Oficina Framework

1.3

Generated by Doxygen 1.8.13

Contents

1 Oficina Framework

1.1 About the project

Oficina is a multiplatform Framework for 2D Indie Games, developed by Lucas Vieira, using the language C++ and the libraries Simple DirectMedia Layer 2.0 and OpenGL. Oficina is focused on bringing several features in a simple and easy way, like, for example:

- Static, unecessary-instances-free system;
- Multiplatform for Windows and Linux;
- Multiarchitecture;
- Dynamic playlist system for ambient music;
- Basic system for sound effects;
- Use of graphics processing resources;
- Support for multiple input types;
- Object management system;
- Memory management system, making it easier to dealocate it, given an exit signal.

For more info, please refer to Installation and Usage.

2 Installation and Usage

2.1 Usage

2.1.1 Controls

If there's a controller connected to the computer, Oficina will detect it as soon as it's initiated. The controls used inside the program are referenced from the Xbox 360 buttons bindings.
Warning: If your controller supports vibration, it'll be activated. The use of vibration and its support detection can be seen on the debugger, as well as a big number of other features.

2.1.2 Debugger

Oficina's internal debugger brings a series of information related to input, screen refresh rate, processor and memory usage (the memory usage is only monitored on Linux system), as well as many other things.
To bring up the debugger, press F1.
To alternate between complete and compact versions, press F2.

2.1.3 Arguments

There are some command line arguments which you can use to setup a few settings.
These arguments can be entered by executing one of the game's startup scripts (e.g. rungame.sh/bat) from the command line (Linux and Windows alike), via a shortcut (Windows), or via another startup script which calls one of the startup scripts with arguments (Linux and Windows alike, too).

2.1.3.1 Multiplayer

It is possible to play using multiplayer on the current Oficina test build.
It is necessary, though, to enable this resource using a command line or a terminal; The option for this resource on the pause menu is unavailable.
THIS ARGUMENT WILL ONLY RUN ON THE TEST PROGRAM.
On Linux, all you need to do is navigate to the game folder and start it with:

```
./rungame.sh -networktarget=XXX.XXX.XXX.XXX:PPPP
```

And on Windows, you can run:

```
rungame.bat -networktarget=XXX.XXX.XXX.XXX:PPPP
```

Where XXX.XXX.XXX.XXX is the IP address of the destination computer, followed by the port you wish to deliver your packages on the other player's computer.
You can also specify which port you want to listen to by using the following argument:

```
./rungame.sh -port=PPPP
```

```
rungame.bat -port=PPPP
```

Where PPPP is the desired port you wish to open.
THIS ARGUMENT WILL ALSO ONLY RUN ON THE TEST PROGRAM.
You can include this and other arguments in the same molds of the network target argument. You can also specify multiple arguments.
Notice that, in some cases, it might be necessary to configure your firewall for the connection to work. Failure on package delivering could be seen on terminal or console, if you execute the program for these means.

2.1.3.2 Fullscreen

To run the game on fullscreen mode, run Oficina including the fullscreen argument.

```
./rungame.sh --fullscreen
```

```
rungame.bat --fullscreen
```

On Linux systems, Oficina will attempt to make the window size fit your current native resolution, but internal OpenGL context resolution will remain the same. This is done because of SDL's screen resolution issues when attempting fullscreen with a resolution that is different from your screen monitor's resolution.

2.2 Installation

2.2.1 Dependancies

Oficina Framework depends on the following libraries:

- SDL2 (SDL 2.0)
- SDL2_image (SDL_image 2.0)
- SDL2_mixer (SDL_mixer 2.0)
- physfs (PhysicsFS)
- OpenGL

Make sure you have these libraries installed when building the library or any application depending on it.
Normally, these libraries will be available on your repository. They are all opensource, so you will not have difficulty trying to find them.
There may be subdependancies on libogg/libvorbis or zlib, so make sure you also install them correctly.

2.2.2 Building

Building Oficina is rather easy.
Having Oficina's source code, navigate to the source folder, and create the required subfolders for building.

```
$ mkdir bin lib obj
```

Then, all you need to do is build it with Makefile, using the following command:

```
$ make
```

Make sure you are using GCC/MinGW as your compiler, and also make sure you have all the needed dependancies installed.
The objects for building Oficina's static libraries will be stored in obj folder, and the static library itself will be stored in lib.
Notice that, running the command "make" alone will only generate the library with debug mode active (liboficinad.a).
If you wish to build Oficina on Release target, you must run the make command like this:

```
$ make target=release
```

This will create liboficina.a on lib/ folder instead.
Make sure to clean the objects generated from the previous compilation, else you may face building problems!
To clean the objects, run:

```
$ make clean
```

2.2.2.1 Test Build Application

You can also build the Test Build application to test Oficina.
To do so, execute the following command:

```
$ make test
```

If you wish to run the Test Build application under Release target, add the target flag to the end of this command:

```
$ make test target=release
```

2.2.3 Installing

Installing Oficina Framework on a Linux environment is also easy.
All you need to do is navigate to the folder and run:

```
$ make install
```

You may need superuser privileges to run this action.
This will build both Debug and Release targets of Oficina and copy them and the required headers to your local prefix' (normally /usr/local) include and lib subfolders.
If you wish to change the prefix to, say, /usr, run the installation using the following command:

```
$ make install prefix=/usr
```

Windows users will want to build each target separately, then manually copy the contents of lib/ into their MinGW/lib folder, and also copy both the folder OficinaFramework (the one containing only headers) and the file GonglyScriptStructure.h into MinGW/include.

2.2.4 Uninstalling

You can also uninstall Oficina Framework using the Makefile.
Just run

```
$ make uninstall
```

You may need superuser privileges to run this action.
If you installed it on another prefix, say, /usr, then you may also want to explicitly show that to the Makefile by running:

```
$ make uninstall prefix=/usr
```

Windows users will want to manually remove both files liboficina.a and liboficinad.a from MinGW/lib folder, the folder MinGW/include/OficinaFramework and the file MinGW/include/GonglyScriptStructure.h.

2.2.5 Building Your Own Application

To build your own application, you must use the correct build flags.
After correctly installing Oficina and its dependancies, you will want to compile your own code like this:

```
# On Linux
$ g++ --std=c++11 your_source_code_and_objects_here -o output_executable_name -loficinad -lSDL2 -
      lSDL2_image -lSDL2_mixer -lGL -lphysfs
# On Windows
$ g++ --std=c++11 your_source_code_and_objects_here -o output_executable_name -loficinad -lmingw32
       -lSDL2main -lSDL2 -lSDL2_image -lSDL2_mixer -lopengl32 -lwsock32 -lphysfs
```

Notice the following:

- Oficina's statically linked through the command -loficinad. Notice that, the presence of the character "d" in the end of the flag refers to a debug compilation. To link for release targets, use the flag -loficina instead.
- Oficina's linking flags MUST STRICTLY come before the flags shown on the commands above, or else there will be undefined reference errors, although order of the dependancy flags are not important, as long as they come after Oficina's (with exception of the -lmingw32 -lSDL2main flags, which must also come between Oficina's and the rest);
- Oficina is built under C++11, so your code must also be built under C++11 specification;
- Oficina's debug static library is built with DEBUG_ENABLED flag active, but it isn't activated by default in your application. To achieve that, add the flag -DDEBUG_ENABLED to your build command.

If you're using an IDE like Code::Blocks, you'll want to configure your building targets to use these linking targets accordingly. This is normally done on C::B by right-clicking the project name on the Projects tree and clicking "Build Options". Creating Defines (like the DEBUG_ENABLED case) can also be done from there.

2.2.6 Command Line Arguments

Any game compiled with Oficina Framework has a number of fixed argument flags that can be run from any application.
These flags are configuration flags, normally for setting framerate and resolution configuration, and are normally used on the Gongly Launcher.
The flags are:

```
--fullscreen
```

Sets the window to full screen size.

```
--resolution=x
```

Sets both Window Size and Resolution to "x".
Refer to Screen Resolution or Window Size for possible values, as they're strictly related.

```
--vsync
```

Enables vertical synchro.
Refer to Enable Vertical Synchro for more details.

```
--fps=x
```

Sets default framerate for the application to "x".
Refer to Framerate for possible values.

```
--diagnostics
```

Enables system diagnostics, if application was compiled in debug mode and has a font binded to the debug window.

```
--no_multithreaded_audio
```

Forces program to run audio loop on the same rendering and logic thread.
This may lower CPU consumption, but may also cause BGM loops to not work properly.

```
--no-bgm
```

Mutes ALL background music.

```
--no-sfx
```

Mutes ALL sound effects.

```
--version
-v
```

Forces application to output which version of Oficina it was compiled with then halts the application.

3 GameArgs Available Options

This page contains examples and usage of GameArgs when invoking EngineCore's Initialize method.

3.1 Example

```
#include "OficinaFramework.h"
using namespace OficinaFramework;

int main(int argc, char** argv)
{
    // Create list of arguments
    std::list<string>* GameArgs = new std::list<string>;
    GameArgs->push_front("gamename=Your Game Name Here");
    GameArgs->push_front("iconpath=icon_name_on_data_folder");
    GameArgs->push_front("resolution=720p");
    GameArgs->push_front("framerate=60Hz");
    GameArgs->push_front("vsync");
    GameArgs->push_front("enable_network");
    GameArgs->push_front("enable_multithreaded_audio");

    // This argument will only be activated if compiled
    // with a debug flag
    #ifdef DEBUG_ENABLED
    GameArgs->push_front("enable_diagnostics");
    #endif

    // Initialize the core
    EngineCore::Initialize(argc, argv, GameArgs);

    // Run game
    return EngineCore::DoGameLoop();
}
```

As you can see above, using GameArgs is fairly simple. All you need to do is create a list of arguments and pass them to the initializer, and they will be handled accordingly.
If a GameArg is not recognized, it'll be prompted to the console, regardless of the build nature.

3.2 GameName

Sets the name of the game (including on game window).
Usage:

```
gamename=x
```

x: String containing the name of the game

3.3 Screen Resolution

Sets the rendering resolution.
Usage:

```
resolution=<Width>x<Height>
```

or

```
resolution=<literal>
```

x: Resolution to be given to the renderer.
Use the syntax <Width>x<Height> after the = sign to specify a custom resolution.

3.3.1 Some default values and literals:

3.3.1.1 16:9 resolutions

- 720p/hd/HD -> 1280x720, 720p
- 768p -> 1366x768
- 900p -> 1600x900
- 1080p/fhd/FullHD -> 1920x1080, 1080p
- 1440p -> 2560x1440
- 2160p/4k/4K -> 3840x2160, 4K

3.3.1.2 4:3 resolutions

- 640x480 -> 640x480
- 800x600 -> 800x600
- 1024x768 -> 1024x768
- 1600x1200 -> 1600x1200

3.4 Window Size

This GameArg is now deprecated.
Use

See also

Screen Resolution instead.
Usage:

```
windowsize=x
```

3.5 Framerate

Sets the window to a fixed refresh rate.
Do not give this GameArg if you intend to use variable rate.
Usage:

```
framerate=x
```

x: Desired refresh rate.

3.5.1 Possible values:

- 05/05FPS/05fps/05hz/05Hz -> 5 hertz (test purposes only!)
- 15/15FPS/15fps/15hz/15Hz -> 15 hertz
- 20/20FPS/20fps/20hz/20Hz -> 20 hertz
- 24/24FPS/24fps/24hz/24Hz -> 24 hertz
- 30/30FPS/30fps/30hz/30Hz -> 30 hertz
- 60/60FPS/60fps/60hz/60Hz -> 60 hertz

3.6 Enable Networking

Enable network communication.
Only use this if any network communication is intended.
Usage:

```
enable_network
```

3.7 Enable Diagnostics

Enable diagnostics mode.
Only enable this for debug, for there could be performance drops.
Usage:

```
enable_diagnostics
```

3.8 Set Icon

Sets the icon for the application.
Usage:

```
iconpath=x
```

x: Path to the icon .png file on Game Path.

3.9 Enable Multithreaded Audio

Enables the audio thread (recomended).
If inactive or failed to initialize, audio management will be embeded on game loop.
Usage:

```
enable_multithreaded_audio
```

3.10 Mute Background Music

Mutes background music.
Usage:

```
mute_bgm
```

3.11 Enable Vertical Synchro

Enables VSync.
Usage:

```
vsync
```

4 Todo List

Class OficinaFramework::EngineCore
- Add listeners system for user-based modules.

Class OficinaFramework::RenderingSystem
- Ditch fixed pipeline completely in favor of using shaders.

5 Module Index

5.1 Modules

Here is a list of all modules:

6 Namespace Index

6.1 Namespace List

Here is a list of all namespaces with brief descriptions:

7 Hierarchical Index

7.1 Class Hierarchy

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

8 Class Index

8.1 Class List

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

9 File Index

9.1 File List

Here is a list of all files with brief descriptions:

10 Module Documentation

10.1 Data Types

Data types with controlled size, ideal for bitwise operations. Based on definitions for integer types.

Typedefs

- typedef uint8_t byte
  8-bit unsigned type.
- typedef uint16_t word
  16-bit unsigned type.
- typedef uint32_t dword
  32-bit unsigned type.
- typedef uint64_t qword
  64-bit unsigned type.
- typedef int8_t byte_s
  8-bit signed type.
- typedef int16_t word_s
  16-bit signed type.
- typedef int32_t dword_s
  32-bit signed type.
- typedef int64_t qword_s
  64-bit signed type.

10.1.1 Detailed Description

Data types with controlled size, ideal for bitwise operations. Based on definitions for integer types.

10.1.2 Typedef Documentation

10.1.2.1 byte

```
typedef uint8_t byte
```

8-bit unsigned type.

10.1.2.2 byte_s

```
typedef int8_t byte_s
```

8-bit signed type.

10.1.2.3 dword

```
typedef uint32_t dword
```

32-bit unsigned type.

10.1.2.4 dword_s

```
typedef int32_t dword_s
```

32-bit signed type.

10.1.2.5 qword

```
typedef uint64_t qword
```

64-bit unsigned type.

10.1.2.6 qword_s

```
typedef int64_t qword_s
```

64-bit signed type.

10.1.2.7 word

```
typedef uint16_t word
```

16-bit unsigned type.

10.1.2.8 word_s

```
typedef int16_t word_s
```

16-bit signed type.

10.2 Color Terminal Printing Defines

Definitions for using with printf, to get colored output.

Macros

- #define KNRM "\x1B[0m"

  Normal ASCII Terminal color.
- #define KRED "\x1B[31m"

  Red ASCII Terminal color.
- #define KGRN "\x1B[32m"

  Green ASCII Terminal color.
- #define KYEL "\x1B[33m"

  Yellow ASCII Terminal color.
- #define KBLU "\x1B[34m"

  Blue ASCII Terminal color.
- #define KMAG "\x1B[35m"

  Magenta ASCII Terminal color.
- #define KCYN "\x1B[36m"

  Cyan ASCII Terminal color.
- #define KWHT "\x1B[37m"

  White ASCII Terminal color.
- #define KRESET "\033[0m"

  Resets any ASCII terminal color.

10.2.1 Detailed Description

Definitions for using with printf, to get colored output.

Warning

This should only have effect on ∗nix-based environments.

10.2.2 Macro Definition Documentation

10.2.2.1 KBLU

```
#define KBLU "\x1B[34m"
```

Blue ASCII Terminal color.

10.2.2.2 KCYN

```
#define KCYN "\x1B[36m"
```

Cyan ASCII Terminal color.

10.2.2.3 KGRN

```
#define KGRN "\x1B[32m"
```

Green ASCII Terminal color.

10.2.2.4 KMAG

```
#define KMAG "\x1B[35m"
```

Magenta ASCII Terminal color.

10.2.2.5 KNRM

```
#define KNRM "\x1B[0m"
```

Normal ASCII Terminal color.

10.2.2.6 KRED

```
#define KRED "\x1B[31m"
```

Red ASCII Terminal color.

10.2.2.7 KRESET

```
#define KRESET "\033[0m"
```

Resets any ASCII terminal color.

10.2.2.8 KWHT

```
#define KWHT "\x1B[37m"
```

White ASCII Terminal color.

10.2.2.9 KYEL

```
#define KYEL "\x1B[33m"
```

Yellow ASCII Terminal color.

10.3 Color Operations

Color-related bitwise operations.

Macros

- #define CLEARMASK_NOT 0x000Fu

  Mask to be binded to the opposite of another color mask for byte extraction.
- #define HIGHLIGHTMASK 0x0777u

  Mask to be used to form highlight mode.
- #define NEXTCOLOR(x) (x = (x << 4))

  Shifts the current byte to the left, so another component is added.
- #define MIXCOLOR(x, y) (x = (x | y))

  Sets the last byte of the mask to the byte component value to be set.
- #define SHADOWMODE(x) (x >> 1)

  Sets a color to its shadowed mode.
- #define HIGHLIGHTMODE(x) ((x >> 1) + HIGHLIGHTMASK)

  Sets a color to its highlighted mode.
- #define GETRCOLOR(x) ((x ^ ~(~x | (CLEARMASK_NOT << 8))) >> 8)

  Gets red component of a color.
- #define GETGCOLOR(x) ((x ^ ~(~x | (CLEARMASK_NOT << 4))) >> 4)

  Gets green component of a color.
- #define GETBCOLOR(x) (x ^ ~(~x | CLEARMASK_NOT))

  Gets blue component of a color.
- #define GETACOLOR(x) ((x ^ ~(~x | (CLEARMASK_NOT << 12))) >> 12)

  Gets alpha component of a color.
- #define MASKTOBYTE(x) (x * 8)

  Transforms a recently extracted color component into a byte to be used by renderer.

Functions

- float MASKTOFLOAT (ColorM c)

  Transforms a recently extracted color component into a float to be used by renderer.

10.3.1 Detailed Description

Color-related bitwise operations.

10.3.2 Macro Definition Documentation

10.3.2.1 CLEARMASK_NOT

```
#define CLEARMASK_NOT 0x000Fu
```

Mask to be binded to the opposite of another color mask for byte extraction.

See also

GETRCOLOR(x)
GETGCOLOR(x)
GETBCOLOR(x)

10.3.2.2 GETACOLOR

```
#define GETACOLOR(
              x ) ((x ^ ~(~x | (CLEARMASK_NOT << 12))) >> 12)
```

Gets alpha component of a color.

Parameters

x	Color mask.

Returns

Alpha component mask.

10.3.2.3 GETBCOLOR

```
#define GETBCOLOR(
              x ) (x ^ ~(~x | CLEARMASK_NOT))
```

Gets blue component of a color.

Parameters

x	Color mask.

Returns

Blue component mask.

10.3.2.4 GETGCOLOR

```
#define GETGCOLOR(
              x ) ((x ^ ~(~x | (CLEARMASK_NOT << 4))) >> 4)
```

Gets green component of a color.

Parameters

x	Color mask.

Returns

Green component mask.

10.3.2.5 GETRCOLOR

```
#define GETRCOLOR(
              x ) ((x ^ ~(~x | (CLEARMASK_NOT << 8))) >> 8)
```

Gets red component of a color.

Parameters

x	Color mask.

Returns

Red component mask.

10.3.2.6 HIGHLIGHTMASK

```
#define HIGHLIGHTMASK 0x0777u
```

Mask to be used to form highlight mode.

See also

HIGHLIGHTMODE(x)

10.3.2.7 HIGHLIGHTMODE

```
#define HIGHLIGHTMODE(
              x ) ((x >> 1) + HIGHLIGHTMASK)
```

Sets a color to its highlighted mode.

Parameters

x	Color mask.

Returns

Highlighted color.

10.3.2.8 MASKTOBYTE

```
#define MASKTOBYTE(
              x ) (x * 8)
```

Transforms a recently extracted color component into a byte to be used by renderer.

Parameters

x	Component mask.

Returns

Component byte.

10.3.2.9 MIXCOLOR

```
#define MIXCOLOR(
              x,
              y ) (x = (x | y))
```

Sets the last byte of the mask to the byte component value to be set.

Parameters

x	Color mask.
y	Color component byte.

10.3.2.10 NEXTCOLOR

```
#define NEXTCOLOR(
              x ) (x = (x << 4))
```

Shifts the current byte to the left, so another component is added.

Parameters

x	Color mask.

10.3.2.11 SHADOWMODE

```
#define SHADOWMODE(
              x ) (x >> 1)
```

Sets a color to its shadowed mode.

Parameters

x	Color mask.

Returns

Shadowed color.

10.3.3 Function Documentation

10.3.3.1 MASKTOFLOAT()

```
float MASKTOFLOAT (
             ColorM c )
```

Transforms a recently extracted color component into a float to be used by renderer.

Parameters

c	Component mask.

Returns

Component float value.

10.4 vec2t Types

Typedefs

- typedef vec2t< double > vec2d
  A class representing a point in 2D space, with double precision.
- typedef vec2t< byte > vec2b
  A class representing a point in 2D space, with byte precision.
- typedef vec2t< word > vec2w
  A class representing a point in 2D space, with word precision.
- typedef vec2t< dword > vec2dw
  A class representing a point in 2D space, with double word precision.
- typedef vec2t< qword > vec2qw
  A class representing a point in 2D space, with quad word precision.
- typedef vec2t< byte_s > vec2sb
  A class representing a point in 2D space, with signed byte precision.
- typedef vec2t< word_s > vec2sw
  A class representing a point in 2D space, with signed word precision.
- typedef vec2t< dword_s > vec2sdw
  A class representing a point in 2D space, with signed double word precision.
- typedef vec2t< qword_s > vec2sqw
  A class representing a point in 2D space, with signed quad word precision.

10.4.1 Detailed Description

Typedefs for using vec2t<T>.

See also

vec2t
Data Types

10.4.2 Typedef Documentation

10.4.2.1 vec2b

```
typedef vec2t<byte> vec2b
```

A class representing a point in 2D space, with byte precision.

10.4.2.2 vec2d

```
typedef vec2t<double> vec2d
```

A class representing a point in 2D space, with double precision.

10.4.2.3 vec2dw

```
typedef vec2t<dword> vec2dw
```

A class representing a point in 2D space, with double word precision.

10.4.2.4 vec2qw

```
typedef vec2t<qword> vec2qw
```

A class representing a point in 2D space, with quad word precision.

10.4.2.5 vec2sb

```
typedef vec2t<byte_s> vec2sb
```

A class representing a point in 2D space, with signed byte precision.

10.4.2.6 vec2sdw

```
typedef vec2t<dword_s> vec2sdw
```

A class representing a point in 2D space, with signed double word precision.

10.4.2.7 vec2sqw

```
typedef vec2t<qword_s> vec2sqw
```

A class representing a point in 2D space, with signed quad word precision.

10.4.2.8 vec2sw

```
typedef vec2t<word_s> vec2sw
```

A class representing a point in 2D space, with signed word precision.

10.4.2.9 vec2w

```
typedef vec2t<word> vec2w
```

A class representing a point in 2D space, with word precision.

10.5 Angle Operations

Functions

- float degtorad (float angle)

  Converts an angle from degrees to radians.
- float radtodeg (float angle)

  Converts an angle from radians to degrees.
- float absolute (float val)

  Gives back the absolute value of another value.
- void clamp (float &value, float min, float max)

  Clamps a value to minimum and maximum values.

10.5.1 Detailed Description

Group of operations destined to angles and such.

10.5.2 Function Documentation

10.5.2.1 absolute()

```
float absolute (
            float val )
```

Gives back the absolute value of another value.

Parameters

val	Value to return its absolute value.

Returns

The absolute value of the given value.

10.5.2.2 clamp()

```
void clamp (
            float & value,
            float min,
            float max )
```

Clamps a value to minimum and maximum values.

Parameters

value	Reference to parameter to be clamped.
min	Minimum value.
max	Maximum value.

10.5.2.3 degtorad()

```
float degtorad (
            float angle )
```

Converts an angle from degrees to radians.

Parameters

angle	The given angle in degrees.

Returns

The given angle in radians.

10.5.2.4 radtodeg()

```
float radtodeg (
            float angle )
```

Converts an angle from radians to degrees.

Parameters

angle	The given angle in radians.

Returns

The given angle in degrees.

11 Namespace Documentation

11.1 OficinaFramework Namespace Reference

Classes

- class AudioSystem

  Groups audio-related management controls. Use this to play background music, sfx, and effects.
- class DiagnosticsSystem

  Controls for monitoring Memory and CPU usage.
- class EngineCore

  The main core of the engine, which handles initialization and game loop automatically.
- class EntitySystem

  Class including common controls for creating entities and entity collections.
- class InputSystem

  Groups all input-related methods and objects. Has built-in support for keyboard, multiple gamepads and mouse.
- class InvalidAssetException

  Exception for asset importing errors.
- class IOSystem

  Provides methods for loading compressed data.
- class NetworkSystem

  Manages all data sending and receiving over network.
- class OficinaException

  Base class for all framework exceptions.
- class RenderingSystem

  Groups rendering-related controls. Use this to allocate and deallocate textures accelerated by GPU, and also for drawing textures or primitives onscreen.
- class ScreenSystem

  Groups screen management controls. Use this class to add/remove screens and set them active or inactive.
- class SystemInitializationErrorException

  Exception for errors when initializing any system.
- class TimingSystem

  Groups framerate and in-game time controls. Use this class for accurate movement according to framerate, as well as setting it to an unlimited, time-based framerate.

12 Class Documentation

12.1 OficinaFramework::NetworkSystem::Address Struct Reference

A struct representing an IPv4 address.

`#include <NetworkSystem.hpp>`

Public Member Functions

- Address ()
  Constructs the address as localhost:1246.
- Address (dword addr, word port)
  Constructs the address as addr:port.
- Address (byte a, byte b, byte c, byte d)
  Constructs the address as a.b.c.d:1246.
- Address (byte a, byte b, byte c, byte d, word port)
  Constructs the address as a.b.c.d:port.
- dword GetAddress () const
  Returns the address.
- word GetNetworkPort () const
  Returns the port.
- Address & operator= (const Address)
- bool operator== (const Address)
- void ReceiveAddress (dword addr, word port)
  Receives address from a socket.
- std::string GetFullAddressf () const
  Returns a formatted, text-like address.

Friends

- std::ostream & operator<< (std::ostream &os, const Address &addr)

12.1.1 Detailed Description

A struct representing an IPv4 address.

12.1.2 Constructor & Destructor Documentation

12.1.2.1 Address() [1/4]

```
OficinaFramework::NetworkSystem::Address::Address ( )
```

Constructs the address as localhost:1246.

12.1.2.2 Address() [2/4]

```
OficinaFramework::NetworkSystem::Address::Address (
            dword addr,
            word port )
```

Constructs the address as addr:port.

Parameters

addr	The final address, as dword.
port	The port, as word.

12.1.2.3 Address() [3/4]

```
OficinaFramework::NetworkSystem::Address::Address (
            byte a,
            byte b,
            byte c,
            byte d )
```

Constructs the address as a.b.c.d:1246.

Parameters

a	First segment of IPv4 address.
b	Second segment of IPv4 address.
c	Third segment of IPv4 address.
d	Fourth segment of IPv4 address.

12.1.2.4 Address() [4/4]

```
OficinaFramework::NetworkSystem::Address::Address (
            byte a,
            byte b,
            byte c,
            byte d,
            word port )
```

Constructs the address as a.b.c.d:port.

Parameters

a	First segment of IPv4 address.
b	Second segment of IPv4 address.
c	Third segment of IPv4 address.
d	Fourth segment of IPv4 address.
port	Port to be directed or listened.

12.1.3 Member Function Documentation

12.1.3.1 GetAddress()

```
dword OficinaFramework::NetworkSystem::Address::GetAddress ( ) const
```

Returns the address.

Returns

The IPv4 address in 32-bit format.

12.1.3.2 GetFullAddressf()

```
std::string OficinaFramework::NetworkSystem::Address::GetFullAddressf ( ) const
```

Returns a formatted, text-like address.

Returns

A string containing the address.

12.1.3.3 GetNetworkPort()

```
word OficinaFramework::NetworkSystem::Address::GetNetworkPort ( ) const
```

Returns the port.

Returns

The IPv4 port in 16-bit format.

12.1.3.4 operator=()

```
Address& OficinaFramework::NetworkSystem::Address::operator= (
            const Address  )
```

12.1.3.5 operator==()

```
bool OficinaFramework::NetworkSystem::Address::operator== (
            const Address  )
```

12.1.3.6 ReceiveAddress()

```
void OficinaFramework::NetworkSystem::Address::ReceiveAddress (
            dword addr,
            word port )
```

Receives address from a socket.

Parameters

addr	Hexadecimal representation of address.
port	Port redirected.

12.1.4 Friends And Related Function Documentation

12.1.4.1 operator<<

```
std::ostream& operator<< (
            std::ostream & os,
            const Address & addr )  [friend]
```

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

- NetworkSystem.hpp

12.2 OficinaFramework::RenderingSystem::Animation Class Reference

Represents an Animation, a set of controls for animating objects using SpriteSheets.

```
#include <RenderingSystem.hpp>
```

Classes

- struct AnimationSpecs

  A struct representing the specs of a single animation.

Public Member Functions

- Animation (SpriteSheet *sheet)

  Constructs an animation.
- ~Animation ()

  Desructs the animation.
- void update ()

  Updates the animation.
- void Draw (vec2 Position, float magnification, Color4 tint, RenderEffect re=MODULATE_EFFECT)

  Draws the animation.
- void dispose ()

  Disposes the animation.
- SpriteSheet * GetSpriteSheet ()

  Gets the SpriteSheet of this Animation.
- void RegisterAnimation (std::string AnimationName, word frame_begin, word frame_end, float speed)

  Registers an animation on the database of this Animation class.

- void RegisterAnimation (std::string AnimationName, word frame_begin, word frame_end, word frame_loop, float speed)

  Registers an animation on the database of this Animation class.
- void RegisterAnimation (std::string AnimationName, AnimationSpecs specs)

  Registers an animation on the database of this Animation class.
- float GetAnimationSpeed () const

  Gets the current speed of the animation.
- void SetAnimationSpeed (float speed)

  Sets the current speed of the animation.
- void SetAnimation (std::string AnimationName)

  Sets the current animation, if registered.
- void SetOrientation (RenderProperty rp)

  Sets the orientation of the animation.
- RenderProperty GetOrientation () const

  Gets the orientation of the animation.
- std::string GetCurrentAnimationName () const

  Gets the name of the currently playing animation.
- dword GetCurrentFrame () const

  Gets the current animation frame playing.
- vec2dw GetFrameSize () const

  Gets the size of a single frame.
- void SetAngle (float angle)

  Sets the current animation angle.
- float GetAngle () const

  Gets the current animation angle.
- void SetAlpha (float alpha)

  Sets the alpha ratio of the animation.
- float GetAlpha () const

  Gets the alpha ratio of the animation.
- float GetDefaultAnimationSpeed () const

  Gets the default animation speed.
- vec2 GetHotspot () const

  Gets the hotspot of the animation.
- void SetSyncToFramerate (bool state)

  Sets or unsets whether the animation should synchonized to the current framerate.
- bool IsSyncToFramerate ()

  Gets whether the animation is synchronized to the current framerate.

