

VRE::VRE_Window
<ul style="list-style-type: none"> <li>- int mWidth</li> <li>- int mHeight</li> <li>- bool mFramebufferResized</li> <li>- const std::string mWindowName</li> <li>- GLFWwindow * mWindow</li> </ul>
<ul style="list-style-type: none"> <li>+ VRE_Window(int width, int height, std::string name)</li> <li>+ ~VRE_Window()</li> <li>+ VRE_Window(const VRE_Window &amp;)=delete</li> <li>+ VRE_Window &amp; operator =(const VRE_Window &amp;)=delete</li> <li>+ bool ShouldClose()</li> <li>+ void CreateWindowSurface(VkInstance instance, VkSurfaceKHR *surface)</li> <li>+ VkExtent2D GetExtent()</li> <li>+ bool HasWindowResized()</li> <li>+ void ResetWindowResizedFlag()</li> <li>+ GLFWwindow * GetGLFWwindow() const</li> <li>- void Init()</li> <li>- static void FrameBufferResizedCallback(GLFWwindow *window, int width, int height)</li> </ul>

-mWindow

VRE::VRE_Device
<ul style="list-style-type: none"> <li>+ const bool mEnableValidationLayers</li> <li>+ VkPhysicalDeviceProperties mProperties</li> <li>- VkInstance mInstance</li> <li>- VkDebugUtilsMessengerEXT mDebugMessenger</li> <li>- VkPhysicalDevice mPhysicalDevice</li> <li>- VkCommandPool mCommandPool</li> <li>- VkDevice mVkDevice</li> <li>- VkSurfaceKHR mVkSurface</li> <li>- VkQueue mGraphicsQueue</li> <li>- VkQueue mPresentQueue</li> <li>- const std::vector&lt;const char * &gt; mValidationLayers</li> <li>- const std::vector&lt;const char * &gt; mDeviceExtensions</li> </ul>
<ul style="list-style-type: none"> <li>+ VRE_Device(VRE_Window &amp;window)</li> <li>+ ~VRE_Device()</li> <li>+ VRE_Device(const VRE_Device &amp;)=delete</li> <li>+ VRE_Device &amp; operator =(const VRE_Device &amp;)=delete</li> <li>+ VRE_Device(VRE_Device &amp;&amp;)=delete</li> <li>+ VRE_Device &amp; operator =(VRE_Device &amp;&amp;)=delete</li> <li>+ VkCommandPool GetCommandPool()</li> <li>+ VkDevice GetVkDevice()</li> <li>+ VkPhysicalDevice GetPhysicalDevice()</li> <li>+ VkSurfaceKHR Surface() and 12 more...</li> <li>- void CreateInstance()</li> <li>- void SetupDebugMessenger()</li> <li>- void CreateSurface()</li> <li>- void PickPhysicalDevice()</li> <li>- void CreateLogicalDevice()</li> <li>- void CreateCommandPool()</li> <li>- bool IsDeviceSuitable(VkPhysicalDevice device)</li> <li>- std::vector&lt;const char * &gt; GetRequiredExtensions()</li> <li>- bool CheckValidationLayerSupport()</li> <li>- QueueFamilyIndices FindQueueFamilies(VkPhysicalDevice device)</li> <li>- void PopulateDebugMessengerCreateInfo(VkDebugUtilsMessengerCreateInfoEXT &amp;createInfo)</li> <li>- void HasGlfwRequiredInstanceExtensions()</li> <li>- bool CheckDeviceExtensionSupport(VkPhysicalDevice device)</li> <li>- SwapChainSupportDetails QuerySwapChainSupport(VkPhysicalDevice device)</li> </ul>

-mDevice

VRE::VRE_Texture
<ul style="list-style-type: none"> <li>- VkImage mTextureImage</li> <li>- VkDeviceMemory mTextureImageMemory</li> <li>- VkImageView mImageView</li> <li>- VkSampler mTextureSampler</li> <li>- VkDescriptorImageInfo mDescriptor</li> <li>- uint32_t mMipLevels</li> </ul>
<ul style="list-style-type: none"> <li>+ VRE_Texture(VRE_Device &amp;device, const std::string &amp;filePath)</li> <li>+ ~VRE_Texture()</li> <li>+ VRE_Texture(const VRE_Texture &amp;)=delete</li> <li>+ VRE_Texture &amp; operator =(const VRE_Texture &amp;)=delete</li> <li>+ VkDescriptorImageInfo GetDescImageInfo() const</li> <li>+ static std::unique_ptr&lt; VRE_Texture &gt; CreateTexture(VRE_Device &amp;device, const std::string &amp;filePath)</li> <li>- void CreateImage(const std::string &amp;filePath)</li> <li>- void GenerateMipmaps(VkImage image, VkFormat imageFormat, int32_t texWidth, int32_t texHeight)</li> <li>- void TransitionImageLayout(VkImage image, VkFormat format, VkImageLayout oldLayout, VkImageLayout newLayout)</li> <li>- void CreateImageView()</li> <li>- void CreateTextureSampler()</li> <li>- void UpdateDescriptorInfo()</li> </ul>