**Lit**

1. **Lit计算结果由什么组成？**

UniversalFragmentPBR + ***MixFog(略)***

1. **UniversalFragmentPBR如何计算？**

GlobalIllumination + 主光源LightingPhysicallyBased + 额外光LightingPhysicallyBased(逐像素) / inputData.vertexLighting \* brdfData.diffuse(逐顶点) + ***emission(略)***

1. **GlobalIllumination如何计算？**

GlobalIllumination = EnvironmentBRDF = indirectDiffuse \* brdfData.diffuse + indirectSpecular \* EnvironmentBRDFSpecular

1. **indirectDiffuse如何计算？**

bakedGI \* ***occlusion***

1. **bakedGI如何计算？**

SAMPLE\_GI(input.lightmapUV, input.vertexSH, inputData.normalWS)

1. **SAMPLE\_GI如何计算？(GI)**

SampleLightmap或者***SampleSHPixel(略)***

1. **SampleLightmap如何计算？**

SAMPLE\_TEXTURE2D\_LIGHTMAP(unity\_Lightmap, samplerunity\_Lightmap, lightmapUV).rgb;

1. **brdfData.diffuse如何计算？**

oneMinusReflectivity = 0.96 - metallic \* 0.96;

albedo = \_BaseMap.rgb \* \_BaseColor.rgb

albedo \* oneMinusReflectivity

1. **indirectSpecular如何计算？**

indirectSpecular = GlossyEnvironmentReflection = SAMPLE\_TEXTURECUBE\_LOD(天空盒) \* ***occlusion***

1. **EnvironmentBRDFSpecular如何计算？**

float surfaceReduction = 1.0 / (brdfData.roughness2 + 1.0);

return surfaceReduction \* lerp(brdfData.specular, brdfData.grazingTerm, fresnelTerm);

1. **brdfData.specular如何计算？**

lerp(kDieletricSpec.rgb, albedo, metallic)

kDieletricSpec = half4(0.04, 0.04, 0.04, 1.0 - 0.04)

1. **brdfData.grazingTerm如何计算？**

saturate(smoothness + reflectivity)

reflectivity ≈ metallic

1. **fresnelTerm如何计算？**

half NoV = saturate(dot(normalWS, viewDirectionWS));

half fresnelTerm = Pow4(1.0 - NoV);

1. **LightingPhysicallyBased如何计算？**

brdf = brdfData.diffuse + brdfData.specular \* DirectBRDFSpecular

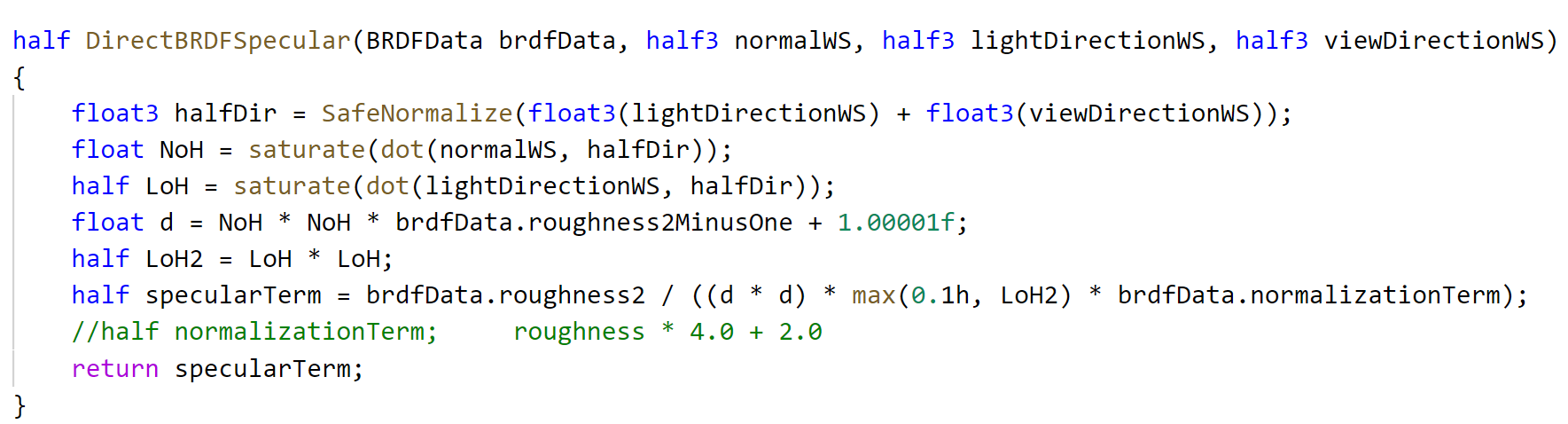
NdotL = saturate(dot(normalWS, lightDirectionWS))

