vengine

0.1.0

Generated by Doxygen 1.9.1

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 3

Class Documentation

3.1 ven::Buffer Class Reference

Class for buffer.

`#include <Buffer.hpp>`

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)

  Map a memory range of this buffer. If successful, mapped points to the specified buffer range.
- void unmap ()

  Unmap a mapped memory range.
- void writeToBuffer (const void *data, VkDeviceSize size=VK_WHOLE_SIZE, VkDeviceSize offset=0) const

  Copies the specified data to the mapped buffer. Default value writes whole buffer range.
- VkResult flush (VkDeviceSize size=VK_WHOLE_SIZE, VkDeviceSize offset=0) const

  Flush a memory range of the buffer to make it visible to the device.
- VkDescriptorBufferInfo descriptorInfo (const VkDeviceSize size=VK_WHOLE_SIZE, const VkDeviceSize offset=0) const

  Create a buffer info descriptor.
- VkResult invalidate (VkDeviceSize size=VK_WHOLE_SIZE, VkDeviceSize offset=0) const

  Invalidate a memory range of the buffer to make it visible to the host.
- void writeToIndex (const void *data, const VkDeviceSize index) const
- VkResult flushIndex (const VkDeviceSize index) const
- VkDescriptorBufferInfo descriptorInfoForIndex (const VkDeviceSize index) const
- VkResult invalidateIndex (const VkDeviceSize index) const
- 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 Detailed Description

Class for buffer.

3.1.2 Member Function Documentation

3.1.2.1 descriptorInfo()

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

3.1.2.2 descriptorInfoForIndex()

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

Create a buffer info descriptor

Parameters

index	Specifies the region given by index ∗ alignmentSize

Returns

VkDescriptorBufferInfo for instance at index

3.1.2.3 flush()

```
VkResult ven::Buffer::flush (
```

```
            VkDeviceSize size = VK_WHOLE_SIZE,
            VkDeviceSize offset = 0 ) const
```

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

3.1.2.4 flushIndex()

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

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

Parameters

index	Used in offset calculation

3.1.2.5 invalidate()

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

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

3.1.2.6 invalidateIndex()

```
VkResult ven::Buffer::invalidateIndex (
            const VkDeviceSize index ) const  [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

3.1.2.7 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

3.1.2.8 unmap()

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

Unmap a mapped memory range.

Note

Does not return a result as vkUnmapMemory can't fail

3.1.2.9 writeToBuffer()

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

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

3.1.2.10 writeToIndex()

```
void ven::Buffer::writeToIndex (
            const void * data,
            const VkDeviceSize index ) const  [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
- const glm::mat4 & **getInverseView** () const

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

- include/VEngine/Camera.hpp

3.6 myLib::Clock Class Reference

Class for time management.

`#include <Clock.hpp>`

Public Member Functions

- void restart ()
  Restart the clock.
- void pause ()
  Pause the clock.
- void resume ()
  Resume the clock.
- Time getElapsedTime () const
  Get the elapsed time since the last restart.

3.6.1 Detailed Description

Class for time management.

3.6.2 Member Function Documentation

3.6.2.1 getElapsedTime()

`Time myLib::Clock::getElapsedTime ( ) const`

Get the elapsed time since the last restart.

Returns

Time The elapsed time

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

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

3.7 ven::DescriptorPool Class Reference

Class for descriptor pool.

`#include <Descriptors.hpp>`

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** (VkDescriptorSetLayout descriptorSetLayout, VkDescriptorSet &descriptor) const
- void **freeDescriptors** (const std::vector< VkDescriptorSet > &descriptors) const
- void **resetPool** () const

Friends

- class **DescriptorWriter**

3.7.1 Detailed Description

Class for descriptor pool.

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

- include/VEngine/Descriptors.hpp

3.8 ven::DescriptorSetLayout Class Reference

Class for descriptor set layout.

`#include <Descriptors.hpp>`

Classes

- class Builder

Public Member Functions

- **DescriptorSetLayout** (Device &device, const std::unordered_map< uint32_t, VkDescriptorSetLayoutBinding > &bindings)
- **DescriptorSetLayout** (const DescriptorSetLayout &)=delete
- DescriptorSetLayout & **operator=** (const DescriptorSetLayout &)=delete
- VkDescriptorSetLayout **getDescriptorSetLayout** () const

Friends

- class **DescriptorWriter**

3.8.1 Detailed Description

Class for descriptor set layout.

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

- include/VEngine/Descriptors.hpp

3.9 ven::DescriptorWriter Class Reference

Class for descriptor writer.

`#include <Descriptors.hpp>`

Public Member Functions

- **DescriptorWriter** (DescriptorSetLayout &setLayout, DescriptorPool &pool)
- DescriptorWriter & **writeBuffer** (uint32_t binding, const VkDescriptorBufferInfo *bufferInfo)
- DescriptorWriter & **writeImage** (uint32_t binding, const VkDescriptorImageInfo *imageInfo)
- bool **build** (VkDescriptorSet &set)
- void **overwrite** (const VkDescriptorSet &set)

3.9.1 Detailed Description

Class for descriptor writer.