12.2.1 Detailed Description

Represents an Animation, a set of controls for animating objects using SpriteSheets.

12.2.2 Constructor & Destructor Documentation

12.2.2.1 Animation()

```
OficinaFramework::RenderingSystem::Animation::Animation (
            SpriteSheet * sheet )
```

Constructs an animation.

Parameters

sheet	Pointer to the SpriteSheet containing the frames to be used.

12.2.2.2 ~Animation()

```
OficinaFramework::RenderingSystem::Animation::~Animation ( )
```

Desructs the animation.

12.2.3 Member Function Documentation

12.2.3.1 dispose()

```
void OficinaFramework::RenderingSystem::Animation::dispose ( )
```

Disposes the animation.

12.2.3.2 Draw()

```
void OficinaFramework::RenderingSystem::Animation::Draw (
            vec2 Position,
            float magnification,
            Color4 tint,
            RenderEffect re = MODULATE_EFFECT )
```

Draws the animation.

Parameters

Position	Position of the center of the animation to be used.
magnification	Magnification of the frame. Defaults to 1.0.

Warning

Use of this to simulate zoom is disencouraged.

Parameters

tint	Color to tint the sprite. Defaults to White (1.0f, 1.0f, 1.0f, 1.0f). The alpha factor will be used to measure intensity of tinting.
re	Effect when rendering the texture. Defaults to MODULATE_EFFECT.

12.2.3.3 GetAlpha()

```
float OficinaFramework::RenderingSystem::Animation::GetAlpha ( ) const
```

Gets the alpha ratio of the animation.

Returns

Alpha ratio of the animation, ranged 0∼1.

12.2.3.4 GetAngle()

```
float OficinaFramework::RenderingSystem::Animation::GetAngle ( ) const
```

Gets the current animation angle.

Returns

The angle of the animation, in degrees.

12.2.3.5 GetAnimationSpeed()

```
float OficinaFramework::RenderingSystem::Animation::GetAnimationSpeed ( ) const
```

Gets the current speed of the animation.

Returns

The current duration of frames on the animation, ranged 0.01f∼1.0f.

12.2.3.6 GetCurrentAnimationName()

```
std::string OficinaFramework::RenderingSystem::Animation::GetCurrentAnimationName ( ) const
```

Gets the name of the currently playing animation.

Returns

The name of the animation that is currently being played.

12.2.3.7 GetCurrentFrame()

```
dword OficinaFramework::RenderingSystem::Animation::GetCurrentFrame ( ) const
```

Gets the current animation frame playing.

Returns

The frame number on the SpriteSheet.

12.2.3.8 GetDefaultAnimationSpeed()

```
float OficinaFramework::RenderingSystem::Animation::GetDefaultAnimationSpeed ( ) const
```

Gets the default animation speed.

Returns

The default animation speed, or 0.0f in case of no animation.

12.2.3.9 GetFrameSize()

```
vec2dw OficinaFramework::RenderingSystem::Animation::GetFrameSize ( ) const
```

Gets the size of a single frame.

Returns

A vec2 with dword precision containing the size of one frame.

12.2.3.10 GetHotspot()

```
vec2 OficinaFramework::RenderingSystem::Animation::GetHotspot ( ) const
```

Gets the hotspot of the animation.

Returns

The point set to be the center of the animation.

12.2.3.11 GetOrientation()

```
RenderProperty OficinaFramework::RenderingSystem::Animation::GetOrientation ( ) const
```

Gets the orientation of the animation.

Returns

Orientation of the animation.

12.2.3.12 GetSpriteSheet()

```
SpriteSheet* OficinaFramework::RenderingSystem::Animation::GetSpriteSheet ( )
```

Gets the SpriteSheet of this Animation.

Returns

A pointer to this animation's SpriteSheet.

12.2.3.13 IsSyncToFramerate()

```
bool OficinaFramework::RenderingSystem::Animation::IsSyncToFramerate ( )
```

Gets whether the animation is synchronized to the current framerate.

Returns

Whether the sync property is active or not.

Warning

It is only advised to keep this on when running on fixed framerates.

12.2.3.14 RegisterAnimation() [1/3]

```
void OficinaFramework::RenderingSystem::Animation::RegisterAnimation (
            std::string AnimationName,
            word frame_begin,
            word frame_end,
            float speed )
```

Registers an animation on the database of this Animation class.

Parameters

AnimationName	Name for the animation to be recognized when called.
frame_begin	Frame in which the animation begins on the SpriteSheet.
frame_end	Frame in which the animation ends on the SpriteSheet.
speed	Speed of each frame of the animation in seconds.

12.2.3.15 RegisterAnimation() [2/3]

```
void OficinaFramework::RenderingSystem::Animation::RegisterAnimation (
            std::string AnimationName,
            word frame_begin,
            word frame_end,
            word frame_loop,
            float speed )
```

Registers an animation on the database of this Animation class.

Parameters

AnimationName	Name for the animation to be recognized when called.
frame_begin	Frame in which the animation begins on the SpriteSheet.
frame_end	Frame in which the animation ends on the SpriteSheet.
frame_loop	Frame in which the animation loop begins on the SpriteSheet.
speed	Speed of each frame of the animation in seconds.

12.2.3.16 RegisterAnimation() [3/3]

```
void OficinaFramework::RenderingSystem::Animation::RegisterAnimation (
            std::string AnimationName,
            AnimationSpecs specs )
```

Registers an animation on the database of this Animation class.

Parameters

AnimationName	Name for the animation to be recognized when called.
specs	Specs of the animation.

See also

RenderingSystem::Animation::AnimationSpecs

12.2.3.17 SetAlpha()

```
void OficinaFramework::RenderingSystem::Animation::SetAlpha (
            float alpha )
```

Sets the alpha ratio of the animation.

Parameters

alpha	Alpha ratio to be given to the animation, ranged 0∼1.

12.2.3.18 SetAngle()

```
void OficinaFramework::RenderingSystem::Animation::SetAngle (
            float angle )
```

Sets the current animation angle.

Parameters

angle	Angle to be given to the animation, in degrees.

12.2.3.19 SetAnimation()

```
void OficinaFramework::RenderingSystem::Animation::SetAnimation (
            std::string AnimationName )
```

Sets the current animation, if registered.

Parameters

AnimationName	Name for the animation to be played.

12.2.3.20 SetAnimationSpeed()

```
void OficinaFramework::RenderingSystem::Animation::SetAnimationSpeed (
            float speed )
```

Sets the current speed of the animation.

Parameters

speed	New duration for each frame of the animation, ranged 0.01f∼1.0f.

12.2.3.21 SetOrientation()

```
void OficinaFramework::RenderingSystem::Animation::SetOrientation (
            RenderProperty rp )
```

Sets the orientation of the animation.

Parameters

rp	Orientation to be given to the animation.

12.2.3.22 SetSyncToFramerate()

```
void OficinaFramework::RenderingSystem::Animation::SetSyncToFramerate (
            bool state )
```

Sets or unsets whether the animation should synchonized to the current framerate.

Parameters

state	State to be given to the sync property.

Warning

It is only advised to keep this on when running on fixed framerates.

12.2.3.23 update()

```
void OficinaFramework::RenderingSystem::Animation::update ( )
```

Updates the animation.

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

- RenderingSystem.hpp

12.3 OficinaFramework::RenderingSystem::Animation::AnimationSpecs Struct Reference

A struct representing the specs of a single animation.

```
#include <RenderingSystem.hpp>
```

Public Member Functions

- AnimationSpecs (word firstframe, word lastframe, float speed)

  Construct an AnimationSpecs.
- AnimationSpecs (word firstframe, word lastframe, word loopframe, float speed)

  Construct an AnimationSpecs.
- AnimationSpecs ()

  Construct an AnimationSpecs.

Public Attributes

- word m_beginning_frame

  Beggining frame of the animation on the sheet.
- word m_ending_frame

  End of the animation on the sheet.
- float m_speed

  Speed of each frame the animation in seconds.
- word m_loop_frame

  A frame for the animation to begin the loop.

12.3.1 Detailed Description

A struct representing the specs of a single animation.

12.3.2 Constructor & Destructor Documentation

12.3.2.1 AnimationSpecs() [1/3]

```
OficinaFramework::RenderingSystem::Animation::AnimationSpecs::AnimationSpecs (
            word firstframe,
            word lastframe,
            float speed )
```

Construct an AnimationSpecs.

Parameters

firstframe	First frame ID of the animation on the sheet.
lastframe	Last frame ID of the animation on the sheet.
speed	Speed of each frame of the animation in seconds. 1.0f and up makes animation static.

12.3.2.2 AnimationSpecs() [2/3]

```
OficinaFramework::RenderingSystem::Animation::AnimationSpecs::AnimationSpecs (
            word firstframe,
```

```
        word lastframe,
        word loopframe,
        float speed )
```

Construct an AnimationSpecs.

Parameters

firstframe	First frame ID of the animation on the sheet.
lastframe	Last frame ID of the animation on the sheet.
loopframe	Frame where the animation loop begins. Will be clamped to firstframe and lastframe.
speed	Speed of each frame of the animation in seconds. 1.0f and up makes animation static.

12.3.2.3 AnimationSpecs() [3/3]

```
OficinaFramework::RenderingSystem::Animation::AnimationSpecs::AnimationSpecs ( )
```

Construct an AnimationSpecs.

12.3.3 Member Data Documentation

12.3.3.1 m_beginning_frame

```
word OficinaFramework::RenderingSystem::Animation::AnimationSpecs::m_beginning_frame
```

Beggining frame of the animation on the sheet.

12.3.3.2 m_ending_frame

```
word OficinaFramework::RenderingSystem::Animation::AnimationSpecs::m_ending_frame
```

End of the animation on the sheet.

12.3.3.3 m_loop_frame

```
word OficinaFramework::RenderingSystem::Animation::AnimationSpecs::m_loop_frame
```

A frame for the animation to begin the loop.

12.3.3.4 m_speed

```
float OficinaFramework::RenderingSystem::Animation::AnimationSpecs::m_speed
```

Speed of each frame the animation in seconds.

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

- RenderingSystem.hpp

12.4 OficinaFramework::AudioSystem::Audio Class Reference

Represents an audio file to be loaded. Cannot be created nor destroyed on its own.

```
#include <AudioSystem.hpp>
```

Public Member Functions

- ALuint operator() ()
  Gets the identificator for the audio file on OpenAL.

Friends

- class AudioSystem
- class AudioSource
- class AudioPool

12.4.1 Detailed Description

Represents an audio file to be loaded. Cannot be created nor destroyed on its own.

12.4.2 Member Function Documentation

12.4.2.1 operator()()

```
ALuint OficinaFramework::AudioSystem::Audio::operator() ( )
```

Gets the identificator for the audio file on OpenAL.

Warning

Do not directly manipulate the Audio, unless you know exactly what you're doing.

Returns

Unsigned integer identificator for the audio file on OpenAL.

12.4.3 Friends And Related Function Documentation

12.4.3.1 AudioPool

```
friend class AudioPool  [friend]
```

12.4.3.2 AudioSource

```
friend class AudioSource  [friend]
```

12.4.3.3 AudioSystem

```
friend class AudioSystem  [friend]
```

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

- AudioSystem.hpp

12.5 OficinaFramework::AudioSystem::AudioPool Class Reference

A class for loading audio, both sound effects or background music.

```
#include <AudioSystem.hpp>
```

Static Public Member Functions

- static Audio * LoadAudio (std::string asset_path, AudioType type, bool loops=false, float loopEndS=0.0f, float loopStartS=0.0f)

  Loads an audio file from PATH.
- static void UnloadAudio (Audio *&audio)

  Unloads an audio.
- static void dispose ()

  Disposes all background music and sound effects.

Friends

- class AudioSystem

12.5.1 Detailed Description

A class for loading audio, both sound effects or background music.

12.5.2 Member Function Documentation

12.5.2.1 dispose()

```
static void OficinaFramework::AudioSystem::AudioPool::dispose ( )  [static]
```

Disposes all background music and sound effects.

12.5.2.2 LoadAudio()

```
static Audio* OficinaFramework::AudioSystem::AudioPool::LoadAudio (
            std::string asset_path,
            AudioType type,
            bool loops = false,
            float loopEndS = 0.0f,
            float loopStartS = 0.0f )  [static]
```

Loads an audio file from PATH.

Parameters

asset_path	Location for audio file inside PATH.
type	Audio type to be loaded.
loops	Whether the audio loops or not.

Note

If you're going to set this to "false", you may just want to ignore this and the next parameters.
If you want the audio to loop itself with the main OpenAL looping utility, though, just set this to true and ignore the next parameters.

Parameters

loopEndS	End-of-the-loop position for looping audio, in seconds.

Note

If this value is smaller than loopStartS - which makes no sense -, the song will loop normally using OpenAL's looping utility.

Parameters

loopStartS	Start-of-the-loop position for looping audio, in seconds.

Note

Audio will start playing from the beginning, so you can configure this value only if you have an intro of sorts. If you don't, just ignore this parameter.

12.5.2.3 UnloadAudio()

```
static void OficinaFramework::AudioSystem::AudioPool::UnloadAudio (
            Audio *& audio )  [static]
```

Unloads an audio.

Parameters

audio	Pointer to Audio to be unloaded.

12.5.3 Friends And Related Function Documentation

12.5.3.1 AudioSystem

```
friend class AudioSystem  [friend]
```

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

- AudioSystem.hpp

12.6 OficinaFramework::AudioSystem::AudioSource Class Reference

Describes the source of the audio, so effects such as positional sound can be used.

```
#include <AudioSystem.hpp>
```

Public Member Functions

- AudioSource ()
  Creates the AudioSource.
- ~AudioSource ()
  Destructs the AudioSource.
- ALuint operator() ()
  Gets the identificator for the audio source on OpenAL.
- ALuint getPitch ()
  Gets the pitch for the Audio Source.
- ALuint getGain ()
  Gets the gain for the Audio Source.
- vec3 getPosition ()

Gets the position for the Audio Source.

- vec3 getVelocity ()

  Gets the velocity for the Audio Source.
- float getElapsedTime ()

  Gets the elapsed time position for the currently playing audio.
- bool isStopped ()

  Gets whether audio is stopped or not.
- bool isPaused ()

  Gets whether audio is paused or not.
- void setPitch (ALuint val)

  Sets pitch for this AudioSource.
- void setGain (ALuint val)

  Sets gain for this AudioSource.
- void setPosition (vec3 val)

  Sets position for this AudioSource.
- void setVelocity (vec3 val)

  Sets velocity for this AudioSource.
- void setElapsedTime (float seconds)

  Sets the elapsed time position for the currently playing audio.
- void Play (Audio *audio)

  Plays audio.
- void Stop ()

  Stops audio, if currently being played.
- void TogglePause ()

  Toggles pause/unpause, if currently being played.
- void Rewind ()

  Rewinds entire audio track to beginning, if currently being played.

Friends

- class AudioSystem

12.6.1 Detailed Description

Describes the source of the audio, so effects such as positional sound can be used.

12.6.2 Constructor & Destructor Documentation

12.6.2.1 AudioSource()

```
OficinaFramework::AudioSystem::AudioSource::AudioSource ( )
```

Creates the AudioSource.

12.6.2.2 ~AudioSource()

```
OficinaFramework::AudioSystem::AudioSource::~AudioSource ( )
```

Destructs the AudioSource.

12.6.3 Member Function Documentation

12.6.3.1 getElapsedTime()

```
float OficinaFramework::AudioSystem::AudioSource::getElapsedTime ( )
```

Gets the elapsed time position for the currently playing audio.

Returns

Elapsed time in seconds.

12.6.3.2 getGain()

```
ALuint OficinaFramework::AudioSystem::AudioSource::getGain ( )
```

Gets the gain for the Audio Source.

Returns

An OpenAL-compatible uint for the gain.

12.6.3.3 getPitch()

```
ALuint OficinaFramework::AudioSystem::AudioSource::getPitch ( )
```

Gets the pitch for the Audio Source.

Returns

An OpenAL-compatible uint for the pitch.

12.6.3.4 getPosition()

```
vec3 OficinaFramework::AudioSystem::AudioSource::getPosition ( )
```

Gets the position for the Audio Source.

Returns

A vec3 for the Audio Source position coordinates.

12.6.3.5 getVelocity()

```
vec3 OficinaFramework::AudioSystem::AudioSource::getVelocity ( )
```

Gets the velocity for the Audio Source.

Returns

A vec3 for the Audio Source velocity values.

12.6.3.6 isPaused()

```
bool OficinaFramework::AudioSystem::AudioSource::isPaused ( )
```

Gets whether audio is paused or not.

12.6.3.7 isStopped()

```
bool OficinaFramework::AudioSystem::AudioSource::isStopped ( )
```

Gets whether audio is stopped or not.

12.6.3.8 operator()()

```
ALuint OficinaFramework::AudioSystem::AudioSource::operator() ( )
```

Gets the identificator for the audio source on OpenAL.

Warning

Do not directly manipulate the AudioSource, unless you know exactly what you're doing.

Returns

Unsigned integer identificator for the audio source on OpenAL.

12.6.3.9 Play()

```
void OficinaFramework::AudioSystem::AudioSource::Play (
            Audio * audio )
```

Plays audio.

Parameters

audio	Pointer to audio to be played.

12.6.3.10 Rewind()

```
void OficinaFramework::AudioSystem::AudioSource::Rewind ( )
```

Rewinds entire audio track to beginning, if currently being played.

12.6.3.11 setElapsedTime()

```
void OficinaFramework::AudioSystem::AudioSource::setElapsedTime (
            float seconds )
```

Sets the elapsed time position for the currently playing audio.

Parameters

seconds	Time to set the audio at, in seconds.

12.6.3.12 setGain()

```
void OficinaFramework::AudioSystem::AudioSource::setGain (
            ALuint val )
```

Sets gain for this AudioSource.

Parameters

val	Unsigned integer representing gain value.

12.6.3.13 setPitch()

```
void OficinaFramework::AudioSystem::AudioSource::setPitch (
            ALuint val )
```

Sets pitch for this AudioSource.

Parameters

val	Unsigned integer representing pitch value.

12.6.3.14 setPosition()

```
void OficinaFramework::AudioSystem::AudioSource::setPosition (
            vec3 val )
```

Sets position for this AudioSource.

Parameters

val	vec3 representing AudioSource position.

12.6.3.15 setVelocity()

```
void OficinaFramework::AudioSystem::AudioSource::setVelocity (
            vec3 val )
```

Sets velocity for this AudioSource.

Parameters

val	vec3 representing AudioSource velocity.

12.6.3.16 Stop()

```
void OficinaFramework::AudioSystem::AudioSource::Stop ( )
```

Stops audio, if currently being played.

12.6.3.17 TogglePause()

```
void OficinaFramework::AudioSystem::AudioSource::TogglePause ( )
```

Toggles pause/unpause, if currently being played.

12.6.4 Friends And Related Function Documentation

12.6.4.1 AudioSystem

```
friend class AudioSystem  [friend]
```

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

- AudioSystem.hpp

12.7 OficinaFramework::AudioSystem Class Reference

Groups audio-related management controls. Use this to play background music, sfx, and effects.

`#include <AudioSystem.hpp>`

Classes

- class Audio

  Represents an audio file to be loaded. Cannot be created nor destroyed on its own.
- class AudioPool

  A class for loading audio, both sound effects or background music.
- class AudioSource

  Describes the source of the audio, so effects such as positional sound can be used.

Public Types

- enum AudioType { OF_AUDIO_TYPE_WAV, OF_AUDIO_TYPE_OGG }

  An enumeration of supported audio types that can be loaded.

Static Public Member Functions

- static void init ()

  Initializes the AudioSystem.
- static void update ()

  Updates the AudioSystem.
- static void dispose ()

  Disposes the AudioSystem.
- static ALuint getListenerPitch ()

  Gets the Audio Listener's pitch.
- static ALuint getListenerGain ()

  Gets the Audio Listener's gain.
- static vec3 getListenerPosition ()

  Gets the Audio Listener's position.
- static vec3 getListenerVelocity ()

  Gets the Audio Listener's velocity.
- static vec3 getListenerOrientationAt ()

  Gets where the Audio Listener is looking at.
- static vec3 getListenerOrientationUp ()

  Gets the Audio Listener's normalized up direction.
- static bool isMute ()

  Checks if AudioSystem is mute.
- static void setListenerPitch (ALuint val)

  Set the Audio Listener's pitch.
- static void setListenerGain (ALuint val)

  Set the Audio Listener's gain.
- static void setListenerPosition (vec3 val)

  Set the Audio Listener's position.
- static void setListenerVelocity (vec3 val)

  Set the Audio Listener's velocity.
- static void setListenerOrientation (vec3 at, vec3 up)

  Set the Audio Listener's orientation.
- static void setMute (bool val)

  Sets the Mute flag for AudioSystem.

Friends

- class EngineCore

12.7.1 Detailed Description

Groups audio-related management controls. Use this to play background music, sfx, and effects.

12.7.2 Member Enumeration Documentation

12.7.2.1 AudioType

```
enum OficinaFramework::AudioSystem::AudioType
```

An enumeration of supported audio types that can be loaded.

Enumerator

OF_AUDIO_TYPE_WAV	WAV audio type. **Warning** Not yet implemented.
OF_AUDIO_TYPE_OGG	OGG Vorbis audio type.

12.7.3 Member Function Documentation

12.7.3.1 dispose()

```
static void OficinaFramework::AudioSystem::dispose ( )  [static]
```

Disposes the AudioSystem.

12.7.3.2 getListenerGain()

```
static ALuint OficinaFramework::AudioSystem::getListenerGain ( )  [static]
```

Gets the Audio Listener's gain.

Returns

An OpenAL-compatible uint for the gain.

12.7.3.3 getListenerOrientationAt()

```
static vec3 OficinaFramework::AudioSystem::getListenerOrientationAt ( )  [static]
```

Gets where the Audio Listener is looking at.

Returns

A vec3 containing the position coordinates.

12.7.3.4 getListenerOrientationUp()

```
static vec3 OficinaFramework::AudioSystem::getListenerOrientationUp ( )  [static]
```

Gets the Audio Listener's normalized up direction.

Returns

A vec3 containing the coordinates.

12.7.3.5 getListenerPitch()

```
static ALuint OficinaFramework::AudioSystem::getListenerPitch ( )  [static]
```

Gets the Audio Listener's pitch.

Returns

An OpenAL-compatible uint for the pitch.

12.7.3.6 getListenerPosition()

```
static vec3 OficinaFramework::AudioSystem::getListenerPosition ( )  [static]
```

Gets the Audio Listener's position.

Returns

A vec3 containing the position coordinates.

12.7.3.7 getListenerVelocity()

```
static vec3 OficinaFramework::AudioSystem::getListenerVelocity ( )  [static]
```

Gets the Audio Listener's velocity.

Returns

A vec3 containing velocity values.

12.7.3.8 init()

```
static void OficinaFramework::AudioSystem::init ( )  [static]
```

Initializes the AudioSystem.

12.7.3.9 isMute()

```
static bool OficinaFramework::AudioSystem::isMute ( )  [static]
```

Checks if AudioSystem is mute.

Returns

Whether the mute flag is active or not.

Warning

AudioSource class will NOT check for mute flags all the time, due to performance. Either mute the audio before or after you need it. Whenever you query a Play command, AudioSource will mute accordingly.

12.7.3.10 setListenerGain()

```
static void OficinaFramework::AudioSystem::setListenerGain (
            ALuint val )  [static]
```

Set the Audio Listener's gain.

Parameters

val	Unsigned integer value that should be set.

12.7.3.11 setListenerOrientation()

```
static void OficinaFramework::AudioSystem::setListenerOrientation (
            vec3 at,
            vec3 up )  [static]
```

Set the Audio Listener's orientation.

Parameters

at	A vec3 indicating where the Listener should look at.
up	A vec3 indicating the up direction for the Listener.

12.7.3.12 setListenerPitch()

```
static void OficinaFramework::AudioSystem::setListenerPitch (
            ALuint val )  [static]
```

Set the Audio Listener's pitch.

Parameters

val	Unsigned integer value that should be set.

12.7.3.13 setListenerPosition()

```
static void OficinaFramework::AudioSystem::setListenerPosition (
            vec3 val )  [static]
```

Set the Audio Listener's position.

Parameters

val	vec3 containing the desired position; prefer normalized.

12.7.3.14 setListenerVelocity()

```
static void OficinaFramework::AudioSystem::setListenerVelocity (
            vec3 val )  [static]
```

Set the Audio Listener's velocity.

Parameters

val	vec3 containing the desired velocity; prefer normalized.

12.7.3.15 setMute()

```
static void OficinaFramework::AudioSystem::setMute (
            bool val )  [static]
```

Sets the Mute flag for AudioSystem.

Parameters

val	Whether sound should be mute or not.

12.7.3.16 update()

```
static void OficinaFramework::AudioSystem::update ( )  [static]
```

Updates the AudioSystem.

Warning

Do not call this if you're using audio multithreading.

12.7.4 Friends And Related Function Documentation

12.7.4.1 EngineCore

```
friend class EngineCore  [friend]
```

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

- AudioSystem.hpp

12.8 Color4 Struct Reference

A struct representing a color.

```
#include <OficinaTypes.hpp>
```

Public Member Functions

- Color4 ()
  Constructs a black color.
- Color4 (float r, float g, float b)
  Constructs a custom opaque color.
- Color4 (float r, float g, float b, float a)
  Constructs a custom color.
- Color4 (Color4 precolor, float a)
  Constructs a color from another, changing only the alpha value.
- ColorM GetMask ()
  Gets the equivalent Color Mask of this color.

Static Public Member Functions

- static Color4 MaskToColor4 (ColorM mask)
  Transforms a mask into a Color4.

Public Attributes

- float r
  Red component, ranged 0.0f ~ 1.0f.
- float g
  Green component, ranged 0.0f ~ 1.0f.
- float b
  Blue component, ranged 0.0f ~ 1.0f.
- float a
  Alpha component, ranged 0.0f ~ 1.0f. 1.0f is completely opaque.

12.8.1 Detailed Description

A struct representing a color.

12.8.2 Constructor & Destructor Documentation

12.8.2.1 Color4() [1/4]

```
Color4::Color4 ( )
```

Constructs a black color.

12.8.2.2 Color4() [2/4]

```
Color4::Color4 (
            float r,
            float g,
            float b )
```

Constructs a custom opaque color.

Parameters

r	Red component to be given to the color, ranged 0.0f ∼ 1.0f.
g	Green component to be given to the color, ranged 0.0f ∼ 1.0f.
b	Blue component to be given to the color, ranged 0.0f ∼ 1.0f.

12.8.2.3 Color4() [3/4]

```
Color4::Color4 (
            float r,
            float g,
            float b,
            float a )
```

Constructs a custom color.

Parameters

r	Red component to be given to the color, ranged 0.0f ∼ 1.0f.
g	Green component to be given to the color, ranged 0.0f ∼ 1.0f.
b	Blue component to be given to the color, ranged 0.0f ∼ 1.0f.
a	Alpha component to be given to the color, ranged 0.0f ∼ 1.0f. Bigger value means more opaque.

12.8.2.4 Color4() [4/4]

```
Color4::Color4 (
            Color4 precolor,
            float a )
```

Constructs a color from another, changing only the alpha value.

Parameters

precolor	Color to be based on.
a	Alpha component to be given to the color, ranged 0.0f ∼ 1.0f.

12.8.3 Member Function Documentation

12.8.3.1 GetMask()

```
ColorM Color4::GetMask ( )
```

Gets the equivalent Color Mask of this color.

Returns

This color's equivalent color mask.

12.8.3.2 MaskToColor4()

```
static Color4 Color4::MaskToColor4 (
            ColorM mask )  [static]
```

Transforms a mask into a Color4.

Returns

A Color4 containing the same components for the mask.

12.8.4 Member Data Documentation

12.8.4.1 a

```
float Color4::a
```

Alpha component, ranged 0.0f $\sim$ 1.0f. 1.0f is completely opaque.

12.8.4.2 b

```
float Color4::b
```

Blue component, ranged 0.0f $\sim$ 1.0f.

12.8.4.3 g

```
float Color4::g
```

Green component, ranged 0.0f $\sim$ 1.0f.

12.8.4.4 r

```
float Color4::r
```

Red component, ranged 0.0f $\sim$ 1.0f.

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

- OficinaTypes.hpp

12.9 OficinaFramework::DiagnosticsSystem Class Reference

Controls for monitoring Memory and CPU usage.

`#include <DiagnosticsSystem.hpp>`

Static Public Member Functions

- static void init ()
  Initializes the system.
- static bool IsInitialized ()
  Gets if the system was initialized.
- static dword GetTotalVirtualMemory ()
  Gets the total virtual memory.
- static dword GetVirtualMemoryUsed ()
  Gets the total virtual memory used by the machine.
- static dword GetProcessVirtualMemoryUsed ()
  Gets the virtual memory used by the current process.
- static dword GetTotalPhysicalMemory ()
  Gets the total physical memory.
- static dword GetPhysicalMemoryUsed ()
  Gets the total physical memory used by the machine.
- static dword GetProcessPhysicalMemoryUsed ()
  Gets the physical memory used by the current process.
- static double GetCPUUsedPercent ()
  Gets the percentage of CPU usage.
- static double GetProcessCPUUsedPercent ()
  Gets the percentage of CPU usage by process.

12.9.1 Detailed Description

Controls for monitoring Memory and CPU usage.

12.9.2 Member Function Documentation

12.9.2.1 GetCPUUsedPercent()

`static double OficinaFramework::DiagnosticsSystem::GetCPUUsedPercent ( ) [static]`

Gets the percentage of CPU usage.

Returns

Machine CPU usage in percent.

12.9.2.2 GetPhysicalMemoryUsed()

```
static dword OficinaFramework::DiagnosticsSystem::GetPhysicalMemoryUsed ( )  [static]
```

Gets the total physical memory used by the machine.

Returns

Size of used physical memory.

12.9.2.3 GetProcessCPUUsedPercent()

```
static double OficinaFramework::DiagnosticsSystem::GetProcessCPUUsedPercent ( )  [static]
```

Gets the percentage of CPU usage by process.

Returns

Process CPU usage in percent.

12.9.2.4 GetProcessPhysicalMemoryUsed()

```
static dword OficinaFramework::DiagnosticsSystem::GetProcessPhysicalMemoryUsed ( )  [static]
```

Gets the physical memory used by the current process.

Returns

Size of physical memory used by application.

12.9.2.5 GetProcessVirtualMemoryUsed()

```
static dword OficinaFramework::DiagnosticsSystem::GetProcessVirtualMemoryUsed ( )  [static]
```

Gets the virtual memory used by the current process.

Returns

Size of virtual memory used by application.

12.9.2.6 GetTotalPhysicalMemory()

```
static dword OficinaFramework::DiagnosticsSystem::GetTotalPhysicalMemory ( )  [static]
```

Gets the total physical memory.

Returns

Size of the physical memory.

12.9.2.7 GetTotalVirtualMemory()

```
static dword OficinaFramework::DiagnosticsSystem::GetTotalVirtualMemory ( )  [static]
```

Gets the total virtual memory.

