

cae

0.0.0

Generated by Doxygen 1.11.0



---

1 cae	1
1.1 Cross-API-Engine   Rendering Engine with multiple dynamic backends . . . . .	1
1.1.1 Prerequisites . . . . .	2
1.1.2 External Libraries . . . . .	2
1.1.3 Contributing . . . . .	2
1.1.4 License . . . . .	2
2 Commit Norms	3
3 Namespace Index	5
3.1 Namespace List . . . . .	5
4 Hierarchical Index	7
4.1 Class Hierarchy . . . . .	7
5 Class Index	9
5.1 Class List . . . . .	9
6 File Index	11
6.1 File List . . . . .	11
7 Namespace Documentation	13
7.1 cae Namespace Reference . . . . .	13
7.2 cae::Audio Namespace Reference . . . . .	13
7.2.1 Variable Documentation . . . . .	13
7.2.1.1 MUTED . . . . .	13
7.2.1.2 VOLUME . . . . .	14
7.3 cae::Network Namespace Reference . . . . .	14
7.3.1 Variable Documentation . . . . .	14
7.3.1.1 HOST . . . . .	14
7.3.1.2 PORT . . . . .	14
7.4 cae::User Namespace Reference . . . . .	14
7.4.1 Variable Documentation . . . . .	14
7.4.1.1 NAME . . . . .	14
7.5 cae::Window Namespace Reference . . . . .	15
7.5.1 Variable Documentation . . . . .	15
7.5.1.1 FULLSCREEN . . . . .	15
7.5.1.2 HEIGHT . . . . .	15
7.5.1.3 MAX_FPS . . . . .	15
7.5.1.4 NAME . . . . .	15
7.5.1.5 VSYNC . . . . .	15
7.5.1.6 WIDTH . . . . .	15
8 Class Documentation	17
8.1 cae::Application Class Reference . . . . .	17
8.1.1 Detailed Description . . . . .	18

8.1.2 Constructor & Destructor Documentation . . . . .	18
8.1.2.1 Application() [1/3] . . . . .	18
8.1.2.2 ~Application() . . . . .	18
8.1.2.3 Application() [2/3] . . . . .	18
8.1.2.4 Application() [3/3] . . . . .	18
8.1.3 Member Function Documentation . . . . .	18
8.1.3.1 operator=( ) [1/2] . . . . .	18
8.1.3.2 operator=( ) [2/2] . . . . .	18
8.2 cae::ArgsConfig Struct Reference . . . . .	19
8.2.1 Detailed Description . . . . .	19
8.2.2 Member Data Documentation . . . . .	19
8.2.2.1 run . . . . .	19
8.3 cae::ArgsHandler Class Reference . . . . .	20
8.3.1 Detailed Description . . . . .	20
8.3.2 Constructor & Destructor Documentation . . . . .	21
8.3.2.1 ArgsHandler() [1/3] . . . . .	21
8.3.2.2 ~ArgsHandler() . . . . .	21
8.3.2.3 ArgsHandler() [2/3] . . . . .	21
8.3.2.4 ArgsHandler() [3/3] . . . . .	21
8.3.3 Member Function Documentation . . . . .	21
8.3.3.1 operator=( ) [1/2] . . . . .	21
8.3.3.2 operator=( ) [2/2] . . . . .	21
8.3.3.3 ParseArgs() . . . . .	21
8.3.3.4 ParseEnv() . . . . .	22
8.4 cae::EnvConfig Struct Reference . . . . .	22
8.4.1 Detailed Description . . . . .	22
8.5 cae::OPGL Class Reference . . . . .	23
8.5.1 Detailed Description . . . . .	24
8.5.2 Constructor & Destructor Documentation . . . . .	24
8.5.2.1 OPGL() [1/3] . . . . .	24
8.5.2.2 ~OPGL() . . . . .	25
8.5.2.3 OPGL() [2/3] . . . . .	25
8.5.2.4 OPGL() [3/3] . . . . .	25
8.5.3 Member Function Documentation . . . . .	25
8.5.3.1 getName() . . . . .	25
8.5.3.2 getType() . . . . .	25
8.5.3.3 operator=( ) [1/2] . . . . .	25
8.5.3.4 operator=( ) [2/2] . . . . .	25
8.6 cae::VULKN Class Reference . . . . .	26
8.6.1 Detailed Description . . . . .	27
8.6.2 Constructor & Destructor Documentation . . . . .	27
8.6.2.1 VULKN() [1/3] . . . . .	27
8.6.2.2 ~VULKN() . . . . .	28

