

GraphicsDevice

- ID3D11Device * device
 - ID3D11DeviceContext * deviceContext
 - IDXGISwapChain * swapChain
 - ID3D11RenderTargetView * renderTargetView
 - ID3D11DepthStencilView * depthStencilView
 - ID3D11Texture2D * depthStencilBuffer
 - int windowWidth
 - int windowHeight
 - HWND hWnd

+ void Initialize(HWND hWnd, int width, int height)
 + void ResizeBuffersAndViews(int width, int height)
 + ID3D11Device * GetDevice() const
 + ID3D11DeviceContext * GetDeviceContext() const
 + IDXGISwapChain * GetSwapChain() const
 + ID3D11RenderTargetView * GetRenderTargetView() const
 + ID3D11DepthStencilView * GetDepthStencilView() const
 + int GetWindowWidth()
 + int GetWindowHeight()
 + HWND GetHWND()

ShaderProgram

- ID3D11VertexShader * vertexShader
 - ID3D11PixelShader * pixelShader
 - ID3D11InputLayout * inputLayout
 - ID3D11Buffer * constantBuffer

+ ShaderProgram(GraphicsDevice *graphicsDevice)
 + void LoadShader(LPCSTR VSFilename, LPCSTR PSFilename)
 + void ClearResources()
 + void UpdateBuffers(ConstantBuffer cb)
 + void Bind()
 - bool CompileHLSL(const std::string &filename, ShaderType shaderType)
 - std::vector< byte > LoadShaderFile(const std::string &filename, ShaderType shaderType)

std::shared_ptr< ShaderProgram >

+ ShaderProgram * operator->()

std::basic_string< Char >

ShaderManager::ShaderInfo

+ std::filesystem::file_time_type lastWriteTime

std::string

std::unordered_map< std::string, ShaderManager::ShaderInfo >

ShaderManager

+ ShaderManager(GraphicsDevice *device, const std::string &directory)
 + ~ShaderManager()
 + std::shared_ptr< ShaderProgram > LoadShader(const std::string &vsName, const std::string &psName)
 + void ReloadShader(const std::string &vsName, const std::string &psName)
 + void CheckForShaderUpdates()
 - std::string GenerateShaderKey(const std::string &vsName, const std::string &psName)

-graphicsDevice

+ptr

+shaderProgram

+elements

+keys

-shaderDirectory

-shaders

-device