Returns

Size of the virtual memory.

12.9.2.8 GetVirtualMemoryUsed()

```
static dword OficinaFramework::DiagnosticsSystem::GetVirtualMemoryUsed ( )  [static]
```

Gets the total virtual memory used by the machine.

Returns

Size of used virtual memory.

12.9.2.9 init()

```
static void OficinaFramework::DiagnosticsSystem::init ( )  [static]
```

Initializes the system.

12.9.2.10 IsInitialized()

```
static bool OficinaFramework::DiagnosticsSystem::IsInitialized ( )  [static]
```

Gets if the system was initialized.

Returns

Whether it was initialized or not.

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

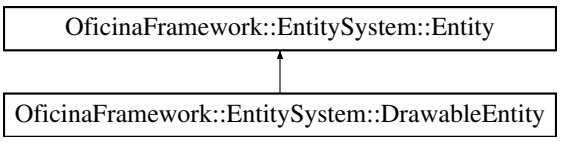
- DiagnosticsSystem.hpp

12.10 OficinaFramework::EntitySystem::DrawableEntity Class Reference

Abstract class representing an entity that can be drawn onscreen.

`#include <EntitySystem.hpp>`

Inheritance diagram for OficinaFramework::EntitySystem::DrawableEntity:



Public Member Functions

- virtual ~DrawableEntity ()

  Destructor for the drawable entity.
- virtual void Initialize ()=0

  Initializes the logic of the drawable entity.
- virtual void Update ()=0

  Updates the drawable entity on runtime.
- virtual void LoadContent ()=0

  Loads content for the drawable entity, such as textures.
- virtual void UnloadContent ()=0

  Unloads content for the drawable entity, such as textures.
- virtual void Draw ()=0

  Draws the drawable entity.
- Color4 GetColor () const

  Gets the internal color of the drawable entity.
- void SetColor (Color4 c)

  Sets the internal color of the drawable entity.
- float GetDrawDepth () const

  Gets the draw depth for drawing.
- void SetDrawDepth (float depth)

  Sets the draw depth.
- float GetScale () const

  Gets the scale value for object size. Implemented as per Gongly Script standard (0.2) (2014).
- void SetScale (float value)

  Sets the scale value for object size. Implemented as per Gongly Script standard (0.2) (2014).
- RenderingSystem::RenderProperty GetOrientation () const

  Gets object orientation. Implemented as per Gongly Script standard (0.2) (2014).
- void SetOrientation (RenderingSystem::RenderProperty value)

  Sets object orientation. Implemented as per Gongly Script standard (0.2) (2014).
- DrawableEntityCollection * GetParent ()

  Gets entity collection that manages this entity.
- void SetParent (DrawableEntityCollection *ec)

  Sets a pointer to the entity collection that manages this entity.

Protected Attributes

- Color4 m_color

  Internal color of the entity. Can be used for color blending. Implemented as per Gongly Script standard (2014).
- float m_depth = 0.0f

  Internal depth for using on draw order of a DrawableEntityCollection. Implemented as per Gongly Script standard (2014).
- float m_scale = 1.0f

  Internal scale value for object size. Implemented as per Gongly Script standard (0.2) (2014).
- RenderingSystem::RenderProperty m_orientation = RenderingSystem::RENDER_NORMALLY

  Internal orientation for object drawing. Implemented as per Gongly Script standard (0.2) (2014).

12.10.1 Detailed Description

Abstract class representing an entity that can be drawn onscreen.

12.10.2 Constructor & Destructor Documentation

12.10.2.1 ~DrawableEntity()

```
virtual OficinaFramework::EntitySystem::DrawableEntity::~DrawableEntity ( )  [inline], [virtual]
```

Destructor for the drawable entity.

References OficinaFramework::EntitySystem::Entity::GetParent(), OficinaFramework::EntitySystem::Entity::↩Initialize(), OficinaFramework::EntitySystem::Entity::SetParent(), and OficinaFramework::EntitySystem::Entity↩::Update().

12.10.3 Member Function Documentation

12.10.3.1 Draw()

```
virtual void OficinaFramework::EntitySystem::DrawableEntity::Draw ( )  [pure virtual]
```

Draws the drawable entity.

12.10.3.2 GetColor()

```
Color4 OficinaFramework::EntitySystem::DrawableEntity::GetColor ( ) const
```

Gets the internal color of the drawable entity.

Returns

Internal color used for effects such as color blending, etc.

12.10.3.3 GetDrawDepth()

```
float OficinaFramework::EntitySystem::DrawableEntity::GetDrawDepth ( ) const
```

Gets the draw depth for drawing.

Returns

Current draw depth.

12.10.3.4 GetOrientation()

```
RenderingSystem::RenderProperty OficinaFramework::EntitySystem::DrawableEntity::GetOrientation
( ) const
```

Gets object orientation. Implemented as per Gongly Script standard (0.2) (2014).

Returns

Object orientation in rendering property form.

12.10.3.5 GetParent()

```
DrawableEntityCollection* OficinaFramework::EntitySystem::DrawableEntity::GetParent ( )
```

Gets entity collection that manages this entity.

Returns

A pointer to the entity collection that manages this entity.

12.10.3.6 GetScale()

```
float OficinaFramework::EntitySystem::DrawableEntity::GetScale ( ) const
```

Gets the scale value for object size. Implemented as per Gongly Script standard (0.2) (2014).

Returns

Scale value of the entity.

12.10.3.7 Initialize()

```
virtual void OficinaFramework::EntitySystem::DrawableEntity::Initialize ( )  [pure virtual]
```

Initializes the logic of the drawable entity.

Implements OficinaFramework::EntitySystem::Entity.

12.10.3.8 LoadContent()

```
virtual void OficinaFramework::EntitySystem::DrawableEntity::LoadContent ( )  [pure virtual]
```

Loads content for the drawable entity, such as textures.

12.10.3.9 SetColor()

```
void OficinaFramework::EntitySystem::DrawableEntity::SetColor (
            Color4 c )
```

Sets the internal color of the drawable entity.

Parameters

c	Internal color to be used for effects such as color blending, etc.

12.10.3.10 SetDrawDepth()

```
void OficinaFramework::EntitySystem::DrawableEntity::SetDrawDepth (
            float depth )
```

Sets the draw depth.

Parameters

depth	Draw depth order to be given to this DrawableEntity. A smaller or negative depth means earlier drawing.

12.10.3.11 SetOrientation()

```
void OficinaFramework::EntitySystem::DrawableEntity::SetOrientation (
            RenderingSystem::RenderProperty value )
```

Sets object orientation. Implemented as per Gongly Script standard (0.2) (2014).

Parameters

value	Desired orientation to be set.

12.10.3.12 SetParent()

```
void OficinaFramework::EntitySystem::DrawableEntity::SetParent (
            DrawableEntityCollection * ec )
```

Sets a pointer to the entity collection that manages this entity.

Parameters

ec	DrawableEntityCollection that manages this entity.

Attention

This method is automatically used when adding the entity to a collection. Use at your own risk.

12.10.3.13 SetScale()

```
void OficinaFramework::EntitySystem::DrawableEntity::SetScale (
            float value )
```

Sets the scale value for object size. Implemented as per Gongly Script standard (0.2) (2014).

Parameters

value	Desired value to be set.

12.10.3.14 UnloadContent()

```
virtual void OficinaFramework::EntitySystem::DrawableEntity::UnloadContent ( )  [pure virtual]
```

Unloads content for the drawable entity, such as textures.

12.10.3.15 Update()

```
virtual void OficinaFramework::EntitySystem::DrawableEntity::Update ( )  [pure virtual]
```

Updates the drawable entity on runtime.

Implements OficinaFramework::EntitySystem::Entity.

12.10.4 Member Data Documentation

12.10.4.1 m_color

```
Color4 OficinaFramework::EntitySystem::DrawableEntity::m_color  [protected]
```

Internal color of the entity. Can be used for color blending. Implemented as per Gongly Script standard (2014).

12.10.4.2 m_depth

```
float OficinaFramework::EntitySystem::DrawableEntity::m_depth = 0.0f  [protected]
```

Internal depth for using on draw order of a DrawableEntityCollection. Implemented as per Gongly Script standard (2014).

12.10.4.3 m_orientation

```
RenderingSystem::RenderProperty OficinaFramework::EntitySystem::DrawableEntity::m_orientation
= RenderingSystem::RENDER_NORMALLY  [protected]
```

Internal orientation for object drawing. Implemented as per Gongly Script standard (0.2) (2014).

12.10.4.4 m_scale

```
float OficinaFramework::EntitySystem::DrawableEntity::m_scale = 1.0f  [protected]
```

Internal scale value for object size. Implemented as per Gongly Script standard (0.2) (2014).

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

- EntitySystem.hpp

12.11 OficinaFramework::EntitySystem::DrawableEntityCollection Class Reference

A collection of DrawableEntities to be used on a screen.

```
#include <EntitySystem.hpp>
```

Public Member Functions

- void Initialize ()

  Inits the collection. If there are already drawable entities inside, also inits them.
- void LoadContent ()

  Loads the collection. If there are already drawable entities inside, loads their content.
- void UnloadContent ()

  Unloads the collection. If there are still drawable entities inside, unloads their content.
- void Update ()

  Updates the drawable entities inside the collection.
- void Draw ()

  Draws the collection. This will draw each DrawableEntity inside the collection.
- void Dispose ()

  Disposes the drawable entities inside the collection.
- void Add (DrawableEntity *ptr)

  Adds a new DrawableEntity to the collection. If collection was already initialized and/or loaded, it'll init and/or load the DrawableEntity immediately.
- void Remove (DrawableEntity *ptr)

  Removes a specific drawable entity on the collection.
- void RemoveAll (std::string identifier)

  Removes all drawable entities with the desired name.
- const DrawableEntity * Get (std::string identifier) const

  Gets the pointer to a specific drawable entity, if it exists on the collection.
- const std::vector< DrawableEntity * > GetAll (std::string identifier) const

  Gets all pointers to drawable entities of the desired name.
- bool IsInitialized () const

Gets if the collection was initialized.

- bool IsContentLoaded () const

  Gets if the collection had its content loaded.

- void ReorderDrawList ()

  Reorders the Draw Order list.

- float GetBiggestDrawDepth ()

  Gets the biggest draw depth currently on the draw order list.

- float GetSmallestDrawDepth ()

  Gets the smallest draw depth currently on the draw order list.

- std::vector< DrawableEntity * >::const_iterator begin () const

  Gets beginning of DrawableEntityCollection.

- std::vector< DrawableEntity * >::const_iterator end () const

  Gets end of DrawableEntityCollection.

12.11.1 Detailed Description

A collection of DrawableEntities to be used on a screen.

See also

EntitySystem::DrawableEntity

12.11.2 Member Function Documentation

12.11.2.1 Add()

```
void OficinaFramework::EntitySystem::DrawableEntityCollection::Add (
            DrawableEntity * ptr )
```

Adds a new DrawableEntity to the collection. If collection was already initialized and/or loaded, it'll init and/or load the DrawableEntity immediately.

Parameters

ptr	Pointer for the actual drawable entity on the collection.

Exceptions

OficinaException	In case the DrawableEntity's name has not been set.

12.11.2.2 begin()

```
std::vector<DrawableEntity*>::const_iterator OficinaFramework::EntitySystem::DrawableEntity↩
Collection::begin ( ) const
```

Gets beginning of DrawableEntityCollection.

Returns

An iterator to the beginning of this DrawableEntityCollection.

12.11.2.3 Dispose()

```
void OficinaFramework::EntitySystem::DrawableEntityCollection::Dispose ( )
```

Disposes the drawable entities inside the collection.

12.11.2.4 Draw()

```
void OficinaFramework::EntitySystem::DrawableEntityCollection::Draw ( )
```

Draws the collection. This will draw each DrawableEntity inside the collection.

12.11.2.5 end()

```
std::vector<DrawableEntity*>::const_iterator OficinaFramework::EntitySystem::DrawableEntity↩
Collection::end ( ) const
```

Gets end of DrawableEntityCollection.

Returns

An iterator to the end of this DrawableEntityCollection.

12.11.2.6 Get()

```
const DrawableEntity* OficinaFramework::EntitySystem::DrawableEntityCollection::Get (
            std::string identifier ) const
```

Gets the pointer to a specific drawable entity, if it exists on the collection.

Parameters

identifier	String identifier for the drawable entity inside the collection.

Returns

The pointer to the DrawableEntity, or NULL if it doesn't exist in the collection.

12.11.2.7 GetAll()

```
const std::vector<DrawableEntity*> OficinaFramework::EntitySystem::DrawableEntityCollection↩
::GetAll (
            std::string identifier ) const
```

Gets all pointers to drawable entities of the desired name.

Parameters

identifier	String identifier for the drawable entities inside the collection.

Returns

A vector containing all found entities, or containing nothing.

12.11.2.8 GetBiggestDrawDepth()

```
float OficinaFramework::EntitySystem::DrawableEntityCollection::GetBiggestDrawDepth ( )
```

Gets the biggest draw depth currently on the draw order list.

Warning

This will not check if the draw order list is ordered after changing the depth of a single entity! You must still call ReorderDrawList.

Returns

The biggest draw depth on the current draw order list.

12.11.2.9 GetSmallestDrawDepth()

```
float OficinaFramework::EntitySystem::DrawableEntityCollection::GetSmallestDrawDepth ( )
```

Gets the smallest draw depth currently on the draw order list.

Warning

This will not check if the draw order list is ordered after changing the depth of a single entity! You must still call ReorderDrawList.

Returns

The smallest draw depth on the current draw order list.

12.11.2.10 Initialize()

```
void OficinaFramework::EntitySystem::DrawableEntityCollection::Initialize ( )
```

Inits the collection. If there are already drawable entities inside, also inits them.

12.11.2.11 IsContentLoaded()

```
bool OficinaFramework::EntitySystem::DrawableEntityCollection::IsContentLoaded ( ) const
```

Gets if the collection had its content loaded.

Returns

Whether the system was loaded or not.

12.11.2.12 IsInitialized()

```
bool OficinaFramework::EntitySystem::DrawableEntityCollection::IsInitialized ( ) const
```

Gets if the collection was initialized.

Returns

Whether the system was initialized or not.

12.11.2.13 LoadContent()

```
void OficinaFramework::EntitySystem::DrawableEntityCollection::LoadContent ( )
```

Loads the collection. If there are already drawable entities inside, loads their content.

12.11.2.14 Remove()

```
void OficinaFramework::EntitySystem::DrawableEntityCollection::Remove (
            DrawableEntity * ptr )
```

Removes a specific drawable entity on the collection.

Parameters

ptr	Pointer to the object to be removed.

12.11.2.15 RemoveAll()

```
void OficinaFramework::EntitySystem::DrawableEntityCollection::RemoveAll (
            std::string identifier )
```

Removes all drawable entities with the desired name.

Parameters

identifier	String identifier for the drawable entity inside the collection.

12.11.2.16 ReorderDrawList()

```
void OficinaFramework::EntitySystem::DrawableEntityCollection::ReorderDrawList ( )
```

Reorders the Draw Order list.

Attention

This should be called in case you change the draw depth of a certain entity on this collection.

12.11.2.17 UnloadContent()

```
void OficinaFramework::EntitySystem::DrawableEntityCollection::UnloadContent ( )
```

Unloads the collection. If there are still drawable entities inside, unloads their content.

12.11.2.18 Update()

```
void OficinaFramework::EntitySystem::DrawableEntityCollection::Update ( )
```

Updates the drawable entities inside the collection.

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

- EntitySystem.hpp

12.12 OficinaFramework::EngineCore Class Reference

The main core of the engine, which handles initialization and game loop automatically.

```
#include <EngineCore.hpp>
```

Static Public Member Functions

- static void Initialize (int argc, char **argv, std::list< std::string > *confv)

  Initializes the engine itself.
- static void Initialize (int argc, char **argv)

  Initializes the engine itself.
- static int DoGameLoop ()
- static void Dispose ()

  Disposes all data on all systems.

12.12.1 Detailed Description

The main core of the engine, which handles initialization and game loop automatically.

Todo
- Add listeners system for user-based modules.

12.12.2 Member Function Documentation

12.12.2.1 Dispose()

```
static void OficinaFramework::EngineCore::Dispose ( )  [static]
```

Disposes all data on all systems.

12.12.2.2 DoGameLoop()

```
static int OficinaFramework::EngineCore::DoGameLoop ( )  [static]
```

Executes the main game loop.

Returns

Game loop results, when game ends.

12.12.2.3 Initialize() [1/2]

```
static void OficinaFramework::EngineCore::Initialize (
            int argc,
            char ** argv,
            std::list< std::string > * confv )  [static]
```

Initializes the engine itself.

Parameters

argc	Number of arguments from main.
argv	Arguments from main.
confv	List of engine-specific string arguments. See GameArgs Available Options for details.

Warning

confv is deleted after this procedure.

12.12.2.4 Initialize() [2/2]

```
static void OficinaFramework::EngineCore::Initialize (
            int argc,
            char ** argv )  [static]
```

Initializes the engine itself.

Parameters

argc	Number of arguments from main.
argv	Arguments from main.

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

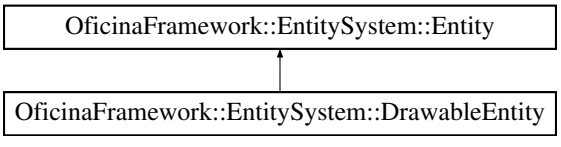
- EngineCore.hpp

12.13 OficinaFramework::EntitySystem::Entity Class Reference

Abstract class representing an entity.

```
#include <EntitySystem.hpp>
```

Inheritance diagram for OficinaFramework::EntitySystem::Entity:



Public Member Functions

- virtual ~Entity ()

  Destructor for the entity.
- virtual void Initialize ()=0

  Initializes the logic of the entity.
- virtual void Update ()=0

  Updates the entity on runtime.
- vec2 GetPosition () const

  Gets the position of the entity.
- void SetPosition (vec2 Position)

  Sets the position of the entity.
- float GetAngle () const

  Gets the angle of the entity.
- void SetAngle (float Angle)

  Sets the angle of the entity.
- std::string GetName () const

  Gets the name of the entity.
- void SetName (std::string name)

  Sets the name of the entity.
- bool GetProperty (byte ID) const

  Gets the state of a single property on the properties mask.
- void SetProperty (byte ID, bool state)

  Sets the state of a single property on the properties mask.
- qword GetProperties () const

  Gets the whole properties mask of the entity.
- void SetProperties (qword newmask)

  Sets the whole properties mask of the entity.
- EntityCollection * GetParent ()

  Gets entity collection that manages this entity.
- void SetParent (EntityCollection *ec)

  Sets a pointer to the entity collection that manages this entity.
- bool IsMarkedForRemoval ()

  Gets if the current entity is marked for removal.
- void RemoveMe ()

  Marks the current entity for removal.

Protected Attributes

- std::string m_name

  Name of the entity.
- vec2 m_position

  Position of the entity, as per Gongly Script standard (2014).
- float m_angle = 0.0f

  Angle of the entity, as per Gongly Script standard (2014).
- qword m_properties = 0x0000000000000000u

  A set of 64 boolean properties fo the entity, as per Gongly Script standard (2014).

12.13.1 Detailed Description

Abstract class representing an entity.

12.13.2 Constructor & Destructor Documentation

12.13.2.1 ~Entity()

```
virtual OficinaFramework::EntitySystem::Entity::~Entity ( )  [inline], [virtual]
```

Destructor for the entity.

References GetAngle(), GetName(), GetParent(), GetPosition(), GetProperties(), GetProperty(), Initialize(), IsMarkedForRemoval(), RemoveMe(), SetAngle(), SetName(), SetParent(), SetPosition(), SetProperties(), SetProperty(), and Update().

12.13.3 Member Function Documentation

12.13.3.1 GetAngle()

```
float OficinaFramework::EntitySystem::Entity::GetAngle ( ) const
```

Gets the angle of the entity.

Returns

Current angle of the entity.

12.13.3.2 GetName()

```
std::string OficinaFramework::EntitySystem::Entity::GetName ( ) const
```

Gets the name of the entity.

Returns

Entity name.

12.13.3.3 GetParent()

```
EntityCollection* OficinaFramework::EntitySystem::Entity::GetParent ( )
```

Gets entity collection that manages this entity.

Returns

A pointer to the entity collection that manages this entity.

12.13.3.4 GetPosition()

```
vec2 OficinaFramework::EntitySystem::Entity::GetPosition ( ) const
```

Gets the position of the entity.

Returns

Current position of the entity.

12.13.3.5 GetProperties()

```
qword OficinaFramework::EntitySystem::Entity::GetProperties ( ) const
```

Gets the whole properties mask of the entity.

Returns

The actual properties mask.

12.13.3.6 GetProperty()

```
bool OficinaFramework::EntitySystem::Entity::GetProperty (
            byte ID ) const
```

Gets the state of a single property on the properties mask.

Parameters

ID	ID ranged 0∼63 of the actual property.

Returns

The actual property state.

Warning

Giving and ID beyond 63 will always return false.

12.13.3.7 Initialize()

```
virtual void OficinaFramework::EntitySystem::Entity::Initialize ( )  [pure virtual]
```

Initializes the logic of the entity.

Implemented in OficinaFramework::EntitySystem::DrawableEntity.

12.13.3.8 IsMarkedForRemoval()

```
bool OficinaFramework::EntitySystem::Entity::IsMarkedForRemoval ( )
```

Gets if the current entity is marked for removal.

Returns

Whether the entity is marked for removal or not.

Warning

If the entity is on a collection, it will be removed on the collection's next update call.

12.13.3.9 RemoveMe()

```
void OficinaFramework::EntitySystem::Entity::RemoveMe ( )
```

Marks the current entity for removal.

Warning

If the entity is on a collection, it will be removed on the collection's next update call.

12.13.3.10 SetAngle()

```
void OficinaFramework::EntitySystem::Entity::SetAngle (
            float Angle )
```

Sets the angle of the entity.

Parameters

Angle	Angle to be given to the entity.

12.13.3.11 SetName()

```
void OficinaFramework::EntitySystem::Entity::SetName (
            std::string name )
```

Sets the name of the entity.

Parameters

name	Name to be given to the entity.

12.13.3.12 SetParent()

```
void OficinaFramework::EntitySystem::Entity::SetParent (
            EntityCollection * ec )
```

Sets a pointer to the entity collection that manages this entity.

Parameters

ec	EntityCollection that manages this entity.

Attention

This method is automatically used when adding the entity to a collection. Use at your own risk.

12.13.3.13 SetPosition()

```
void OficinaFramework::EntitySystem::Entity::SetPosition (
            vec2 Position )
```

Sets the position of the entity.

Parameters

Position	Position to be given to the entity.

12.13.3.14 SetProperties()

```
void OficinaFramework::EntitySystem::Entity::SetProperties (
            qword newmask )
```

Sets the whole properties mask of the entity.

Parameters

newmask	Mask to replace the properties mask for.

Warning

Use this carefully! You might mess up with your entities' properties.

12.13.3.15 SetProperty()

```
void OficinaFramework::EntitySystem::Entity::SetProperty (
            byte ID,
            bool state )
```

Sets the state of a single property on the properties mask.

Parameters

ID	ID ranged 0∼63 of the actual property.
state	Desired state of the property to be set.

Warning

Giving and ID beyond 63 will have no effect.

12.13.3.16 Update()

```
virtual void OficinaFramework::EntitySystem::Entity::Update ( )  [pure virtual]
```

Updates the entity on runtime.

Implemented in OficinaFramework::EntitySystem::DrawableEntity.

12.13.4 Member Data Documentation

12.13.4.1 m_angle

```
float OficinaFramework::EntitySystem::Entity::m_angle = 0.0f  [protected]
```

Angle of the entity, as per Gongly Script standard (2014).

12.13.4.2 m_name

```
std::string OficinaFramework::EntitySystem::Entity::m_name  [protected]
```

Name of the entity.

12.13.4.3 m_position

```
vec2 OficinaFramework::EntitySystem::Entity::m_position  [protected]
```

Position of the entity, as per Gongly Script standard (2014).

12.13.4.4 m_properties

```
qword OficinaFramework::EntitySystem::Entity::m_properties = 0x0000000000000000u  [protected]
```

A set of 64 boolean properties fo the entity, as per Gongly Script standard (2014).

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

- EntitySystem.hpp

12.14 OficinaFramework::EntitySystem::EntityCollection Class Reference

A collection of Entities to be used on a screen.

`#include <EntitySystem.hpp>`

Public Member Functions

- void Initialize ()
  Inits the collection. If there are already entities inside, also inits them.
- void Update ()
  Updates the entities inside the collection.
- void Dispose ()
  Disposes the entities inside the collection.
- void Add (Entity *ptr)
  Adds a new entity to the collection. If collection was already initialized, it'll init the Entity immediately.
- void Remove (Entity *ptr)
  Removes a specific entity on the collection.
- void RemoveAll (std::string identifier)
  Removes all entities with the desired name on the collection.
- const Entity * Get (std::string identifier) const
  Gets the pointer to a specific entity, if it exists on the collection.
- const std::vector< Entity * > GetAll (std::string identifier) const
  Gets all pointers to entities of the desired name.
- bool IsInitialized ()
  Gets if the collection was initialized.
- std::vector< Entity * >::const_iterator begin () const
  Gets beginning of EntityCollection.
- std::vector< Entity * >::const_iterator end () const
  Gets end of EntityCollection.

12.14.1 Detailed Description

A collection of Entities to be used on a screen.

See also

EntitySystem::Entity

12.14.2 Member Function Documentation

12.14.2.1 Add()

```
void OficinaFramework::EntitySystem::EntityCollection::Add (
            Entity * ptr )
```

Adds a new entity to the collection. If collection was already initialized, it'll init the Entity immediately.

Parameters

ptr	Pointer for the actual entity on the collection.

Exceptions

OficinaException	In case the Entity's name has not been set.

12.14.2.2 begin()

```
std::vector<Entity*>::const_iterator OficinaFramework::EntitySystem::EntityCollection::begin (
) const
```

Gets beginning of EntityCollection.

Returns

An iterator to the beginning of this EntityCollection.

12.14.2.3 Dispose()

```
void OficinaFramework::EntitySystem::EntityCollection::Dispose ( )
```

Disposes the entities inside the collection.

12.14.2.4 end()

```
std::vector<Entity*>::const_iterator OficinaFramework::EntitySystem::EntityCollection::end ( )
const
```

Gets end of EntityCollection.

Returns

An iterator to the end of this EntityCollection.

12.14.2.5 Get()

```
const Entity* OficinaFramework::EntitySystem::EntityCollection::Get (
            std::string identifier ) const
```

Gets the pointer to a specific entity, if it exists on the collection.

Parameters

identifier	String identifier for the entity inside the collection.

Returns

The pointer to the Entity, or NULL if it doesn't exist in the collection.

12.14.2.6 GetAll()

```
const std::vector<Entity*> OficinaFramework::EntitySystem::EntityCollection::GetAll (
            std::string identifier ) const
```

Gets all pointers to entities of the desired name.

Parameters

identifier	String identifier for the entities inside the collection.

Returns

A vector containing all found entities, or containing nothing.

12.14.2.7 Initialize()

```
void OficinaFramework::EntitySystem::EntityCollection::Initialize ( )
```

Inits the collection. If there are already entities inside, also inits them.

12.14.2.8 IsInitialized()

```
bool OficinaFramework::EntitySystem::EntityCollection::IsInitialized ( )
```

Gets if the collection was initialized.

Returns

Whether the system was initialized or not.

12.14.2.9 Remove()

```
void OficinaFramework::EntitySystem::EntityCollection::Remove (
            Entity * ptr )
```

Removes a specific entity on the collection.

Parameters

ptr	Removes the desired entity.

12.14.2.10 RemoveAll()

```
void OficinaFramework::EntitySystem::EntityCollection::RemoveAll (
            std::string identifier )
```

Removes all entities with the desired name on the collection.

Parameters

identifier	Name of the entities to be removed.

12.14.2.11 Update()

```
void OficinaFramework::EntitySystem::EntityCollection::Update ( )
```

Updates the entities inside the collection.

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

- EntitySystem.hpp

12.15 OficinaFramework::EntitySystem Class Reference

Class including common controls for creating entities and entity collections.

```
#include <EntitySystem.hpp>
```

Classes

- class DrawableEntity

  Abstract class representing an entity that can be drawn onscreen.
- class DrawableEntityCollection

  A collection of DrawableEntities to be used on a screen.
- class Entity

  Abstract class representing an entity.
- class EntityCollection

  A collection of Entities to be used on a screen.
- class IBuilder

  An interface for creating an Entity Builder, specially if it is supposed to be loaded from a script.

12.15.1 Detailed Description

Class including common controls for creating entities and entity collections.

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

- EntitySystem.hpp

12.16 OficinaFramework::RenderingSystem::Font Class Reference

Represents a Font, a texture with monospace characters to be used to draw text onscreen.

`#include <RenderingSystem.hpp>`

Public Member Functions

- Font (Texture *t, vec2dw CharacterSize, vec2b PaddingSize)
  Constructs a font.
- ~Font ()
  Disposes font accordingly.
- vec2dw GetCharacterSize () const
  Gets the size of a single character.
- vec2b GetPaddingSize () const
  Gets the size of the padding around a single character.
- void DrawString (vec2 Position, std::string Text, float magnification, Color4 tint, float transparency)
  Draws a specific string in a region.
- void DrawString (vec2 Position, std::string Text, float magnification)
  Draws a specific string in a region.
- void DrawString (vec2 Position, std::string Text)
  Draws a specific string in a region.
- vec2 MeasureString (std::string Text, float magnification) const
  Calculates the total area consumed by the text on screen.

12.16.1 Detailed Description

Represents a Font, a texture with monospace characters to be used to draw text onscreen.

Warning

The characters font range must be 31 ~ 126 in ASCII table.

12.16.2 Constructor & Destructor Documentation

12.16.2.1 Font()

```
OficinaFramework::RenderingSystem::Font::Font (
            Texture * t,
            vec2dw CharacterSize,
            vec2b PaddingSize )
```

Constructs a font.

Parameters

t	Texture to be used as source for characters.

Warning

The texture must contain monospace ASCII characters ranging from 31 (before white space) to '∼'.
DO NOT dispose the texture by yourself. The system will take care of it, once you delete the font pointer.

Parameters

CharacterSize	Size occupied by each character on the texture.
PaddingSize	Thickness of the padding around each character of the atlas. x specifies left and right boundaries, y specifies up and down boundaries.

12.16.2.2 ∼Font()

```
OficinaFramework::RenderingSystem::Font::∼Font ( )
```

Disposes font accordingly.

12.16.3 Member Function Documentation

12.16.3.1 DrawString() [1/3]

```
void OficinaFramework::RenderingSystem::Font::DrawString (
            vec2 Position,
            std::string Text,
            float magnification,
            Color4 tint,
            float transparency )
```

Draws a specific string in a region.

Warning

The text included must be ASCII characters ranging from 31∼126.

