

vengine

0.1.0

Generated by Doxygen 1.9.1

1 Class Index	1
1.1 Class List	1
2 Class Documentation	3
2.1 ven::Buffer Class Reference	3
2.1.1 Member Function Documentation	3
2.1.1.1 descriptorInfo()	3
2.1.1.2 descriptorInfoForIndex()	4
2.1.1.3 flush()	4
2.1.1.4 flushIndex()	5
2.1.1.5 invalidate()	5
2.1.1.6 invalidateIndex()	5
2.1.1.7 map()	6
2.1.1.8 unmap()	6
2.1.1.9 writeToBuffer()	6
2.1.1.10 writeToIndex()	7
2.2 ven::DescriptorPool::Builder Class Reference	7
2.3 ven::DescriptorSetLayout::Builder Class Reference	7
2.4 ven::Model::Builder Struct Reference	8
2.5 ven::Camera Class Reference	8
2.6 myLib::Clock Class Reference	8
2.7 ven::DescriptorPool Class Reference	9
2.8 ven::DescriptorSetLayout Class Reference	9
2.9 ven::DescriptorWriter Class Reference	10
2.10 ven::Device Class Reference	10
2.11 ven::Engine Class Reference	11
2.12 ven::FrameCounter Class Reference	11
2.13 ven::FrameInfo Struct Reference	11
2.14 ven::GlobalUbo Struct Reference	12
2.15 ven::KeyboardController Class Reference	12
2.16 ven::KeyboardController::KeyMappings Struct Reference	12
2.17 ven::Model Class Reference	13
2.18 ven::Object Class Reference	13
2.19 ven::PipelineConfigInfo Struct Reference	14
2.20 ven::PointLight Struct Reference	14
2.21 ven::PointLightComponent Struct Reference	15
2.22 ven::PointLightSystem Class Reference	15
2.23 ven::QueueFamilyIndices Struct Reference	15
2.24 myLib::Random Class Reference	16
2.25 ven::Renderer Class Reference	16
2.26 ven::RenderSystem Class Reference	16
2.27 ven::Shaders Class Reference	17

2.28 ven::SimplePushConstantData Struct Reference	17
2.29 ven::SwapChain Class Reference	17
2.30 ven::SwapChainSupportDetails Struct Reference	18
2.31 myLib::Time Class Reference	18
2.32 ven::Transform3DComponent Struct Reference	18
2.33 ven::Model::Vertex Struct Reference	19
2.34 ven::Window Class Reference	19
Index	21

Chapter 1

Class Index

1.1 Class List

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

ven::Buffer	3
ven::DescriptorPool::Builder	7
ven::DescriptorSetLayout::Builder	7
ven::Model::Builder	8
ven::Camera	8
myLib::Clock	8
ven::DescriptorPool	9
ven::DescriptorSetLayout	9
ven::DescriptorWriter	10
ven::Device	10
ven::Engine	11
ven::FrameCounter	11
ven::FrameInfo	11
ven::GlobalUbo	12
ven::KeyboardController	12
ven::KeyboardController::KeyMappings	12
ven::Model	13
ven::Object	13
ven::PipelineConfigInfo	14
ven::PointLight	14
ven::PointLightComponent	15
ven::PointLightSystem	15
ven::QueueFamilyIndices	15
myLib::Random	16
ven::Renderer	16
ven::RenderSystem	16
ven::Shaders	17
ven::SimplePushConstantData	17
ven::SwapChain	17
ven::SwapChainSupportDetails	18
myLib::Time	18
ven::Transform3DComponent	18
ven::Model::Vertex	19
ven::Window	19

Chapter 2

Class Documentation

2.1 ven::Buffer Class Reference

Public Member Functions

- **Buffer** ([Device](#) &device, VkDeviceSize instanceSize, uint32_t instanceCount, VkBufferUsageFlags usageFlags, 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](#) (const void *data, VkDeviceSize size=VK_WHOLE_SIZE, VkDeviceSize offset=0) const
- VkResult [flush](#) (VkDeviceSize size=VK_WHOLE_SIZE, VkDeviceSize offset=0) const
- VkDescriptorBufferInfo [descriptorInfo](#) (const VkDeviceSize size=VK_WHOLE_SIZE, const VkDeviceSize offset=0) const
- VkResult [invalidate](#) (VkDeviceSize size=VK_WHOLE_SIZE, VkDeviceSize offset=0) const
- 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

2.1.1 Member Function Documentation

2.1.1.1 descriptorInfo()

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

Create a m_buffer info descriptor

Parameters

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

Returns

VkDescriptorBufferInfo of specified offset and range

2.1.1.2 descriptorInfoForIndex()

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

Create a m_buffer info descriptor

Parameters

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

Returns

VkDescriptorBufferInfo for instance at index

2.1.1.3 flush()

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

Flush a m_memory range of the m_buffer to make it visible to the device

Note

Only required for non-coherent m_memory

Parameters

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

Returns

VkResult of the flush call

2.1.1.4 flushIndex()

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

Flush the m_memory range at index * m_alignmentSize of the m_buffer to make it visible to the device

Parameters

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

2.1.1.5 invalidate()

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

Invalidate a m_memory range of the m_buffer to make it visible to the host

Note

Only required for non-coherent m_memory

Parameters

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

Returns

VkResult of the invalidate call

2.1.1.6 invalidateIndex()

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

Invalidate a m_memory range of the m_buffer to make it visible to the host

Note

Only required for non-coherent m_memory

Parameters

<i>index</i>	Specifies the region to invalidate: index * m_alignmentSize
--------------	---

Returns

VkResult of the invalidate call

2.1.1.7 map()

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

Map a m_memory range of this m_buffer. If successful, m_mapped points to the specified m_buffer range.

