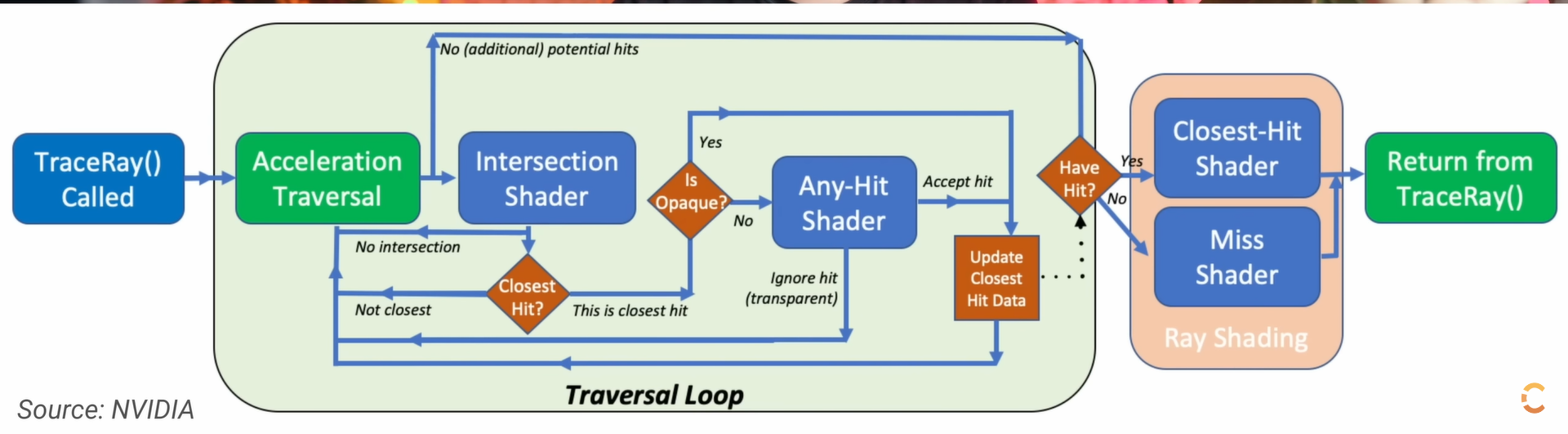
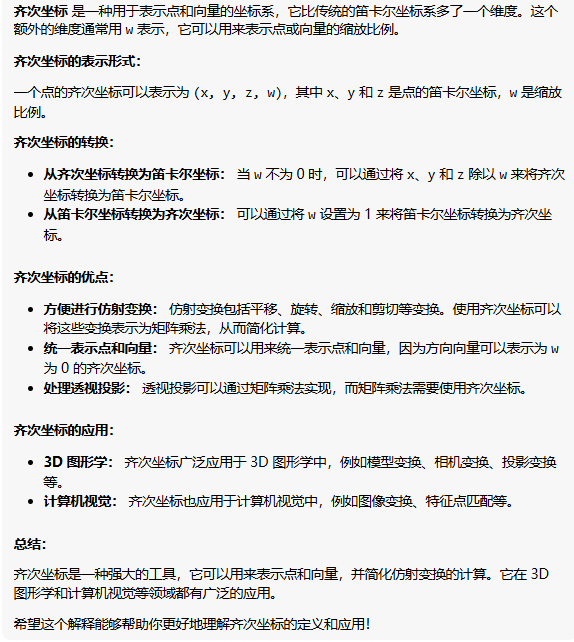
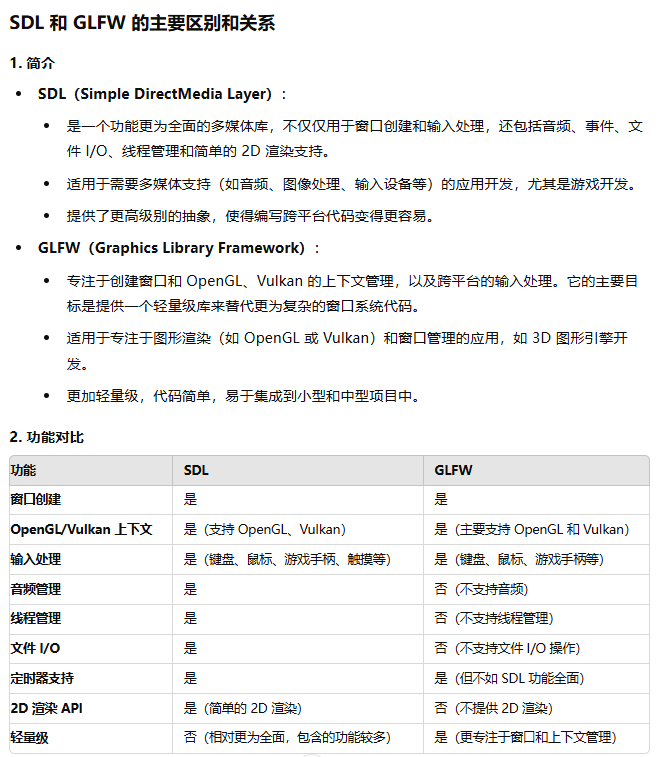
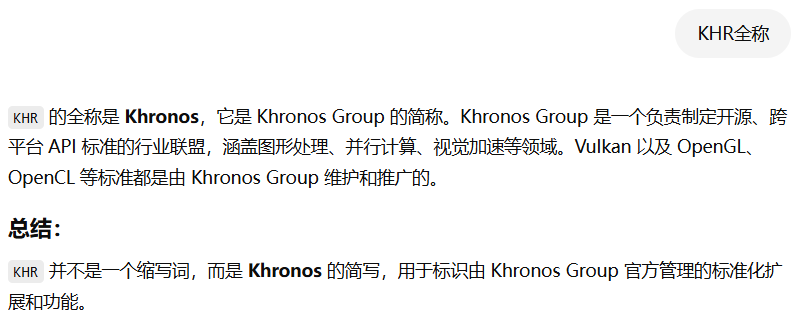


RayTracing pipeline





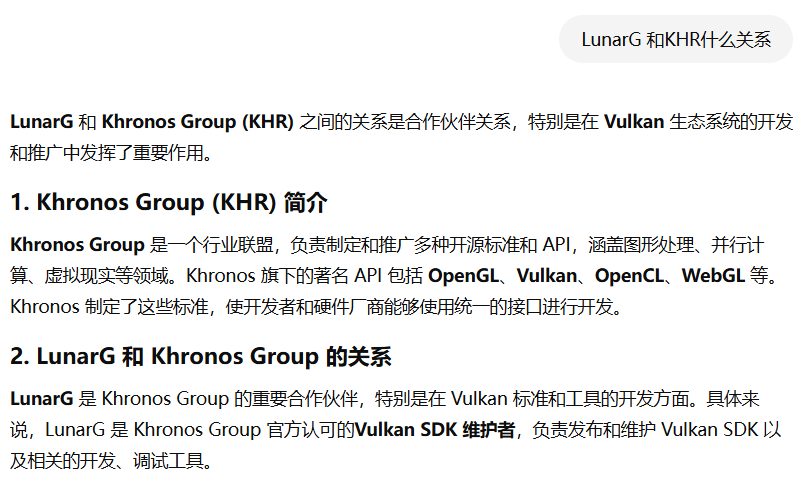


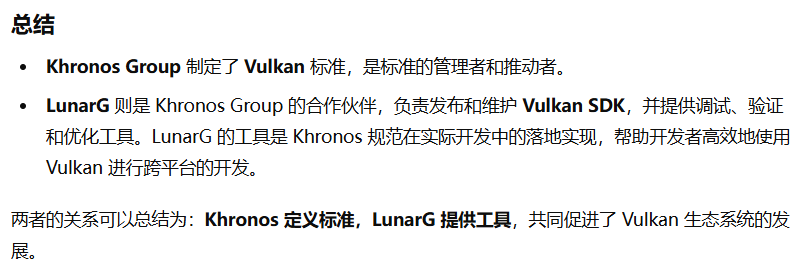


**Validation Layer** ：Vulkan 验证层（**Validation Layers**）的主要作用是帮助开发者在运行时检查 Vulkan API 的使用是否符合规范，从而发现和纠正错误。这些错误可能包括无效的参数、资源未正确释放、同步问题等。它是一个开发阶段非常有用的调试工具，能够帮助你提前捕捉潜在的错误。