---

8.6.2.3 VULKN() [2/3] . . . . .	28
8.6.2.4 VULKN() [3/3] . . . . .	28
8.6.3 Member Function Documentation . . . . .	28
8.6.3.1 getName() . . . . .	28
8.6.3.2 getType() . . . . .	28
8.6.3.3 operator=() [1/2] . . . . .	28
8.6.3.4 operator=() [2/2] . . . . .	28
 9 File Documentation . . . . .	29
9.1 /home/masina/Projects/Cross-API-Engine/CONTRIBUTING.md File Reference . . . . .	29
9.2 /home/masina/Projects/Cross-API-Engine/include/CAE/Application.hpp File Reference . . . . .	29
9.2.1 Detailed Description . . . . .	30
9.3 Application.hpp . . . . .	30
9.4 /home/masina/Projects/Cross-API-Engine/include/CAE/ArgsHandler.hpp File Reference . . . . .	30
9.4.1 Detailed Description . . . . .	31
9.5 ArgsHandler.hpp . . . . .	31
9.6 /home/masina/Projects/Cross-API-Engine/include/CAE/Common.hpp File Reference . . . . .	32
9.6.1 Detailed Description . . . . .	32
9.7 Common.hpp . . . . .	32
9.8 /home/masina/Projects/Cross-API-Engine/include/CAE/Generated/Version.hpp File Reference . . . . .	33
9.8.1 Macro Definition Documentation . . . . .	33
9.8.1.1 BUILD_TYPE . . . . .	33
9.8.1.2 GIT_COMMIT_HASH . . . . .	34
9.8.1.3 GIT_TAG . . . . .	34
9.8.1.4 PROJECT_NAME . . . . .	34
9.8.1.5 PROJECT_VERSION . . . . .	34
9.8.1.6 PROJECT_VERSION_MAJOR . . . . .	34
9.8.1.7 PROJECT_VERSION_MINOR . . . . .	34
9.8.1.8 PROJECT_VERSION_PATCH . . . . .	34
9.9 Version.hpp . . . . .	35
9.10 /home/masina/Projects/Cross-API-Engine/plugins/Renderer/OpenGL/include/OPGL/↔ OPGL.hpp File Reference . . . . .	35
9.10.1 Detailed Description . . . . .	36
9.11 OPGL.hpp . . . . .	36
9.12 /home/masina/Projects/Cross-API-Engine/plugins/Renderer/OpenGL/src/entrypoint.cpp File Reference . . . . .	37
9.12.1 Function Documentation . . . . .	37
9.12.1.1 entryPoint() . . . . .	37
9.13 entrypoint.cpp . . . . .	38
9.14 /home/masina/Projects/Cross-API-Engine/plugins/Renderer/Vulkan/src/entrypoint.cpp File Reference . . . . .	38
9.14.1 Function Documentation . . . . .	38
9.14.1.1 entryPoint() . . . . .	38

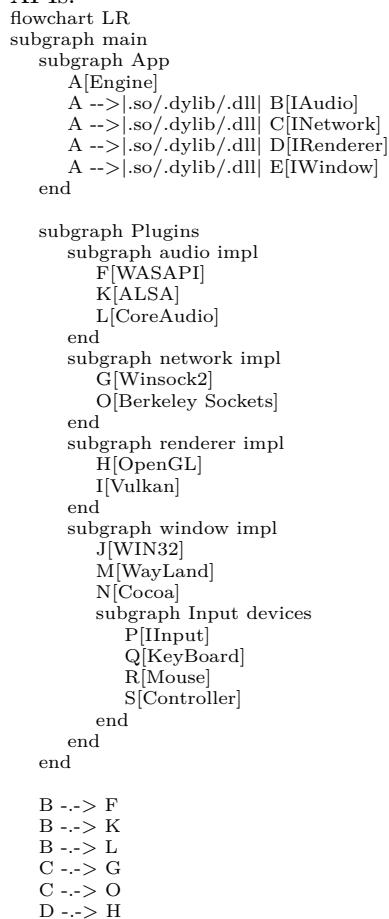
9.15 entrypoint.cpp . . . . .	39
9.16 /home/masina/Projects/Cross-API-Engine/plugins/Renderer/Vulkan/include/VULKN/← VULKN.hpp File Reference . . . . .	39
9.16.1 Detailed Description . . . . .	40
9.17 VULKN.hpp . . . . .	40
9.18 /home/masina/Projects/Cross-API-Engine/README.md File Reference . . . . .	41
9.19 /home/masina/Projects/Cross-API-Engine/src/application.cpp File Reference . . . . .	41
9.20 application.cpp . . . . .	41
9.21 /home/masina/Projects/Cross-API-Engine/src/argsHandler.cpp File Reference . . . . .	42
9.21.1 Macro Definition Documentation . . . . .	42
9.21.1.1 APP_EXTENSION . . . . .	42
9.21.2 Variable Documentation . . . . .	42
9.21.2.1 ARGS_MAP . . . . .	42
9.21.2.2 HELP_MSG . . . . .	43
9.21.2.3 VERSION_MSG . . . . .	43
9.22 argsHandler.cpp . . . . .	43
9.23 /home/masina/Projects/Cross-API-Engine/src/main.cpp File Reference . . . . .	44
9.23.1 Function Documentation . . . . .	44
9.23.1.1 main() . . . . .	44
9.24 main.cpp . . . . .	45
Index . . . . .	47