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** (Window &window)
- **Device** (const Device &)=delete
- Device & **operator=** (const Device &)=delete
- **Device** (Device &&)=delete
- Device & **operator=** (Device &&)=delete
- VkCommandPool **getCommandPool** () const
- VkDevice **device** () const
- VkSurfaceKHR **surface** () const
- VkQueue **graphicsQueue** () const
- VkQueue **presentQueue** () const
- SwapChainSupportDetails **getSwapChainSupport** () const
- uint32_t **findMemoryType** (uint32_t typeFilter, VkMemoryPropertyFlags propertiesp) const
- QueueFamilyIndices **findPhysicalQueueFamilies** () const
- VkFormat **findSupportedFormat** (const std::vector< VkFormat > &candidates, VkImageTiling tiling, Vk↩ FormatFeatureFlags features) const
- void **createBuffer** (VkDeviceSize size, VkBufferUsageFlags usage, VkMemoryPropertyFlags propertiesp, VkBuffer &buffer, VkDeviceMemory &bufferMemory) const
- VkCommandBuffer **beginSingleTimeCommands** () const
- void **endSingleTimeCommands** (VkCommandBuffer commandBuffer) const
- void **copyBuffer** (VkBuffer srcBuffer, VkBuffer dstBuffer, VkDeviceSize size) const
- void **copyBufferToImage** (VkBuffer buffer, VkImage image, uint32_t width, uint32_t height, uint32_t layer↩ Count) const
- void **createImageWithInfo** (const VkImageCreateInfo &imageInfo, VkMemoryPropertyFlags properties, VkImage &image, VkDeviceMemory &imageMemory) const
- VkPhysicalDevice **getPhysicalDevice** () const
- VkQueue **getGraphicsQueue** () const

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::FrameCounter Class Reference

Public Member Functions

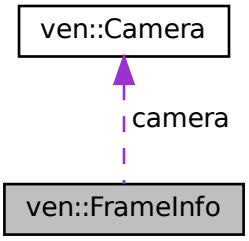
- void **update** (const float deltaTime)
- float **getFps** () const
- float **getFrameTime** () const

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

- include/VEngine/FrameCounter.hpp

3.13 ven::FrameInfo Struct Reference

Collaboration diagram for ven::FrameInfo:



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

3.14 ven::GlobalUbo Struct Reference

Public Attributes

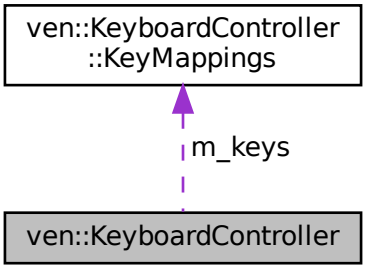
- glm::mat4 **projection** {1.F}
- glm::mat4 **view** {1.F}
- glm::mat4 **inverseView** {1.F}
- glm::vec4 **ambientLightColor** {1.F, 1.F, 1.F, .02F}
- std::array< PointLight, MAX_LIGHTS > **pointLights**
- int **numLights**

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

- include/VEngine/FrameInfo.hpp

3.15 ven::KeyboardController Class Reference

Collaboration diagram for ven::KeyboardController:



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

3.16 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

3.17 ven::Model Class Reference

Classes

- struct Builder
- struct Vertex

Public Member Functions

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

Static Public Member Functions

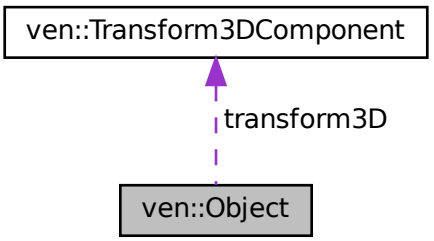
- static std::unique_ptr< Model > **createModelFromFile** (Device &device, const std::string &filename)

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

- include/VEngine/Model.hpp

3.18 ven::Object Class Reference

Collaboration diagram for ven::Object:



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< 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

3.19 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

3.20 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

3.21 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

3.22 ven::PointLightSystem Class Reference

Class for point light system.

`#include <PointLightSystem.hpp>`

Public Member Functions

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

Static Public Member Functions

- static void **update** (const FrameInfo &frameInfo, GlobalUbo &ubo)

3.22.1 Detailed Description

Class for point light system.

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

- include/VEngine/System/PointLightSystem.hpp

3.23 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

3.24 myLib::Random Class Reference

Class for random number generation.

`#include <Random.hpp>`

Static Public Member Functions

- static int randomInt (int min, int max)

  Generate a random integer between min and max.
- static int **randomInt** ()
- static float randomFloat (float min, float max)
- static float **randomFloat** ()

3.24.1 Detailed Description

Class for random number generation.

3.24.2 Member Function Documentation

3.24.2.1 randomFloat()

```
static float myLib::Random::randomFloat (
            float min,
            float max )  [static]
```

Parameters

min	The minimum value
max	The maximum value

Returns

float The random float

3.24.2.2 randomInt()

```
static int myLib::Random::randomInt (
            int min,
            int max )  [static]
```

Generate a random integer between min and max.

Parameters

min	The minimum value
max	The maximum value

Returns

int The random integer

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

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

3.25 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) const

Static Public Member Functions

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

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

- include/VEngine/Renderer.hpp

3.26 ven::RenderSystem Class Reference

Class for render system.

```
#include <RenderSystem.hpp>
```

Public Member Functions

- **RenderSystem** (Device &device, VkRenderPass renderPass, VkDescriptorSetLayout globalSetLayout)
- **RenderSystem** (const RenderSystem &)=delete
- RenderSystem & **operator=** (const RenderSystem &)=delete
- void **renderObjects** (const FrameInfo &frameInfo) const

3.26.1 Detailed Description

Class for render system.

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

- include/VEngine/System/RenderSystem.hpp

3.27 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** (const VkCommandBuffer commandBuffer) const

Static Public Member Functions

- static void **defaultPipelineConfigInfo** (PipelineConfigInfo &configInfo)

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

- include/VEngine/Shaders.hpp

3.28 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

3.29 ven::SwapChain Class Reference

Public Member Functions

- **SwapChain** (Device &deviceRef, const VkExtent2D windowExtentRef)
- **SwapChain** (Device &deviceRef, const VkExtent2D windowExtentRef, std::shared_ptr< SwapChain > previous)
- **SwapChain** (const SwapChain &)=delete
- SwapChain & **operator=** (const SwapChain &)=delete
- VkFramebuffer **getFrameBuffer** (const unsigned long index) const
- VkRenderPass **getRenderPass** () const
- VkImageView **getImageView** (const int index) const
- size_t **imageCount** () const
- VkFormat **getSwapChainImageFormat** () const
- VkExtent2D **getSwapChainExtent** () const
- uint32_t **width** () const
- uint32_t **height** () const
- float **extentAspectRatio** () const
- VkFormat **findDepthFormat** () const
- VkResult **acquireNextImage** (uint32_t *imageIndex) const
- 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

3.30 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

3.31 myLib::Time Class Reference

Class used for time management.

