cae

0.0.0

Generated by Doxygen 1.11.0

Chapter 1

cae

1.1 Cross-API-Engine | Rendering Engine with multiple dynamic backends

Cross-API-Engine is a rendering engine designed to support multiple graphics APIs dynamically. It allows developers to switch between different rendering backends such as OpenGL, Vulkan, DirectX at runtime. It is useful to do benchmarks during development or to support multiple platforms with different graphics APIs.

```
flowchart LR
subgraph main
    subgraph App
        A[Engine]
        A -->|.so/.dylib/.dll| B[IAudio]
        A -->|.so/.dylib/.dll| C[INetwork]
        A -->|.so/.dylib/.dll| D[IRenderer]
        A -->|.so/.dylib/.dll| E[IWindow]
    end

    subgraph Plugins
        subgraph audio impl
            F[WASAPI]
            K[ALSA]
            L[CoreAudio]
        end
        subgraph network impl
            G[Winsock2]
            O[Berkeley Sockets]
        end
        subgraph renderer impl
            H[OpenGL]
            I[Vulkan]
        end
        subgraph window impl
            J[WIN32]
            M[WayLand]
            N[Cocoa]
            subgraph Input devices
                P[IInput]
                Q[KeyBoard]
                R[Mouse]
                S[Controller]
            end
        end
    end

    B -.-> F
    B -.-> K
    B -.-> L
    C -.-> G
    C -.-> O
    D -.-> H
    D -.-> I
    E -.-> P
    E -.-> J
    E -.-> M
    E -.-> N
    P -.-> S
    P -.-> R
    P -.-> Q
end
```

1.1.1 Prerequisites

Make sure you have the following dependencies installed on your system:

- CMake 4.0.0
- C++23
- Vulkan SDK

1.1.2 External Libraries

- GLFW: For creating windows, receiving input, and managing OpenGL and Vulkan contexts.
- Google Test: A testing framework for C++.
- ImGui: Immediate Mode Graphical User Interface for real-time debugging and tool development.
- stb: A set of single-file public domain libraries for graphics, image loading, and more.

1.1.3 Contributing

Want to contribute? See CONTRIBUTING.md.

1.1.4 License

This project is licensed under the MIT License - see the LICENSE file for details.

Chapter 2

Commit Norms

Commit Type	Description
build	Changes that affect the build system or external dependencies (npm, make, etc.)
ci	Changes related to integration files and scripts or configuration (Travis, Ansible, BrowserStack, etc.)
feat	Addition of a new feature
fix	Bug fix
perf	Performance improvements
refactor	Modification that neither adds a new feature nor improves performance
style	Change that does not affect functionality or semantics (indentation, formatting, adding space, renaming a variable, etc.)
docs	Writing or updating documentation
test	Addition or modification of tests

Chapter 3

Namespace Index

3.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 4

Hierarchical Index

4.1 Class Hierarchy

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

Chapter 5

Class Index

5.1 Class List

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

Chapter 6

File Index

6.1 File List

Here is a list of all files with brief descriptions:

Chapter 7

Namespace Documentation

7.1 cae Namespace Reference

Namespaces

- namespace Audio
- namespace Network
- namespace User
- namespace Window

Classes

- class Application

  Main class.
- struct ArgsConfig
- class ArgsHandler

  Class to handle command line arguments.
- struct EnvConfig
- class OPGL

  Class for the OpenGL plugin.
- class VULKN

  Class for the Vulkan plugin.

7.2 cae::Audio Namespace Reference

Variables

- constexpr auto VOLUME = 50.F
- constexpr auto MUTED = false

7.2.1 Variable Documentation

7.2.1.1 MUTED

auto cae::Audio::MUTED = false [inline], [constexpr]

Definition at line 14 of file Common.hpp.

7.2.1.2 VOLUME

auto cae::Audio::VOLUME = 50.F [inline], [constexpr]

Definition at line 13 of file Common.hpp.

7.3 cae::Network Namespace Reference

Variables

- constexpr auto HOST = "127.0.0.1"
- constexpr auto PORT = 4242

7.3.1 Variable Documentation

7.3.1.1 HOST

auto cae::Network::HOST = "127.0.0.1" [inline], [constexpr]

Definition at line 18 of file Common.hpp.

7.3.1.2 PORT

auto cae::Network::PORT = 4242 [inline], [constexpr]

Definition at line 19 of file Common.hpp.

7.4 cae::User Namespace Reference

Variables

- constexpr auto NAME = "User"

7.4.1 Variable Documentation

7.4.1.1 NAME

auto cae::User::NAME = "User" [inline], [constexpr]

Definition at line 23 of file Common.hpp.

7.5 cae::Window Namespace Reference

Variables

- constexpr auto HEIGHT = 1920
- constexpr auto WIDTH = 1080
- constexpr auto NAME = ”CAE - Cross API Engine”
- constexpr auto FULLSCREEN = false
- constexpr auto VSYNC = false
- constexpr auto MAX_FPS = 90

7.5.1 Variable Documentation

7.5.1.1 FULLSCREEN

auto cae::Window::FULLSCREEN = false [inline], [constexpr]

Definition at line 30 of file Common.hpp.

7.5.1.2 HEIGHT

auto cae::Window::HEIGHT = 1920 [inline], [constexpr]

Definition at line 27 of file Common.hpp.

7.5.1.3 MAX_FPS

auto cae::Window::MAX_FPS = 90 [inline], [constexpr]

Definition at line 32 of file Common.hpp.