# 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.



```
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:

<a href="#">cae</a>	.....	13
<a href="#">cae::Audio</a>	.....	13
<a href="#">cae::Network</a>	.....	14
<a href="#">cae::User</a>	.....	14
<a href="#">cae::Window</a>	.....	15



## Chapter 4

# Hierarchical Index

### 4.1 Class Hierarchy

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

cae::Application . . . . .	17
cae::ArgsConfig . . . . .	19
cae::ArgsHandler . . . . .	20
cae::EnvConfig . . . . .	22
IRenderer	
cae::OPGL . . . . .	23
cae::VULKN . . . . .	26



# Chapter 5

## Class Index

### 5.1 Class List

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

<a href="#">cae::Application</a>	
Main class	17
<a href="#">cae::ArgsConfig</a>	
	19
<a href="#">cae::ArgsHandler</a>	
Class to handle command line arguments	20
<a href="#">cae::EnvConfig</a>	
	22
<a href="#">cae::OPGL</a>	
Class for the OpenGL plugin	23
<a href="#">cae::VULKN</a>	
Class for the Vulkan plugin	26



# Chapter 6

## File Index

### 6.1 File List

Here is a list of all files with brief descriptions:

/home/masina/Projects/Cross-API-Engine/include/CAE/ <a href="#">Application.hpp</a>	29
This file contains the Application class declaration . . . . .	
/home/masina/Projects/Cross-API-Engine/include/CAE/ <a href="#">ArgsHandler.hpp</a>	30
This file contains the ArgsHandler class declaration . . . . .	
/home/masina/Projects/Cross-API-Engine/include/CAE/ <a href="#">Common.hpp</a>	32
This file contains . . . . .	
/home/masina/Projects/Cross-API-Engine/include/CAE/Generated/ <a href="#">Version.hpp</a>	33
/home/masina/Projects/Cross-API-Engine/plugins/Renderer/OpenGL/include/OPGL/ <a href="#">OPGL.hpp</a>	35
This file contains the OPGL class declaration . . . . .	
/home/masina/Projects/Cross-API-Engine/plugins/Renderer/OpenGL/src/ <a href="#">entrypoint.cpp</a>	37
/home/masina/Projects/Cross-API-Engine/plugins/Renderer/Vulkan/include/VULKN/ <a href="#">VULKN.hpp</a>	39
This file contains the VULKN class declaration . . . . .	
/home/masina/Projects/Cross-API-Engine/plugins/Renderer/Vulkan/src/ <a href="#">entrypoint.cpp</a>	38
/home/masina/Projects/Cross-API-Engine/src/ <a href="#">application.cpp</a>	41
/home/masina/Projects/Cross-API-Engine/src/ <a href="#">argsHandler.cpp</a>	42
/home/masina/Projects/Cross-API-Engine/src/ <a href="#">main.cpp</a>	44



# 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:

cae::Application
+ Application() + ~Application() + Application(const Application &)=delete + operator=(const Application &)=delete + Application(Application &&)=delete + operator=(Application &&)=delete

#### 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:

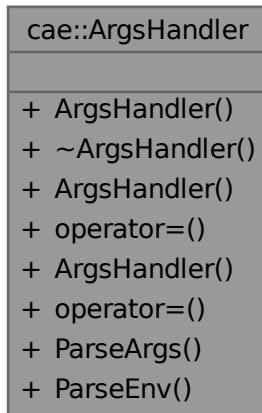
- /home/masina/Projects/Cross-API-Engine/include/CAE/[ArgsHandler.hpp](#)

## 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:



### 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:



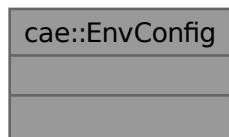
The documentation for this class was generated from the following files:

- [/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:

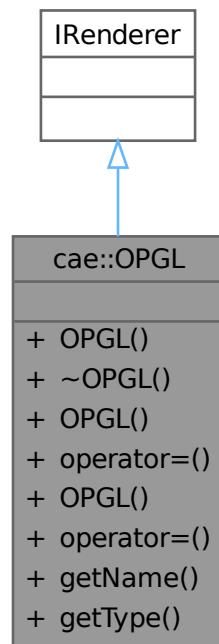
- [/home/masina/Projects/Cross-API-Engine/include/CAE/ArgsHandler.hpp](#)

## 8.5 cae::OPGL Class Reference

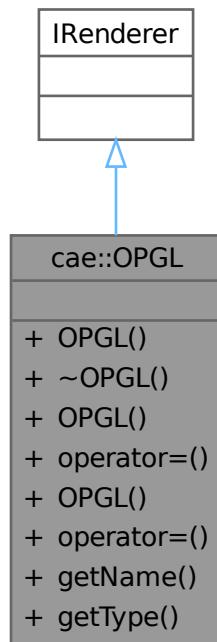
Class for the OpenGL plugin.

