

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 README	3
3 README	5
4 README	7
5 README	9
6 README	11
7 Commit Norms	13
8 LICENSE	15
9 Namespace Index	17
9.1 Namespace List	17
10 Hierarchical Index	19
10.1 Class Hierarchy	19
11 Class Index	21
11.1 Class List	21
12 File Index	23
12.1 File List	23
13 Namespace Documentation	25
13.1 cae Namespace Reference	25
13.1.1 Enumeration Type Documentation	27
13.1.1.1 GamepadAxis	27
13.1.1.2 GamepadButton	27
13.1.1.3 KeyCode	27
13.1.1.4 KeyState	29
13.1.1.5 MouseButton	30
13.2 cae::Audio Namespace Reference	30
13.2.1 Variable Documentation	30
13.2.1.1 MUTED	30
13.2.1.2 VOLUME	30
13.3 cae::Message Namespace Reference	31
13.3.1 Variable Documentation	31
13.3.1.1 HELP_MSG	31

13.3.1.2 VERSION_MSG	31
13.4 cae::Network Namespace Reference	31
13.4.1 Variable Documentation	31
13.4.1.1 HOST	31
13.4.1.2 PORT	32
13.5 cae::Plugins Namespace Reference	32
13.6 cae::Plugins::Name Namespace Reference	32
13.6.1 Variable Documentation	32
13.6.1.1 RENDERER_OPENGL	32
13.6.1.2 RENDERER_VULKAN	32
13.6.1.3 WINDOW_GLFW	32
13.7 cae::Renderer Namespace Reference	32
13.7.1 Variable Documentation	33
13.7.1.1 FRAME_RATE_LIMIT	33
13.7.1.2 VSYNC	33
13.8 cae::User Namespace Reference	33
13.8.1 Variable Documentation	33
13.8.1.1 NAME	33
13.9 cae::Window Namespace Reference	33
13.9.1 Variable Documentation	33
13.9.1.1 FULLSCREEN	33
13.9.1.2 HEIGHT	34
13.9.1.3 NAME	34
13.9.1.4 WIDTH	34
14 Class Documentation	35
14.1 cae::AppConfig Struct Reference	35
14.1.1 Detailed Description	36
14.1.2 Member Data Documentation	37
14.1.2.1 engineConfig	37
14.1.2.2 envConfig	37
14.2 cae::Application Class Reference	37
14.2.1 Detailed Description	39
14.2.2 Constructor & Destructor Documentation	39
14.2.2.1 Application() [1/3]	39
14.2.2.2 ~Application()	39
14.2.2.3 Application() [2/3]	40
14.2.2.4 Application() [3/3]	40
14.2.3 Member Function Documentation	40
14.2.3.1 operator=() [1/2]	40
14.2.3.2 operator=() [2/2]	40
14.2.3.3 parseEngineConf()	40
14.2.3.4 setupEngine()	41

14.2.3.5 start()	41
14.2.3.6 stop()	41
14.2.4 Member Data Documentation	41
14.2.4.1 m_appConfig	41
14.2.4.2 m_engine	42
14.2.4.3 m_pluginLoader	42
14.3 cae::ArgsConfig Struct Reference	42
14.3.1 Detailed Description	43
14.3.2 Member Data Documentation	43
14.3.2.1 config_path	43
14.3.2.2 run	43
14.4 cae::ArgsHandler Class Reference	43
14.4.1 Detailed Description	44
14.4.2 Constructor & Destructor Documentation	44
14.4.2.1 ArgsHandler() [1/3]	44
14.4.2.2 ~ArgsHandler()	44
14.4.2.3 ArgsHandler() [2/3]	44
14.4.2.4 ArgsHandler() [3/3]	44
14.4.3 Member Function Documentation	44
14.4.3.1 operator=() [1/2]	44
14.4.3.2 operator=() [2/2]	45
14.4.3.3 ParseArgs()	45
14.4.3.4 ParseEnv()	45
14.5 cae::Engine Class Reference	46
14.5.1 Detailed Description	47
14.5.2 Constructor & Destructor Documentation	47
14.5.2.1 Engine() [1/3]	47
14.5.2.2 ~Engine()	47
14.5.2.3 Engine() [2/3]	48
14.5.2.4 Engine() [3/3]	48
14.5.3 Member Function Documentation	48
14.5.3.1 getAudio()	48
14.5.3.2 getClock()	48
14.5.3.3 getInput()	48
14.5.3.4 getNetwork()	48
14.5.3.5 getRenderer()	49
14.5.3.6 getWindow()	49
14.5.3.7 operator=() [1/2]	49
14.5.3.8 operator=() [2/2]	49
14.5.3.9 run()	49
14.5.3.10 stop()	49
14.5.4 Member Data Documentation	49
14.5.4.1 m_audioPlugin	49

14.5.4.2 m_clock	50
14.5.4.3 m_inputPlugin	50
14.5.4.4 m_networkPlugin	50
14.5.4.5 m_rendererPlugin	50
14.5.4.6 m_windowPlugin	50
14.6 cae::EngineConfig Struct Reference	51
14.6.1 Detailed Description	52
14.6.2 Member Data Documentation	52
14.6.2.1 audio_master_volume	52
14.6.2.2 audio_muted	52
14.6.2.3 network_host	52
14.6.2.4 network_port	52
14.6.2.5 renderer_frame_rate_limit	52
14.6.2.6 renderer_vsync	53
14.6.2.7 windowFullscreen	53
14.6.2.8 window_height	53
14.6.2.9 window_name	53
14.6.2.10 window_width	53
14.7 cae::EnvConfig Struct Reference	54
14.7.1 Detailed Description	54
14.7.2 Member Data Documentation	54
14.7.2.1 pwd	54
14.7.2.2 user_name	55
14.8 cae::GLFW Class Reference	55
14.8.1 Detailed Description	58
14.8.2 Constructor & Destructor Documentation	58
14.8.2.1 GLFW() [1/3]	58
14.8.2.2 ~GLFW()	58
14.8.2.3 GLFW() [2/3]	58
14.8.2.4 GLFW() [3/3]	59
14.8.3 Member Function Documentation	59
14.8.3.1 close()	59
14.8.3.2 create()	59
14.8.3.3 frameBufferResizeCallback()	59
14.8.3.4 getName()	59
14.8.3.5 getNativeHandle()	59
14.8.3.6 getPlatform()	60
14.8.3.7 getType()	60
14.8.3.8 getWindowSize()	60
14.8.3.9 operator=() [1/2]	60
14.8.3.10 operator=() [2/2]	60
14.8.3.11 pollEvents()	60
14.8.3.12 resetResizedFlag()	60

14.8.3.13 setIcon()	61
14.8.3.14 shouldClose()	61
14.8.3.15 wasResized()	61
14.8.4 Member Data Documentation	61
14.8.4.1 m_frameBufferResized	61
14.8.4.2 m_frameBufferSize	61
14.8.4.3 m_window	62
14.9 cae::IAudio Interface Reference	62
14.9.1 Detailed Description	63
14.9.2 Constructor & Destructor Documentation	63
14.9.2.1 ~IAudio()	63
14.10 cae::IGamepad Interface Reference	64
14.10.1 Detailed Description	65
14.10.2 Constructor & Destructor Documentation	65
14.10.2.1 ~IGamepad()	65
14.11 cae::IInput Interface Reference	65
14.11.1 Detailed Description	67
14.11.2 Constructor & Destructor Documentation	67
14.11.2.1 ~IInput()	67
14.11.3 Member Function Documentation	67
14.11.3.1 getGamepads()	67
14.11.3.2 getKeyboard()	67
14.11.3.3 getMouse()	67
14.11.4 Member Data Documentation	67
14.11.4.1 m_gamepads	67
14.11.4.2 m_keyboard	67
14.11.4.3 m_mouse	68
14.12 cae::IKeyboard Interface Reference	68
14.12.1 Detailed Description	69
14.12.2 Constructor & Destructor Documentation	69
14.12.2.1 ~IKeyboard()	69
14.12.3 Member Function Documentation	69
14.12.3.1 isKeyPressed()	69
14.12.4 Member Data Documentation	70
14.12.4.1 m_keyMap	70
14.13 cae::IMouse Interface Reference	70
14.13.1 Detailed Description	71
14.13.2 Constructor & Destructor Documentation	71
14.13.2.1 ~IMouse()	71
14.14 cae::INetwork Interface Reference	72
14.14.1 Detailed Description	73
14.14.2 Constructor & Destructor Documentation	73
14.14.2.1 ~INetwork()	73

14.14.3 Member Function Documentation	73
14.14.3.1 connect()	73
14.15 cae::IRenderer Interface Reference	73
14.15.1 Detailed Description	75
14.15.2 Constructor & Destructor Documentation	75
14.15.2.1 ~IRenderer()	75
14.15.3 Member Function Documentation	75
14.15.3.1 initialize()	75
14.16 cae::IWindow Interface Reference	76
14.16.1 Detailed Description	77
14.16.2 Constructor & Destructor Documentation	78
14.16.2.1 ~IWindow()	78
14.16.3 Member Function Documentation	78
14.16.3.1 close()	78
14.16.3.2 create()	78
14.16.3.3 getNativeHandle()	78
14.16.3.4 getWindowSize()	78
14.16.3.5 pollEvents()	78
14.16.3.6 resetResizedFlag()	78
14.16.3.7 setIcon()	79
14.16.3.8 shouldClose()	79
14.16.3.9 wasResized()	79
14.17 cae::NativeWindowHandle Struct Reference	79
14.17.1 Detailed Description	80
14.17.2 Member Data Documentation	80
14.17.2.1 display	80
14.17.2.2 window	80
14.18 cae::OPGL Class Reference	80
14.18.1 Detailed Description	83
14.18.2 Constructor & Destructor Documentation	83
14.18.2.1 OPGL() [1/3]	83
14.18.2.2 ~OPGL()	83
14.18.2.3 OPGL() [2/3]	83
14.18.2.4 OPGL() [3/3]	83
14.18.3 Member Function Documentation	83
14.18.3.1 getName()	83
14.18.3.2 getPlatform()	83
14.18.3.3 getType()	83
14.18.3.4 initialize()	84
14.18.3.5 operator=() [1/2]	84
14.18.3.6 operator=() [2/2]	84
14.19 cae::VULKN Class Reference	84
14.19.1 Detailed Description	87

14.19.2 Constructor & Destructor Documentation	87
14.19.2.1 VULKN() [1/3]	87
14.19.2.2 ~VULKN()	87
14.19.2.3 VULKN() [2/3]	87
14.19.2.4 VULKN() [3/3]	87
14.19.3 Member Function Documentation	87
14.19.3.1 getName()	87
14.19.3.2 getPlatform()	87
14.19.3.3 getType()	87
14.19.3.4 initialize()	88
14.19.3.5 operator=() [1/2]	88
14.19.3.6 operator=() [2/2]	88
14.20 cae::WindowSize Struct Reference	88
14.20.1 Detailed Description	89
14.20.2 Member Data Documentation	89
14.20.2.1 height	89
14.20.2.2 width	89
14.21 cae::X11 Class Reference	89
14.21.1 Detailed Description	92
14.21.2 Constructor & Destructor Documentation	92
14.21.2.1 X11() [1/3]	92
14.21.2.2 ~X11()	92
14.21.2.3 X11() [2/3]	92
14.21.2.4 X11() [3/3]	93
14.21.3 Member Function Documentation	93
14.21.3.1 close()	93
14.21.3.2 create()	93
14.21.3.3 getName()	93
14.21.3.4 getNativeHandle()	93
14.21.3.5 getPlatform()	93
14.21.3.6 getType()	94
14.21.3.7 getWindowSize()	94
14.21.3.8 operator=() [1/2]	94
14.21.3.9 operator=() [2/2]	94
14.21.3.10 pollEvents()	94
14.21.3.11 resetResizedFlag()	94
14.21.3.12 setIcon()	95
14.21.3.13 shouldClose()	95
14.21.3.14 wasResized()	95
14.21.4 Member Data Documentation	95
14.21.4.1 m_display	95
14.21.4.2 m_frameBufferResized	95
14.21.4.3 m_frameBufferSize	95

14.21.4.4 m_shouldClose	96
14.21.4.5 m_window	96
14.21.4.6 m_wmDeleteMessage	96
15 File Documentation	97
15.1 CONTRIBUTING.md File Reference	97
15.2 include/CAE/Application.hpp File Reference	97
15.2.1 Detailed Description	98
15.3 Application.hpp	98
15.4 include/CAE/ArgsHandler.hpp File Reference	99
15.4.1 Detailed Description	99
15.5 ArgsHandler.hpp	100
15.6 include/CAE/Common.hpp File Reference	100
15.6.1 Detailed Description	102
15.6.2 Macro Definition Documentation	102
15.6.2.1 APP_EXTENSION	102
15.7 Common.hpp	102
15.8 include/CAE/Engine/Engine.hpp File Reference	103
15.8.1 Detailed Description	104
15.9 Engine.hpp	104
15.10 include/CAE/Generated/Version.hpp File Reference	105
15.10.1 Macro Definition Documentation	106
15.10.1.1 BUILD_TYPE	106
15.10.1.2 GIT_COMMIT_HASH	106
15.10.1.3 GIT_TAG	106
15.10.1.4 PROJECT_NAME	106
15.10.1.5 PROJECT_VERSION	106
15.10.1.6 PROJECT_VERSION_MAJOR	106
15.10.1.7 PROJECT_VERSION_MINOR	106
15.10.1.8 PROJECT_VERSION_PATCH	107
15.11 Version.hpp	107
15.12 LICENSE.md File Reference	107
15.13 modules/Interfaces/include/Interfaces/IAudio.hpp File Reference	107
15.13.1 Detailed Description	108
15.14 IAudio.hpp	108
15.15 modules/Interfaces/include/Interfaces/INetwork.hpp File Reference	109
15.15.1 Detailed Description	110
15.16 INetwork.hpp	110
15.17 modules/Interfaces/include/Interfaces/Input/IGamepad.hpp File Reference	110
15.17.1 Detailed Description	112
15.18 IGamepad.hpp	112
15.19 modules/Interfaces/include/Interfaces/Input/IInput.hpp File Reference	112
15.19.1 Detailed Description	113

15.20 IInput.hpp	113
15.21 modules/Interfaces/include/Interfaces/Input/IKeyboard.hpp File Reference	114
15.21.1 Detailed Description	115
15.22 IKeyboard.hpp	115
15.23 modules/Interfaces/include/Interfaces/Input/IMouse.hpp File Reference	116
15.23.1 Detailed Description	117
15.24 IMouse.hpp	117
15.25 modules/Interfaces/include/Interfaces/Input/Key/Gamepad.hpp File Reference	118
15.26 Gamepad.hpp	119
15.27 modules/Interfaces/include/Interfaces/Input/Key/Keyboard.hpp File Reference	119
15.27.1 Detailed Description	121
15.28 Keyboard.hpp	121
15.29 modules/Interfaces/include/Interfaces/Input/Key/Mouse.hpp File Reference	122
15.29.1 Detailed Description	123
15.30 Mouse.hpp	123
15.31 modules/Interfaces/include/Interfaces/IRenderer.hpp File Reference	124
15.31.1 Detailed Description	124
15.32 IRenderer.hpp	125
15.33 modules/Interfaces/include/Interfaces/IWindow.hpp File Reference	125
15.33.1 Detailed Description	126
15.34 IWindow.hpp	126
15.35 plugins/Renderer/OpenGL/include/OPGL/OPGL.hpp File Reference	127
15.35.1 Detailed Description	128
15.36 OPGL.hpp	128
15.37 plugins/Renderer/Vulkan/include/VULKN/VULKN.hpp File Reference	129
15.37.1 Detailed Description	130
15.38 VULKN.hpp	130
15.39 plugins/Renderer/Vulkan/src/VULKN.cpp File Reference	130
15.40 VULKN.cpp	130
15.41 plugins/Audio/ALSA/src/entrypoint.cpp File Reference	131
15.42 entrypoint.cpp	131
15.43 plugins/Audio/Core/src/entrypoint.cpp File Reference	131
15.44 entrypoint.cpp	131
15.45 plugins/Audio/OpenAL/src/entrypoint.cpp File Reference	131
15.46 entrypoint.cpp	131
15.47 plugins/Audio/Pulse/src/entrypoint.cpp File Reference	131
15.48 entrypoint.cpp	131
15.49 plugins/Audio/XAudio2/src/entrypoint.cpp File Reference	131
15.50 entrypoint.cpp	131
15.51 plugins/Input/Cocoa/src/entrypoint.cpp File Reference	131
15.52 entrypoint.cpp	131
15.53 plugins/Input/Win32/src/entrypoint.cpp File Reference	132
15.54 entrypoint.cpp	132

15.55 plugins/Input/X11/src/entrypoint.cpp File Reference	132
15.56 entrypoint.cpp	132
15.57 plugins/Network/Asio/src/entrypoint.cpp File Reference	132
15.58 entrypoint.cpp	132
15.59 plugins/Network/Posix/src/entrypoint.cpp File Reference	132
15.60 entrypoint.cpp	132
15.61 plugins/Network/WinSock/src/entrypoint.cpp File Reference	132
15.62 entrypoint.cpp	132
15.63 plugins/Renderer/DirectX12/src/entrypoint.cpp File Reference	132
15.64 entrypoint.cpp	132
15.65 plugins/Renderer/Metal/src/entrypoint.cpp File Reference	133
15.66 entrypoint.cpp	133
15.67 plugins/Renderer/OpenGL/src/entrypoint.cpp File Reference	133
15.67.1 Function Documentation	133
15.67.1.1 entryPoint()	133
15.68 entrypoint.cpp	134
15.69 plugins/Renderer/Vulkan/src/entrypoint.cpp File Reference	134
15.69.1 Function Documentation	134
15.69.1.1 entryPoint()	134
15.70 entrypoint.cpp	135
15.71 plugins/Window/Cocoa/src/entrypoint.cpp File Reference	135
15.72 entrypoint.cpp	135
15.73 plugins/Window/GLFW/src/entrypoint.cpp File Reference	135
15.73.1 Function Documentation	136
15.73.1.1 entryPoint()	136
15.74 entrypoint.cpp	136
15.75 plugins/Window/Win32/src/entrypoint.cpp File Reference	136
15.76 entrypoint.cpp	136
15.77 plugins/Window/X11/src/entrypoint.cpp File Reference	136
15.77.1 Function Documentation	137
15.77.1.1 entryPoint()	137
15.78 entrypoint.cpp	137
15.79 plugins/Window/GLFW/include/GLFW/GLFW.hpp File Reference	138
15.79.1 Detailed Description	139
15.80 GLFW.hpp	139
15.81 plugins/Window/GLFW/src/glfw.cpp File Reference	140
15.82 glfw.cpp	140
15.83 plugins/Audio/README.md File Reference	141
15.84 plugins/Input/README.md File Reference	141
15.85 plugins/Network/README.md File Reference	141
15.86 plugins/Renderer/README.md File Reference	141
15.87 plugins/Window/README.md File Reference	141
15.88 README.md File Reference	141

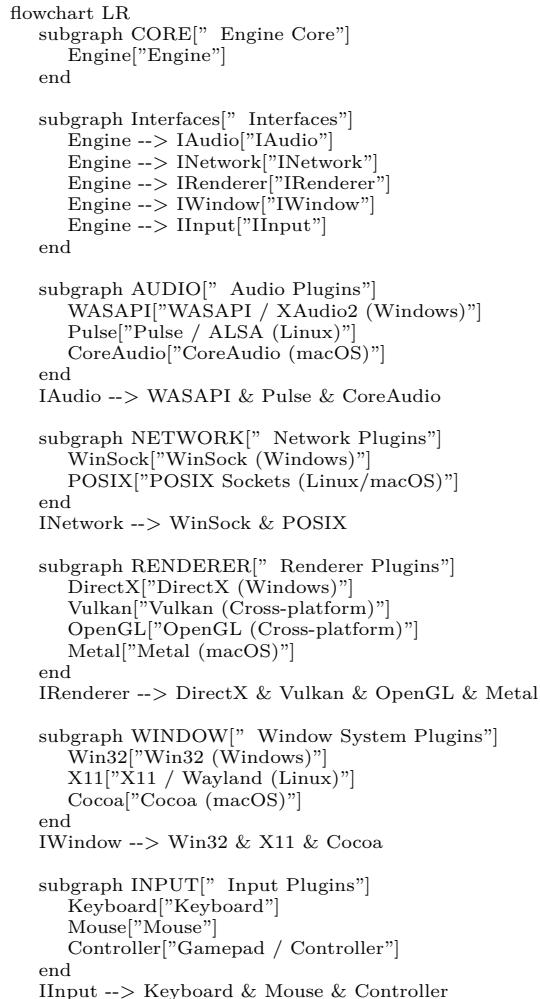
15.89 plugins/Window/X11/include/X11/X11.hpp File Reference	141
15.89.1 Detailed Description	142
15.90 X11.hpp	143
15.91 plugins/Window/X11/src/x11.cpp File Reference	143
15.92 x11.cpp	144
15.93 src/application.cpp File Reference	145
15.93.1 Function Documentation	146
15.93.1.1 loadPlugins()	146
15.94 application.cpp	146
15.95 src/argsHandler.cpp File Reference	147
15.96 argsHandler.cpp	148
15.97 src/conf.cpp File Reference	149
15.97.1 Typedef Documentation	149
15.97.1.1 json	149
15.98 conf.cpp	149
15.99 src/engine/engine.cpp File Reference	151
15.100 engine.cpp	151
15.101 src/main.cpp File Reference	152
15.101.1 Function Documentation	152
15.101.1.1 main()	152
15.102 main.cpp	153
Index	155

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.



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

README

Chapter 3

README

Chapter 4

README

Chapter 5

README

Chapter 6

README

Chapter 7

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 8

LICENSE

MIT License

Copyright (c) 2025 Masina Elliot

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Chapter 9

Namespace Index

9.1 Namespace List

Here is a list of all namespaces with brief descriptions:

cae	25
cae::Audio	30
cae::Message	31
cae::Network	31
cae::Plugins	32
cae::Plugins::Name	32
cae::Renderer	32
cae::User	33
cae::Window	33

Chapter 10

Hierarchical Index

10.1 Class Hierarchy

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

cae:: AppConfig	35
cae:: Application	37
cae:: ArgsConfig	42
cae:: ArgsHandler	43
cae:: Engine	46
cae:: EngineConfig	51
cae:: EnvConfig	54
utl:: IPlugin	
cae:: IAudio	62
cae:: IGamepad	64
cae:: IInput	65
cae:: IKeyboard	68
cae:: IMouse	70
cae:: INetwork	72
cae:: IRenderer	73
cae:: OPGL	80
cae:: VULKN	84
cae:: IWindow	76
cae:: GLFW	55
cae:: X11	89
cae:: NativeWindowHandle	79
cae:: WindowSize	88

Chapter 11

Class Index

11.1 Class List

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

cae:: AppConfig	35
cae:: Application	
Main class	37
cae:: ArgsConfig	42
cae:: ArgsHandler	
Class to handle command line arguments	43
cae:: Engine	
Engine class	46
cae:: EngineConfig	51
cae:: EnvConfig	54
cae:: GLFW	
Class for the GLFW plugin	55
cae:: IAudio	
Interface for audio	62
cae:: IGamepad	
Interface for gamepad	64
cae:: IInput	
Interface for audio	65
cae:: IKeyboard	
Interface for keyboard	68
cae:: IMouse	
Interface for mouse	70
cae:: INetwork	
Interface for network	72
cae:: IRenderer	
Interface for renderer	73
cae:: IWindow	
Interface for window	76
cae:: NativeWindowHandle	79
cae:: OPGL	
Class for the OpenGL plugin	80
cae:: VULKN	
Class for the Vulkan plugin	84
cae:: WindowSize	88
cae:: X11	
Class for the X11 plugin	89

Chapter 12

File Index

12.1 File List

Here is a list of all files with brief descriptions:

include/CAE/ Application.hpp	
This file contains the Application class declaration	97
include/CAE/ ArgsHandler.hpp	
This file contains the ArgsHandler class declaration	99
include/CAE/ Common.hpp	
This file contains	100
include/CAE/Engine/ Engine.hpp	
This file contains the engine class declaration	103
include/CAE/Generated/ Version.hpp	
modules/Interfaces/include/Interfaces/ IAudio.hpp	
This file contains the audio interface	107
modules/Interfaces/include/Interfaces/ INetwork.hpp	
This file contains the network interface	109
modules/Interfaces/include/Interfaces/ IRenderer.hpp	
This file contains the Renderer interface	124
modules/Interfaces/include/Interfaces/ IWindow.hpp	
This file contains the Window interface	125
modules/Interfaces/include/Interfaces/Input/ IGamepad.hpp	
This file contains the input gamepad interface	110
modules/Interfaces/include/Interfaces/Input/ IInput.hpp	
This file contains the input interface	112
modules/Interfaces/include/Interfaces/Input/ IKeyboard.hpp	
This file contains the input keyboard interface	114
modules/Interfaces/include/Interfaces/Input/ IMouse.hpp	
This file contains the input mouse interface	116
modules/Interfaces/include/Interfaces/Input/Key/ Gamepad.hpp	
modules/Interfaces/include/Interfaces/Input/Key/ Keyboard.hpp	
This file contains the keyboard keys	119
modules/Interfaces/include/Interfaces/Input/Key/ Mouse.hpp	
This file contains the gamepad keys	122
plugins/Audio/ALSA/src/ entrypoint.cpp	
plugins/Audio/Core/src/ entrypoint.cpp	
plugins/Audio/OpenAL/src/ entrypoint.cpp	
plugins/Audio/Pulse/src/ entrypoint.cpp	
plugins/Audio/XAudio2/src/ entrypoint.cpp	

plugins/Input/Cocoa/src/ entrypoint.cpp	131
plugins/Input/Win32/src/ entrypoint.cpp	132
plugins/Input/X11/src/ entrypoint.cpp	132
plugins/Network/Asio/src/ entrypoint.cpp	132
plugins/Network/Posix/src/ entrypoint.cpp	132
plugins/Network/WinSock/src/ entrypoint.cpp	132
plugins/Renderer/DirectX12/src/ entrypoint.cpp	132
plugins/Renderer/Metal/src/ entrypoint.cpp	133
plugins/Renderer/OpenGL/include/OPGL/ OPGL.hpp	
This file contains the OPGL class declaration	127
plugins/Renderer/OpenGL/src/ entrypoint.cpp	133
plugins/Renderer/Vulkan/include/VULKN/ VULKN.hpp	
This file contains the VULKN class declaration	129
plugins/Renderer/Vulkan/src/ entrypoint.cpp	134
plugins/Renderer/Vulkan/src/ VULKN.hpp	130
plugins/Window/Cocoa/src/ entrypoint.cpp	135
plugins/Window/GLFW/include/GLFW/ GLFW.hpp	
This file contains the GLFW class declaration	138
plugins/Window/GLFW/src/ entrypoint.cpp	135
plugins/Window/GLFW/src/glfw.cpp	140
plugins/Window/Win32/src/ entrypoint.cpp	136
plugins/Window/X11/include/X11/ X11.hpp	
This file contains the X11 class declaration	141
plugins/Window/X11/src/ entrypoint.cpp	136
plugins/Window/X11/src/x11.cpp	143
src/ application.cpp	145
src/ argsHandler.cpp	147
src/ conf.cpp	149
src/ main.cpp	152
src/engine/ engine.cpp	151

Chapter 13

Namespace Documentation

13.1 cae Namespace Reference

Namespaces

- namespace [Audio](#)
- namespace [Message](#)
- namespace [Network](#)
- namespace [Plugins](#)
- namespace [Renderer](#)
- namespace [User](#)
- namespace [Window](#)

Classes

- struct [AppConfig](#)
- class [Application](#)
 - Main class.
- struct [ArgsConfig](#)
- class [ArgsHandler](#)
 - Class to handle command line arguments.
- class [Engine](#)
 - Engine class.
- struct [EngineConfig](#)
- struct [EnvConfig](#)
- class [GLFW](#)
 - Class for the [GLFW](#) plugin.
- interface [IAudio](#)
 - Interface for audio.
- interface [IGamepad](#)
 - Interface for gamepad.
- interface [IInput](#)
 - Interface for audio.
- interface [IKeyboard](#)
 - Interface for keyboard.
- interface [IMouse](#)

- Interface for mouse.
- interface [INetwork](#)
Interface for network.
- interface [IRenderer](#)
Interface for renderer.
- interface [IWindow](#)
Interface for window.
- struct [NativeWindowHandle](#)
- class [OPGL](#)
Class for the OpenGL plugin.
- class [VULKN](#)
Class for the Vulkan plugin.
- struct [WindowSize](#)
- class [X11](#)
Class for the [X11](#) plugin.

Enumerations

- enum class [GamepadButton](#) : uint8_t {

A = 0 , B , X , Y ,

Back , Guide , Start , LThumb ,

RThumb , LShoulder , RShoulder , DPadUp ,

DPadDown , DPadLeft , DPadRight }
- enum class [GamepadAxis](#) : uint8_t {

LeftX = 0 , LeftY , RightX , RightY ,

TriggerLeft , TriggerRight }
- enum [KeyState](#) : std::uint8_t { Pressed = 0 , Released = 1 , Held = 2 , Toggled = 3 }
- enum class [KeyCode](#) : uint8_t {

A , B , C , D ,

E , F , G , H ,

I , J , K , L ,

M , N , O , P ,

Q , R , S , T ,

U , V , W , X ,

Y , Z , Num0 , Num1 ,

Num2 , Num3 , Num4 , Num5 ,

Num6 , Num7 , Num8 , Num9 ,

Escape , F1 , F2 , F3 ,

F4 , F5 , F6 , F7 ,

F8 , F9 , F10 , F11 ,

F12 , Left , Right , Up ,

Down , Home , End , PageUp ,

PageDown , Insert , Delete , Backspace ,

Tab , Enter , Space , LShift ,

RShift , LCtrl , RCtrl , LAlt ,

RAlt , LSUPER , RSUPER , Numpad0 ,

Numpad1 , Numpad2 , Numpad3 , Numpad4 ,

Numpad5 , Numpad6 , Numpad7 , Numpad8 ,

Numpad9 , NumpadAdd , NumpadSubtract , NumpadMultiply ,

NumpadDivide , CapsLock , NumLock , ScrollLock ,

Count }
- enum class [MouseButton](#) : uint8_t {

Left = 0 , Right , Middle , XButton1 ,

XButton2 , WheelUp , WheelDown }

13.1.1 Enumeration Type Documentation

13.1.1.1 GamepadAxis

enum class [cae::GamepadAxis](#) : uint8_t [strong]

Enumerator

LeftX	
LeftY	
RightX	
RightY	
TriggerLeft	
TriggerRight	

Definition at line [32](#) of file [Gamepad.hpp](#).