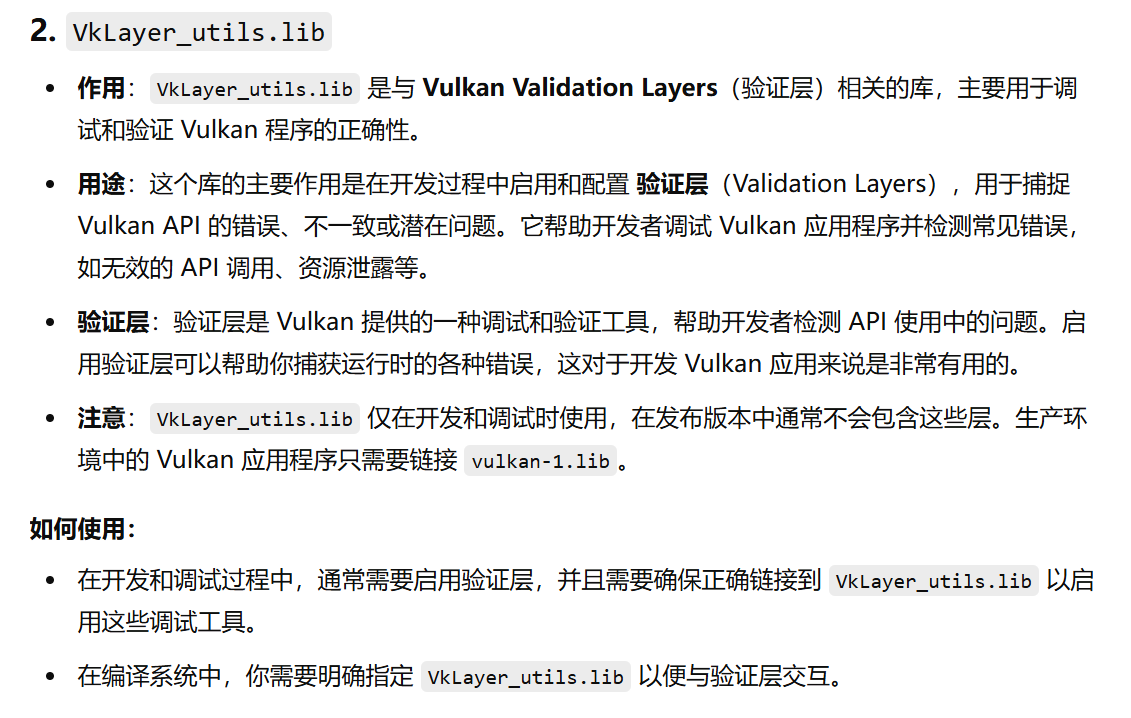




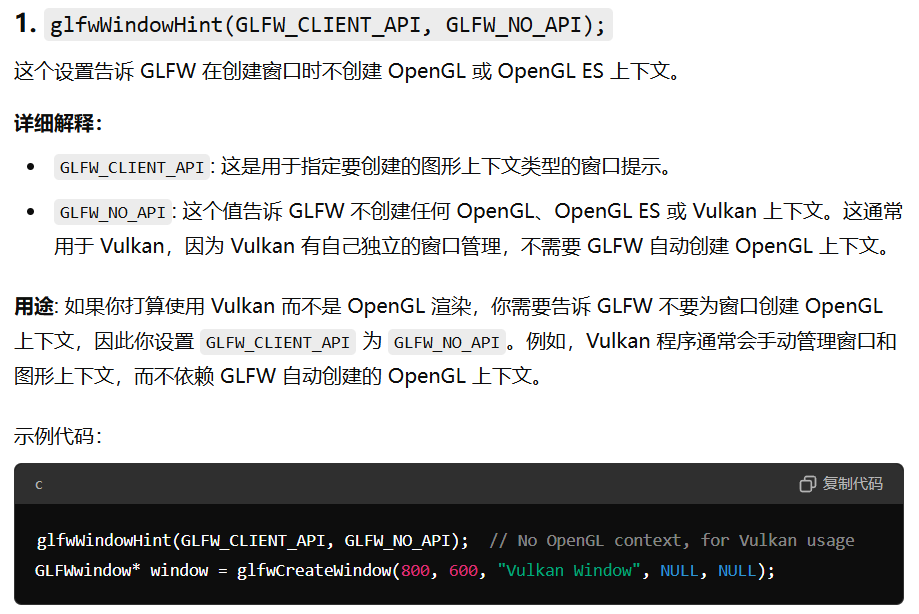




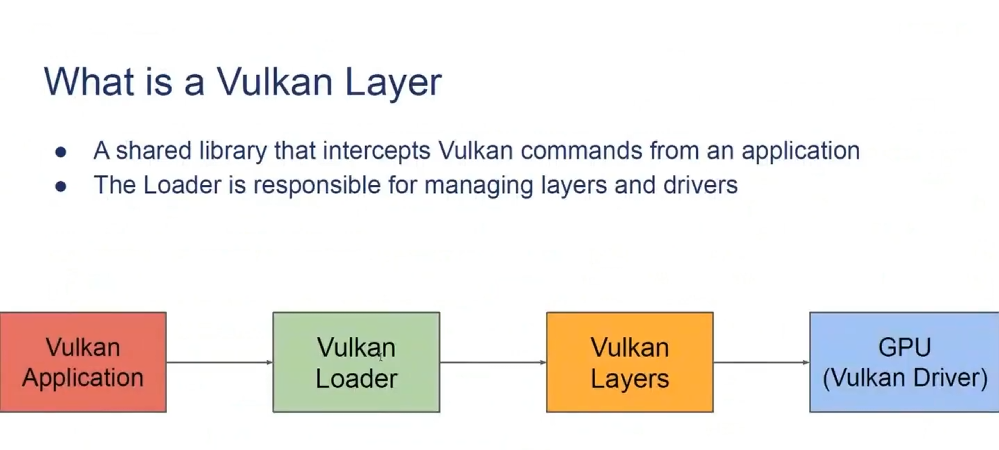




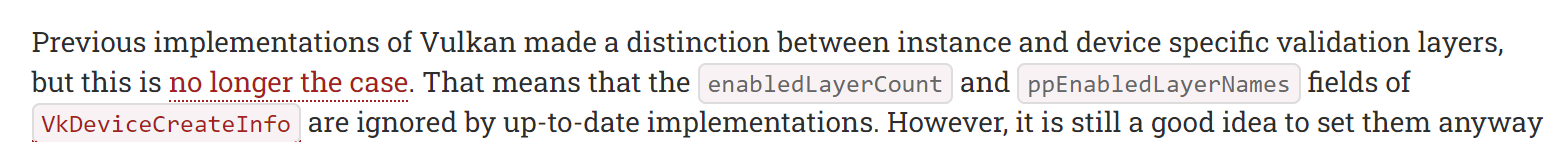


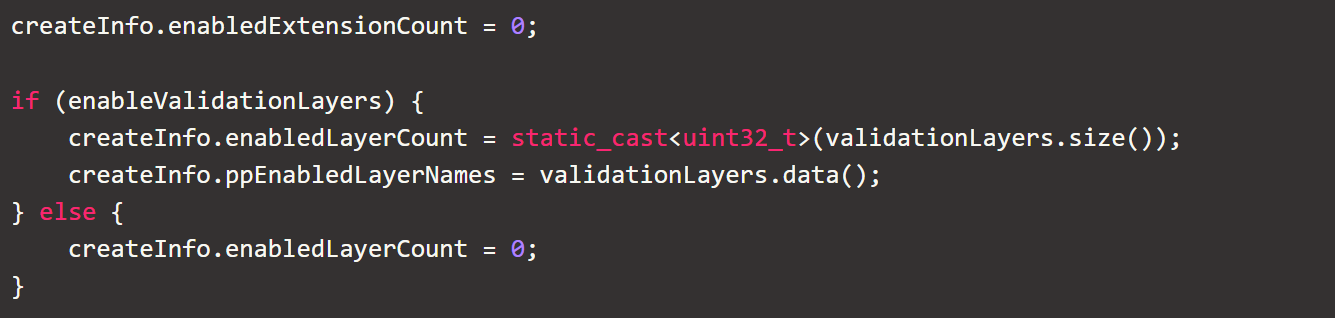