7.5.1.4 NAME

auto cae::Window::NAME = ”CAE - Cross API Engine” [inline], [constexpr]

Definition at line 29 of file Common.hpp.

7.5.1.5 VSYNC

auto cae::Window::VSYNC = false [inline], [constexpr]

Definition at line 31 of file Common.hpp.

7.5.1.6 WIDTH

auto cae::Window::WIDTH = 1080 [inline], [constexpr]

Definition at line 28 of file Common.hpp.

Chapter 8

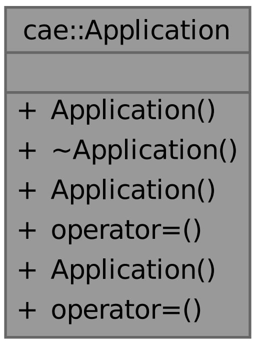
Class Documentation

8.1 cae::Application Class Reference

Main class.

#include <Application.hpp>

Collaboration diagram for cae::Application:



Public Member Functions

- Application ()
- ∼Application ()=default
- Application (const Application &)=delete
- Application & operator= (const Application &)=delete
- Application (Application &&)=delete
- Application & operator= (Application &&)=delete

8.1.1 Detailed Description

Main class.

Definition at line 17 of file Application.hpp.

8.1.2 Constructor & Destructor Documentation

8.1.2.1 Application() [1/3]

cae::Application::Application ()

Definition at line 10 of file application.cpp.

8.1.2.2 ∼Application()

cae::Application::∼Application () [default]

8.1.2.3 Application() [2/3]

```
cae::Application::Application (
                    const Application & ) [delete]
```

8.1.2.4 Application() [3/3]

```
cae::Application::Application (
                    Application && ) [delete]
```

8.1.3 Member Function Documentation

8.1.3.1 operator=() [1/2]

```
Application & cae::Application::operator= (
                    Application && ) [delete]
```

8.1.3.2 operator=() [2/2]

```
Application & cae::Application::operator= (
                    const Application & ) [delete]
```

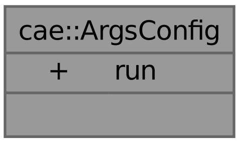
The documentation for this class was generated from the following files:

- /home/masina/Projects/Cross-API-Engine/include/CAE/Application.hpp
- /home/masina/Projects/Cross-API-Engine/src/application.cpp

8.2 cae::ArgsConfig Struct Reference

#include <ArgsHandler.hpp>

Collaboration diagram for cae::ArgsConfig:



Public Attributes

- bool run {false}

8.2.1 Detailed Description

Definition at line 12 of file ArgsHandler.hpp.

8.2.2 Member Data Documentation

8.2.2.1 run

bool cae::ArgsConfig::run {false}

Definition at line 14 of file ArgsHandler.hpp.

Referenced by cae::ArgsHandler::ParseArgs().

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

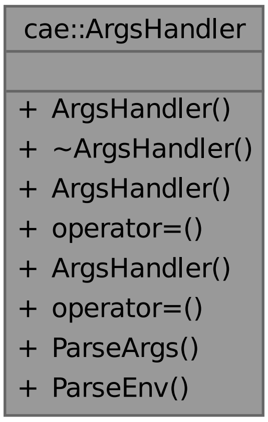
- /home/masina/Projects/Cross-API-Engine/include/CAE/ArgsHandler.hpp

8.3 cae::ArgsHandler Class Reference

Class to handle command line arguments.

#include <ArgsHandler.hpp>

Collaboration diagram for cae::ArgsHandler:



Public Member Functions

- ArgsHandler ()=default
- ~ArgsHandler ()=default
- ArgsHandler (const ArgsHandler &)=delete
- ArgsHandler & operator= (const ArgsHandler &)=delete
- ArgsHandler (ArgsHandler &&)=delete
- ArgsHandler & operator= (ArgsHandler &&)=delete

Static Public Member Functions

- static ArgsConfig ParseArgs (int argc, const char ∗const ∗argv)
- static EnvConfig ParseEnv (const char ∗const ∗envp)

8.3.1 Detailed Description

Class to handle command line arguments.

Definition at line 25 of file ArgsHandler.hpp.

8.3.2 Constructor & Destructor Documentation

8.3.2.1 ArgsHandler() [1/3]

cae::ArgsHandler::ArgsHandler () [default]

8.3.2.2 ~ArgsHandler()

cae::ArgsHandler::~ArgsHandler () [default]

8.3.2.3 ArgsHandler() [2/3]

cae::ArgsHandler::ArgsHandler (
    const ArgsHandler &) [delete]

8.3.2.4 ArgsHandler() [3/3]

cae::ArgsHandler::ArgsHandler (
    ArgsHandler &&) [delete]

8.3.3 Member Function Documentation

8.3.3.1 operator=() [1/2]

ArgsHandler & cae::ArgsHandler::operator= (
    ArgsHandler &&) [delete]

8.3.3.2 operator=() [2/2]

ArgsHandler & cae::ArgsHandler::operator= (
    const ArgsHandler &) [delete]

8.3.3.3 ParseArgs()

cae::ArgsConfig cae::ArgsHandler::ParseArgs (
    int argc,
    const char ∗const ∗ argv) [static]

Definition at line 30 of file argsHandler.cpp.

References ARGS_MAP, and cae::ArgsConfig::run.

Referenced by main().

Here is the caller graph for this function:

main
cae::ArgsHandler::ParseArgs

8.3.3.4 ParseEnv()

cae::EnvConfig cae::ArgsHandler::ParseEnv (
const char ∗const ∗ envp) [static]

Definition at line 45 of file argsHandler.cpp.