13.1.1.2 GamepadButton

enum class [cae::GamepadButton](#) : uint8_t [strong]

Enumerator

A	
B	
X	
Y	
Back	
Guide	
Start	
LThumb	
RThumb	
LShoulder	
RShoulder	
DPadUp	
DPadDown	
DPadLeft	
DPadRight	

Definition at line [13](#) of file [Gamepad.hpp](#).

13.1.1.3 KeyCode

enum class [cae::KeyCode](#) : uint8_t [strong]

Enumerator

A	
---	--

Enumerator

B	
C	
D	
E	
F	
G	
H	
I	
J	
K	
L	
M	
N	
O	
P	
Q	
R	
S	
T	
U	
V	
W	
X	
Y	
Z	
Num0	
Num1	
Num2	
Num3	
Num4	
Num5	
Num6	
Num7	
Num8	
Num9	
Escape	
F1	
F2	
F3	
F4	
F5	
F6	
F7	
F8	
F9	
F10	
F11	
F12	
Left	

Enumerator

Right	
Up	
Down	
Home	
End	
PageUp	
PageDown	
Insert	
Delete	
Backspace	
Tab	
Enter	
Space	
LShift	
RShift	
LCtrl	
RCtrl	
LAAlt	
RAlt	
LSuper	
RSuper	
Numpad0	
Numpad1	
Numpad2	
Numpad3	
Numpad4	
Numpad5	
Numpad6	
Numpad7	
Numpad8	
Numpad9	
NumpadAdd	
NumpadSubtract	
NumpadMultiply	
NumpadDivide	
CapsLock	
NumLock	
ScrollLock	
Count	

Definition at line 22 of file [Keyboard.hpp](#).

13.1.1.4 KeyState

```
enum cae::KeyState : std::uint8_t
```

Enumerator

Pressed	
Released	
Held	
Toggled	

Definition at line 14 of file [Keyboard.hpp](#).

13.1.1.5 MouseButton

```
enum class cae::MouseButton : uint8_t [strong]
```

Enumerator

Left	
Right	
Middle	
XButton1	
XButton2	
WheelUp	
WheelDown	

Definition at line 13 of file [Mouse.hpp](#).

13.2 cae::Audio Namespace Reference

Variables

- `constexpr auto VOLUME = 1.F`
- `constexpr auto MUTED = false`

13.2.1 Variable Documentation

13.2.1.1 MUTED

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

Definition at line 25 of file [Common.hpp](#).

13.2.1.2 VOLUME

```
auto cae::Audio::VOLUME = 1.F [inline], [constexpr]
```

Definition at line 24 of file [Common.hpp](#).

13.3 cae::Message Namespace Reference

Variables

- static constexpr std::string_view **HELP_MSG**
- static constexpr std::string_view **VERSION_MSG**

13.3.1 Variable Documentation

13.3.1.1 HELP_MSG

std::string_view cae::Message::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"
    " -c, --config <path>  Specify JSON configuration file"
```

Definition at line 30 of file [Common.hpp](#).

Referenced by [cae::ArgsHandler::ParseArgs\(\)](#).

13.3.1.2 VERSION_MSG

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

Initial value:

```
= PROJECT_NAME
    " v" PROJECT_VERSION " " BUILD_TYPE "(" GIT_TAG ", commit " GIT_COMMIT_HASH ") " __DATE__
    " __TIME__"
```

Definition at line 35 of file [Common.hpp](#).

Referenced by [cae::Application::Application\(\)](#), and [cae::ArgsHandler::ParseArgs\(\)](#).

13.4 cae::Network Namespace Reference

Variables

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

13.4.1 Variable Documentation

13.4.1.1 HOST

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

Definition at line 41 of file [Common.hpp](#).

13.4.1.2 PORT

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

Definition at line 42 of file [Common.hpp](#).

13.5 cae::Plugins Namespace Reference

Namespaces

- namespace [Name](#)

13.6 cae::Plugins::Name Namespace Reference

Variables

- constexpr auto [RENDERER_OPENGL](#) = "OpenGL"
- constexpr auto [RENDERER_VULKAN](#) = "Vulkan"
- constexpr auto [WINDOW\(GLFW](#) = "GLFW"

13.6.1 Variable Documentation

13.6.1.1 RENDERER_OPENGL

```
auto cae::Plugins::Name::RENDERER_OPENGL = "OpenGL" [inline], [constexpr]
```

Definition at line 47 of file [Common.hpp](#).

13.6.1.2 RENDERER_VULKAN

```
auto cae::Plugins::Name::RENDERER_VULKAN = "Vulkan" [inline], [constexpr]
```

Definition at line 48 of file [Common.hpp](#).

13.6.1.3 WINDOW(GLFW

```
auto cae::Plugins::Name::WINDOW(GLFW = "GLFW" [inline], [constexpr]
```

Definition at line 49 of file [Common.hpp](#).

13.7 cae::Renderer Namespace Reference

Variables

- constexpr auto [VSYNC](#) = false
- constexpr auto [FRAME_RATE_LIMIT](#) = 90

13.7.1 Variable Documentation

13.7.1.1 FRAME_RATE_LIMIT

```
auto cae::Renderer::FRAME_RATE_LIMIT = 90 [inline], [constexpr]
```

Definition at line 55 of file [Common.hpp](#).

13.7.1.2 VSYNC

```
auto cae::Renderer::VSYNC = false [inline], [constexpr]
```

Definition at line 54 of file [Common.hpp](#).

13.8 cae::User Namespace Reference

Variables

- `constexpr auto NAME = "User"`

13.8.1 Variable Documentation

13.8.1.1 NAME

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

Definition at line 60 of file [Common.hpp](#).

13.9 cae::Window Namespace Reference

Variables

- `constexpr uint16_t WIDTH = 1920`
- `constexpr uint16_t HEIGHT = 1080`
- `constexpr auto NAME = "CAE - Cross API Engine"`
- `constexpr auto FULLSCREEN = false`

13.9.1 Variable Documentation

13.9.1.1 FULLSCREEN

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

Definition at line 68 of file [Common.hpp](#).

13.9.1.2 HEIGHT

```
uint16_t cae::Window::HEIGHT = 1080 [inline], [constexpr]
```

Definition at line [66](#) of file [Common.hpp](#).

13.9.1.3 NAME

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

Definition at line [67](#) of file [Common.hpp](#).

13.9.1.4 WIDTH

```
uint16_t cae::Window::WIDTH = 1920 [inline], [constexpr]
```

Definition at line [65](#) of file [Common.hpp](#).

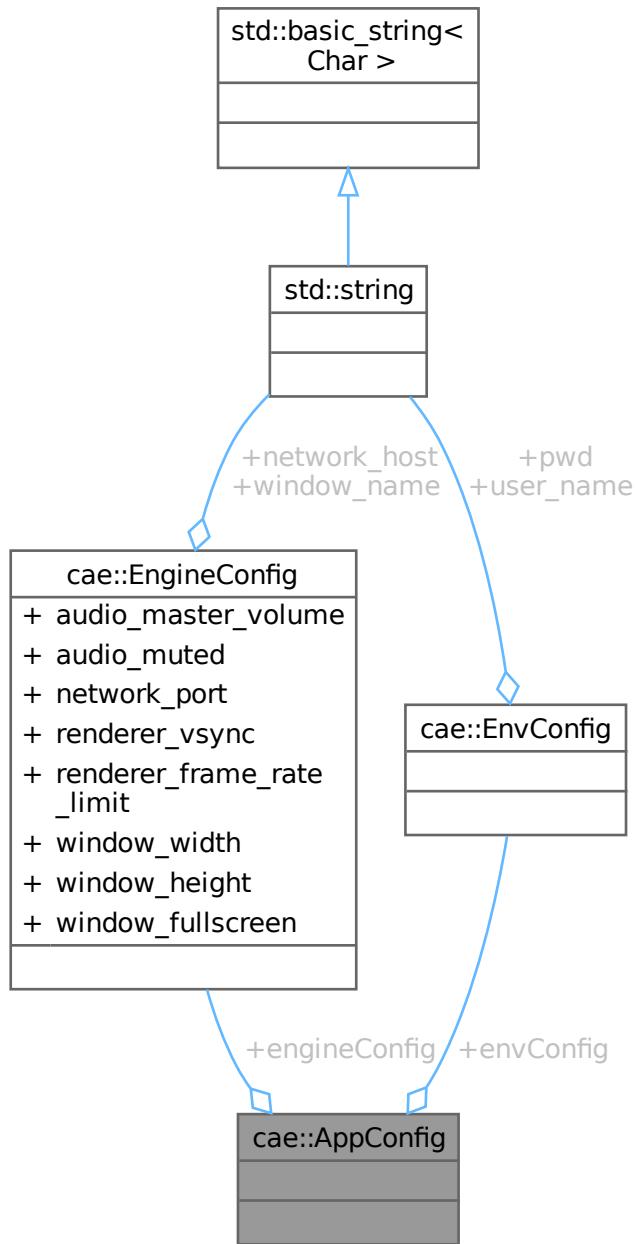
Chapter 14

Class Documentation

14.1 cae::AppConfig Struct Reference

```
#include <Application.hpp>
```

Collaboration diagram for cae::AppConfig:



Public Attributes

- `EngineConfig engineConfig`
- `EnvConfig envConfig`

14.1.1 Detailed Description

Definition at line 17 of file [Application.hpp](#).

14.1.2 Member Data Documentation

14.1.2.1 engineConfig

[EngineConfig](#) cae::AppConfig::engineConfig

Definition at line 19 of file [Application.hpp](#).

Referenced by [cae::Application::Application\(\)](#).

14.1.2.2 envConfig

[EnvConfig](#) cae::AppConfig::envConfig

Definition at line 20 of file [Application.hpp](#).

Referenced by [cae::Application::Application\(\)](#).

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

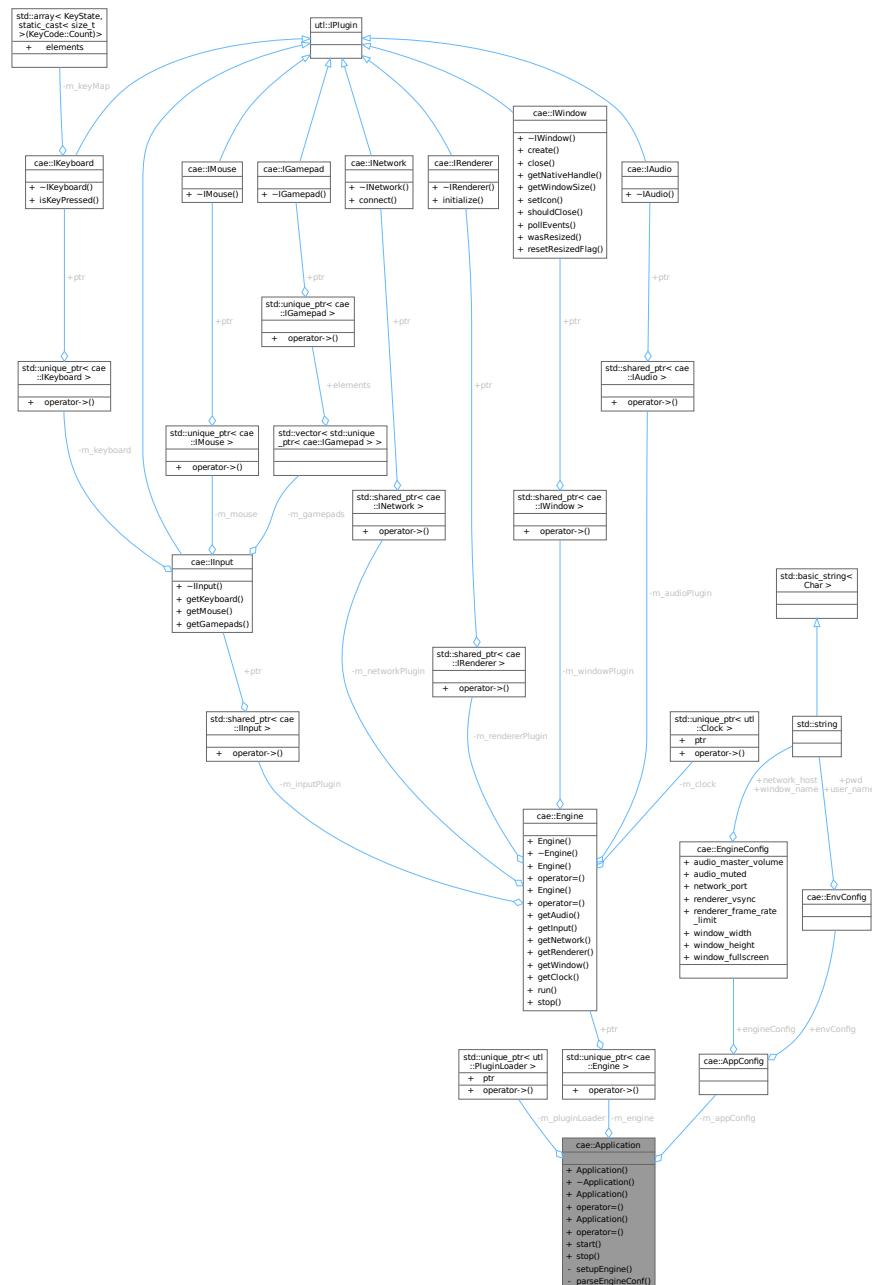
- include/CAE/[Application.hpp](#)

14.2 cae::Application Class Reference

Main class.

```
#include <Application.hpp>
```

Collaboration diagram for cae::Application:



Public Member Functions

- `Application (const ArgsConfig &argsConfig, const EnvConfig &envConfig)`
- `~Application ()=default`
- `Application (const Application &)=delete`
- `Application & operator= (const Application &)=delete`
- `Application (Application &&)=delete`
- `Application & operator= (Application &&)=delete`
- `void start () const`
- `void stop ()`

Private Member Functions

- void `setupEngine` (const std::string &rendererName, const std::string &windowName)

Static Private Member Functions

- static `EngineConfig parseEngineConf` (const std::string &path)

Private Attributes

- std::unique_ptr< utl::PluginLoader > `m_pluginLoader` = nullptr
- std::unique_ptr< Engine > `m_engine` = nullptr
- `AppConfig m_appConfig`

14.2.1 Detailed Description

Main class.

Definition at line 28 of file `Application.hpp`.

14.2.2 Constructor & Destructor Documentation

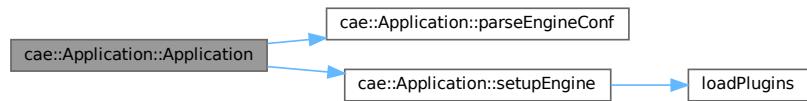
14.2.2.1 Application() [1/3]

```
cae::Application::Application (
    const ArgsConfig & argsConfig,
    const EnvConfig & envConfig)
```

Definition at line 34 of file `application.cpp`.

References `cae::ArgsConfig::config_path`, `cae::AppConfig::engineConfig`, `cae::AppConfig::envConfig`, `m_appConfig`, `parseEngineConf()`, `setupEngine()`, and `cae::Message::VERSION_MSG`.

Here is the call graph for this function:



14.2.2.2 ~Application()

```
cae::Application::~Application () [default]
```

14.2.2.3 Application() [2/3]

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

14.2.2.4 Application() [3/3]

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

14.2.3 Member Function Documentation

14.2.3.1 operator=() [1/2]

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

14.2.3.2 operator=() [2/2]

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

14.2.3.3 parseEngineConf()

```
cae::EngineConfig cae::Application::parseEngineConf (
    const std::string & path) [static], [private]
```

Definition at line 10 of file [conf.cpp](#).

References [cae::EngineConfig::audio_master_volume](#), [cae::EngineConfig::audio_muted](#), [cae::EngineConfig::network_host](#), [cae::EngineConfig::network_port](#), [cae::EngineConfig::renderer_frame_rate_limit](#), [cae::EngineConfig::renderer_vsync](#), [cae::EngineConfig::window_fullscreen](#), [cae::EngineConfig::window_height](#), [cae::EngineConfig::window_name](#), and [cae::EngineConfig::window_width](#).

Referenced by [Application\(\)](#).

Here is the caller graph for this function:



14.2.3.4 setupEngine()

```
void cae::Application::setupEngine (
    const std::string & rendererName,
    const std::string & windowName) [private]
```

Definition at line 55 of file [application.cpp](#).

References [loadPlugins\(\)](#).

Referenced by [Application\(\)](#).

Here is the call graph for this function:



Here is the caller graph for this function:



14.2.3.5 start()

```
void cae::Application::start () const
```

Definition at line 82 of file [application.cpp](#).

14.2.3.6 stop()

```
void cae::Application::stop ()
```

Definition at line 84 of file [application.cpp](#).

14.2.4 Member Data Documentation

14.2.4.1 m_appConfig

```
 AppConfig cae::Application::m_appConfig [private]
```

Definition at line 51 of file [Application.hpp](#).

Referenced by [Application\(\)](#).

14.2.4.2 m_engine

```
std::unique_ptr<Engine> cae::Application::m_engine = nullptr [private]
```

Definition at line 49 of file [Application.hpp](#).

14.2.4.3 m_pluginLoader

```
std::unique_ptr<utl::PluginLoader> cae::Application::m_pluginLoader = nullptr [private]
```

Definition at line 48 of file [Application.hpp](#).

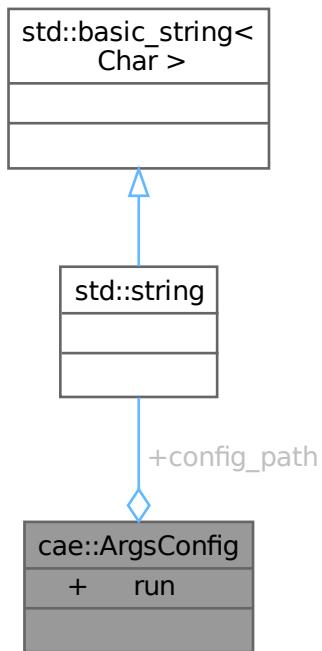
The documentation for this class was generated from the following files:

- include/CAE/[Application.hpp](#)
- src/[application.cpp](#)
- src/[conf.cpp](#)

14.3 cae::ArgsConfig Struct Reference

```
#include <ArgsHandler.hpp>
```

Collaboration diagram for cae::ArgsConfig:



Public Attributes

- bool `run` = false
- std::string `config_path`

14.3.1 Detailed Description

Definition at line 14 of file [ArgsHandler.hpp](#).

14.3.2 Member Data Documentation

14.3.2.1 config_path

std::string cae::ArgsConfig::config_path

Definition at line 17 of file [ArgsHandler.hpp](#).

Referenced by [cae::Application::Application\(\)](#), and [cae::ArgsHandler::ParseArgs\(\)](#).

14.3.2.2 run

bool cae::ArgsConfig::run = false

Definition at line 16 of file [ArgsHandler.hpp](#).

Referenced by [main\(\)](#), and [cae::ArgsHandler::ParseArgs\(\)](#).

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

- include/CAE/[ArgsHandler.hpp](#)

14.4 cae::ArgsHandler Class Reference

Class to handle command line arguments.

```
#include <ArgsHandler.hpp>
```

Collaboration diagram for cae::ArgsHandler:

cae::ArgsHandler
+ ArgsHandler() + ~ArgsHandler() + ArgsHandler() + operator=(+ ArgsHandler() + operator=(+ ParseArgs() + ParseEnv()

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)`

14.4.1 Detailed Description

Class to handle command line arguments.

Definition at line 30 of file [ArgsHandler.hpp](#).

14.4.2 Constructor & Destructor Documentation

14.4.2.1 ArgsHandler() [1/3]

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

14.4.2.2 ~ArgsHandler()

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

14.4.2.3 ArgsHandler() [2/3]

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

14.4.2.4 ArgsHandler() [3/3]

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

14.4.3 Member Function Documentation

14.4.3.1 operator=() [1/2]

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

14.4.3.2 operator=() [2/2]

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

14.4.3.3 ParseArgs()

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

Definition at line 8 of file [argsHandler.cpp](#).

References [cae::ArgsConfig::config_path](#), [cae::Message::HELP_MSG](#), [cae::ArgsConfig::run](#), and [cae::Message::VERSION_MSG](#).

Referenced by [main\(\)](#).

Here is the caller graph for this function:



14.4.3.4 ParseEnv()

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

Definition at line 52 of file [argsHandler.cpp](#).

References [cae::EnvConfig::pwd](#), and [cae::EnvConfig::user_name](#).

Referenced by [main\(\)](#).

Here is the caller graph for this function:



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

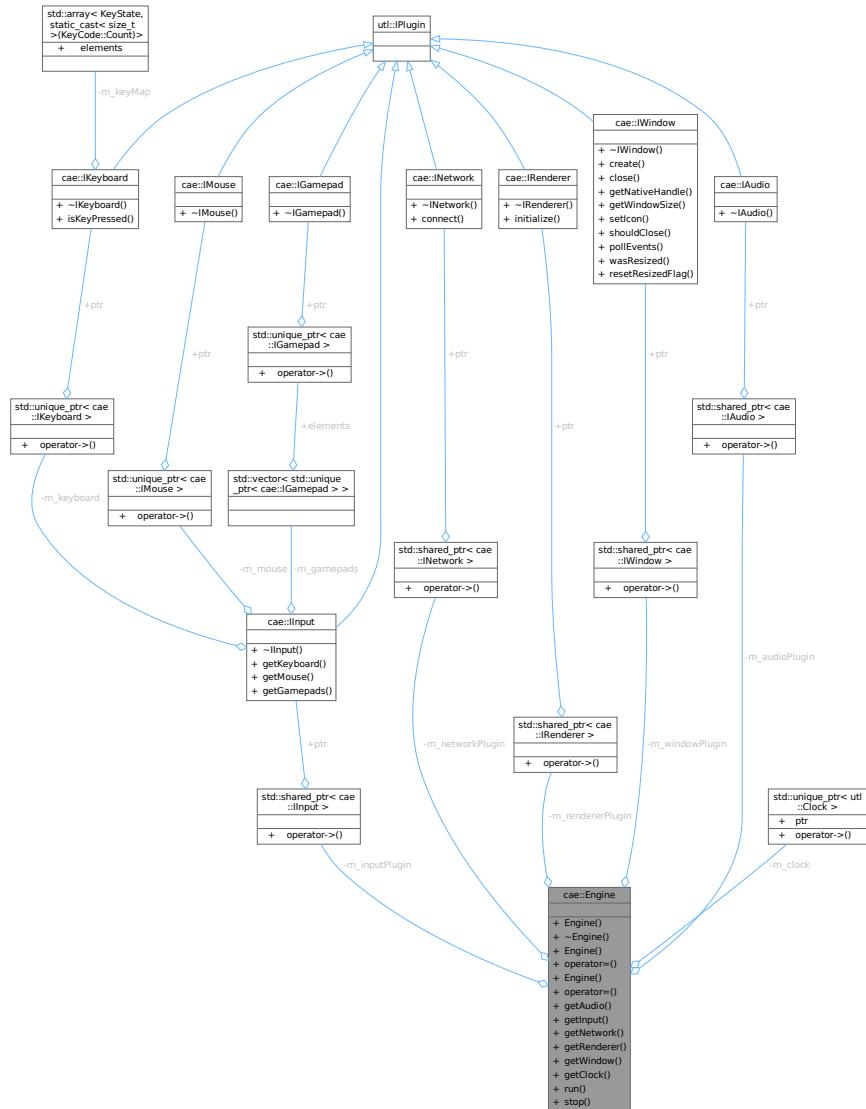
- include/CAE/[ArgsHandler.hpp](#)
- src/[argsHandler.cpp](#)

14.5 cae::Engine Class Reference

[Engine](#) class.

```
#include <Engine.hpp>
```

Collaboration diagram for cae::Engine:



Public Member Functions

- `Engine (const EngineConfig &config, const std::function< std::shared_ptr< IAudio >()> &audioFactory, const std::function< std::shared_ptr< IInput >()> &inputFactory, const std::function< std::shared_ptr< INetwork >()> &networkFactory, const std::function< std::shared_ptr< IRenderer >()> &rendererFactory, const std::function< std::shared_ptr< IWindow >()> &windowFactory)`
- `~Engine ()=default`
- `Engine (const Engine &)=delete`

- `Engine & operator= (const Engine &)=delete`
- `Engine (Engine &&)=delete`
- `Engine & operator= (Engine &&)=delete`
- `const std::shared_ptr< IAudio > & getAudio () const`
- `const std::shared_ptr< IInput > & getInput () const`
- `const std::shared_ptr< INetwork > & getNetwork () const`
- `const std::shared_ptr< IRenderer > & getRenderer () const`
- `const std::shared_ptr< IWindow > & getWindow () const`
- `const std::unique_ptr< utl::Clock > & getClock ()`
- `void run () const`
- `void stop ()`

Private Attributes

- `std::shared_ptr< IAudio > m_audioPlugin = nullptr`
- `std::shared_ptr< IInput > m_inputPlugin = nullptr`
- `std::shared_ptr< INetwork > m_networkPlugin = nullptr`
- `std::shared_ptr< IRenderer > m_rendererPlugin = nullptr`
- `std::shared_ptr< IWindow > m_windowPlugin = nullptr`
- `std::unique_ptr< utl::Clock > m_clock = nullptr`

14.5.1 Detailed Description

[Engine](#) class.

Definition at line 45 of file [Engine.hpp](#).

14.5.2 Constructor & Destructor Documentation

14.5.2.1 Engine() [1/3]

```
cae::Engine::Engine (
    const EngineConfig & config,
    const std::function< std::shared_ptr< IAudio >()> & audioFactory,
    const std::function< std::shared_ptr< IInput >()> & inputFactory,
    const std::function< std::shared_ptr< INetwork >()> & networkFactory,
    const std::function< std::shared_ptr< IRenderer >()> & rendererFactory,
    const std::function< std::shared_ptr< IWindow >()> & windowFactory)
```

Definition at line 5 of file [engine.cpp](#).

References `cae::EngineConfig::audio_master_volume`, `cae::EngineConfig::audio_muted`, `m_windowPlugin`, `cae::EngineConfig::network_host`, `cae::EngineConfig::network_port`, `cae::EngineConfig::renderer_frame_rate_limit`, `cae::EngineConfig::renderer_vsync`, `cae::EngineConfig::window_fullscreen`, `cae::EngineConfig::window_height`, `cae::EngineConfig::window_name`, and `cae::EngineConfig::window_width`.

14.5.2.2 ~Engine()

```
cae::Engine::~Engine () [default]
```

14.5.2.3 Engine() [2/3]

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

14.5.2.4 Engine() [3/3]

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

14.5.3 Member Function Documentation

14.5.3.1 getAudio()

```
const std::shared_ptr< IAudio > & cae::Engine::getAudio () const [inline], [nodiscard]
```

Definition at line [61](#) of file [Engine.hpp](#).

References [m_audioPlugin](#).

14.5.3.2 getClock()

```
const std::unique_ptr< utl::Clock > & cae::Engine::getClock () [inline], [nodiscard]
```

Definition at line [67](#) of file [Engine.hpp](#).

References [m_clock](#).

14.5.3.3 getInput()

```
const std::shared_ptr< IInput > & cae::Engine::getInput () const [inline], [nodiscard]
```

Definition at line [62](#) of file [Engine.hpp](#).

References [m_inputPlugin](#).

14.5.3.4 getNetwork()

```
const std::shared_ptr< INetwork > & cae::Engine::getNetwork () const [inline], [nodiscard]
```

Definition at line [63](#) of file [Engine.hpp](#).

References [m_networkPlugin](#).

14.5.3.5 getRenderer()

```
const std::shared_ptr<IRenderer> & cae::Engine::getRenderer () const [inline], [nodiscard]
```

Definition at line 64 of file [Engine.hpp](#).

References [m_rendererPlugin](#).

14.5.3.6 getWindow()

```
const std::shared_ptr<IWindow> & cae::Engine::getWindow () const [inline], [nodiscard]
```

Definition at line 65 of file [Engine.hpp](#).

References [m_windowPlugin](#).

14.5.3.7 operator=() [1/2]

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

14.5.3.8 operator=() [2/2]

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

14.5.3.9 run()

```
void cae::Engine::run () const
```

Definition at line 29 of file [engine.cpp](#).

14.5.3.10 stop()

```
void cae::Engine::stop ()
```

Definition at line 37 of file [engine.cpp](#).

14.5.4 Member Data Documentation

14.5.4.1 m_audioPlugin

```
std::shared_ptr<IAudio> cae::Engine::m_audioPlugin = nullptr [private]
```

Definition at line 73 of file [Engine.hpp](#).

Referenced by [getAudio\(\)](#).

14.5.4.2 m_clock

```
std::unique_ptr<utl::Clock> cae::Engine::m_clock = nullptr [private]
```

Definition at line [79](#) of file [Engine.hpp](#).

Referenced by [getClock\(\)](#).

14.5.4.3 m_inputPlugin