`#include <Time.hpp>`

Public Member Functions

- Time (const double seconds)

  Construct a new Time object.
- int asSeconds () const

  Transform the time to seconds.
- int asMilliseconds () const

  Transform the time to milliseconds.
- int asMicroseconds () const

  Transform the time to microseconds.

3.31.1 Detailed Description

Class used for time management.

3.31.2 Member Function Documentation

3.31.2.1 asMicroseconds()

```
int myLib::Time::asMicroseconds ( ) const  [inline]
```

Transform the time to microseconds.

Returns

int The time in microseconds

3.31.2.2 asMilliseconds()

```
int myLib::Time::asMilliseconds ( ) const  [inline]
```

Transform the time to milliseconds.

Returns

int The time in milliseconds

3.31.2.3 asSeconds()

```
int myLib::Time::asSeconds ( ) const  [inline]
```

Transform the time to seconds.

Returns

int The time in seconds

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

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

3.32 ven::Transform3DComponent Struct Reference

Public Member Functions

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

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

3.33 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

3.34 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) const
- 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

Chapter 4

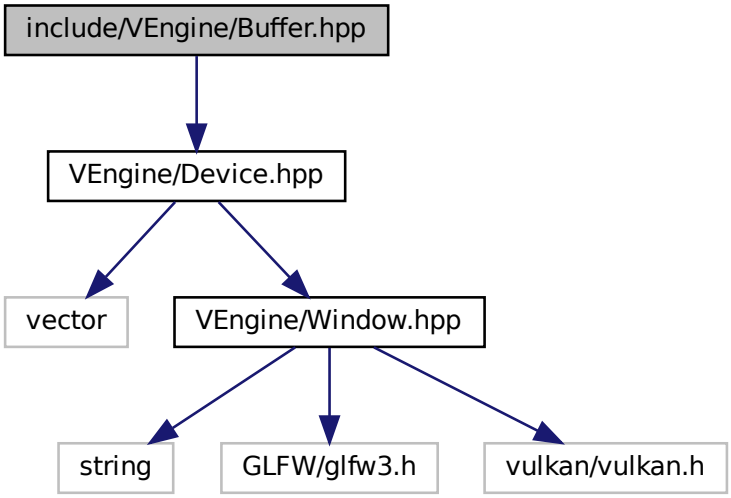
File Documentation

4.1 include/VEngine/Buffer.hpp File Reference

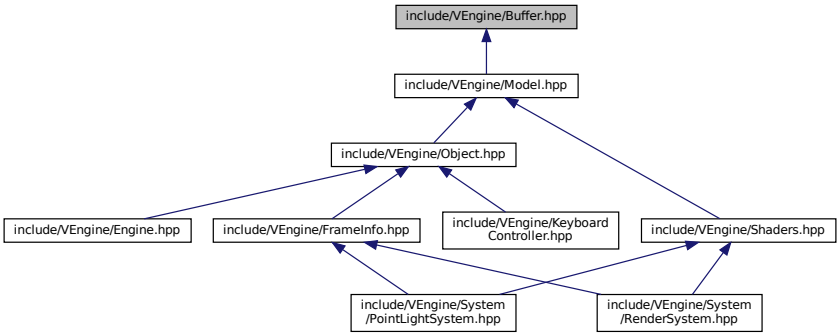
This file contains the Buffer class.

```
#include "VEngine/Device.hpp"
```

Include dependency graph for Buffer.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ven::Buffer

  Class for buffer.

4.1.1 Detailed Description

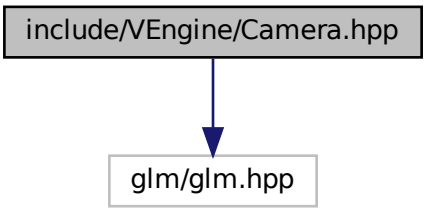
This file contains the Buffer class.

4.2 include/VEngine/Camera.hpp File Reference

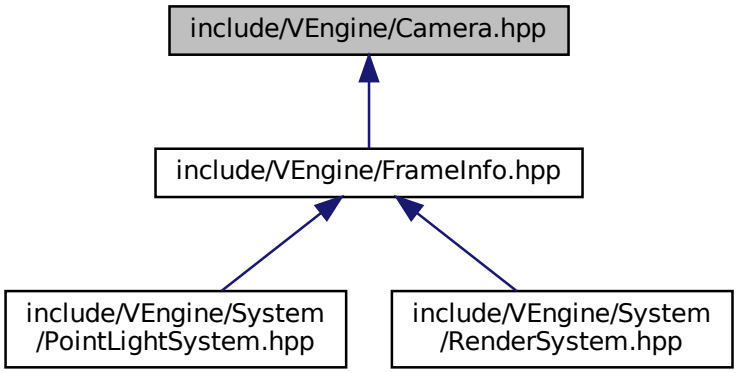
This file contains the Camera class.

```
#include <glm/glm.hpp>
```

Include dependency graph for Camera.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ven::Camera

4.2.1 Detailed Description

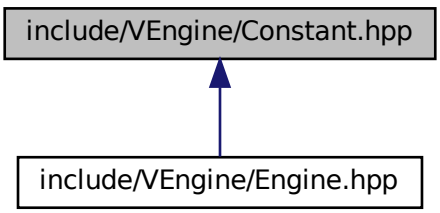
This file contains the Camera class.

This file contains the KeyboardController class.