Referenced by main().

Here is the caller graph for this function:

main
cae::ArgsHandler::ParseEnv

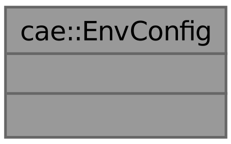
The documentation for this class was generated from the following files:

- /home/masina/Projects/Cross-API-Engine/include/CAE/ArgsHandler.hpp
- /home/masina/Projects/Cross-API-Engine/src/argsHandler.cpp

8.4 cae::EnvConfig Struct Reference

#include <ArgsHandler.hpp>

Collaboration diagram for cae::EnvConfig:



8.4.1 Detailed Description

Definition at line 16 of file ArgsHandler.hpp.

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

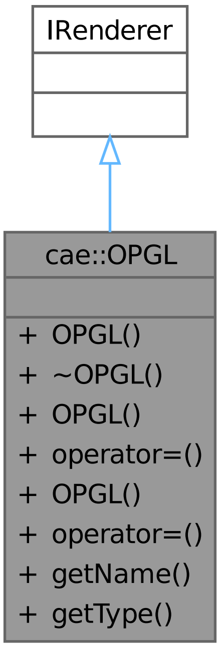
- /home/masina/Projects/Cross-API-Engine/include/CAE/ArgsHandler.hpp

8.5 cae::OPGL Class Reference

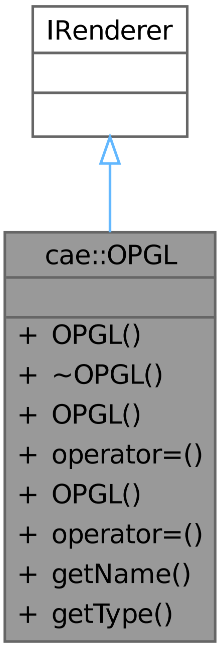
Class for the OpenGL plugin.

#include <OPGL.hpp>

Inheritance diagram for cae::OPGL:



Collaboration diagram for cae::OPGL:



Public Member Functions

- OPGL ()=default
- ~OPGL () override=default
- OPGL (const OPGL &)=delete
- OPGL & operator= (const OPGL &)=delete
- OPGL (OPGL &&)=delete
- OPGL & operator= (OPGL &&)=delete
- std::string getName () const override
- utl::PluginType getType () const override

8.5.1 Detailed Description

Class for the OpenGL plugin.

Definition at line 19 of file OPGL.hpp.

8.5.2 Constructor & Destructor Documentation

8.5.2.1 OPGL() [1/3]

cae::OPGL::OPGL () [default]

8.5.2.2 ~OPGL()

cae::OPGL::~OPGL () [override], [default]

8.5.2.3 OPGL() [2/3]

```
cae::OPGL::OPGL (
            const OPGL & ) [delete]
```

8.5.2.4 OPGL() [3/3]

```
cae::OPGL::OPGL (
            OPGL && ) [delete]
```

8.5.3 Member Function Documentation

8.5.3.1 getName()

std::string cae::OPGL::getName () const [inline], [nodiscard], [override]

Definition at line 31 of file OPGL.hpp.

8.5.3.2 getType()

utl::PluginType cae::OPGL::getType () const [inline], [nodiscard], [override]

Definition at line 32 of file OPGL.hpp.

8.5.3.3 operator=() [1/2]

```
OPGL & cae::OPGL::operator= (
            const OPGL & ) [delete]
```

8.5.3.4 operator=() [2/2]

```
OPGL & cae::OPGL::operator= (
            OPGL && ) [delete]
```

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

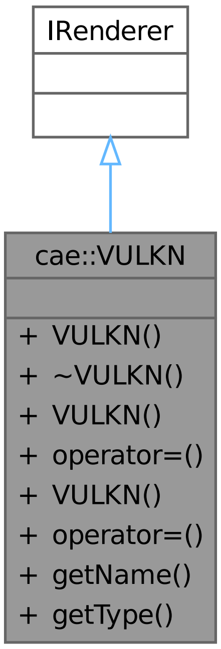
- /home/masina/Projects/Cross-API-Engine/plugins/Renderer/OpenGL/include/OPGL/OPGL.hpp

8.6 cae::VULKN Class Reference

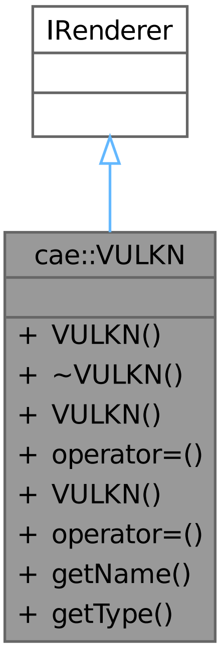
Class for the Vulkan plugin.

#include <VULKN.hpp>

Inheritance diagram for cae::VULKN:



Collaboration diagram for cae::VULKN:



Public Member Functions

- VULKN ()=default
- ~VULKN () override=default
- VULKN (const VULKN &)=delete
- VULKN & operator= (const VULKN &)=delete
- VULKN (VULKN &&)=delete
- VULKN & operator= (VULKN &&)=delete
- std::string getName () const override
- utl::PluginType getType () const override

8.6.1 Detailed Description

Class for the Vulkan plugin.

Definition at line 19 of file VULKN.hpp.

