

vengine

0.1.0

Generated by Doxygen 1.9.1



<b>1 Hierarchical Index</b>	<b>1</b>
1.1 Class Hierarchy	1
<b>2 Class Index</b>	<b>3</b>
2.1 Class List	3
<b>3 Class Documentation</b>	<b>5</b>
3.1 ven::Buffer Class Reference	5
3.1.1 Member Function Documentation	5
3.1.1.1 descriptorInfo()	5
3.1.1.2 descriptorInfoForIndex()	6
3.1.1.3 flush()	6
3.1.1.4 flushIndex()	7
3.1.1.5 invalidate()	7
3.1.1.6 invalidateIndex()	7
3.1.1.7 map()	8
3.1.1.8 unmap()	8
3.1.1.9 writeToBuffer()	8
3.1.1.10 writeToIndex()	9
3.2 ven::DescriptorPool::Builder Class Reference	9
3.3 ven::DescriptorSetLayout::Builder Class Reference	9
3.4 ven::Model::Builder Struct Reference	10
3.5 ven::Camera Class Reference	10
3.6 myLib::Clock Class Reference	10
3.7 ven::DescriptorPool Class Reference	11
3.8 ven::DescriptorSetLayout Class Reference	11
3.9 ven::DescriptorWriter Class Reference	12
3.10 ven::Device Class Reference	12
3.11 ven::Engine Class Reference	13
3.12 ven::FrameInfo Struct Reference	13
3.13 ven::KeyboardController Class Reference	13
3.14 ven::KeyboardController::KeyMappings Struct Reference	14
3.15 ven::Model Class Reference	14
3.16 ven::Object Class Reference	15
3.17 ven::PipelineConfigInfo Struct Reference	15
3.18 gui::PluginLoader Class Reference	16
3.19 gui::PluginLoader::PluginLoaderException Class Reference	16
3.20 ven::QueueFamilyIndices Struct Reference	17
3.21 myLib::Random Class Reference	17
3.22 ven::Renderer Class Reference	17
3.23 ven::RenderSystem Class Reference	18
3.24 ven::Shaders Class Reference	18
3.25 ven::SimplePushConstantData Struct Reference	18

3.26 ven::SwapChain Class Reference . . . . .	19
3.27 ven::SwapChainSupportDetails Struct Reference . . . . .	19
3.28 myLib::Time Class Reference . . . . .	20
3.29 ven::Transform3DComponent Struct Reference . . . . .	20
3.30 ven::Model::Vertex Struct Reference . . . . .	20
3.31 ven::Window Class Reference . . . . .	21
<b>Index</b>	<b>23</b>

# Chapter 1

## Hierarchical Index

### 1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

ven::Buffer . . . . .	5
ven::DescriptorPool::Builder . . . . .	9
ven::DescriptorSetLayout::Builder . . . . .	9
ven::Model::Builder . . . . .	10
ven::Camera . . . . .	10
myLib::Clock . . . . .	10
ven::DescriptorPool . . . . .	11
ven::DescriptorSetLayout . . . . .	11
ven::DescriptorWriter . . . . .	12
ven::Device . . . . .	12
ven::Engine . . . . .	13
std::exception	
gui::PluginLoader::PluginLoaderException . . . . .	16
ven::FrameInfo . . . . .	13
ven::KeyboardController . . . . .	13
ven::KeyboardController::KeyMappings . . . . .	14
ven::Model . . . . .	14
ven::Object . . . . .	15
ven::PipelineConfigInfo . . . . .	15
gui::PluginLoader . . . . .	16
ven::QueueFamilyIndices . . . . .	17
myLib::Random . . . . .	17
ven::Renderer . . . . .	17
ven::RenderSystem . . . . .	18
ven::Shaders . . . . .	18
ven::SimplePushConstantData . . . . .	18
ven::SwapChain . . . . .	19
ven::SwapChainSupportDetails . . . . .	19
myLib::Time . . . . .	20
ven::Transform3DComponent . . . . .	20
ven::Model::Vertex . . . . .	20
ven::Window . . . . .	21