```
#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:

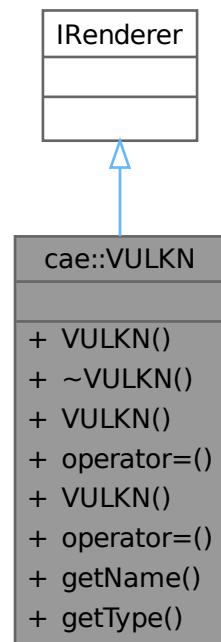
- /home/masina/Projects/Cross-API-Engine/plugins/Renderer/OpenGL/include/OPGL/[OPGL.hpp](#)

## 8.6 cae::VULKN Class Reference

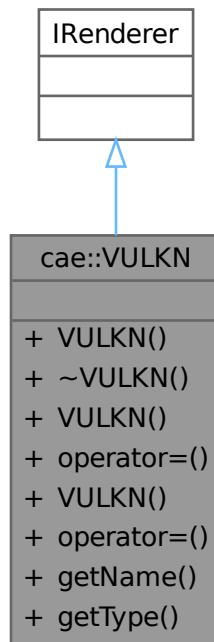
Class for the Vulkan plugin.

```
#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

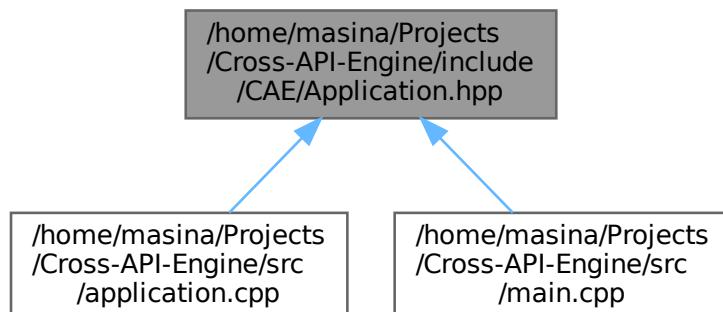
## File Documentation

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

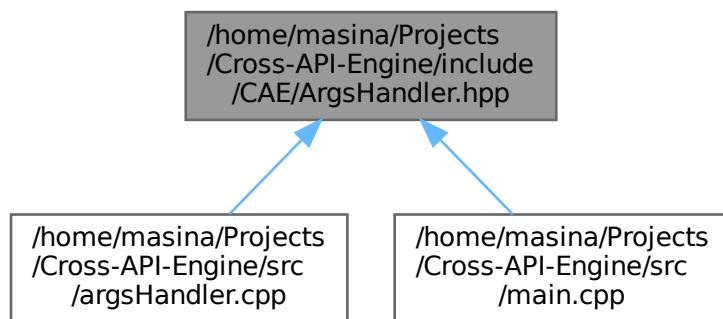
[Go to the documentation of this file.](#)

```
00001 /**
00002 /// @file Application.hpp
00003 /// @brief This file contains the Application class declaration
00004 /// @namespace cae
00005 /**
00006
00007 #pragma once
00008
00009 namespace cae
00010 {
00011
00012 /**
00013 /// @class Application
00014 /// @brief Main class
00015 /// @namespace cae
00016 /**
00017 class Application
00018 {
00019
00020     public:
00021         Application();
00022         ~Application() = default;
00023
00024         Application(const Application &) = delete;
00025         Application &operator=(const Application &) = delete;
00026         Application(Application &&) = delete;
00027         Application &operator=(Application &&) = delete;
00028
00029     }; // class Application
00030
00031 } // namespace cae
```

## 9.4 /home/masina/Projects/Cross-API-Engine/include/CAE/ArgsHandler.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.](#)

```

00001 /**
00002 /// @file ArgsHandler.hpp
00003 /// @brief This file contains the ArgsHandler class declaration
00004 /// @namespace cae
00005 /**
00006
00007 #pragma once
00008
00009 namespace cae
00010 {
00011
00012     struct ArgsConfig
00013     {
00014         bool run{false};
00015     };
00016     struct EnvConfig
00017     {
00018     };
00019
00020 /**
00021 /// @class ArgsHandler
00022 /// @brief Class to handle command line arguments
00023 /// @namespace cae
00024 /**
00025 class ArgsHandler
00026 {
00027
00028     public:
00029         ArgsHandler() = default;
00030         ~ArgsHandler() = default;
00031
00032         ArgsHandler(const ArgsHandler &) = delete;
00033         ArgsHandler &operator=(const ArgsHandler &) = delete;
00034         ArgsHandler(ArgsHandler &&) = delete;
00035         ArgsHandler &operator=(ArgsHandler &&) = delete;
00036
00037         static ArgsConfig ParseArgs(int argc, const char *const *argv);
00038         static EnvConfig ParseEnv(const char *const *envp);
00039
00040     private:
00041 }; // class ArgsHandler
00042
00043 } // 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.](#)

```
00001 /**
00002 /**
00003 /**
00004 /**
00005 /**
00006
00007 #pragma once
00008
00009 namespace cae
00010 {
00011     namespace Audio
00012     {
00013         inline constexpr auto VOLUME = 50.F;
00014         inline constexpr auto MUTED = false;
00015     } // namespace Audio
00016     namespace Network
00017     {
00018         inline constexpr auto HOST = "127.0.0.1";
00019         inline constexpr auto PORT = 4242;
00020     } // namespace Network
00021     namespace User
```