Parameters

Position	Position on screen in which the text will be drawn.
Text	Text that must be rendered. Only ASCII text is valid.
magnification	Size for the text to be multiplied by. Defaults to 1.
tint	Color to tint the font. Defaults to White (1.0f, 1.0f, 1.0f, 1.0f). The alpha factor will be used to measure intensity of tinting.
transparency	Transparency of the text. 1 is opaque, 0 is completely transparent.

12.16.3.2 DrawString() [2/3]

```
void OficinaFramework::RenderingSystem::Font::DrawString (
            vec2 Position,
            std::string Text,
            float magnification )
```

Draws a specific string in a region.

Warning

The text included must be ASCII characters ranging from 31∼126.

Parameters

Position	Position on screen in which the text will be drawn.
Text	Text that must be rendered. Only ASCII text is valid.
magnification	Size for the text to be multiplied by. Defaults to 1.

12.16.3.3 DrawString() [3/3]

```
void OficinaFramework::RenderingSystem::Font::DrawString (
            vec2 Position,
            std::string Text )
```

Draws a specific string in a region.

Warning

The text included must be ASCII characters ranging from 31∼126.

Parameters

Position	Position on screen in which the text will be drawn.
Text	Text that must be rendered. Only ASCII text is valid.

12.16.3.4 GetCharacterSize()

```
vec2dw OficinaFramework::RenderingSystem::Font::GetCharacterSize ( ) const
```

Gets the size of a single character.

Returns

A vec2 with dword precision showing the size of a character.

12.16.3.5 GetPaddingSize()

```
vec2b OficinaFramework::RenderingSystem::Font::GetPaddingSize ( ) const
```

Gets the size of the padding around a single character.

Returns

A vec2 with byte precision showing the thickness of the padding border.

12.16.3.6 MeasureString()

```
vec2 OficinaFramework::RenderingSystem::Font::MeasureString (
            std::string Text,
            float magnification ) const
```

Calculates the total area consumed by the text on screen.

Parameters

Text	Rendered text to have its space calculated.
magnification	Size for the text to be multiplied by. Defaults to 1.

Returns

A vec2, with float precision, showing the total occupied size by the string.

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

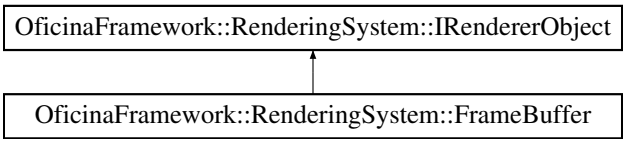
- RenderingSystem.hpp

12.17 OficinaFramework::RenderingSystem::FrameBuffer Class Reference

Describes a Frame Buffer object.

```
#include <RenderingSystem.hpp>
```

Inheritance diagram for OficinaFramework::RenderingSystem::FrameBuffer:



Public Member Functions

- FrameBuffer ()
  Creates a frame buffer object.
- ~FrameBuffer ()
  Deletes the frame buffer object.
- void Bind () override
  Binds the framebuffer to GL_FRAMEBUFFER.
- void Unbind () override
  Unbinds the framebuffer from GL_FRAMEBUFFER.
- bool IsBound () override
  Checks if the framebuffer is bound to GL_FRAMEBUFFER.
- GLuint operator() () override
  Provides direct access to the FrameBuffer as OpenGL object.
- void AttachRenderBuffer (RenderBuffer rb, GLenum attachment)
  Attaches a RenderBuffer to the current FrameBuffer.
- GLenum CheckStatus ()
  Checks the status of the current FrameBuffer.

12.17.1 Detailed Description

Describes a Frame Buffer object.

12.17.2 Constructor & Destructor Documentation

12.17.2.1 FrameBuffer()

```
OficinaFramework::RenderingSystem::FrameBuffer::FrameBuffer ( )
```

Creates a frame buffer object.

Note

The ctor will effectively generate the object on GPU.

12.17.2.2 ~FrameBuffer()

```
OficinaFramework::RenderingSystem::FrameBuffer::~FrameBuffer ( )
```

Deletes the frame buffer object.

Note

The dtor will effectively delete the object on GPU.

12.17.3 Member Function Documentation

12.17.3.1 AttachRenderBuffer()

```
void OficinaFramework::RenderingSystem::FrameBuffer::AttachRenderBuffer (
            RenderBuffer rb,
            GLenum attachment )
```

Attaches a RenderBuffer to the current FrameBuffer.

Parameters

rb	Related RenderBuffer.
attachment	RenderBuffer attachment type.

12.17.3.2 Bind()

```
void OficinaFramework::RenderingSystem::FrameBuffer::Bind ( )  [override], [virtual]
```

Binds the framebuffer to GL_FRAMEBUFFER.

Implements OficinaFramework::RenderingSystem::IRendererObject.

12.17.3.3 CheckStatus()

```
GLenum OficinaFramework::RenderingSystem::FrameBuffer::CheckStatus ( )
```

Checks the status of the current FrameBuffer.

Returns

An indicator for the current state of the FrameBuffer.

12.17.3.4 IsBound()

```
bool OficinaFramework::RenderingSystem::FrameBuffer::IsBound ( )  [override], [virtual]
```

Checks if the framebuffer is bound to GL_FRAMEBUFFER.

Implements OficinaFramework::RenderingSystem::IRendererObject.

12.17.3.5 operator()()

```
GLuint OficinaFramework::RenderingSystem::FrameBuffer::operator() ( )  [override], [virtual]
```

Provides direct access to the FrameBuffer as OpenGL object.

Returns

The object's name on GPU.

Implements OficinaFramework::RenderingSystem::IRendererObject.

12.17.3.6 Unbind()

```
void OficinaFramework::RenderingSystem::FrameBuffer::Unbind ( )  [override], [virtual]
```

Unbinds the framebuffer from GL_FRAMEBUFFER.

Implements OficinaFramework::RenderingSystem::IRendererObject.

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

- RenderingSystem.hpp

12.18 OficinaFramework::EntitySystem::IBuilder Class Reference

An interface for creating an Entity Builder, specially if it is supposed to be loaded from a script.

```
#include <EntitySystem.hpp>
```

Public Member Functions

- virtual ~IBuilder ()
  Virtual destructor for the interface.
- virtual Entity * CreateLogical (void *script_handler)=0
  Creates a logical entity.
- virtual DrawableEntity * CreateDrawable (void *script_handler)=0
  Creates a drawable entity.

12.18.1 Detailed Description

An interface for creating an Entity Builder, specially if it is supposed to be loaded from a script.

12.18.2 Constructor & Destructor Documentation

12.18.2.1 ~IBuilder()

```
virtual OficinaFramework::EntitySystem::IBuilder::~IBuilder ( )  [inline], [virtual]
```

Virtual destructor for the interface.

12.18.3 Member Function Documentation

12.18.3.1 CreateDrawable()

```
virtual DrawableEntity* OficinaFramework::EntitySystem::IBuilder::CreateDrawable (
            void * script_handler )  [pure virtual]
```

Creates a drawable entity.

Parameters

script_handler	Handler for the loaded script.

See also

IOSystem::ScriptTools

Returns

An instance to a drawable entity.

Note

In case the entity is not documented, NULL should be returned so the framework handles a warning message.

12.18.3.2 CreateLogical()

```
virtual Entity* OficinaFramework::EntitySystem::IBuilder::CreateLogical (
            void * script_handler )  [pure virtual]
```

Creates a logical entity.

Parameters

script_handler	Handler for the loaded script.

See also

IOSystem::ScriptTools

Returns

An instance to a logical entity.

Note

In case the entity is not documented, NULL should be returned so the framework handles a warning message.

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

- EntitySystem.hpp

12.19 OficinaFramework::InputSystem Class Reference

Groups all input-related methods and objects. Has built-in support for keyboard, multiple gamepads and mouse.

```
#include <InputSystem.hpp>
```

Classes

- struct State

  Represents a state for the input.

Public Types

- enum Type { Type::KEYBOARD = 0x00u, Type::XBOXPAD = 0x7Fu, Type::JOYPAD = 0xFFu }

  Represents the current input type on the input system.
- enum ThumbStick { ThumbStick::LEFTSTICK = 0x00u, ThumbStick::NOSTICK = 0x7Fu, ThumbStick::RIGHTSTICK = 0xFFu }

  Represents one thumbstick on a gamepad.
- enum ThumbStickAxis { ThumbStickAxis::HORIZONTAL = 0x00u, ThumbStickAxis::NOAXIS = 0x7Fu, ThumbStickAxis::VERTICAL = 0xFFu }

  Represents an axis of a thumbstick.
- enum ThumbStickAxisSignal { ThumbStickAxisSignal::CENTERED = 0x7FFFu, ThumbStickAxisSignal::POSITIVE = 0xFFFEu, ThumbStickAxisSignal::NEGATIVE = 0x0000u }

  Represents the signal of a thumbstick's axis or a Trigger.
- enum GamePadTrigger { GamePadTrigger::LEFTTRIGGER = 0x00u, GamePadTrigger::RIGHTTRIGGER = 0xFFu }

  Represents a trigger of the gamepad.
- enum GamePadButton {
  GamePadButton::START = 0x00u, GamePadButton::SELECT = 0x01u, GamePadButton::A = 0x02u, GamePadButton::B = 0x03u,
  GamePadButton::X = 0x04u, GamePadButton::Y = 0x05u, GamePadButton::LSTICK = 0x06u, GamePadButton::RSTICK = 0x07u,
  GamePadButton::HAT_UP = 0x08u, GamePadButton::HAT_DOWN = 0x09u, GamePadButton::HAT_LEFT = 0x0Au, GamePadButton::HAT_RIGHT = 0x0Bu,
  GamePadButton::LSHOULDER1 = 0x0Cu, GamePadButton::LSHOULDER2 = 0x0Du, GamePadButton::RSHOULDER1 = 0x0Eu, GamePadButton::RSHOULDER2 = 0x0Fu,
  GamePadButton::BIGBUTTON = 0x10u }

  Represents a button of the gamepad.
- enum MouseButton { MouseButton::LEFTMB = 0x00u, MouseButton::MIDDLEMB = 0x01u, MouseButton::RIGHTMB = 0x02u }

  Represents a button of the mouse.

Static Public Member Functions

- static void init ()

  Initializes the Input System and Joysticks, if connected.
- static void dispose ()

  Disposes all Input System and Joysticks.
- static void Update ()

  Updates the system.
- static void SetType (Type t)

  Sets the type of input for the input system.
- static void Set (GamePadButton btn, bool st)

  Sets the state of a gamepad button. Also works for keyboard, as key bindings have equivalent gamepad buttons.
- static void SetMouse (MouseButton btn, bool st)

  Sets the state of a mouse button.
- static void SetKeyboard (ThumbStick thumb, ThumbStickAxis axis, ThumbStickAxisSignal sig)

  Sets the state of the thumbsticks according to a keyboard key press.

- static void SetJoystick (ThumbStick thumb, ThumbStickAxis axis, word_s sig)

  Sets the state of the thumbsticks according to a joypad thumbstick move.
- static void SetTrigger (GamePadTrigger trig, word_s pos)

  Sets the state of a trigger according to a joypad trigger move, if supported.
- static void SetMouse (vec2dw pos)

  Sets the state of the mouse position.
- static void SetDeadZone (word dz)

  Sets the thumbsticks' dead zone value.
- static bool PressingButton (GamePadButton btn)

  Gets if a specific button is being pressed and held down.
- static bool PressedButton (GamePadButton btn)

  Gets if a specific button was pressed, at the pressing moment.
- static bool PressingMouse (MouseButton btn)

  Gets if a specific mouse button is being pressed and held down.
- static bool PressedMouse (MouseButton btn)

  Gets if a specific mouse button was pressed, at the pressing moment.
- static Type GetType ()

  Gets the current type of input being received.
- static vec2 GetLeftStick ()

  Gets the current position of the left thumbstick.
- static vec2 GetRightStick ()

  Gets the current position of the right thumbstick.
- static word GetDeadZone ()

  Gets the current Dead Zone of the thumbsticks.
- static float GetTrigger (GamePadTrigger tr)

  Gets the current position of a gamepad trigger.
- static vec2dw GetMousePos ()

  Gets the current position of the mouse.
- static State GetState ()

  Gets the current state of input.
- static State GetOldState ()

  Gets the old state of input.
- static void InitJoystick ()

  Tries to initialize a joystick, if there's any connected.
- static bool IsJoystickAvailable ()

  Shows if a joystick is available.
- static void StartJoystickRumbleSupport ()

  Starts the rumble support for the joystick, if available.
- static void StopJoystickRumbleSupport ()

  Stops the rumble support for the joystick, if active.
- static bool IsJoystickRumbleSupported ()

  Gets the rumble support for the joystick.
- static void Rumble (float strength, dword duration)

  Rumbles the joystick, if the rumble support is active.
- static bool IsXboxController ()

  Checks if the currently attached joystick is an Xbox Controller.
- static void CallExitCommand ()

  Calls the exit command input, setting the Exit flag to true.
- static bool IsExitFlagActive ()

  Checks if the exit flag has been activated.
- static void ShowMouse ()

Shows mouse pointer, if hidden.

- static void HideMouse ()

  Hides mouse pointer, if visible.

- static void BindKey (SDL_Keycode key, GamePadButton btn)

  Binds a keyboard key to a gamepad button.

- static void UnbindKey (SDL_Keycode key)

  Unbinds a keyboard key to a gamepad button.

- static void BindStick (SDL_Keycode key, ThumbStick stick, ThumbStickAxis axis, ThumbStickAxisSignal signal)

  Binds a keyboard key to a gamepad thumbstick movement.

- static void UnbindStick (ThumbStick stick, ThumbStickAxis axis, ThumbStickAxisSignal signal)

  Unbinds a keyboard key to a gamepad thumbstick movement. If the key is not binded, nothing happens.

- static void BindDefaultKeys ()

  Binds default keyboard keys to the application.
  Here's the equivalency table for the default bindings.

- static bool PressingKey (SDL_Scancode code)

  Gets if a keyboard key is being pressed.

- static bool PressedKey (SDL_Scancode code)

  Gets if a keyboard key was pressed.

- static bool MovedStick (ThumbStick thumbstick, ThumbStickAxis axis, ThumbStickAxisSignal signal)

  Gets if a thumbstick was moved to a given direction (current frame only).

- static std::string GetInputDeviceName ()

  Gets the current input device name.

12.19.1 Detailed Description

Groups all input-related methods and objects. Has built-in support for keyboard, multiple gamepads and mouse.

12.19.2 Member Enumeration Documentation

12.19.2.1 GamePadButton

```
enum OficinaFramework::InputSystem::GamePadButton  [strong]
```

Represents a button of the gamepad.

Enumerator

START	START button of gamepad.
SELECT	SELECT/BACK button of gamepad.
A	A button of gamepad.
B	B button of gamepad.
X	X button of gamepad.
Y	Y button of gamepad.
LSTICK	Left Stick (press) button of gamepad.
RSTICK	Right Stick (press) button of gamepad.
HAT_UP	UP directional of digital pad from gamepad.

Enumerator

HAT_DOWN	DOWN directional of digital pad from gamepad.
HAT_LEFT	LEFT directional of digital pad from gamepad.
HAT_RIGHT	RIGHT directional of digital pad from gamepad.
LSHOULDER1	Left Shoulder 1/Left Bumper button from gamepad.
LSHOULDER2	Left Shoulder 2/Left Trigger button from gamepad.
RSHOULDER1	Right Shoulder 1/Right Bumper button from gamepad.
RSHOULDER2	Right Shoulder 2/Right Trigger button from gamepad.
BIGBUTTON	Big Button/Xbox Button of gamepad.

12.19.2.2 GamePadTrigger

`enum OficinaFramework::InputSystem::GamePadTrigger [strong]`

Represents a trigger of the gamepad.

Enumerator

LEFTTRIGGER	Left Gamepad Trigger.
RIGHTTRIGGER	Right Gamepad Trigger.

12.19.2.3 MouseButton

`enum OficinaFramework::InputSystem::MouseButton [strong]`

Represents a button of the mouse.

Enumerator

LEFTMB	Left Mouse Button.
MIDDLEMB	Middle Mouse Button.
RIGHTMB	Right Mouse Button.

12.19.2.4 ThumbStick

`enum OficinaFramework::InputSystem::ThumbStick [strong]`

Represents one thumbstick on a gamepad.

Enumerator

LEFTSTICK	Left Thumbstick.
NOSTICK	No Thumbstick - used to represent an alternative axis or a Trigger.
RIGHTSTICK	Right Thumbstick.

12.19.2.5 ThumbStickAxis

```
enum OficinaFramework::InputSystem::ThumbStickAxis  [strong]
```

Represents an axis of a thumbstick.

Enumerator

HORIZONTAL	Horizontal axis.
NOAXIS	No axis - used to represent an alternative axis or a Trigger.
VERTICAL	Vertical axis.

12.19.2.6 ThumbStickAxisSignal

```
enum OficinaFramework::InputSystem::ThumbStickAxisSignal  [strong]
```

Represents the signal of a thumbstick's axis or a Trigger.

Enumerator

CENTERED	Represents a centered axis.
POSITIVE	Represents an axis with positive value (beyond center).
NEGATIVE	Represents an axis with negative value (short of center).

12.19.2.7 Type

```
enum OficinaFramework::InputSystem::Type  [strong]
```

Represents the current input type on the input system.

Enumerator

KEYBOARD	Keyboard input type.
XBOXPAD	Xbox Gamepad input type.
JOYPAD	Generic Joypad input type.

12.19.3 Member Function Documentation

12.19.3.1 BindDefaultKeys()

```
static void OficinaFramework::InputSystem::BindDefaultKeys ( )  [static]
```

Binds default keyboard keys to the application.
Here's the equivalency table for the default bindings.

Controller equivalent	Keyboard equivalent
Digital Hat Up	1
Digital Hat Right	2
Digital Hat Down	3
Digital Hat Left	4
Button A	K
Button B	L
Button X	J
Button Y	I
Start	ENTER/RETURN
Select	BACKSPACE
Right Shoulder 1	E
Right Shoulder 2	SPACE
Left Shoulder 1	Q
Left Shoulder 2	LEFT CTRL
Press Left Stick	U
Press Right Stick	O
Left Stick Up	W
Left Stick Down	S
Left Stick Left	A
Left Stick Right	D
Right Stick Up	UP ARROW KEY
Right Stick Down	DOWN ARROW KEY
Right Stick Left	LEFT ARROW KEY
Right Stick Right	RIGHT ARROW KEY

12.19.3.2 BindKey()

```
static void OficinaFramework::InputSystem::BindKey (
            SDL_Keycode key,
            GamePadButton btn )  [static]
```

Binds a keyboard key to a gamepad button.

Warning

If the button is already binded to a key, the key will be replaced; additionally, if the key is already binded to a button or thumbstick movement, the binding process will be ignored.

Parameters

key	The key to bind to a button.
btn	The button to be binded to.

12.19.3.3 BindStick()

```
static void OficinaFramework::InputSystem::BindStick (
            SDL_Keycode key,
            ThumbStick stick,
            ThumbStickAxis axis,
            ThumbStickAxisSignal signal )  [static]
```

Binds a keyboard key to a gamepad thumbstick movement.

Warning

If the thumbstick movement is already binded to a key, the key will be replaced; additionally, if the key is already binded to a button or a thumbstick movement, the binding process will be ignored.

Parameters

key	The key to bind to the stick movement.
stick	Desired thumbstick.
axis	Desired axis of thumbstick.
signal	Desired signal of the axis.

12.19.3.4 CallExitCommand()

```
static void OficinaFramework::InputSystem::CallExitCommand ( )  [static]
```

Calls the exit command input, setting the Exit flag to true.

12.19.3.5 dispose()

```
static void OficinaFramework::InputSystem::dispose ( )  [static]
```

Disposes all Input System and Joysticks.

12.19.3.6 GetDeadZone()

```
static word OficinaFramework::InputSystem::GetDeadZone ( )  [static]
```

Gets the current Dead Zone of the thumbsticks.

Returns

The word-precision dead zone value.

12.19.3.7 GetInputDeviceName()

```
static std::string OficinaFramework::InputSystem::GetInputDeviceName ( )  [static]
```

Gets the current input device name.

Returns

The name of the current main input device which the input is gotten.

12.19.3.8 GetLeftStick()

```
static vec2 OficinaFramework::InputSystem::GetLeftStick ( )  [static]
```

Gets the current position of the left thumbstick.

Returns

float-precision vec2 with the left thumbstick position.

12.19.3.9 GetMousePos()

```
static vec2dw OficinaFramework::InputSystem::GetMousePos ( )  [static]
```

Gets the current position of the mouse.

Returns

Mouse position related to the left upper corner of the screen, in pixels.

12.19.3.10 GetOldState()

```
static State OficinaFramework::InputSystem::GetOldState ( )  [static]
```

Gets the old state of input.

Returns

InputSystem::State of the input on the previous frame.

12.19.3.11 GetRightStick()

```
static vec2 OficinaFramework::InputSystem::GetRightStick ( )  [static]
```

Gets the current position of the right thumbstick.

Returns

A float-precision vec2 with the right thumbstick position.

12.19.3.12 GetState()

```
static State OficinaFramework::InputSystem::GetState ( )  [static]
```

Gets the current state of input.

Returns

InputSystem::State of the input on the current frame.

12.19.3.13 GetTrigger()

```
static float OficinaFramework::InputSystem::GetTrigger (
            GamePadTrigger tr )  [static]
```

Gets the current position of a gamepad trigger.

Parameters

tr	Desired trigger to get its value.

Returns

A float-precision value with the trigger position.

12.19.3.14 GetType()

```
static Type OficinaFramework::InputSystem::GetType ( )  [static]
```

Gets the current type of input being received.

Returns

The InputSystem::Type of input currently being processed.

12.19.3.15 HideMouse()

```
static void OficinaFramework::InputSystem::HideMouse ( )  [static]
```

Hides mouse pointer, if visible.

12.19.3.16 init()

```
static void OficinaFramework::InputSystem::init ( )  [static]
```

Initializes the Input System and Joysticks, if connected.

12.19.3.17 InitJoystick()

```
static void OficinaFramework::InputSystem::InitJoystick ( )  [static]
```

Tries to initialize a joystick, if there's any connected.

12.19.3.18 IsExitFlagActive()

```
static bool OficinaFramework::InputSystem::IsExitFlagActive ( )  [static]
```

Checks if the exit flag has been activated.

Returns

Whether the user has sent the Quit signal to the window, via controller or by closing it.

12.19.3.19 IsJoystickAvailable()

```
static bool OficinaFramework::InputSystem::IsJoystickAvailable ( )  [static]
```

Shows if a joystick is available.

Returns

Whether at least one joystick is attached.

12.19.3.20 IsJoystickRumbleSupported()

```
static bool OficinaFramework::InputSystem::IsJoystickRumbleSupported ( )  [static]
```

Gets the rumble support for the joystick.

Returns

Whether rumble is supported or not.

12.19.3.21 IsXboxController()

```
static bool OficinaFramework::InputSystem::IsXboxController ( )  [static]
```

Checks if the currently attached joystick is an Xbox Controller.

Returns

Whether the attached controller has Xbox in its name or not.

12.19.3.22 MovedStick()

```
static bool OficinaFramework::InputSystem::MovedStick (
            ThumbStick thumbstick,
            ThumbStickAxis axis,
            ThumbStickAxisSignal signal )  [static]
```

Gets if a thumbstick was moved to a given direction (current frame only).

Returns

Whether the thumbstick was moved on the given direction or not.

Parameters

thumbstick	Related thumbstick.
axis	Axis to be compared.
signal	Signal of the axis.

12.19.3.23 PressedButton()

```
static bool OficinaFramework::InputSystem::PressedButton (
            GamePadButton btn )  [static]
```

Gets if a specific button was pressed, at the pressing moment.

Parameters

btn	Button to make the comparision.

Returns

If the button was being pressed at the specific frame of this call.

12.19.3.24 PressedKey()

```
static bool OficinaFramework::InputSystem::PressedKey (
            SDL_Scancode code )  [static]
```

Gets if a keyboard key was pressed.

Returns

Whether the key is being pressed at the current instant or not.

12.19.3.25 PressedMouse()

```
static bool OficinaFramework::InputSystem::PressedMouse (
            MouseButton btn )  [static]
```

Gets if a specific mouse button was pressed, at the pressing moment.

Parameters

btn	Mouse button to make the comparision.

Returns

If the mouse button was being pressed at the specific frame of this call.

12.19.3.26 PressingButton()

```
static bool OficinaFramework::InputSystem::PressingButton (
            GamePadButton btn )  [static]
```

Gets if a specific button is being pressed and held down.

Parameters

btn	Button to make the comparision.

Returns

If the button is being pressed and held.

12.19.3.27 PressingKey()

```
static bool OficinaFramework::InputSystem::PressingKey (
            SDL_Scancode code )  [static]
```

Gets if a keyboard key is being pressed.

Returns

Whether the key is being pressed or not.

12.19.3.28 PressingMouse()

```
static bool OficinaFramework::InputSystem::PressingMouse (
            MouseButton btn )  [static]
```

Gets if a specific mouse button is being pressed and held down.

Parameters

btn	Mouse button to make the comparision.

Returns

If the mouse button is being pressed and held.

12.19.3.29 Rumble()

```
static void OficinaFramework::InputSystem::Rumble (
            float strength,
            dword duration )  [static]
```

Rumbles the joystick, if the rumble support is active.

Parameters

strength	The strength of the rumble (a value between 0 and 1).
duration	The duration of the rumble, in miliseconds.

12.19.3.30 Set()

```
static void OficinaFramework::InputSystem::Set (
            GamePadButton btn,
            bool st )  [static]
```

Sets the state of a gamepad button. Also works for keyboard, as key bindings have equivalent gamepad buttons.

Parameters

btn	The desired button to have the state set.
st	The state of the gamepad button.

12.19.3.31 SetDeadZone()

```
static void OficinaFramework::InputSystem::SetDeadZone (
            word dz )  [static]
```

Sets the thumbsticks' dead zone value.

Parameters

dz	Desired dead zone value.

12.19.3.32 SetJoystick()

```
static void OficinaFramework::InputSystem::SetJoystick (
            ThumbStick thumb,
            ThumbStickAxis axis,
            word_s sig )  [static]
```

Sets the state of the thumbsticks according to a joypad thumbstick move.

Parameters

thumb	The desired thumbstick.
axis	The axis of the desired thumbstick.
sig	Position of the thumbstick on the related axis.

12.19.3.33 SetKeyboard()

```
static void OficinaFramework::InputSystem::SetKeyboard (
            ThumbStick thumb,
            ThumbStickAxis axis,
            ThumbStickAxisSignal sig )  [static]
```

Sets the state of the thumbsticks according to a keyboard key press.

Parameters

thumb	The desired thumbstick.
axis	The axis of the desired thumbstick.
sig	Axis' signal of the desired thumbstick.

12.19.3.34 SetMouse() [1/2]

```
static void OficinaFramework::InputSystem::SetMouse (
            MouseButton btn,
            bool st )  [static]
```

Sets the state of a mouse button.

Parameters

btn	The desired mouse button to have the state set.
st	The state of the mouse button.

12.19.3.35 SetMouse() [2/2]

```
static void OficinaFramework::InputSystem::SetMouse (
            vec2dw pos )  [static]
```

Sets the state of the mouse position.

Parameters

pos	Desired mouse position to be set.

12.19.3.36 SetTrigger()

```
static void OficinaFramework::InputSystem::SetTrigger (
            GamePadTrigger trig,
            word_s pos )  [static]
```

Sets the state of a trigger according to a joypad trigger move, if supported.

Parameters

trig	The desired trigger.
pos	The position of the trigger on its related axis.

12.19.3.37 SetType()

```
static void OficinaFramework::InputSystem::SetType (
            Type t )  [static]
```

Sets the type of input for the input system.

Parameters

t	Type of input to be set.

12.19.3.38 ShowMouse()

```
static void OficinaFramework::InputSystem::ShowMouse ( )  [static]
```

Shows mouse pointer, if hidden.

12.19.3.39 StartJoystickRumbleSupport()

```
static void OficinaFramework::InputSystem::StartJoystickRumbleSupport ( )  [static]
```

Starts the rumble support for the joystick, if available.

12.19.3.40 StopJoystickRumbleSupport()

```
static void OficinaFramework::InputSystem::StopJoystickRumbleSupport ( )  [static]
```

Stops the rumble support for the joystick, if active.

12.19.3.41 UnbindKey()

```
static void OficinaFramework::InputSystem::UnbindKey (
            SDL_Keycode key )  [static]
```

Unbinds a keyboard key to a gamepad button.

Warning

If the key is not binded, nothing happens.

Parameters

key	The key to unbind to a button.

12.19.3.42 UnbindStick()

```
static void OficinaFramework::InputSystem::UnbindStick (
            ThumbStick stick,
            ThumbStickAxis axis,
            ThumbStickAxisSignal signal )  [static]
```

Unbinds a keyboard key to a gamepad thumbstick movement. If the key is not binded, nothing happens.

Parameters

stick	Desired thumbstick to have a movement unbinded.
axis	Desired axis of thumbstick to have a movement unbinded.
signal	Desired signal of the axis to have a movement unbinded.

12.19.3.43 Update()

```
static void OficinaFramework::InputSystem::Update ( )  [static]
```

Updates the system.

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

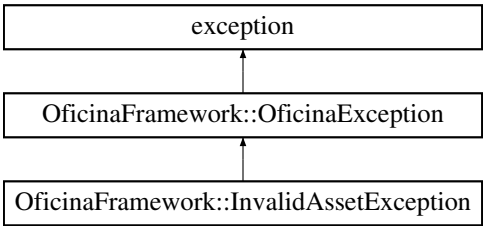
- InputSystem.hpp

12.20 OficinaFramework::InvalidAssetException Class Reference

Exception for asset importing errors.

```
#include <OficinaExceptions.hpp>
```

Inheritance diagram for OficinaFramework::InvalidAssetException:



Public Member Functions

- InvalidAssetException ()

  Instantiates the exception with default error message.
- InvalidAssetException (std::string message)

  Instantiates the exception with a given error message.
- InvalidAssetException (std::string message, std::string assetname)

  Instantiates the exception with a given error message. Also identifies the asset with errors.
- std::string GetAssetName ()

  A string containing the asset name or path.
- std::string GetMessageAndAssetName ()

  A string containing the message and the asset name.
- virtual const char * what () const throw ()

  Returns the message and troubling asset name for the actual system exception display.

Additional Inherited Members

12.20.1 Detailed Description

Exception for asset importing errors.

12.20.2 Constructor & Destructor Documentation

12.20.2.1 InvalidAssetException() [1/3]

```
OficinaFramework::InvalidAssetException::InvalidAssetException ( )  [inline]
```

Instantiates the exception with default error message.

12.20.2.2 InvalidAssetException() [2/3]

```
OficinaFramework::InvalidAssetException::InvalidAssetException (
            std::string message )  [inline]
```

Instantiates the exception with a given error message.

Parameters

message	The message to be set to the exception.

