

Developer Support Engineer Interview test Answers

1. Please solve the following problems. You can use the Unity Documentation, Scripting Reference, Stack Overflow, Google, etc:

1.1.

Briefly, we need to extend the shader to receive the lighting information from the main lighting in addition to the illumination data from Light Probes .

We just have to use the **ShadeSH9** function from **UnityCG.cginc** include file that will do all the work.

```
Shader "MyShader/Diffuse With LightProbes" {
    Properties{ [NoScaleOffset] _MainTex("Texture", 2D) = "white" {} }
    SubShader{
        Pass{
            Tags {
                "LightMode" = "ForwardBase"
            }
            CGPROGRAM
            #pragma vertex v
            #pragma fragment f

            #include "UnityCG.cginc" // for UnityObjectToWorldNormal
            #include "UnityLightingCommon.cginc" // for _LightColor0
            sampler2D _MainTex;

            struct v2f
            {
                float2 uv : TEXCOORD0;
                float4 vertex : SV_POSITION;
                fixed4 diff : COLOR0; // diffuse lighting color
            };

            v2f v(appdata_base vertex_data)
            {
                v2f o;
                o.vertex = UnityObjectToClipPos(vertex_data.vertex);
                o.uv = vertex_data.texcoord;

                // get vertex normal in world space
                half3 worldNormal = UnityObjectToWorldNormal(vertex_data.normal);

                // dot product between normal and light direction for
                // standard diffuse (Lambert) lighting
                half nl = max(0, dot(worldNormal, _WorldSpaceLightPos0.xyz));

                // factor in the light color
                o.diff = nl * _LightColor0;

                // ShadeSH9 function from UnityCG.cginc evaluates
                // illumination from ambient or light probes,
                // using world space normal

                //Here we add it to the diffuse lighting from the main light
                o.diff.rgb += ShadeSH9(half4(worldNormal, 1));
                return o;
            }

            fixed4 f(v2f input_fragment) : SV_Target
            {
                fixed4 col = tex2D(_MainTex, input_fragment.uv);

                // multiply by lighting
                col *= input_fragment.diff;
                return col;
            }
        }
    }
}
```

2. Please, try to answer the following questions in your own words:

2.1 Describe what each of these technologies are and what they can be used for:

2.1.1 Scriptable Build Pipeline

We must start with the **meaning of scriptable**, which in this scope means that the element involved can be modified, extended and improved through scripts.

With this in mind the SBP is **an alternative to the other Build Pipeline** methods that Unity uses, presented in a C # package and whose main advantage is a pre-defined build flow for building AssetBundles that provides greater flexibility and speed in the process.

2.1.2 Scriptable Render Pipeline

Is a feature that **allows users to control the rendering process through C # scripts**. An SRP, it is the core of the popular Universal Render Pipeline (URP), High Definition Render Pipeline (HDRP) or (LWRP).

2.1.3 Addressables

It is an **asset with a path or "address"** assigned so that **it can be loaded** into the unity application **at the time it is requested**.

2.1.4 IL2CPP

There is a concept in C # in which the script can be converted to an **intermediate language** that is neither source code nor machine language.

IL2CPP is an alternative **backend scripting** (which drives unity scripting) that offers advantages such as speed and stability for specific platforms.

2.1.5. Nested Prefabs

A prefab can be made up of other prefabs. These are called nested prefabs and retain the binding and properties of their original prefab and the benefits of the prefab system.

2.2. Mention at least two problems of Unity's non-incremental Garbage Collector.

Slow operations as GC spikes, or the need to increase the reserved memory space which is another slow operation.

2.3. Explain which of these is better and why? Unity LTS, TECH release, Beta or Alpha?

In my opinion, both the LTS and the Techstream are the best option for these two scenarios respectively: if we are looking for a **stable product with support**, **LTS** is definitely the best choice, but if the goal is to **explore, experiment** and satisfy curiosity, the **Tech stream** is your guy.