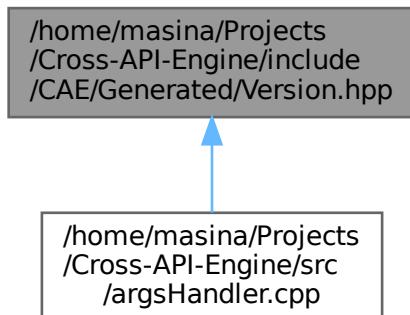
```

00022  {
00023      inline constexpr auto NAME = "User";
00024  }
00025  namespace Window
00026  {
00027      inline constexpr auto HEIGHT = 1920;
00028      inline constexpr auto WIDTH = 1080;
00029      inline constexpr auto NAME = "CAE - Cross API Engine";
00030      inline constexpr auto FULLSCREEN = false;
00031      inline constexpr auto VSYNC = false;
00032      inline constexpr auto MAX_FPS = 90;
00033 } // namespace Window
00034 } // 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.](#)

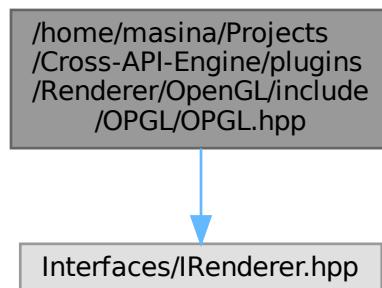
```
00001 #pragma once
00002 //
00003 //=====
00004 // DO NOT EDIT THIS FILE MANUALLY. IT IS GENERATED BY CMAKE DURING THE BUILD PROCESS.
00005 //=====
00006
00007 #define PROJECT_NAME "cae"
00008 #define PROJECT_VERSION "0.0.0"
00009 #define PROJECT_VERSION_MAJOR "0"
00010 #define PROJECT_VERSION_MINOR "0"
00011 #define PROJECT_VERSION_PATCH "0"
00012
00013 #define GIT_COMMIT_HASH "0d462a3"
00014 #define GIT_TAG "0d462a3"
00015 #define BUILD_TYPE "Release"
```

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

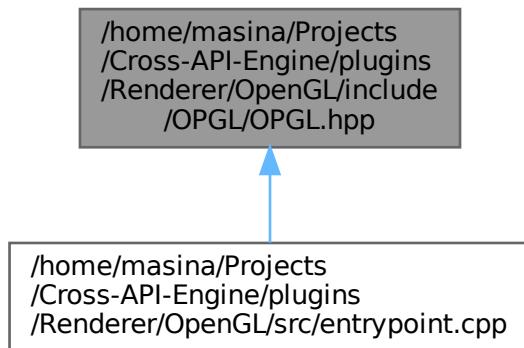
This file contains the OPGL class declaration.

```
#include "Interfaces/IRenderer.hpp"
```

Include dependency graph for OPGL.hpp:



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



## 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.](#)

```

00001 /**
00002 /// @file OPGL.hpp
00003 /// @brief This file contains the OPGL class declaration
00004 /// @namespace cae
00005 /**
00006
00007 #pragma once
00008
00009 #include "Interfaces/IRenderer.hpp"
00010
00011 namespace cae
00012 {
00013
00014 /**
00015 /// @class OPGL
00016 /// @brief Class for the OpenGL plugin
00017 /// @namespace cae
00018 /**
00019 class OPGL final : public IRenderer
  
```

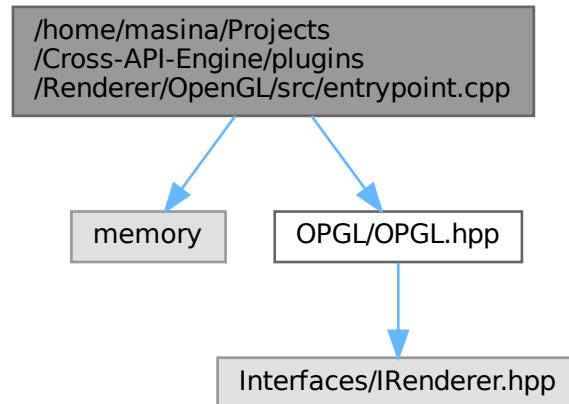
```

00020  {
00021
00022  public:
00023      OPGL() = default;
00024      ~OPGL() override = default;
00025
00026      OPGL(const OPGL &) = delete;
00027      OPGL &operator=(const OPGL &) = delete;
00028      OPGL(OPGL &&) = delete;
00029      OPGL &operator=(OPGL &&) = delete;
00030
00031      [[nodiscard]] std::string getName() const override { return "OpenGL"; }
00032      [[nodiscard]] utl::PluginType getType() const override { return utl::PluginType::RENDERER; }
00033
00034  }; // class OPGL
00035
00036 } // 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