12.20.2.3 InvalidAssetException() [3/3]

```
OficinaFramework::InvalidAssetException::InvalidAssetException (
```

```
            std::string message,
            std::string assetname )  [inline]
```

Instantiates the exception with a given error message. Also identifies the asset with errors.

Parameters

message	The message to be set to the exception.
assetname	The path or name of the problematic asset.

12.20.3 Member Function Documentation

12.20.3.1 GetAssetName()

```
std::string OficinaFramework::InvalidAssetException::GetAssetName ( )
```

A string containing the asset name or path.

Returns

Name or path of the problematic asset.

12.20.3.2 GetMessageAndAssetName()

```
std::string OficinaFramework::InvalidAssetException::GetMessageAndAssetName ( )
```

A string containing the message and the asset name.

Returns

Exception message, plus the problematic asset name.

12.20.3.3 what()

```
virtual const char* OficinaFramework::InvalidAssetException::what ( ) const throw )  [inline],
[virtual]
```

Returns the message and troubling asset name for the actual system exception display.

Returns

See also

GetMessageAndAssetName

Reimplemented from OficinaFramework::OficinaException.

References OficinaFramework::OficinaException::message.

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

- OficinaExceptions.hpp

12.21 OficinaFramework::IOSystem Class Reference

Provides methods for loading compressed data.

`#include <IOSystem.hpp>`

Classes

- class ScriptStream

  Reads a script as a byte stream.
- class ScriptTools

  A class for opening and loading Gongly Script data. Works since Gongly Script v1.0.

Static Public Member Functions

- static void init (const char *currentpath)

  Initializes the path.
- static void dispose ()

  Closes the path.
- static bool AddToSearchPath (std::string file)

  Adds a compressed file to the beginning of the search path.
- static byte * Load (std::string asset_path, qword_s &size)

  Loads a raw file, from path.
- static SDL_Surface * LoadTexture (std::string asset_path)

  Loads a texture, PNG format, from path.
- static bool IsBigEndian ()

  Check whether the current system is big endian.
- static ScriptStream * LoadScriptByteStream (std::string script_path)

  Loads an entire entity script from path, as a binary stream. If you're not using IOSystem::ScriptTools, remember to unload it.
- static void SwapEndianness16 (word &val)

  Swaps endianness of 16-bit variable.
- static void SwapEndianness16 (word_s &val)
- static void SwapEndianness32 (dword &val)

  Swaps endianness of 32-bit variable.
- static void SwapEndianness32 (int &val)
- static void SwapEndianness64 (qword &val)

  Swaps endianness of 64-bit variable.
- static void SwapEndianness64 (qword_s &val)
- static void SwapEndiannessF (float &val)

  Swaps endianness of float variable.
- static void SwapEndiannessD (double &val)

  Swaps endianness of double variable.
- static bool IsInitialized ()

  Gets if the IOSystem is initialized.

12.21.1 Detailed Description

Provides methods for loading compressed data.

12.21.2 Member Function Documentation

12.21.2.1 AddToSearchPath()

```
static bool OficinaFramework::IOSystem::AddToSearchPath (
            std::string file )  [static]
```

Adds a compressed file to the beginning of the search path.

Returns

Whether it could be added or not.

12.21.2.2 dispose()

```
static void OficinaFramework::IOSystem::dispose ( )  [static]
```

Closes the path.

12.21.2.3 init()

```
static void OficinaFramework::IOSystem::init (
            const char * currentpath )  [static]
```

Initializes the path.

Parameters

currentpath	Current running path - argv's [0] position.

12.21.2.4 IsBigEndian()

```
static bool OficinaFramework::IOSystem::IsBigEndian ( )  [static]
```

Check whether the current system is big endian.

Returns

Whether the system is big endian or not.

12.21.2.5 IsInitialized()

```
static bool OficinaFramework::IOSystem::IsInitialized ( )  [static]
```

Gets if the IOSystem is initialized.

Returns

Whether it is initialized or not.

12.21.2.6 Load()

```
static byte* OficinaFramework::IOSystem::Load (
            std::string asset_path,
            qword_s & size )  [static]
```

Loads a raw file, from path.

Returns

A pointer to the loaded file, or NULL if it can't be loaded.

Parameters

asset_path	Path to the asset on the actual Path.
size	Size of structure to be loaded.

12.21.2.7 LoadScriptByteStream()

```
static ScriptStream* OficinaFramework::IOSystem::LoadScriptByteStream (
            std::string script_path )  [static]
```

Loads an entire entity script from path, as a binary stream. If you're not using IOSystem::ScriptTools, remember to unload it.

Parameters

script_path	Path to the script on the actual Path.

12.21.2.8 LoadTexture()

```
static SDL_Surface* OficinaFramework::IOSystem::LoadTexture (
            std::string asset_path )  [static]
```

Loads a texture, PNG format, from path.

Returns

A pointer to the loaded SDL_Surface, or NULL if it can't be loaded.

Parameters

asset_path	Path to the asset on the actual Path.

12.21.2.9 SwapEndianness16() [1/2]

```
static void OficinaFramework::IOSystem::SwapEndianness16 (
            word & val )  [static]
```

Swaps endianness of 16-bit variable.

Parameters

val	Value to be swapped.

12.21.2.10 SwapEndianness16() [2/2]

```
static void OficinaFramework::IOSystem::SwapEndianness16 (
            word_s & val )  [static]
```

12.21.2.11 SwapEndianness32() [1/2]

```
static void OficinaFramework::IOSystem::SwapEndianness32 (
            dword & val )  [static]
```

Swaps endianness of 32-bit variable.

Parameters

val	Value to be swapped.

12.21.2.12 SwapEndianness32() [2/2]

```
static void OficinaFramework::IOSystem::SwapEndianness32 (
            int & val )  [static]
```

12.21.2.13 SwapEndianness64() [1/2]

```
static void OficinaFramework::IOSystem::SwapEndianness64 (
            qword & val )  [static]
```

Swaps endianness of 64-bit variable.

Parameters

val	Value to be swapped.

12.21.2.14 SwapEndianness64() [2/2]

```
static void OficinaFramework::IOSystem::SwapEndianness64 (
            qword_s & val )  [static]
```

12.21.2.15 SwapEndiannessD()

```
static void OficinaFramework::IOSystem::SwapEndiannessD (
            double & val )  [static]
```

Swaps endianness of double variable.

Parameters

val	Value to be swapped.

12.21.2.16 SwapEndiannessF()

```
static void OficinaFramework::IOSystem::SwapEndiannessF (
            float & val )  [static]
```

Swaps endianness of float variable.

Parameters

val	Value to be swapped.

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

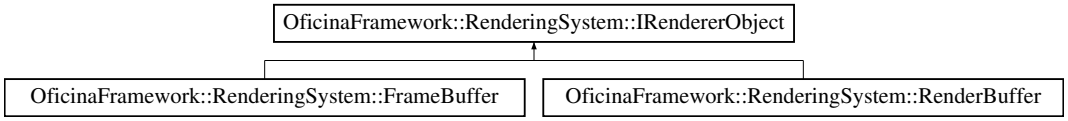
- IOSystem.hpp

12.22 OficinaFramework::RenderingSystem::IRendererObject Class Reference

Interface for GPU-related objects.

`#include <RenderingSystem.hpp>`

Inheritance diagram for OficinaFramework::RenderingSystem::IRendererObject:



Public Member Functions

- virtual ~IRendererObject ()
- virtual GLuint operator() ()=0
- virtual void Bind ()=0
- virtual void Unbind ()=0
- virtual bool IsBound ()=0

12.22.1 Detailed Description

Interface for GPU-related objects.

12.22.2 Constructor & Destructor Documentation

12.22.2.1 ~IRendererObject()

`virtual OficinaFramework::RenderingSystem::IRendererObject::~IRendererObject ( ) [inline], [virtual]`

12.22.3 Member Function Documentation

12.22.3.1 Bind()

`virtual void OficinaFramework::RenderingSystem::IRendererObject::Bind ( ) [pure virtual]`

Implemented in OficinaFramework::RenderingSystem::FrameBuffer, and OficinaFramework::RenderingSystem::RenderBuffer.

12.22.3.2 IsBound()

```
virtual bool OficinaFramework::RenderingSystem::IRendererObject::IsBound ( )  [pure virtual]
```

Implemented in OficinaFramework::RenderingSystem::FrameBuffer, and OficinaFramework::RenderingSystem::↩ RenderBuffer.

12.22.3.3 operator()()

```
virtual GLuint OficinaFramework::RenderingSystem::IRendererObject::operator() ( )  [pure virtual]
```

Implemented in OficinaFramework::RenderingSystem::FrameBuffer, and OficinaFramework::RenderingSystem::↩ RenderBuffer.

12.22.3.4 Unbind()

```
virtual void OficinaFramework::RenderingSystem::IRendererObject::Unbind ( )  [pure virtual]
```

Implemented in OficinaFramework::RenderingSystem::FrameBuffer, and OficinaFramework::RenderingSystem::↩ RenderBuffer.

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

- RenderingSystem.hpp

12.23 OficinaFramework::NetworkSystem Class Reference

Manages all data sending and receiving over network.

```
#include <NetworkSystem.hpp>
```

Classes

- struct Address

  A struct representing an IPv4 address.
- class Socket

  A class representing a socket, used to control ports for comunication with other computers around the web.

Static Public Member Functions

- static void init ()

  Initializes the system.
- static void dispose ()

  Disposes the system.
- static bool IsInitialized ()

  Shows if the network system has been initialized.

Static Public Attributes

- static const word DefaultPort = 1246

  Default socket port for OficinaFramework.

12.23.1 Detailed Description

Manages all data sending and receiving over network.

See also

http://gafferongames.com/

12.23.2 Member Function Documentation

12.23.2.1 dispose()

```
static void OficinaFramework::NetworkSystem::dispose ( )  [static]
```

Disposes the system.

12.23.2.2 init()

```
static void OficinaFramework::NetworkSystem::init ( )  [static]
```

Initializes the system.

Exceptions

SystemInitializationErrorException	

12.23.2.3 IsInitialized()

```
static bool OficinaFramework::NetworkSystem::IsInitialized ( )  [static]
```

Shows if the network system has been initialized.

Returns

Whether it is initialized or not.

12.23.3 Member Data Documentation

12.23.3.1 DefaultPort

```
const word OficinaFramework::NetworkSystem::DefaultPort = 1246  [static]
```

Default socket port for OficinaFramework.

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

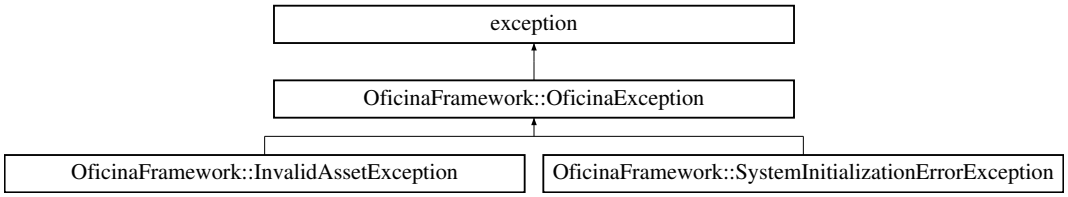
- NetworkSystem.hpp

12.24 OficinaFramework::OficinaException Class Reference

Base class for all framework exceptions.

```
#include <OficinaExceptions.hpp>
```

Inheritance diagram for OficinaFramework::OficinaException:



Public Member Functions

- OficinaException ()
  Instantiates the exception with default error message.
- OficinaException (std::string message)
  Instantiates the exception with a given error message.
- std::string GetMessage ()
  Gets the exception message.
- virtual const char * what () const throw ()
  Returns the message for the actual system exception display.

Protected Attributes

- std::string message

12.24.1 Detailed Description

Base class for all framework exceptions.

12.24.2 Constructor & Destructor Documentation

12.24.2.1 OficinaException() [1/2]

```
OficinaFramework::OficinaException::OficinaException ( )
```

Instantiates the exception with default error message.

12.24.2.2 OficinaException() [2/2]

```
OficinaFramework::OficinaException::OficinaException (
            std::string message )
```

Instantiates the exception with a given error message.

Parameters

message	The message to be set to the exception.

12.24.3 Member Function Documentation

12.24.3.1 GetMessage()

```
std::string OficinaFramework::OficinaException::GetMessage ( )
```

Gets the exception message.

Returns

An std::string containing the exception message.

12.24.3.2 what()

```
virtual const char* OficinaFramework::OficinaException::what ( ) const throw )   [inline],
[virtual]
```

Returns the message for the actual system exception display.

Returns

See also

GetMessage

Reimplemented in OficinaFramework::SystemInitializationErrorException, and OficinaFramework::InvalidAsset↩ Exception.

12.24.4 Member Data Documentation

12.24.4.1 message

```
std::string OficinaFramework::OficinaException::message  [protected]
```

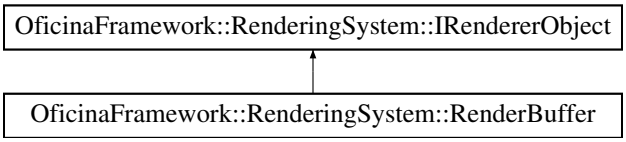
The documentation for this class was generated from the following file:

- OficinaExceptions.hpp

12.25 OficinaFramework::RenderingSystem::RenderBuffer Class Reference

```
#include <RenderingSystem.hpp>
```

Inheritance diagram for OficinaFramework::RenderingSystem::RenderBuffer:



Public Member Functions

- RenderBuffer ()

  Creates a render buffer object.
- ~RenderBuffer ()

  Deletes the render buffer object.
- void Bind () override

  Binds the RenderBuffer to GL_RENDERBUFFER.
- void Unbind () override

  Unbinds the RenderBuffer from GL_RENDERBUFFER.
- bool IsBound () override

  checks if RenderBuffer is bound to GL_RENDERBUFFER.
- GLuint operator() () override

  Provides direct access to the RenderBuffer as OpenGL object.
- void SetFormat (GLenum internalFormat, vec2dw size)

  Formats the RenderBuffer for usage.

12.25.1 Constructor & Destructor Documentation

12.25.1.1 RenderBuffer()

```
OficinaFramework::RenderingSystem::RenderBuffer::RenderBuffer ( )
```

Creates a render buffer object.

Note

The ctor will effectively generate the object on GPU.

12.25.1.2 ~RenderBuffer()

```
OficinaFramework::RenderingSystem::RenderBuffer::~RenderBuffer ( )
```

Deletes the render buffer object.

Note

The dtor will effectively delete the object on GPU.

12.25.2 Member Function Documentation

12.25.2.1 Bind()

```
void OficinaFramework::RenderingSystem::RenderBuffer::Bind ( )  [override], [virtual]
```

Binds the RenderBuffer to GL_RENDERBUFFER.

Implements OficinaFramework::RenderingSystem::IRendererObject.

12.25.2.2 IsBound()

```
bool OficinaFramework::RenderingSystem::RenderBuffer::IsBound ( )  [override], [virtual]
```

checks if RenderBuffer is bound to GL_RENDERBUFFER.

Implements OficinaFramework::RenderingSystem::IRendererObject.

12.25.2.3 operator()()

```
GLuint OficinaFramework::RenderingSystem::RenderBuffer::operator() ( )  [override], [virtual]
```

Provides direct access to the RenderBuffer as OpenGL object.

Returns

The object's name on GPU.

Implements OficinaFramework::RenderingSystem::IRendererObject.

12.25.2.4 SetFormat()

```
void OficinaFramework::RenderingSystem::RenderBuffer::SetFormat (
            GLenum internalFormat,
            vec2dw size )
```

Formats the RenderBuffer for usage.

Parameters

internalFormat	Internal format of the RenderBuffer, depending on its kind.
size	Width and height of the RenderBuffer.

12.25.2.5 Unbind()

```
void OficinaFramework::RenderingSystem::RenderBuffer::Unbind ( )  [override], [virtual]
```

Unbinds the RenderBuffer from GL_RENDERBUFFER.

Implements OficinaFramework::RenderingSystem::IRendererObject.

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

- RenderingSystem.hpp

12.26 OficinaFramework::RenderingSystem Class Reference

Groups rendering-related controls. Use this to allocate and deallocate textures accelerated by GPU, and also for drawing textures or primitives onscreen.

```
#include <RenderingSystem.hpp>
```

Classes

- class Animation

  Represents an Animation, a set of controls for animating objects using SpriteSheets.
- class Font

  Represents a Font, a texture with monospace characters to be used to draw text onscreen.
- class FrameBuffer

  Describes a Frame Buffer object.
- class IRendererObject

  Interface for GPU-related objects.
- class RenderBuffer
- class SpriteSheet

  Represents a Sprite Sheet, a texture containing frames used for animating objects such as characters.
- class Texture

  Represents a texture. Use RenderingSystem::TexturePool to allocate a new Texture.
- class TexturePool

  Represents a structure that can manage the allocation and deallocation of textures.

Public Types

- enum RenderProperty { RENDER_NORMALLY, RENDER_FLIP_X, RENDER_FLIP_Y, RENDER_FLIP_B↩OTH }

  Rendering properties to be used.
- enum RenderEffect {
  MODULATE_EFFECT, REPLACE_EFFECT, CHROMAKEY_EFFECT, INVERSION_EFFECT,
  LIGHT_EFFECT }

  Rendering effects to be used.

Static Public Member Functions

- static void glClearColorM (ColorM c)

  glClearColor equivalent for a color mask (color or (color)ColorDef)
- static void glClearAccumM (ColorM c)

  glClearAccum equivalent for a color mask (color or (color)ColorDef)
- static void glColorM (ColorM c, float transparency)

  glColor equivalent for a color mask (color or (color)ColorDef)
- static void glColorM (ColorM c)

  glColor equivalent for a color mask (color or (color)ColorDef)
- static void SetCameraPosition (vec2 CameraCenter)

  Sets the position of the current viewport.
- static vec2 GetCameraPosition ()

  Gets the central camera position.
- static vec2 GetViewportPosition ()

  Gets the viewport position.
- static void init ()

  Initializes the OpenGL renderer.
- static void dispose ()

  Disposes the active textures and etc.
- static void SetViewportSize (vec2dw size)

  Sets the size of the viewport.
- static vec2dw GetViewportSize ()

  Gets the size of the viewport.
- static void SetResolution (vec2dw res)

  Sets the resolution for the application.
- static vec2dw GetResolution ()

  Gets the current application resolution.
- static void glTranslateToViewportPos ()

  Translates viewport to current camera position.
- static float GetZoomFactor ()

  Gets the camera zoom factor.
- static void SetZoomFactor (float value)

  Sets the camera zoom factor.
- static void DrawRectangle (vec2 Position, vec2 Size, float angle, ColorDef Color, float transparency)

  Draws a rectangle on screen.
- static void DrawTriangle (vec2 BaricenterPosition, float HalfWidth, float HalfHeight, float angle, ColorDef Color, float transparency)

  Draws an equilateral triangle on screen.
- static void DrawTriangle (vec2 BaricenterPosition, vec2 Vertex1, vec2 Vertex2, vec2 Vertex3, float angle, ColorDef Color, float transparency)

  Draws an triangle on screen.
- static void DrawTriangle (vec2 BaricenterPosition, vec2 Vertices[3], float angle, ColorDef Color, float transparency)

  Draws an triangle on screen.
- static void SetLinearFiltering (bool state)

  Disables or enables the state of linear filtering.
- static bool GetLinearFilteringState ()

  Gets the state of linear filtering.
- static bool IsARBDebugActive ()

  Checks for ARB debug (only works with DEBUG_ENABLED define).

- static void CreateDefaultBuffer ()

  Creates the default buffer. Normally used when resizing the window.
- static void DestroyDefaultBuffer ()

  Destroys the default buffer. Normally used when resizing the window.
- static FrameBuffer * GetDefaultBuffer ()

  Gets the default buffer where the scene is rendered to.

12.26.1 Detailed Description

Groups rendering-related controls. Use this to allocate and deallocate textures accelerated by GPU, and also for drawing textures or primitives onscreen.

Todo • Ditch fixed pipeline completely in favor of using shaders.

12.26.2 Member Enumeration Documentation

12.26.2.1 RenderEffect

`enum OficinaFramework::RenderingSystem::RenderEffect`

Rendering effects to be used.

Note

For advanced effects, please consider using GLSL Shaders.

Enumerator

MODULATE_EFFECT	Modulate alpha and tint onto texel color.
REPLACE_EFFECT	Skip tint and non-100% alpha and use texel colors.
CHROMAKEY_EFFECT	Replace alpha by tint.
INVERSION_EFFECT	Modulate alpha and tint onto texel color, then invert it.
LIGHT_EFFECT	Add alpha and tint onto texel color, then also add top colors.

12.26.2.2 RenderProperty

`enum OficinaFramework::RenderingSystem::RenderProperty`

Rendering properties to be used.

Enumerator

RENDER_NORMALLY	Renders normally.
RENDER_FLIP_X	Renders flipping on X axis.
RENDER_FLIP_Y	Renders flipping on Y axis.
RENDER_FLIP_BOTH	Renders flippling both axis.

12.26.3 Member Function Documentation

12.26.3.1 CreateDefaultBuffer()

```
static void OficinaFramework::RenderingSystem::CreateDefaultBuffer ( )  [static]
```

Creates the default buffer. Normally used when resizing the window.

Warning

Do not use this unless you are aware of what you're doing!

12.26.3.2 DestroyDefaultBuffer()

```
static void OficinaFramework::RenderingSystem::DestroyDefaultBuffer ( )  [static]
```

Destroys the default buffer. Normally used when resizing the window.

Warning

Do not use this unless you are aware of what you're doing!

12.26.3.3 dispose()

```
static void OficinaFramework::RenderingSystem::dispose ( )  [static]
```

Disposes the active textures and etc.

12.26.3.4 DrawRectangle()

```
static void OficinaFramework::RenderingSystem::DrawRectangle (
            vec2 Position,
            vec2 Size,
            float angle,
            ColorDef Color,
            float transparency )  [static]
```

Draws a rectangle on screen.

Parameters

Position	Position of the upper left corner of the rectangle on screen.
Size	Size of the rectangle.
angle	Angle of the rectangle around the upper left corner, in degrees. Defaults to 0.0f.
Color	Color of the rectangle.
transparency	Alpha ratio of the rectangle. Defaults to 1.0f.

12.26.3.5 DrawTriangle() [1/3]

```
static void OficinaFramework::RenderingSystem::DrawTriangle (
            vec2 BaricenterPosition,
            float HalfWidth,
            float HalfHeight,
            float angle,
            ColorDef Color,
            float transparency )  [static]
```

Draws an equilateral triangle on screen.

Parameters

BaricenterPosition	Position of the exact triangle center.
HalfWidth	Half size of bottom edge's length.
HalfHeight	Half of the size from the straight line described from top vertex to bottom edge.
angle	Angle for turning the triangle towards baricenter, in degrees. Defaults to 0.0f.
Color	Color of the triangle.
transparency	Alpha ratio of the rectangle. Defaults to 1.0f.

12.26.3.6 DrawTriangle() [2/3]

```
static void OficinaFramework::RenderingSystem::DrawTriangle (
            vec2 BaricenterPosition,
            vec2 Vertex1,
            vec2 Vertex2,
            vec2 Vertex3,
            float angle,
            ColorDef Color,
            float transparency )  [static]
```

Draws an triangle on screen.

Parameters

BaricenterPosition	Position of the exact triangle center.
Vertex1	First vertex of the triangle.
Vertex2	Second vertex of the triangle.
Vertex3	Third vertex of the triangle.
angle	Angle for turning the triangle towards baricenter, in degrees. Defaults to 0.0f.
Color	Color of the triangle.
transparency	Alpha ratio of the rectangle. Defaults to 1.0f.

12.26.3.7 DrawTriangle() [3/3]

```
static void OficinaFramework::RenderingSystem::DrawTriangle (
            vec2 BaricenterPosition,
            vec2 Vertices[3],
            float angle,
            ColorDef Color,
            float transparency )  [static]
```

Draws an triangle on screen.

Parameters

BaricenterPosition	Position of the exact triangle center.
Vertices	Vertices of the triangle.
angle	Angle for turning the triangle towards baricenter, in degrees. Defaults to 0.0f.
Color	Color of the triangle.
transparency	Alpha ratio of the rectangle. Defaults to 1.0f.

12.26.3.8 GetCameraPosition()

```
static vec2 OficinaFramework::RenderingSystem::GetCameraPosition ( )  [static]
```

Gets the central camera position.

Returns

The center position of the camera related to the viewport.

12.26.3.9 GetDefaultBuffer()

```
static FrameBuffer* OficinaFramework::RenderingSystem::GetDefaultBuffer ( )  [static]
```

Gets the default buffer where the scene is rendered to.

12.26.3.10 GetLinearFilteringState()

```
static bool OficinaFramework::RenderingSystem::GetLinearFilteringState ( )  [static]
```

Gets the state of linear filtering.

Returns

Whether linear filtering is active or inactive.

12.26.3.11 GetResolution()

```
static vec2dw OficinaFramework::RenderingSystem::GetResolution ( )  [static]
```

Gets the current application resolution.

Returns

Current internal resolution, despite resizing.

12.26.3.12 GetViewportPosition()

```
static vec2 OficinaFramework::RenderingSystem::GetViewportPosition ( )  [static]
```

Gets the viewport position.

Returns

The upper top position of the viewport.

12.26.3.13 GetViewportSize()

```
static vec2dw OficinaFramework::RenderingSystem::GetViewportSize ( )  [static]
```

Gets the size of the viewport.

Returns

The viewport size.

12.26.3.14 GetZoomFactor()

```
static float OficinaFramework::RenderingSystem::GetZoomFactor ( )  [static]
```

Gets the camera zoom factor.

Returns

Camera zoom factor. 1.0f equals 100%.

12.26.3.15 glClearAccumM()

```
static void OficinaFramework::RenderingSystem::glClearAccumM (
            ColorM c )  [static]
```

glClearAccum equivalent for a color mask (color or (color)ColorDef)

Parameters

c	Color to fill the accumulation buffer after cleaning it.

12.26.3.16 glClearColorM()

```
static void OficinaFramework::RenderingSystem::glClearColorM (
            ColorM c )  [static]
```

glClearColor equivalent for a color mask (color or (color)ColorDef)

Parameters

c	Color to fill the back buffer after cleaning it.

12.26.3.17 glColorM() [1/2]

```
static void OficinaFramework::RenderingSystem::glColorM (
            ColorM c,
            float transparency )  [static]
```

glColor equivalent for a color mask (color or (color)ColorDef)

Parameters

c	Color to bind to the next vertex (before the call of another glColor function).
transparency	Alpha ratio of the color.

12.26.3.18 glColorM() [2/2]

```
static void OficinaFramework::RenderingSystem::glColorM (
            ColorM c )  [static]
```

glColor equivalent for a color mask (color or (color)ColorDef)

Parameters

c	Color to bind to the next vertex (before the call of another glColor function).

12.26.3.19 glTranslateToViewportPos()

```
static void OficinaFramework::RenderingSystem::glTranslateToViewportPos ( )  [static]
```

Translates viewport to current camera position.

12.26.3.20 init()

```
static void OficinaFramework::RenderingSystem::init ( )  [static]
```

Initializes the OpenGL renderer.

12.26.3.21 IsARBDebugActive()

```
static bool OficinaFramework::RenderingSystem::IsARBDebugActive ( )  [static]
```

Checks for ARB debug (only works with DEBUG_ENABLED define).

Returns

Whether ARB debug is active or not

12.26.3.22 SetCameraPosition()

```
static void OficinaFramework::RenderingSystem::SetCameraPosition (
            vec2 CameraCenter )  [static]
```

Sets the position of the current viewport.

Parameters

CameraCenter	Float-precision vec2 camera center position.

12.26.3.23 SetLinearFiltering()

```
static void OficinaFramework::RenderingSystem::SetLinearFiltering (
            bool state )  [static]
```

Disables or enables the state of linear filtering.

Parameters

state	Whether linear filtering should be active or inactive.

12.26.3.24 SetResolution()

```
static void OficinaFramework::RenderingSystem::SetResolution (
            vec2dw res )  [static]
```

Sets the resolution for the application.

Parameters

res	Desired resolution.

12.26.3.25 SetViewportSize()

```
static void OficinaFramework::RenderingSystem::SetViewportSize (
            vec2dw size )  [static]
```

Sets the size of the viewport.

Parameters

size	New size for the viewport.

12.26.3.26 SetZoomFactor()

```
static void OficinaFramework::RenderingSystem::SetZoomFactor (
            float value )  [static]
```

Sets the camera zoom factor.

Parameters

value	Zoom factor to be given to camera. 1.0f equals 100%.

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

- RenderingSystem.hpp

12.27 OficinaFramework::ScreenSystem::Screen Class Reference

A class representing a Screen to be rendered on the screen manager.

```
#include <ScreenSystem.hpp>
```

Public Member Functions

- virtual ~Screen ()

  Destroys the screen and disposes it.
- virtual void Initialize ()=0

  Initializes the screen, with the first logical values.
- virtual void ReInitialize ()

  Reinitializes the screen. Used to reposition objects when a reinitialization is needed, but content must not be unloaded.
- virtual void LoadContent ()=0

  Loads screen content, such as textures and other data.
- virtual void UnloadContent ()=0

  Unloads screen content, such as textures and other data.
- virtual void Update ()=0

  Updates the screen logic.
- virtual void Draw ()=0

  Draws the screen graphics.
- bool IsRemovable () const

  Gets if this screen was set to be removed and disposed.
- bool IsContentLoaded () const

  Gets if the screen had its content data fully loaded.
- bool IsActive () const

  Gets if the screen is active and can be updated.
- bool IsVisible () const

  Gets if the screen is visible and can be drawn.
- bool IsInitialized () const

  Gets if the screeen is initialized.
- virtual void RemoveMe () final

  Sets the screen to be removed and be disposed.
- virtual void SetActive (bool state) final

  Sets the screen active or not, changing its ability to be updated.
- virtual void SetVisible (bool state) final

  Sets the screen visible or not, changing its ability to be drawn.
- virtual void ReInitMe () final

  Sets the screen to be reinitialized.
- virtual void ReInitMe_dont () final

  Unsets the screen to reinitialization.
- bool IsReinitializable () const

  Gets if the screen was set to be reinitialized.
- virtual float GetDepth () final

  Gets the depth of the current screen. The lower the depth, the more on the back it is.
- virtual void SetDepth (float depth) final

  Sets the depth of the current screen. The lower the depth, the more on the back it is.

12.27.1 Detailed Description

A class representing a Screen to be rendered on the screen manager.

12.27.2 Constructor & Destructor Documentation

12.27.2.1 ~Screen()

```
virtual OficinaFramework::ScreenSystem::Screen::~Screen ( )  [virtual]
```

Destroys the screen and disposes it.

12.27.3 Member Function Documentation

12.27.3.1 Draw()

```
virtual void OficinaFramework::ScreenSystem::Screen::Draw ( )  [pure virtual]
```

Draws the screen graphics.

12.27.3.2 GetDepth()

```
virtual float OficinaFramework::ScreenSystem::Screen::GetDepth ( )  [final], [virtual]
```

Gets the depth of the current screen. The lower the depth, the more on the back it is.