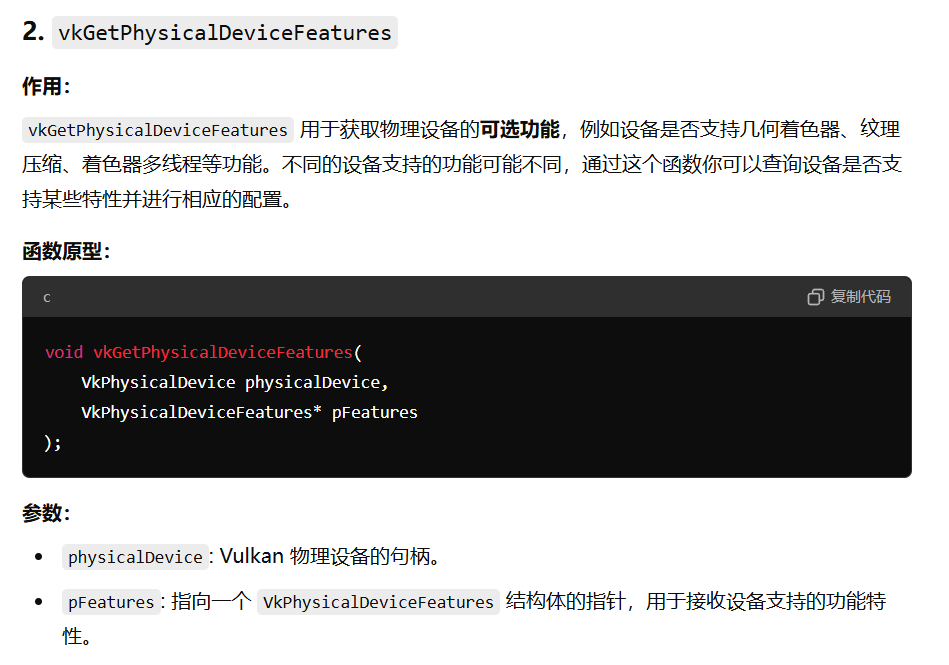


新版本不再区分device和instance的验证层所以这两个参数实际是被新版本忽略的







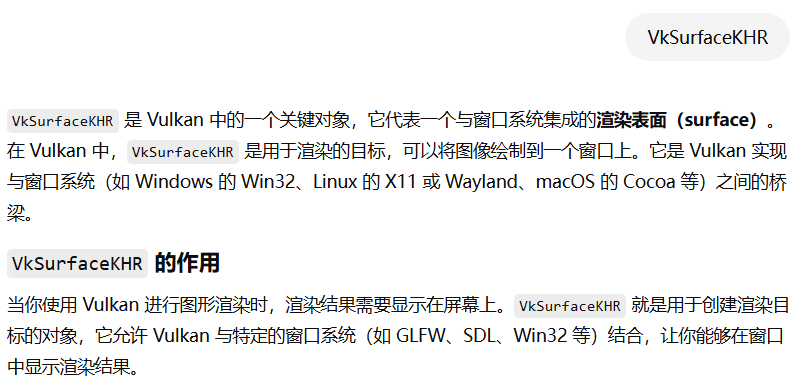








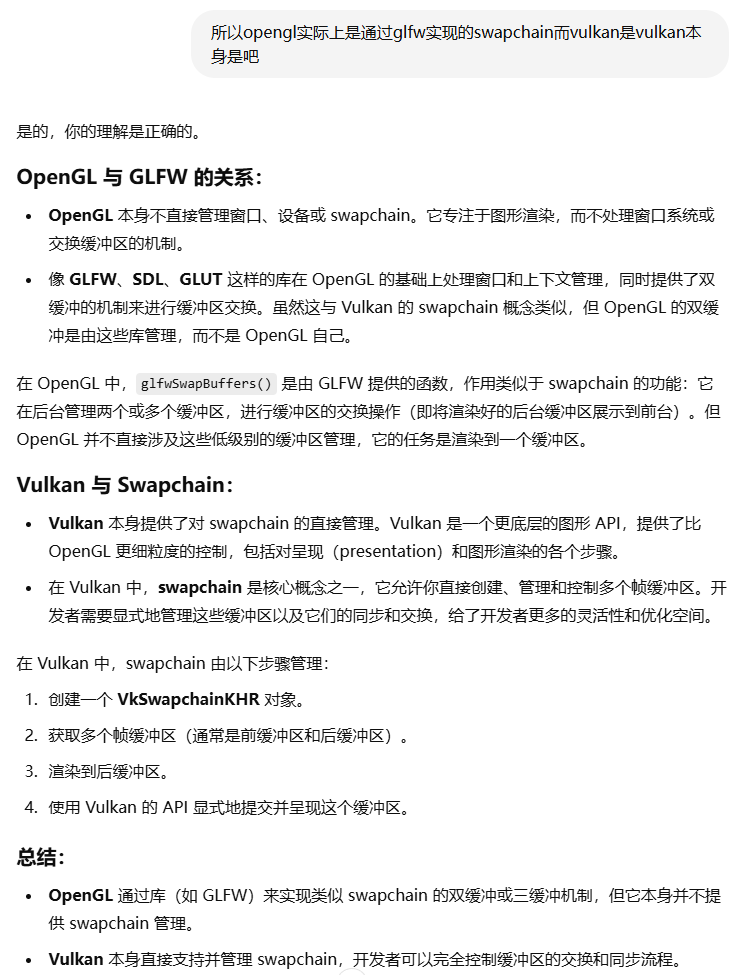
VkSurfaceKHR创建是与平台相关的，而GLFW本身就是平台无关的，所以不需要额外配置，直接用glfw内置函数就行。