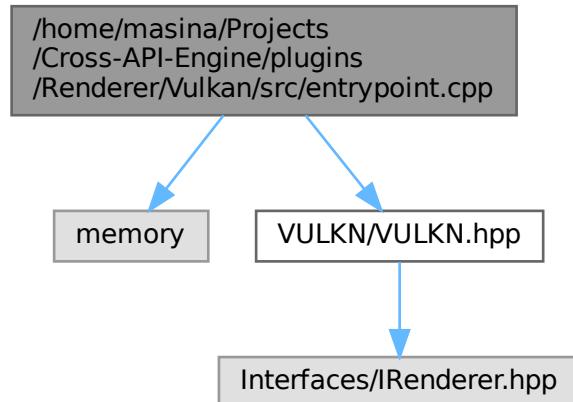
[Go to the documentation of this file.](#)

```
00001 #include <memory>
00002
00003 #include "OPGL/OPGL.hpp"
00004
00005 extern "C"
00006 {
00007     cae::IRenderer *entryPoint() { return std::make_unique<cae::OPGL>().release(); }
00008 }
```

## 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.](#)

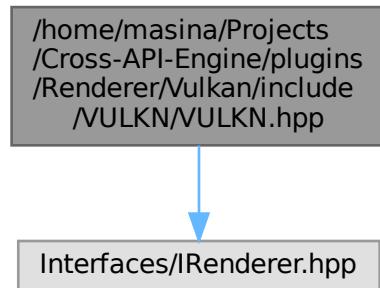
```
00001 #include <memory>
00002
00003 #include "VULKN/VULKN.hpp"
00004
00005 extern "C"
00006 {
00007     cae::IRenderer *entryPoint() { return std::make_unique<cae::VULKN>().release(); }
00008 }
```

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

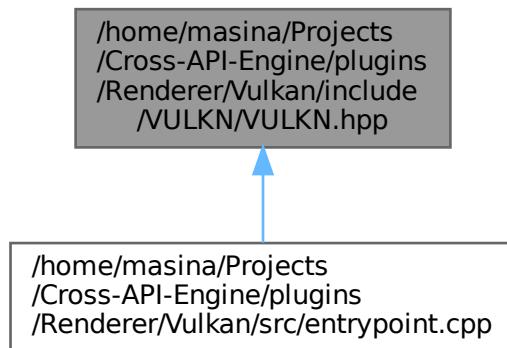
This file contains the VULKN class declaration.

```
#include "Interfaces/IRenderer.hpp"
```

Include dependency graph for VULKN.hpp:



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



## 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.](#)

```

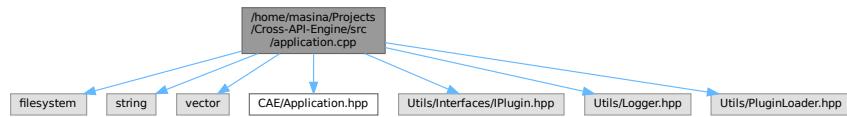
00001 /**
00002 /**
00003 /**
00004 /**
00005 /**
00006
00007 #pragma once
00008
00009 #include "Interfaces/IRenderer.hpp"
00010
00011 namespace cae
00012 {
00013
00014 /**
00015 /**
00016 /**
00017 /**
00018 /**
00019 class VULKN final : public IRenderer
00020 {
00021
00022     public:
00023         VULKN() = default;
00024         ~VULKN() override = default;
00025
00026         VULKN(const VULKN &) = delete;
00027         VULKN &operator=(const VULKN &) = delete;
00028         VULKN(VULKN &&) = delete;
00029         VULKN &operator=(VULKN &&) = delete;
00030
00031         [[nodiscard]] std::string getName() const override { return "Vulkan"; }
00032         [[nodiscard]] utl::PluginType getType() const override { return utl::PluginType::RENDERER; }
00033
00034     }; // class VULKN
00035
00036 } // 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:
```