4.3 include/VEngine/Constant.hpp File Reference

This file contains the constant values used in the project.

This graph shows which files directly or indirectly include this file:



Typedefs

- using **ven::return_type_t** = enum ReturnType :uint8_t { VEN_SUCCESS=0, VEN_FAILURE=1 }

4.3.1 Detailed Description

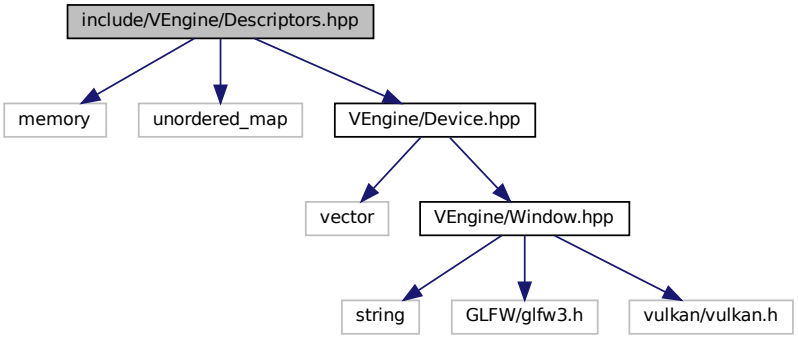
This file contains the constant values used in the project.

4.4 include/VEngine/Descriptors.hpp File Reference

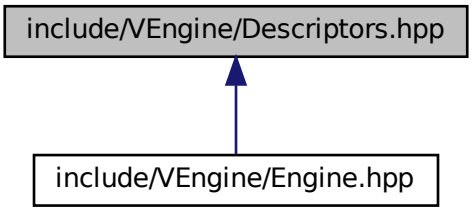
This file contains the Descriptors class.

```
#include <memory>
#include <unordered_map>
#include "VEngine/Device.hpp"
```

Include dependency graph for Descriptors.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ven::DescriptorSetLayout

  Class for descriptor set layout.
- class ven::DescriptorSetLayout::Builder
- class ven::DescriptorPool

  Class for descriptor pool.
- class ven::DescriptorPool::Builder
- class ven::DescriptorWriter

  Class for descriptor writer.

4.4.1 Detailed Description

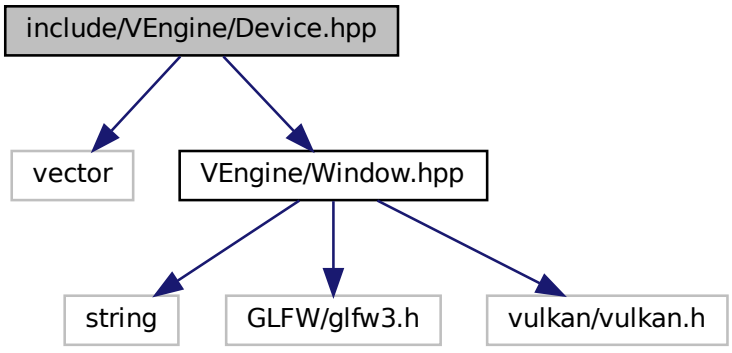
This file contains the Descriptors class.

4.5 include/VEngine/Device.hpp File Reference

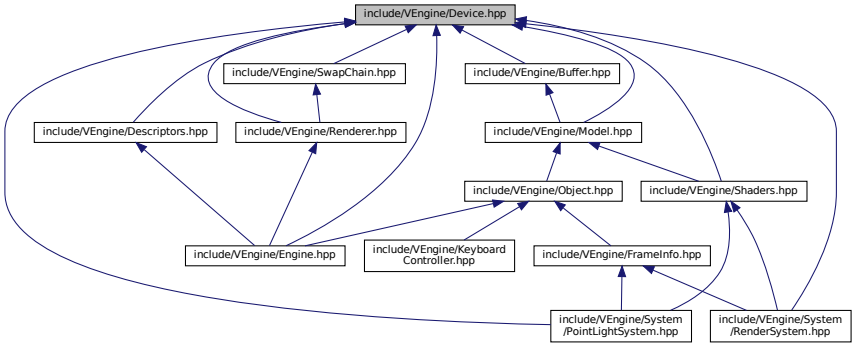
This file contains the Device class.

```
#include <vector>
#include "VEngine/Window.hpp"
```

Include dependency graph for Device.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- struct ven::SwapChainSupportDetails
- struct ven::QueueFamilyIndices
- class ven::Device

4.5.1 Detailed Description

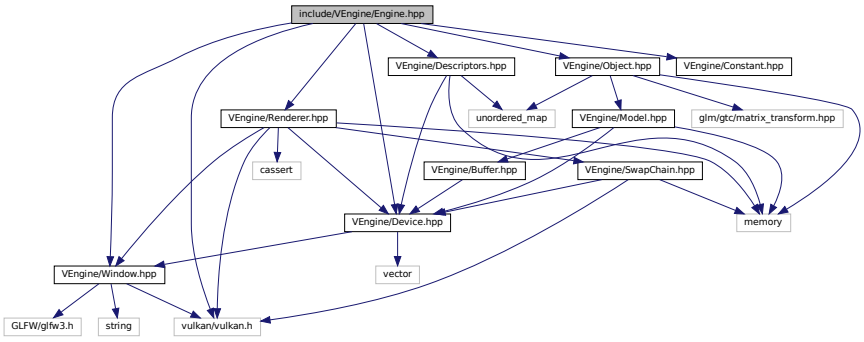
This file contains the Device class.