8.6.2 Constructor & Destructor Documentation

8.6.2.1 VULKN() [1/3]

cae::VULKN::VULKN () [default]

8.6.2.2 ∼VULKN()

cae::VULKN::∼VULKN () [override], [default]

8.6.2.3 VULKN() [2/3]

cae::VULKN::VULKN (
    const VULKN &) [delete]

8.6.2.4 VULKN() [3/3]

cae::VULKN::VULKN (
    VULKN &&) [delete]

8.6.3 Member Function Documentation

8.6.3.1 getName()

std::string cae::VULKN::getName () const [inline], [nodiscard], [override]

Definition at line 31 of file VULKN.hpp.

8.6.3.2 getType()

utl::PluginType cae::VULKN::getType () const [inline], [nodiscard], [override]

Definition at line 32 of file VULKN.hpp.

8.6.3.3 operator=() [1/2]

VULKN & cae::VULKN::operator= (
    const VULKN &) [delete]

8.6.3.4 operator=() [2/2]

VULKN & cae::VULKN::operator= (
    VULKN &&) [delete]

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

- /home/masina/Projects/Cross-API-Engine/plugins/Renderer/Vulkan/include/VULKN/VULKN.hpp

Chapter 9

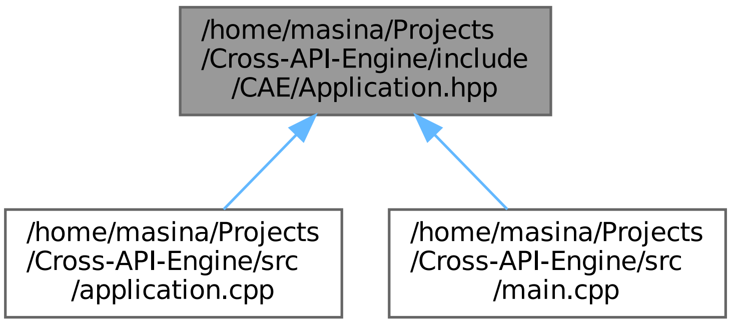
File Documentation

9.1 /home/masina/Projects/Cross-API-Engine/CONTRIBUTING.md File Reference

9.2 /home/masina/Projects/Cross-API-Engine/include/CAE/↩ Application.hpp File Reference

This file contains the Application class declaration.

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



Classes

- class cae::Application
  Main class.

Namespaces

- namespace cae

9.2.1 Detailed Description

This file contains the Application class declaration.

Definition in file Application.hpp.

9.3 Application.hpp

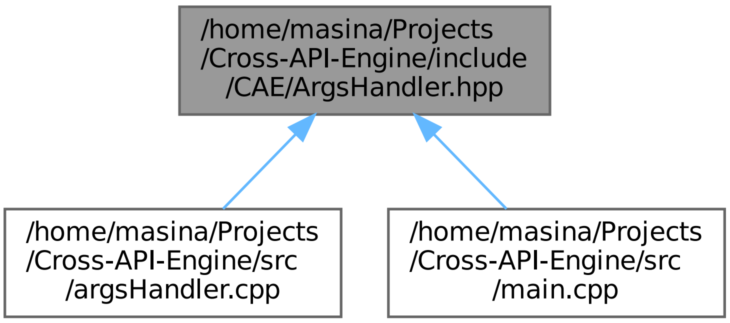
Go to the documentation of this file.

```
///
/// @file Application.hpp
/// @brief This file contains the Application class declaration
/// @namespace cae
///

#pragma once

namespace cae
{

    ///
    /// @class Application
    /// @brief Main class
    /// @namespace cae
    ///
    class Application
    {

        public:
            Application();
            ~Application() = default;

            Application(const Application &) = delete;
            Application &operator=(const Application &) = delete;
            Application(Application &&) = delete;
            Application &operator=(Application &&) = delete;

    }; // class Application

} // namespace cae
```

9.4 /home/masina/Projects/Cross-API-Engine/include/CAE/Args↩ Handler.hpp File Reference

This file contains the ArgsHandler class declaration.

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



Classes

- struct cae::ArgsConfig
- struct cae::EnvConfig
- class cae::ArgsHandler

  Class to handle command line arguments.

Namespaces

- namespace cae

9.4.1 Detailed Description

This file contains the ArgsHandler class declaration.

Definition in file ArgsHandler.hpp.

9.5 ArgsHandler.hpp

Go to the documentation of this file.

```
///
/// @file ArgsHandler.hpp
/// @brief This file contains the ArgsHandler class declaration
/// @namespace cae
///

#pragma once

namespace cae
{

    struct ArgsConfig
    {
            bool run{false};
    };
    struct EnvConfig
    {
    };

    ///
    /// @class ArgsHandler
    /// @brief Class to handle command line arguments
    /// @namespace cae
    ///
    class ArgsHandler
    {

        public:
            ArgsHandler() = default;
            ~ArgsHandler() = default;

            ArgsHandler(const ArgsHandler &) = delete;
            ArgsHandler &operator=(const ArgsHandler &) = delete;
            ArgsHandler(ArgsHandler &&) = delete;
            ArgsHandler &operator=(ArgsHandler &&) = delete;

            static ArgsConfig ParseArgs(int argc, const char *const *argv);
            static EnvConfig ParseEnv(const char *const *envp);

        private:
    }; // class ArgsHandler

} // namespace cae
```

9.6 /home/masina/Projects/Cross-API-Engine/include/CAE/Common.hpp File Reference

This file contains.