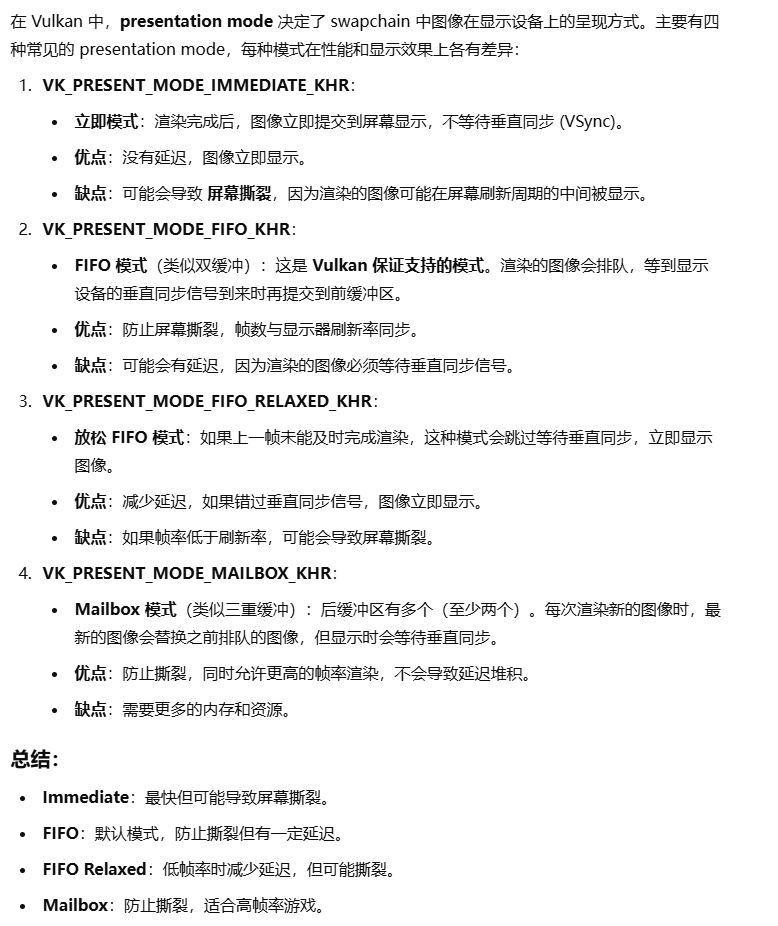






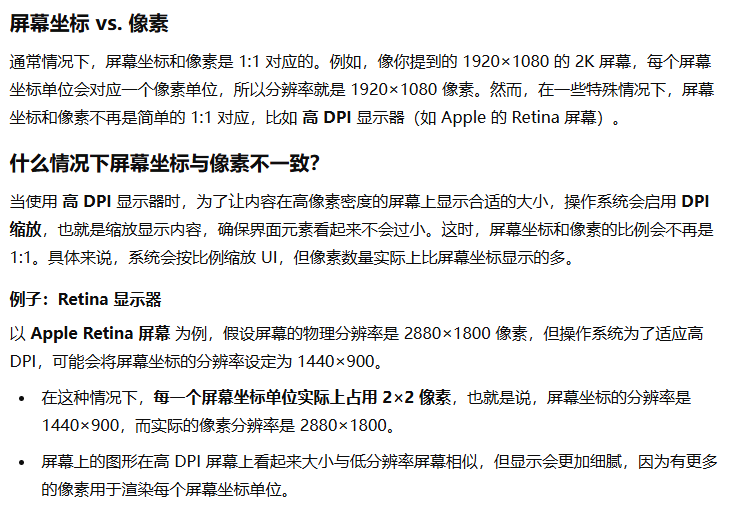


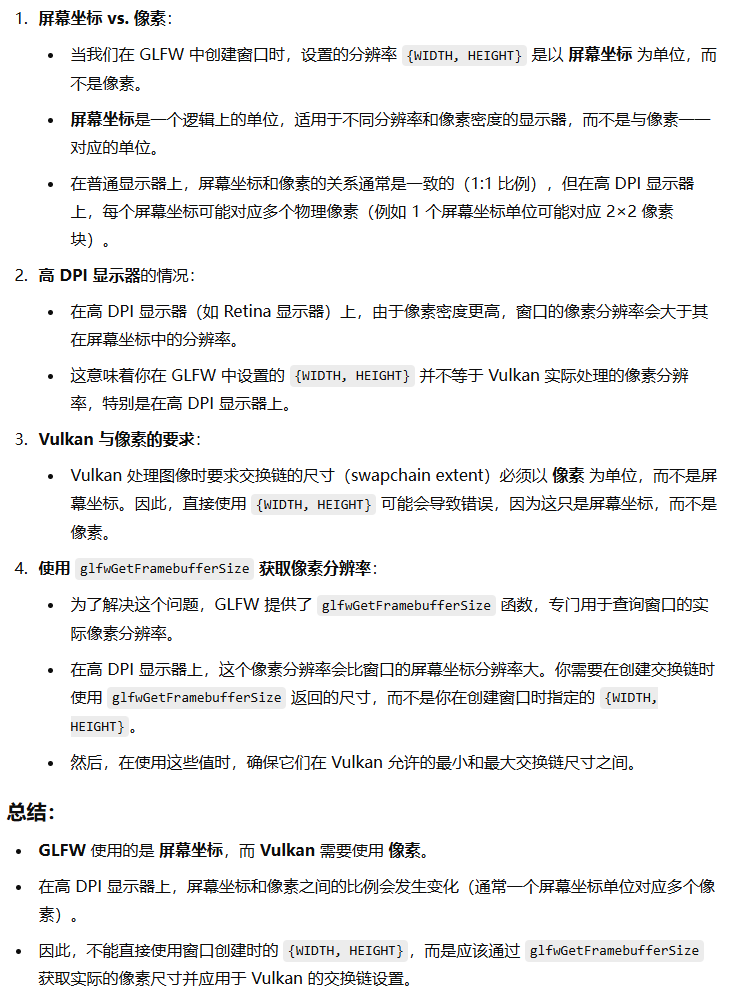




**屏幕坐标和像素并不总是1：1**

**（DPI** ： **Dots Per Inch** 的缩写，表示每英寸显示器上所包含的像素点数，用来衡量屏幕的像素密度。DPI 越高，屏幕的像素密度就越大，图像显示会更加细腻。**）**

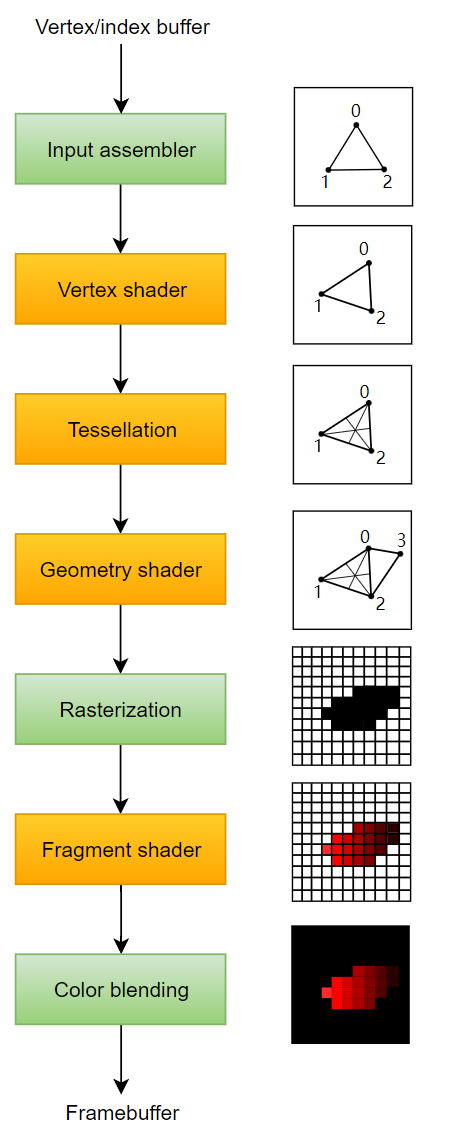




绿色的阶段被称为固定功能阶段。这些阶段允许您使用参数调整它们的操作，但是它们的工作方式是预定义的。