## Chapter 2

# Class Index

### 2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">ven::Buffer</a>	5
<a href="#">ven::DescriptorPool::Builder</a>	9
<a href="#">ven::DescriptorSetLayout::Builder</a>	9
<a href="#">ven::Model::Builder</a>	10
<a href="#">ven::Camera</a>	10
<a href="#">myLib::Clock</a>	10
<a href="#">ven::DescriptorPool</a>	11
<a href="#">ven::DescriptorSetLayout</a>	11
<a href="#">ven::DescriptorWriter</a>	12
<a href="#">ven::Device</a>	12
<a href="#">ven::Engine</a>	13
<a href="#">ven::FrameInfo</a>	13
<a href="#">ven::KeyboardController</a>	13
<a href="#">ven::KeyboardController::KeyMappings</a>	14
<a href="#">ven::Model</a>	14
<a href="#">ven::Object</a>	15
<a href="#">ven::PipelineConfigInfo</a>	15
<a href="#">gui::PluginLoader</a>	16
<a href="#">gui::PluginLoader::PluginLoaderException</a>	16
<a href="#">ven::QueueFamilyIndices</a>	17
<a href="#">myLib::Random</a>	17
<a href="#">ven::Renderer</a>	17
<a href="#">ven::RenderSystem</a>	18
<a href="#">ven::Shaders</a>	18
<a href="#">ven::SimplePushConstantData</a>	18
<a href="#">ven::SwapChain</a>	19
<a href="#">ven::SwapChainSupportDetails</a>	19
<a href="#">myLib::Time</a>	20
<a href="#">ven::Transform3DComponent</a>	20
<a href="#">ven::Model::Vertex</a>	20
<a href="#">ven::Window</a>	21





## Chapter 3

# Class Documentation

### 3.1 `ven::Buffer` Class Reference

#### Public Member Functions

- **Buffer** ([Device](#) &device, VkDeviceSize instanceSize, uint32\_t instanceCount, VkBufferUsageFlags usage←Flags, VkMemoryPropertyFlags memoryPropertyFlags, VkDeviceSize minOffsetAlignment=1)
- **Buffer** (const [Buffer](#) &)=delete
- [Buffer](#) & **operator=** (const [Buffer](#) &)=delete
- VkResult [map](#) (VkDeviceSize size=VK\_WHOLE\_SIZE, VkDeviceSize offset=0)
- void [unmap](#) ()
- void [writeToBuffer](#) (void \*data, VkDeviceSize size=VK\_WHOLE\_SIZE, VkDeviceSize offset=0)
- VkResult [flush](#) (VkDeviceSize size=VK\_WHOLE\_SIZE, VkDeviceSize offset=0)
- VkDescriptorBufferInfo [descriptorInfo](#) (VkDeviceSize size=VK\_WHOLE\_SIZE, VkDeviceSize offset=0)
- VkResult [invalidate](#) (VkDeviceSize size=VK\_WHOLE\_SIZE, VkDeviceSize offset=0)
- void [writeToIndex](#) (void \*data, int index)
- VkResult [flushIndex](#) (int index)
- VkDescriptorBufferInfo [descriptorInfoForIndex](#) (int index)
- VkResult [invalidateIndex](#) (int index)
- VkBuffer **getBuffer** () const
- void \* **getMappedMemory** () const
- uint32\_t **getInstanceCount** () const
- VkDeviceSize **getInstanceSize** () const
- VkDeviceSize **getAlignmentSize** () const
- VkBufferUsageFlags **getUsageFlags** () const
- VkMemoryPropertyFlags **getMemoryPropertyFlags** () const
- VkDeviceSize **getBufferSize** () const

#### 3.1.1 Member Function Documentation

##### 3.1.1.1 `descriptorInfo()`

```
VkDescriptorBufferInfo ven::Buffer::descriptorInfo (  
    VkDeviceSize size = VK_WHOLE_SIZE,  
    VkDeviceSize offset = 0 ) [inline]
```

Create a buffer info descriptor

**Parameters**

<i>size</i>	(Optional) Size of the memory range of the descriptor
<i>offset</i>	(Optional) Byte offset from beginning

**Returns**

VkDescriptorBufferInfo of specified offset and range

**3.1.1.2 descriptorInfoForIndex()**

```
VkDescriptorBufferInfo ven::Buffer::descriptorInfoForIndex (
    int index ) [inline]
```