Namespaces

- namespace cae
- namespace cae::Audio
- namespace cae::Network
- namespace cae::User
- namespace cae::Window

Variables

- constexpr auto cae::Audio::VOLUME = 50.F
- constexpr auto cae::Audio::MUTED = false
- constexpr auto cae::Network::HOST = "127.0.0.1"
- constexpr auto cae::Network::PORT = 4242
- constexpr auto cae::User::NAME = "User"
- constexpr auto cae::Window::HEIGHT = 1920
- constexpr auto cae::Window::WIDTH = 1080
- constexpr auto cae::Window::NAME = "CAE - Cross API Engine"
- constexpr auto cae::Window::FULLSCREEN = false
- constexpr auto cae::Window::VSYNC = false
- constexpr auto cae::Window::MAX_FPS = 90

9.6.1 Detailed Description

This file contains.

Definition in file Common.hpp.

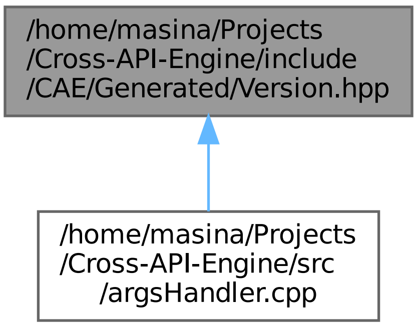
9.7 Common.hpp

Go to the documentation of this file.

```
///
/// @file Common.hpp
/// @brief This file contains
/// @namespace cae
///

#pragma once

namespace cae
{
    namespace Audio
    {
        inline constexpr auto VOLUME = 50.F;
        inline constexpr auto MUTED = false;
    } // namespace Audio
    namespace Network
    {
        inline constexpr auto HOST = "127.0.0.1";
        inline constexpr auto PORT = 4242;
    } // namespace Network
    namespace User
    {
        inline constexpr auto NAME = "User";
    }
    namespace Window
    {
        inline constexpr auto HEIGHT = 1920;
        inline constexpr auto WIDTH = 1080;
        inline constexpr auto NAME = "CAE - Cross API Engine";
        inline constexpr auto FULLSCREEN = false;
        inline constexpr auto VSYNC = false;
        inline constexpr auto MAX_FPS = 90;
    } // namespace Window
} // namespace cae
```

9.8 /home/masina/Projects/Cross-API-Engine/include/CAE/↩ Generated/Version.hpp File Reference

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



Macros

- #define PROJECT_NAME "cae"
- #define PROJECT_VERSION "0.0.0"
- #define PROJECT_VERSION_MAJOR "0"
- #define PROJECT_VERSION_MINOR "0"
- #define PROJECT_VERSION_PATCH "0"
- #define GIT_COMMIT_HASH "0d462a3"
- #define GIT_TAG "0d462a3"
- #define BUILD_TYPE "Release"

9.8.1 Macro Definition Documentation

9.8.1.1 BUILD_TYPE

```
#define BUILD_TYPE "Release"
```

Definition at line 15 of file Version.hpp.

9.8.1.2 GIT_COMMIT_HASH

#define GIT_COMMIT_HASH "0d462a3"

Definition at line 13 of file Version.hpp.

9.8.1.3 GIT_TAG

#define GIT_TAG "0d462a3"

Definition at line 14 of file Version.hpp.

9.8.1.4 PROJECT_NAME

#define PROJECT_NAME "cae"

Definition at line 7 of file Version.hpp.

9.8.1.5 PROJECT_VERSION

#define PROJECT_VERSION "0.0.0"

Definition at line 8 of file Version.hpp.

9.8.1.6 PROJECT_VERSION_MAJOR

#define PROJECT_VERSION_MAJOR "0"

Definition at line 9 of file Version.hpp.

9.8.1.7 PROJECT_VERSION_MINOR

#define PROJECT_VERSION_MINOR "0"

Definition at line 10 of file Version.hpp.

9.8.1.8 PROJECT_VERSION_PATCH

#define PROJECT_VERSION_PATCH "0"

Definition at line 11 of file Version.hpp.

9.9 Version.hpp

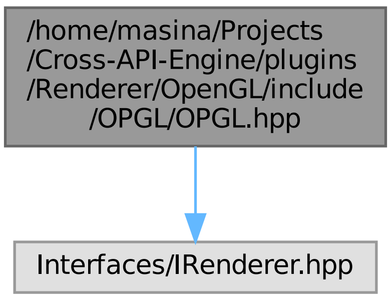
Go to the documentation of this file.

```
#pragma once

// ==========================================================================
// DO NOT EDIT THIS FILE MANUALLY. IT IS GENERATED BY CMAKE DURING THE BUILD PROCESS.
// ==========================================================================

#define PROJECT_NAME "cae"
#define PROJECT_VERSION "0.0.0"
#define PROJECT_VERSION_MAJOR "0"
#define PROJECT_VERSION_MINOR "0"
#define PROJECT_VERSION_PATCH "0"

#define GIT_COMMIT_HASH "0d462a3"
#define GIT_TAG "0d462a3"
#define BUILD_TYPE "Release"
```

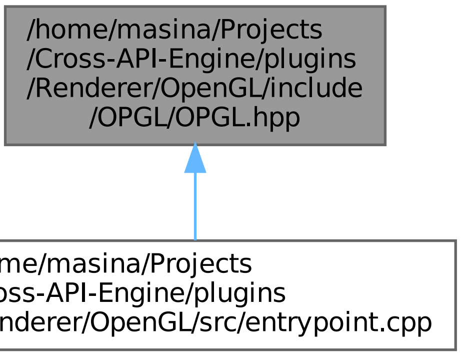
9.10 /home/masina/Projects/Cross-API-Engine/plugins/Renderer/↩ OpenGL/include/OPGL/OPGL.hpp File Reference

This file contains the OPGL class declaration.