radiance = lightColor \* (lightAttenuation \* NdotL)

brdf \* radiance

1. **DirectBRDFSpecular如何计算？**

roughness2 / ( NoH2 \* (roughness2 - 1) + 1 )2 \* (LoH2 \* (roughness + 0.5) \* 4.0)



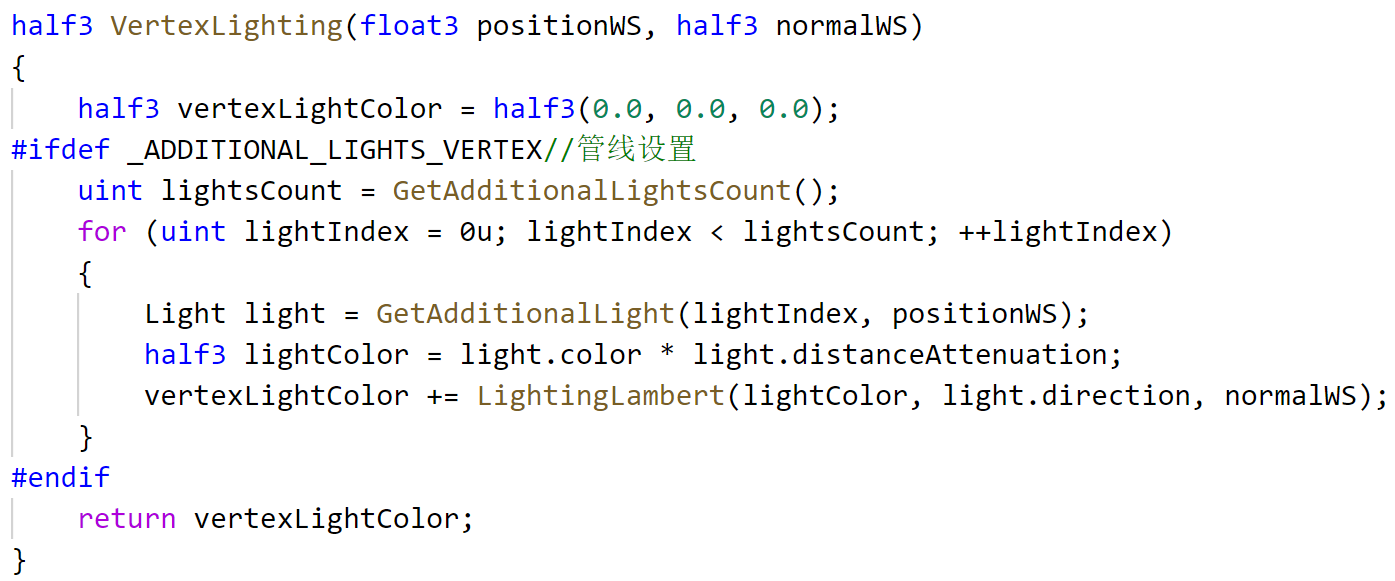
1. **lightAttenuation如何计算？**

light.distanceAttenuation \* ***light.shadowAttenuation***

1. **light.distanceAttenuation如何计算？**

DistanceAttenuation \* ***AngleAttenuation(略)***

1. **inputData.vertexLighting如何计算？**



**21，逐顶点，逐像素和逐对象**

**22,** 额外光LightingPhysicallyBased计算时，光源是如何确定的？