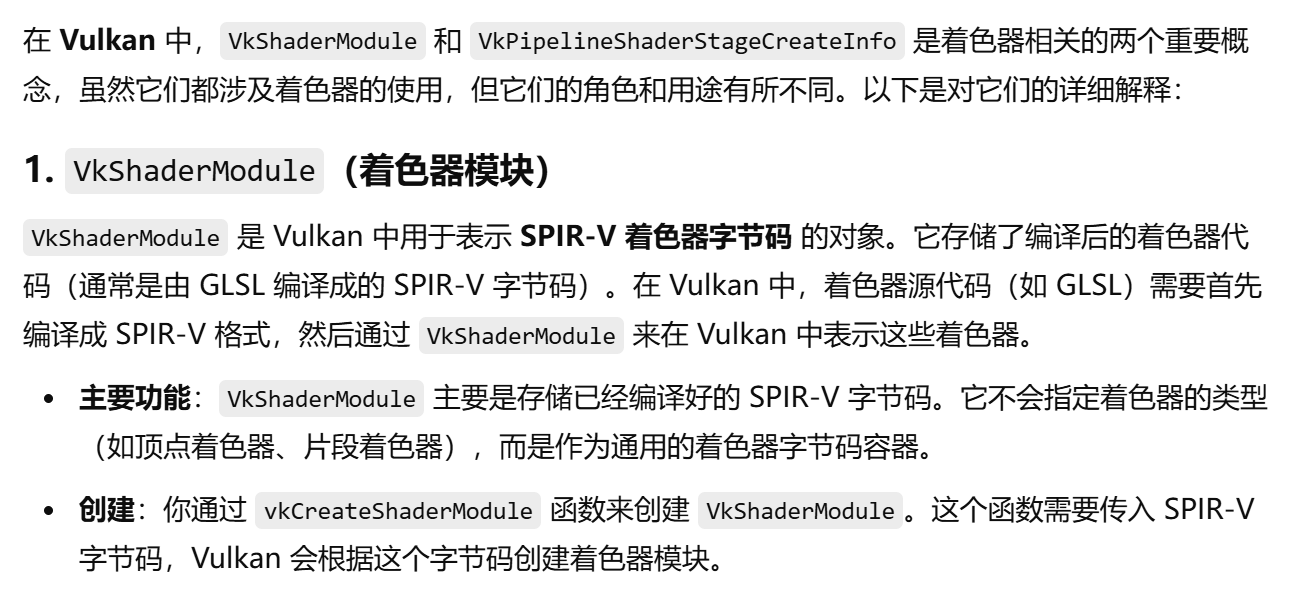
橙色的阶段是可编程的，这意味着您可以将自己的代码上传到图形卡上，以应用您想要的操作。这允许你使用片段着色器，例如，实现从纹理和照明到光线追踪器的任何东西。这些程序同时运行在多个GPU内核上，以并行处理多个对象，如顶点和碎片。这些阶段允许您使用参数调整它们的操作，但是它们的工作方式是预定义的。

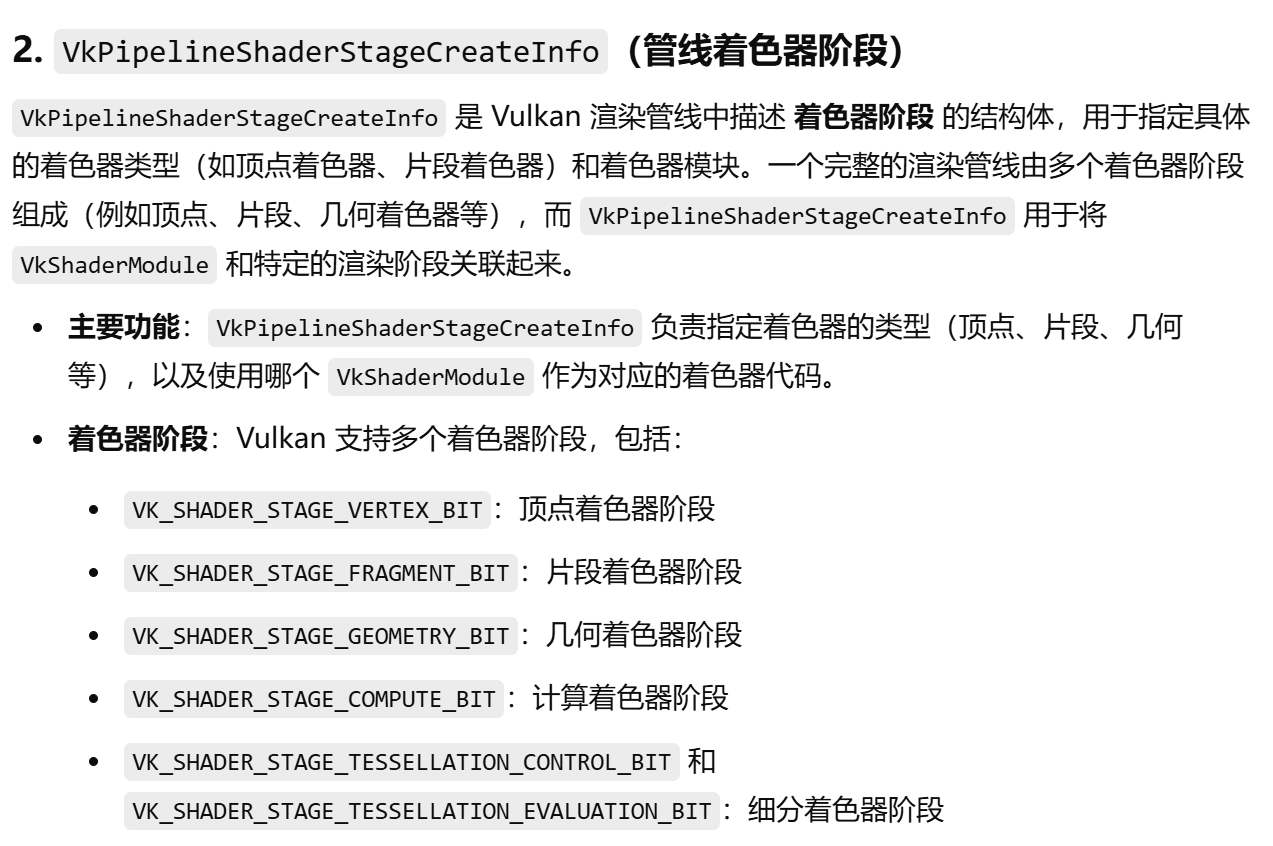
注意：Vulkan中Pipeline是immutable的！！！若要改变某些参数，不能像OpenGL那样直接改，而是需要再创建一个新的Pipeline。