#include "Interfaces/IRenderer.hpp"
Include dependency graph for OPGL.hpp:



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


/home/masina/Projects /Cross-API-Engine/plugins /Renderer/OpenGL/include /OPGL/OPGL.hpp
/home/masina/Projects /Cross-API-Engine/plugins /Renderer/OpenGL/src/entrypoint.cpp

Classes

- class cae::OPGL

  Class for the OpenGL plugin.

Namespaces

- namespace cae

9.10.1 Detailed Description

This file contains the OPGL class declaration.

Definition in file OPGL.hpp.

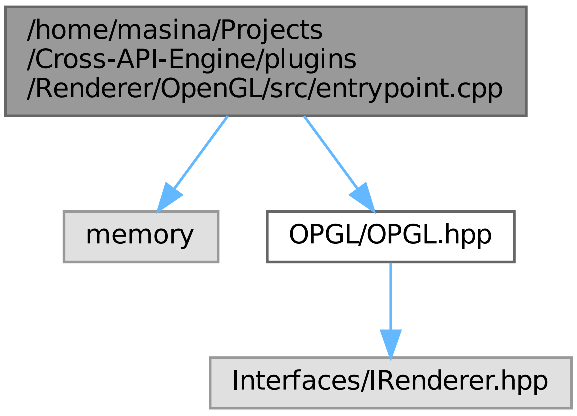
9.11 OPGL.hpp

Go to the documentation of this file.

```
///
/// @file OPGL.hpp
/// @brief This file contains the OPGL class declaration
/// @namespace cae
///

#pragma once

#include "Interfaces/IRenderer.hpp"

namespace cae
{

    ///
    /// @class OPGL
    /// @brief Class for the OpenGL plugin
    /// @namespace cae
    ///
    class OPGL final : public IRenderer
    {

        public:
            OPGL() = default;
            ~OPGL() override = default;

            OPGL(const OPGL &) = delete;
            OPGL &operator=(const OPGL &) = delete;
            OPGL(OPGL &&) = delete;
            OPGL &operator=(OPGL &&) = delete;

            [[nodiscard]] std::string getName() const override { return "OpenGL"; }
            [[nodiscard]] utl::PluginType getType() const override { return utl::PluginType::RENDERER; }

    }; // class OPGL

} // namespace cae
```

9.12 /home/masina/Projects/Cross-API-Engine/plugins/Renderer/OpenGL/src/entrypoint.cpp File Reference

#include <memory>
#include "OPGL/OPGL.hpp"
Include dependency graph for entrypoint.cpp:



Functions

- cae::IRenderer * entryPoint ()

9.12.1 Function Documentation

9.12.1.1 entryPoint()

cae::IRenderer * entryPoint ()

Definition at line 7 of file entrypoint.cpp.

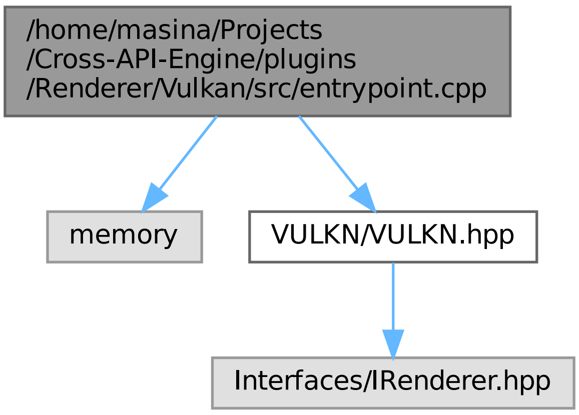
9.13 entrypoint.cpp

Go to the documentation of this file.

```
#include <memory>

#include "OPGL/OPGL.hpp"

extern "C"
{
    cae::IRenderer *entryPoint() { return std::make_unique<cae::OPGL>().release(); }
}
```

9.14 /home/masina/Projects/Cross-API-Engine/plugins/Renderer/↩ Vulkan/src/entrypoint.cpp File Reference

#include <memory>
#include "VULKN/VULKN.hpp"
Include dependency graph for entrypoint.cpp:



Functions

- cae::IRenderer ∗ entryPoint ()

9.14.1 Function Documentation

9.14.1.1 entryPoint()

cae::IRenderer ∗ entryPoint ()

Definition at line 7 of file entrypoint.cpp.

9.15 entrypoint.cpp

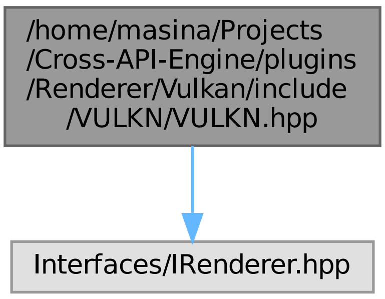
Go to the documentation of this file.

```
#include <memory>

#include "VULKN/VULKN.hpp"

extern "C"
{
    cae::IRenderer *entryPoint() { return std::make_unique<cae::VULKN>().release(); }
}
```

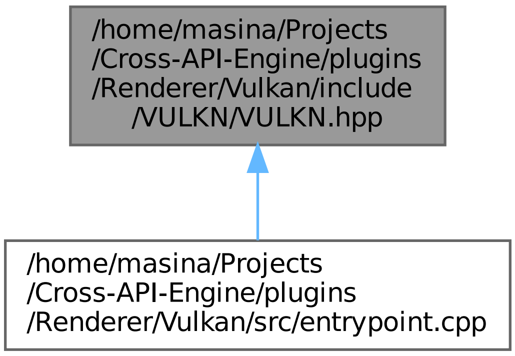
9.16 /home/masina/Projects/Cross-API-Engine/plugins/Renderer/↩ Vulkan/include/VULKN/VULKN.hpp File Reference

This file contains the VULKN class declaration.