4.6 include/VEngine/Engine.hpp File Reference

This file contains the Engine class.

```
#include <vulkan/vulkan.h>
#include "VEngine/Window.hpp"
#include "VEngine/Constant.hpp"
#include "VEngine/Device.hpp"
#include "VEngine/Object.hpp"
#include "VEngine/Renderer.hpp"
#include "VEngine/Descriptors.hpp"
```

Include dependency graph for Engine.hpp:



Classes

- class ven::Engine

4.6.1 Detailed Description

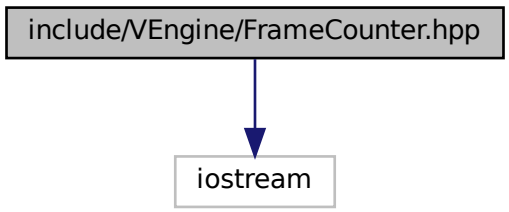
This file contains the Engine class.

4.7 include/VEngine/FrameCounter.hpp File Reference

This file contains the FrameCounter class.

```
#include <iostream>
```

Include dependency graph for FrameCounter.hpp:



Classes

- class ven::FrameCounter

4.7.1 Detailed Description

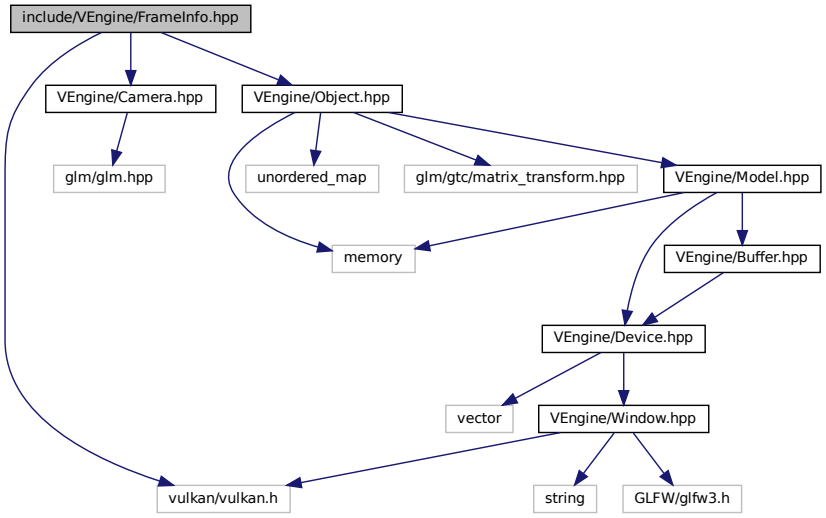
This file contains the FrameCounter class.

4.8 include/VEngine/FrameInfo.hpp File Reference

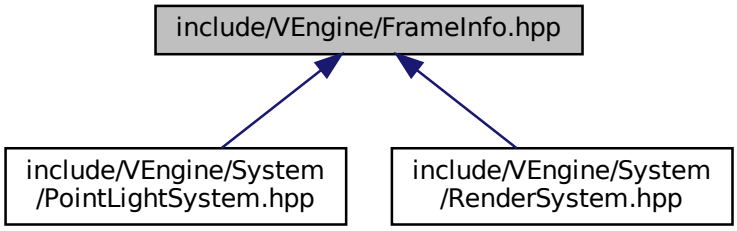
This file contains the FrameInfo class.

```
#include <vulkan/vulkan.h>
#include "VEngine/Camera.hpp"
#include "VEngine/Object.hpp"
```

Include dependency graph for FrameInfo.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- struct ven::PointLight
- struct ven::GlobalUbo
- struct ven::FrameInfo

4.8.1 Detailed Description

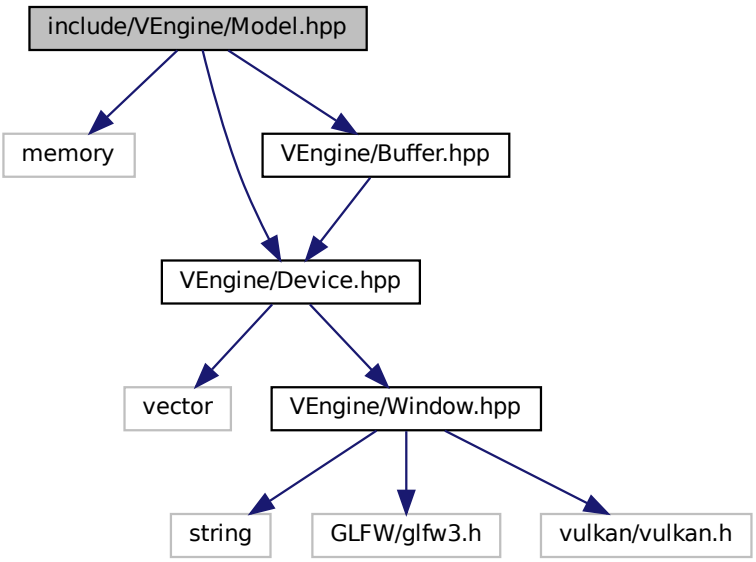
This file contains the FrameInfo class.

4.9 include/VEngine/Model.hpp File Reference

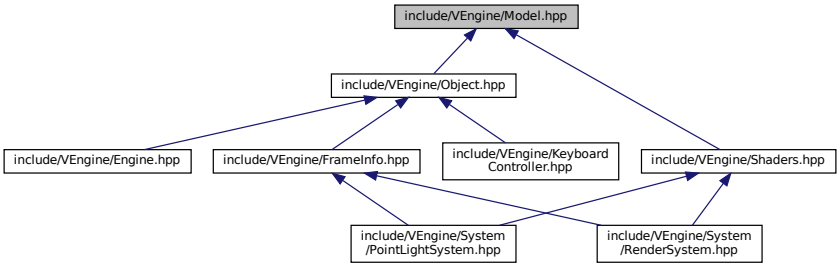
This file contains the Model class.

```
#include <memory>
#include "VEngine/Device.hpp"
#include "VEngine/Buffer.hpp"
```

Include dependency graph for Model.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ven::Model
- struct ven::Model::Vertex
- struct ven::Model::Builder

