# vengine

0.1.0

Generated by Doxygen 1.9.1

**1 Hierarchical Index** 1

1.1 Class Hierarchy . . . 1

**2 Class Index** 3

2.1 Class List . . . 3

**3 Class Documentation** 5

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::GlobalUbo Struct Reference . . . 13

3.14 ven::KeyboardController Class Reference . . . 14

3.15 ven::KeyboardController::KeyMappings Struct Reference . . . 14

3.16 ven::Model Class Reference . . . 14

3.17 ven::Object Class Reference . . . 15

3.18 ven::PipelineConfigInfo Struct Reference . . . 16

3.19 gui::PluginLoader Class Reference . . . 16

3.20 gui::PluginLoader::PluginLoaderException Class Reference . . . 17

3.21 ven::PointLight Struct Reference . . . 17

3.22 ven::PointLightComponent Struct Reference . . . 17

3.23 ven::PointLightSystem Class Reference . . . 18

3.24 ven::QueueFamilyIndices Struct Reference . . . 18

3.25 myLib::Random Class Reference . . . 18

3.26 ven::Renderer Class Reference . . . . . 19
3.27 ven::RenderSystem Class Reference . . . . . 19
3.28 ven::Shaders Class Reference . . . . . 19
3.29 ven::SimplePushConstantData Struct Reference . . . . . 20
3.30 ven::SwapChain Class Reference . . . . . 20
3.31 ven::SwapChainSupportDetails Struct Reference . . . . . 21
3.32 myLib::Time Class Reference . . . . . 21
3.33 ven::Transform3DComponent Struct Reference . . . . . 21
3.34 ven::Model::Vertex Struct Reference . . . . . 22
3.35 ven::Window Class Reference . . . . . 22

**Index** **23**

# 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 . . . 17
ven::FrameInfo . . . 13
ven::GlobalUbo . . . 13
ven::KeyboardController . . . 14
ven::KeyboardController::KeyMappings . . . 14
ven::Model . . . 14
ven::Object . . . 15
ven::PipelineConfigInfo . . . 16
gui::PluginLoader . . . 16
ven::PointLight . . . 17
ven::PointLightComponent . . . 17
ven::PointLightSystem . . . 18
ven::QueueFamilyIndices . . . 18
myLib::Random . . . 18
ven::Renderer . . . 19
ven::RenderSystem . . . 19
ven::Shaders . . . 19
ven::SimplePushConstantData . . . 20
ven::SwapChain . . . 20
ven::SwapChainSupportDetails . . . 21
myLib::Time . . . 21
ven::Transform3DComponent . . . 21
ven::Model::Vertex . . . 22
ven::Window . . . 22

# Chapter 2

# Class Index

## 2.1 Class List

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


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
ven::FrameInfo . . . 13
ven::GlobalUbo . . . 13
ven::KeyboardController . . . 14
ven::KeyboardController::KeyMappings . . . 14
ven::Model . . . 14
ven::Object . . . 15
ven::PipelineConfigInfo . . . 16
gui::PluginLoader . . . 16
gui::PluginLoader::PluginLoaderException . . . 17
ven::PointLight . . . 17
ven::PointLightComponent . . . 17
ven::PointLightSystem . . . 18
ven::QueueFamilyIndices . . . 18
myLib::Random . . . 18
ven::Renderer . . . 19
ven::RenderSystem . . . 19
ven::Shaders . . . 19
ven::SimplePushConstantData . . . 20
ven::SwapChain . . . 20
ven::SwapChainSupportDetails . . . 21
myLib::Time . . . 21
ven::Transform3DComponent . . . 21
ven::Model::Vertex . . . 22
ven::Window . . . 22

# 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**

| | |
|---|---|
| *size* | (Optional) Size of the memory range of the descriptor |
| *offset* | (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**

| | |
|---|---|
| *index* | 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**

| | |
|---|---|
| *size* | (Optional) Size of the memory range to flush. Pass VK_WHOLE_SIZE to flush the complete buffer range. |
| *offset* | (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**

| | |
|---|---|
| *index* | 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**

| | |
|---|---|
| *size* | (Optional) Size of the memory range to invalidate. Pass VK_WHOLE_SIZE to invalidate the complete buffer range. |
| *offset* | (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**

| | |
|---|---|
| *index* | Specifies the region to invalidate: index ∗ 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**

| | |
|---|---|
| *size* | (Optional) Size of the memory range to map. Pass VK_WHOLE_SIZE to map the complete buffer range. |
| *offset* | (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**

| | |
|---|---|
| *data* | Pointer to the data to copy |
| *size* | (Optional) Size of the data to copy. Pass VK_WHOLE_SIZE to flush the complete buffer range. |
| *offset* | (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**

| | |
|---|---|
| *data* | Pointer to the data to copy |
| *index* | 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 & **writeImage** (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**
- Object::Map & **objects**