```
std::shared_ptr<IInput> cae::Engine::m_inputPlugin = nullptr [private]
```

Definition at line [74](#) of file [Engine.hpp](#).

Referenced by [getInput\(\)](#).

14.5.4.4 m_networkPlugin

```
std::shared_ptr<INetwork> cae::Engine::m_networkPlugin = nullptr [private]
```

Definition at line [75](#) of file [Engine.hpp](#).

Referenced by [getNetwork\(\)](#).

14.5.4.5 m_rendererPlugin

```
std::shared_ptr<IRenderer> cae::Engine::m_rendererPlugin = nullptr [private]
```

Definition at line [76](#) of file [Engine.hpp](#).

Referenced by [getRenderer\(\)](#).

14.5.4.6 m_windowPlugin

```
std::shared_ptr<IWindow> cae::Engine::m_windowPlugin = nullptr [private]
```

Definition at line [77](#) of file [Engine.hpp](#).

Referenced by [Engine\(\)](#), and [getWindow\(\)](#).

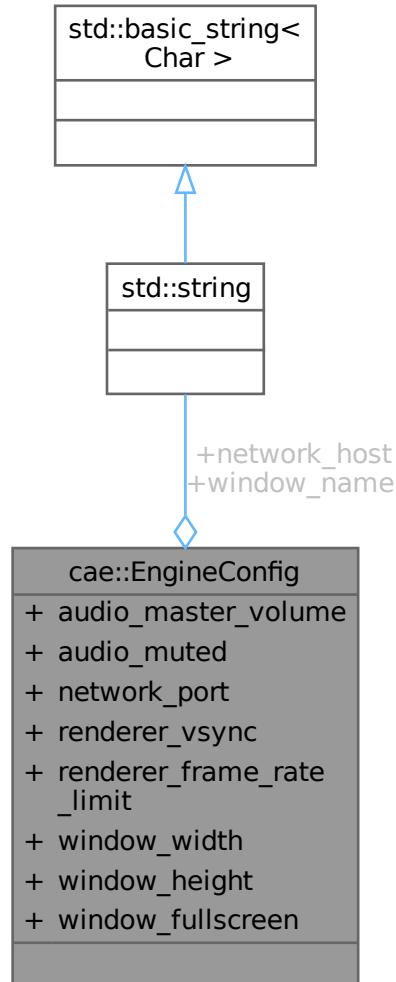
The documentation for this class was generated from the following files:

- include/CAE/Engine/[Engine.hpp](#)
- src/engine/[engine.cpp](#)

14.6 cae::EngineConfig Struct Reference

```
#include <Engine.hpp>
```

Collaboration diagram for cae::EngineConfig:



Public Attributes

- float `audio_master_volume` = `Audio::VOLUME`
- bool `audio_muted` = `Audio::MUTED`
- std::string `network_host` = `Network::HOST`
- uint16_t `network_port` = `Network::PORT`
- bool `renderer_vsync` = `Renderer::VSYNC`
- uint16_t `renderer_frame_rate_limit` = `Renderer::FRAME_RATE_LIMIT`
- uint16_t `window_width` = `Window::WIDTH`
- uint16_t `window_height` = `Window::HEIGHT`
- bool `windowFullscreen` = `Window::FULLSCREEN`
- std::string `window_name` = `Window::NAME`

14.6.1 Detailed Description

Definition at line 23 of file [Engine.hpp](#).

14.6.2 Member Data Documentation

14.6.2.1 audio_master_volume

```
float cae::EngineConfig::audio_master_volume = Audio::VOLUME
```

Definition at line 25 of file [Engine.hpp](#).

Referenced by [cae::Engine::Engine\(\)](#), and [cae::Application::parseEngineConf\(\)](#).

14.6.2.2 audio_muted

```
bool cae::EngineConfig::audio_muted = Audio::MUTED
```

Definition at line 26 of file [Engine.hpp](#).

Referenced by [cae::Engine::Engine\(\)](#), and [cae::Application::parseEngineConf\(\)](#).

14.6.2.3 network_host

```
std::string cae::EngineConfig::network_host = Network::HOST
```

Definition at line 28 of file [Engine.hpp](#).

Referenced by [cae::Engine::Engine\(\)](#), and [cae::Application::parseEngineConf\(\)](#).

14.6.2.4 network_port

```
uint16_t cae::EngineConfig::network_port = Network::PORT
```

Definition at line 29 of file [Engine.hpp](#).

Referenced by [cae::Engine::Engine\(\)](#), and [cae::Application::parseEngineConf\(\)](#).

14.6.2.5 renderer_frame_rate_limit

```
uint16_t cae::EngineConfig::renderer_frame_rate_limit = Renderer::FRAME\_RATE\_LIMIT
```

Definition at line 32 of file [Engine.hpp](#).

Referenced by [cae::Engine::Engine\(\)](#), and [cae::Application::parseEngineConf\(\)](#).

14.6.2.6 renderer_vsync

```
bool cae::EngineConfig::renderer_vsync = Renderer::VSYNC
```

Definition at line 31 of file [Engine.hpp](#).

Referenced by [cae::Engine::Engine\(\)](#), and [cae::Application::parseEngineConf\(\)](#).

14.6.2.7 window_fullscreen

```
bool cae::EngineConfig::windowFullscreen = Window::FULLSCREEN
```

Definition at line 36 of file [Engine.hpp](#).

Referenced by [cae::Engine::Engine\(\)](#), and [cae::Application::parseEngineConf\(\)](#).

14.6.2.8 window_height

```
uint16_t cae::EngineConfig::windowHeight = Window::HEIGHT
```

Definition at line 35 of file [Engine.hpp](#).

Referenced by [cae::Engine::Engine\(\)](#), and [cae::Application::parseEngineConf\(\)](#).

14.6.2.9 window_name

```
std::string cae::EngineConfig::windowName = Window::NAME
```

Definition at line 37 of file [Engine.hpp](#).

Referenced by [cae::Engine::Engine\(\)](#), and [cae::Application::parseEngineConf\(\)](#).

14.6.2.10 window_width

```
uint16_t cae::EngineConfig::windowWidth = Window::WIDTH
```

Definition at line 34 of file [Engine.hpp](#).

Referenced by [cae::Engine::Engine\(\)](#), and [cae::Application::parseEngineConf\(\)](#).

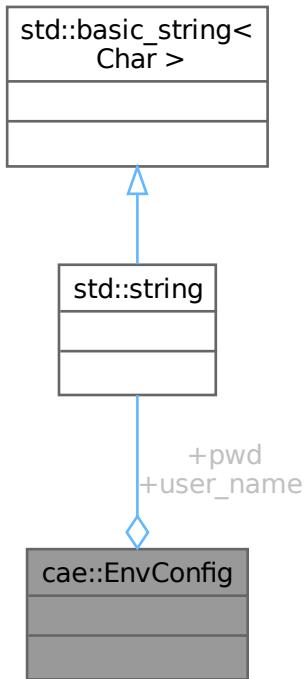
The documentation for this struct was generated from the following file:

- include/CAE/Engine/[Engine.hpp](#)

14.7 cae::EnvConfig Struct Reference

```
#include <ArgsHandler.hpp>
```

Collaboration diagram for cae::EnvConfig:



Public Attributes

- `std::string user_name`
- `std::string pwd`

14.7.1 Detailed Description

Definition at line 19 of file [ArgsHandler.hpp](#).

14.7.2 Member Data Documentation

14.7.2.1 `pwd`

`std::string cae::EnvConfig::pwd`

Definition at line 22 of file [ArgsHandler.hpp](#).

Referenced by [cae::ArgsHandler::ParseEnv\(\)](#).

14.7.2.2 user_name

std::string cae::EnvConfig::user_name

Definition at line 21 of file [ArgsHandler.hpp](#).

Referenced by [cae::ArgsHandler::ParseEnv\(\)](#).

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

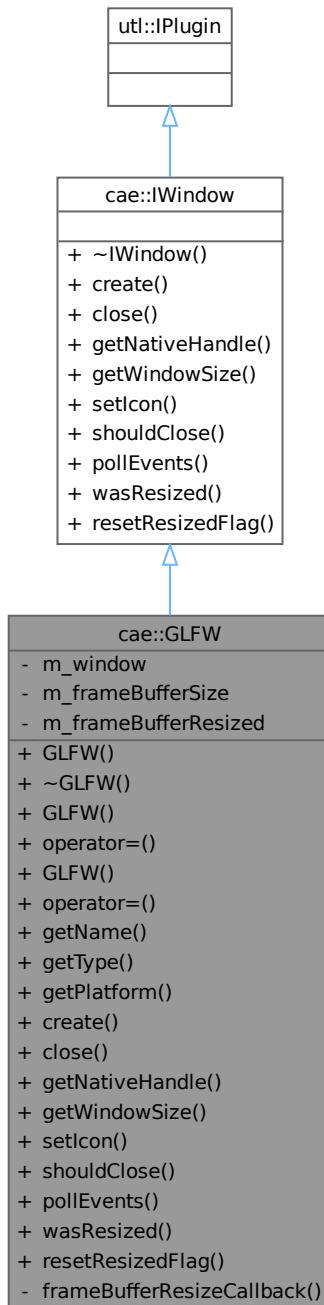
- include/CAE/[ArgsHandler.hpp](#)

14.8 cae::GLFW Class Reference

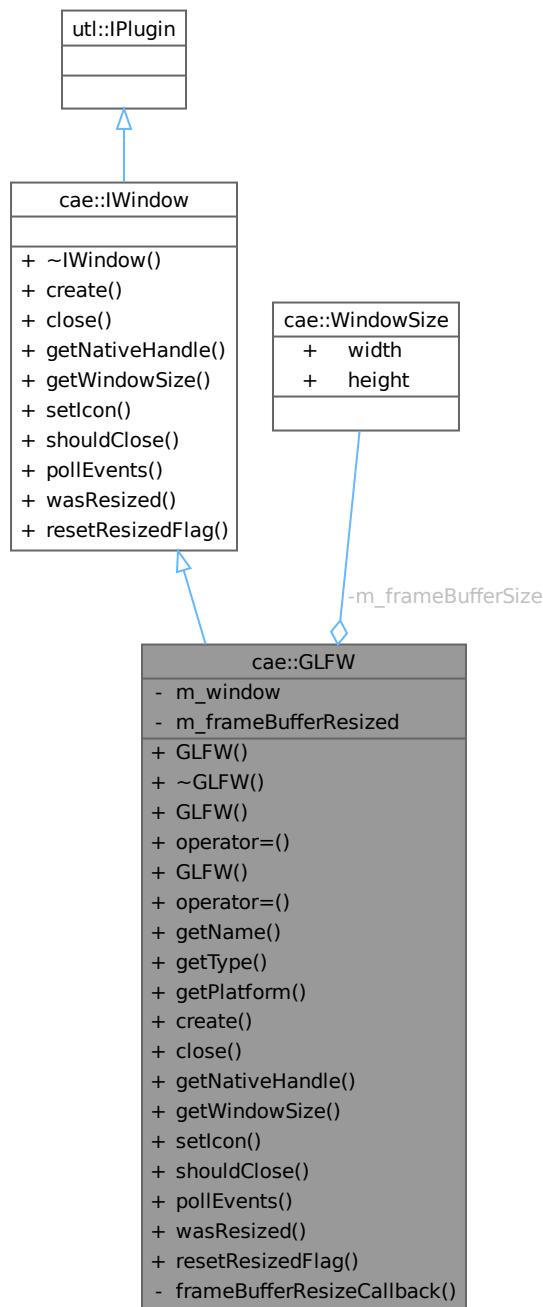
Class for the [GLFW](#) plugin.

```
#include <GLFW.hpp>
```

Inheritance diagram for cae::GLFW:



Collaboration diagram for cae::GLFW:



Public Member Functions

- **GLFW ()**=default
- **~GLFW ()** override=default
- **GLFW (const GLFW &)**=delete
- **GLFW & operator= (const GLFW &)**=delete
- **GLFW (GLFW &&)**=delete

- `GLFW & operator= (GLFW &&) = delete`
- `std::string getName () const override`
- `utl::PluginType getType () const override`
- `utl::PluginPlatform getPlatform () const override`
- `bool create (const std::string &name, WindowSize size) override`
- `void close () override`
- `NativeWindowHandle getNativeHandle () const override`
- `WindowSize getWindowSize () const override`
- `bool setIcon (const std::string &path) const override`
- `bool shouldClose () const override`
- `void pollEvents () override`
- `bool wasResized () const override`
- `void resetResizedFlag () override`

Public Member Functions inherited from `cae::IWindow`

- `~IWindow () override = default`

Static Private Member Functions

- `static void frameBufferResizeCallback (GLFWwindow *window, int width, int height)`

Private Attributes

- `GLFWwindow * m_window = nullptr`
- `WindowSize m_frameBufferSize`
- `bool m_frameBufferResized = false`

14.8.1 Detailed Description

Class for the `GLFW` plugin.

Definition at line 30 of file `GLFW.hpp`.

14.8.2 Constructor & Destructor Documentation

14.8.2.1 `GLFW()` [1/3]

`cae::GLFW::GLFW () [default]`

14.8.2.2 `~GLFW()`

`cae::GLFW::~GLFW () [override], [default]`

14.8.2.3 `GLFW()` [2/3]

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

14.8.2.4 GLFW() [3/3]

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

14.8.3 Member Function Documentation

14.8.3.1 close()

```
void cae::GLFW::close () [override], [virtual]
```

Implements [cae::IWindow](#).

Definition at line [37](#) of file [glfw.cpp](#).

14.8.3.2 create()

```
bool cae::GLFW::create (
    const std::string & name,
    WindowSize size) [override], [virtual]
```

Implements [cae::IWindow](#).

Definition at line [13](#) of file [glfw.cpp](#).

References [cae::WindowSize::height](#), and [cae::WindowSize::width](#).

14.8.3.3 frameBufferResizeCallback()

```
void cae::GLFW::frameBufferResizeCallback (
    GLFWwindow * window,
    int width,
    int height) [static], [private]
```

Definition at line [6](#) of file [glfw.cpp](#).

References [m_frameBufferResized](#).

14.8.3.4 getName()

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

Definition at line [42](#) of file [GLFW.hpp](#).

14.8.3.5 getNativeHandle()

```
cae::NativeWindowHandle cae::GLFW::getNativeHandle () const [nodiscard], [override], [virtual]
```

Implements [cae::IWindow](#).

Definition at line [55](#) of file [glfw.cpp](#).

References [cae::NativeWindowHandle::window](#).

14.8.3.6 getPlatform()

```
utl::PluginPlatform cae::GLFW::getPlatform () const [inline], [nodiscard], [override]
```

Definition at line 44 of file [GLFW.hpp](#).

14.8.3.7 getType()

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

Definition at line 43 of file [GLFW.hpp](#).

14.8.3.8 getWindowSize()

```
cae::WindowSize cae::GLFW::getWindowSize () const [nodiscard], [override], [virtual]
```

Implements [cae::IWindow](#).

Definition at line 47 of file [glfw.cpp](#).

14.8.3.9 operator=() [1/2]

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

14.8.3.10 operator=() [2/2]

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

14.8.3.11 pollEvents()

```
void cae::GLFW::pollEvents () [inline], [override], [virtual]
```

Implements [cae::IWindow](#).

Definition at line 55 of file [GLFW.hpp](#).

14.8.3.12 resetResizedFlag()

```
void cae::GLFW::resetResizedFlag () [inline], [override], [virtual]
```

Implements [cae::IWindow](#).

Definition at line 58 of file [GLFW.hpp](#).

References [m_frameBufferResized](#).

14.8.3.13 setIcon()

```
bool cae::GLFW::setIcon (
    const std::string & path) const [nodiscard], [override], [virtual]
```

Implements [cae::IWindow](#).

Definition at line [71](#) of file [glfw.cpp](#).

14.8.3.14 shouldClose()

```
bool cae::GLFW::shouldClose () const [inline], [nodiscard], [override], [virtual]
```

Implements [cae::IWindow](#).

Definition at line [54](#) of file [GLFW.hpp](#).

References [m_window](#).

14.8.3.15 wasResized()

```
bool cae::GLFW::wasResized () const [inline], [override], [virtual]
```

Implements [cae::IWindow](#).

Definition at line [57](#) of file [GLFW.hpp](#).

References [m_frameBufferResized](#).

14.8.4 Member Data Documentation

14.8.4.1 m_frameBufferResized

```
bool cae::GLFW::m_frameBufferResized = false [private]
```

Definition at line [65](#) of file [GLFW.hpp](#).

Referenced by [frameBufferResizeCallback\(\)](#), [resetResizedFlag\(\)](#), and [wasResized\(\)](#).

14.8.4.2 m_frameBufferSize

```
WindowSize cae::GLFW::m_frameBufferSize [private]
```

Definition at line [64](#) of file [GLFW.hpp](#).

14.8.4.3 m_window

GLFWwindow* cae::GLFW::m_window = nullptr [private]

Definition at line 63 of file [GLFW.hpp](#).

Referenced by [shouldClose\(\)](#).

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

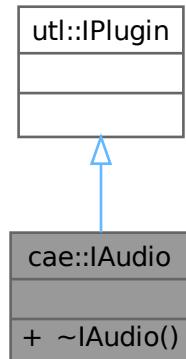
- [plugins/Window/GLFW/include/GLFW/GLFW.hpp](#)
- [plugins/Window/GLFW/src/glfw.cpp](#)

14.9 cae::IAudio Interface Reference

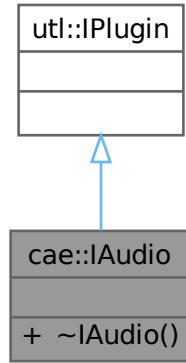
Interface for audio.

```
#include <IAudio.hpp>
```

Inheritance diagram for cae::IAudio:



Collaboration diagram for cae::IAudio:



Public Member Functions

- `~IAudio ()` override=default

14.9.1 Detailed Description

Interface for audio.

Definition at line 19 of file [IAudio.hpp](#).

14.9.2 Constructor & Destructor Documentation

14.9.2.1 `~IAudio()`

`cae::IAudio::~IAudio () [override], [default]`

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

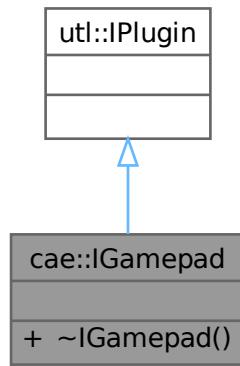
- `modules/Interfaces/include/Interfaces/IAudio.hpp`

14.10 cae::IGamepad Interface Reference

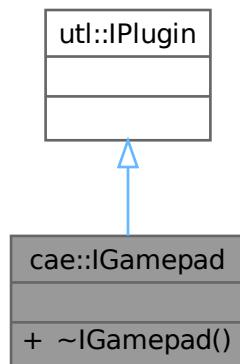
Interface for gamepad.

```
#include <IGamepad.hpp>
```

Inheritance diagram for cae::IGamepad:



Collaboration diagram for cae::IGamepad:



Public Member Functions

- `~IGamepad` () override=default

14.10.1 Detailed Description

Interface for gamepad.

Definition at line 19 of file [IGamepad.hpp](#).

14.10.2 Constructor & Destructor Documentation

14.10.2.1 ~IGamepad()

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

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

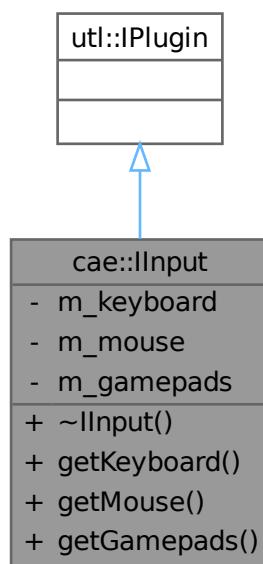
- modules/Interfaces/include/Interfaces/Input/[IGamepad.hpp](#)

14.11 cae::IInput Interface Reference

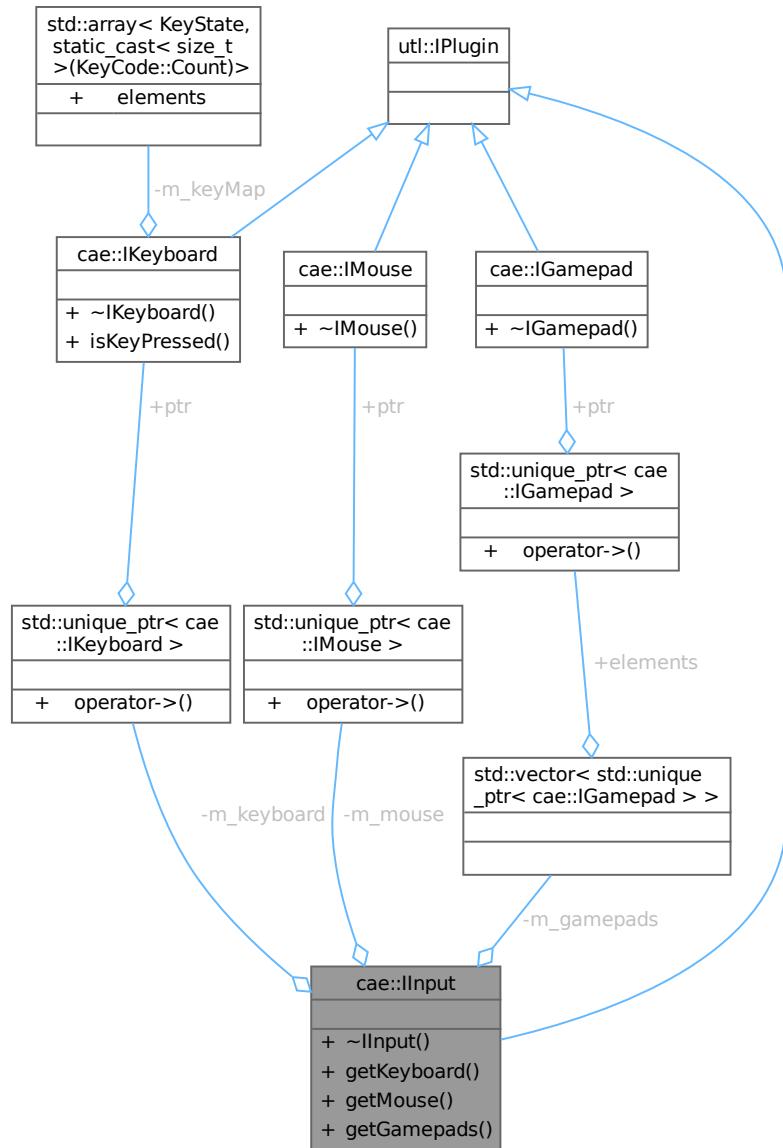
Interface for audio.

```
#include <IInput.hpp>
```

Inheritance diagram for cae::IInput:



Collaboration diagram for cae::IInput:



Public Member Functions

- `~IInput ()` override=default
- `virtual const std::unique_ptr< IKeyboard > & getKeyboard () const =0`
- `virtual const std::unique_ptr< IMouse > & getMouse () const =0`
- `virtual const std::vector< std::unique_ptr< IGamepad > > & getGamepads () const =0`

Private Attributes

- `std::unique_ptr< IKeyboard > m_keyboard`
- `std::unique_ptr< IMouse > m_mouse`
- `std::vector< std::unique_ptr< IGamepad > > m_gamepads`

14.11.1 Detailed Description

Interface for audio.

Definition at line 25 of file [IInput.hpp](#).

14.11.2 Constructor & Destructor Documentation

14.11.2.1 ~IInput()

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

14.11.3 Member Function Documentation

14.11.3.1 getGamepads()

virtual const std::vector< std::unique_ptr< [IGamepad](#) > > & cae::IInput::getGamepads () const [pure virtual]

14.11.3.2 getKeyboard()

virtual const std::unique_ptr< [IKeyboard](#) > & cae::IInput::getKeyboard () const [pure virtual]

14.11.3.3 getMouse()

virtual const std::unique_ptr< [IMouse](#) > & cae::IInput::getMouse () const [pure virtual]

14.11.4 Member Data Documentation

14.11.4.1 m_gamepads

std::vector<std::unique_ptr<[IGamepad](#)>> cae::IInput::m_gamepads [private]

Definition at line 38 of file [IInput.hpp](#).

14.11.4.2 m_keyboard

std::unique_ptr<[IKeyboard](#)> cae::IInput::m_keyboard [private]

Definition at line 36 of file [IInput.hpp](#).

14.11.4.3 m_mouse

```
std::unique_ptr<IMouse> cae::IInput::m_mouse [private]
```

Definition at line 37 of file [IInput.hpp](#).

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

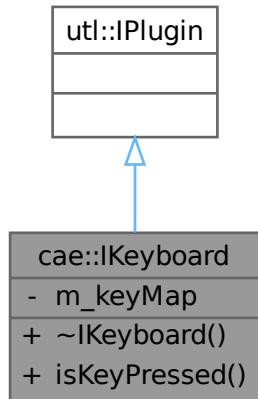
- [modules/Interfaces/include/Interfaces/Input/IInput.hpp](#)

14.12 cae::IKeyboard Interface Reference

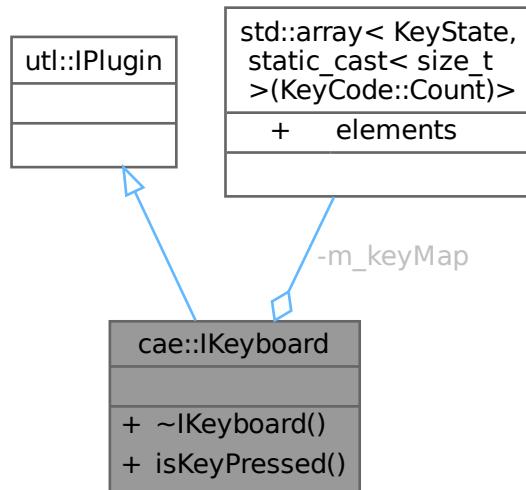
Interface for keyboard.

```
#include <IKeyboard.hpp>
```

Inheritance diagram for cae::IKeyboard:



Collaboration diagram for cae::IKeyboard:



Public Member Functions

- `~IKeyboard ()` override=default
- virtual bool `isKeyPressed (KeyCode keyCode) const =0`

Private Attributes

- `std::array< KeyState, static_cast< size_t >(KeyCode::Count)> m_keyMap {}`

14.12.1 Detailed Description

Interface for keyboard.

Definition at line 23 of file [IKeyboard.hpp](#).

14.12.2 Constructor & Destructor Documentation

14.12.2.1 ~IKeyboard()

`cae::IKeyboard::~IKeyboard () [override], [default]`

14.12.3 Member Function Documentation

14.12.3.1 isKeyPressed()

```
virtual bool cae::IKeyboard::isKeyPressed (
    KeyCode keyCode) const [pure virtual]
```

14.12.4 Member Data Documentation

14.12.4.1 m_keyMap

```
std::array<KeyState, static_cast<size_t>(KeyCode::Count)> cae::IKeyboard::m_keyMap {} [private]
```

Definition at line 32 of file [IKeyboard.hpp](#).

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

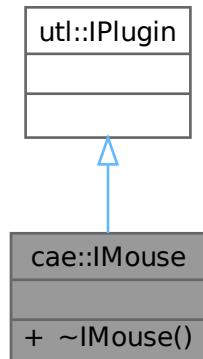
- [modules/Interfaces/include/Interfaces/Input/IKeyboard.hpp](#)

14.13 cae::IMouse Interface Reference

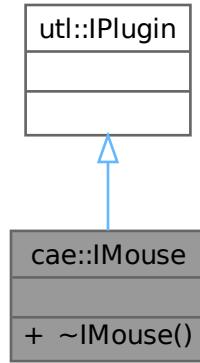
Interface for mouse.

```
#include <IMouse.hpp>
```

Inheritance diagram for cae::IMouse:



Collaboration diagram for cae::IMouse:



Public Member Functions

- `~IMouse ()` override=default

14.13.1 Detailed Description

Interface for mouse.

Definition at line 19 of file [IMouse.hpp](#).

14.13.2 Constructor & Destructor Documentation

14.13.2.1 ~IMouse()

`cae::IMouse::~IMouse () [override], [default]`

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

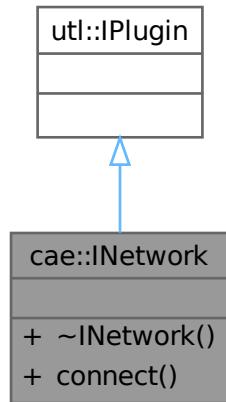
- [modules/Interfaces/include/Interfaces/Input/IMouse.hpp](#)

14.14 cae::INetwork Interface Reference

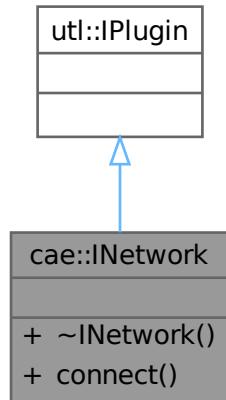
Interface for network.