Returns

Depth of the screen on the manager.

12.27.3.3 Initialize()

```
virtual void OficinaFramework::ScreenSystem::Screen::Initialize ( )  [pure virtual]
```

Initializes the screen, with the first logical values.

Attention

Call this method to fully complement your derived class.

12.27.3.4 IsActive()

```
bool OficinaFramework::ScreenSystem::Screen::IsActive ( ) const
```

Gets if the screen is active and can be updated.

Returns

Whether the screen can be updated or not.

Note

Use this method inside your override of the Update method to separate the code that must still be executed, even though the screen is inactive.

12.27.3.5 IsContentLoaded()

```
bool OficinaFramework::ScreenSystem::Screen::IsContentLoaded ( ) const
```

Gets if the screen had its content data fully loaded.

Returns

Wether the screen had its assets fully loaded or not.

12.27.3.6 IsInitialized()

```
bool OficinaFramework::ScreenSystem::Screen::IsInitialized ( ) const
```

Gets if the screeen is initialized.

Returns

Whether the screen was initialized or not.

12.27.3.7 IsReinitializable()

```
bool OficinaFramework::ScreenSystem::Screen::IsReinitializable ( ) const
```

Gets if the screen was set to be reinitialized.

12.27.3.8 IsRemovable()

```
bool OficinaFramework::ScreenSystem::Screen::IsRemovable ( ) const
```

Gets if this screen was set to be removed and disposed.

Returns

Whether the screen is set to be removed or not.

12.27.3.9 IsVisible()

```
bool OficinaFramework::ScreenSystem::Screen::IsVisible ( ) const
```

Gets if the screen is visible and can be drawn.

Returns

Whether the screen can be drawn or not.

12.27.3.10 LoadContent()

```
virtual void OficinaFramework::ScreenSystem::Screen::LoadContent ( )  [pure virtual]
```

Loads screen content, such as textures and other data.

Attention

To prevent early usage of unallocated data, you must ALWAYS call Screen::LoadContent on the end of this method's override.

12.27.3.11 ReInitialize()

```
virtual void OficinaFramework::ScreenSystem::Screen::ReInitialize ( )  [virtual]
```

Reinitializes the screen. Used to reposition objects when a reinitialization is needed, but content must not be unloaded.

Attention

Implementation is optional, but is all up to the developer.

12.27.3.12 ReInitMe()

```
virtual void OficinaFramework::ScreenSystem::Screen::ReInitMe ( )  [final], [virtual]
```

Sets the screen to be reinitialized.

12.27.3.13 ReInitMe_dont()

```
virtual void OficinaFramework::ScreenSystem::Screen::ReInitMe_dont ( )  [final], [virtual]
```

Unsets the screen to reinitialization.

12.27.3.14 RemoveMe()

```
virtual void OficinaFramework::ScreenSystem::Screen::RemoveMe ( )  [final], [virtual]
```

Sets the screen to be removed and be disposed.

12.27.3.15 SetActive()

```
virtual void OficinaFramework::ScreenSystem::Screen::SetActive (
            bool state )  [final], [virtual]
```

Sets the screen active or not, changing its ability to be updated.

Parameters

state	The state of activity to be given to the screen.

12.27.3.16 SetDepth()

```
virtual void OficinaFramework::ScreenSystem::Screen::SetDepth (
            float depth )  [final], [virtual]
```

Sets the depth of the current screen. The lower the depth, the more on the back it is.

Parameters

depth	Depth to be given to the screen on the manager.

12.27.3.17 SetVisible()

```
virtual void OficinaFramework::ScreenSystem::Screen::SetVisible (
            bool state )  [final], [virtual]
```

Sets the screen visible or not, changing its ability to be drawn.

Parameters

state	The state of drawability to be given to the screen.

12.27.3.18 UnloadContent()

```
virtual void OficinaFramework::ScreenSystem::Screen::UnloadContent ( )  [pure virtual]
```

Unloads screen content, such as textures and other data.

12.27.3.19 Update()

```
virtual void OficinaFramework::ScreenSystem::Screen::Update ( )  [pure virtual]
```

Updates the screen logic.

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

- ScreenSystem.hpp

12.28 OficinaFramework::ScreenSystem Class Reference

Groups screen management controls. Use this class to add/remove screens and set them active or inactive.

```
#include <ScreenSystem.hpp>
```

Classes

- class Screen

  A class representing a Screen to be rendered on the screen manager.

Static Public Member Functions

- static void init (std::string windowname, vec2dw windowsize, bool fullscreen, std::string iconpath, bool vsync)

  Initializes the Window and the Screen System.
- static void dispose ()

  Disposes the Screen System and deletes the Window.
- static void AddScreen (Screen *scr)

  Enqueues a screen to be added to the list.
- static void AddScreen (Screen *scr, float Depth)

  Enqueues a screen to be added to the list.
- static void RemoveScreen (Screen *scr)

  Sets a screen to be removed and disposed.
- static void UnloadAllScreens ()

  Unloads the content of all screen on the list, then removes them.
- static void UpdateScreens ()

  Updates the logic of all screens on the list.
- static void DrawScreens ()

  Draws all screens that are not marked as invisible.
- static void ClearScreens ()

  Clears all screens from the list. This will remove the screens from list and dispose them.
- static void SortScreens ()

  Sorts screens by depth on the manager. Useful if you manually changed the depth of a certain screen.
- static float GetBiggestDepth ()

  Gets the biggest depth value available on the manager.
- static float GetSmallestDepth ()

  Gets the smallest depth value available on the manager.
- static SDL_Window * GetWindowHandle ()

  Offers direct access to the SDL2 window handle.
- static vec2dw GetWindowSize ()

  Gets the window size (do not confuse with viewport).
- static void SetWindowSize (vec2dw size)

  Sets the window size (do not confuse with viewport).
- static bool IsFullScreen ()

  Gets the window mode.
- static void SetFullScreen (bool state)

  Sets the fullscreen mode.
- static bool IsDebugActive ()

  Gets if the debug mode is active.
- static void SetDebug (bool state)

  Sets the debug state.
- static void Debug_AddLine (std::string text)

  Adds a text line to the debugger.
- static void Debug_SetFont (RenderingSystem::Font *font)

  Sets the main font to be used on the debugger.
- static void Debug_ToggleMinimalist ()

  Toggles debug as minimalistic or not.
- static void SwapWindow ()

  Swaps window for a new draw routine.
- static void ClearWindow ()

  Clears window of artifacts.
- static bool IsLoadingThreadBusy ()

  Gets whether the loading thread is currently loading screens or not.

Friends

- class EngineCore

12.28.1 Detailed Description

Groups screen management controls. Use this class to add/remove screens and set them active or inactive.

12.28.2 Member Function Documentation

12.28.2.1 AddScreen() [1/2]

```
static void OficinaFramework::ScreenSystem::AddScreen (
            Screen * scr )  [static]
```

Enqueues a screen to be added to the list.

Parameters

scr	Pointer to the screen to be added.

Exceptions

OficinaException	Runtime error exception, if pointer is null.

12.28.2.2 AddScreen() [2/2]

```
static void OficinaFramework::ScreenSystem::AddScreen (
            Screen * scr,
            float Depth )  [static]
```

Enqueues a screen to be added to the list.

Parameters

scr	Pointer to the screen to be added.
Depth	Depth to be given to the screen on initialization. This will reflect on update and draw order.

Exceptions

OficinaException	Runtime error exception, if pointer is null.

12.28.2.3 ClearScreens()

```
static void OficinaFramework::ScreenSystem::ClearScreens ( )  [static]
```

Clears all screens from the list. This will remove the screens from list and dispose them.

12.28.2.4 ClearWindow()

```
static void OficinaFramework::ScreenSystem::ClearWindow ( )  [static]
```

Clears window of artifacts.

12.28.2.5 Debug_AddLine()

```
static void OficinaFramework::ScreenSystem::Debug_AddLine (
            std::string text )  [static]
```

Adds a text line to the debugger.

Parameters

text	The text to be added to the debugger.

12.28.2.6 Debug_SetFont()

```
static void OficinaFramework::ScreenSystem::Debug_SetFont (
            RenderingSystem::Font * font )  [static]
```

Sets the main font to be used on the debugger.

Parameters

font	A pointer to the font to be used.

Warning

Please allocate a font that will be only used on the debugger. The system itself will unload the font when necessary.

12.28.2.7 Debug_ToggleMinimalist()

```
static void OficinaFramework::ScreenSystem::Debug_ToggleMinimalist ( )  [static]
```

Toggles debug as minimalistic or not.

12.28.2.8 dispose()

```
static void OficinaFramework::ScreenSystem::dispose ( )  [static]
```

Disposes the Screen System and deletes the Window.

12.28.2.9 DrawScreens()

```
static void OficinaFramework::ScreenSystem::DrawScreens ( )  [static]
```

Draws all screens that are not marked as invisible.

12.28.2.10 GetBiggestDepth()

```
static float OficinaFramework::ScreenSystem::GetBiggestDepth ( )  [static]
```

Gets the biggest depth value available on the manager.

Warning

This method will assume that the screen list is sorted!

Returns

Biggest depth value of the screen that is most in front.

12.28.2.11 GetSmallestDepth()

```
static float OficinaFramework::ScreenSystem::GetSmallestDepth ( )  [static]
```

Gets the smallest depth value available on the manager.

Warning

This method will assume that the screen list is sorted!

Returns

Smallest depth value of the screen that is most in back.

12.28.2.12 GetWindowHandle()

```
static SDL_Window* OficinaFramework::ScreenSystem::GetWindowHandle ( )  [static]
```

Offers direct access to the SDL2 window handle.

Warning

Be careful when dealing directly with this. Oficina manipulates the window directly on initialization and runtime, so there is no need for advanced operations, unless you need the handle for third-party APIs.

Returns

Pointer for the SDL2 Window.

12.28.2.13 GetWindowSize()

```
static vec2dw OficinaFramework::ScreenSystem::GetWindowSize ( )  [static]
```

Gets the window size (do not confuse with viewport).

Returns

A vec2 with dword precision with the window size.

12.28.2.14 init()

```
static void OficinaFramework::ScreenSystem::init (
            std::string windowname,
            vec2dw windowsize,
            bool fullscreen,
            std::string iconpath,
            bool vsync )  [static]
```

Initializes the Window and the Screen System.

Parameters

windowname	Caption of the main window.
windowsize	Size of the window.
fullscreen	Whether the main window is in fullscreen mode or not.
iconpath	Path to the main window's icon. Can be set to an empty string in case none is intended.
vsync	Whether VSync is active or not.

Exceptions

SystemInitializationErrorException	InvalidAssetException

12.28.2.15 IsDebugActive()

```
static bool OficinaFramework::ScreenSystem::IsDebugActive ( )  [static]
```

Gets if the debug mode is active.

Returns

Whether debug is active or not.

12.28.2.16 IsFullScreen()

```
static bool OficinaFramework::ScreenSystem::IsFullScreen ( )  [static]
```

Gets the window mode.

Returns

Whether the window is fullscreen or not.

12.28.2.17 IsLoadingThreadBusy()

```
static bool OficinaFramework::ScreenSystem::IsLoadingThreadBusy ( )  [static]
```

Gets whether the loading thread is currently loading screens or not.

Returns

If the loading thread is in a work.

12.28.2.18 RemoveScreen()

```
static void OficinaFramework::ScreenSystem::RemoveScreen (
            Screen * scr )  [static]
```

Sets a screen to be removed and disposed.

Parameters

scr	A pointer to the screen to be removed.

12.28.2.19 SetDebug()

```
static void OficinaFramework::ScreenSystem::SetDebug (
            bool state )  [static]
```

Sets the debug state.

Parameters

state	State of the debug to be set to.

12.28.2.20 SetFullScreen()

```
static void OficinaFramework::ScreenSystem::SetFullScreen (
            bool state )  [static]
```

Sets the fullscreen mode.

Parameters

state	Whether to set the display to fullscreen or not.

12.28.2.21 SetWindowSize()

```
static void OficinaFramework::ScreenSystem::SetWindowSize (
            vec2dw size )  [static]
```

Sets the window size (do not confuse with viewport).

Parameters

size	A vec2 with dword precision indicating new window size.

12.28.2.22 SortScreens()

```
static void OficinaFramework::ScreenSystem::SortScreens ( )  [static]
```

Sorts screens by depth on the manager. Useful if you manually changed the depth of a certain screen.

Warning

Sorting screens on gameloop may give unexpected behaviour on the current frame.

12.28.2.23 SwapWindow()

```
static void OficinaFramework::ScreenSystem::SwapWindow ( )  [static]
```

Swaps window for a new draw routine.

12.28.2.24 UnloadAllScreens()

```
static void OficinaFramework::ScreenSystem::UnloadAllScreens ( )  [static]
```

Unloads the content of all screen on the list, then removes them.

12.28.2.25 UpdateScreens()

```
static void OficinaFramework::ScreenSystem::UpdateScreens ( )  [static]
```

Updates the logic of all screens on the list.

12.28.3 Friends And Related Function Documentation

12.28.3.1 EngineCore

```
friend class EngineCore  [friend]
```

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

- ScreenSystem.hpp

12.29 OficinaFramework::IOSystem::ScriptStream Class Reference

Reads a script as a byte stream.

```
#include <IOSystem.hpp>
```

Public Member Functions

- ScriptStream ()
  Creates an empty script stream.
- ScriptStream (byte *bytestream, qword_s size)
  Creates a script stream, opening a bytestream.
- ~ScriptStream ()
  Disposes a script stream.
- void open (byte *bytestream, qword_s size)
  Opens a bytestream as script.
- bool read (void *data, size_t size)
  Reads from the bytestream.
- void close ()
  Closes the bytestream.
- bool eof ()
  Checks for EOF.
- qword_s scriptsize ()
  Gets the script size.
- qword_s toEOF ()
  Gets the number of bytes until EOF.

12.29.1 Detailed Description

Reads a script as a byte stream.

12.29.2 Constructor & Destructor Documentation

12.29.2.1 ScriptStream() [1/2]

```
OficinaFramework::IOSystem::ScriptStream::ScriptStream ( )
```

Creates an empty script stream.

12.29.2.2 ScriptStream() [2/2]

```
OficinaFramework::IOSystem::ScriptStream::ScriptStream (
            byte * bytestream,
            qword_s size )
```

Creates a script stream, opening a bytestream.

Parameters

bytestream	Raw bytes for the stream.
size	Size of the byte stream.

12.29.2.3 ~ScriptStream()

```
OficinaFramework::IOSystem::ScriptStream::~ScriptStream ( )
```

Disposes a script stream.

12.29.3 Member Function Documentation

12.29.3.1 close()

```
void OficinaFramework::IOSystem::ScriptStream::close ( )
```

Closes the bytestream.

12.29.3.2 eof()

```
bool OficinaFramework::IOSystem::ScriptStream::eof ( )
```

Checks for EOF.

Returns

Whether EOF was reached or not.

12.29.3.3 open()

```
void OficinaFramework::IOSystem::ScriptStream::open (
            byte * bytestream,
            qword_s size )
```

Opens a bytestream as script.

Parameters

bytestream	Raw bytes for the stream.
size	Size of the byte stream.

12.29.3.4 read()

```
bool OficinaFramework::IOSystem::ScriptStream::read (
```

```
            void * data,
            size_t size )
```

Reads from the bytestream.

Parameters

data	Pointer to the destination.
size	Number of bytes to be read.

Returns

Whether the number of bytes was read or not. If returned 'false', means that either the stream was not initialized, or the required number of bytes surpasses the length between the stream pointer and EOF.

12.29.3.5 scriptsize()

```
qword_s OficinaFramework::IOSystem::ScriptStream::scriptsize ( )
```

Gets the script size.

Returns

Number of bytes on the script.

12.29.3.6 toEOF()

```
qword_s OficinaFramework::IOSystem::ScriptStream::toEOF ( )
```

Gets the number of bytes until EOF.

Returns

Number of bytes until EOF is reached.

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

- IOSystem.hpp

12.30 OficinaFramework::IOSystem::ScriptTools Class Reference

A class for opening and loading Gongly Script data. Works since Gongly Script v1.0.

```
#include <IOSystem.hpp>
```

Public Member Functions

- ScriptTools (ScriptStream ∗stream)

  Creates a ScriptTools and opens a specific script.
- ScriptTools ()

  Creates a new ScriptTools with no file loaded.
- ∼ScriptTools ()

  Disposes the ScriptTools.
- void LoadScript (ScriptStream ∗stream)

  Loads a specific script.
- void UnloadScript ()

  Unloads a script, if loaded.
- bool IsScriptLoaded ()

  Checks for loaded scripts.
- uint16_t GetEntityID ()

  Retrieves the next entity's ID and checks for EOF.
- int GetInt ()

  Retrieves an INT (32-bit signed int) from the loaded script stream.
- float GetFloat ()

  Retrieves a FLOAT (16-bit float) from the loaded script stream.
- double GetDouble ()

  Retrieves a DOUBLE (32-bit float) from the loaded script stream.
- bool GetBool ()

  Retrieves a BOOL (8-bit int, translated to C++ bool type) from the loaded script stream.
- char ∗ GetString ()

  Retrieves a STRING (32-bit unsigned int with number of characters + n_characters ∗ signed char, translated to C++ char∗ type) from the loaded script stream.
- vec2 GetVec2 ()

  Retrieves a VEC2 (2 ∗ 16-bit float, translated to Oficina's vec2 type) from the loaded script stream.
- vec3 GetVec3 ()

  Retrieves a VEC3 (3 ∗ 16-bit float, translated to Oficina's vec3 type) from the loaded script stream.
- Color4 GetVec4 ()

  Retrieves a VEC4 (4 ∗ 16-bit float, translated to Oficina's Color4 type) from the loaded script stream.
- byte GetByte ()

  Retrieves a BYTE (8-bit unsigned int, byte) from the loaded script stream.
- word GetWord ()

  Retrieves a WORD (16-bit unsigned int, word) from the loaded script stream.
- dword GetDword ()

  Retrieves a DWORD (32-bit unsigned int, dword) from the loaded script stream.
- qword GetQword ()

  Retrieves a QWORD (64-bit unsigned int, qword) from the loaded script stream.
- byte_s GetByte_s ()

  Retrieves a BYTE_S (8-bit signed int, byte_s) from the loaded script stream.
- word_s GetWord_s ()

  Retrieves a WORD_S (16-bit signed int, word_s) from the loaded script stream.
- dword_s GetDword_s ()

  Retrieves a DWORD_S (32-bit signed int, dword_s) from the loaded script stream.
- qword_s GetQword_s ()

  Retrieves a QWORD_S (64-bit signed int, qword_s) from the loaded script stream.
- bool IsEOF ()

  Checks if script has reached EOF manually.
- qword_s ToEOF ()

  Gets the number of bytes until EOF.

12.30.1 Detailed Description

A class for opening and loading Gongly Script data. Works since Gongly Script v1.0.

12.30.2 Constructor & Destructor Documentation

12.30.2.1 ScriptTools() [1/2]

```
OficinaFramework::IOSystem::ScriptTools::ScriptTools (
            ScriptStream * stream )
```

Creates a ScriptTools and opens a specific script.

Parameters

stream	Byte stream containing the script data.

12.30.2.2 ScriptTools() [2/2]

```
OficinaFramework::IOSystem::ScriptTools::ScriptTools ( )  [inline]
```

Creates a new ScriptTools with no file loaded.

References OficinaFramework::IOSystem::IsInitialized(), OficinaFramework::IOSystem::SwapEndianness16(), OficinaFramework::IOSystem::SwapEndianness32(), OficinaFramework::IOSystem::SwapEndianness64(), OficinaFramework::IOSystem::SwapEndiannessD(), and OficinaFramework::IOSystem::SwapEndiannessF().

12.30.2.3 ~ScriptTools()

```
OficinaFramework::IOSystem::ScriptTools::~ScriptTools ( )
```

Disposes the ScriptTools.

12.30.3 Member Function Documentation

12.30.3.1 GetBool()

```
bool OficinaFramework::IOSystem::ScriptTools::GetBool ( )
```

Retrieves a BOOL (8-bit int, translated to C++ bool type) from the loaded script stream.

12.30.3.2 GetByte()

```
byte OficinaFramework::IOSystem::ScriptTools::GetByte ( )
```

Retrieves a BYTE (8-bit unsigned int, byte) from the loaded script stream.

12.30.3.3 GetByte_s()

```
byte_s OficinaFramework::IOSystem::ScriptTools::GetByte_s ( )
```

Retrieves a BYTE_S (8-bit signed int, byte_s) from the loaded script stream.

12.30.3.4 GetDouble()

```
double OficinaFramework::IOSystem::ScriptTools::GetDouble ( )
```

Retrieves a DOUBLE (32-bit float) from the loaded script stream.

12.30.3.5 GetDword()

```
dword OficinaFramework::IOSystem::ScriptTools::GetDword ( )
```

Retrieves a DWORD (32-bit unsigned int, dword) from the loaded script stream.

12.30.3.6 GetDword_s()

```
dword_s OficinaFramework::IOSystem::ScriptTools::GetDword_s ( )
```

Retrieves a DWORD_S (32-bit signed int, dword_s) from the loaded script stream.

12.30.3.7 GetEntityID()

```
uint16_t OficinaFramework::IOSystem::ScriptTools::GetEntityID ( )
```

Retrieves the next entity's ID and checks for EOF.

Returns

0 for end-of-file, or the ID of the entity. First valid value is 1.

12.30.3.8 GetFloat()

```
float OficinaFramework::IOSystem::ScriptTools::GetFloat ( )
```

Retrieves a FLOAT (16-bit float) from the loaded script stream.

12.30.3.9 GetInt()

```
int OficinaFramework::IOSystem::ScriptTools::GetInt ( )
```

Retrieves an INT (32-bit signed int) from the loaded script stream.

12.30.3.10 GetQword()

```
qword OficinaFramework::IOSystem::ScriptTools::GetQword ( )
```

Retrieves a QWORD (64-bit unsigned int, qword) from the loaded script stream.

12.30.3.11 GetQword_s()

```
qword_s OficinaFramework::IOSystem::ScriptTools::GetQword_s ( )
```

Retrieves a QWORD_S (64-bit signed int, qword_s) from the loaded script stream.

12.30.3.12 GetString()

```
char* OficinaFramework::IOSystem::ScriptTools::GetString ( )
```

Retrieves a STRING (32-bit unsigned int with number of characters + n_characters * signed char, translated to C++ char* type) from the loaded script stream.

12.30.3.13 GetVec2()

```
vec2 OficinaFramework::IOSystem::ScriptTools::GetVec2 ( )
```

Retrieves a VEC2 (2 * 16-bit float, translated to Oficina's vec2 type) from the loaded script stream.

12.30.3.14 GetVec3()

```
vec3 OficinaFramework::IOSystem::ScriptTools::GetVec3 ( )
```

Retrieves a VEC3 (3 * 16-bit float, translated to Oficina's vec3 type) from the loaded script stream.

12.30.3.15 GetVec4()

```
Color4 OficinaFramework::IOSystem::ScriptTools::GetVec4 ( )
```

Retrieves a VEC4 (4 ∗ 16-bit float, translated to Oficina's Color4 type) from the loaded script stream.

12.30.3.16 GetWord()

```
word OficinaFramework::IOSystem::ScriptTools::GetWord ( )
```

Retrieves a WORD (16-bit unsigned int, word) from the loaded script stream.

12.30.3.17 GetWord_s()

```
word_s OficinaFramework::IOSystem::ScriptTools::GetWord_s ( )
```

Retrieves a WORD_S (16-bit signed int, word_s) from the loaded script stream.

12.30.3.18 IsEOF()

```
bool OficinaFramework::IOSystem::ScriptTools::IsEOF ( )
```

Checks if script has reached EOF manually.

12.30.3.19 IsScriptLoaded()

```
bool OficinaFramework::IOSystem::ScriptTools::IsScriptLoaded ( )
```

Checks for loaded scripts.

Returns

Whether any script is loaded or not.

12.30.3.20 LoadScript()

```
void OficinaFramework::IOSystem::ScriptTools::LoadScript (
            ScriptStream * stream )
```

Loads a specific script.

Parameters

stream	Byte stream containing the script data.

12.30.3.21 ToEOF()

```
qword_s OficinaFramework::IOSystem::ScriptTools::ToEOF ( )
```

Gets the number of bytes until EOF.

Returns

Number of bytes until EOF is reached.

12.30.3.22 UnloadScript()

```
void OficinaFramework::IOSystem::ScriptTools::UnloadScript ( )
```

Unloads a script, if loaded.

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

- IOSystem.hpp

12.31 OficinaFramework::NetworkSystem::Socket Class Reference

A class representing a socket, used to control ports for comunication with other computers around the web.

```
#include <NetworkSystem.hpp>
```

Public Member Functions

- Socket ()
  Constructs a Socket.
- ~Socket ()
  Destructs a Socket.
- bool Open (word port)
  Opens a port for this socket.
- void Close ()
  Closes the port for this socket.
- bool Send (void *data, size_t datasize, Address destination)
  Sends packet to a specific address.
- bool Receive (Address &recv_sender, void *recv_data, size_t expected_size)
  Receives a packet.
- word GetNetworkPort () const
  Gets the port binded by the socket.
- void SetNetworkPort (word port)
  Sets the port for a socket, if not initialized.

12.31.1 Detailed Description

A class representing a socket, used to control ports for comunication with other computers around the web.

12.31.2 Constructor & Destructor Documentation

12.31.2.1 Socket()

```
OficinaFramework::NetworkSystem::Socket::Socket ( )
```

Constructs a Socket.

12.31.2.2 ~Socket()

```
OficinaFramework::NetworkSystem::Socket::~Socket ( )
```

Destructs a Socket.

12.31.3 Member Function Documentation

12.31.3.1 Close()

```
void OficinaFramework::NetworkSystem::Socket::Close ( )
```

Closes the port for this socket.

12.31.3.2 GetNetworkPort()

```
word OficinaFramework::NetworkSystem::Socket::GetNetworkPort ( ) const
```

Gets the port binded by the socket.

12.31.3.3 Open()

```
bool OficinaFramework::NetworkSystem::Socket::Open (
            word port )
```

Opens a port for this socket.

Warning

Only open a port if you initialized the NetworkSystem itself! Not initializing may cause serious errors on Windows.

Parameters

port	Port to be binded.

Exceptions

SystemInitializationErrorException	

12.31.3.4 Receive()

```
bool OficinaFramework::NetworkSystem::Socket::Receive (
            Address & recv_sender,
            void * recv_data,
            size_t expected_size )
```

Receives a packet.

Parameters

recv_sender	Outputs address of the sender.

Warning

Address validations must be done out of the socket. It'll try to overwrite the given field.

Parameters

recv_data	Pointer to where the received data should be copied to.
expected_size	Expected size of the given packet.

12.31.3.5 Send()

```
bool OficinaFramework::NetworkSystem::Socket::Send (
            void * data,
            size_t datasize,
            Address destination )
```

Sends packet to a specific address.

Parameters

data	Pointer to the data stream to be sent.
datasize	Size of the data stream to be sent.
destination	Destination of the packet.

Warning

If a port number is given on this Address, it'll be ignored.

12.31.3.6 SetNetworkPort()

```
void OficinaFramework::NetworkSystem::Socket::SetNetworkPort (
            word port )
```

Sets the port for a socket, if not initialized.

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

- NetworkSystem.hpp

12.32 OficinaFramework::RenderingSystem::SpriteSheet Class Reference

Represents a Sprite Sheet, a texture containing frames used for animating objects such as characters.

```
#include <RenderingSystem.hpp>
```

Public Member Functions

- SpriteSheet (vec2dw FrameSize, vec2b PaddingThickness)

  Constructs a spritesheet.
- ~SpriteSheet ()

  Destructor to SpriteSheet class.
- SpriteSheet (vec2dw FrameSize, vec2b PaddingThickness, vec2 Hotspot)

  Constructs a spritesheet.
- void AppendTexture (Texture *t)

  Appends a texture to the spritesheet. Appended textures will act as if they were a single texture that extends the one that was previously added.
- void DrawFrame (vec2 Position, word frame, float magnification, Color4 tint, float transparency, float angle, RenderProperty rp=RENDER_NORMALLY, RenderEffect re=MODULATE_EFFECT)

  Draws a certain frame of the spritesheet.
- void DrawFrame (vec2 Position, word frame, float magnification, Color4 tint, float transparency, RenderProperty rp=RENDER_NORMALLY, RenderEffect re=MODULATE_EFFECT)

  Draws a certain frame of the spritesheet.
- void DrawFrame (vec2 Position, word frame, float magnification, RenderProperty rp=RENDER_NORMALLY, RenderEffect re=MODULATE_EFFECT)

  Draws a certain frame of the spritesheet.
- void DrawFrame (vec2 Position, word frame, RenderProperty rp=RENDER_NORMALLY, RenderEffect re=MODULATE_EFFECT)

  Draws a certain frame of the spritesheet.
- vec2dw GetFrameSize () const

  Gets the size of a single frame.
- vec2 GetHotspot () const

  Gets the hotspot of the frames.

12.32.1 Detailed Description

Represents a Sprite Sheet, a texture containing frames used for animating objects such as characters.

12.32.2 Constructor & Destructor Documentation

12.32.2.1 SpriteSheet() [1/2]

```
OficinaFramework::RenderingSystem::SpriteSheet::SpriteSheet (
            vec2dw FrameSize,
            vec2b PaddingThickness )
```

Constructs a spritesheet.

Warning

Do not dispose this texture on your own, only through the SpriteSheet.

Parameters

FrameSize	Size of each frame on the texture.
PaddingThickness	Thickness of the padding around each frame.

12.32.2.2 ~SpriteSheet()

```
OficinaFramework::RenderingSystem::SpriteSheet::~SpriteSheet ( )
```

Destructor to SpriteSheet class.

12.32.2.3 SpriteSheet() [2/2]

```
OficinaFramework::RenderingSystem::SpriteSheet::SpriteSheet (
            vec2dw FrameSize,
            vec2b PaddingThickness,
            vec2 Hotspot )
```

Constructs a spritesheet.

Warning

Do not dispose this texture on your own, only through the SpriteSheet.

Parameters

FrameSize	Size of each frame on the texture.
PaddingThickness	Thickness of the padding around each frame.
Hotspot	Central point of every frame on the animation.

12.32.3 Member Function Documentation

12.32.3.1 AppendTexture()

```
void OficinaFramework::RenderingSystem::SpriteSheet::AppendTexture (
            Texture * t )
```

Appends a texture to the spritesheet. Appended textures will act as if they were a single texture that extends the one that was previously added.

Warning

Appended textures will have the same framesize and hotspot of the whole spritesheet, and frame count is only related to texture width.

Parameters

t	Pointer to the texture to be appended.

12.32.3.2 DrawFrame() [1/4]

```
void OficinaFramework::RenderingSystem::SpriteSheet::DrawFrame (
            vec2 Position,
            word frame,
            float magnification,
            Color4 tint,
            float transparency,
            float angle,
            RenderProperty rp = RENDER_NORMALLY,
            RenderEffect re = MODULATE_EFFECT )
```

Draws a certain frame of the spritesheet.