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

- include/VEngine/FrameInfo.hpp

## 3.13 ven::GlobalUbo Struct Reference

### Public Attributes

- glm::mat4 **projection** {1.F}
- glm::mat4 **view** {1.F}
- glm::vec4 **ambientLightColor** {1.F, 1.F, 1.F, .02F}
- PointLight **pointLights** [MAX_LIGHTS]
- int **numLights**

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

- include/VEngine/FrameInfo.hpp

## 3.14 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

## 3.15 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

## 3.16 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

## 3.17 ven::Object Class Reference

### Public Types

- using **Map** = std::unordered_map< id_t, Object >

### 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** ()
- static Object **makePointLight** (float intensity=10.F, float radius=0.1F, glm::vec3 color=glm::vec3(1.F))

### Public Attributes

- std::shared_ptr< ven::Model > **model** {}
- glm::vec3 **color** {}
- Transform3DComponent **transform3D** {}
- std::unique_ptr< PointLightComponent > **pointLight** = nullptr

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

- include/VEngine/Object.hpp

## 3.18 ven::PipelineConfigInfo Struct Reference

### Public Member Functions

- **PipelineConfigInfo** (const PipelineConfigInfo &)=delete
- PipelineConfigInfo & **operator=** (const PipelineConfigInfo &)=delete

### Public Attributes

- std::vector< VkVertexInputBindingDescription > **bindingDescriptions** {}
- std::vector< VkVertexInputAttributeDescription > **attributeDescriptions** {}
- 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

## 3.19 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

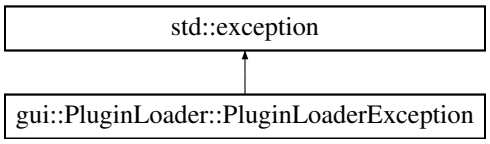
- static PluginLoader & **getInstance** ()

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

- include/VEngine/PluginLoader.hpp

## 3.20 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

## 3.21 ven::PointLight Struct Reference

### Public Attributes

- glm::vec4 **position** {}
- glm::vec4 **color** {}

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

- include/VEngine/FrameInfo.hpp

## 3.22 ven::PointLightComponent Struct Reference

### Public Attributes

- float **lightIntensity** = 1.0F

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

- include/VEngine/Object.hpp

## 3.23 ven::PointLightSystem Class Reference

### Public Member Functions

- **PointLightSystem** (Device &device, VkRenderPass renderPass, VkDescriptorSetLayout globalSetLayout)
- **PointLightSystem** (const PointLightSystem &)=delete
- PointLightSystem & **operator=** (const PointLightSystem &)=delete
- void **update** (FrameInfo &frameInfo, GlobalUbo &ubo)
- void **render** (FrameInfo &frameInfo)

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

- include/VEngine/System/PointLightSystem.hpp

## 3.24 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

## 3.25 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

## 3.26 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

## 3.27 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)

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

- include/VEngine/System/RenderSystem.hpp

## 3.28 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

## 3.29 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/System/RenderSystem.hpp

## 3.30 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

## 3.31 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

## 3.32 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

## 3.33 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

## 3.34 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

## 3.35 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 * **getGLFWindow** () 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, 17

invalidate
    ven::Buffer, 7
invalidateIndex
    ven::Buffer, 7

map
    ven::Buffer, 8
myLib::Clock, 10
myLib::Random, 18
myLib::Time, 21

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::GlobalUbo, 13
ven::KeyboardController, 14
ven::KeyboardController::KeyMappings, 14
ven::Model, 14
ven::Model::Builder, 10
ven::Model::Vertex, 22
ven::Object, 15
ven::PipelineConfigInfo, 16
ven::PointLight, 17
ven::PointLightComponent, 17
ven::PointLightSystem, 18
ven::QueueFamilyIndices, 18
ven::Renderer, 19
ven::RenderSystem, 19
ven::Shaders, 19
ven::SimplePushConstantData, 20
ven::SwapChain, 20
ven::SwapChainSupportDetails, 21
ven::Transform3DComponent, 21
ven::Window, 22

writeToBuffer
    ven::Buffer, 8
writeToIndex
    ven::Buffer, 9