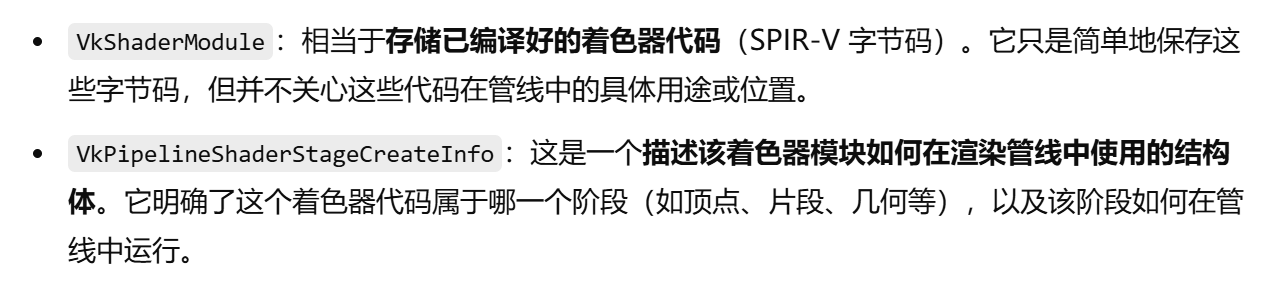


VulkanSDK中的glslc.exe可以将glsl编译为spir-v字节码！

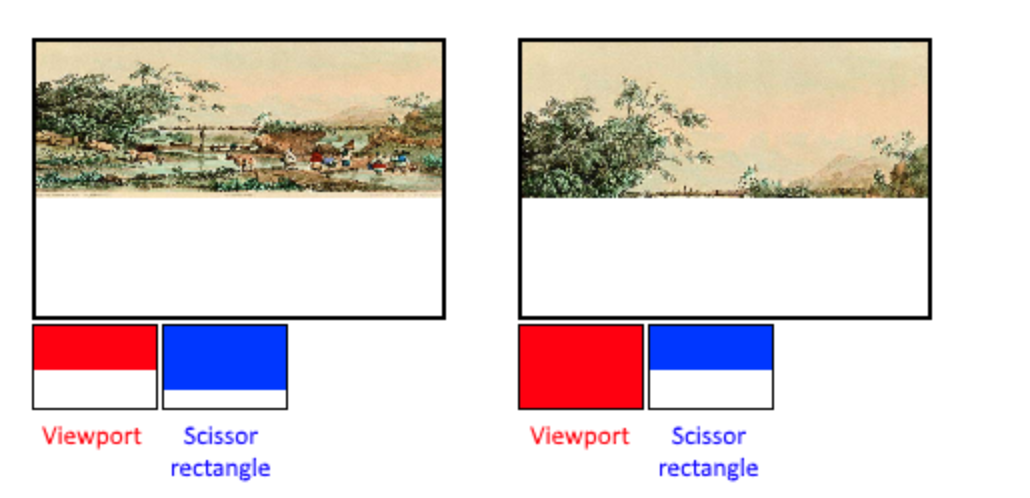








Viewport & Scissor

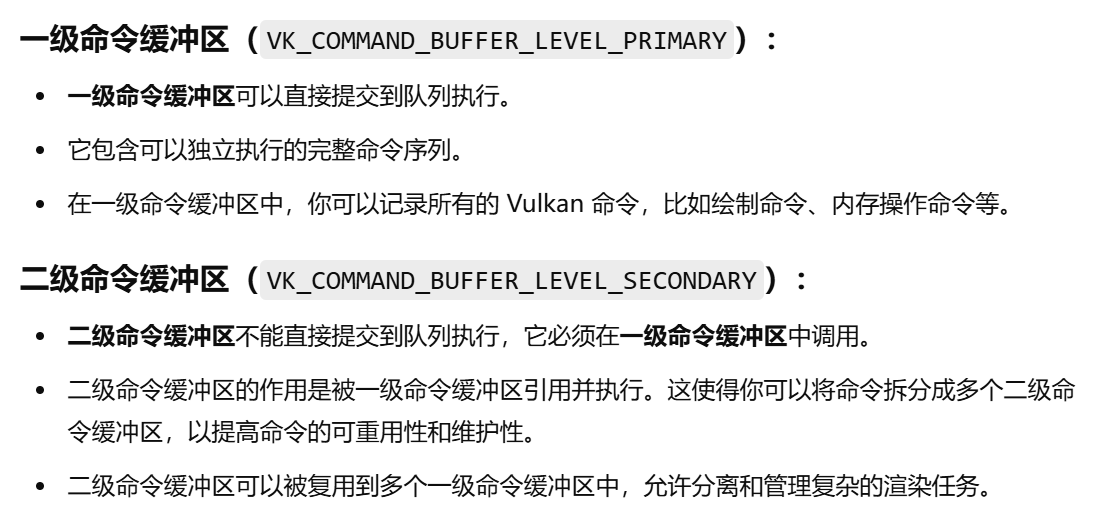


显然Viewport是把整个图像缩放到对应区域，再由Scissor确定那一部分画面可以被渲染到Scissor指定的区域！！！

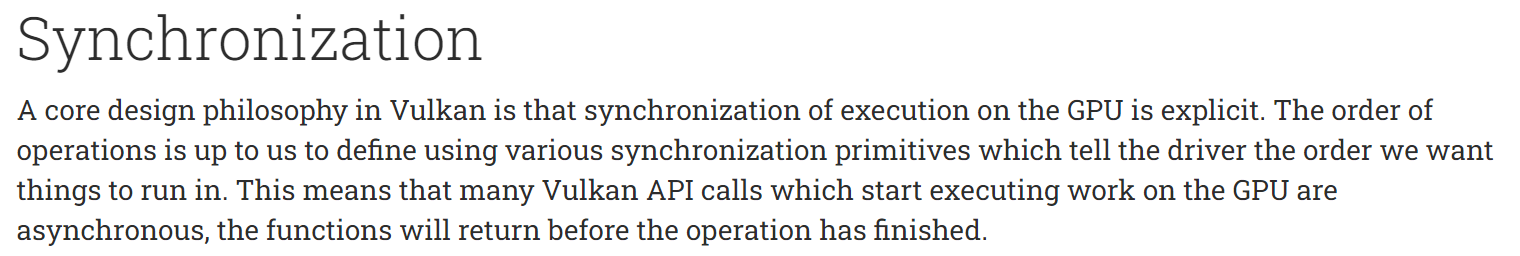






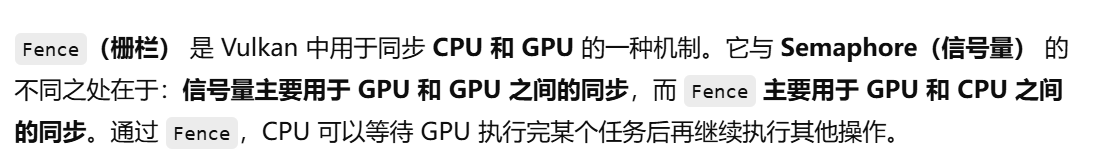


Vulkan的核心优势：gpu默认异步，需要显示同步



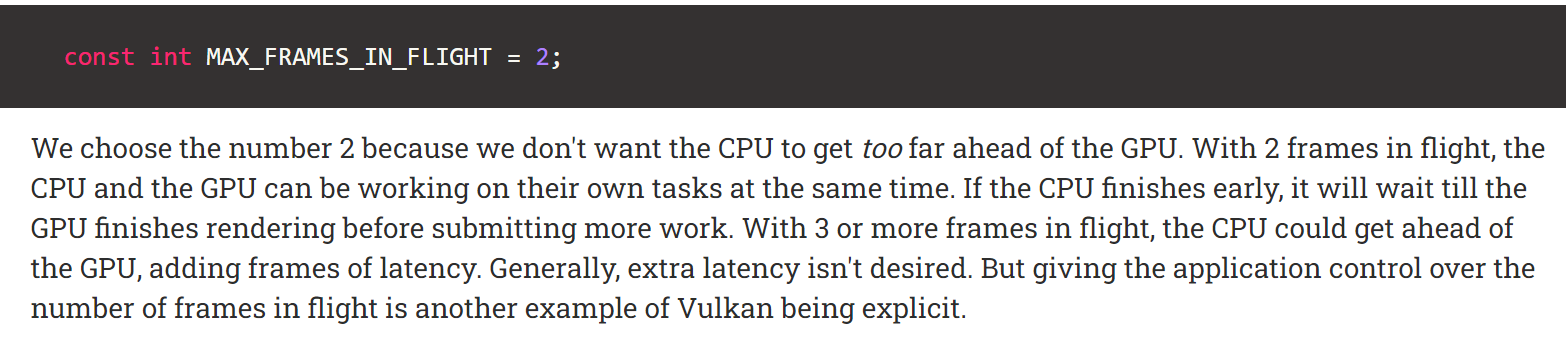


通过Fence可以显示的同步CPU和GPU！！！

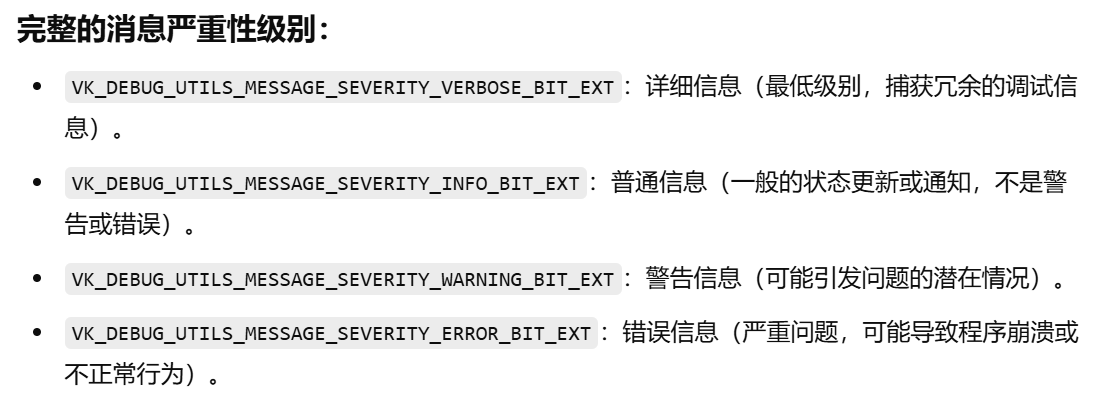




通常2帧下CPU和GPU同步，再多CPU会超过GPU（但这个因该取决于具体GPU和CPU）！！！

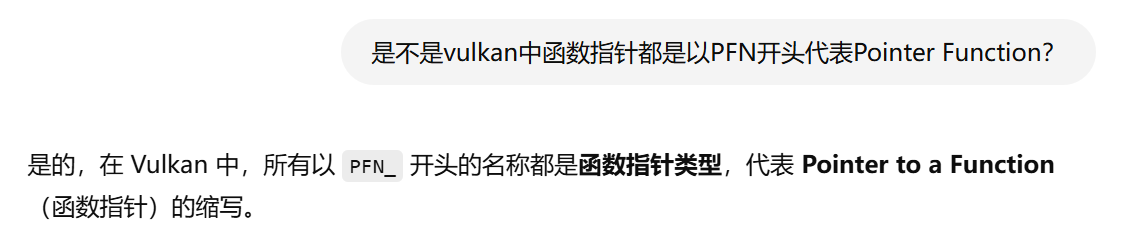


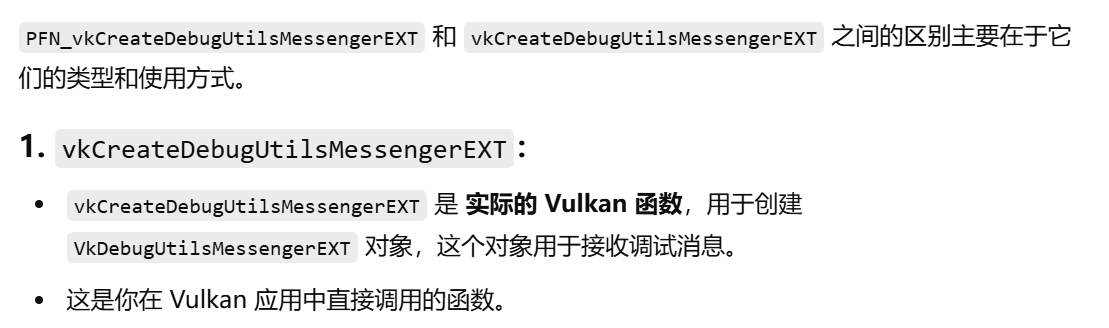
VkDebugUtilsMessengerCreateInfoEXT中的messageSeverity