2.4. What is your preferred version control system and why do you prefer it over others?

To be honest, Git is fast, versatile, and has worked well for me, as well as the love I feel for Github. I do not have experience with many more version control systems so I am not going to give a broader personal opinion.

2.5. What is your favorite IDE and why?

Definitely **Visual Studio**. Because it offers a giant versatility, a **wide number of platforms and languages**, and amazing stability and performance, as well as the wonderful **Unity integration**, which is the framework I use every day.

2.6. What issues or limitations have you recently experienced using Unity?

The last issue that I remember was the shader variants compiling time when using LWRP, and other small issues that always got fixed updating the Unity version.

2.7. What strategies or best practices can be used to optimize the CPU and GPU usage in an application made with Unity

There are a lot of strategies and good practices in unity to improve performance. You can start with doing good research and planning for the project, making use of version control. Profiling your project frequently, especially when implementing a new component; review memory and resource consumption. Regarding assets, be careful to import them correctly and use the most optimal sizes and formats (textures, video, audio, 3DModels), use Asset Bundles, and take into consideration the target platform or platforms of the project. Regarding scripting: use good programming practices, use object pools, understand the Player Loop, be careful with real-time instances and unoptimized code. Take care of the interactions between objects with physics. Regarding the GPU and graphics it is a good idea to use LODs, forward rendering, use optimal shaders, consider proper batching and careful draw calls, use VFX, Jobs, and so on. And finally put yourself in the user's shoes to anticipate possible issues.

2.8. How do you catch and investigate crashes happening in a released game?

It is important to try to imagine which component may be failing based on the behavior of the fault. Attach debuggers to the project and developer builds are useful strategies. For instance, android logcat, or running the app connected to the console with usb provides important information in real time about the app functioning.

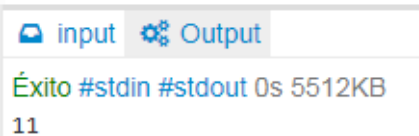
2.9. Compare the following function and macro definitions. In what cases will they produce different results and/or side effects?

Without a correct manage the macros can lead to issues and unexpected behaviors (calculations) , inappropriate uses in wrong scopes, and so on.

For instance:

Macro output with an input of $2 + 3$ because compiler replace the macro definition with $2+3*2+3$:

```
1  #include <stdio.h>
2
3  #define square(val) (val*val)
4
5  int main() {
6      printf("%d\n", square(2 + 3));
7      return 0;
8  }
9
```



input Output

Éxito #stdin #stdout 0s 5512KB

11

2.10. What is the package manager in Unity and what is the alternative way of adding a package than via the package manager UI?

It's an **interface** for **visualizing**, **controlling** and **installing** packages in the Unity editor with a **version control system**. You can either download packages from sources as Unity Github and install these packages importing directly into the Unity project in the Unity editor interface with the contextual menu, or even draggin the package from explorer to the editor.

2.11. Examine the following function. What does it accomplish?

Use Bit manipulation to check if a given number is a power of 2.

References:

1.

- <https://docs.unity3d.com/Manual/SL-VertexFragmentShaderExamples.html>
- <https://programmerah.com/call-unity-with-lightmap-and-light-probes-in-shader-13701/>
- <https://forum.unity.com/threads/job-system-output-scalar-value-from-job.512194/>
- <https://docs.unity3d.com/ScriptReference/Color.html>
- <https://docs.unity3d.com/Manual/com.unity.jobs.html>
- https://www.youtube.com/watch?v=C56bbgtPr_w&t=1228s&ab_channel=CodeMonkey

2.

- <https://learn.unity.com/tutorial/fixing-performance-problems#5c7f8528edbc2a002053b595>
- <https://docs.unity3d.com/Manual/Glossary.html>
- <https://stackoverflow.com/questions/3427585/understanding-the-bitwise-and-operator/3427633#3427633>