4.9.1 Detailed Description

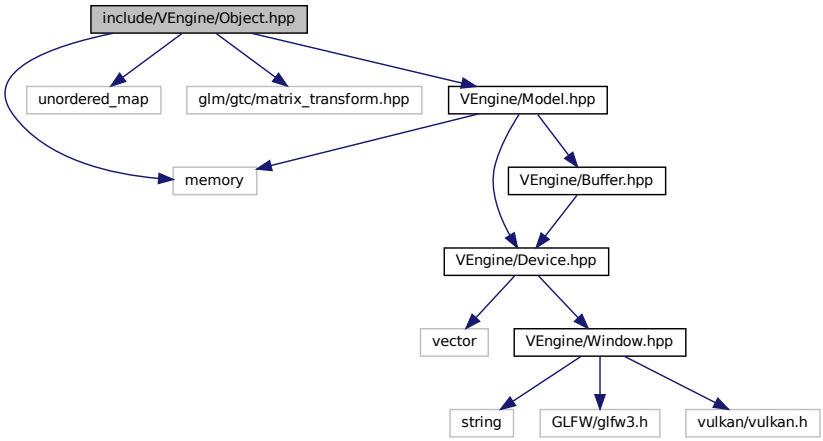
This file contains the Model class.

4.10 include/VEngine/Object.hpp File Reference

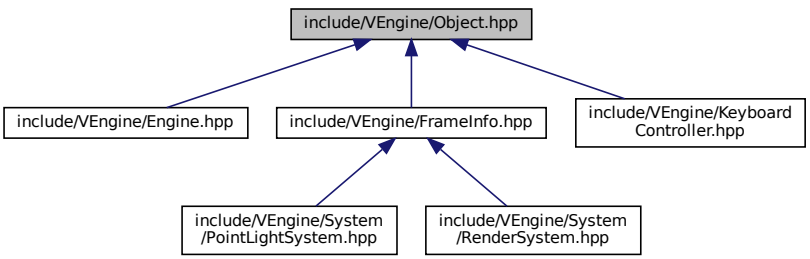
This file contains the Object class.

```
#include <memory>
#include <unordered_map>
#include <glm/gtc/matrix_transform.hpp>
#include "VEngine/Model.hpp"
```

Include dependency graph for Object.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- struct ven::Transform3DComponent
- struct ven::PointLightComponent
- class ven::Object

Typedefs

- using **ven::id_t** = unsigned int

4.10.1 Detailed Description

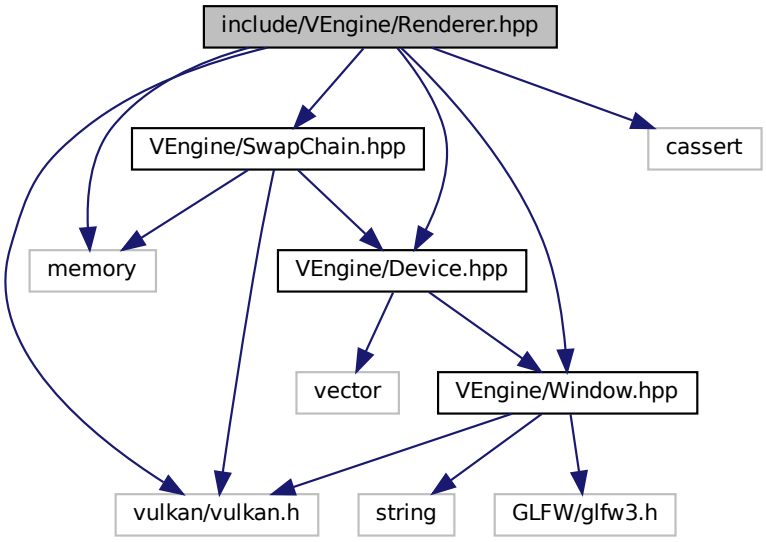
This file contains the Object class.

4.11 include/VEngine/Renderer.hpp File Reference

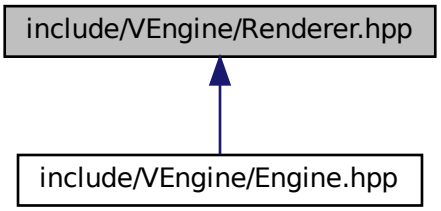
This file contains the Renderer class.

```
#include <memory>
#include <cassert>
#include <vulkan/vulkan.h>
#include "VEngine/Window.hpp"
#include "VEngine/Device.hpp"
#include "VEngine/SwapChain.hpp"
```

Include dependency graph for Renderer.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ven::Renderer

4.11.1 Detailed Description

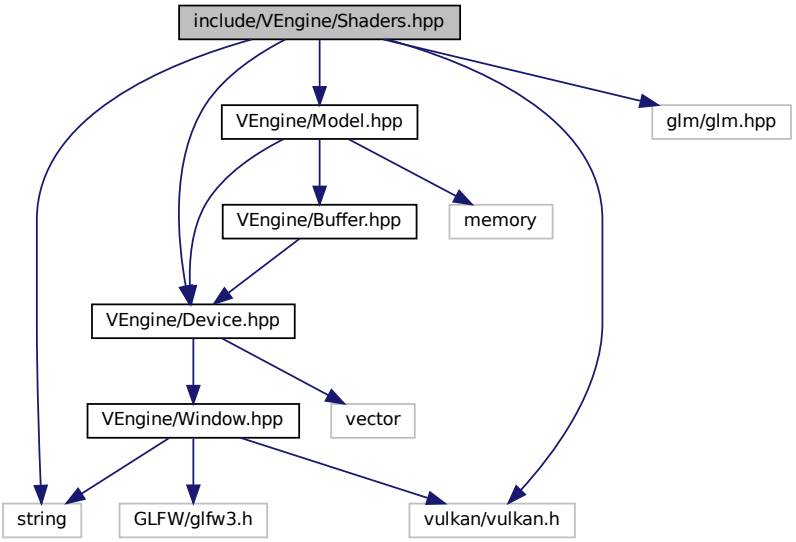
This file contains the Renderer class.