Parameters

Position	Position of the frame's exact center.
frame	Index of the frame to be drawn.

Attention

To know a frame's index, consider all frames on a straight line. The first frame is frame 0.

Parameters

magnification	Magnification ratio of the frame to be drawn. Defaults to 1.0f.

Attention

Use of this parameter to simulate zoom is disencouraged.

Parameters

tint	Color to tint the sprite. Defaults to White (1.0f, 1.0f, 1.0f, 1.0f). The alpha factor will be used to measure intensity of tinting.
transparency	Transparency ratio of the frame. Must be a value between 0.0f and 1.0f, being 1.0f fully opaque.
angle	Angle of the frame.
rp	Flip of the frame. Use this to perform orientation changes.
re	Effect when rendering the texture. Defaults to MODULATE_EFFECT.

12.32.3.3 DrawFrame() [2/4]

```
void OficinaFramework::RenderingSystem::SpriteSheet::DrawFrame (
            vec2 Position,
            word frame,
            float magnification,
            Color4 tint,
            float transparency,
            RenderProperty rp = RENDER_NORMALLY,
            RenderEffect re = MODULATE_EFFECT )
```

Draws a certain frame of the spritesheet.

Parameters

Position	Position of the frame's exact center.
frame	Index of the frame to be drawn.

Attention

To know a frame's index, consider all frames on a straight line. The first frame is frame 0.

Parameters

magnification	Magnification ratio of the frame to be drawn. Defaults to 1.0f.

Attention

Use of this parameter to simulate zoom is disencouraged.

Parameters

tint	Color to tint the sprite. Defaults to White (1.0f, 1.0f, 1.0f, 1.0f). The alpha factor will be used to measure intensity of tinting.
transparency	Transparency ratio of the frame. Must be a value between 0.0f and 1.0f, being 1.0f fully opaque.
rp	Flip of the frame. Use this to perform orientation changes.
re	Effect when rendering the texture. Defaults to MODULATE_EFFECT.

12.32.3.4 DrawFrame() [3/4]

```
void OficinaFramework::RenderingSystem::SpriteSheet::DrawFrame (
            vec2 Position,
            word frame,
            float magnification,
            RenderProperty rp = RENDER_NORMALLY,
            RenderEffect re = MODULATE_EFFECT )
```

Draws a certain frame of the spritesheet.

Parameters

Position	Position of the frame's exact center.
frame	Index of the frame to be drawn.

Attention

To know a frame's index, consider all frames on a straight line. The first frame is frame 0..

Parameters

magnification	Magnification ratio of the frame to be drawn. Defaults to 1.0f.

Attention

Use of this parameter to simulate zoom is disencouraged.

Parameters

rp	Flip of the frame. Use this to perform orientation changes.
re	Effect when rendering the texture. Defaults to MODULATE_EFFECT.

12.32.3.5 DrawFrame() [4/4]

```
void OficinaFramework::RenderingSystem::SpriteSheet::DrawFrame (
            vec2 Position,
            word frame,
            RenderProperty rp = RENDER_NORMALLY,
            RenderEffect re = MODULATE_EFFECT )
```

Draws a certain frame of the spritesheet.

Parameters

Position	Position of the frame's exact center.
frame	Index of the frame to be drawn.

Attention

To know a frame's index, consider all frames on a straight line. The first frame is frame 0.

Parameters

rp	Flip of the frame. Use this to perform orientation changes.
re	Effect when rendering the texture. Defaults to MODULATE_EFFECT.

12.32.3.6 GetFrameSize()

```
vec2dw OficinaFramework::RenderingSystem::SpriteSheet::GetFrameSize ( ) const
```

Gets the size of a single frame.

Returns

The frame size.

12.32.3.7 GetHotspot()

```
vec2 OficinaFramework::RenderingSystem::SpriteSheet::GetHotspot ( ) const
```

Gets the hotspot of the frames.

Returns

Point to be considered the center for a frame.

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

- RenderingSystem.hpp

12.33 OficinaFramework::InputSystem::State Struct Reference

Represents a state for the input.

`#include <InputSystem.hpp>`

Public Member Functions

- State ()
  Constructor for the struct. Initializes all values to default.
- State & operator= (const State)
  Receives the values of another State and sets them for itself.
- State & operator| (const State)

Public Attributes

- bool Buttons [17]
  Represents the state of gamepad buttons.
- bool MouseButtons [3]
  Represents the state of mouse buttons.
- vec2dw MousePosition
  A dword vec2 representing mouse pointer position on window.
- vec2w LeftStick
  A word vec2 representing gamepad's left stick position.
- vec2w RightStick
  A word vec2 representing gamepad's right stick position.
- word LeftTrigger
- word RightTrigger
- byte KeyboardState [255]
  An array representing keyboard key states.

12.33.1 Detailed Description

Represents a state for the input.

12.33.2 Constructor & Destructor Documentation

12.33.2.1 State()

`OficinaFramework::InputSystem::State::State ( )`

Constructor for the struct. Initializes all values to default.

12.33.3 Member Function Documentation

12.33.3.1 operator=()

```
State& OficinaFramework::InputSystem::State::operator= (
            const State  )
```

Receives the values of another State and sets them for itself.

12.33.3.2 operator"|()

```
State& OficinaFramework::InputSystem::State::operator| (
            const State  )
```

Receives the values of another State and blends them with itself. If a button is not pressed in the current state, but it is in the other state, then the resulting state will contain the button as pressed.

12.33.4 Member Data Documentation

12.33.4.1 Buttons

```
bool OficinaFramework::InputSystem::State::Buttons[17]
```

Represents the state of gamepad buttons.

12.33.4.2 KeyboardState

```
byte OficinaFramework::InputSystem::State::KeyboardState[255]
```

An array representing keyboard key states.

12.33.4.3 LeftStick

```
vec2w OficinaFramework::InputSystem::State::LeftStick
```

A word vec2 representing gamepad's left stick position.

12.33.4.4 LeftTrigger

```
word OficinaFramework::InputSystem::State::LeftTrigger
```

A word representing gamepad's left trigger position. In case it is a button or it is a keyboard key, it'll always assume the values (word)ThumbStickAxisSignal::POSITIVE OR (word)ThumbStickAxisSignal::NEGATIVE.

See also

InputSystem::ThumbStickAxisSignal

12.33.4.5 MouseButtons

```
bool OficinaFramework::InputSystem::State::MouseButtons[3]
```

Represents the state of mouse buttons.

12.33.4.6 MousePosition

```
vec2dw OficinaFramework::InputSystem::State::MousePosition
```

A dword vec2 representing mouse pointer position on window.

12.33.4.7 RightStick

```
vec2w OficinaFramework::InputSystem::State::RightStick
```

A word vec2 representing gamepad's right stick position.

12.33.4.8 RightTrigger

```
word OficinaFramework::InputSystem::State::RightTrigger
```

A word representing gamepad's right trigger position. In case it is a button or it is a keyboard key, it'll always assume the values (word)ThumbStickAxisSignal::POSITIVE OR (word)ThumbStickAxisSignal::NEGATIVE.

See also

InputSystem::ThumbStickAxisSignal

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

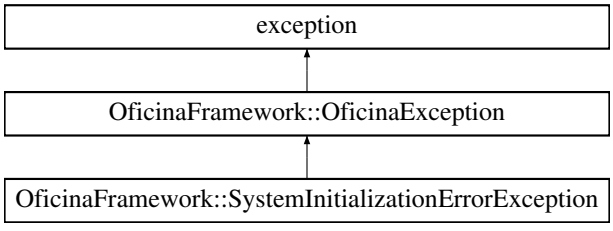
- InputSystem.hpp

12.34 OficinaFramework::SystemInitializationErrorException Class Reference

Exception for errors when initializing any system.

```
#include <OficinaExceptions.hpp>
```

Inheritance diagram for OficinaFramework::SystemInitializationErrorException:



Public Member Functions

- SystemInitializationErrorException ()

  Instantiates the exception with default error message.
- SystemInitializationErrorException (std::string message)

  Instantiates the exception with a given error message.
- SystemInitializationErrorException (std::string message, std::string systemname)

  Instantiates the exception with a given error message. Also identifies the System with errors.
- std::string GetSystemName ()

  A string containing the system name.
- std::string GetMessageAndSystemName ()

  A string containing the message and the system name.
- virtual const char * what () const throw ()

  Returns the message and troubling internal system name for the actual system exception display.

Additional Inherited Members

12.34.1 Detailed Description

Exception for errors when initializing any system.

12.34.2 Constructor & Destructor Documentation

12.34.2.1 SystemInitializationErrorException() [1/3]

```
OficinaFramework::SystemInitializationErrorException::SystemInitializationErrorException ( )
[inline]
```

Instantiates the exception with default error message.

12.34.2.2 SystemInitializationErrorException() [2/3]

```
OficinaFramework::SystemInitializationErrorException::SystemInitializationErrorException (
            std::string message )  [inline]
```

Instantiates the exception with a given error message.

Parameters

message	The message to be set to the exception.

12.34.2.3 SystemInitializationErrorException() [3/3]

```
OficinaFramework::SystemInitializationErrorException::SystemInitializationErrorException (
```

```
            std::string message,
            std::string systemname )  [inline]
```

Instantiates the exception with a given error message. Also identifies the System with errors.

Parameters

message	The message to be set to the exception.
systemname	The path or name of the problematic system.

12.34.3 Member Function Documentation

12.34.3.1 GetMessageAndSystemName()

```
std::string OficinaFramework::SystemInitializationErrorException::GetMessageAndSystemName ( )
```

A string containing the message and the system name.

Returns

Exception message, plus the problematic system name.

12.34.3.2 GetSystemName()

```
std::string OficinaFramework::SystemInitializationErrorException::GetSystemName ( )
```

A string containing the system name.

Returns

Name of the problematic system.

12.34.3.3 what()

```
virtual const char* OficinaFramework::SystemInitializationErrorException::what ( ) const throw
)   [inline], [virtual]
```

Returns the message and troubling internal system name for the actual system exception display.

Returns

See also

GetMessageAndSystemName

Reimplemented from OficinaFramework::OficinaException.

References OficinaFramework::OficinaException::message.

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

- OficinaExceptions.hpp

12.35 OficinaFramework::RenderingSystem::Texture Class Reference

Represents a texture. Use RenderingSystem::TexturePool to allocate a new Texture.

`#include <RenderingSystem.hpp>`

Public Member Functions

- vec2dw GetSize () const

  Size of the image loaded into the texture.
- GLuint GetName () const

  Name of the texture, as recognized by OpenGL.
- std::string GetPath () const

  Path to the image on disk or PATH.
- void Draw (vec2 Position, vec2 DestinationSize, vec2 SrcPosition, vec2 SrcSize, float angle, float transparency, vec2 Hotspot, Color4 tint, RenderProperty rp=RENDER_NORMALLY, RenderEffect re=MODULATE_EFFECT)

  Draws a rectangle of the texture on a destination rectangle.
- void Draw (vec2 Position, vec2 DestinationSize, vec2 SrcPosition, vec2 SrcSize, float angle, float transparency, Color4 tint, RenderProperty rp=RENDER_NORMALLY, RenderEffect re=MODULATE_EFFECT)

  Draws a rectangle of the texture on a destination rectangle.
- void Draw (vec2 Position, vec2 DestinationSize, float angle, float transparency, Color4 tint, RenderProperty rp=RENDER_NORMALLY, RenderEffect re=MODULATE_EFFECT)

  Draws the current texture on a destination rectangle.
- void Draw (vec2 Position, vec2 DestinationSize, float angle, Color4 tint, RenderProperty rp=RENDER_NORMALLY, RenderEffect re=MODULATE_EFFECT)

  Draws the current texture on a destination rectangle.
- void Draw (vec2 Position, vec2 DestinationSize, Color4 tint, RenderProperty rp=RENDER_NORMALLY, RenderEffect re=MODULATE_EFFECT)

  Draws the current texture on a destination rectangle.
- void Draw (vec2 Position, float transparency, Color4 tint, RenderProperty rp=RENDER_NORMALLY, RenderEffect re=MODULATE_EFFECT)

  Draws the current texture on a certain point.
- void Draw (vec2 Position, Color4 tint, float scale, RenderProperty rp=RENDER_NORMALLY, RenderEffect re=MODULATE_EFFECT)

  Draws the current texture on a certain point.
- void Draw (vec2 Position, Color4 tint, RenderProperty rp=RENDER_NORMALLY, RenderEffect re=MODULATE_EFFECT)

  Draws the current texture on a certain point.

Friends

- class RenderingSystem::TexturePool

12.35.1 Detailed Description

Represents a texture. Use RenderingSystem::TexturePool to allocate a new Texture.

12.35.2 Member Function Documentation

12.35.2.1 Draw() [1/8]

```
void OficinaFramework::RenderingSystem::Texture::Draw (
            vec2 Position,
            vec2 DestinationSize,
            vec2 SrcPosition,
            vec2 SrcSize,
            float angle,
            float transparency,
            vec2 Hotspot,
            Color4 tint,
            RenderProperty rp = RENDER_NORMALLY,
            RenderEffect re = MODULATE_EFFECT )
```

Draws a rectangle of the texture on a destination rectangle.

Parameters

Position	Top left vertex of the destination quad.
DestinationSize	Width and height of the destination quad.
SrcPosition	Top left vertex of the source quad.
SrcSize	Width and height of the source quad.
angle	Angle in degrees, relative to the top left of the image, in which it should be rotated.
transparency	Alpha rating for rendering the whole texture.
Hotspot	Point on the texture to be considered the center of the texture.
tint	Color to tint the sprite. Defaults to White (1.0f, 1.0f, 1.0f, 1.0f). The alpha factor will be used to measure intensity of tinting.
rp	Property regarding whether the texture will be flipped in some way or not. Defaults to RENDER_NORMALLY.
re	Effect when rendering the texture. Defaults to MODULATE_EFFECT.

12.35.2.2 Draw() [2/8]

```
void OficinaFramework::RenderingSystem::Texture::Draw (
            vec2 Position,
            vec2 DestinationSize,
            vec2 SrcPosition,
            vec2 SrcSize,
            float angle,
            float transparency,
            Color4 tint,
            RenderProperty rp = RENDER_NORMALLY,
            RenderEffect re = MODULATE_EFFECT )
```

Draws a rectangle of the texture on a destination rectangle.

Parameters

Position	Top left vertex of the destination quad.
DestinationSize	Width and height of the destination quad.
SrcPosition	Top left vertex of the source quad.
SrcSize	Width and height of the source quad.
angle	Angle in degrees, relative to the top left of the image, in which it should be rotated.
transparency	Alpha rating for rendering the whole texture.
tint	Color to tint the sprite. Defaults to White (1.0f, 1.0f, 1.0f, 1.0f). The alpha factor will be used to measure intensity of tinting.
rp	Property regarding whether the texture will be flipped in some way or not. Defaults to RENDER_NORMALLY.
re	Effect when rendering the texture. Defaults to MODULATE_EFFECT.

12.35.2.3 Draw() [3/8]

```
void OficinaFramework::RenderingSystem::Texture::Draw (
            vec2 Position,
            vec2 DestinationSize,
            float angle,
            float transparency,
            Color4 tint,
            RenderProperty rp = RENDER_NORMALLY,
            RenderEffect re = MODULATE_EFFECT )
```

Draws the current texture on a destination rectangle.

Parameters

Position	Top left vertex of the destination quad.
DestinationSize	Width and height of the destination quad.
angle	Angle in degrees, relative to the top left of the image, in which it should be rotated.
transparency	Alpha rating for rendering the whole texture.
tint	Color to tint the sprite. Defaults to White (1.0f, 1.0f, 1.0f, 1.0f). The alpha factor will be used to measure intensity of tinting.
rp	Property regarding whether the texture will be flipped in some way or not. Defaults to RENDER_NORMALLY.
re	Effect when rendering the texture. Defaults to MODULATE_EFFECT.

12.35.2.4 Draw() [4/8]

```
void OficinaFramework::RenderingSystem::Texture::Draw (
            vec2 Position,
            vec2 DestinationSize,
            float angle,
            Color4 tint,
```

```
            RenderProperty rp = RENDER_NORMALLY,
            RenderEffect re = MODULATE_EFFECT )
```

Draws the current texture on a destination rectangle.

Parameters

Position	Top left vertex of the destination quad.
DestinationSize	Width and height of the destination quad.
angle	Angle in degrees, relative to the top left of the image, in which it should be rotated.
tint	Color to tint the sprite. Defaults to White (1.0f, 1.0f, 1.0f, 1.0f). The alpha factor will be used to measure intensity of tinting.
rp	Property regarding whether the texture will be flipped in some way or not. Defaults to RENDER_NORMALLY.
re	Effect when rendering the texture. Defaults to MODULATE_EFFECT.

12.35.2.5 Draw() [5/8]

```
void OficinaFramework::RenderingSystem::Texture::Draw (
            vec2 Position,
            vec2 DestinationSize,
            Color4 tint,
            RenderProperty rp = RENDER_NORMALLY,
            RenderEffect re = MODULATE_EFFECT )
```

Draws the current texture on a destination rectangle.

Parameters

Position	Top left vertex of the destination quad.
DestinationSize	Width and height of the destination quad.
tint	Color to tint the sprite. Defaults to White (1.0f, 1.0f, 1.0f, 1.0f). The alpha factor will be used to measure intensity of tinting.
rp	Property regarding whether the texture will be flipped in some way or not. Defaults to RENDER_NORMALLY.
re	Effect when rendering the texture. Defaults to MODULATE_EFFECT.

12.35.2.6 Draw() [6/8]

```
void OficinaFramework::RenderingSystem::Texture::Draw (
            vec2 Position,
            float transparency,
            Color4 tint,
            RenderProperty rp = RENDER_NORMALLY,
            RenderEffect re = MODULATE_EFFECT )
```

Draws the current texture on a certain point.

Parameters

Position	Top left vertex of the texture to be rendered.
transparency	Alpha rating for rendering the whole texture.

Parameters

tint	Color to tint the sprite. Defaults to White (1.0f, 1.0f, 1.0f, 1.0f). The alpha factor will be used to measure intensity of tinting.
rp	Property regarding whether the texture will be flipped in some way or not. Defaults to RENDER_NORMALLY.
re	Effect when rendering the texture. Defaults to MODULATE_EFFECT.

12.35.2.7 Draw() [7/8]

```
void OficinaFramework::RenderingSystem::Texture::Draw (
            vec2 Position,
            Color4 tint,
            float scale,
            RenderProperty rp = RENDER_NORMALLY,
            RenderEffect re = MODULATE_EFFECT )
```

Draws the current texture on a certain point.

Parameters

Position	Top left vertex of the texture to be rendered.
tint	Color to tint the sprite. Defaults to White (1.0f, 1.0f, 1.0f, 1.0f). The alpha factor will be used to measure intensity of tinting.
scale	Magnification of the texture to be drawn. Defaults to 1x (1.0f).
rp	Property regarding whether the texture will be flipped in some way or not. Defaults to RENDER_NORMALLY.
re	Effect when rendering the texture. Defaults to MODULATE_EFFECT.

12.35.2.8 Draw() [8/8]

```
void OficinaFramework::RenderingSystem::Texture::Draw (
            vec2 Position,
            Color4 tint,
            RenderProperty rp = RENDER_NORMALLY,
            RenderEffect re = MODULATE_EFFECT )
```

Draws the current texture on a certain point.

Parameters

Position	Top left vertex of the texture to be rendered.
tint	Color to tint the sprite. Defaults to White (1.0f, 1.0f, 1.0f, 1.0f). The alpha factor will be used to measure intensity of tinting.
rp	Property regarding whether the texture will be flipped in some way or not. Defaults to RENDER_NORMALLY.
re	Effect when rendering the texture. Defaults to MODULATE_EFFECT.

12.35.2.9 GetName()

```
GLuint OficinaFramework::RenderingSystem::Texture::GetName ( ) const
```

Name of the texture, as recognized by OpenGL.

Returns

Texture identifier on the GPU.

12.35.2.10 GetPath()

```
std::string OficinaFramework::RenderingSystem::Texture::GetPath ( ) const
```

Path to the image on disk or PATH.

Returns

Actual path to the texture on PATH.

12.35.2.11 GetSize()

```
vec2dw OficinaFramework::RenderingSystem::Texture::GetSize ( ) const
```

Size of the image loaded into the texture.

Returns

A dword-precision vec2 containing the texture's width and height.

12.35.3 Friends And Related Function Documentation

12.35.3.1 RenderingSystem::TexturePool

```
friend class RenderingSystem::TexturePool  [friend]
```

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

- RenderingSystem.hpp

12.36 OficinaFramework::RenderingSystem::TexturePool Class Reference

Represents a structure that can manage the allocation and deallocation of textures.

`#include <RenderingSystem.hpp>`

Static Public Member Functions

- static Texture * LoadTexture (std::string asset_path)

  Allocates a texture.
- static Texture * LoadDefaultFontTexture ()

  Loads the texture from the default font type.
- static Font * LoadDefaultFont ()

  Loads the default font type.
- static void DisposeTexture (Texture *&t)

  Deallocates texture from memory.
- static void Clear ()

  Removes all textures from the pool. This will also render all Texture variables linked to this pool invalid.

12.36.1 Detailed Description

Represents a structure that can manage the allocation and deallocation of textures.

See also

Texture

12.36.2 Member Function Documentation

12.36.2.1 Clear()

```
static void OficinaFramework::RenderingSystem::TexturePool::Clear ( )  [static]
```

Removes all textures from the pool. This will also render all Texture variables linked to this pool invalid.

12.36.2.2 DisposeTexture()

```
static void OficinaFramework::RenderingSystem::TexturePool::DisposeTexture (
            Texture *& t )  [static]
```

Deallocates texture from memory.

Parameters

t	Texture to be deallocated.

12.36.2.3 LoadDefaultFont()

```
static Font* OficinaFramework::RenderingSystem::TexturePool::LoadDefaultFont ( )  [static]
```

Loads the default font type.

Warning

As it does not produce a pointer to the used texture, the texture itself will be added to the TexturePool and will only be disposed when the application exits. If you're in extreme need for memory, use this wisely.

Returns

A pointer to the font to be used.

12.36.2.4 LoadDefaultFontTexture()

```
static Texture* OficinaFramework::RenderingSystem::TexturePool::LoadDefaultFontTexture ( )
[static]
```

Loads the texture from the default font type.

Attention

Exact sizes for this font: each tile is 6x11; padding is 1 pixel on each margin.

Returns

A pointer to the texture containing the default font sheet.

12.36.2.5 LoadTexture()

```
static Texture* OficinaFramework::RenderingSystem::TexturePool::LoadTexture (
            std::string asset_path )  [static]
```

Allocates a texture.

Parameters

asset_path	Path to the asset; root is always "data" folder.

Returns

A reference to the texture.

Exceptions

InvalidAssetException	

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

- RenderingSystem.hpp

12.37 OficinaFramework::TimingSystem::TimeSpan Class Reference

Class designed to count a period of time.

`#include <TimingSystem.hpp>`

Public Member Functions

- TimeSpan ()

  Constructs the TimeSpan.
- void Start ()

  Starts counting the time.
- dword GetSpan () const

  Gets the time span.
- void End ()

  Ends the time span.

12.37.1 Detailed Description

Class designed to count a period of time.

12.37.2 Constructor & Destructor Documentation

12.37.2.1 TimeSpan()

`OficinaFramework::TimingSystem::TimeSpan::TimeSpan ( )`

Constructs the TimeSpan.

12.37.3 Member Function Documentation

12.37.3.1 End()

```
void OficinaFramework::TimingSystem::TimeSpan::End ( )
```

Ends the time span.

12.37.3.2 GetSpan()

```
dword OficinaFramework::TimingSystem::TimeSpan::GetSpan ( ) const
```

Gets the time span.

Returns

The time span, in miliseconds. If not started, returns 0.

12.37.3.3 Start()

```
void OficinaFramework::TimingSystem::TimeSpan::Start ( )
```

Starts counting the time.

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

- TimingSystem.hpp

12.38 OficinaFramework::TimingSystem Class Reference

Groups framerate and in-game time controls. Use this class for accurate movement according to framerate, as well as setting it to an unlimited, time-based framerate.

```
#include <TimingSystem.hpp>
```

Classes

- class TimeSpan
  Class designed to count a period of time.

Static Public Member Functions

- static void init ()
  Initializes the timing system with no framerate cap.
- static void init (double DefaultFrameRate)
  Initializes the timing system with a framerate cap.
- static void update ()
  Updates the timing sytem.
- static double GetDeltaTime ()
  Gets the delta time between this and the last frame.
- static double GetFPS ()
  Gets the framerate for the current frame.
- static float StepCorrection ()
  Calculates a correction factor for the speed of a step.

12.38.1 Detailed Description

Groups framerate and in-game time controls. Use this class for accurate movement according to framerate, as well as setting it to an unlimited, time-based framerate.

12.38.2 Member Function Documentation

12.38.2.1 GetDeltaTime()

```
static double OficinaFramework::TimingSystem::GetDeltaTime ( )  [static]
```

Gets the delta time between this and the last frame.

Returns

The delta time, in miliseconds, with double precision.

12.38.2.2 GetFPS()

```
static double OficinaFramework::TimingSystem::GetFPS ( )  [static]
```

Gets the framerate for the current frame.

Returns

The current framerate, with double precision.

12.38.2.3 init() [1/2]

```
static void OficinaFramework::TimingSystem::init ( )  [static]
```

Initializes the timing system with no framerate cap.

12.38.2.4 init() [2/2]

```
static void OficinaFramework::TimingSystem::init (
            double DefaultFrameRate )  [static]
```

Initializes the timing system with a framerate cap.

Parameters

DefaultFrameRate	The default frame rate to be used.

12.38.2.5 StepCorrection()

```
static float OficinaFramework::TimingSystem::StepCorrection ( )  [static]
```

Calculates a correction factor for the speed of a step.

Returns

The step correction factor.

12.38.2.6 update()

```
static void OficinaFramework::TimingSystem::update ( )  [static]
```

Updates the timing sytem.

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

- TimingSystem.hpp

12.39 vec2 Class Reference

A class representing a point in 2D space.

```
#include <OficinaTypes.hpp>
```

Public Member Functions

- vec2 ()
  Constructs class with null values.
- vec2 (float n)
  Constructs class with two equal values.
- vec2 (float X, float Y)
  Constructs class with two values.
- vec2 & operator= (const vec2)
- vec2 & operator= (const float)
- const vec2 operator+ (const vec2)
- const vec2 operator+ (const float)
- vec2 & operator+= (const vec2)
- vec2 & operator+= (const float)
- const vec2 operator- (const vec2)
- const vec2 operator- (const float)
- vec2 & operator-= (const vec2)
- vec2 & operator-= (const float)
- const vec2 operator∗ (const vec2)
- const vec2 operator∗ (const float)
- vec2 & operator∗= (const vec2)

- vec2 & operator*= (const float)
- const vec2 operator/ (const vec2)
- const vec2 operator/ (const float)
- vec2 & operator/= (const vec2)
- vec2 & operator/= (const float)
- bool operator> (const vec2)
- bool operator> (const float)
- bool operator< (const vec2)
- bool operator< (const float)
- bool operator>= (const vec2)
- bool operator>= (const float)
- bool operator<= (const vec2)
- bool operator<= (const float)
- bool operator== (const vec2)
- bool operator== (const float)
- bool operator!= (const vec2)
- bool operator!= (const float)
- vec2 getTruncated ()

  Returns a copy of this vec2 truncated to the nearest integer.
- void truncate ()

  Truncates the coordinates of this.
- float length ()

  Gets the magnitude (length) of this vector.
- vec2 getNormalized ()

  Gets a normalized version of this vector.
- void normalize ()

  Normalizes this vector.
- void clamp (vec2 origin, vec2 end)

  Encloses this vector inside a box, with its edges described by two other vectors.
- std::string toString ()

  Getsa string with the vec2 values.

Static Public Member Functions

- static vec2 Zero ()

  Returns a vec2 with both coordinates having the value 0.0f.
- static vec2 One ()

  Returns a vec2 with both coordinates having the value 1.0f.
- static vec2 Up ()

  Returns a vec2 with coordinates indicating the Up direction.
- static vec2 Down ()

  Returns a vec2 with coordinates indicating the Down direction.
- static vec2 Left ()

  Returns a vec2 with coordinates indicating the Left direction.
- static vec2 Right ()

  Returns a vec2 with coordinates indicating the Right direction.
- static float distance (vec2 first, vec2 second)

  Calculates the distance between two vectors.
- static float squareDistance (vec2 first, vec2 second)

  Calculates the square distance between two vectors. Useful if you don't want the Square Root operation to be executed on the value for performance reasons.

Public Attributes

- union {
  struct {
  float x
  X coordinate of point.
  float y
  Y coordinate of point.
  }
  struct {
  float v [2]
  Array of coordinates of the point.
  }
  };

Friends

- std::ostream & operator<< (std::ostream &oss, const vec2 &v)

12.39.1 Detailed Description

A class representing a point in 2D space.

12.39.2 Constructor & Destructor Documentation

12.39.2.1 vec2() [1/3]

```
vec2::vec2 ( )
```

Constructs class with null values.

12.39.2.2 vec2() [2/3]

```
vec2::vec2 (
            float n )
```

Constructs class with two equal values.

Parameters

n	Value to be given to coordinates.

12.39.2.3 vec2() [3/3]

```
vec2::vec2 (
            float X,
            float Y )
```

Constructs class with two values.

Parameters

X	Value to be given to x coordinate.
Y	Value to be given to y coordinate.

12.39.3 Member Function Documentation

12.39.3.1 clamp()

```
void vec2::clamp (
            vec2 origin,
            vec2 end )
```

Encloses this vector inside a box, with its edges described by two other vectors.

Parameters

origin	Minimum values for the vector.
end	Maximum values for the vector.

Warning

In case a single coordinate of origin is bigger than end's, the vector will not be clamped on that coordinate.

12.39.3.2 distance()

```
static float vec2::distance (
            vec2 first,
            vec2 second )  [static]
```

Calculates the distance between two vectors.

Parameters

first	First vector.
second	Second vector.

Returns

Distance value between the two desired points.

12.39.3.3 Down()

```
static vec2 vec2::Down ( )  [static]
```

Returns a vec2 with coordinates indicating the Down direction.

Returns

A down-facing vec2.

12.39.3.4 getNormalized()

```
vec2 vec2::getNormalized ( )
```

Gets a normalized version of this vector.

Returns

A normalized vector.

12.39.3.5 getTruncated()

```
vec2 vec2::getTruncated ( )
```

Returns a copy of this vec2 truncated to the nearest integer.

Returns

The truncated vec2 of this.

12.39.3.6 Left()

```
static vec2 vec2::Left ( )  [static]
```

Returns a vec2 with coordinates indicating the Left direction.

Returns

A left-facing vec2.

12.39.3.7 length()

```
float vec2::length ( )
```

Gets the magnitude (length) of this vector.

Returns

The magnitude of this vector.

12.39.3.8 normalize()

```
void vec2::normalize ( )
```

Normalizes this vector.