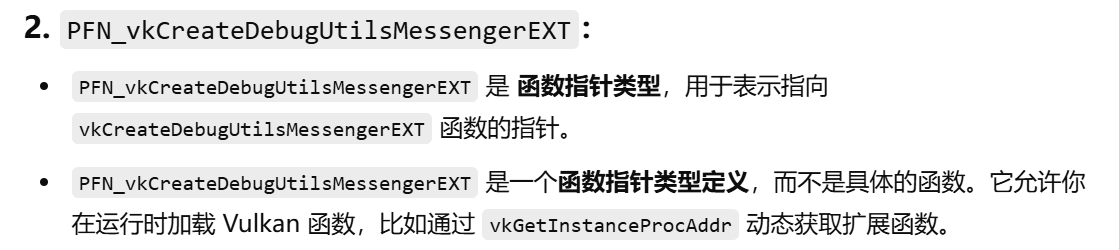


VkDebugUtilsMessengerCreateInfoEXT中的messageType

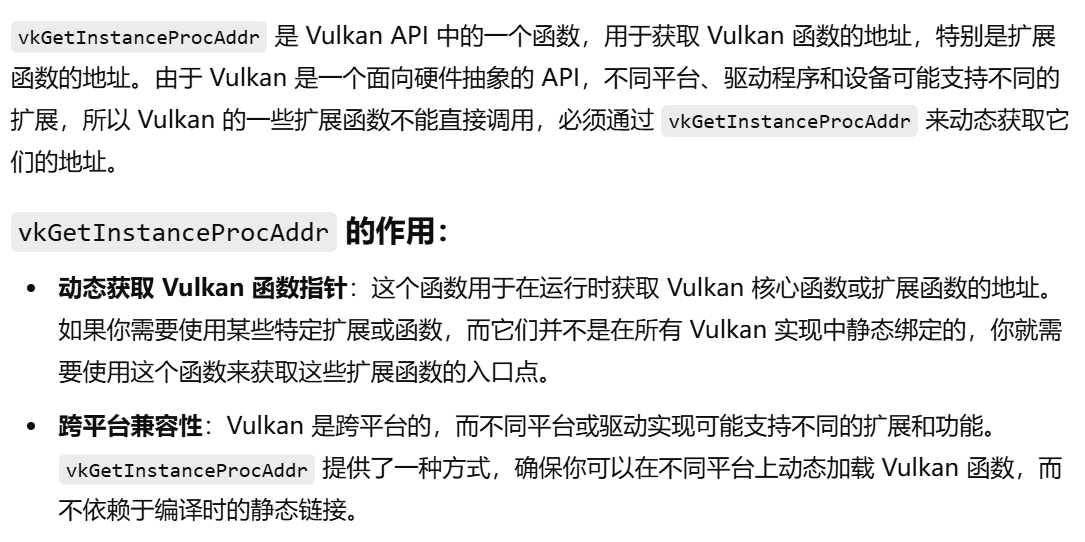


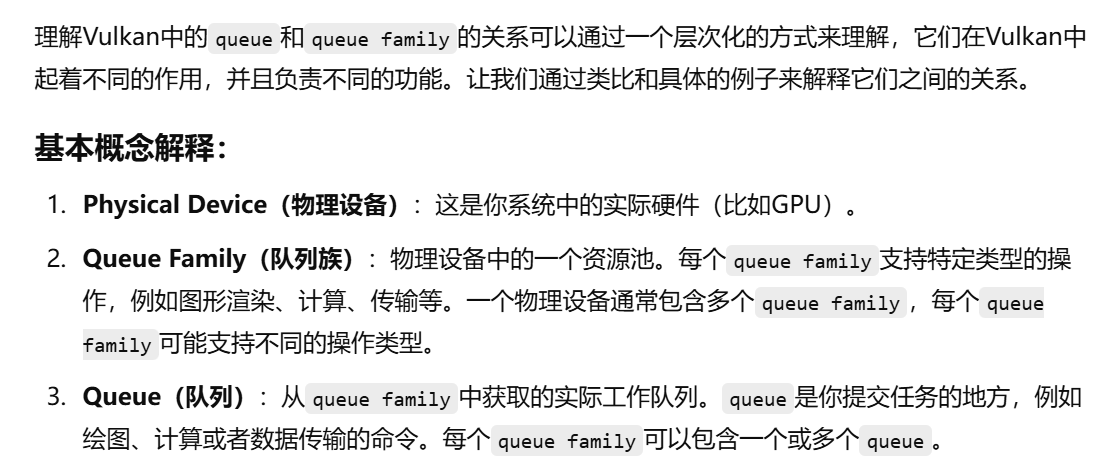


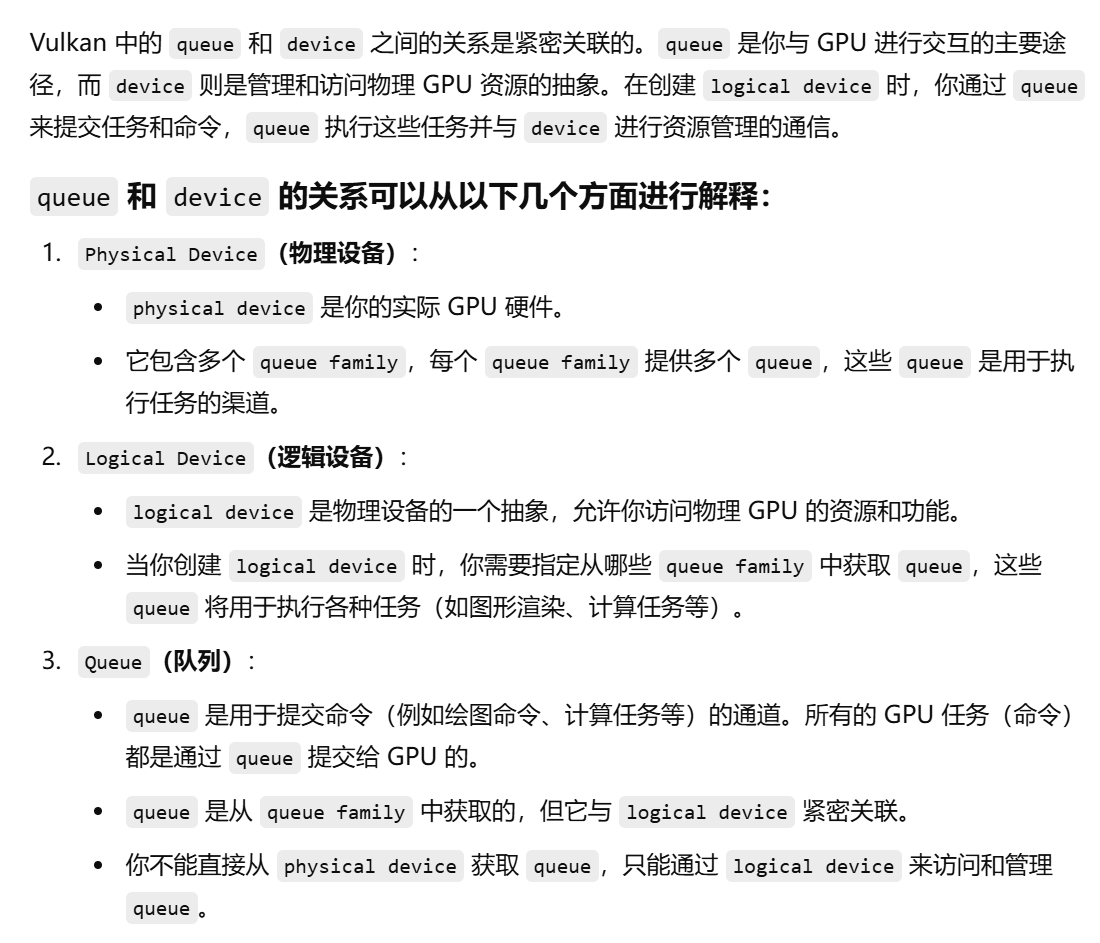


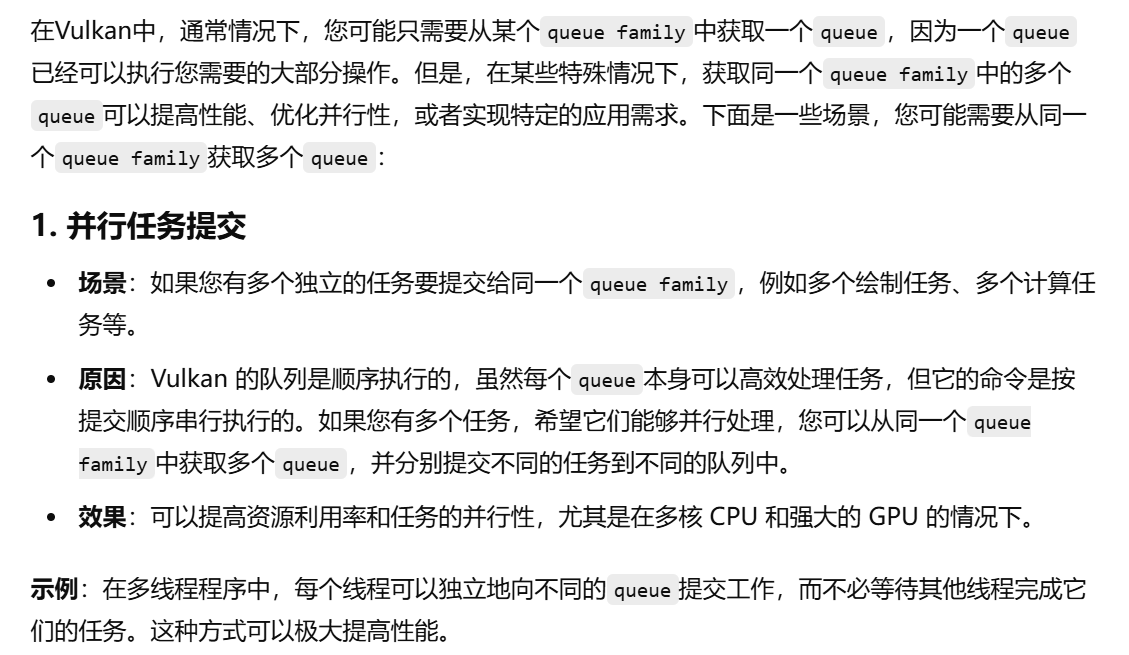


扩展中的函数并没有默认加载，所以需要用vkGetInstanceProcAddr函数运行时动态加载！