4.12 include/VEngine/Shaders.hpp File Reference

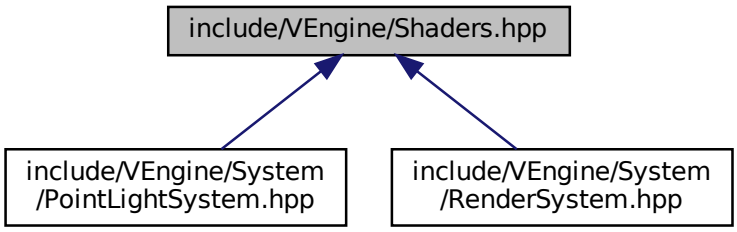
This file contains the Shader class.

```
#include <string>
#include <vulkan/vulkan.h>
#include <glm/glm.hpp>
#include "VEngine/Device.hpp"
#include "VEngine/Model.hpp"
```

Include dependency graph for Shaders.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- struct ven::PipelineConfigInfo
- class ven::Shaders

4.12.1 Detailed Description

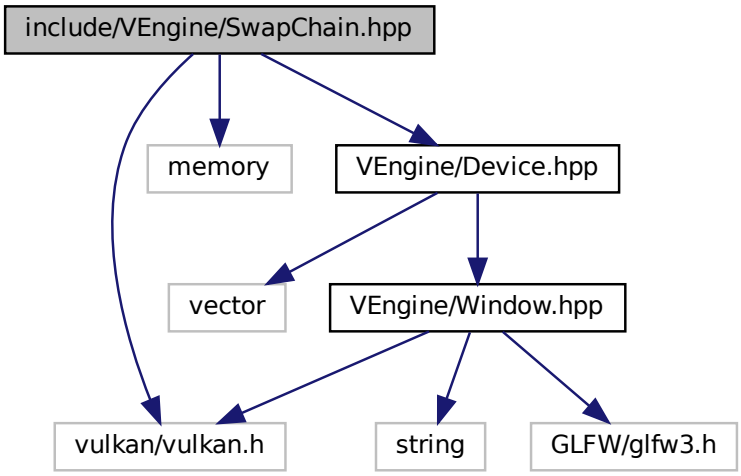
This file contains the Shader class.

4.13 include/VEngine/SwapChain.hpp File Reference

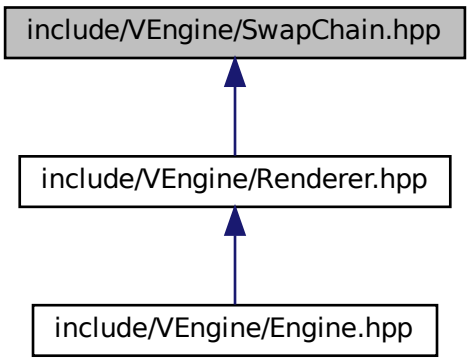
This file contains the Shader class.

```
#include <vulkan/vulkan.h>
#include <memory>
#include "VEngine/Device.hpp"
```

Include dependency graph for SwapChain.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ven::SwapChain

4.13.1 Detailed Description

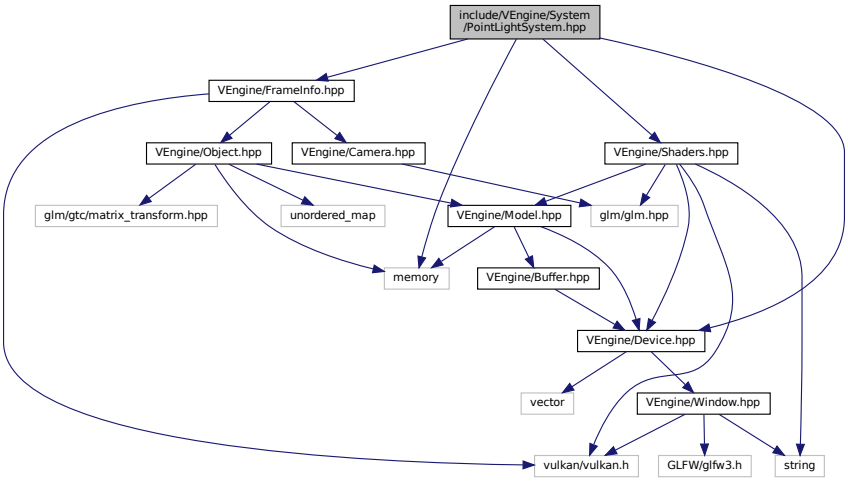
This file contains the Shader class.

4.14 include/VEngine/System/PointLightSystem.hpp File Reference

This file contains the PointLightSystem class.

```
#include <memory>
#include "VEngine/Device.hpp"
#include "VEngine/Shaders.hpp"
#include "VEngine/FrameInfo.hpp"
```

Include dependency graph for PointLightSystem.hpp:



Classes

- class ven::PointLightSystem

  Class for point light system.

4.14.1 Detailed Description

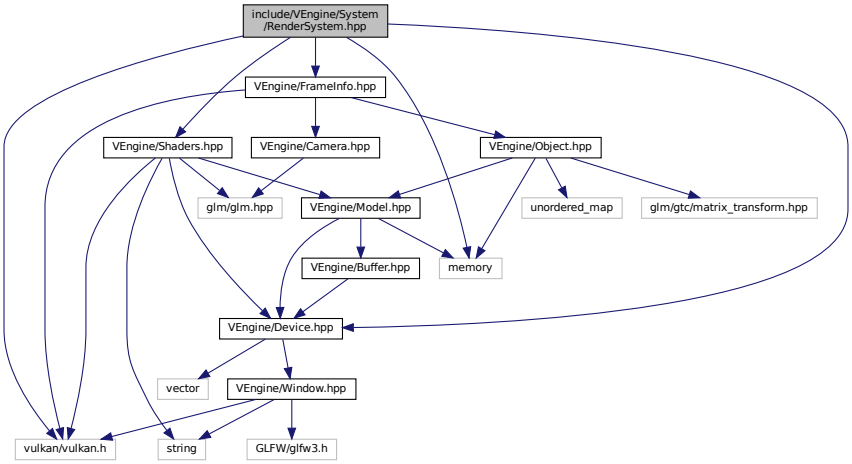
This file contains the PointLightSystem class.