```
#include <INetwork.hpp>
```

Inheritance diagram for cae::INetwork:



Collaboration diagram for cae::INetwork:



Public Member Functions

- `~INetwork ()` override=default
- virtual bool `connect (const std::string &host, uint16_t port)=0`

14.14.1 Detailed Description

Interface for network.

Definition at line 19 of file [INetwork.hpp](#).

14.14.2 Constructor & Destructor Documentation

14.14.2.1 ~INetwork()

```
cae::INetwork::~INetwork () [override], [default]
```

14.14.3 Member Function Documentation

14.14.3.1 connect()

```
virtual bool cae::INetwork::connect (
    const std::string & host,
    uint16_t port) [pure virtual]
```

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

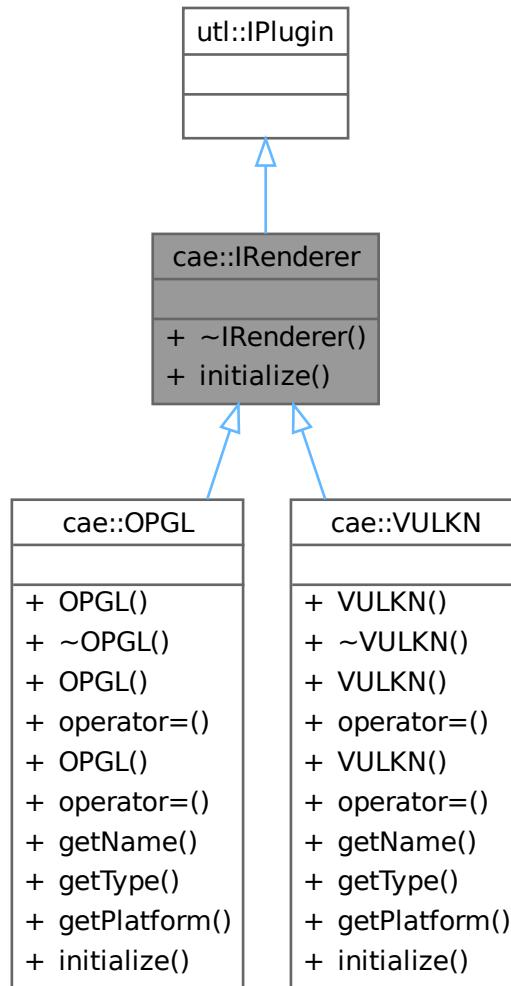
- modules/Interfaces/include/Interfaces/[INetwork.hpp](#)

14.15 cae::IRenderer Interface Reference

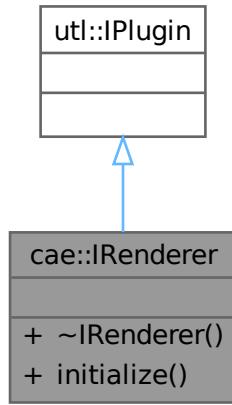
Interface for renderer.

```
#include <IRenderer.hpp>
```

Inheritance diagram for cae::IRenderer:



Collaboration diagram for cae::IRenderer:



Public Member Functions

- `~IRenderer ()` override=default
- virtual void `initialize (void *nativeWindowHandle)=0`

14.15.1 Detailed Description

Interface for renderer.

Definition at line 19 of file [IRenderer.hpp](#).

14.15.2 Constructor & Destructor Documentation

14.15.2.1 `~IRenderer()`

`cae::IRenderer::~IRenderer ()` [override], [default]

14.15.3 Member Function Documentation

14.15.3.1 `initialize()`

```

virtual void cae::IRenderer::initialize (
    void * nativeWindowHandle) [pure virtual]
  
```

Implemented in [cae::OPGL](#), and [cae::VULKN](#).

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

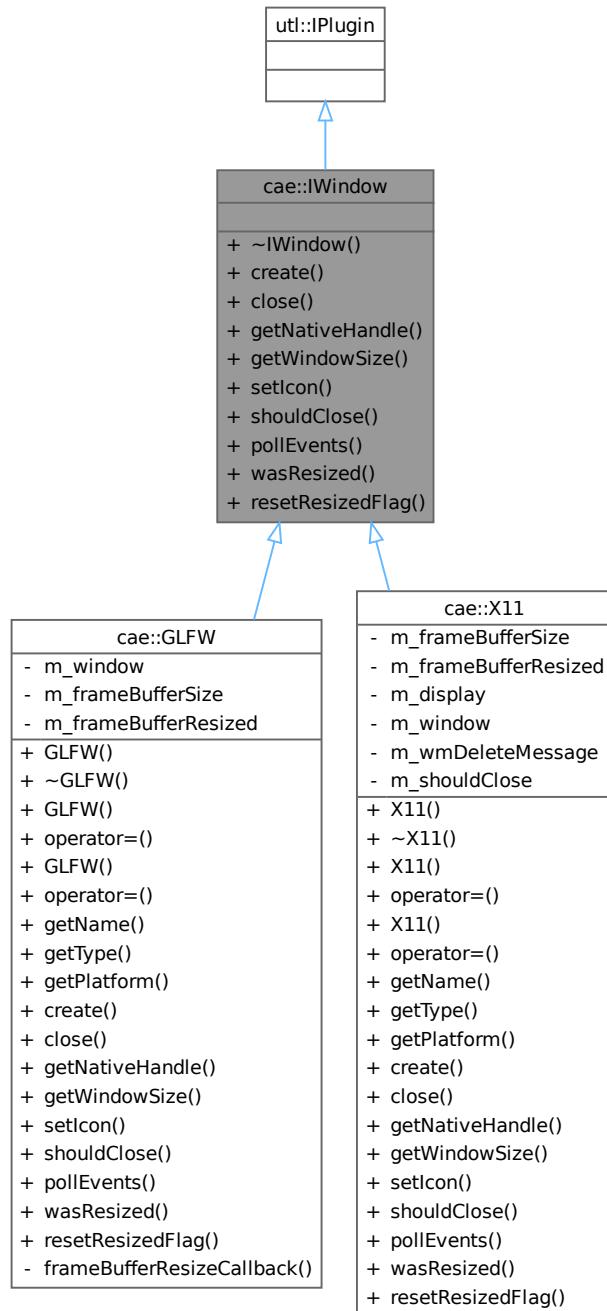
- `modules/Interfaces/include/Interfaces/IRenderer.hpp`

14.16 cae::IWindow Interface Reference

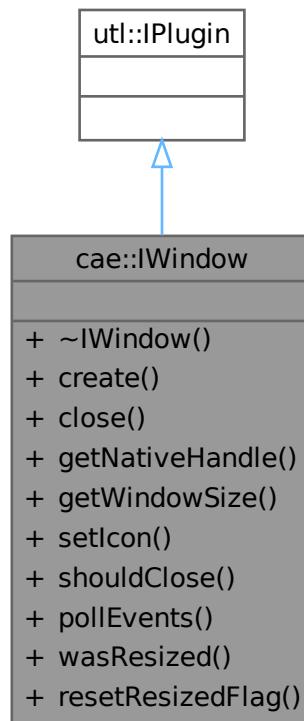
Interface for window.

```
#include <IWindow.hpp>
```

Inheritance diagram for cae::IWindow:



Collaboration diagram for cae::IWindow:



Public Member Functions

- `~IWindow () override=default`
- `virtual bool create (const std::string &name, WindowSize size)=0`
- `virtual void close ()=0`
- `virtual NativeWindowHandle getNativeHandle () const =0`
- `virtual WindowSize getWindowSize () const =0`
- `virtual bool setIcon (const std::string &path) const =0`
- `virtual bool shouldClose () const =0`
- `virtual void pollEvents ()=0`
- `virtual bool wasResized () const =0`
- `virtual void resetResizedFlag ()=0`

14.16.1 Detailed Description

Interface for window.

Definition at line 31 of file [IWindow.hpp](#).

14.16.2 Constructor & Destructor Documentation

14.16.2.1 ~IWindow()

```
cae::IWindow::~IWindow () [override], [default]
```

14.16.3 Member Function Documentation

14.16.3.1 close()

```
virtual void cae::IWindow::close () [pure virtual]
```

Implemented in [cae::GLFW](#), and [cae::X11](#).

14.16.3.2 create()

```
virtual bool cae::IWindow::create (
    const std::string & name,
    WindowSize size) [pure virtual]
```

Implemented in [cae::GLFW](#), and [cae::X11](#).

14.16.3.3 getNativeHandle()

```
virtual NativeWindowHandle cae::IWindow::getNativeHandle () const [pure virtual]
```

Implemented in [cae::GLFW](#), and [cae::X11](#).

14.16.3.4 getWindowSize()

```
virtual WindowSize cae::IWindow::getWindowSize () const [pure virtual]
```

Implemented in [cae::GLFW](#), and [cae::X11](#).

14.16.3.5 pollEvents()

```
virtual void cae::IWindow::pollEvents () [pure virtual]
```

Implemented in [cae::GLFW](#), and [cae::X11](#).

14.16.3.6 resetResizedFlag()

```
virtual void cae::IWindow::resetResizedFlag () [pure virtual]
```

Implemented in [cae::GLFW](#), and [cae::X11](#).

14.16.3.7 setIcon()

```
virtual bool cae::IWindow::setIcon (
    const std::string & path) const [pure virtual]
```

Implemented in [cae::GLFW](#), and [cae::X11](#).

14.16.3.8 shouldClose()

```
virtual bool cae::IWindow::shouldClose () const [pure virtual]
```

Implemented in [cae::GLFW](#), and [cae::X11](#).

14.16.3.9 wasResized()

```
virtual bool cae::IWindow::wasResized () const [pure virtual]
```

Implemented in [cae::GLFW](#), and [cae::X11](#).

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

- modules/Interfaces/include/Interfaces/IWindow.hpp

14.17 cae::NativeWindowHandle Struct Reference

```
#include <IWindow.hpp>
```

Collaboration diagram for cae::NativeWindowHandle:

cae::NativeWindowHandle
+ window
+ display

Public Attributes

- void * **window**
- void * **display**

14.17.1 Detailed Description

Definition at line 20 of file [IWindow.hpp](#).

14.17.2 Member Data Documentation

14.17.2.1 display

`void* cae::NativeWindowHandle::display`

Definition at line 23 of file [IWindow.hpp](#).

14.17.2.2 window

`void* cae::NativeWindowHandle::window`

Definition at line 22 of file [IWindow.hpp](#).

Referenced by [cae::GLFW::getNativeHandle\(\)](#), and [cae::X11::getNativeHandle\(\)](#).

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

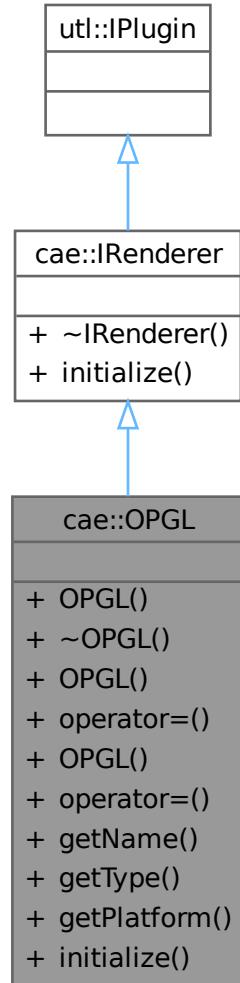
- [modules/Interfaces/include/Interfaces/IWindow.hpp](#)

14.18 cae::OPGL Class Reference

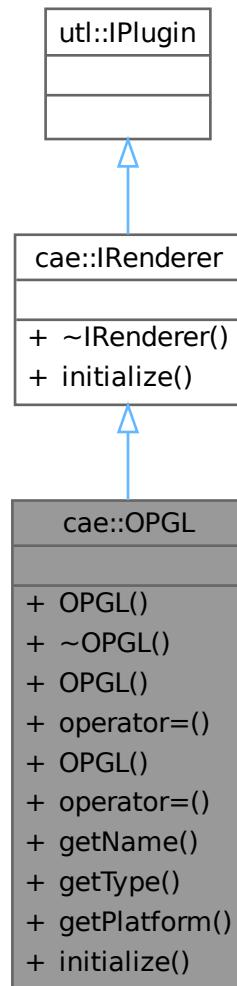
Class for the OpenGL plugin.

`#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`
- `utl::PluginPlatform getPlatform () const override`
- `void initialize (void *nativeWindowHandle) override`

Public Member Functions inherited from `cae::IRenderer`

- `~IRenderer () override=default`

14.18.1 Detailed Description

Class for the OpenGL plugin.

Definition at line 19 of file [OPGL.hpp](#).

14.18.2 Constructor & Destructor Documentation

14.18.2.1 OPGL() [1/3]

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

14.18.2.2 ~OPGL()

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

14.18.2.3 OPGL() [2/3]

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

14.18.2.4 OPGL() [3/3]

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

14.18.3 Member Function Documentation

14.18.3.1 getName()

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

Definition at line 31 of file [OPGL.hpp](#).

14.18.3.2 getPlatform()

```
utl::PluginPlatform cae::OPGL::getPlatform () const [inline], [nodiscard], [override]
```

Definition at line 33 of file [OPGL.hpp](#).

14.18.3.3 getType()

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

Definition at line 32 of file [OPGL.hpp](#).

14.18.3.4 initialize()

```
void cae::OPGL::initialize (
    void * nativeWindowHandle) [inline], [override], [virtual]
```

Implements [cae::IRenderer](#).

Definition at line [35](#) of file [OPGL.hpp](#).

14.18.3.5 operator=() [1/2]

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

14.18.3.6 operator=() [2/2]

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

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

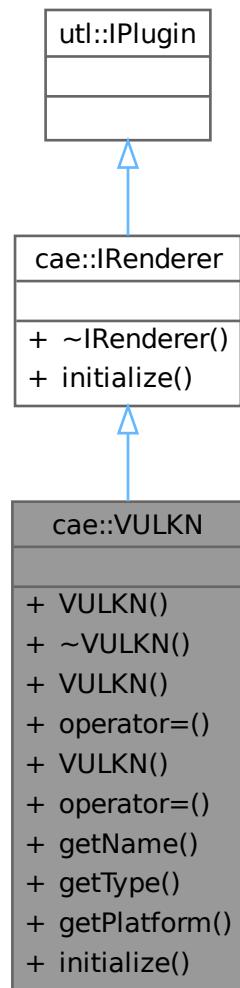
- [plugins/Renderer/OpenGL/include/OPGL/OPGL.hpp](#)

14.19 cae::VULKN Class Reference

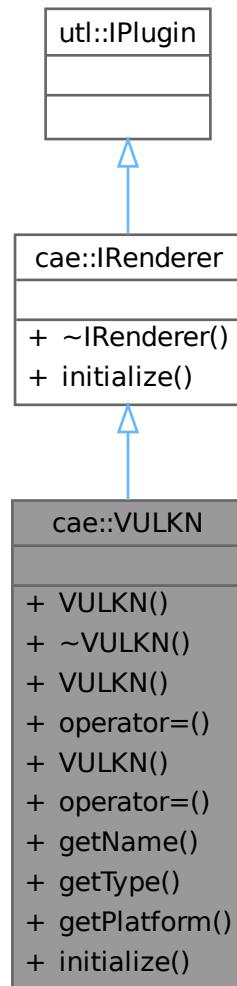
Class for the Vulkan plugin.

```
#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
- `utl::PluginPlatform getPlatform ()` const override
- `void initialize (void *nativeWindowHandle)` override

Public Member Functions inherited from `cae::IRenderer`

- `~IRenderer ()` override=default

14.19.1 Detailed Description

Class for the Vulkan plugin.

Definition at line 19 of file [VULKN.hpp](#).

14.19.2 Constructor & Destructor Documentation

14.19.2.1 VULKN() [1/3]

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

14.19.2.2 ~VULKN()

```
cae::VULKN::~VULKN () [override], [default]
```

14.19.2.3 VULKN() [2/3]

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

14.19.2.4 VULKN() [3/3]

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

14.19.3 Member Function Documentation

14.19.3.1 getName()

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

Definition at line 31 of file [VULKN.hpp](#).

14.19.3.2 getPlatform()

```
utl::PluginPlatform cae::VULKN::getPlatform () const [inline], [nodiscard], [override]
```

Definition at line 33 of file [VULKN.hpp](#).

14.19.3.3 getType()

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

Definition at line 32 of file [VULKN.hpp](#).

14.19.3.4 initialize()

```
void cae::VULKN::initialize (
    void * nativeWindowHandle) [inline], [override], [virtual]
```

Implements [cae::IRenderer](#).

Definition at line 35 of file [VULKN.hpp](#).

14.19.3.5 operator=() [1/2]

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

14.19.3.6 operator=() [2/2]

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

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

- [plugins/Renderer/Vulkan/include/VULKN/VULKN.hpp](#)

14.20 cae::WindowSize Struct Reference

```
#include <IWindow.hpp>
```

Collaboration diagram for cae::WindowSize:

cae::WindowSize
+ width
+ height

Public Attributes

- `uint16_t width`
- `uint16_t height`

14.20.1 Detailed Description

Definition at line 14 of file [IWindow.hpp](#).

14.20.2 Member Data Documentation

14.20.2.1 height

`uint16_t cae::WindowSize::height`

Definition at line 17 of file [IWindow.hpp](#).

Referenced by [cae::GLFW::create\(\)](#), and [cae::X11::create\(\)](#).

14.20.2.2 width

`uint16_t cae::WindowSize::width`

Definition at line 16 of file [IWindow.hpp](#).

Referenced by [cae::GLFW::create\(\)](#), and [cae::X11::create\(\)](#).

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

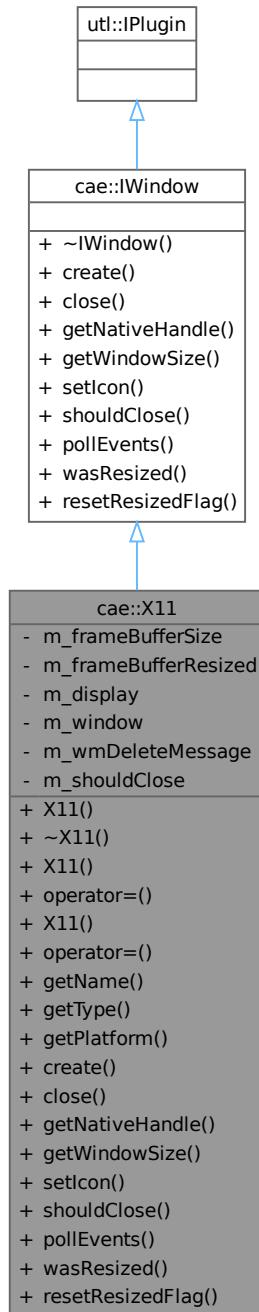
- [modules/Interfaces/include/Interfaces/IWindow.hpp](#)

14.21 cae::X11 Class Reference

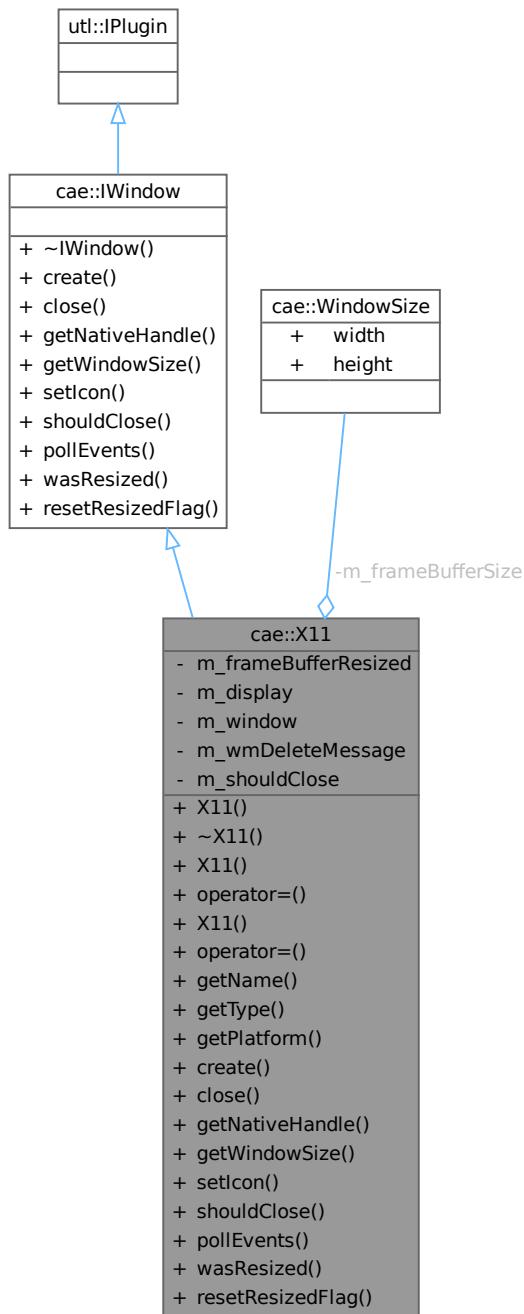
Class for the [X11](#) plugin.

```
#include <X11.hpp>
```

Inheritance diagram for cae::X11:



Collaboration diagram for cae::X11:



Public Member Functions

- **X11 ()**=default
- **~X11 ()** override=default
- **X11 (const X11 &)**=delete
- **X11 & operator= (const X11 &)**=delete
- **X11 (X11 &&)**=delete

- `X11 & operator= (X11 &&) = delete`
- `std::string getName () const override`
- `utl::PluginType getType () const override`
- `utl::PluginPlatform getPlatform () const override`
- `bool create (const std::string &name, WindowSize size) override`
- `void close () override`
- `NativeWindowHandle getNativeHandle () const override`
- `WindowSize getWindowSize () const override`
- `bool setIcon (const std::string &path) const override`
- `bool shouldClose () const override`
- `void pollEvents () override`
- `bool wasResized () const override`
- `void resetResizedFlag () override`

Public Member Functions inherited from `cae::IWindow`

- `~IWindow () override = default`

Private Attributes

- `WindowSize m_frameBufferSize`
- `bool m_frameBufferResized = false`
- `Display * m_display = nullptr`
- `Window m_window = 0`
- `Atom m_wmDeleteMessage = 0`
- `bool m_shouldClose = false`

14.21.1 Detailed Description

Class for the `X11` plugin.

Definition at line 19 of file `X11.hpp`.

14.21.2 Constructor & Destructor Documentation

14.21.2.1 X11() [1/3]

`cae::X11::X11 () [default]`

14.21.2.2 ~X11()

`cae::X11::~X11 () [override], [default]`

14.21.2.3 X11() [2/3]

```
cae::X11::X11 (
    const X11 & ) [delete]
```

14.21.2.4 X11() [3/3]

```
cae::X11::X11 (
    X11 && ) [delete]
```

14.21.3 Member Function Documentation

14.21.3.1 close()

```
void cae::X11::close () [override], [virtual]
```

Implements [cae::IWindow](#).

Definition at line 44 of file [x11.cpp](#).

References [cae::U](#).

14.21.3.2 create()

```
bool cae::X11::create (
    const std::string & name,
    WindowSize size) [override], [virtual]
```

Implements [cae::IWindow](#).

Definition at line 8 of file [x11.cpp](#).

References [cae::WindowSize::height](#), [m_display](#), [m_frameBufferSize](#), [m_window](#), [m_wmDeleteMessage](#), [cae::U](#), and [cae::WindowSize::width](#).

14.21.3.3 getName()

```
std::string cae::X11::getName () const [inline], [nodiscard], [override]
```

Definition at line 31 of file [X11.hpp](#).

14.21.3.4 getNativeHandle()

```
NativeWindowHandle cae::X11::getNativeHandle () const [inline], [nodiscard], [override], [virtual]
```

Implements [cae::IWindow](#).

Definition at line 38 of file [X11.hpp](#).

References [m_display](#), [m_window](#), and [cae::NativeWindowHandle::window](#).

14.21.3.5 getPlatform()

```
utl::PluginPlatform cae::X11::getPlatform () const [inline], [nodiscard], [override]
```

Definition at line 33 of file [X11.hpp](#).

14.21.3.6 `getType()`

`utl::PluginType cae::X11::getType () const [inline], [nodiscard], [override]`

Definition at line 32 of file [X11.hpp](#).

14.21.3.7 `getWindowSize()`

`cae::WindowSize cae::X11::getWindowSize () const [nodiscard], [override], [virtual]`

Implements [cae::IWindow](#).

Definition at line 55 of file [x11.cpp](#).

References [cae::U](#).

14.21.3.8 `operator=()` [1/2]

`X11 & cae::X11::operator= (`
 `const X11 &) [delete]`

14.21.3.9 `operator=()` [2/2]

`X11 & cae::X11::operator= (`
 `X11 &&) [delete]`

14.21.3.10 `pollEvents()`

`void cae::X11::pollEvents () [override], [virtual]`

Implements [cae::IWindow](#).

Definition at line 75 of file [x11.cpp](#).

14.21.3.11 `resetResizedFlag()`

`void cae::X11::resetResizedFlag () [inline], [override], [virtual]`

Implements [cae::IWindow](#).

Definition at line 50 of file [X11.hpp](#).

References [m_frameBufferResized](#).

14.21.3.12 setIcon()

```
bool cae::X11::setIcon (
    const std::string & path) const [nodiscard], [override], [virtual]
```

Implements [cae::IWindow](#).

Definition at line [67](#) of file [x11.cpp](#).

14.21.3.13 shouldClose()

```
bool cae::X11::shouldClose () const [nodiscard], [override], [virtual]
```

Implements [cae::IWindow](#).

Definition at line [73](#) of file [x11.cpp](#).

14.21.3.14 wasResized()

```
bool cae::X11::wasResized () const [inline], [override], [virtual]
```

Implements [cae::IWindow](#).

Definition at line [49](#) of file [X11.hpp](#).

References [m_frameBufferResized](#).

14.21.4 Member Data Documentation

14.21.4.1 m_display

```
Display* cae::X11::m_display = nullptr [private]
```

Definition at line [56](#) of file [X11.hpp](#).

Referenced by [create\(\)](#), and [getNativeHandle\(\)](#).

14.21.4.2 m_frameBufferResized

```
bool cae::X11::m_frameBufferResized = false [mutable], [private]
```

Definition at line [54](#) of file [X11.hpp](#).

Referenced by [resetResizedFlag\(\)](#), and [wasResized\(\)](#).

14.21.4.3 m_frameBufferSize

```
WindowSize cae::X11::m_frameBufferSize [private]
```

Definition at line [53](#) of file [X11.hpp](#).

Referenced by [create\(\)](#).

14.21.4.4 m_shouldClose

bool cae::X11::m_shouldClose = false [private]

Definition at line 59 of file [X11.hpp](#).

14.21.4.5 m_window

Window cae::X11::m_window = 0 [private]

Definition at line 57 of file [X11.hpp](#).

Referenced by [create\(\)](#), and [getNativeHandle\(\)](#).

14.21.4.6 m_wmDeleteMessage

Atom cae::X11::m_wmDeleteMessage = 0 [private]

Definition at line 58 of file [X11.hpp](#).

Referenced by [create\(\)](#).

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

- [plugins/Window/X11/include/X11/X11.hpp](#)
- [plugins/Window/X11/src/x11.cpp](#)

Chapter 15

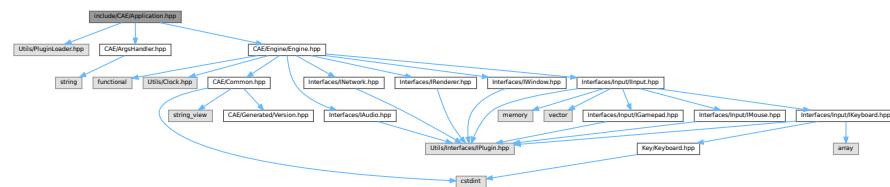
File Documentation

15.1 CONTRIBUTING.md File Reference

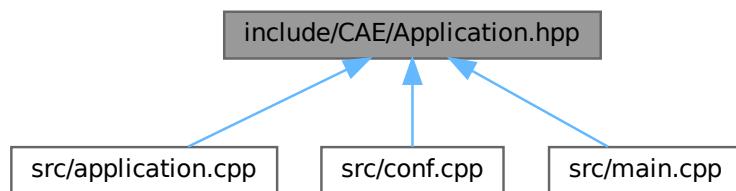
15.2 include/CAE/Application.hpp File Reference

This file contains the Application class declaration.

```
#include <Utils/PluginLoader.hpp>
#include "CAE/ArgsHandler.hpp"
#include "CAE/Engine/Engine.hpp"
Include dependency graph for Application.hpp:
```



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



Classes

- struct `cae::AppConfig`
- class `cae::Application`

Main class.

Namespaces

- namespace `cae`

15.2.1 Detailed Description

This file contains the Application class declaration.

Definition in file [Application.hpp](#).

15.3 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 #include <Utils/PluginLoader.hpp>
00010
00011 #include "CAE/ArgsHandler.hpp"
00012 #include "CAE/Engine/Engine.hpp"
00013
00014 namespace cae
00015 {
00016
00017     struct AppConfig
00018     {
00019         EngineConfig engineConfig;
00020         EnvConfig envConfig;
00021     };
00022
00023 /**
00024  * @class Application
00025  * @brief Main class
00026  * @namespace cae
00027 /**
00028 class Application
00029 {
00030
00031     public:
00032         Application(const ArgsConfig &argsConfig, const EnvConfig &envConfig);
00033         ~Application() = default;
00034
00035         Application(const Application &) = delete;
00036         Application &operator=(const Application &) = delete;
00037         Application(Application &&) = delete;
00038         Application &operator=(Application &&) = delete;
00039
00040         void start() const;
00041         void stop();
00042
00043     private:
00044         void setupEngine(const std::string &rendererName, const std::string &windowName);
00045
00046         static EngineConfig parseEngineConf(const std::string &path);
00047
00048         std::unique_ptr<utl::PluginLoader> m_pluginLoader = nullptr;
00049         std::unique_ptr<Engine> m_engine = nullptr;
00050
00051         AppConfig m_appConfig;
00052     }; // class Application
00053
00054 } // namespace cae

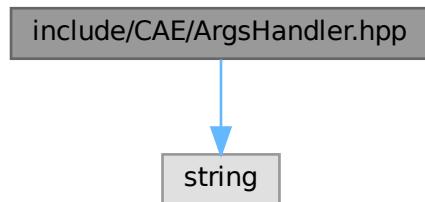
```

15.4 include/CAE/ArgsHandler.hpp File Reference

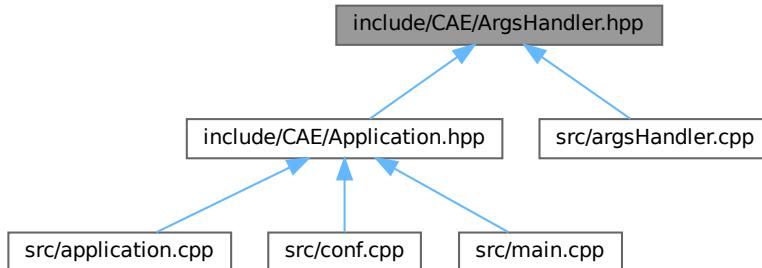
This file contains the ArgsHandler class declaration.