## 9.20 application.cpp

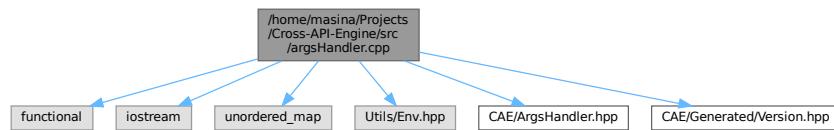
[Go to the documentation of this file.](#)

```
00001 #include <filesystem>
00002 #include <string>
00003 #include <vector>
00004
00005 #include "CAE/Application.hpp"
00006 #include "Utils/Interfaces/IPlugin.hpp"
00007 #include "Utils/Logger.hpp"
00008 #include "Utils/PluginLoader.hpp"
00009
00010 cae::Application::Application()
00011 {
00012     try
00013     {
00014         const std::filesystem::path pluginDir{PLUGINS_DIR};
00015         utl::PluginLoader pluginManager;
00016         std::vector<std::string> loadedPlugins;
00017         for (const auto &entry : std::filesystem::directory_iterator(pluginDir))
00018         {
00019             if (!entry.is_regular_file() || entry.path().extension() != PLUGINS_EXTENSION)
00020             {
00021                 continue;
00022             }
00023             if (const std::string pluginPath = entry.path().string();
00024                 pluginManager.loadPlugin<utl::IPlugin>(pluginPath) != nullptr)
00025             {
00026                 loadedPlugins.push_back(entry.path().filename().string());
00027             }
00028             else
00029             {
00030                 utl::Logger::log("Failed to load plugin: " + pluginPath, utl::LogLevel::WARNING);
00031             }
00032         }
00033         if (loadedPlugins.empty())
00034         {
00035             utl::Logger::log("No plugins loaded from directory: " + pluginDir.string(), utl::LogLevel::WARNING);
00036         }
00037     }
00038     catch (const std::exception &e)
00039     {
00040         std::cerr << "Error: " << e.what() << '\n';
00041     }
00042 }
```

## 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`.

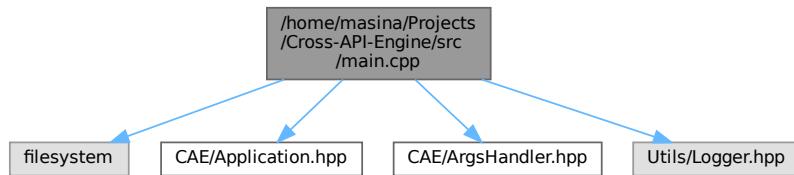
## 9.22 argsHandler.cpp

[Go to the documentation of this file.](#)

```
00001 #include <functional>
00002 #include <iostream>
00003 #include <unordered_map>
00004
00005 #ifdef _WIN32
00006 #define APP_EXTENSION ".exe"
00007 #else
00008 #define APP_EXTENSION ""
00009 #endif
00010
00011 #include "Utils/Env.hpp"
00012
00013 #include "CAE/ArgsHandler.hpp"
00014 #include "CAE/Generated/Version.hpp"
00015
00016 static constexpr std::string_view HELP_MSG = "Usage: " PROJECT_NAME APP_EXTENSION " [options]\n\n"
00017             "Options:\n"
00018             " -h, --help      Show this help message\n"
00019             " -v, --version   Show version information\n";
00020
00021 static constexpr std::string_view VERSION_MSG = PROJECT_NAME
00022   " v" PROJECT_VERSION " " BUILD_TYPE "(" GIT_TAG ", commit " GIT_COMMIT_HASH ") " __DATE__ "
  __TIME__ "\n";
00023
00024 static const std::unordered_map<std::string, std::function<void()>> ARGS_MAP = {
00025   {"-h", []() { std::cout << HELP_MSG; }},
00026   {"-help", []() { std::cout << HELP_MSG; }},
00027   {"-v", []() { std::cout << VERSION_MSG; }},
00028   {"--version", []() { std::cout << VERSION_MSG; }}};
00029
00030 cae::ArgsConfig cae::ArgsHandler::ParseArgs(const int argc, const char *const *argv)
00031 {
00032   if (argc <= 1)
00033   {
00034     return {.run = true};
00035   }
00036   const std::string arg1{argv[1]};
00037   if (const auto it = ARGS_MAP.find(arg1); it != ARGS_MAP.end())
00038   {
00039     it->second();
00040     return {.run = false};
00041   }
00042   throw std::runtime_error("Unknown argument: " + arg1 + ". Use -h or --help to see available options.");
00043 }
00044
00045 cae::EnvConfig cae::ArgsHandler::ParseEnv(const char *const *envp)
00046 {
00047   for (const auto &fst, snd) : utl::getEnvMap(envp)
00048   {
00049     std::cout << "var:" << fst << ":" << snd << '\n';
00050   }
00051   return {};
00052 }
```

## 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 \*const argv, const char \*const \*envp)

#### 9.23.1 Function Documentation

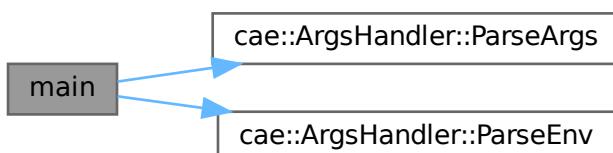
##### 9.23.1.1 [main\(\)](#)

```
int main (
    const int argc,
    const char *const * 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.](#)

