

VkInstance

// Returns global extension properties.

```
VkResult vkEnumerateInstanceExtensionProperties(  
    const char* pLayerName, // usually nullptr  
    uint32_t* pPropertyCount,  
    VkExtensionProperties* pProperties  
)
```

VkExtensionProperties

```
char extensionName[VK_MAX_EXTENSION_NAME_SIZE]  
    // 256 in vulkan_core.h  
uint32_t specVersion
```

```
const char** glfwGetRequiredInstanceExtensions( uint32_t* count )
```

Ex.

```
[0] "VK_KHR_surface"  
[1] "VK_KHR_xcb_surface"
```

Ex.

```
VK_KHR_device_group_creation  1  
VK_KHR_display                23  
VK_KHR_external_fence_capabilities  1  
VK_KHR_external_memory_capabilities  1  
VK_KHR_external_semaphore_capabilities  1  
VK_KHR_get_display_properties2  1  
VK_KHR_get_physical_device_properties2  2  
VK_KHR_get_surface_capabilities2  1  
VK_KHR_surface                25  
VK_KHR_surface_protected_capabilities  1  
VK_KHR_wayland_surface        6  
VK_KHR_xcb_surface            6  
VK_KHR_xlib_surface           6  
VK_EXT_acquire_drm_display    1  
VK_EXT_acquire_xlib_display   1  
VK_EXT_debug_report           10  
VK_EXT_direct_mode_display    1  
VK_EXT_display_surface_counter  1  
VK_EXT_debug_utils            2  
VK_KHR_portability_enumeration 1
```

VkApplicationInfo

```
sType = VK_STRUCTURE_TYPE_APPLICATION_INFO  
pApplicationName = <APP_NAME>  
applicationVersion = VK_MAKE_VERSION(1, 0, 0)  
pEngineName = <USER_ENGINE_NAME> : ...:  
engineVersion = VK_MAKE_VERSION(1, 0, 0)  
apiVersion = VK_API_VERSION_1_0 // Minimum Version that App uses
```

```
VKAPI_ATTR (VkBool32 VKAPI_CALL *)(  
    VkDebugUtilsMessageSeverityFlagBitsEXT messageSeverity,  
    VkDebugUtilsMessageTypeFlagsEXT messageType,  
    const VkDebugUtilsMessengerCallbackDataEXT* pCallbackData,  
    void* pUserData  
) // returning true aborts the program.
```

VkDebugUtilsMessengerCreateInfoEXT

```
sType = VK_STRUCTURE_TYPE_DEBUG_UTILS_MESSENGER_CREATE_INFO_EXT  
messageSeverity = VK_DEBUG_UTILS_MESSAGE_SEVERITY_VERBOSE_BIT_EXT  
                  VK_DEBUG_UTILS_MESSAGE_SEVERITY_WARNING_BIT_EXT  
                  | VK_DEBUG_UTILS_MESSAGE_SEVERITY_ERROR_BIT_EXT  
messageType = VK_DEBUG_UTILS_MESSAGE_TYPE_GENERAL_BIT_EXT  
              | VK_DEBUG_UTILS_MESSAGE_TYPE_VALIDATION_BIT_EXT  
              | VK_DEBUG_UTILS_MESSAGE_TYPE_PERFORMANCE_BIT_EXT  
pfnUserCallback
```

VkInstanceCreateInfo

```
sType = VK_STRUCTURE_TYPE_INSTANCE_CREATE_INFO  
pApplicationInfo  
enabledExtensionCount  
ppEnabledExtensionNames  
enabledLayerCount = 1  
    // or 0 if the validation layer is not needed.  
ppEnabledLayerNames  
    = (const char* const*)"VK_LAYER_KHRONOS_validation"  
pNext
```

```
VkResult vkCreateInstance(
```

```
    const VkInstanceCreateInfo* pCreateInfo,  
    const VkAllocationCallbacks* pAllocator, // usually nullptr  
    VkInstance* pInstance  
)
```

VkInstance

```
void vkDestroyInstance(
```

```
    VkInstance instance,  
    const VkAllocationCallbacks* pAllocator // usually nullptr  
)
```

VkInstance