```
#include <string>
```

Include dependency graph for ArgsHandler.hpp:



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`

15.4.1 Detailed Description

This file contains the ArgsHandler class declaration.

Definition in file [ArgsHandler.hpp](#).

15.5 ArgsHandler.hpp

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

```

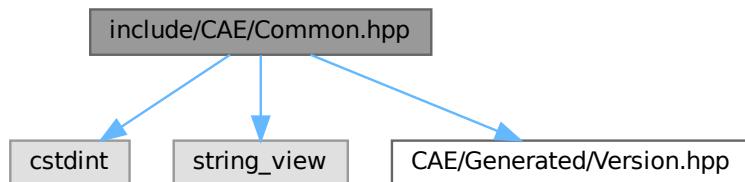
00001 /**
00002  * @file ArgsHandler.hpp
00003  * @brief This file contains the ArgsHandler class declaration
00004  * @namespace cae
00005 /**
00006
00007 #pragma once
00008
00009 #include <string>
00010
00011 namespace cae
00012 {
00013
00014     struct ArgsConfig
00015     {
00016         bool run = false;
00017         std::string config_path;
00018     };
00019     struct EnvConfig
00020     {
00021         std::string user_name;
00022         std::string pwd;
00023     };
00024
00025 /**
00026  * @class ArgsHandler
00027  * @brief Class to handle command line arguments
00028  * @namespace cae
00029 /**
00030 class ArgsHandler
00031 {
00032
00033     public:
00034         ArgsHandler() = default;
00035         ~ArgsHandler() = default;
00036
00037         ArgsHandler(const ArgsHandler &) = delete;
00038         ArgsHandler &operator=(const ArgsHandler &) = delete;
00039         ArgsHandler(ArgsHandler &&) = delete;
00040         ArgsHandler &operator=(ArgsHandler &&) = delete;
00041
00042         static ArgsConfig ParseArgs(int argc, const char *const *argv);
00043         static EnvConfig ParseEnv(const char *const *envp);
00044
00045     private:
00046 }; // class ArgsHandler
00047
00048 } // namespace cae

```

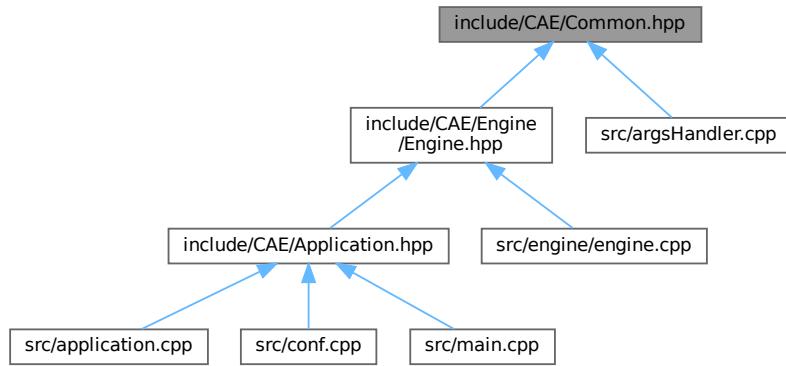
15.6 include/CAE/Common.hpp File Reference

This file contains.

```
#include <cstdint>
#include <string_view>
#include "CAE/Generated/Version.hpp"
Include dependency graph for Common.hpp:
```



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



Namespaces

- namespace `cae`
- namespace `cae::Audio`
- namespace `cae::Message`
- namespace `cae::Network`
- namespace `cae::Plugins`
- namespace `cae::Plugins::Name`
- namespace `cae::Renderer`
- namespace `cae::User`
- namespace `cae::Window`

Macros

- `#define APP_EXTENSION ""`

Variables

- `constexpr auto cae::Audio::VOLUME = 1.F`
- `constexpr auto cae::Audio::MUTED = false`
- `static constexpr std::string_view cae::Message::HELP_MSG`
- `static constexpr std::string_view cae::Message::VERSION_MSG`
- `constexpr auto cae::Network::HOST = "127.0.0.1"`
- `constexpr auto cae::Network::PORT = 4242`
- `constexpr auto cae::Plugins::Name::RENDERER_OPENGL = "OpenGL"`
- `constexpr auto cae::Plugins::Name::RENDERER_VULKAN = "Vulkan"`
- `constexpr auto cae::Plugins::Name::WINDOW_GLFW = "GLFW"`
- `constexpr auto cae::Renderer::VSYNC = false`
- `constexpr auto cae::Renderer::FRAME_RATE_LIMIT = 90`
- `constexpr auto cae::User::NAME = "User"`
- `constexpr uint16_t cae::Window::WIDTH = 1920`
- `constexpr uint16_t cae::Window::HEIGHT = 1080`
- `constexpr auto cae::Window::NAME = "CAE - Cross API Engine"`
- `constexpr auto cae::Window::FULLSCREEN = false`

15.6.1 Detailed Description

This file contains.

Definition in file [Common.hpp](#).

15.6.2 Macro Definition Documentation

15.6.2.1 APP_EXTENSION

```
#define APP_EXTENSION ""
```

Definition at line 15 of file [Common.hpp](#).

15.7 Common.hpp

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

```
00001 /**
00002 /**
00003 /**
00004 /**
00005 /**
00006
00007 #pragma once
00008
00009 #include <cstdint>
00010 #include <string_view>
00011
00012 #ifdef _WIN32
00013 #define APP_EXTENSION ".exe"
00014 #else
00015 #define APP_EXTENSION ""
00016 #endif
00017
00018 #include "CAE/Generated/Version.hpp"
00019
00020 namespace cae
00021 {
00022     namespace Audio
00023     {
00024         inline constexpr auto VOLUME = 1.F;
00025         inline constexpr auto MUTED = false;
00026     } // namespace Audio
00027
00028     namespace Message
00029     {
00030         static constexpr std::string_view HELP_MSG = "Usage: " PROJECT_NAME APP_EXTENSION " [options]\n\n"
00031                     "Options:\n"
00032                     " -h, --help           Show this help message\n"
00033                     " -v, --version        Show version information\n"
00034                     " -c, --config <path>   Specify JSON configuration file";
00035         static constexpr std::string_view VERSION_MSG = PROJECT_NAME
00036             " v" PROJECT_VERSION " " BUILD_TYPE "(" GIT_TAG ", commit " GIT_COMMIT_HASH ") "
00037             DATE " " TIME;
00038     } // namespace Message
00039
00040     namespace Network
00041     {
00042         inline constexpr auto HOST = "127.0.0.1";
00043         inline constexpr auto PORT = 4242;
00044     } // namespace Network
00045
00046     namespace Plugins::Name
00047     {
00048         inline constexpr auto RENDERER_OPENGL = "OpenGL";
00049         inline constexpr auto RENDERER_VULKAN = "Vulkan";
00050         inline constexpr auto WINDOW_GLFW = "GLFW";
00051     } // namespace Plugins::Name
00052
00053     namespace Renderer
00054     {
```

```

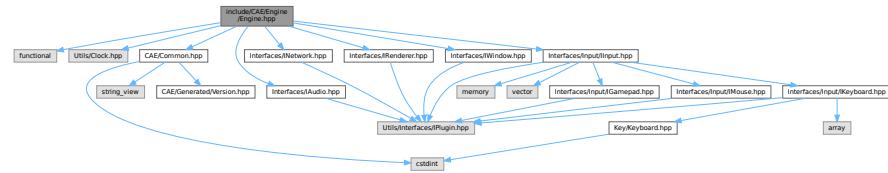
00054     inline constexpr auto VSYNC = false;
00055     inline constexpr auto FRAME_RATE_LIMIT = 90;
00056 } // namespace Renderer
00057
00058 namespace User
00059 {
00060     inline constexpr auto NAME = "User";
00061 } // namespace User
00062
00063 namespace Window
00064 {
00065     inline constexpr uint16_t WIDTH = 1920;
00066     inline constexpr uint16_t HEIGHT = 1080;
00067     inline constexpr auto NAME = "CAE - Cross API Engine";
00068     inline constexpr auto FULLSCREEN = false;
00069 } // namespace Window
00070 } // namespace cae

```

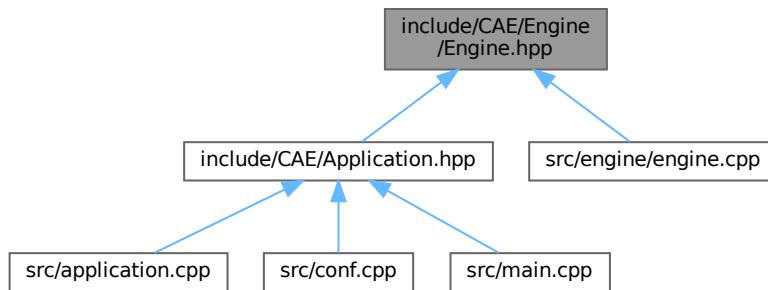
15.8 include/CAE/Engine/Engine.hpp File Reference

This file contains the engine class declaration.

```
#include <functional>
#include <Utils/Clock.hpp>
#include "CAE/Common.hpp"
#include "Interfaces/IAudio.hpp"
#include "Interfaces/INetwork.hpp"
#include "Interfaces/IRenderer.hpp"
#include "Interfaces/IWindow.hpp"
#include "Interfaces/Input/IInput.hpp"
Include dependency graph for Engine.hpp:
```



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



Classes

- struct `cae::EngineConfig`
- class `cae::Engine`
 `Engine` class.

Namespaces

- namespace `cae`

15.8.1 Detailed Description

This file contains the engine class declaration.

Definition in file [Engine.hpp](#).

15.9 Engine.hpp

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

```

00001 /**
00002  * @file Engine.hpp
00003  * @brief This file contains the engine class declaration
00004  * @namespace cae
00005 /**
00006
00007 #pragma once
00008
00009 #include <functional>
00010
00011 #include <Utils/Clock.hpp>
00012
00013 #include "CAE/Common.hpp"
00014 #include "Interfaces/IAudio.hpp"
00015 #include "Interfaces/INetwork.hpp"
00016 #include "Interfaces/IRenderer.hpp"
00017 #include "Interfaces/IWindow.hpp"
00018 #include "Interfaces/Input/IInput.hpp"
00019
00020 namespace cae
00021 {
00022
00023     struct EngineConfig
00024     {
00025         float audio_master_volume = Audio::VOLUME;
00026         bool audio_muted = Audio::MUTED;
00027
00028         std::string network_host = Network::HOST;
00029         uint16_t network_port = Network::PORT;
00030
00031         bool renderer_vsync = Renderer::VSYNC;
00032         uint16_t renderer_frame_rate_limit = Renderer::FRAME_RATE_LIMIT;
00033
00034         uint16_t window_width = Window::WIDTH;
00035         uint16_t window_height = Window::HEIGHT;
00036         bool window_fullscreen = Window::FULLSCREEN;
00037         std::string window_name = Window::NAME;
00038     };
00039
00040 /**
00041  * @class Engine
00042  * @brief Engine class
00043  * @namespace cae
00044 /**
00045 class Engine
00046 {
00047
00048     public:
00049         Engine(const EngineConfig &config, const std::function<std::shared_ptr<IAudio>()> &audioFactory,
00050                 const std::function<std::shared_ptr<IInput>()> &inputFactory,
00051                 const std::function<std::shared_ptr<INetwork>()> &networkFactory,

```

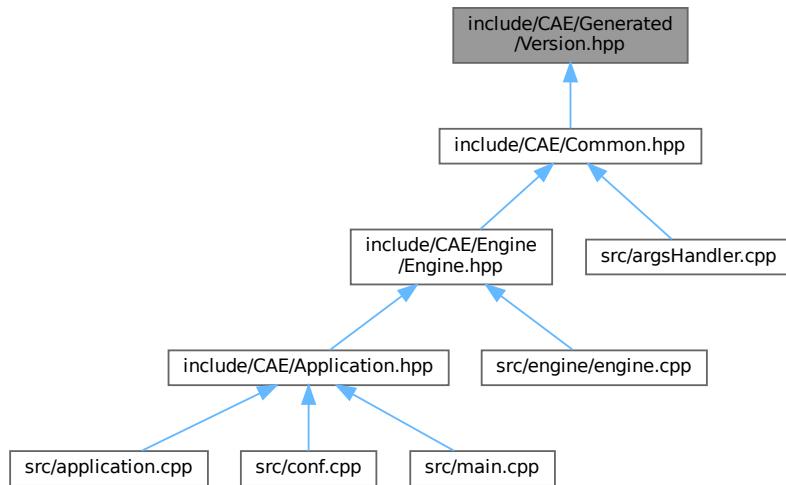
```

00052         const std::function<std::shared_ptr<IRenderer>()> &rendererFactory,
00053         const std::function<std::shared_ptr<IWindow>()> &windowFactory);
00054     ~Engine() = default;
00055
00056     Engine(const Engine &) = delete;
00057     Engine &operator=(const Engine &) = delete;
00058     Engine(Engine &&) = delete;
00059     Engine &operator=(Engine &&) = delete;
00060
00061     [[nodiscard]] const std::shared_ptr<IAudio> &getAudio() const { return m_audioPlugin; }
00062     [[nodiscard]] const std::shared_ptr<IInput> &getInput() const { return m_inputPlugin; }
00063     [[nodiscard]] const std::shared_ptr<INetwork> &getNetwork() const { return m_networkPlugin; }
00064     [[nodiscard]] const std::shared_ptr<IRenderer> &getRenderer() const { return m_rendererPlugin; }
00065     [[nodiscard]] const std::shared_ptr<IWindow> &getWindow() const { return m_windowPlugin; }
00066
00067     [[nodiscard]] const std::unique_ptr<utl::Clock> &getClock() { return m_clock; }
00068
00069     void run() const;
00070     void stop();
00071
00072 private:
00073     std::shared_ptr<IAudio> m_audioPlugin = nullptr;
00074     std::shared_ptr<IInput> m_inputPlugin = nullptr;
00075     std::shared_ptr<INetwork> m_networkPlugin = nullptr;
00076     std::shared_ptr<IRenderer> m_rendererPlugin = nullptr;
00077     std::shared_ptr<IWindow> m_windowPlugin = nullptr;
00078
00079     std::unique_ptr<utl::Clock> m_clock = nullptr;
00080
00081 }; // class Engine
00082 } // namespace cae

```

15.10 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 "9438779"
- #define GIT_TAG "9438779"
- #define BUILD_TYPE "Release"

15.10.1 Macro Definition Documentation

15.10.1.1 BUILD_TYPE

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

Definition at line 15 of file [Version.hpp](#).

15.10.1.2 GIT_COMMIT_HASH

```
#define GIT_COMMIT_HASH "9438779"
```

Definition at line 13 of file [Version.hpp](#).

15.10.1.3 GIT_TAG

```
#define GIT_TAG "9438779"
```

Definition at line 14 of file [Version.hpp](#).

15.10.1.4 PROJECT_NAME

```
#define PROJECT_NAME "cae"
```

Definition at line 7 of file [Version.hpp](#).

15.10.1.5 PROJECT_VERSION

```
#define PROJECT_VERSION "0.0.0"
```

Definition at line 8 of file [Version.hpp](#).

15.10.1.6 PROJECT_VERSION_MAJOR

```
#define PROJECT_VERSION_MAJOR "0"
```

Definition at line 9 of file [Version.hpp](#).

15.10.1.7 PROJECT_VERSION_MINOR

```
#define PROJECT_VERSION_MINOR "0"
```

Definition at line 10 of file [Version.hpp](#).

15.10.1.8 PROJECT_VERSION_PATCH

```
#define PROJECT_VERSION_PATCH "0"
```

Definition at line 11 of file [Version.hpp](#).

15.11 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 "9438779"
00014 #define GIT_TAG "9438779"
00015 #define BUILD_TYPE "Release"
```

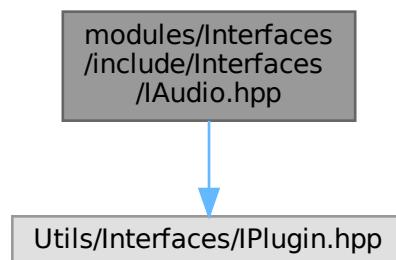
15.12 LICENSE.md File Reference

15.13 modules/Interfaces/include/Interfaces/IAudio.hpp File Reference

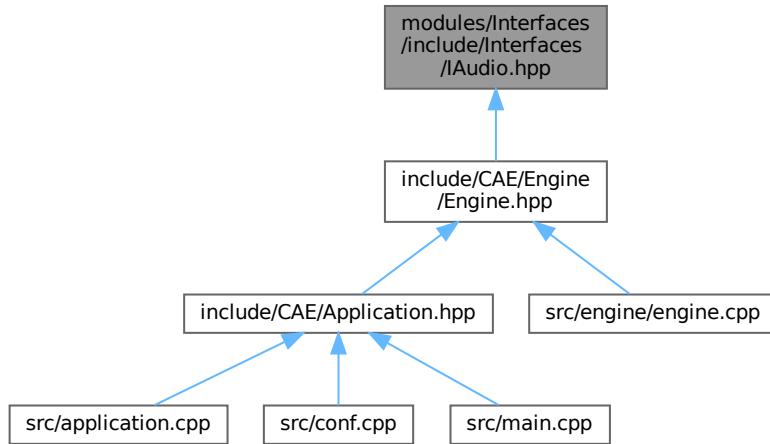
This file contains the audio interface.

```
#include "Utils/Interfaces/IPlugin.hpp"
```

Include dependency graph for IAudio.hpp:



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



Classes

- interface [cae::IAudio](#)
Interface for audio.

Namespaces

- namespace [cae](#)

15.13.1 Detailed Description

This file contains the audio interface.

Definition in file [IAudio.hpp](#).

15.14 IAudio.hpp

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

```

00001 /**
00002  * @file IAudio.hpp
00003  * @brief This file contains the audio interface
00004  * @namespace cae
00005 /**
00006
00007 #pragma once
00008
00009 #include "Utils/Interfaces/IPPlugin.hpp"
00010
00011 namespace cae
00012 {
00013
00014 /**
00015  * @interface IAudio
00016  * @brief Interface for audio
  
```

```

00017  /// @namespace cae
00018  /**
00019  class IAudio : public utl::IPlugin
00020  {
00021      public:
00022          ~IAudio() override = default;
00023  }; // interface IAudio
00026
00027 } // namespace cae

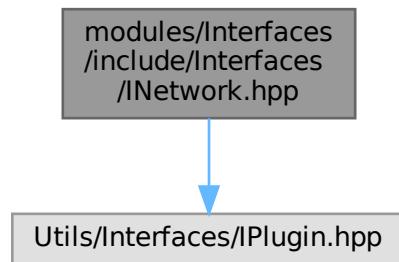
```

15.15 modules/Interfaces/include/Interfaces/INetwork.hpp File Reference

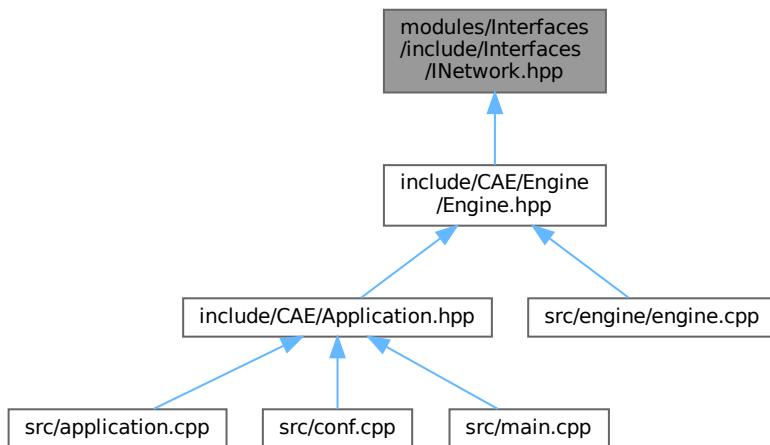
This file contains the network interface.

#include "Utils/Interfaces/IPlugin.hpp"

Include dependency graph for INetwork.hpp:



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



Classes

- interface [cae::INetwork](#)

Interface for network.

Namespaces

- namespace [cae](#)

15.15.1 Detailed Description

This file contains the network interface.

Definition in file [INetwork.hpp](#).

15.16 INetwork.hpp

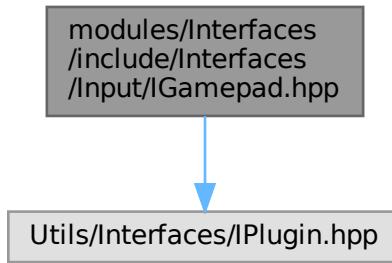
[Go to the documentation of this file.](#)

```
00001 /**
00002 /// @file INetwork.hpp
00003 /// @brief This file contains the network interface
00004 /// @namespace cae
00005 /**
00006
00007 #pragma once
00008
00009 #include "Utils/Interfaces/IPlugin.hpp"
00010
00011 namespace cae
00012 {
00013
00014 /**
00015 /// @interface INetwork
00016 /// @brief Interface for network
00017 /// @namespace cae
00018 /**
00019 class INetwork : public utl::IPlugin
00020 {
00021
00022     public:
00023         ~INetwork() override = default;
00024
00025         virtual bool connect(const std::string &host, uint16_t port) = 0;
00026
00027 }; // interface INetwork
00028
00029 } // namespace cae
```

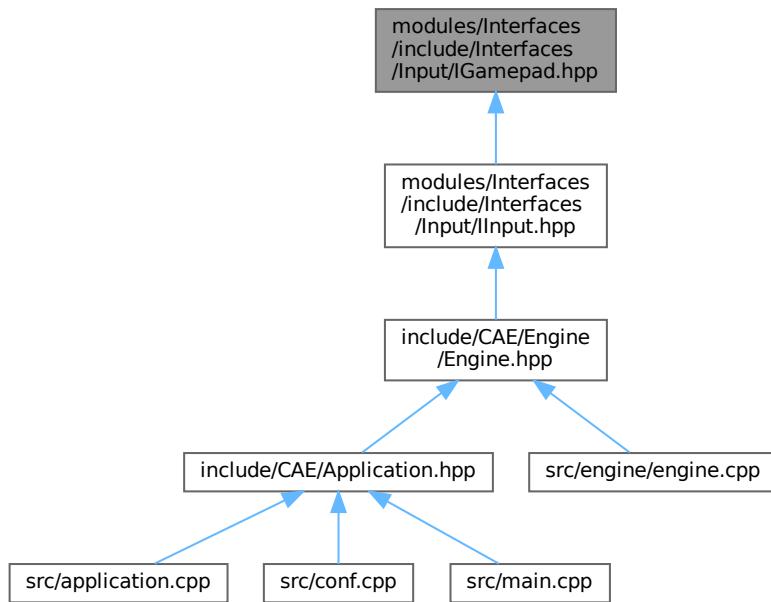
15.17 modules/Interfaces/include/Interfaces/Input/IGamepad.hpp File Reference

This file contains the input gamepad interface.

```
#include "Utils/Interfaces/IPlugin.hpp"
Include dependency graph for IGamepad.hpp:
```



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



Classes

- interface `cae::IGamepad`
Interface for gamepad.

Namespaces

- namespace `cae`

15.17.1 Detailed Description

This file contains the input gamepad interface.

Definition in file [IGamepad.hpp](#).

15.18 IGamepad.hpp

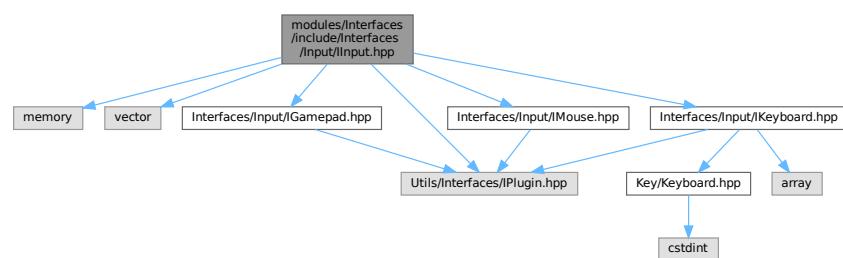
[Go to the documentation of this file.](#)

```
00001 /**
00002 /// @file IGamepad.hpp
00003 /// @brief This file contains the input gamepad interface
00004 /// @namespace cae
00005 /**
00006
00007 #pragma once
00008
00009 #include "Utils/Interfaces/IPPlugin.hpp"
00010
00011 namespace cae
00012 {
00013
00014 /**
00015 /// @interface IGamepad
00016 /// @brief Interface for gamepad
00017 /// @namespace cae
00018 /**
00019 class IGamepad : public utl::IPPlugin
00020 {
00021
00022     public:
00023         ~IGamepad() override = default;
00024
00025     }; // interface IGamepad
00026
00027 } // namespace cae
```

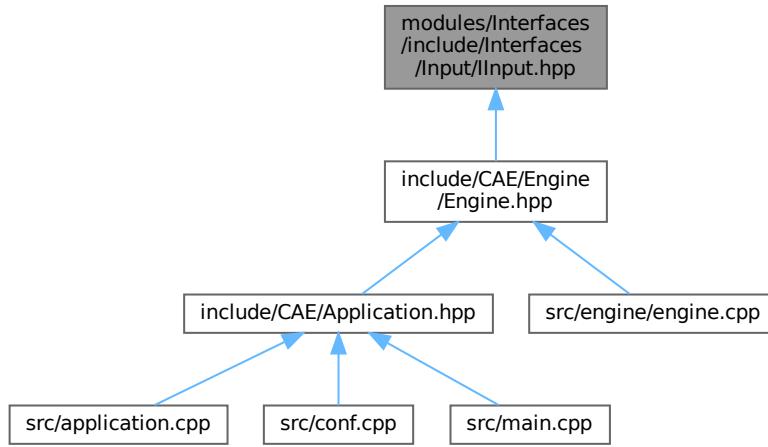
15.19 modules/Interfaces/include/Interfaces/Input/IInput.hpp File Reference

This file contains the input interface.

```
#include <memory>
#include <vector>
#include "Interfaces/Input/IGamepad.hpp"
#include "Interfaces/Input/IKeyboard.hpp"
#include "Interfaces/Input/IMouse.hpp"
#include "Utils/Interfaces/IPPlugin.hpp"
Include dependency graph for IInput.hpp:
```



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



Classes

- interface [cae::IIInput](#)
Interface for audio.

Namespaces

- namespace [cae](#)

15.19.1 Detailed Description

This file contains the input interface.

Definition in file [IIInput.hpp](#).

15.20 IIInput.hpp

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

```

00001 /**
00002 /**
00003 /**
00004 /**
00005 /**
00006
00007 #pragma once
00008
00009 #include <memory>
00010 #include <vector>
00011
00012 #include "Interfaces/Input/IGamepad.hpp"
00013 #include "Interfaces/Input/IKeyboard.hpp"
00014 #include "Interfaces/Input/IMouse.hpp"
00015 #include "Utils/Interfaces/IPPlugin.hpp"
00016
  
```

```

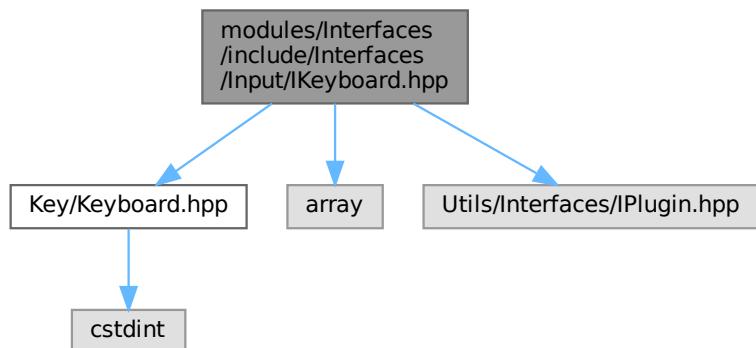
00017 namespace cae
00018 {
00019
00020     /**
00021     /// @interface IInput
00022     /// @brief Interface for audio
00023     /// @namespace cae
00024     /**
00025     class IInput : public utl::IPlugin
00026     {
00027
00028         public:
00029             ~IInput() override = default;
00030
00031             virtual const std::unique_ptr<IKeyboard> &getKeyboard() const = 0;
00032             virtual const std::unique_ptr<IMouse> &getMouse() const = 0;
00033             virtual const std::vector<std::unique_ptr<IGamepad>> &getGamepads() const = 0;
00034
00035         private:
00036             std::unique_ptr<IKeyboard> m_keyboard;
00037             std::unique_ptr<IMouse> m_mouse;
00038             std::vector<std::unique_ptr<IGamepad>> m_gamepads;
00039     }; // interface IInput
00040
00041 } // namespace cae

```

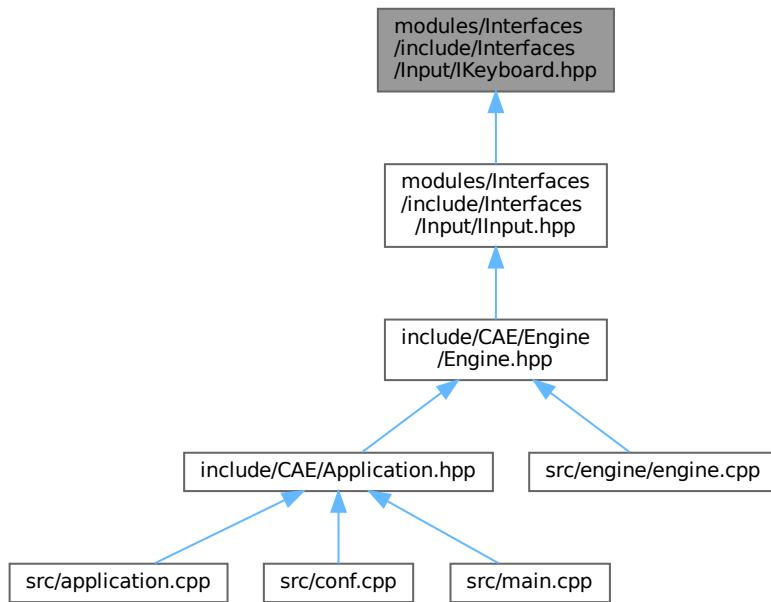
15.21 modules/Interfaces/include/Interfaces/Input/IKeyboard.hpp File Reference

This file contains the input keyboard interface.

```
#include "Key/Keyboard.hpp"
#include <array>
#include "Utils/Interfaces/IPlugin.hpp"
Include dependency graph for IKeyboard.hpp:
```



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



Classes

- interface `cae::IKeyboard`
Interface for keyboard.