Create a buffer info descriptor

**Parameters**

<i>index</i>	Specifies the region given by index * alignmentSize
--------------	---

**Returns**

VkDescriptorBufferInfo for instance at index

**3.1.1.3 flush()**

```
VkResult ven::Buffer::flush (
    VkDeviceSize size = VK_WHOLE_SIZE,
    VkDeviceSize offset = 0 )
```

Flush a memory range of the buffer to make it visible to the device

**Note**

Only required for non-coherent memory

**Parameters**

<i>size</i>	(Optional) Size of the memory range to flush. Pass VK_WHOLE_SIZE to flush the complete buffer range.
<i>offset</i>	(Optional) Byte offset from beginning

**Returns**

VkResult of the flush call

**3.1.1.4 flushIndex()**

```
VkResult ven::Buffer::flushIndex (
    int index ) [inline]
```

Flush the memory range at index \* alignmentSize of the buffer to make it visible to the device

**Parameters**

<i>index</i>	Used in offset calculation
--------------	----------------------------

**3.1.1.5 invalidate()**

```
VkResult ven::Buffer::invalidate (
    VkDeviceSize size = VK_WHOLE_SIZE,
    VkDeviceSize offset = 0 )
```

Invalidate a memory range of the buffer to make it visible to the host

**Note**

Only required for non-coherent memory

**Parameters**

<i>size</i>	(Optional) Size of the memory range to invalidate. Pass VK_WHOLE_SIZE to invalidate the complete buffer range.
<i>offset</i>	(Optional) Byte offset from beginning

**Returns**

VkResult of the invalidate call

**3.1.1.6 invalidateIndex()**

```
VkResult ven::Buffer::invalidateIndex (
    int index ) [inline]
```

Invalidate a memory range of the buffer to make it visible to the host

**Note**

Only required for non-coherent memory

**Parameters**

<i>index</i>	Specifies the region to invalidate: $\text{index} * \text{alignmentSize}$
--------------	---

**Returns**

VkResult of the invalidate call

**3.1.1.7 map()**

```
VkResult ven::Buffer::map (
    VkDeviceSize size = VK_WHOLE_SIZE,
    VkDeviceSize offset = 0 )
```

Map a memory range of this buffer. If successful, mapped points to the specified buffer range.

**Parameters**

<i>size</i>	(Optional) Size of the memory range to map. Pass VK_WHOLE_SIZE to map the complete buffer range.
<i>offset</i>	(Optional) Byte offset from beginning

**Returns**

VkResult of the buffer mapping call

**3.1.1.8 unmap()**

```
void ven::Buffer::unmap ( )
```

Unmap a mapped memory range

**Note**

Does not return a result as vkUnmapMemory can't fail

**3.1.1.9 writeToBuffer()**

```
void ven::Buffer::writeToBuffer (
    void * data,
    VkDeviceSize size = VK_WHOLE_SIZE,
    VkDeviceSize offset = 0 )
```

Copies the specified data to the mapped buffer. Default value writes whole buffer range

## Parameters

<i>data</i>	Pointer to the data to copy
<i>size</i>	(Optional) Size of the data to copy. Pass VK_WHOLE_SIZE to flush the complete buffer range.
<i>offset</i>	(Optional) Byte offset from beginning of mapped region

## 3.1.1.10 writeToIndex()

```
void ven::Buffer::writeToIndex (
    void * data,
    int index ) [inline]
```

Copies "instanceSize" bytes of data to the mapped buffer at an offset of index \* alignmentSize

## Parameters

<i>data</i>	Pointer to the data to copy
<i>index</i>	Used in offset calculation

The documentation for this class was generated from the following file:

- include/VEngine/Buffer.hpp

## 3.2 ven::DescriptorPool::Builder Class Reference

## Public Member Functions

- **Builder** ([Device](#) &device)
- **Builder** & **addPoolSize** (VkDescriptorType descriptorType, uint32\_t count)
- **Builder** & **setPoolFlags** (VkDescriptorPoolCreateFlags flags)
- **Builder** & **setMaxSets** (uint32\_t count)
- std::unique\_ptr< [DescriptorPool](#) > **build** () const

The documentation for this class was generated from the following file:

- include/VEngine/Descriptors.hpp

## 3.3 ven::DescriptorSetLayout::Builder Class Reference

## Public Member Functions

- **Builder** ([Device](#) &device)
- **Builder** & **addBinding** (uint32\_t binding, VkDescriptorType descriptorType, VkShaderStageFlags stage↔Flags, uint32\_t count=1)
- std::unique\_ptr< [DescriptorSetLayout](#) > **build** () const

The documentation for this class was generated from the following file:

- include/VEngine/Descriptors.hpp

### 3.4 ven::Model::Builder Struct Reference

#### Public Member Functions

- void **loadModel** (const std::string &filename)

#### Public Attributes

- std::vector< [Vertex](#) > **vertices** {}
- std::vector< uint32\_t > **indices** {}

The documentation for this struct was generated from the following file:

- include/VEngine/Model.hpp

### 3.5 ven::Camera Class Reference

#### Public Member Functions

- void **setOrthographicProjection** (float left, float right, float top, float bottom, float near, float far)
- void **setPerspectiveProjection** (float fovy, float aspect, float near, float far)
- void **setViewDirection** (glm::vec3 position, glm::vec3 direction, glm::vec3 up=glm::vec3{0.F, -1.F, 0.F})
- void **setViewTarget** (glm::vec3 position, glm::vec3 target, glm::vec3 up=glm::vec3{0.F, -1.F, 0.F})
- void **setViewYXZ** (glm::vec3 position, glm::vec3 rotation)
- const glm::mat4 & **getProjection** () const
- const glm::mat4 & **getView** () const

The documentation for this class was generated from the following file:

- include/VEngine/Camera.hpp

### 3.6 myLib::Clock Class Reference

#### Public Member Functions

- void **restart** ()
- void **pause** ()
- void **resume** ()
- [Time](#) **getElapsedTime** () const

The documentation for this class was generated from the following file:

- lib/static/myLib/include/myLib/Clock/Clock.hpp

## 3.7 ven::DescriptorPool Class Reference

### Classes

- class [Builder](#)

### Public Member Functions

- **DescriptorPool** ([Device](#) &device, uint32\_t maxSets, VkDescriptorPoolCreateFlags poolFlags, const std::vector< VkDescriptorPoolSize > &poolSizes)
- **DescriptorPool** (const [DescriptorPool](#) &)=delete
- [DescriptorPool](#) & **operator=** (const [DescriptorPool](#) &)=delete
- bool **allocateDescriptor** (const VkDescriptorSetLayout descriptorSetLayout, VkDescriptorSet &descriptor) const
- void **freeDescriptors** (std::vector< VkDescriptorSet > &descriptors) const
- void **resetPool** ()

### Friends

- class **DescriptorWriter**

The documentation for this class was generated from the following file:

- include/VEngine/Descriptors.hpp

## 3.8 ven::DescriptorSetLayout Class Reference

### Classes

- class [Builder](#)

### Public Member Functions

- **DescriptorSetLayout** ([Device](#) &device, std::unordered\_map< uint32\_t, VkDescriptorSetLayoutBinding > bindings)
- **DescriptorSetLayout** (const [DescriptorSetLayout](#) &)=delete
- [DescriptorSetLayout](#) & **operator=** (const [DescriptorSetLayout](#) &)=delete
- VkDescriptorSetLayout **getDescriptorSetLayout** () const

### Friends

- class **DescriptorWriter**

The documentation for this class was generated from the following file:

- include/VEngine/Descriptors.hpp

### 3.9 ven::DescriptorWriter Class Reference

#### Public Member Functions

- **DescriptorWriter** ([DescriptorSetLayout](#) &setLayout, [DescriptorPool](#) &pool)
- [DescriptorWriter](#) & **writeBuffer** (uint32\_t binding, VkDescriptorBufferInfo \*bufferInfo)
- [DescriptorWriter](#) & **writelnImage** (uint32\_t binding, VkDescriptorImageInfo \*imageInfo)
- bool **build** (VkDescriptorSet &set)
- void **overwrite** (VkDescriptorSet &set)

The documentation for this class was generated from the following file:

- include/VEngine/Descriptors.hpp

### 3.10 ven::Device Class Reference

#### Public Member Functions

- **Device** ([ven::Window](#) &window)
- **Device** (const [Device](#) &)=delete
- [Device](#) & **operator=** (const [Device](#) &)=delete
- **Device** ([Device](#) &&)=delete
- [Device](#) & **operator=** ([Device](#) &&)=delete
- VkCommandPool **getCommandPool** ()
- VkDevice **device** ()
- VkSurfaceKHR **surface** ()
- VkQueue **graphicsQueue** ()
- VkQueue **presentQueue** ()
- [SwapChainSupportDetails](#) **getSwapChainSupport** ()
- uint32\_t **findMemoryType** (uint32\_t typeFilter, VkMemoryPropertyFlags properties)
- [QueueFamilyIndices](#) **findPhysicalQueueFamilies** ()
- VkFormat **findSupportedFormat** (const std::vector< VkFormat > &candidates, VkImageTiling tiling, VkFormatFeatureFlags features)
- void **createBuffer** (VkDeviceSize size, VkBufferUsageFlags usage, VkMemoryPropertyFlags properties, VkBuffer &buffer, VkDeviceMemory &bufferMemory)
- VkCommandBuffer **beginSingleTimeCommands** ()
- void **endSingleTimeCommands** (VkCommandBuffer commandBuffer)
- void **copyBuffer** (VkBuffer srcBuffer, VkBuffer dstBuffer, VkDeviceSize size)
- void **copyBufferToImage** (VkBuffer buffer, VkImage image, uint32\_t width, uint32\_t height, uint32\_t layerCount)
- void **createImageWithInfo** (const VkImageCreateInfo &imageInfo, VkMemoryPropertyFlags properties, VkImage &image, VkDeviceMemory &imageMemory)

#### Public Attributes

- const bool **enableValidationLayers** = true
- VkPhysicalDeviceProperties **m\_properties**

The documentation for this class was generated from the following file:

- include/VEngine/Device.hpp



## 3.11 ven::Engine Class Reference

### Public Member Functions

- **Engine** (uint32\_t=DEFAULT\_WIDTH, uint32\_t=DEFAULT\_HEIGHT, const std::string &title=DEFAULT\_TITLE.data())
- **Engine** (const [Engine](#) &)=delete
- **Engine operator=** (const [Engine](#) &)=delete
- **Window** & **getWindow** ()
- void **mainLoop** ()

The documentation for this class was generated from the following file:

- include/VEngine/Engine.hpp

## 3.12 ven::FrameInfo Struct Reference

### Public Attributes

- int **frameIndex**
- float **frameTime**
- VkCommandBuffer **commandBuffer**
- [Camera](#) & **camera**
- VkDescriptorSet **globalDescriptorSet**

The documentation for this struct was generated from the following file:

- include/VEngine/FrameInfo.hpp

## 3.13 ven::KeyboardController Class Reference

### Classes

- struct [KeyMappings](#)

### Public Member Functions

- void **moveInPlaneXZ** (GLFWwindow \*window, float dt, [Object](#) &object) const

### Public Attributes

- [KeyMappings](#) **m\_keys** {}
- float **m\_moveSpeed** {3.F}
- float **m\_lookSpeed** {1.5F}

The documentation for this class was generated from the following file:

- include/VEngine/KeyboardController.hpp

### 3.14 ven::KeyboardController::KeyMappings Struct Reference

#### Public Attributes

- int **moveLeft** = GLFW\_KEY\_A
- int **moveRight** = GLFW\_KEY\_D
- int **moveForward** = GLFW\_KEY\_W
- int **moveBackward** = GLFW\_KEY\_S
- int **moveUp** = GLFW\_KEY\_SPACE
- int **moveDown** = GLFW\_KEY\_LEFT\_SHIFT
- int **lookLeft** = GLFW\_KEY\_LEFT
- int **lookRight** = GLFW\_KEY\_RIGHT
- int **lookUp** = GLFW\_KEY\_UP
- int **lookDown** = GLFW\_KEY\_DOWN

The documentation for this struct was generated from the following file:

- include/VEngine/KeyboardController.hpp

### 3.15 ven::Model Class Reference

#### Classes

- struct [Builder](#)
- struct [Vertex](#)