#include "Interfaces/IRenderer.hpp"
Include dependency graph for VULKN.hpp:


/home/masina/Projects /Cross-API-Engine/plugins /Renderer/Vulkan/include /VULKN/VULKN.hpp
Interfaces/IRenderer.hpp

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


/home/masina/Projects /Cross-API-Engine/plugins /Renderer/Vulkan/include /VULKN/VULKN.hpp
/home/masina/Projects /Cross-API-Engine/plugins /Renderer/Vulkan/src/entrypoint.cpp

Classes

- class cae::VULKN

  Class for the Vulkan plugin.

Namespaces

- namespace cae

9.16.1 Detailed Description

This file contains the VULKN class declaration.

Definition in file VULKN.hpp.

9.17 VULKN.hpp

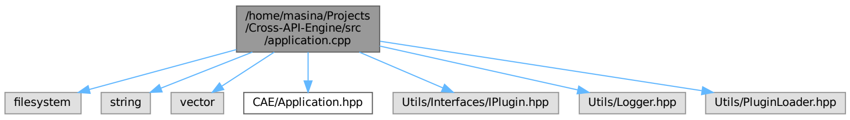
Go to the documentation of this file.

```
///
/// @file VULKN.hpp
/// @brief This file contains the VULKN class declaration
/// @namespace cae
///

#pragma once

#include "Interfaces/IRenderer.hpp"

namespace cae
{

    ///
    /// @class VULKN
    /// @brief Class for the Vulkan plugin
    /// @namespace cae
    ///
    class VULKN final : public IRenderer
    {

        public:
            VULKN() = default;
            ~VULKN() override = default;

            VULKN(const VULKN &) = delete;
            VULKN &operator=(const VULKN &) = delete;
            VULKN(VULKN &&) = delete;
            VULKN &operator=(VULKN &&) = delete;

            [[nodiscard]] std::string getName() const override { return "Vulkan"; }
            [[nodiscard]] utl::PluginType getType() const override { return utl::PluginType::RENDERER; }

    }; // class VULKN

} // namespace cae
```

9.18 /home/masina/Projects/Cross-API-Engine/README.md File Reference

9.19 /home/masina/Projects/Cross-API-Engine/src/application.cpp File Reference

#include <filesystem>
#include <string>
#include <vector>
#include "CAE/Application.hpp"
#include "Utils/Interfaces/IPlugin.hpp"
#include "Utils/Logger.hpp"
#include "Utils/PluginLoader.hpp"
Include dependency graph for application.cpp:


/home/masina/Projects/Cross-API-Engine/src/application.cpp
filesystem
string
vector
CAE/Application.hpp
Utils/Interfaces/IPlugin.hpp
Utils/Logger.hpp
Utils/PluginLoader.hpp

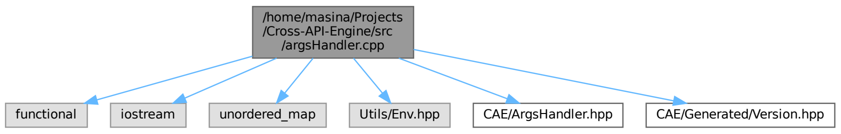
9.20 application.cpp

Go to the documentation of this file.

```
#include <filesystem>
#include <string>
#include <vector>

#include "CAE/Application.hpp"
#include "Utils/Interfaces/IPlugin.hpp"
#include "Utils/Logger.hpp"
#include "Utils/PluginLoader.hpp"

cae::Application::Application()
{
    try
    {
        const std::filesystem::path pluginDir{PLUGINS_DIR};
        utl::PluginLoader pluginManager;
        std::vector<std::string> loadedPlugins;
        for (const auto &entry : std::filesystem::directory_iterator(pluginDir))
        {
            if (!entry.is_regular_file() || entry.path().extension() != PLUGINS_EXTENSION)
            {
                continue;
            }
            if (const std::string pluginPath = entry.path().string();
                pluginManager.loadPlugin<utl::IPlugin>(pluginPath) != nullptr)
            {
                loadedPlugins.push_back(entry.path().filename().string());
            }
            else
            {
                utl::Logger::log("Failed to load plugin: " + pluginPath, utl::LogLevel::WARNING);
            }
        }
        if (loadedPlugins.empty())
        {
            utl::Logger::log("No plugins loaded from directory: " + pluginDir.string(), utl::LogLevel::WARNING);
        }
    }
    catch (const std::exception &e)
    {
        std::cerr << "Error: " << e.what() << '\n';
    }
}
```

9.21 /home/masina/Projects/Cross-API-Engine/src/argsHandler.cpp File Reference

#include <functional>
#include <iostream>
#include <unordered_map>
#include "Utils/Env.hpp"
#include "CAE/ArgsHandler.hpp"
#include "CAE/Generated/Version.hpp"
Include dependency graph for argsHandler.cpp:



Macros

- #define APP_EXTENSION ""

Variables

- static constexpr std::string_view HELP_MSG
- static constexpr std::string_view VERSION_MSG
- static const std::unordered_map< std::string, std::function< void()> > ARGS_MAP

9.21.1 Macro Definition Documentation

9.21.1.1 APP_EXTENSION

#define APP_EXTENSION ""

Definition at line 8 of file argsHandler.cpp.