```
00001 #include <filesystem>
00002
00003 #include "CAE/Application.hpp"
00004 #include "CAE/ArgsHandler.hpp"
00005 #include "Utils/Logger.hpp"
00006
00007 int main(const int argc, const char *const *const argv, const char *const *const envp)
00008 {
00009     utl::Logger::init();
00010     cae::ArgsHandler argsHandler{};
00011     try
00012     {
00013         auto [run] = cae::ArgsHandler::ParseArgs(argc, argv);
00014         auto env = cae::ArgsHandler::ParseEnv(envp);
00015         if (!run)
00016         {
00017             return EXIT_SUCCESS;
00018         }
00019         cae::Application app;
00020     }
00021     catch (const std::exception &e)
00022     {
00023         std::cerr << "Error: " << e.what() << '\n';
00024         return EXIT_FAILURE;
00025     }
00026     return EXIT_SUCCESS;
00027 }
```



## Index

/home/masina/Projects/Cross-API-Engine/CONTRIBUTION.rst, 33  
29

/home/masina/Projects/Cross-API-Engine/README.md, 13  
41 cae::Application, 17

/home/masina/Projects/Cross-API-Engine/include/CAE/Application.hpp, 18  
29, 30 Application, 18

/home/masina/Projects/Cross-API-Engine/include/CAE/ArgsHandler.hpp, 18  
30, 31 cae::ArgsConfig, 19

/home/masina/Projects/Cross-API-Engine/include/CAE/Common.hpp, 19  
32 cae::ArgsHandler, 20

/home/masina/Projects/Cross-API-Engine/include/CAE/ArgsHandler.hpp, 21  
33, 35 ArgsHandler, 21

/home/masina/Projects/Cross-API-Engine/plugins/Render/OpenGL/include/OPGL/OPGL.hpp, 36  
35, 36 ParseArgs, 21

/home/masina/Projects/Cross-API-Engine/plugins/Render/OpenGL/src/entrypoint.cpp, 37, 38  
cae::Audio, 13

/home/masina/Projects/Cross-API-Engine/plugins/Render/VULKAN/include/VULKN/VULKN.hpp, 39, 40 VOLUME, 13

/home/masina/Projects/Cross-API-Engine/plugins/Render/VULKAN/src/entrypoint.cpp, 38, 39 cae::Network, 14

/home/masina/Projects/Cross-API-Engine/src/application.hpp, 14  
41 PORT, 14

/home/masina/Projects/Cross-API-Engine/src/argsHandler.hpp, 23  
42, 43 ~OPGL, 24

/home/masina/Projects/Cross-API-Engine/src/main.cpp, 25  
44, 45 getName, 25  
getType, 25  
operator=, 25  
OPGL, 24, 25

~Application  
cae::Application, 18

~ArgsHandler  
cae::ArgsHandler, 21

~OPGL  
cae::OPGL, 24

~VULKN  
cae::VULKN, 27

APP\_EXTENSION  
argsHandler.cpp, 42

Application  
cae::Application, 18

ARGS\_MAP  
argsHandler.cpp, 42

ArgsHandler  
cae::ArgsHandler, 21

argsHandler.cpp  
APP\_EXTENSION, 42  
ARGS\_MAP, 42  
HELP\_MSG, 42  
VERSION\_MSG, 43

BUILD\_TYPE

FULLSCREEN  
cae::Window, 15

getName  
cae::OPGL, 25  
cae::VULKN, 28

getType  
cae::OPGL, 25  
cae::VULKN, 28

GIT\_COMMIT\_HASH  
Version.hpp, 33

GIT\_TAG  
Version.hpp, 34

HEIGHT  
cae::Window, 15

HELP\_MSG  
argsHandler.cpp, 42

HOST  
cae::Network, 14

main  
main.cpp, 44

main.cpp  
main, 44

MAX\_FPS  
cae::Window, 15

MUTED  
cae::Audio, 13

NAME  
cae::User, 14  
cae::Window, 15

operator=  
cae::Application, 18  
cae::ArgsHandler, 21  
cae::OPGL, 25  
cae::VULKN, 28

OPGL  
cae::OPGL, 24, 25

ParseArgs  
cae::ArgsHandler, 21

ParseEnv  
cae::ArgsHandler, 21

PORT  
cae::Network, 14

PROJECT\_NAME  
Version.hpp, 34

PROJECT\_VERSION  
Version.hpp, 34

PROJECT\_VERSION\_MAJOR  
Version.hpp, 34

PROJECT\_VERSION\_MINOR  
Version.hpp, 34

PROJECT\_VERSION\_PATCH  
Version.hpp, 34

run

cae::ArgsConfig, 19

Version.hpp  
BUILD\_TYPE, 33  
GIT\_COMMIT\_HASH, 33  
GIT\_TAG, 34  
PROJECT\_NAME, 34  
PROJECT\_VERSION, 34  
PROJECT\_VERSION\_MAJOR, 34  
PROJECT\_VERSION\_MINOR, 34  
PROJECT\_VERSION\_PATCH, 34

VERSION\_MSG  
argsHandler.cpp, 43

VOLUME  
cae::Audio, 13

VSYNC  
cae::Window, 15

VULKN  
cae::VULKN, 27, 28

WIDTH  
cae::Window, 15