Parameters

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

Returns

VkResult of the m_buffer mapping call

2.1.1.8 unmap()

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

Unmap a m_mapped m_memory range

Note

Does not return a result as vkUnmapMemory can't fail

2.1.1.9 writeToBuffer()

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

Copies the specified data to the m_mapped m_buffer. Default value writes whole m_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 m_buffer range.
<i>offset</i>	(Optional) Byte offset from beginning of m_mapped region

2.1.1.10 writeToIndex()

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

Copies "m_instanceSize" bytes of data to the m_mapped m_buffer at an offset of index * m_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

2.2 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

2.3 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

2.4 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

2.5 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

2.6 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/local/static/myLib/include/myLib/Clock/Clock.hpp

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

Friends

- class **DescriptorWriter**

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

- include/VEngine/Descriptors.hpp

2.8 ven::DescriptorSetLayout Class Reference

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**

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

- include/VEngine/Descriptors.hpp

2.9 ven::DescriptorWriter Class Reference

Public Member Functions

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

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

- include/VEngine/Descriptors.hpp

2.10 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 properties) const
- [QueueFamilyIndices](#) **findPhysicalQueueFamilies** () const
- VkFormat **findSupportedFormat** (const std::vector< VkFormat > &candidates, VkImageTiling tiling, VkFormatFeatureFlags features) const
- void **createBuffer** (VkDeviceSize size, VkBufferUsageFlags usage, VkMemoryPropertyFlags properties, 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 layerCount) 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

2.11 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

2.12 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/FrameCouter.hpp

2.13 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

2.14 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

2.15 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

2.16 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

2.17 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

2.18 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< 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`

2.19 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`

2.20 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/FramelInfo.hpp`

2.21 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

2.22 ven::PointLightSystem Class Reference

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)

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

- include/VEngine/System/PointLightSystem.hpp

2.23 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

2.24 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/local/static/myLib/include/myLib/Random.hpp

2.25 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

2.26 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** (const [FrameInfo](#) &frameInfo) const

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

- include/VEngine/System/RenderSystem.hpp

2.27 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

2.28 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

2.29 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

2.30 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

2.31 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/local/static/myLib/include/myLib/Clock/Time.hpp

2.32 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

2.33 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

2.34 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 * **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, 3](#)
- descriptorInfoForIndex
 - [ven::Buffer, 4](#)
- flush
 - [ven::Buffer, 4](#)
- flushIndex
 - [ven::Buffer, 5](#)
- invalidate
 - [ven::Buffer, 5](#)
- invalidateIndex
 - [ven::Buffer, 5](#)
- map
 - [ven::Buffer, 6](#)
- myLib::Clock, [8](#)
- myLib::Random, [16](#)
- myLib::Time, [18](#)
- unmap
 - [ven::Buffer, 6](#)
- ven::Buffer, [3](#)
 - [descriptorInfo, 3](#)
 - [descriptorInfoForIndex, 4](#)
 - [flush, 4](#)
 - [flushIndex, 5](#)
 - [invalidate, 5](#)
 - [invalidateIndex, 5](#)
 - [map, 6](#)
 - [unmap, 6](#)
 - [writeToBuffer, 6](#)
 - [writeToIndex, 7](#)
- ven::Camera, [8](#)
- ven::DescriptorPool, [9](#)
- ven::DescriptorPool::Builder, [7](#)
- ven::DescriptorSetLayout, [9](#)
- ven::DescriptorSetLayout::Builder, [7](#)
- ven::DescriptorWriter, [10](#)
- ven::Device, [10](#)
- ven::Engine, [11](#)
- ven::FrameCounter, [11](#)
- ven::FrameInfo, [11](#)
- ven::GlobalUbo, [12](#)
- ven::KeyboardController, [12](#)
- ven::KeyboardController::KeyMappings, [12](#)
- ven::Model, [13](#)
- ven::Model::Builder, [8](#)
- ven::Model::Vertex, [19](#)
- ven::Object, [13](#)
- ven::PipelineConfigInfo, [14](#)
- ven::PointLight, [14](#)
- ven::PointLightComponent, [15](#)
- ven::PointLightSystem, [15](#)
- ven::QueueFamilyIndices, [15](#)
- ven::Renderer, [16](#)
- ven::RenderSystem, [16](#)
- ven::Shaders, [17](#)
- ven::SimplePushConstantData, [17](#)
- ven::SwapChain, [17](#)
- ven::SwapChainSupportDetails, [18](#)
- ven::Transform3DComponent, [18](#)
- ven::Window, [19](#)
- writeToBuffer
 - [ven::Buffer, 6](#)
- writeToIndex
 - [ven::Buffer, 7](#)