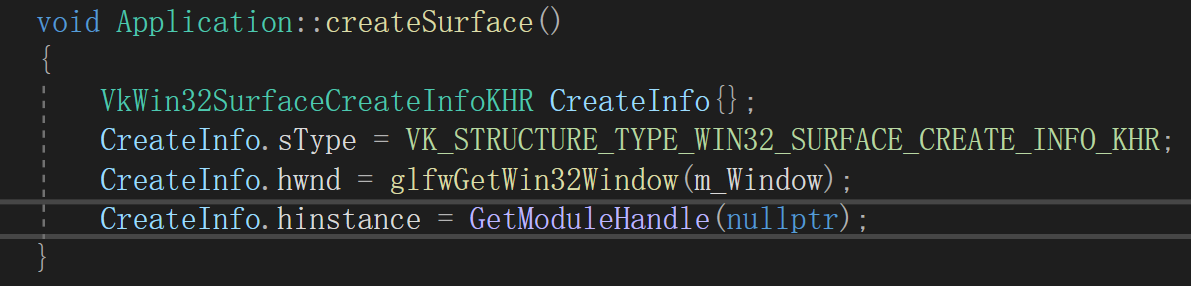
Queue通常只需要一个，GPU也只允许你创建少量的queue，即使多线程也多数情况下只需要一个！！！

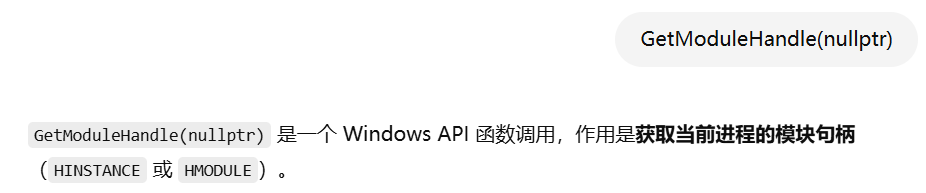


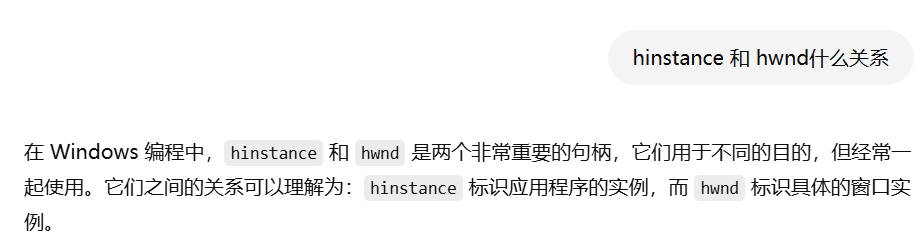
**Surface：**



Win32窗口需要创建Surface的hwnd（窗口实例）和hinstance（程序实例）！！！







**并不是所有GPU的Graphics队列族都及支持渲染又支持present！！！**