Namespaces

- namespace `cae`

15.21.1 Detailed Description

This file contains the input keyboard interface.

Definition in file [IKeyboard.hpp](#).

15.22 IKeyboard.hpp

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

```

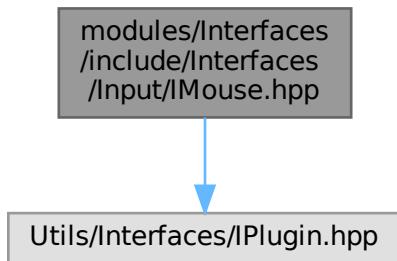
00001 /**
00002 /// @file IKeyboard.hpp
00003 /// @brief This file contains the input keyboard interface
00004 /// @namespace cae
00005 /**
00006
00007 #pragma once
00008
00009 #include "Key/Keyboard.hpp"
  
```

```
00010
00011 #include <array>
00012
00013 #include "Utils/Interfaces/IPlugin.hpp"
00014
00015 namespace cae
00016 {
00017
00018     /**
00019     * @interface IKeyboard
00020     * @brief Interface for keyboard
00021     * @namespace cae
00022     */
00023     class IKeyboard : public utl::IPlugin
00024     {
00025
00026     public:
00027         ~IKeyboard() override = default;
00028
00029         virtual bool isKeyPressed(KeyCode keyCode) const = 0;
00030
00031     private:
00032         std::array<KeyState, static_cast<size_t>(KeyCode::Count)> m_keyMap{};
00033     }; // interface IKeyboard
00034
00035 } // namespace cae
```

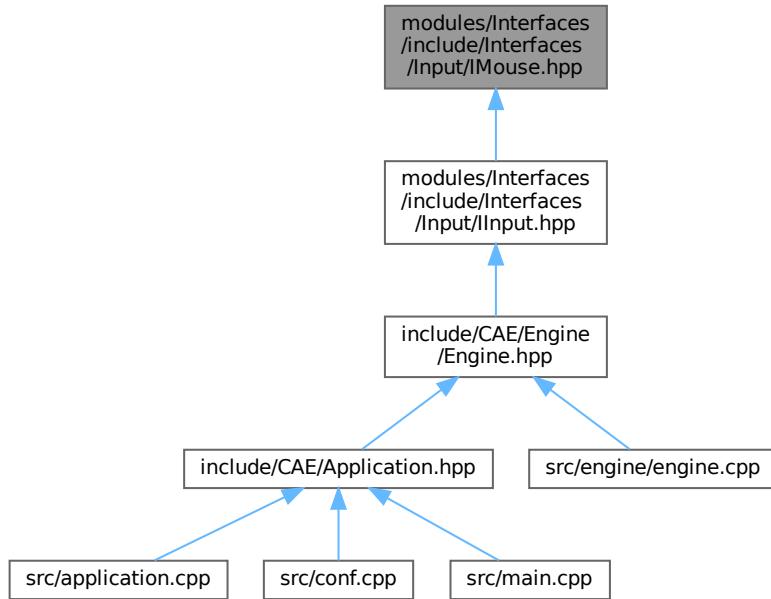
15.23 modules/Interfaces/include/Interfaces/Input/IMouse.hpp File Reference

This file contains the input mouse interface.

```
#include "Utils/Interfaces/IPlugin.hpp"
Include dependency graph for IMouse.hpp:
```



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



Classes

- interface `cae::IMouse`
Interface for mouse.

Namespaces

- namespace `cae`

15.23.1 Detailed Description

This file contains the input mouse interface.

Definition in file [IMouse.hpp](#).

15.24 IMouse.hpp

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

```

00001 /**
00002 /// @file IMouse.hpp
00003 /// @brief This file contains the input mouse interface
00004 /// @namespace cae
00005 /**
00006
00007 #pragma once
00008
00009 #include "Utils/Interfaces/IPlugin.hpp"

```

```

00010
00011 namespace cae
00012 {
00013
00014 /**
00015  * @interface IMouse
00016  * @brief Interface for mouse
00017  * @namespace cae
00018 /**
00019 class IMouse : public utl::IPlugin
00020 {
00021
00022     public:
00023         ~IMouse() override = default;
00024
00025     }; // interface IMouse
00026
00027 } // namespace cae

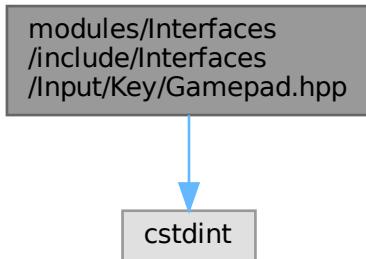
```

15.25 modules/Interfaces/include/Interfaces/Input/Key/Gamepad.hpp

File Reference

#include <cstdint>

Include dependency graph for Gamepad.hpp:



Namespaces

- namespace `cae`

Enumerations

- enum class `cae::GamepadButton` : `uint8_t` {
 `cae::A` = 0 , `cae::B` , `cae::X` , `cae::Y` ,
 `cae::Back` , `cae::Guide` , `cae::Start` , `cae::LThumb` ,
 `cae::RThumb` , `cae::LShoulder` , `cae::RShoulder` , `cae::DPadUp` ,
 `cae::DPadDown` , `cae::DPadLeft` , `cae::DPadRight` }
- enum class `cae::GamepadAxis` : `uint8_t` {
 `cae::LeftX` = 0 , `cae::LeftY` , `cae::RightX` , `cae::RightY` ,
 `cae::TriggerLeft` , `cae::TriggerRight` }

15.26 Gamepad.hpp

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

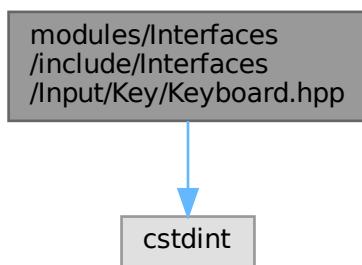
```
00001 /**
00002 /// @file Mouse.hpp
00003 /// @brief This file contains the gamepad keys
00004 /// @namespace cae
00005 /**
00006
00007 #pragma once
00008
00009 #include <cstdint>
00010
00011 namespace cae
00012 {
00013     enum class GamepadButton : uint8_t
00014     {
00015         A = 0,
00016         B,
00017         X,
00018         Y,
00019         Back,
00020         Guide,
00021         Start,
00022         LThumb,
00023         RThumb,
00024         LShoulder,
00025         RShoulder,
00026         DPadUp,
00027         DPadDown,
00028         DPadLeft,
00029         DPadRight
00030     };
00031
00032     enum class GamepadAxis : uint8_t
00033     {
00034         LeftX = 0,
00035         LeftY,
00036         RightX,
00037         RightY,
00038         TriggerLeft,
00039         TriggerRight
00040     };
00041 } // namespace cae
```

15.27 modules/Interfaces/include/Interfaces/Input/Key/Keyboard.hpp File Reference

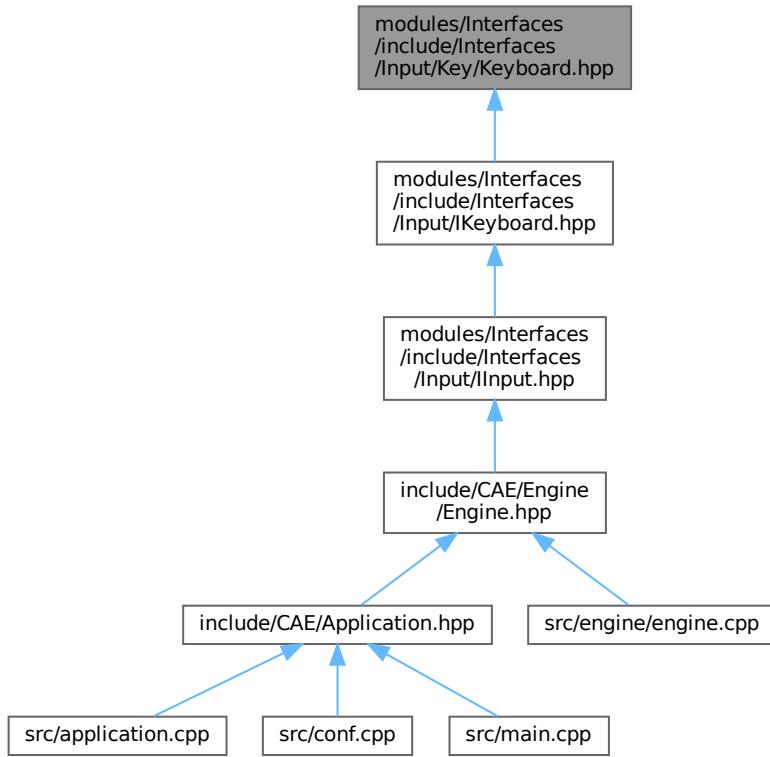
This file contains the keyboard keys.

```
#include <cstdint>
```

Include dependency graph for Keyboard.hpp:



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



Namespaces

- namespace `cae`

Enumerations

- enum `cae::KeyState` : `std::uint8_t` { `cae::Pressed` = 0 , `cae::Released` = 1 , `cae::Held` = 2 , `cae::Toggled` = 3 }
- enum class `cae::KeyCode` : `uint8_t` {
 `cae::A` , `cae::B` , `cae::C` , `cae::D` ,
 `cae::E` , `cae::F` , `cae::G` , `cae::H` ,
 `cae::I` , `cae::J` , `cae::K` , `cae::L` ,
 `cae::M` , `cae::N` , `cae::O` , `cae::P` ,
 `cae::Q` , `cae::R` , `cae::S` , `cae::T` ,
 `cae::U` , `cae::V` , `cae::W` , `cae::X` ,
 `cae::Y` , `cae::Z` , `cae::Num0` , `cae::Num1` ,
 `cae::Num2` , `cae::Num3` , `cae::Num4` , `cae::Num5` ,
 `cae::Num6` , `cae::Num7` , `cae::Num8` , `cae::Num9` ,
 `cae::Escape` , `cae::F1` , `cae::F2` , `cae::F3` ,
 `cae::F4` , `cae::F5` , `cae::F6` , `cae::F7` ,
 `cae::F8` , `cae::F9` , `cae::F10` , `cae::F11` ,
 `cae::F12` , `cae::Left` , `cae::Right` , `cae::Up` ,
 `cae::Down` , `cae::Home` , `cae::End` , `cae::PageUp` ,
 }

```

cae::PageDown , cae::Insert , cae::Delete , cae::Backspace ,
cae::Tab , cae::Enter , cae::Space , cae::LShift ,
cae::RShift , cae::LCtrl , cae::RCtrl , cae::LAlt ,
cae::RAlt , cae::LSuper , cae::RSuper , cae::Numpad0 ,
cae::Numpad1 , cae::Numpad2 , cae::Numpad3 , cae::Numpad4 ,
cae::Numpad5 , cae::Numpad6 , cae::Numpad7 , cae::Numpad8 ,
cae::Numpad9 , cae::NumpadAdd , cae::NumpadSubtract , cae::NumpadMultiply ,
cae::NumpadDivide , cae::CapsLock , cae::NumLock , cae::ScrollLock ,
cae::Count }
```

15.27.1 Detailed Description

This file contains the keyboard keys.

Definition in file [Keyboard.hpp](#).

15.28 Keyboard.hpp

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

```

00001 /**
00002 /**
00003 /**
00004 /**
00005 /**
00006
00007 #pragma once
00008
00009 #include <cstdint>
00010
00011 namespace cae
00012 {
00013
00014     enum KeyState : std::uint8_t
00015     {
00016         Pressed = 0,
00017         Released = 1,
00018         Held = 2,
00019         Toggled = 3,
00020     };
00021
00022     enum class KeyCode : uint8_t
00023     {
00024         A,
00025         B,
00026         C,
00027         D,
00028         E,
00029         F,
00030         G,
00031         H,
00032         I,
00033         J,
00034         K,
00035         L,
00036         M,
00037         N,
00038         O,
00039         P,
00040         Q,
00041         R,
00042         S,
00043         T,
00044         U,
00045         V,
00046         W,
00047         X,
00048         Y,
00049         Z,
00050
00051         Num0,
00052         Num1,
00053         Num2,
```

```

00054     Num3,
00055     Num4,
00056     Num5,
00057     Num6,
00058     Num7,
00059     Num8,
00060     Num9,
00061
00062     Escape,
00063     F1,
00064     F2,
00065     F3,
00066     F4,
00067     F5,
00068     F6,
00069     F7,
00070     F8,
00071     F9,
00072     F10,
00073     F11,
00074     F12,
00075
00076     Left,
00077     Right,
00078     Up,
00079     Down,
00080     Home,
00081     End,
00082     PageUp,
00083     PageDown,
00084     Insert,
00085     Delete,
00086     Backspace,
00087     Tab,
00088     Enter,
00089     Space,
00090
00091     LShift,
00092     RShift,
00093     LCtrl,
00094     RCtrl,
00095     LAlt,
00096     RAlt,
00097     LSUPER,
00098     RSUPER,
00099
00100    Numpad0,
00101    Numpad1,
00102    Numpad2,
00103    Numpad3,
00104    Numpad4,
00105    Numpad5,
00106    Numpad6,
00107    Numpad7,
00108    Numpad8,
00109    Numpad9,
00110    NumpadAdd,
00111    NumpadSubtract,
00112    NumpadMultiply,
00113    NumpadDivide,
00114
00115    CapsLock,
00116    NumLock,
00117    ScrollLock,
00118
00119    Count
00120  };
00121
00122 } // namespace cae

```

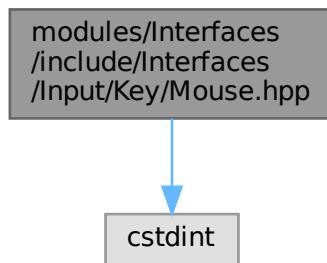
15.29 modules/Interfaces/include/Interfaces/Input/Key/Mouse.hpp

File Reference

This file contains the gamepad keys.

```
#include <cstdint>
```

Include dependency graph for Mouse.hpp:



Namespaces

- namespace `cae`

Enumerations

- enum class `cae::MouseButton` : `uint8_t` {
 `cae::Left` = 0 , `cae::Right` , `cae::Middle` , `cae::XButton1` ,
 `cae::XButton2` , `cae::WheelUp` , `cae::WheelDown` }

15.29.1 Detailed Description

This file contains the gamepad keys.

This file contains the mouse keys.

Definition in file [Mouse.hpp](#).

15.30 Mouse.hpp

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

```

00001 /**
00002 /// @file Mouse.hpp
00003 /// @brief This file contains the mouse keys
00004 /// @namespace cae
00005 /**
00006
00007 #pragma once
00008
00009 #include <cstdint>
00010
00011 namespace cae
00012 {
00013     enum class MouseButton : uint8_t
00014     {
00015         Left = 0,
00016         Right,
00017         Middle,
00018         XButton1,
00019         XButton2,
00020         WheelUp,
00021         WheelDown
00022     };
00023 } // namespace cae

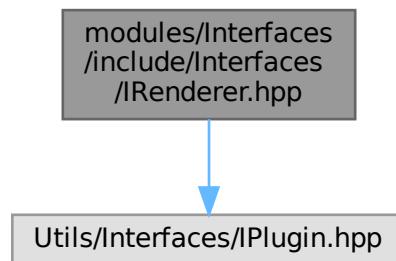
```

15.31 modules/Interfaces/include/Interfaces/IRenderer.hpp File Reference

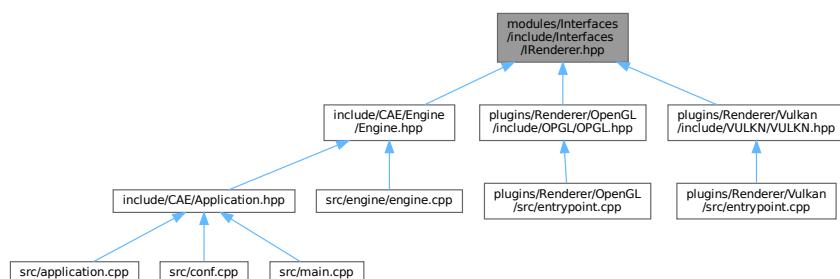
This file contains the Renderer interface.

```
#include "Utils/Interfaces/IPlugin.hpp"
```

Include dependency graph for IRenderer.hpp:



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



Classes

- interface [cae::IRenderer](#)
Interface for renderer.

Namespaces

- namespace [cae](#)

15.31.1 Detailed Description

This file contains the Renderer interface.

Definition in file [IRenderer.hpp](#).

15.32 IRenderer.hpp

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

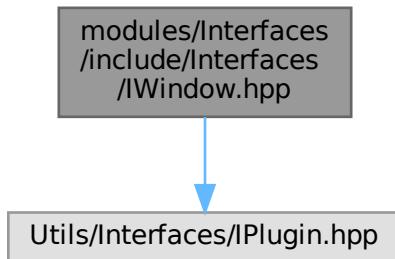
```
00001 /**
00002 /// @file IRenderer.hpp
00003 /// @brief This file contains the Renderer interface
00004 /// @namespace cae
00005 /**
00006
00007 #pragma once
00008
00009 #include "Utils/Interfaces/IPPlugin.hpp"
00010
00011 namespace cae
00012 {
00013
00014 /**
00015 /// @interface IRenderer
00016 /// @brief Interface for renderer
00017 /// @namespace cae
00018 /**
00019 class IRenderer : public utl::IPPlugin
00020 {
00021
00022     public:
00023         ~IRenderer() override = default;
00024
00025     virtual void initialize(void *nativeWindowHandle) = 0;
00026 }; // interface IRender
00027
00028 } // namespace cae
```

15.33 modules/Interfaces/include/Interfaces/IWindow.hpp File Reference

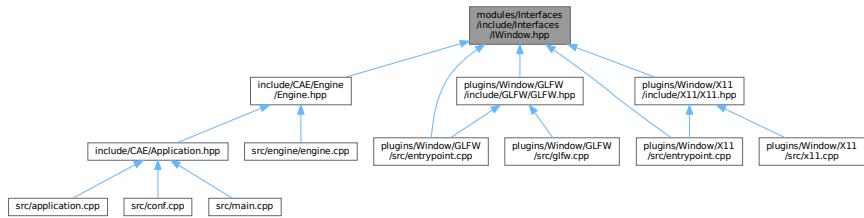
This file contains the Window interface.

#include "Utils/Interfaces/IPPlugin.hpp"

Include dependency graph for IWindow.hpp:



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



Classes

- struct `cae::WindowSize`
- struct `cae::NativeWindowHandle`
- interface `cae::IWindow`

Interface for window.

Namespaces

- namespace `cae`

15.33.1 Detailed Description

This file contains the Window interface.

Definition in file `IWindow.hpp`.

15.34 IWindow.hpp

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

```

00001 /**
00002  * @file IWindow.hpp
00003  * @brief This file contains the Window interface
00004  * @namespace cae
00005 /**
00006
00007 #pragma once
00008
00009 #include "Utils/Interfaces/IPlugin.hpp"
00010
00011 namespace cae
00012 {
00013
00014     struct WindowSize
00015     {
00016         uint16_t width;
00017         uint16_t height;
00018     };
00019
00020     struct NativeWindowHandle
00021     {
00022         void *window;
00023         void *display;
00024     };
00025
00026 /**
00027  * @interface IWindow
00028  * @brief Interface for window
  
```

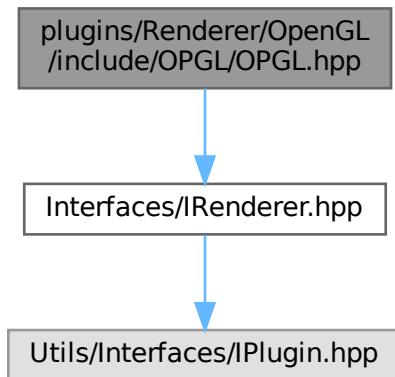
```
00029  /// @namespace cae
00030  ///
00031  class IWindow : public utl::IPlugin
00032  {
00033
00034      public:
00035          ~IWindow() override = default;
00036
00037          virtual bool create(const std::string &name, WindowSize size) = 0;
00038          virtual void close() = 0;
00039
00040          virtual NativeWindowHandle getNativeHandle() const = 0;
00041          virtual WindowSize getWindowSize() const = 0;
00042
00043          virtual bool setIcon(const std::string &path) const = 0;
00044
00045          virtual bool shouldClose() const = 0;
00046          virtual void pollEvents() = 0;
00047
00048          virtual bool wasResized() const = 0;
00049          virtual void resetResizedFlag() = 0;
00050
00051      private:
00052  }; // interface IWindow
00053
00054 } // namespace cae
```

15.35 plugins/Renderer/OpenGL/include/OPGL/OPGL.hpp File Reference

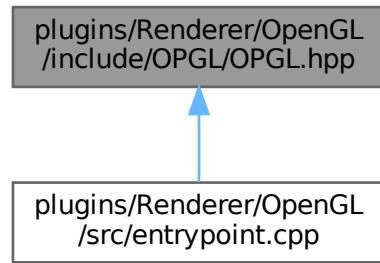
This file contains the OPGL class declaration.

```
#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](#)

15.35.1 Detailed Description

This file contains the OPGL class declaration.

Definition in file [OPGL.hpp](#).

15.36 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     [[nodiscard]] utl::PluginPlatform getPlatform() const override { return utl::PluginPlatform::ALL; }
00034
00035     void initialize(void *nativeWindowHandle) override {}
00036 }; // class OPGL
00037 } // namespace cae

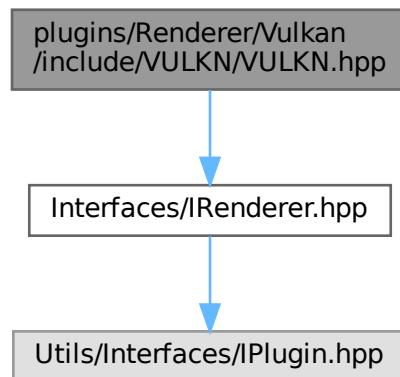
```

15.37 plugins/Renderer/Vulkan/include/VULKN/VULKN.hpp File Reference

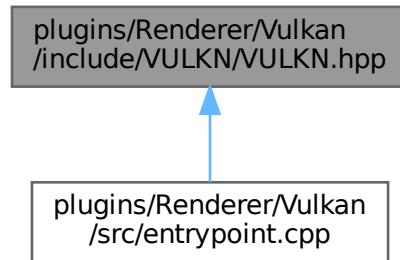
This file contains the VULKN class declaration.

#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](#)

15.37.1 Detailed Description

This file contains the VULKN class declaration.

Definition in file [VULKN.hpp](#).

15.38 VULKN.hpp

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

```
00001 /**
00002 /// @file VULKN.hpp
00003 /// @brief This file contains the VULKN 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 VULKN
00016 /// @brief Class for the Vulkan plugin
00017 /// @namespace cae
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         [[nodiscard]] utl::PluginPlatform getPlatform() const override { return utl::PluginPlatform::ALL; }
00034
00035         void initialize(void *nativeWindowHandle) override {}
00036     }; // class VULKN
00037 } // namespace cae
```

15.39 plugins/Renderer/Vulkan/src/VULKN.cpp File Reference

15.40 VULKN.cpp

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

00001

15.41 plugins/Audio/ALSA/src/entrypoint.cpp File Reference

15.42 entrypoint.cpp

[Go to the documentation of this file.](#)
00001

15.43 plugins/Audio/Core/src/entrypoint.cpp File Reference

15.44 entrypoint.cpp

[Go to the documentation of this file.](#)
00001

15.45 plugins/Audio/OpenAL/src/entrypoint.cpp File Reference

15.46 entrypoint.cpp

[Go to the documentation of this file.](#)
00001

15.47 plugins/Audio/Pulse/src/entrypoint.cpp File Reference

15.48 entrypoint.cpp

[Go to the documentation of this file.](#)
00001

15.49 plugins/Audio/XAudio2/src/entrypoint.cpp File Reference

15.50 entrypoint.cpp

[Go to the documentation of this file.](#)
00001

15.51 plugins/Input/Cocoa/src/entrypoint.cpp File Reference

15.52 entrypoint.cpp

[Go to the documentation of this file.](#)
00001

15.53 plugins/Input/Win32/src/entrypoint.cpp File Reference

15.54 entrypoint.cpp

[Go to the documentation of this file.](#)
00001

15.55 plugins/Input/X11/src/entrypoint.cpp File Reference

15.56 entrypoint.cpp

[Go to the documentation of this file.](#)
00001

15.57 plugins/Network/Asio/src/entrypoint.cpp File Reference

15.58 entrypoint.cpp

[Go to the documentation of this file.](#)
00001

15.59 plugins/Network/Posix/src/entrypoint.cpp File Reference

15.60 entrypoint.cpp

[Go to the documentation of this file.](#)
00001

15.61 plugins/Network/WinSock/src/entrypoint.cpp File Reference

15.62 entrypoint.cpp

[Go to the documentation of this file.](#)
00001

15.63 plugins/Renderer/DirectX12/src/entrypoint.cpp File Reference

15.64 entrypoint.cpp

[Go to the documentation of this file.](#)
00001

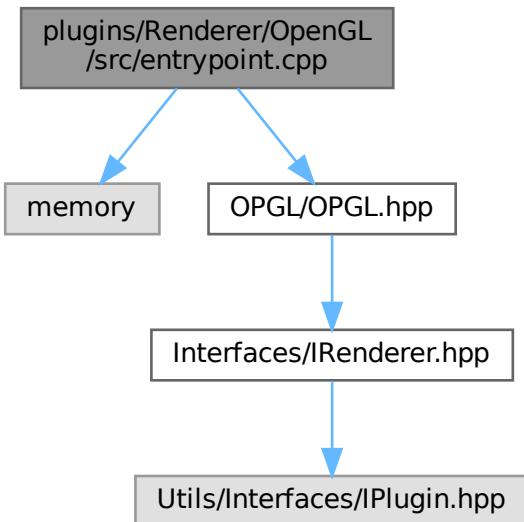
15.65 plugins/Renderer/Metal/src/entrypoint.cpp File Reference

15.66 entrypoint.cpp

[Go to the documentation of this file.](#)
00001

15.67 plugins/Renderer/OpenGL/src/entrypoint.cpp File Reference

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



Functions

- `cae::IRenderer * entryPoint ()`

15.67.1 Function Documentation

15.67.1.1 entryPoint()

```
cae::IRenderer * entryPoint ()
```

Definition at line 7 of file [entrypoint.cpp](#).

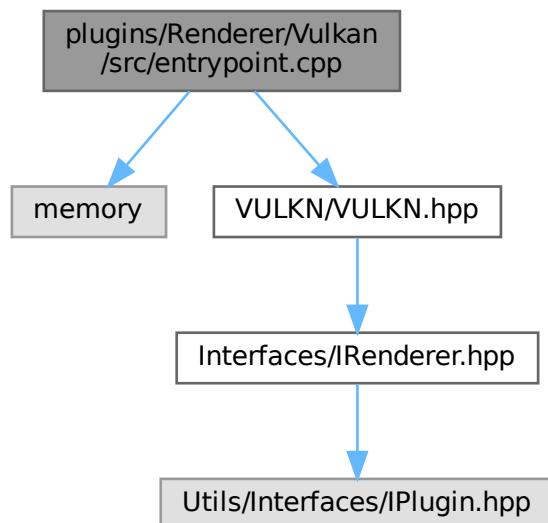
15.68 entrypoint.cpp

[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 }
```

15.69 plugins/Renderer/Vulkan/src/entrypoint.cpp File Reference

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



Functions

- `cae::IRenderer * entryPoint ()`

15.69.1 Function Documentation

15.69.1.1 entryPoint()

`cae::IRenderer * entryPoint ()`

Definition at line 7 of file [entrypoint.cpp](#).

15.70 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 }
```

15.71 plugins/Window/Cocoa/src/entrypoint.cpp File Reference

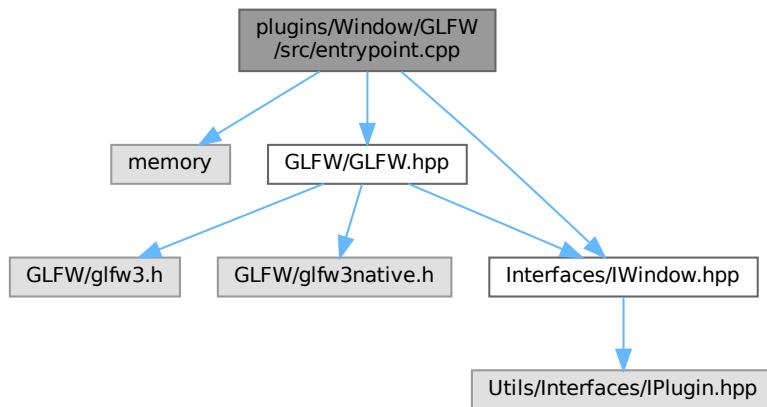
15.72 entrypoint.cpp

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

00001

15.73 plugins/Window/GLFW/src/entrypoint.cpp File Reference

```
#include <memory>
#include "GLFW/GLFW.hpp"
#include "Interfaces/IWindow.hpp"
Include dependency graph for entrypoint.cpp:
```



Functions

- `cae::IWindow * entryPoint ()`

15.73.1 Function Documentation

15.73.1.1 entryPoint()

```
cae::IWindow * entryPoint ()
```

Definition at line 8 of file [entrypoint.cpp](#).

