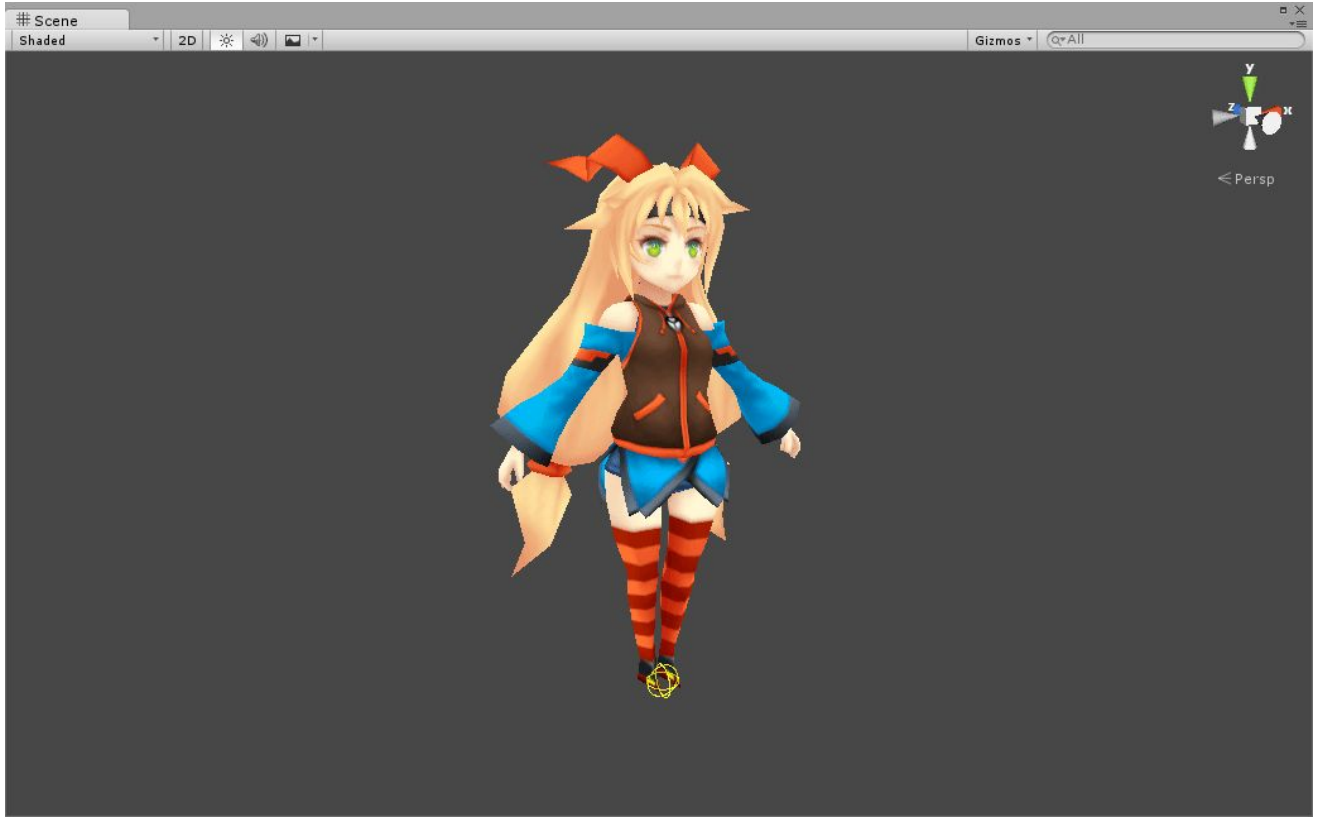
Streamer Color



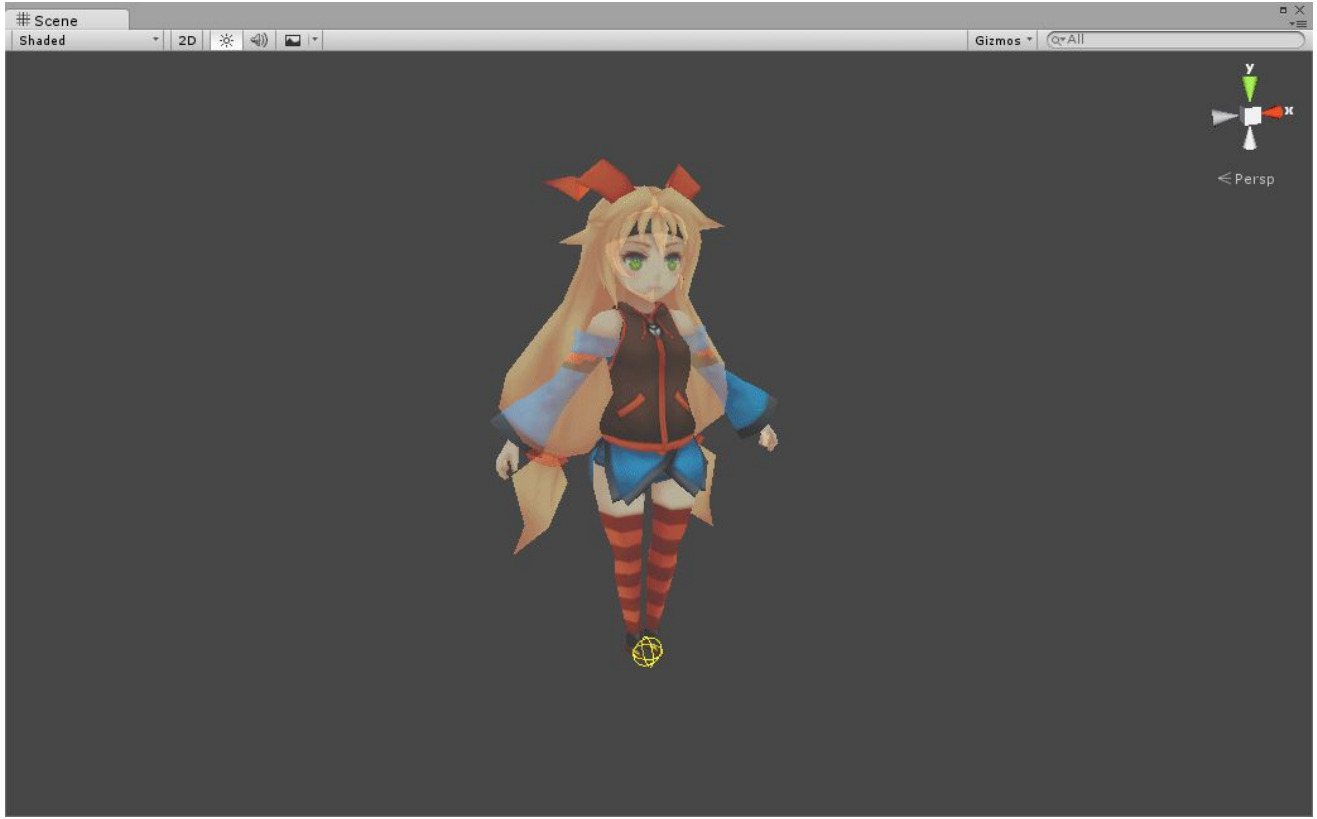
Direction Offset Clamp



Outline



Fade



Q & A

Thanks ~