4.15 include/VEngine/System/RenderSystem.hpp File Reference

This file contains the RenderSystem class.

```
#include <memory>
#include <vulkan/vulkan.h>
#include "VEngine/Device.hpp"
#include "VEngine/Shaders.hpp"
#include "VEngine/FrameInfo.hpp"
```

Include dependency graph for RenderSystem.hpp:



Classes

- struct ven::SimplePushConstantData
- class ven::RenderSystem

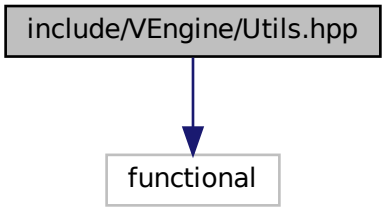
  Class for render system.

4.15.1 Detailed Description

This file contains the RenderSystem class.

4.16 include/VEngine/Utils.hpp File Reference

```
#include <functional>
```
Include dependency graph for Utils.hpp:



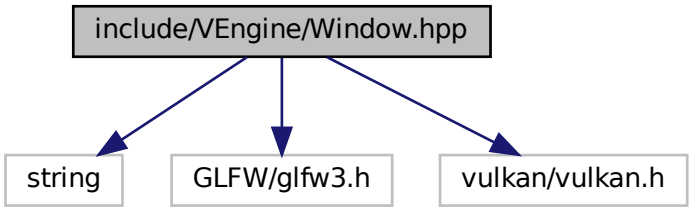
Functions

- template<typename T , typename... Rest>
  void **ven::hashCombine** (std::size_t &seed, const T &v, const Rest &... rest)

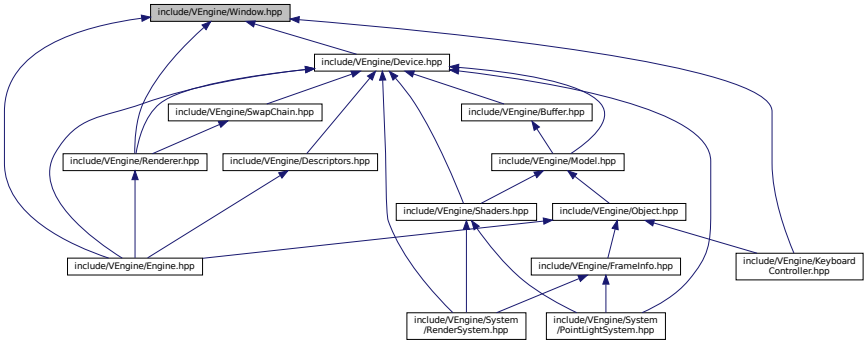
4.17 include/VEngine/Window.hpp File Reference

This file contains the Window class.

```
#include <string>
#include <GLFW/glfw3.h>
#include <vulkan/vulkan.h>
```
Include dependency graph for Window.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class ven::Window

4.17.1 Detailed Description

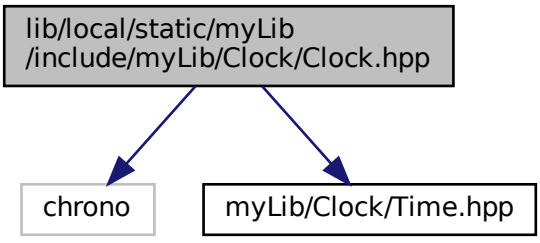
This file contains the Window class.

4.18 lib/local/static/myLib/include/myLib/Clock/Clock.hpp File Reference

Clock class for time management.

```
#include <chrono>
#include "myLib/Clock/Time.hpp"
```

Include dependency graph for Clock.hpp:



Classes

- class myLib::Clock

  Class for time management.

Typedefs

- using TimePoint = std::chrono::time_point< std::chrono::high_resolution_clock >

  TimePoint is a type alias for a time point which is a very long and complicated type in the standard library.

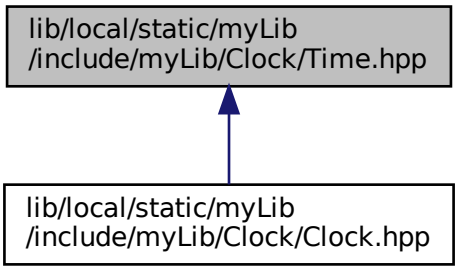
4.18.1 Detailed Description

Clock class for time management.

4.19 lib/local/static/myLib/include/myLib/Clock/Time.hpp File Reference

Class for time management.

This graph shows which files directly or indirectly include this file:



Classes

- class myLib::Time

  Class used for time management.

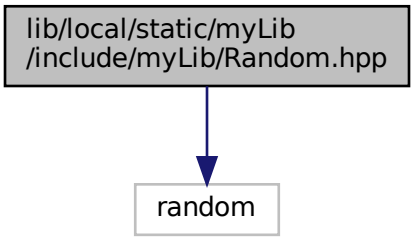
4.19.1 Detailed Description

Class for time management.

4.20 lib/local/static/myLib/include/myLib/Random.hpp File Reference

Class for random number generation.

`#include <random>`
Include dependency graph for Random.hpp:



Classes

- class myLib::Random

  Class for random number generation.

4.20.1 Detailed Description

Class for random number generation.

Index