15.74 entrypoint.cpp

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

```
00001 #include <memory>
00002
00003 #include "GLFW/GLFW.hpp"
00004 #include "Interfaces/IWindow.hpp"
00005
00006 extern "C"
00007 {
00008     cae::IWindow *entryPoint() { return std::make_unique<cae::GLFW>().release(); }
00009 }
```

15.75 plugins/Window/Win32/src/entrypoint.cpp File Reference

15.76 entrypoint.cpp

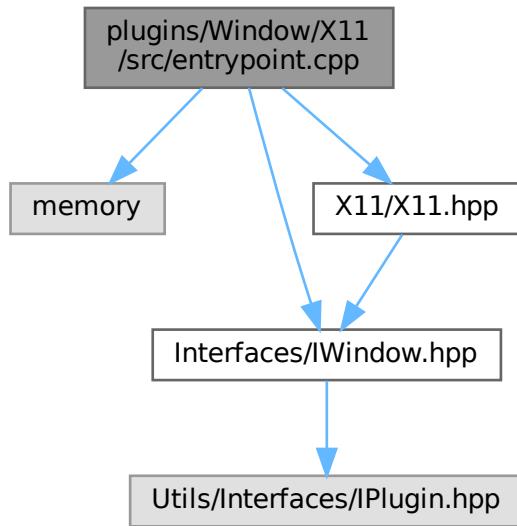
[Go to the documentation of this file.](#)

```
00001
```

15.77 plugins/Window/X11/src/entrypoint.cpp File Reference

```
#include <memory>
#include "Interfaces/IWindow.hpp"
#include "X11/X11.hpp"
```

Include dependency graph for entrypoint.cpp:



Functions

- `cae::IWindow * entryPoint ()`

15.77.1 Function Documentation

15.77.1.1 entryPoint()

`cae::IWindow * entryPoint ()`

Definition at line 8 of file [entrypoint.cpp](#).

15.78 entrypoint.cpp

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

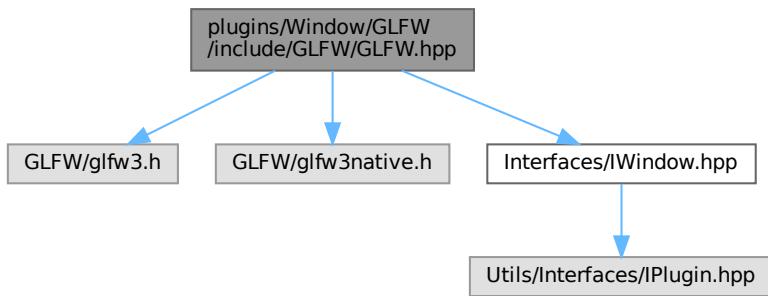
```

00001 #include <memory>
00002
00003 #include "Interfaces/IWindow.hpp"
00004 #include "X11/X11.hpp"
00005
00006 extern "C"
00007 {
00008     cae::IWindow *entryPoint() { return std::make_unique<cae::X11>().release(); }
00009 }
```

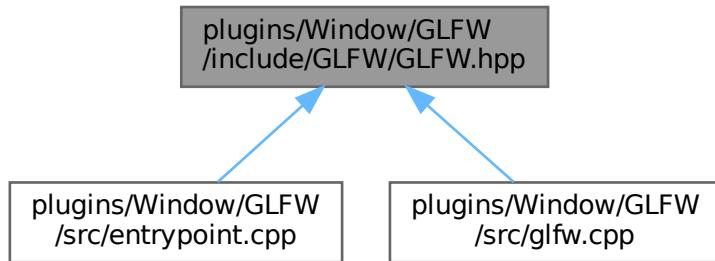
15.79 plugins/Window/GLFW/include/GLFW/GLFW.hpp File Reference

This file contains the GLFW class declaration.

```
#include <GLFW/glfw3.h>
#include <GLFW/glfw3native.h>
#include "Interfaces/IWindow.hpp"
Include dependency graph for GLFW.hpp:
```



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



Classes

- class [cae::GLFW](#)
Class for the [GLFW](#) plugin.

Namespaces

- namespace [cae](#)

15.79.1 Detailed Description

This file contains the GLFW class declaration.

Definition in file [GLFW.hpp](#).

15.80 GLFW.hpp

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

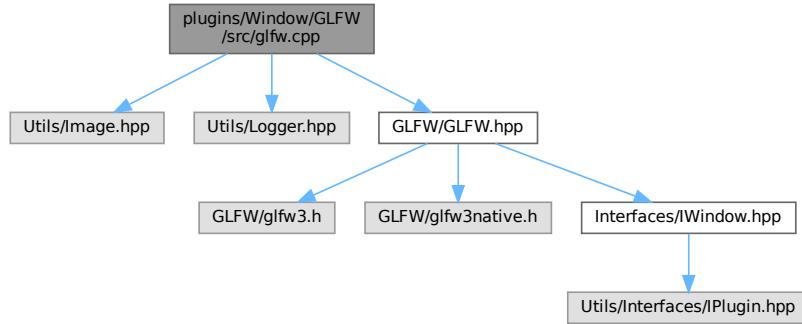
```

00001 /**
00002 /// @file GLFW.hpp
00003 /// @brief This file contains the GLFW class declaration
00004 /// @namespace cae
00005 /**
00006
00007 #pragma once
00008
00009 #include <GLFW/glfw3.h>
00010
00011 #if defined(_WIN32)
00012 #define GLFW_EXPOSE_NATIVE_WIN32
00013 #elif defined(__linux__)
00014 #define GLFW_EXPOSE_NATIVE_X11
00015 #elif defined(__APPLE__)
00016 #define GLFW_EXPOSE_NATIVE_COCOA
00017 #endif
00018 #include <GLFW/glfw3native.h>
00019
00020 #include "Interfaces/IWindow.hpp"
00021
00022 namespace cae
00023 {
00024
00025 /**
00026 /// @class GLFW
00027 /// @brief Class for the GLFW plugin
00028 /// @namespace cae
00029 /**
00030 class GLFW final : public IWindow
00031 {
00032
00033     public:
00034         GLFW() = default;
00035         ~GLFW() override = default;
00036
00037         GLFW(const GLFW &) = delete;
00038         GLFW &operator=(const GLFW &) = delete;
00039         GLFW(GLFW &&) = delete;
00040         GLFW &operator=(GLFW &&) = delete;
00041
00042         [[nodiscard]] std::string getName() const override { return "GLFW"; }
00043         [[nodiscard]] utl::PluginType getType() const override { return utl::PluginType::WINDOW; }
00044         [[nodiscard]] utl::PluginPlatform getPlatform() const override { return utl::PluginPlatform::ALL; }
00045
00046         bool create(const std::string &name, WindowSize size) override;
00047         void close() override;
00048
00049         [[nodiscard]] NativeWindowHandle getNativeHandle() const override;
00050         WindowSize getWindowSize() const override;
00051
00052         [[nodiscard]] bool setIcon(const std::string &path) const override;
00053
00054         [[nodiscard]] bool shouldClose() const override { return glfwWindowShouldClose(m_window) != 0; }
00055         void pollEvents() override { glfwPollEvents(); }
00056
00057         bool wasResized() const override { return m_frameBufferResized; }
00058         void resetResizedFlag() override { m_frameBufferResized = false; }
00059
00060     private:
00061         static void frameBufferResizeCallback(GLFWwindow *window, int width, int height);
00062
00063         GLFWwindow *m_window = nullptr;
00064         WindowSize m_frameBufferSize;
00065         bool m_frameBufferResized = false;
00066
00067     }; // class GLFW
00068 } // namespace cae

```

15.81 plugins/Window/GLFW/src/glfw.cpp File Reference

```
#include <Utils/Image.hpp>
#include <Utils/Logger.hpp>
#include "GLFW/GLFW.hpp"
Include dependency graph for glfw.cpp:
```



15.82 glfw.cpp

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

```
00001 #include <Utils/Image.hpp>
00002 #include <Utils/Logger.hpp>
00003
00004 #include "GLFW/GLFW.hpp"
00005
00006 void cae::GLFW::frameBufferResizeCallback(GLFWwindow *window, int width, int height)
00007 {
00008     auto *const self = static_cast<GLFW *>(glfwGetWindowUserPointer(window));
00009     self->m_frameBufferResized = true;
00010     self->m_frameBufferSize = {static_cast<uint16_t>(width), static_cast<uint16_t>(height)};
00011 }
00012
00013 bool cae::GLFW::create(const std::string &name, const WindowSize size)
00014 {
00015     m_window = nullptr;
00016     if (glfwInit() == 0)
00017     {
00018         return false;
00019     }
00020
00021     glfwWindowHint(GLFW_CLIENT_API, GLFW_NO_API);
00022     glfwWindowHint(GLFW_RESIZABLE, GLFW_TRUE);
00023     m_window = glfwCreateWindow(size.width, size.height, name.c_str(), nullptr, nullptr);
00024     if (m_window == nullptr)
00025     {
00026         glfwTerminate();
00027         utl::Logger::log("Failed to create GLFW window", utl::LogLevel::WARNING);
00028
00029         return false;
00030     }
00031     glfwSetWindowUserPointer(m_window, this);
00032     glfwSetFrameBufferSizeCallback(m_window, frameBufferResizeCallback);
00033
00034     return true;
00035 }
00036
00037 void cae::GLFW::close()
00038 {
00039     if (m_window)
00040     {
00041         glfwDestroyWindow(m_window);
00042         m_window = nullptr;
00043     }
  
```

```
00044     glfwTerminate();
00045 }
00046
00047 cae::WindowSize cae::GLFW::getWindowSize() const
00048 {
00049     int width = 0;
00050     int height = 0;
00051     glfwGetWindowSize(m_window, &width, &height);
00052     return {.width = static_cast<uint16_t>(width), .height = static_cast<uint16_t>(height)};
00053 }
00054
00055 cae::NativeWindowHandle cae::GLFW::getNativeHandle() const
00056 {
00057     NativeWindowHandle handle{};
00058 #if defined(_WIN32)
00059     handle.window = glfwGetWin32Window(m_glfwWindow);
00060     handle.display = GetModuleHandle(nullptr);
00061 #elif defined(__linux__)
00062     handle.window = (void*)(uintptr_t)glfwGetX11Window(m_window);
00063     handle.display = glfwGetX11Display();
00064 #elif defined(__APPLE__)
00065     handle.window = glfwGetCocoaWindow(m_glfwWindow);
00066     handle.display = nullptr;
00067 #endif
00068     return handle;
00069 }
00070
00071 bool cae::GLFW::setIcon(const std::string &path) const
00072 {
00073     static const utl::Image image(path);
00074     if (image.pixels == nullptr)
00075     {
00076         return false;
00077     }
00078     static const GLFWimage appIcon{.width = image.width, .height = image.height, .pixels = image.pixels};
00079     glfwSetWindowIcon(m_window, 1, &appIcon);
00080     return true;
00081 }
```

15.83 plugins/Audio/README.md File Reference

15.84 plugins/Input/README.md File Reference

15.85 plugins/Network/README.md File Reference

15.86 plugins/Renderer/README.md File Reference

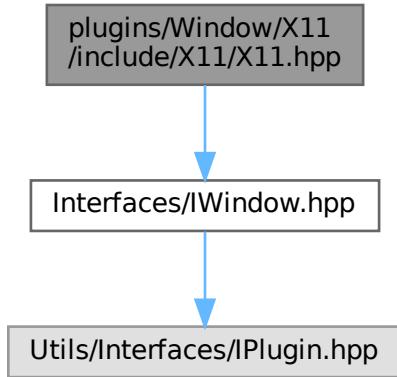
15.87 plugins/Window/README.md File Reference

15.88 README.md File Reference

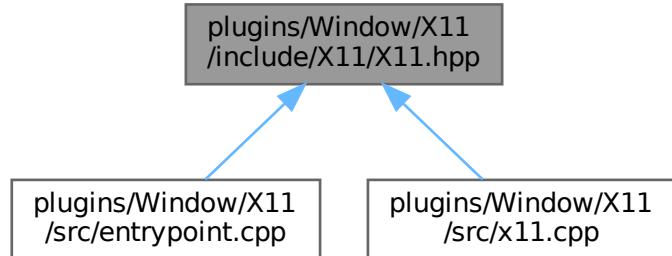
15.89 plugins/Window/X11/include/X11/X11.hpp File Reference

This file contains the X11 class declaration.

```
#include "Interfaces/IWindow.hpp"
Include dependency graph for X11.hpp:
```



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



Classes

- class [cae::X11](#)
Class for the [X11](#) plugin.

Namespaces

- namespace [cae](#)

15.89.1 Detailed Description

This file contains the X11 class declaration.

Definition in file [X11.hpp](#).

15.90 X11.hpp

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

```

00001 /**
00002 /// @file X11.hpp
00003 /// @brief This file contains the X11 class declaration
00004 /// @namespace cae
00005 /**
00006
00007 #pragma once
00008
00009 #include "Interfaces/IWindow.hpp"
00010
00011 namespace cae
00012 {
00013
00014 /**
00015 /// @class X11
00016 /// @brief Class for the X11 plugin
00017 /// @namespace cae
00018 /**
00019 class X11 final : public IWindow
00020 {
00021
00022     public:
00023         X11() = default;
00024         ~X11() override = default;
00025
00026         X11(const X11 &) = delete;
00027         X11 &operator=(const X11 &) = delete;
00028         X11(X11 &&) = delete;
00029         X11 &operator=(X11 &&) = delete;
00030
00031         [[nodiscard]] std::string getName() const override { return "X11"; }
00032         [[nodiscard]] utl::PluginType getType() const override { return utl::PluginType::WINDOW; }
00033         [[nodiscard]] utl::PluginPlatform getPlatform() const override { return utl::PluginPlatform::LINUX; }
00034
00035         bool create(const std::string &name, WindowSize size) override;
00036         void close() override;
00037
00038         [[nodiscard]] NativeWindowHandle getNativeHandle() const override
00039         {
00040             return {.window = reinterpret_cast<void *>(m_window), .display = reinterpret_cast<void *>(m_display)};
00041         }
00042         [[nodiscard]] WindowSize getWindowSize() const override;
00043
00044         [[nodiscard]] bool setIcon(const std::string &path) const override;
00045
00046         [[nodiscard]] bool shouldClose() const override;
00047         void pollEvents() override;
00048
00049         bool wasResized() const override { return m_frameBufferResized; }
00050         void resetResizedFlag() override { m_frameBufferResized = false; }
00051
00052     private:
00053         WindowSize m_frameBufferSize;
00054         mutable bool m_frameBufferResized = false;
00055
00056         Display *m_display = nullptr;
00057         Window m_window = 0;
00058         Atom m_wmDeleteMessage = 0;
00059         bool m_shouldClose = false;
00060
00061     }; // class GLFW
00062 } // namespace cae

```

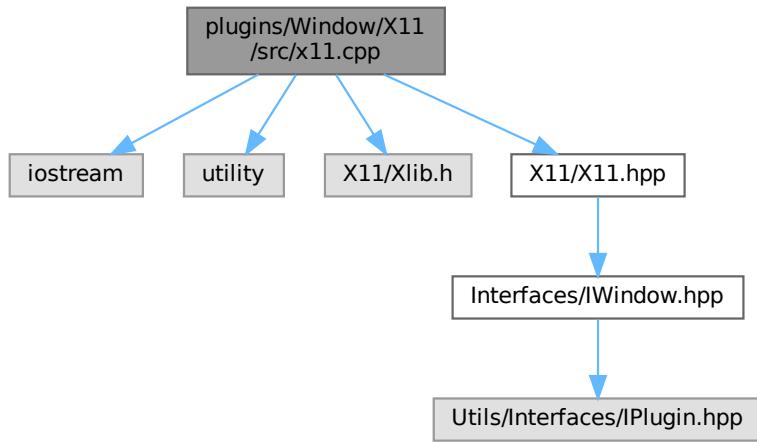
15.91 plugins/Window/X11/src/x11.cpp File Reference

```

#include <iostream>
#include <utility>
#include <X11/Xlib.h>
#include "X11/X11.hpp"

```

Include dependency graph for x11.cpp:



15.92 x11.cpp

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

```

00001 #include <iostream>
00002 #include <utility>
00003
00004 #include <X11/Xlib.h>
00005
00006 #include "X11/X11.hpp"
00007
00008 bool cae::X11::create(const std::string &name, const WindowSize size)
00009 {
00010     m_display = XOpenDisplay(nullptr);
00011     if (m_display == nullptr)
00012     {
00013         std::cerr << "[X11] Failed to open X display\n";
00014         return false;
00015     }
00016
00017     const int screen = DefaultScreen(m_display);
00018     const Window root = RootWindow(m_display, screen);
00019
00020     m_window = XCreateSimpleWindow(m_display, root, 0, 0, size.width, size.height, 1, BlackPixel(m_display, screen),
00021                                     WhitePixel(m_display, screen));
00022
00023     if (m_window == 0U)
00024     {
00025         std::cerr << "[X11] Failed to create X11 window\n";
00026         return false;
00027     }
00028
00029     XStoreName(m_display, m_window, name.c_str());
00030
00031     XSelectInput(m_display, m_window, ExposureMask | KeyPressMask | StructureNotifyMask);
00032
00033     m_wmDeleteMessage = XIInternAtom(m_display, "WM_DELETE_WINDOW", False);
00034     XSetWMProtocols(m_display, m_window, &m_wmDeleteMessage, 1);
00035
00036     XMapWindow(m_display, m_window);
00037     XFlush(m_display);
00038
00039     m_frameBufferSize = size;
00040
00041     return true;
00042 }
00043
00044 void cae::X11::close()
00045 {

```

```

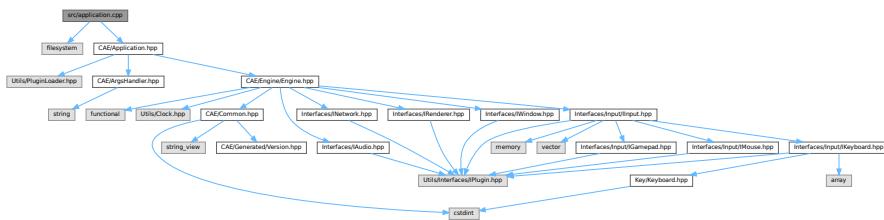
00046     if (m_display != nullptr && m_window != 0U)
00047     {
00048         XDestroyWindow(m_display, m_window);
00049         XCcloseDisplay(m_display);
00050         m_display = nullptr;
00051         m_window = 0;
00052     }
00053 }
00054
00055     cae::WindowSize cae::X11::getWindowSize() const
00056 {
00057     if (m_display == nullptr || m_window == 0U)
00058     {
00059         return m_frameBufferSize;
00060     }
00061
00062     XWindowAttributes attrs;
00063     XGetWindowAttributes(m_display, m_window, &attrs);
00064     return {.width = static_cast<uint16_t>(attrs.width), .height = static_cast<uint16_t>(attrs.height)};
00065 }
00066
00067     bool cae::X11::setIcon(const std::string &path) const
00068 {
00069     std::cerr << "[X11] setIcon() not implemented yet (" << path << ")\n";
00070     return false;
00071 }
00072
00073     bool cae::X11::shouldClose() const { return m_shouldClose; }
00074
00075     void cae::X11::pollEvents()
00076 {
00077     while (XPending(m_display) != 0)
00078     {
00079         XEvent event;
00080         XNextEvent(m_display, &event);
00081
00082         switch (event.type)
00083         {
00084             case Expose:
00085             {
00086                 XGCValues gcValues;
00087                 GC gc = XCreateGC(m_display, m_window, 0, &gcValues);
00088
00089                 XColor color;
00090                 const Colormap colormap = DefaultColormap(m_display, DefaultScreen(m_display));
00091                 color.red = 0x0000;
00092                 color.green = 0x0000;
00093                 color.blue = 0x0000;
00094                 color.flags = DoRed | DoGreen | DoBlue;
00095                 XAllocColor(m_display, colormap, &color);
00096
00097                 XSetForeground(m_display, gc, color.pixel);
00098
00099                 XFillRectangle(m_display, m_window, gc, 0, 0, m_frameBufferSize.width, m_frameBufferSize.height);
00100
00101                 XFreeGC(m_display, gc);
00102                 break;
00103             }
00104             case ConfigureNotify:
00105                 m_frameBufferResized = true;
00106                 m_frameBufferSize.width = event.xconfigure.width;
00107                 m_frameBufferSize.height = event.xconfigure.height;
00108                 break;
00109             case ClientMessage:
00110                 if (std::cmp_equal(event.xclient.data.l[0], m_wmDeleteMessage))
00111                 {
00112                     m_shouldClose = true;
00113                 }
00114                 break;
00115             default:
00116                 break;
00117         }
00118     }
00119
00120     XFlush(m_display);
00121 }

```

15.93 src/application.cpp File Reference

```
#include <filesystem>
#include "CAE/Application.hpp"
```

Include dependency graph for application.cpp:



Functions

- static std::vector< std::shared_ptr< utl::IPlugin > > loadPlugins (const std::unique_ptr< utl::PluginLoader > &loader)

15.93.1 Function Documentation

15.93.1.1 loadPlugins()

```
static std::vector< std::shared_ptr< utl::IPlugin > > loadPlugins (
    const std::unique_ptr< utl::PluginLoader > & loader) [static]
```

Definition at line 5 of file [application.cpp](#).

Referenced by [cae::Application::setupEngine\(\)](#).

Here is the caller graph for this function:



15.94 application.cpp

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

```
00001 #include <filesystem>
00002
00003 #include "CAE/Application.hpp"
00004
00005 static std::vector<std::shared_ptr<utl::IPPlugin>> loadPlugins(const std::unique_ptr<utl::PluginLoader> &loader)
00006 {
00007     const std::filesystem::path pluginDir{PLUGINS_DIR};
00008     std::vector<std::shared_ptr<utl::IPPlugin>> loadedPlugins;
00009
00010     for (const auto &entry : std::filesystem::directory_iterator(pluginDir))
00011     {
00012         if (!entry.is_regular_file() || entry.path().extension() != PLUGINS_EXTENSION)
00013         {
00014             continue;
00015         }
00016         const std::string pluginPath = entry.path().string();
00017         if (auto plugin = loader->loadPlugin<utl::IPPlugin>(pluginPath); plugin != nullptr)
00018         {
```

```

00019     loadedPlugins.push_back(plugin);
00020 }
00021 else
00022 {
00023     utl::Logger::log("Failed to load plugin: " + pluginPath, utl::LogLevel::WARNING);
00024 }
00025 }
00026 if (loadedPlugins.empty())
00027 {
00028     utl::Logger::log("No plugins loaded from directory: " + pluginDir.string(), utl::LogLevel::WARNING);
00029 }
00030
00031 return loadedPlugins;
00032 }
00033
00034 cae::Application::Application(const ArgsConfig &argsConfig, const EnvConfig &envConfig)
00035 : m_pluginLoader(std::make_unique<utl::PluginLoader>())
00036 {
00037     utl::Logger::log("PROJECT INFO:\n" + std::string(Message::VERSION_MSG), utl::LogLevel::INFO);
00038
00039 try
00040 {
00041     m_appConfig.envConfig = envConfig;
00042
00043     if (!argsConfig.config_path.empty())
00044     {
00045         m_appConfig.engineConfig = parseEngineConf(argsConfig.config_path);
00046     }
00047     setupEngine("Vulkan", "X11");
00048 }
00049 catch (const std::exception &e)
00050 {
00051     std::cerr << "Error: " << e.what() << '\n';
00052 }
00053 }
00054
00055 void cae::Application::setupEngine(const std::string &rendererName, const std::string &windowName)
00056 {
00057     std::shared_ptr<IWindow> windowPlugin = nullptr;
00058     std::shared_ptr<IRenderer> rendererPlugin = nullptr;
00059
00060     for (auto &plugin : loadPlugins(m_pluginLoader))
00061     {
00062         if (const auto renderer = std::dynamic_pointer_cast<IRenderer>(plugin))
00063         {
00064             if (renderer->getName() == rendererName)
00065             {
00066                 rendererPlugin = renderer;
00067             }
00068         }
00069         if (const auto window = std::dynamic_pointer_cast<IWindow>(plugin))
00070         {
00071             if (window->getName() == windowName)
00072             {
00073                 windowPlugin = window;
00074             }
00075         }
00076     }
00077     m_engine = std::make_unique<Engine>(
00078         m_appConfig.engineConfig, []() { return nullptr; }, []() { return nullptr; }, []() { return nullptr; },
00079         [rendererPlugin]() { return rendererPlugin; }, [windowPlugin]() { return windowPlugin; });
00080 }
00081
00082 void cae::Application::start() const { m_engine->run(); }
00083
00084 void cae::Application::stop()
00085 {
00086     m_engine->stop();
00087
00088     m_pluginLoader = nullptr;
00089     m_engine = nullptr;
00090 }

```

15.95 src/argsHandler.cpp File Reference

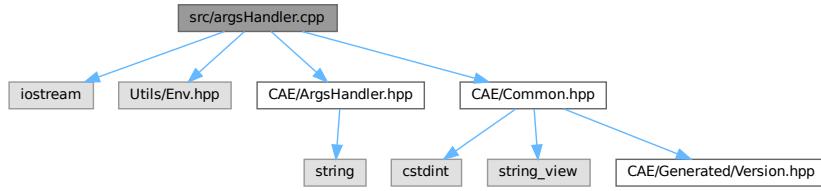
```

#include <iostream>
#include <Utils/Env.hpp>
#include "CAE/ArgsHandler.hpp"

```

```
#include "CAE/Common.hpp"
```

Include dependency graph for argsHandler.cpp:



15.96 argsHandler.cpp

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

```

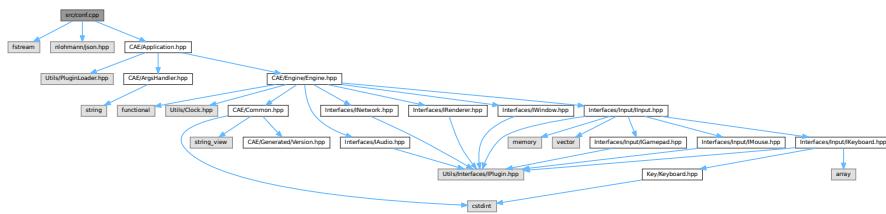
00001 #include <iostream>
00002
00003 #include <Utils/Env.hpp>
00004
00005 #include "CAE/ArgsHandler.hpp"
00006 #include "CAE/Common.hpp"
00007
00008 cae::ArgsConfig cae::ArgsHandler::ParseArgs(const int argc, const char *const *argv)
00009 {
00010     ArgsConfig config;
00011     config.run = true;
00012
00013     if (argc <= 1)
00014     {
00015         return config;
00016     }
00017
00018     for (int i = 1; i < argc; ++i)
00019     {
00020         std::string arg = argv[i];
00021
00022         if (arg == "-h" || arg == "--help")
00023         {
00024             std::cout << Message::HELP_MSG;
00025             config.run = false;
00026             return config;
00027         }
00028         if (arg == "-v" || arg == "--version")
00029         {
00030             std::cout << Message::VERSION_MSG;
00031             config.run = false;
00032             return config;
00033         }
00034         if (arg == "-c" || arg == "--config")
00035         {
00036             if (i + 1 >= argc)
00037             {
00038                 throw std::runtime_error("Missing value for argument " + arg);
00039             }
00040
00041             config.config_path = argv[++i];
00042         }
00043         else
00044         {
00045             throw std::runtime_error("Unknown argument: " + arg + ". Use -h or --help to see available options.");
00046         }
00047     }
00048
00049     return config;
00050 }
00051
00052 cae::EnvConfig cae::ArgsHandler::ParseEnv(const char *const *envp)
00053 {
00054     EnvConfig config;
00055     const auto envMap = utl::getEnvMap(envp);
00056
00057     if (envMap.contains("USER"))
00058     {
  
```

```

00059     config.user_name = envMap.at("USER");
00060 }
00061 if (envMap.contains("PWD"))
00062 {
00063     config.pwd = envMap.at("PWD");
00064 }
00065
00066 return config;
00067 }
```

15.97 src/conf.cpp File Reference

```
#include <fstream>
#include <nlohmann/json.hpp>
#include "CAE/Application.hpp"
Include dependency graph for conf.cpp:
```



Typedefs

- using `json` = `nlohmann::json`

15.97.1 Typedef Documentation

15.97.1.1 json

```
using json = nlohmann::json
```

Definition at line 8 of file `conf.cpp`.