#### Public Member Functions

- **Model** ([Device](#) &device, const [Model::Builder](#) &builder)
- **Model** (const [Model](#) &)=delete
- void **operator=** (const [Model](#) &)=delete
- void **bind** (VkCommandBuffer commandBuffer)
- void **draw** (VkCommandBuffer commandBuffer) const

#### Static Public Member Functions

- static std::unique\_ptr< [Model](#) > **createModelFromFile** ([Device](#) &device, const std::string &filename)

The documentation for this class was generated from the following file:

- include/VEngine/Model.hpp

## 3.16 ven::Object Class Reference

### Public Member Functions

- **Object** (const [Object](#) &)=delete
- [Object](#) & **operator=** (const [Object](#) &)=delete
- **Object** ([Object](#) &&)=default
- [Object](#) & **operator=** ([Object](#) &&)=default
- `id_t` **getId** () const

### Static Public Member Functions

- static [Object](#) **createObject** ()

### Public Attributes

- `std::shared_ptr< ven::Model >` **model** {}
- `glm::vec3` **color** {}
- [Transform3DComponent](#) **transform3D** {}

The documentation for this class was generated from the following file:

- `include/VEngine/Object.hpp`

## 3.17 ven::PipelineConfigInfo Struct Reference

### Public Member Functions

- **PipelineConfigInfo** (const [PipelineConfigInfo](#) &)=delete
- [PipelineConfigInfo](#) & **operator=** (const [PipelineConfigInfo](#) &)=delete

### Public Attributes

- `VkPipelineInputAssemblyStateCreateInfo` **inputAssemblyInfo** {}
- `VkPipelineRasterizationStateCreateInfo` **rasterizationInfo** {}
- `VkPipelineMultisampleStateCreateInfo` **multisampleInfo** {}
- `VkPipelineColorBlendAttachmentState` **colorBlendAttachment** {}
- `VkPipelineColorBlendStateCreateInfo` **colorBlendInfo** {}
- `VkPipelineDepthStencilStateCreateInfo` **depthStencilInfo** {}
- `std::vector< VkDynamicState >` **dynamicStateEnables**
- `VkPipelineDynamicStateCreateInfo` **dynamicStateInfo** {}
- `VkPipelineLayout` **pipelineLayout** = nullptr
- `VkRenderPass` **renderPass** = nullptr
- `uint32_t` **subpass** = 0

The documentation for this struct was generated from the following file:

- `include/VEngine/Shaders.hpp`

## 3.18 gui::PluginLoader Class Reference

### Classes

- class [PluginLoaderException](#)

### Public Types

- using **PluginCreator** = std::unique\_ptr< IPlugin >(\*)()

### Public Member Functions

- template<typename T >  
std::unique\_ptr< T > **getPlugin** (const std::string &pluginName)
- void **closePlugins** ()

### Static Public Member Functions

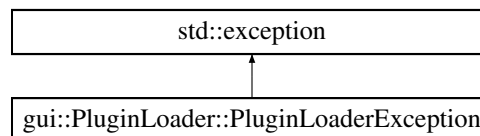
- static [PluginLoader](#) & **getInstance** ()

The documentation for this class was generated from the following file:

- include/VEngine/PluginLoader.hpp

## 3.19 gui::PluginLoader::PluginLoaderException Class Reference

Inheritance diagram for gui::PluginLoader::PluginLoaderException:



### Public Member Functions

- **PluginLoaderException** (std::string msg)
- const char \* **what** () const noexcept override

The documentation for this class was generated from the following file:

- include/VEngine/PluginLoader.hpp

## 3.20 ven::QueueFamilyIndices Struct Reference

### Public Member Functions

- bool **isComplete** () const

### Public Attributes

- uint32\_t **graphicsFamily** {}
- uint32\_t **presentFamily** {}
- bool **graphicsFamilyHasValue** = false
- bool **presentFamilyHasValue** = false

The documentation for this struct was generated from the following file:

- include/VEngine/Device.hpp

## 3.21 myLib::Random Class Reference

### Static Public Member Functions

- static int **randomInt** (int min, int max)
- static int **randomInt** ()
- static float **randomFloat** (float min, float max)
- static float **randomFloat** ()

The documentation for this class was generated from the following file:

- lib/static/myLib/include/myLib/Random.hpp

## 3.22 ven::Renderer Class Reference

### Public Member Functions

- **Renderer** ([Window](#) &>window, [Device](#) &device)
- **Renderer** (const [Renderer](#) &)=delete
- **Renderer** & **operator=** (const [Renderer](#) &)=delete
- VkRenderPass **getSwapChainRenderPass** () const
- float **getAspectRatio** () const
- bool **isFrameInProgress** () const
- VkCommandBuffer **getCurrentCommandBuffer** () const
- int **getFrameIndex** () const
- VkCommandBuffer **beginFrame** ()
- void **endFrame** ()
- void **beginSwapChainRenderPass** (VkCommandBuffer commandBuffer)

## Static Public Member Functions

- static void **endSwapChainRenderPass** (VkCommandBuffer commandBuffer)

The documentation for this class was generated from the following file:

- include/VEngine/Renderer.hpp

## 3.23 ven::RenderSystem Class Reference

### Public Member Functions

- **RenderSystem** ([Device](#) &device, VkRenderPass renderPass, VkDescriptorSetLayout globalSetLayout)
- **RenderSystem** (const [RenderSystem](#) &)=delete
- [RenderSystem](#) & **operator=** (const [RenderSystem](#) &)=delete
- void **renderObjects** ([FrameInfo](#) &frameInfo, std::vector< [ven::Object](#) > &objects)

The documentation for this class was generated from the following file:

- include/VEngine/RenderSystem.hpp

## 3.24 ven::Shaders Class Reference

### Public Member Functions

- **Shaders** ([Device](#) &device, const std::string &vertFilepath, const std::string &fragFilepath, const [PipelineConfigInfo](#) &configInfo)
- **Shaders** (const [Shaders](#) &)=delete
- [Shaders](#) & **operator=** (const [Shaders](#) &)=delete
- void **bind** (VkCommandBuffer commandBuffer)

### Static Public Member Functions

- static void **defaultPipelineConfigInfo** ([PipelineConfigInfo](#) &configInfo)

The documentation for this class was generated from the following file:

- include/VEngine/Shaders.hpp

## 3.25 ven::SimplePushConstantData Struct Reference

### Public Attributes

- glm::mat4 **modelMatrix** {1.F}
- glm::mat4 **normalMatrix** {1.F}

The documentation for this struct was generated from the following file:

- include/VEngine/RenderSystem.hpp

## 3.26 ven::SwapChain Class Reference

### Public Member Functions

- **SwapChain** ([Device](#) &deviceRef, VkExtent2D windowExtent)
- **SwapChain** ([Device](#) &deviceRef, VkExtent2D windowExtent, std::shared\_ptr< [SwapChain](#) > previous)
- **SwapChain** (const [SwapChain](#) &)=delete
- [SwapChain](#) & **operator=** (const [SwapChain](#) &)=delete
- VkFramebuffer **getFrameBuffer** (unsigned long index)
- VkRenderPass **getRenderPass** ()
- VkImageView **getImageView** (int index)
- size\_t **imageCount** ()
- VkFormat **getSwapChainImageFormat** ()
- VkExtent2D **getSwapChainExtent** ()
- uint32\_t **width** () const
- uint32\_t **height** () const
- float **extentAspectRatio** () const
- VkFormat **findDepthFormat** ()
- VkResult **acquireNextImage** (uint32\_t \*imageIndex)
- VkResult **submitCommandBuffers** (const VkCommandBuffer \*buffers, const uint32\_t \*imageIndex)
- bool **compareSwapFormats** (const [SwapChain](#) &swapChainp) const

### Static Public Attributes

- static constexpr int **MAX\_FRAMES\_IN\_FLIGHT** = 2

The documentation for this class was generated from the following file:

- include/VEngine/SwapChain.hpp

## 3.27 ven::SwapChainSupportDetails Struct Reference

### Public Attributes

- VkSurfaceCapabilitiesKHR **capabilities**
- std::vector< VkSurfaceFormatKHR > **formats**
- std::vector< VkPresentModeKHR > **presentModes**

The documentation for this struct was generated from the following file:

- include/VEngine/Device.hpp

## 3.28 myLib::Time Class Reference

### Public Member Functions

- **Time** (const double seconds)
- int **asSeconds** () const
- int **asMilliseconds** () const
- int **asMicroseconds** () const

The documentation for this class was generated from the following file:

- lib/static/myLib/include/myLib/Clock/Time.hpp

## 3.29 ven::Transform3DComponent Struct Reference

### Public Member Functions

- glm::mat4 **mat4** () const
- glm::mat3 **normalMatrix** ()

### Public Attributes

- glm::vec3 **translation** {}
- glm::vec3 **scale** {1.F, 1.F, 1.F}
- glm::vec3 **rotation** {}

The documentation for this struct was generated from the following file:

- include/VEngine/Object.hpp

## 3.30 ven::Model::Vertex Struct Reference

### Public Member Functions

- bool **operator==** (const [Vertex](#) &other) const

### Static Public Member Functions

- static std::vector< VkVertexInputBindingDescription > **getBindingDescriptions** ()
- static std::vector< VkVertexInputAttributeDescription > **getAttributeDescriptions** ()



## Public Attributes

- glm::vec3 **position** {}
- glm::vec3 **color** {}
- glm::vec3 **normal** {}
- glm::vec2 **uv** {}

The documentation for this struct was generated from the following file:

- include/VEngine/Model.hpp

## 3.31 ven::Window Class Reference

### Public Member Functions

- **Window** (const uint32\_t width, const uint32\_t height, const std::string &title)
- GLFWwindow \* **createWindow** (uint32\_t width, uint32\_t height, const std::string &title)
- void **createWindowSurface** (VkInstance instance, VkSurfaceKHR \*surface)
- GLFWwindow \* **getGLFWWindow** () const
- VkExtent2D **getExtent** () const
- bool **wasWindowResized** () const
- void **resetWindowResizedFlag** ()

The documentation for this class was generated from the following file:

- include/VEngine/Window.hpp



# Index

- descriptorInfo
  - ven::Buffer, [5](#)
- descriptorInfoForIndex
  - ven::Buffer, [6](#)
- flush
  - ven::Buffer, [6](#)
- flushIndex
  - ven::Buffer, [7](#)
- gui::PluginLoader, [16](#)
- gui::PluginLoader::PluginLoaderException, [16](#)
- invalidate
  - ven::Buffer, [7](#)
- invalidateIndex
  - ven::Buffer, [7](#)
- map
  - ven::Buffer, [8](#)
- myLib::Clock, [10](#)
- myLib::Random, [17](#)
- myLib::Time, [20](#)
- unmap
  - ven::Buffer, [8](#)
- ven::Buffer, [5](#)
  - descriptorInfo, [5](#)
  - descriptorInfoForIndex, [6](#)
  - flush, [6](#)
  - flushIndex, [7](#)
  - invalidate, [7](#)
  - invalidateIndex, [7](#)
  - map, [8](#)
  - unmap, [8](#)
  - writeToBuffer, [8](#)
  - writeToIndex, [9](#)
- ven::Camera, [10](#)
- ven::DescriptorPool, [11](#)
- ven::DescriptorPool::Builder, [9](#)
- ven::DescriptorSetLayout, [11](#)
- ven::DescriptorSetLayout::Builder, [9](#)
- ven::DescriptorWriter, [12](#)
- ven::Device, [12](#)
- ven::Engine, [13](#)
- ven::FrameInfo, [13](#)
- ven::KeyboardController, [13](#)
- ven::KeyboardController::KeyMappings, [14](#)
- ven::Model, [14](#)
- ven::Model::Builder, [10](#)
- ven::Model::Vertex, [20](#)
- ven::Object, [15](#)
- ven::PipelineConfigInfo, [15](#)
- ven::QueueFamilyIndices, [17](#)
- ven::Renderer, [17](#)
- ven::RenderSystem, [18](#)
- ven::Shaders, [18](#)
- ven::SimplePushConstantData, [18](#)
- ven::SwapChain, [19](#)
- ven::SwapChainSupportDetails, [19](#)
- ven::Transform3DComponent, [20](#)
- ven::Window, [21](#)
- writeToBuffer
  - ven::Buffer, [8](#)
- writeToIndex
  - ven::Buffer, [9](#)