9.21.2 Variable Documentation

9.21.2.1 ARGS_MAP

const std::unordered_map<std::string, std::function<void()> > ARGS_MAP [static]

Initial value:
```
= {
    {"-h", []() { std::cout « HELP_MSG; }},
    {"--help", []() { std::cout « HELP_MSG; }},
    {"-v", []() { std::cout « VERSION_MSG; }},
    {"--version", []() { std::cout « VERSION_MSG; }}}
```

Definition at line 24 of file argsHandler.cpp.

Referenced by cae::ArgsHandler::ParseArgs().

9.21.2.2 HELP_MSG

std::string_view HELP_MSG [static], [constexpr]

Initial value:
```
= "Usage: " PROJECT_NAME APP_EXTENSION " [options]\n\n"
                                    "Options:\n"
                                    "  -h, --help       Show this help message\n"
                                    "  -v, --version    Show version information\n"
```

Definition at line 16 of file argsHandler.cpp.

9.21.2.3 VERSION_MSG

std::string_view VERSION_MSG [static], [constexpr]

Initial value:
```
= PROJECT_NAME
   " v" PROJECT_VERSION " " BUILD_TYPE " (" GIT_TAG ", commit " GIT_COMMIT_HASH ") " __DATE__ " "
     __TIME__ "\n"
```

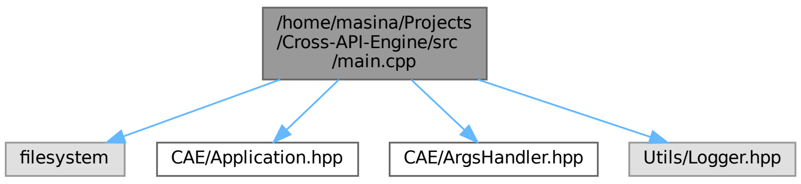
Definition at line 21 of file argsHandler.cpp.

9.22 argsHandler.cpp

Go to the documentation of this file.
```
#include <functional>
#include <iostream>
#include <unordered_map>

#ifdef _WIN32
#define APP_EXTENSION ".exe"
#else
#define APP_EXTENSION ""
#endif

#include "Utils/Env.hpp"

#include "CAE/ArgsHandler.hpp"
#include "CAE/Generated/Version.hpp"

static constexpr std::string_view HELP_MSG = "Usage: " PROJECT_NAME APP_EXTENSION " [options]\n\n"
                                             "Options:\n"
                                             "  -h, --help       Show this help message\n"
                                             "  -v, --version    Show version information\n";

static constexpr std::string_view VERSION_MSG = PROJECT_NAME
    " v" PROJECT_VERSION " " BUILD_TYPE " (" GIT_TAG ", commit " GIT_COMMIT_HASH ") " __DATE__ "
  " __TIME__ "\n";

static const std::unordered_map<std::string, std::function<void()>> ARGS_MAP = {
    {"-h", []() { std::cout << HELP_MSG; }},
    {"--help", []() { std::cout << HELP_MSG; }},
    {"-v", []() { std::cout << VERSION_MSG; }},
    {"--version", []() { std::cout << VERSION_MSG; }}};

cae::ArgsConfig cae::ArgsHandler::ParseArgs(const int argc, const char *const *argv)
{
    if (argc <= 1)
    {
        return {.run = true};
    }
    const std::string arg1{argv[1]};
    if (const auto it = ARGS_MAP.find(arg1); it != ARGS_MAP.end())
    {
        it->second();
        return {.run = false};
    }
    throw std::runtime_error("Unknown argument: " + arg1 + ". Use -h or --help to see available options.");
}

cae::EnvConfig cae::ArgsHandler::ParseEnv(const char *const *envp)
{
    for (const auto &[fst, snd] : utl::getEnvMap(envp))
    {
        std::cout << "var:" << fst << ":" << snd << '\n';
    }
    return {};
}
```

9.23 /home/masina/Projects/Cross-API-Engine/src/main.cpp File Reference

#include <filesystem>
#include "CAE/Application.hpp"
#include "CAE/ArgsHandler.hpp"
#include "Utils/Logger.hpp"
Include dependency graph for main.cpp:



Functions

- int main (const int argc, const char ∗const ∗argv, const char ∗const ∗envp)

9.23.1 Function Documentation

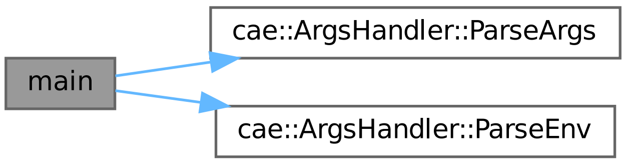
9.23.1.1 main()

```
int main (
        const int argc,
        const char ∗const ∗ argv,
        const char ∗const ∗ envp)
```

Definition at line 7 of file main.cpp.

References cae::ArgsHandler::ParseArgs(), and cae::ArgsHandler::ParseEnv().

Here is the call graph for this function:



9.24 main.cpp

Go to the documentation of this file.

```
#include <filesystem>

#include "CAE/Application.hpp"
#include "CAE/ArgsHandler.hpp"
#include "Utils/Logger.hpp"

int main(const int argc, const char *const *argv, const char *const *envp)
{
    utl::Logger::init();
    cae::ArgsHandler argsHandler{};
    try
    {
        auto [run] = cae::ArgsHandler::ParseArgs(argc, argv);
        auto env = cae::ArgsHandler::ParseEnv(envp);
        if (!run)
        {
            return EXIT_SUCCESS;
        }
        cae::Application app;
    }
    catch (const std::exception &e)
    {
        std::cerr << "Error: " << e.what() << '\n';
        return EXIT_FAILURE;
    }
    return EXIT_SUCCESS;
}
```

Index