15.98 conf.cpp

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

```

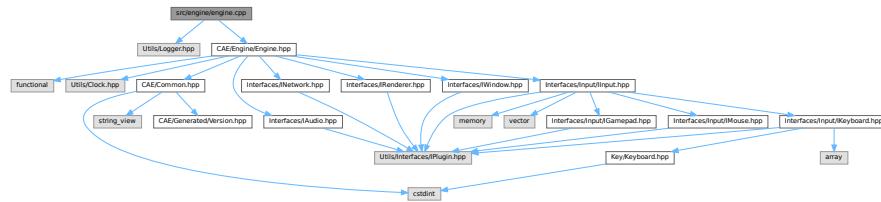
00001 #include <fstream>
00002
00003 #include <nlohmann/json.hpp>
00004
00005 #include "CAE/Application.hpp"
00006
00007 namespace fs = std::filesystem;
00008 using json = nlohmann::json;
00009
00010 cae::EngineConfig cae::Application::parseEngineConf(const std::string &path)
00011 {
00012     const fs::path filePath(path);
00013     if (!fs::exists(filePath))
00014     {
00015         std::cerr << "Config file not found: " << filePath << '\n';
00016         return {};
00017 }
```

```

00017    }
00018    if (!fs::is_regular_file(filePath))
00019    {
00020        std::cerr << "Config path is not a regular file: " << filePath << '\n';
00021        return {};
00022    }
00023
00024    std::ifstream file(filePath);
00025    if (!file.is_open())
00026    {
00027        std::cerr << "Failed to open config file: " << filePath << '\n';
00028        return {};
00029    }
00030
00031    json j;
00032    try
00033    {
00034        file >> j;
00035    }
00036    catch (const json::parse_error &e)
00037    {
00038        std::cerr << "Failed to parse JSON config (" << filePath << "): " + std::string(e.what()) << '\n';
00039        return {};
00040    }
00041    cae::EngineConfig config;
00042    utl::Logger::log("Loading config: " + filePath.string(), utl::LogLevel::INFO);
00043    if (j.contains("audio"))
00044    {
00045        const auto &audio = j["audio"];
00046        if (audio.contains("masterVolume") && audio["masterVolume"].is_number())
00047        {
00048            config.audio_master_volume = audio["masterVolume"];
00049        }
00050        if (audio.contains("muted") && audio["muted"].is_boolean())
00051        {
00052            config.audio_muted = audio["muted"];
00053        }
00054    }
00055    if (j.contains("network"))
00056    {
00057        const auto &network = j["network"];
00058        if (network.contains("host") && network["host"].is_string())
00059        {
00060            config.network_host = network["host"];
00061        }
00062        if (network.contains("port") && network["port"].is_number_unsigned())
00063        {
00064            config.network_port = network["port"];
00065        }
00066    }
00067    if (j.contains("renderer"))
00068    {
00069        const auto &renderer = j["renderer"];
00070        if (renderer.contains("vsync") && renderer["vsync"].is_boolean())
00071        {
00072            config.renderer_vsync = renderer["vsync"];
00073        }
00074        if (renderer.contains("frameRateLimit") && renderer["frameRateLimit"].is_number_unsigned())
00075        {
00076            config.renderer_frame_rate_limit = renderer["frameRateLimit"];
00077        }
00078    }
00079    if (j.contains("window"))
00080    {
00081        const auto &window = j["window"];
00082        if (window.contains("width") && window["width"].is_number_unsigned())
00083        {
00084            config.window_width = window["width"];
00085        }
00086        if (window.contains("height") && window["height"].is_number_unsigned())
00087        {
00088            config.window_height = window["height"];
00089        }
00090        if (window.contains("fullscreen") && window["fullscreen"].is_boolean())
00091        {
00092            config.window_fullscreen = window["fullscreen"];
00093        }
00094        if (window.contains("name") && window["name"].is_string())
00095        {
00096            config.window_name = window["name"];
00097        }
00098    }
00099
00100    return config;
00101 }
```

15.99 src/engine/engine.cpp File Reference

```
#include <Utils/Logger.hpp>
#include "CAE/Engine/Engine.hpp"
Include dependency graph for engine.cpp:
```



15.100 engine.cpp

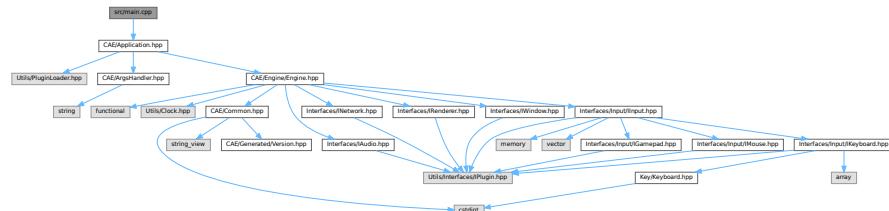
[Go to the documentation of this file.](#)

```
00001 #include <Utils/Logger.hpp>
00002
00003 #include "CAE/Engine/Engine.hpp"
00004
00005 cae::Engine::Engine(const EngineConfig &config, const std::function<std::shared_ptr<IAudio>()> &audioFactory,
00006     const std::function<std::shared_ptr<IInput>()> &inputFactory,
00007     const std::function<std::shared_ptr<INetwork>()> &networkFactory,
00008     const std::function<std::shared_ptr<IRenderer>()> &rendererFactory,
00009     const std::function<std::shared_ptr<IWindow>()> &windowFactory)
00010 : m_audioPlugin(audioFactory()), m_inputPlugin(inputFactory()), m_networkPlugin(networkFactory()),
00011     m_rendererPlugin(rendererFactory()), m_windowPlugin(windowFactory()),
00012     m_clock(std::make_unique<utl::Clock>())
00013 {
00014     utl::Logger::log("Loading engine with configuration:", utl::LogLevel::INFO);
00015     utl::Logger::log("\tAudio master volume: " + std::to_string(config.audio_master_volume), utl::LogLevel::INFO);
00016     utl::Logger::log("\tAudio muted: " + std::string(config.audio_muted ? "true" : "false"), utl::LogLevel::INFO);
00017     utl::Logger::log("\tNetwork host: " + config.network_host, utl::LogLevel::INFO);
00018     utl::Logger::log("\tNetwork port: " + std::to_string(config.network_port), utl::LogLevel::INFO);
00019     utl::Logger::log("\tRenderer vsync: " + std::string(config.renderer_vsync ? "true" : "false"), utl::LogLevel::INFO);
00020     utl::Logger::log("\tRenderer frame rate limit: " + std::to_string(config.renderer_frame_rate_limit),
00021                     utl::LogLevel::INFO);
00022     utl::Logger::log("\tWindow width: " + std::to_string(config.window_width), utl::LogLevel::INFO);
00023     utl::Logger::log("\tWindow height: " + std::to_string(config.window_height), utl::LogLevel::INFO);
00024     utl::Logger::log("\tWindow fullscreen: " + std::string(config.window_fullscreen ? "true" : "false"),
00025                     utl::LogLevel::INFO);
00026     utl::Logger::log("\tWindow name: " + config.window_name, utl::LogLevel::INFO);
00027     m_windowPlugin->create(config.window_name, {.width = config.window_width, .height = config.window_height});
00028 }
00029 void cae::Engine::run() const
00030 {
00031     while (!m_windowPlugin->shouldClose())
00032     {
00033         m_windowPlugin->pollEvents();
00034     }
00035 }
00036
00037 void cae::Engine::stop()
00038 {
00039     utl::Logger::log("Stopping engine...", utl::LogLevel::INFO);
00040     m_windowPlugin->close();
00041
00042     m_audioPlugin = nullptr;
00043     m_inputPlugin = nullptr;
00044     m_networkPlugin = nullptr;
00045     m_rendererPlugin = nullptr;
00046     m_windowPlugin = nullptr;
00047 }
```

15.101 src/main.cpp File Reference

```
#include "CAE/Application.hpp"
```

Include dependency graph for main.cpp:



Functions

- int [main](#) (const int argc, const char *const *argv, const char *const *envp)

15.101.1 Function Documentation

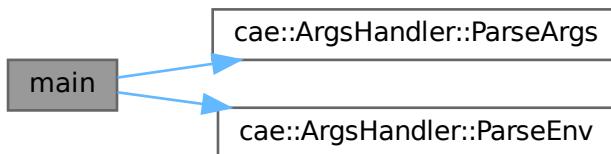
15.101.1.1 [main\(\)](#)

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

Definition at line 3 of file [main.cpp](#).

References [cae::ArgsHandler::ParseArgs\(\)](#), [cae::ArgsHandler::ParseEnv\(\)](#), and [cae::ArgsConfig::run](#).

Here is the call graph for this function:



15.102 main.cpp

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

```
00001 #include "CAE/Application.hpp"
00002
00003 int main(const int argc, const char *const *argv, const char *const *envp)
00004 {
00005     std::unique_ptr<cae::Application> app = nullptr;
00006
00007     utl::Logger::init();
00008     try
00009     {
00010         cae::ArgsConfig argsConfig = cae::ArgsHandler::ParseArgs(argc, argv);
00011         cae::EnvConfig envConfig = cae::ArgsHandler::ParseEnv(envp);
00012         if (!argsConfig.run)
00013         {
00014             return EXIT_SUCCESS;
00015         }
00016         app = std::make_unique<cae::Application>(argsConfig, envConfig);
00017         app->start();
00018         app->stop();
00019     }
00020     catch (const std::exception &e)
00021     {
00022         std::cerr << "Error: " << e.what() << '\n';
00023         return EXIT_FAILURE;
00024     }
00025     catch (...)
00026     {
00027         std::cerr << "Unknown error occurred\n";
00028         return EXIT_FAILURE;
00029     }
00030     return EXIT_SUCCESS;
00031 }
```


Index

~Application
 cae::Application, 39

~ArgsHandler
 cae::ArgsHandler, 44

~Engine
 cae::Engine, 47

~GLFW
 cae::GLFW, 58

~IAudio
 cae::IAudio, 63

~IGamepad
 cae::IGamepad, 65

~IInput
 cae::IInput, 67

~IKeyboard
 cae::IKeyboard, 69

~IMouse
 cae::IMouse, 71

~INetwork
 cae::INetwork, 73

~IRenderer
 cae::IRenderer, 75

~IWindow
 cae::IWindow, 78

~OPGL
 cae::OPGL, 83

~VULKN
 cae::VULKN, 87

~X11
 cae::X11, 92

A
 cae, 27

APP_EXTENSION
 Common.hpp, 102

Application
 cae::Application, 39, 40

application.cpp
 loadPlugins, 146

ArgsHandler
 cae::ArgsHandler, 44

audio_master_volume
 cae::EngineConfig, 52

audio_muted
 cae::EngineConfig, 52

B
 cae, 27, 28

Back
 cae, 27

Backspace
 cae, 29

BUILD_TYPE
 Version.hpp, 106

C
 cae, 28

cae, 1, 25
 A, 27
 B, 27, 28
 Back, 27
 Backspace, 29
 C, 28
 CapsLock, 29
 Count, 29
 D, 28
 Delete, 29
 Down, 29
 DPadDown, 27
 DPadLeft, 27
 DPadRight, 27
 DPadUp, 27
 E, 28
 End, 29
 Enter, 29
 Escape, 28
 F, 28
 F1, 28
 F10, 28
 F11, 28
 F12, 28
 F2, 28
 F3, 28
 F4, 28
 F5, 28
 F6, 28
 F7, 28
 F8, 28
 F9, 28
 G, 28
 GamepadAxis, 27
 GamepadButton, 27
 Guide, 27
 H, 28
 Held, 30
 Home, 29
 I, 28
 Insert, 29
 J, 28
 K, 28

KeyCode, 27
 KeyState, 29
 L, 28
 LAlt, 29
 LCtrl, 29
 Left, 28, 30
 LeftX, 27
 LeftY, 27
 LShift, 29
 LShoulder, 27
 LSUPER, 29
 LThumb, 27
 M, 28
 Middle, 30
 MouseButton, 30
 N, 28
 Num0, 28
 Num1, 28
 Num2, 28
 Num3, 28
 Num4, 28
 Num5, 28
 Num6, 28
 Num7, 28
 Num8, 28
 Num9, 28
 NumLock, 29
 Numpad0, 29
 Numpad1, 29
 Numpad2, 29
 Numpad3, 29
 Numpad4, 29
 Numpad5, 29
 Numpad6, 29
 Numpad7, 29
 Numpad8, 29
 Numpad9, 29
 NumpadAdd, 29
 NumpadDivide, 29
 NumpadMultiply, 29
 NumpadSubtract, 29
 O, 28
 P, 28
 PageDown, 29
 PageUp, 29
 Pressed, 30
 Q, 28
 R, 28
 RAlt, 29
 RCtrl, 29
 Released, 30
 Right, 29, 30
 RightX, 27
 RightY, 27
 RShift, 29
 RShoulder, 27
 RSUPER, 29
 RThumb, 27
 S, 28
 ScrollLock, 29
 Space, 29
 Start, 27
 T, 28
 Tab, 29
 Toggled, 30
 TriggerLeft, 27
 TriggerRight, 27
 U, 28
 Up, 29
 V, 28
 W, 28
 WheelDown, 30
 WheelUp, 30
 X, 27, 28
 XButton1, 30
 XButton2, 30
 Y, 27, 28
 Z, 28
 cae::AppConfig, 35
 engineConfig, 37
 envConfig, 37
 cae::Application, 37
 ~Application, 39
 Application, 39, 40
 m_appConfig, 41
 m_engine, 41
 m_pluginLoader, 42
 operator=, 40
 parseEngineConf, 40
 setupEngine, 40
 start, 41
 stop, 41
 cae::ArgsConfig, 42
 config_path, 43
 run, 43
 cae::ArgsHandler, 43
 ~ArgsHandler, 44
 ArgsHandler, 44
 operator=, 44
 ParseArgs, 45
 ParseEnv, 45
 cae::Audio, 30
 MUTED, 30
 VOLUME, 30
 cae::Engine, 46
 ~Engine, 47
 Engine, 47, 48
 getAudio, 48
 getClock, 48
 getInput, 48
 getNetwork, 48
 getRenderer, 48
 getWindow, 49
 m_audioPlugin, 49
 m_clock, 49
 m_inputPlugin, 50

m_networkPlugin, 50
m_rendererPlugin, 50
m_windowPlugin, 50
operator=, 49
run, 49
stop, 49
cae::EngineConfig, 51
 audio_master_volume, 52
 audio_muted, 52
 network_host, 52
 network_port, 52
 renderer_frame_rate_limit, 52
 renderer_vsync, 52
 windowFullscreen, 53
 window_height, 53
 window_name, 53
 window_width, 53
cae::EnvConfig, 54
 pwd, 54
 user_name, 54
cae::GLFW, 55
 ~GLFW, 58
 close, 59
 create, 59
 frameBufferResizeCallback, 59
 getName, 59
 getNativeHandle, 59
 getPlatform, 59
 getType, 60
 getWindowSize, 60
 GLFW, 58
 m_frameBufferResized, 61
 m_frameBufferSize, 61
 m_window, 61
 operator=, 60
 pollEvents, 60
 resetResizedFlag, 60
 setIcon, 60
 shouldClose, 61
 wasResized, 61
cae::IAudio, 62
 ~IAudio, 63
cae::IGamepad, 64
 ~IGamepad, 65
cae::IInput, 65
 ~IInput, 67
 getGamepads, 67
 getKeyboard, 67
 getMouse, 67
 m_gamepads, 67
 m_keyboard, 67
 m_mouse, 67
cae::IKeyboard, 68
 ~IKeyboard, 69
 isKeyPressed, 69
 m_keyMap, 70
cae::IMouse, 70
 ~IMouse, 71
cae::INetwork, 72
 ~INetwork, 73
 connect, 73
cae::IRenderer, 73
 ~IRenderer, 75
 initialize, 75
cae::IWindow, 76
 ~IWindow, 78
 close, 78
 create, 78
 getNativeHandle, 78
 getWindowSize, 78
 pollEvents, 78
 resetResizedFlag, 78
 setIcon, 78
 shouldClose, 79
 wasResized, 79
cae::Message, 31
 HELP_MSG, 31
 VERSION_MSG, 31
cae::NativeWindowHandle, 79
 display, 80
 window, 80
cae::Network, 31
 HOST, 31
 PORT, 31
cae::OPGL, 80
 ~OPGL, 83
 getName, 83
 getPlatform, 83
 getType, 83
 initialize, 83
 operator=, 84
 OPGL, 83
cae::Plugins, 32
cae::Plugins::Name, 32
 RENDERER_OPENGL, 32
 RENDERER_VULKAN, 32
 WINDOW(GLFW, 32
cae::Renderer, 32
 FRAME_RATE_LIMIT, 33
 VSYNC, 33
cae::User, 33
 NAME, 33
cae::VULKN, 84
 ~VULKN, 87
 getName, 87
 getPlatform, 87
 getType, 87
 initialize, 87
 operator=, 88
 VULKN, 87
cae::Window, 33
 FULLSCREEN, 33
 HEIGHT, 33
 NAME, 34
 WIDTH, 34
cae::WindowSize, 88

height, 89
 width, 89
 cae::X11, 89
 ~X11, 92
 close, 93
 create, 93
 getName, 93
 getNativeHandle, 93
 getPlatform, 93
 getType, 93
 getWindowSize, 94
 m_display, 95
 m_frameBufferResized, 95
 m_frameBufferSize, 95
 m_shouldClose, 95
 m_window, 96
 m_wmDeleteMessage, 96
 operator=, 94
 pollEvents, 94
 resetResizedFlag, 94
 setIcon, 94
 shouldClose, 95
 wasResized, 95
 X11, 92
 CapsLock
 cae, 29
 close
 cae::GLFW, 59
 cae::IWindow, 78
 cae::X11, 93
 Commit Norms, 13
 Common.hpp
 APP_EXTENSION, 102
 conf.cpp
 json, 149
 config_path
 cae::ArgsConfig, 43
 connect
 cae::INetwork, 73
 CONTRIBUTING.md, 97
 Count
 cae, 29
 create
 cae::GLFW, 59
 cae::IWindow, 78
 cae::X11, 93

 D
 cae, 28
 Delete
 cae, 29
 display
 cae::NativeWindowHandle, 80
 Down
 cae, 29
 DPadDown
 cae, 27
 DPadLeft
 cae, 27

 DPadRight
 cae, 27
 DPadUp
 cae, 27

 E
 cae, 28
 End
 cae, 29
 Engine
 cae::Engine, 47, 48
 engineConfig
 cae::AppConfig, 37
 Enter
 cae, 29
 entryPoint
 entrypoint.cpp, 133, 134, 136, 137
 entrypoint.cpp
 entryPoint, 133, 134, 136, 137
 envConfig
 cae::AppConfig, 37
 Escape
 cae, 28

 F
 cae, 28
 F1
 cae, 28
 F10
 cae, 28
 F11
 cae, 28
 F12
 cae, 28
 F2
 cae, 28
 F3
 cae, 28
 F4
 cae, 28
 F5
 cae, 28
 F6
 cae, 28
 F7
 cae, 28
 F8
 cae, 28
 F9
 cae, 28
 FRAME_RATE_LIMIT
 cae::Renderer, 33
 frameBufferResizeCallback
 cae::GLFW, 59
 FULLSCREEN
 cae::Window, 33

 G
 cae, 28

GamepadAxis
 cae, 27
GamepadButton
 cae, 27
getAudio
 cae::Engine, 48
getClock
 cae::Engine, 48
getGamepads
 cae::IInput, 67
getInput
 cae::Engine, 48
getKeyboard
 cae::IInput, 67
getMouse
 cae::IInput, 67
getName
 cae::GLFW, 59
 cae::OPGL, 83
 cae::VULKN, 87
 cae::X11, 93
getNativeHandle
 cae::GLFW, 59
 cae::IWindow, 78
 cae::X11, 93
getNetwork
 cae::Engine, 48
getPlatform
 cae::GLFW, 59
 cae::OPGL, 83
 cae::VULKN, 87
 cae::X11, 93
getRenderer
 cae::Engine, 48
getType
 cae::GLFW, 60
 cae::OPGL, 83
 cae::VULKN, 87
 cae::X11, 93
getWindow
 cae::Engine, 49
getWindowSize
 cae::GLFW, 60
 cae::Iwindow, 78
 cae::X11, 94
GIT_COMMIT_HASH
 Version.hpp, 106
GIT_TAG
 Version.hpp, 106
GLFW
 cae::GLFW, 58
Guide
 cae, 27

H
 cae, 28
HEIGHT
 cae::Window, 33
height

cae::WindowSize, 89
Held
 cae, 30
HELP_MSG
 cae::Message, 31
Home
 cae, 29
HOST
 cae::Network, 31

I
 cae, 28
include/CAE/Application.hpp, 97, 98
include/CAE/ArgsHandler.hpp, 99, 100
include/CAE/Common.hpp, 100, 102
include/CAE/Engine/Engine.hpp, 103, 104
include/CAE/Generated/Version.hpp, 105, 107
initialize
 cae::IRenderer, 75
 cae::OPGL, 83
 cae::VULKN, 87
Insert
 cae, 29
isKeyPressed
 cae::IKeyboard, 69

J
 cae, 28
json
 conf.cpp, 149

K
 cae, 28
KeyCode
 cae, 27
KeyState
 cae, 29

L
 cae, 28
LAlt
 cae, 29
LCtrl
 cae, 29
Left
 cae, 28, 30
LeftX
 cae, 27
LeftY
 cae, 27
LICENSE, 15
LICENSE.md, 107
loadPlugins
 application.cpp, 146
LShift
 cae, 29
LShoulder
 cae, 27
LSuper

cae, 29
LThumb
 cae, 27
M
 cae, 28
m_appConfig
 cae::Application, 41
m_audioPlugin
 cae::Engine, 49
m_clock
 cae::Engine, 49
m_display
 cae::X11, 95
m_engine
 cae::Application, 41
m_frameBufferResized
 cae::GLFW, 61
 cae::X11, 95
m_frameBufferSize
 cae::GLFW, 61
 cae::X11, 95
m_gamepads
 cae::IInput, 67
m_inputPlugin
 cae::Engine, 50
m_keyboard
 cae::IInput, 67
m_keyMap
 cae::IKeyboard, 70
m_mouse
 cae::IInput, 67
m_networkPlugin
 cae::Engine, 50
m_pluginLoader
 cae::Application, 42
m_rendererPlugin
 cae::Engine, 50
m_shouldClose
 cae::X11, 95
m_window
 cae::GLFW, 61
 cae::X11, 96
m_windowPlugin
 cae::Engine, 50
m_wmDeleteMessage
 cae::X11, 96
main
 main.cpp, 152
main.cpp
 main, 152
Middle
 cae, 30
modules/include/Interfaces/IAudio.hpp, 107, 108
modules/include/Interfaces/INetwork.hpp, 109, 110
modules/include/Interfaces/Input/IGamepad.hpp, 29, 110, 112
modules/include/Interfaces/Input/IInput.hpp, 112, 113
modules/include/Interfaces/Input/IKeyboard.hpp, 114, 115
modules/include/Interfaces/Input/IMouse.hpp, 116, 117
modules/include/Interfaces/Input/Key/Gamepad.hpp, 118, 119
modules/include/Interfaces/Input/Key/Keyboard.hpp, 119, 121
modules/include/Interfaces/Input/Key/Mouse.hpp, 122, 123
modules/include/Interfaces/IRenderer.hpp, 124, 125
modules/include/Interfaces/IWindow.hpp, 125, 126
MouseButton
 cae, 30
MUTED
 cae::Audio, 30
N
 cae, 28
NAME
 cae::User, 33
 cae::Window, 34
network_host
 cae::EngineConfig, 52
network_port
 cae::EngineConfig, 52
Num0
 cae, 28
Num1
 cae, 28
Num2
 cae, 28
Num3
 cae, 28
Num4
 cae, 28
Num5
 cae, 28
Num6
 cae, 28
Num7
 cae, 28
Num8
 cae, 28
Num9
 cae, 28
NumLock
 cae, 29
Numpad0
 cae, 29
Numpad1
Numpad2
Numpad3

cae, 29
Numpad4
cae, 29
Numpad5
cae, 29
Numpad6
cae, 29
Numpad7
cae, 29
Numpad8
cae, 29
Numpad9
cae, 29
NumpadAdd
cae, 29
NumpadDivide
cae, 29
NumpadMultiply
cae, 29
NumpadSubtract
cae, 29

O
cae, 28
operator=
cae::Application, 40
cae::ArgsHandler, 44
cae::Engine, 49
cae::GLFW, 60
cae::OPGL, 84
cae::VULKN, 88
cae::X11, 94
OPGL
cae::OPGL, 83

P
cae, 28
PageDown
cae, 29
PageUp
cae, 29
ParseArgs
cae::ArgsHandler, 45
parseEngineConf
cae::Application, 40
ParseEnv
cae::ArgsHandler, 45
plugins/Audio/ALSA/src/entrypoint.cpp, 131
plugins/Audio/Core/src/entrypoint.cpp, 131
plugins/Audio/OpenAL/src/entrypoint.cpp, 131
plugins/Audio/Pulse/src/entrypoint.cpp, 131
plugins/Audio/README.md, 141
plugins/Audio/XAudio2/src/entrypoint.cpp, 131
plugins/Input/Cocoa/src/entrypoint.cpp, 131
plugins/Input/README.md, 141
plugins/Input/Win32/src/entrypoint.cpp, 132
plugins/Input/X11/src/entrypoint.cpp, 132
plugins/Network/Asio/src/entrypoint.cpp, 132
plugins/Network/Posix/src/entrypoint.cpp, 132

plugins/Network/README.md, 141
plugins/Network/WinSock/src/entrypoint.cpp,
132
plugins/Renderer/DirectX12/src/entrypoint.cpp,
132
plugins/Renderer/Metal/src/entrypoint.cpp, 133
plugins/Renderer/OpenGL/include/OPGL/OPGL.hpp,
127, 128
plugins/Renderer/OpenGL/src/entrypoint.cpp,
133, 134
plugins/Renderer/README.md, 141
plugins/Renderer/Vulkan/include/VULKN/VULKN.hpp,
129, 130
plugins/Renderer/Vulkan/src/entrypoint.cpp, 134,
135
plugins/Renderer/Vulkan/src/VULKN.cpp, 130
plugins/Window/Cocoa/src/entrypoint.cpp, 135
plugins/Window/GLFW/include/GLFW/GLFW.hpp,
138, 139
plugins/Window/GLFW/src/entrypoint.cpp, 135,
136
plugins/Window/GLFW/src/glfw.cpp, 140
plugins/Window/README.md, 141
plugins/Window/Win32/src/entrypoint.cpp, 136
plugins/Window/X11/include/X11/X11.hpp, 141,
143
plugins/Window/X11/src/entrypoint.cpp, 136, 137
plugins/Window/X11/src/x11.cpp, 143, 144

pollEvents
cae::GLFW, 60
cae::IWindow, 78
cae::X11, 94

PORT
cae::Network, 31

Pressed
cae, 30

PROJECT_NAME
Version.hpp, 106

PROJECT_VERSION
Version.hpp, 106

PROJECT_VERSION_MAJOR
Version.hpp, 106

PROJECT_VERSION_MINOR
Version.hpp, 106

PROJECT_VERSION_PATCH
Version.hpp, 106

pwd
cae::EnvConfig, 54

Q
cae, 28

R
cae, 28

RAlt
cae, 29

RCtrl
cae, 29

README, 3, 5, 7, 9, 11

README.md, 141
 Released
 cae, 30
 renderer_frame_rate_limit
 cae::EngineConfig, 52
 RENDERER_OPENGL
 cae::Plugins::Name, 32
 renderer_vsync
 cae::EngineConfig, 52
 RENDERER_VULKAN
 cae::Plugins::Name, 32
 resetResizedFlag
 cae::GLFW, 60
 cae::IWindow, 78
 cae::X11, 94
 Right
 cae, 29, 30
 RightX
 cae, 27
 RightY
 cae, 27
 RShift
 cae, 29
 RShoulder
 cae, 27
 RSuper
 cae, 29
 RThumb
 cae, 27
 run
 cae::ArgsConfig, 43
 cae::Engine, 49

S
 cae, 28
 ScrollLock
 cae, 29
 setIcon
 cae::GLFW, 60
 cae::IWindow, 78
 cae::X11, 94
 setupEngine
 cae::Application, 40
 shouldClose
 cae::GLFW, 61
 cae::IWindow, 79
 cae::X11, 95
 Space
 cae, 29
 src/application.cpp, 145, 146
 src/argsHandler.cpp, 147, 148
 src/conf.cpp, 149
 src/engine/engine.cpp, 151
 src/main.cpp, 152, 153
 Start
 cae, 27
 start
 cae::Application, 41
 stop
 cae::Application, 41
 cae::Engine, 49

cae::Application, 41
 cae::Engine, 49

T
 cae, 28
 Tab
 cae, 29
 Toggled
 cae, 30
 TriggerLeft
 cae, 27
 TriggerRight
 cae, 27

U
 cae, 28

Up
 cae, 29

user_name
 cae::EnvConfig, 54

V
 cae, 28

Version.hpp
 BUILD_TYPE, 106
 GIT_COMMIT_HASH, 106
 GIT_TAG, 106
 PROJECT_NAME, 106
 PROJECT_VERSION, 106
 PROJECT_VERSION_MAJOR, 106
 PROJECT_VERSION_MINOR, 106
 PROJECT_VERSION_PATCH, 106

VERSION_MSG
 cae::Message, 31

VOLUME
 cae::Audio, 30

VSYNC
 cae::Renderer, 33

VULKN
 cae::VULKN, 87

W
 cae, 28

wasResized
 cae::GLFW, 61
 cae::IWindow, 79
 cae::X11, 95

WheelDown
 cae, 30

WheelUp
 cae, 30

WIDTH
 cae::Window, 34

width
 cae::WindowSize, 89

window
 cae::NativeWindowHandle, 80

windowFullscreen
 cae::EngineConfig, 53

WINDOW_GLFW
 cae::Plugins::Name, [32](#)
window_height
 cae::EngineConfig, [53](#)
window_name
 cae::EngineConfig, [53](#)
window_width
 cae::EngineConfig, [53](#)

X
 cae, [27](#), [28](#)

X11
 cae::X11, [92](#)

XButton1
 cae, [30](#)

XButton2
 cae, [30](#)

Y
 cae, [27](#), [28](#)

Z
 cae, [28](#)