12.39.3.9 One()

```
static vec2 vec2::One ( )  [static]
```

Returns a vec2 with both coordinates having the value 1.0f.

Returns

An unitary-value vec2.

12.39.3.10 operator"!=() [1/2]

```
bool vec2::operator!= (
            const vec2  )
```

12.39.3.11 operator"!=() [2/2]

```
bool vec2::operator!= (
            const float  )
```

12.39.3.12 operator∗() [1/2]

```
const vec2 vec2::operator* (
            const vec2  )
```

12.39.3.13 operator*() [2/2]

```
const vec2 vec2::operator* (
            const float  )
```

12.39.3.14 operator*=() [1/2]

```
vec2& vec2::operator*= (
            const vec2  )
```

12.39.3.15 operator*=() [2/2]

```
vec2& vec2::operator*= (
            const float  )
```

12.39.3.16 operator+() [1/2]

```
const vec2 vec2::operator+ (
            const vec2  )
```

12.39.3.17 operator+() [2/2]

```
const vec2 vec2::operator+ (
            const float  )
```

12.39.3.18 operator+=() [1/2]

```
vec2& vec2::operator+= (
            const vec2  )
```

12.39.3.19 operator+=() [2/2]

```
vec2& vec2::operator+= (
            const float  )
```

12.39.3.20 operator-() [1/2]

```
const vec2 vec2::operator- (
            const vec2  )
```

12.39.3.21 operator-() [2/2]

```
const vec2 vec2::operator- (
            const float  )
```

12.39.3.22 operator-=() [1/2]

```
vec2& vec2::operator-= (
            const vec2  )
```

12.39.3.23 operator-=() [2/2]

```
vec2& vec2::operator-= (
            const float  )
```

12.39.3.24 operator/() [1/2]

```
const vec2 vec2::operator/ (
            const vec2  )
```

12.39.3.25 operator/() [2/2]

```
const vec2 vec2::operator/ (
            const float  )
```

12.39.3.26 operator/=() [1/2]

```
vec2& vec2::operator/= (
            const vec2  )
```

12.39.3.27 operator/=() [2/2]

```
vec2& vec2::operator/= (
            const float  )
```

12.39.3.28 operator<() [1/2]

```
bool vec2::operator< (
            const vec2  )
```

12.39.3.29 operator<() [2/2]

```
bool vec2::operator< (
            const float  )
```

12.39.3.30 operator<=() [1/2]

```
bool vec2::operator<= (
            const vec2  )
```

12.39.3.31 operator<=() [2/2]

```
bool vec2::operator<= (
            const float  )
```

12.39.3.32 operator=() [1/2]

```
vec2& vec2::operator= (
            const vec2  )
```

12.39.3.33 operator=() [2/2]

```
vec2& vec2::operator= (
            const float  )
```

12.39.3.34 operator==() [1/2]

```
bool vec2::operator== (
            const vec2  )
```

12.39.3.35 operator==() [2/2]

```
bool vec2::operator== (
            const float  )
```

12.39.3.36 operator>() [1/2]

```
bool vec2::operator> (
            const vec2  )
```

12.39.3.37 operator>() [2/2]

```
bool vec2::operator> (
            const float  )
```

12.39.3.38 operator>=() [1/2]

```
bool vec2::operator>= (
            const vec2  )
```

12.39.3.39 operator>=() [2/2]

```
bool vec2::operator>= (
            const float  )
```

12.39.3.40 Right()

```
static vec2 vec2::Right ( )  [static]
```

Returns a vec2 with coordinates indicating the Right direction.

Returns

A right-facing vec2.

12.39.3.41 squareDistance()

```
static float vec2::squareDistance (
            vec2 first,
            vec2 second )  [static]
```

Calculates the square distance between two vectors. Useful if you don't want the Square Root operation to be executed on the value for performance reasons.

Parameters

first	First vector.
second	Second vector.

Returns

Square distance value between the two desired points.

12.39.3.42 toString()

```
std::string vec2::toString ( )
```

Getsa string with the vec2 values.

Returns

Values inside vec2, in format {x, y}.

12.39.3.43 truncate()

```
void vec2::truncate ( )
```

Truncates the coordinates of this.

12.39.3.44 Up()

```
static vec2 vec2::Up ( )  [static]
```

Returns a vec2 with coordinates indicating the Up direction.

Returns

An up-facing vec2.

12.39.3.45 Zero()

```
static vec2 vec2::Zero ( )  [static]
```

Returns a vec2 with both coordinates having the value 0.0f.

Returns

A null-values vec2.

12.39.4 Friends And Related Function Documentation

12.39.4.1 operator<<

```
std::ostream& operator<< (
            std::ostream & oss,
            const vec2 & v )  [friend]
```

12.39.5 Member Data Documentation

12.39.5.1 "@1

```
union { ... }
```

12.39.5.2 v

```
float vec2::v[2]
```

Array of coordinates of the point.

12.39.5.3 x

```
float vec2::x
```

X coordinate of point.

12.39.5.4 y

```
float vec2::y
```

Y coordinate of point.

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

- OficinaTypes.hpp

12.40 vec2t< T > Class Template Reference

A class representing a point in 2D space, using a given data type. Only works with predefined types. See the vec2t Types module for more details.

```
#include <OficinaTypes.hpp>
```

Public Member Functions

- vec2t ()

  Constructs class with null values.
- vec2t (T n)

  Constructs class with two equal values.
- vec2t (T X, T Y)

  Constructs class with two values.
- vec2t< T > & operator= (const vec2t< T >)
- vec2t< T > & operator= (const T)
- const vec2t< T > operator+ (const vec2t< T >)
- const vec2t< T > operator+ (const T)
- vec2t< T > & operator+= (const vec2t< T >)
- vec2t< T > & operator+= (const T)
- const vec2t< T > operator- (const vec2t< T >)
- const vec2t< T > operator- (const T)
- vec2t< T > & operator-= (const vec2t< T >)
- vec2t< T > & operator-= (const T)
- const vec2t< T > operator∗ (const vec2t< T >)
- const vec2t< T > operator∗ (const T)
- vec2t< T > & operator∗= (const vec2t< T >)
- vec2t< T > & operator∗= (const T)
- const vec2t< T > operator/ (const vec2t< T >)
- const vec2t< T > operator/ (const T)
- vec2t< T > & operator/= (const vec2t< T >)
- vec2t< T > & operator/= (const T)
- bool operator> (const vec2t< T >)
- bool operator> (const T)
- bool operator< (const vec2t< T >)
- bool operator< (const T)
- bool operator>= (const vec2t< T >)
- bool operator>= (const T)
- bool operator<= (const vec2t< T >)
- bool operator<= (const T)
- bool operator== (const vec2t< T >)
- bool operator== (const T)
- bool operator!= (const vec2t< T >)
- bool operator!= (const T)
- vec2 toVec2 ()

  Converts this class to a vec2.

Static Public Member Functions

- static vec2t< T > Zero ()

  Returns a vec2t with both coordinates having the value 0.
- static vec2t< T > One ()

  Returns a vec2t with both coordinates having the value 1.
- static vec2t< T > Up ()

  Returns a vec2t with coordinates indicating the Up direction.
- static vec2t< T > Down ()

  Returns a vec2t with coordinates indicating the Down direction.
- static vec2t< T > Left ()

  Returns a vec2t with coordinates indicating the Left direction.
- static vec2t< T > Right ()

  Returns a vec2t with coordinates indicating the Right direction.

Public Attributes

- union {
  struct {
  T x
  X coordinate of point.
  T y
  Y coordinate of point.
  }
  struct {
  T v [2]
  Array of coordinates of the point.
  }
  };

12.40.1 Detailed Description

template<typename T>
class vec2t< T >

A class representing a point in 2D space, using a given data type. Only works with predefined types. See the vec2t Types module for more details.

See also

vec2t Types

12.40.2 Constructor & Destructor Documentation

12.40.2.1 vec2t() [1/3]

```
template<typename T>
vec2t< T >::vec2t ( )
```

Constructs class with null values.

12.40.2.2 vec2t() [2/3]

```
template<typename T>
vec2t< T >::vec2t (
            T n )
```

Constructs class with two equal values.

Parameters

n	Value to be given to coordinates.

12.40.2.3 vec2t() [3/3]

```
template<typename T>
vec2t< T >::vec2t (
            T X,
            T Y )
```

Constructs class with two values.

Parameters

X	Value to be given to x coordinate.
Y	Value to be given to y coordinate.

12.40.3 Member Function Documentation

12.40.3.1 Down()

```
template<typename T>
static vec2t<T> vec2t< T >::Down ( )  [static]
```

Returns a vec2t with coordinates indicating the Down direction.

Returns

A down-facing vec2t.

12.40.3.2 Left()

```
template<typename T>
static vec2t<T> vec2t< T >::Left ( )  [static]
```

Returns a vec2t with coordinates indicating the Left direction.

Returns

A left-facing vec2t.

12.40.3.3 One()

```
template<typename T>
static vec2t<T> vec2t< T >::One ( )  [static]
```

Returns a vec2t with both coordinates having the value 1.

Returns

An unitary-value vec2t.

12.40.3.4 operator"!=() [1/2]

```
template<typename T>
bool vec2t< T >::operator!= (
            const vec2t< T >  )
```

12.40.3.5 operator"!=() [2/2]

```
template<typename T>
bool vec2t< T >::operator!= (
            const T  )
```

12.40.3.6 operator*() [1/2]

```
template<typename T>
const vec2t<T> vec2t< T >::operator* (
            const vec2t< T >  )
```

12.40.3.7 operator*() [2/2]

```
template<typename T>
const vec2t<T> vec2t< T >::operator* (
            const T  )
```

12.40.3.8 operator*=() [1/2]

```
template<typename T>
vec2t<T>& vec2t< T >::operator*= (
            const vec2t< T >  )
```

12.40.3.9 operator*=() [2/2]

```
template<typename T>
vec2t<T>& vec2t< T >::operator*= (
            const T  )
```

12.40.3.10 operator+() [1/2]

```
template<typename T>
const vec2t<T> vec2t< T >::operator+ (
            const vec2t< T >  )
```

12.40.3.11 operator+() [2/2]

```
template<typename T>
const vec2t<T> vec2t< T >::operator+ (
            const T  )
```

12.40.3.12 operator+=() [1/2]

```
template<typename T>
vec2t<T>& vec2t< T >::operator+= (
            const vec2t< T >  )
```

12.40.3.13 operator+=() [2/2]

```
template<typename T>
vec2t<T>& vec2t< T >::operator+= (
            const T  )
```

12.40.3.14 operator-() [1/2]

```
template<typename T>
const vec2t<T> vec2t< T >::operator- (
            const vec2t< T >  )
```

12.40.3.15 operator-() [2/2]

```
template<typename T>
const vec2t<T> vec2t< T >::operator- (
            const T  )
```

12.40.3.16 operator-=() [1/2]

```
template<typename T>
vec2t<T>& vec2t< T >::operator-= (
            const vec2t< T >  )
```

12.40.3.17 operator-=() [2/2]

```
template<typename T>
vec2t<T>& vec2t< T >::operator-= (
            const T  )
```

12.40.3.18 operator/() [1/2]

```
template<typename T>
const vec2t<T> vec2t< T >::operator/ (
            const vec2t< T >  )
```

12.40.3.19 operator/() [2/2]

```
template<typename T>
const vec2t<T> vec2t< T >::operator/ (
            const T  )
```

12.40.3.20 operator/=() [1/2]

```
template<typename T>
vec2t<T>& vec2t< T >::operator/= (
            const vec2t< T >  )
```

12.40.3.21 operator/=() [2/2]

```
template<typename T>
vec2t<T>& vec2t< T >::operator/= (
            const T  )
```

12.40.3.22 operator<() [1/2]

```
template<typename T>
bool vec2t< T >::operator< (
            const vec2t< T >  )
```

12.40.3.23 operator<() [2/2]

```
template<typename T>
bool vec2t< T >::operator< (
            const T  )
```

12.40.3.24 operator<=() [1/2]

```
template<typename T>
bool vec2t< T >::operator<= (
            const vec2t< T >  )
```

12.40.3.25 operator<=() [2/2]

```
template<typename T>
bool vec2t< T >::operator<= (
            const T  )
```

12.40.3.26 operator=() [1/2]

```
template<typename T>
vec2t<T>& vec2t< T >::operator= (
            const vec2t< T >  )
```

12.40.3.27 operator=() [2/2]

```
template<typename T>
vec2t<T>& vec2t< T >::operator= (
            const T  )
```

12.40.3.28 operator==() [1/2]

```
template<typename T>
bool vec2t< T >::operator== (
            const vec2t< T >  )
```

12.40.3.29 operator==() [2/2]

```
template<typename T>
bool vec2t< T >::operator== (
            const T  )
```

12.40.3.30 operator>() [1/2]

```
template<typename T>
bool vec2t< T >::operator> (
             const vec2t< T >  )
```

12.40.3.31 operator>() [2/2]

```
template<typename T>
bool vec2t< T >::operator> (
             const T  )
```

12.40.3.32 operator>=() [1/2]

```
template<typename T>
bool vec2t< T >::operator>= (
             const vec2t< T >  )
```

12.40.3.33 operator>=() [2/2]

```
template<typename T>
bool vec2t< T >::operator>= (
             const T  )
```

12.40.3.34 Right()

```
template<typename T>
static vec2t<T> vec2t< T >::Right ( )  [static]
```

Returns a vec2t with coordinates indicating the Right direction.

Returns

A right-facing vec2t.

12.40.3.35 toVec2()

```
template<typename T>
vec2 vec2t< T >::toVec2 ( )
```

Converts this class to a vec2.

Returns

A vec2 equivalent of this class.

12.40.3.36 Up()

```
template<typename T>
static vec2t<T> vec2t< T >::Up ( )  [static]
```

Returns a vec2t with coordinates indicating the Up direction.

Returns

An up-facing vec2t.

12.40.3.37 Zero()

```
template<typename T>
static vec2t<T> vec2t< T >::Zero ( )  [static]
```

Returns a vec2t with both coordinates having the value 0.

Returns

A null-values vec2t.

12.40.4 Member Data Documentation

12.40.4.1 "@7

```
union { ... }
```

12.40.4.2 v

```
template<typename T>
T vec2t< T >::v[2]
```

Array of coordinates of the point.

12.40.4.3 x

```
template<typename T>
T vec2t< T >::x
```

X coordinate of point.

12.40.4.4 y

```
template<typename T>
T vec2t< T >::y
```

Y coordinate of point.

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

- OficinaTypes.hpp

12.41 vec3 Class Reference

A class representing a point in 3D space.

```
#include <OficinaTypes.hpp>
```

Public Member Functions

- vec3 ()
  Constructs class with null values.
- vec3 (float n)
  Constructs class with two equal values.
- vec3 (float X, float Y, float Z)
  Constructs class with two values.
- vec3 & operator= (const vec3)
- vec3 & operator= (const float)
- const vec3 operator+ (const vec3)
- const vec3 operator+ (const float)
- vec3 & operator+= (const vec3)
- vec3 & operator+= (const float)
- const vec3 operator- (const vec3)
- const vec3 operator- (const float)
- vec3 & operator-= (const vec3)
- vec3 & operator-= (const float)
- const vec3 operator∗ (const vec3)
- const vec3 operator∗ (const float)
- vec3 & operator∗= (const vec3)
- vec3 & operator∗= (const float)
- const vec3 operator/ (const vec3)
- const vec3 operator/ (const float)
- vec3 & operator/= (const vec3)
- vec3 & operator/= (const float)
- bool operator> (const vec3)
- bool operator> (const float)
- bool operator< (const vec3)
- bool operator< (const float)
- bool operator>= (const vec3)
- bool operator>= (const float)
- bool operator<= (const vec3)
- bool operator<= (const float)
- bool operator== (const vec3)

- bool operator== (const float)
- bool operator!= (const vec3)
- bool operator!= (const float)
- vec3 getTruncated ()

  Returns a copy of this vec3 truncated to the nearest integer.
- void truncate ()

  Truncates the coordinates of this.
- float length ()

  Gets the magnitude (length) of this vector.
- vec3 getNormalized ()

  Gets a normalized version of this vector.
- void normalize ()

  Normalizes this vector.
- void clamp (vec3 origin, vec3 end)

  Encloses this vector inside a box, with its edges described by two other vectors.
- std::string toString ()

  Getsa string with the vec3 values.

Static Public Member Functions

- static vec3 Zero ()

  Returns a vec3 with both coordinates having the value 0.0f.
- static vec3 One ()

  Returns a vec3 with both coordinates having the value 1.0f.
- static vec3 Up ()

  Returns a vec3 with coordinates indicating the Up direction.
- static vec3 Down ()

  Returns a vec3 with coordinates indicating the Down direction.
- static vec3 Left ()

  Returns a vec3 with coordinates indicating the Left direction.
- static vec3 Right ()

  Returns a vec3 with coordinates indicating the Right direction.
- static vec3 Front ()

  Returns a vec3 with coordinates indicating the Front direction.
- static vec3 Back ()

  Returns a vec3 with coordinates indicating the Back direction.
- static float distance (vec3 first, vec3 second)

  Calculates the distance between two vectors.
- static float squareDistance (vec3 first, vec3 second)

  Calculates the square distance between two vectors. Useful if you don't want the Square Root operation to be executed on the value for performance reasons.

Public Attributes

- union {
  struct {
  float x
  X coordinate of point.
  float y
  Y coordinate of point.
  float z
  Z coordinate of point.
  }
  struct {
  float v [3]
  Array of coordinates of the point.
  }
  };

Friends

- std::ostream & operator<< (std::ostream &oss, const vec3 &v)

12.41.1 Detailed Description

A class representing a point in 3D space.

12.41.2 Constructor & Destructor Documentation

12.41.2.1 vec3() [1/3]

```
vec3::vec3 ( )
```

Constructs class with null values.

12.41.2.2 vec3() [2/3]

```
vec3::vec3 (
            float n )
```

Constructs class with two equal values.

Parameters

n	Value to be given to coordinates.

12.41.2.3 vec3() [3/3]

```
vec3::vec3 (
            float X,
            float Y,
            float Z )
```

Constructs class with two values.

Parameters

X	Value to be given to x coordinate.
Y	Value to be given to y coordinate.
Z	Value to be given to y coordinate.

12.41.3 Member Function Documentation

12.41.3.1 Back()

```
static vec3 vec3::Back ( )  [static]
```

Returns a vec3 with coordinates indicating the Back direction.

Returns

A back-facing vec3.

12.41.3.2 clamp()

```
void vec3::clamp (
            vec3 origin,
            vec3 end )
```

Encloses this vector inside a box, with its edges described by two other vectors.

Parameters

origin	Minimum values for the vector.
end	Maximum values for the vector.

Warning

In case a single coordinate of origin is bigger than end's, the vector will not be clamped on that coordinate.

12.41.3.3 distance()

```
static float vec3::distance (
            vec3 first,
            vec3 second )  [static]
```

Calculates the distance between two vectors.

Parameters

first	First vector.
second	Second vector.

Returns

Distance value between the two desired points.

12.41.3.4 Down()

```
static vec3 vec3::Down ( )  [static]
```

Returns a vec3 with coordinates indicating the Down direction.

Returns

A down-facing vec3.

12.41.3.5 Front()

```
static vec3 vec3::Front ( )  [static]
```

Returns a vec3 with coordinates indicating the Front direction.

Returns

A front-facing vec3.

12.41.3.6 getNormalized()

```
vec3 vec3::getNormalized ( )
```

Gets a normalized version of this vector.

Returns

A normalized vector.

12.41.3.7 getTruncated()

```
vec3 vec3::getTruncated ( )
```

Returns a copy of this vec3 truncated to the nearest integer.

Returns

The truncated vec3 of this.

12.41.3.8 Left()

```
static vec3 vec3::Left ( )  [static]
```

Returns a vec3 with coordinates indicating the Left direction.

Returns

A left-facing vec3.

12.41.3.9 length()

```
float vec3::length ( )
```

Gets the magnitude (length) of this vector.

Returns

The magnitude of this vector.

12.41.3.10 normalize()

```
void vec3::normalize ( )
```

Normalizes this vector.

12.41.3.11 One()

```
static vec3 vec3::One ( )  [static]
```

Returns a vec3 with both coordinates having the value 1.0f.

Returns

An unitary-value vec3.

12.41.3.12 operator"!=() [1/2]

```
bool vec3::operator!= (
            const vec3  )
```

12.41.3.13 operator"!=() [2/2]

```
bool vec3::operator!= (
            const float  )
```

12.41.3.14 operator*() [1/2]

```
const vec3 vec3::operator* (
            const vec3  )
```

12.41.3.15 operator*() [2/2]

```
const vec3 vec3::operator* (
            const float  )
```

12.41.3.16 operator*=() [1/2]

```
vec3& vec3::operator*= (
            const vec3  )
```

12.41.3.17 operator*=() [2/2]

```
vec3& vec3::operator*= (
            const float  )
```

12.41.3.18 operator+() [1/2]

```
const vec3 vec3::operator+ (
            const vec3  )
```

12.41.3.19 operator+() [2/2]

```
const vec3 vec3::operator+ (
            const float  )
```

12.41.3.20 operator+=() [1/2]

```
vec3& vec3::operator+= (
            const vec3  )
```

12.41.3.21 operator+=() [2/2]

```
vec3& vec3::operator+= (
            const float  )
```

12.41.3.22 operator-() [1/2]

```
const vec3 vec3::operator- (
            const vec3  )
```

12.41.3.23 operator-() [2/2]

```
const vec3 vec3::operator- (
            const float  )
```

12.41.3.24 operator-=() [1/2]

```
vec3& vec3::operator-= (
            const vec3  )
```

12.41.3.25 operator-=() [2/2]

```
vec3& vec3::operator-= (
            const float  )
```

12.41.3.26 operator/() [1/2]

```
const vec3 vec3::operator/ (
            const vec3  )
```

12.41.3.27 operator/() [2/2]

```
const vec3 vec3::operator/ (
            const float  )
```

12.41.3.28 operator/=() [1/2]

```
vec3& vec3::operator/= (
            const vec3  )
```

12.41.3.29 operator/=() [2/2]

```
vec3& vec3::operator/= (
            const float  )
```

12.41.3.30 operator<() [1/2]

```
bool vec3::operator< (
            const vec3  )
```

12.41.3.31 operator<() [2/2]

```
bool vec3::operator< (
            const float  )
```

12.41.3.32 operator<=() [1/2]

```
bool vec3::operator<= (
            const vec3  )
```

12.41.3.33 operator<=() [2/2]

```
bool vec3::operator<= (
            const float  )
```

12.41.3.34 operator=() [1/2]

```
vec3& vec3::operator= (
            const vec3  )
```

12.41.3.35 operator=() [2/2]

```
vec3& vec3::operator= (
            const float  )
```

12.41.3.36 operator==() [1/2]

```
bool vec3::operator== (
            const vec3  )
```

12.41.3.37 operator==() [2/2]

```
bool vec3::operator== (
            const float  )
```

12.41.3.38 operator>() [1/2]

```
bool vec3::operator> (
            const vec3  )
```

12.41.3.39 operator>() [2/2]

```
bool vec3::operator> (
            const float  )
```

12.41.3.40 operator>=() [1/2]

```
bool vec3::operator>= (
            const vec3  )
```

12.41.3.41 operator>=() [2/2]

```
bool vec3::operator>= (
            const float  )
```

12.41.3.42 Right()

```
static vec3 vec3::Right ( )  [static]
```

Returns a vec3 with coordinates indicating the Right direction.

Returns

A right-facing vec3.

12.41.3.43 squareDistance()

```
static float vec3::squareDistance (
            vec3 first,
            vec3 second )  [static]
```

Calculates the square distance between two vectors. Useful if you don't want the Square Root operation to be executed on the value for performance reasons.

Parameters

first	First vector.
second	Second vector.

Returns

Square distance value between the two desired points.

12.41.3.44 toString()

```
std::string vec3::toString ( )
```

Getsa string with the vec3 values.

Returns

Values inside vec3, in format {x, y}.

12.41.3.45 truncate()

```
void vec3::truncate ( )
```

Truncates the coordinates of this.

12.41.3.46 Up()

```
static vec3 vec3::Up ( )  [static]
```

Returns a vec3 with coordinates indicating the Up direction.

Returns

An up-facing vec3.

12.41.3.47 Zero()

```
static vec3 vec3::Zero ( )  [static]
```

Returns a vec3 with both coordinates having the value 0.0f.

Returns

A null-values vec3.

12.41.4 Friends And Related Function Documentation

12.41.4.1 operator<<

```
std::ostream& operator<< (
            std::ostream & oss,
            const vec3 & v )  [friend]
```

12.41.5 Member Data Documentation

12.41.5.1 "@13

```
union { ... }
```

12.41.5.2 v

```
float vec3::v[3]
```

Array of coordinates of the point.

12.41.5.3 x

```
float vec3::x
```

X coordinate of point.

12.41.5.4 y

```
float vec3::y
```

Y coordinate of point.

12.41.5.5 z

```
float vec3::z
```

Z coordinate of point.

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

- OficinaTypes.hpp

13 File Documentation

13.1 AudioSystem.hpp File Reference

```
#include <OficinaFramework/OficinaTypes.hpp>
#include <AL/al.h>
#include <AL/alc.h>
#include <map>
#include <stack>
#include <vector>
```

Classes

- class OficinaFramework::AudioSystem

  Groups audio-related management controls. Use this to play background music, sfx, and effects.
- class OficinaFramework::AudioSystem::Audio

  Represents an audio file to be loaded. Cannot be created nor destroyed on its own.
- class OficinaFramework::AudioSystem::AudioSource

  Describes the source of the audio, so effects such as positional sound can be used.
- class OficinaFramework::AudioSystem::AudioPool

  A class for loading audio, both sound effects or background music.

Namespaces

- OficinaFramework

13.2 DiagnosticsSystem.hpp File Reference

```
#include <OficinaFramework/OficinaTypes.hpp>
```

Classes

- class OficinaFramework::DiagnosticsSystem

  Controls for monitoring Memory and CPU usage.

Namespaces

- OficinaFramework

13.3 EngineCore.hpp File Reference

```
#include <OficinaFramework/OficinaTypes.hpp>
#include <string>
#include <list>
#include <SDL2/SDL.h>
#include <SDL2/SDL_thread.h>
```

Classes

- class OficinaFramework::EngineCore

  The main core of the engine, which handles initialization and game loop automatically.

Namespaces

- OficinaFramework

13.4 EntitySystem.hpp File Reference

```
#include <OficinaFramework/OficinaTypes.hpp>
#include <OficinaFramework/RenderingSystem.hpp>
#include <string>
#include <vector>
#include <stack>
```

Classes

- class OficinaFramework::EntitySystem

  Class including common controls for creating entities and entity collections.
- class OficinaFramework::EntitySystem::Entity

  Abstract class representing an entity.
- class OficinaFramework::EntitySystem::DrawableEntity

  Abstract class representing an entity that can be drawn onscreen.
- class OficinaFramework::EntitySystem::EntityCollection

  A collection of Entities to be used on a screen.
- class OficinaFramework::EntitySystem::DrawableEntityCollection

  A collection of DrawableEntities to be used on a screen.
- class OficinaFramework::EntitySystem::IBuilder

  An interface for creating an Entity Builder, specially if it is supposed to be loaded from a script.

Namespaces

- OficinaFramework

13.5 gameargs.dox File Reference

13.6 InputSystem.hpp File Reference

```
#include <OficinaFramework/OficinaTypes.hpp>
#include <SDL2/SDL.h>
#include <map>
```

Classes

- class OficinaFramework::InputSystem

  Groups all input-related methods and objects. Has built-in support for keyboard, multiple gamepads and mouse.
- struct OficinaFramework::InputSystem::State

  Represents a state for the input.

Namespaces

- OficinaFramework

13.7 IOSystem.hpp File Reference

```
#include <OficinaFramework/OficinaTypes.hpp>
#include <SDL2/SDL.h>
#include <physfs.h>
```

Classes

- class OficinaFramework::IOSystem

  Provides methods for loading compressed data.
- class OficinaFramework::IOSystem::ScriptStream

  Reads a script as a byte stream.
- class OficinaFramework::IOSystem::ScriptTools

  A class for opening and loading Gongly Script data. Works since Gongly Script v1.0.

Namespaces

- OficinaFramework

13.8 main_page.dox File Reference

13.9 NetworkSystem.hpp File Reference

```
#include <OficinaFramework/OficinaTypes.hpp>
```

Classes

- class OficinaFramework::NetworkSystem

  Manages all data sending and receiving over network.
- struct OficinaFramework::NetworkSystem::Address

  A struct representing an IPv4 address.
- class OficinaFramework::NetworkSystem::Socket

  A class representing a socket, used to control ports for comunication with other computers around the web.

Namespaces

- OficinaFramework

13.10 OficinaExceptions.hpp File Reference

```
#include <exception>
#include <string>
```

Classes

- class OficinaFramework::OficinaException

  Base class for all framework exceptions.
- class OficinaFramework::InvalidAssetException

  Exception for asset importing errors.
- class OficinaFramework::SystemInitializationErrorException

  Exception for errors when initializing any system.

Namespaces

- OficinaFramework

13.11 OficinaFramework.hpp File Reference

```
#include <OficinaFramework/InputSystem.hpp>
#include <OficinaFramework/DiagnosticsSystem.hpp>
#include <OficinaFramework/RenderingSystem.hpp>
#include <OficinaFramework/TimingSystem.hpp>
#include <OficinaFramework/NetworkSystem.hpp>
#include <OficinaFramework/ScreenSystem.hpp>
#include <OficinaFramework/AudioSystem.hpp>
#include <OficinaFramework/EntitySystem.hpp>
#include <OficinaFramework/IOSystem.hpp>
#include <OficinaFramework/EngineCore.hpp>
```

Macros

- #define OFICINA_FRAMEWORK_VERSION "v1.3 beta"

  Gives information about the version of OficinaFramework.

13.11.1 Macro Definition Documentation

13.11.1.1 OFICINA_FRAMEWORK_VERSION

```
#define OFICINA_FRAMEWORK_VERSION "v1.3 beta"
```

Gives information about the version of OficinaFramework.

13.12 OficinaTypes.hpp File Reference

```
#include <cstdint>
#include <cstdio>
#include <iostream>
#include <cstdarg>
```

Classes

- struct Color4

  A struct representing a color.
- class vec2

  A class representing a point in 2D space.
- class vec2t< T >

  A class representing a point in 2D space, using a given data type. Only works with predefined types. See the vec2t Types module for more details.
- class vec3

  A class representing a point in 3D space.

Macros

- #define M_PI 3.14159265358979323846
- #define M_TAU 6.28318530717958647692
- #define M_TWO_PI M_TAU
- #define M_THREEQUARTERS_PI 2.35619449019234492883
- #define M_HALF_PI 1.57079632679489661923
- #define M_QUARTER_PI 0.78539816339744830961
- #define OF_PLATFORM_UNKNOWN
- #define OF_ARCHITECTURE_UNKNOWN
- #define ARCHITECTURE "unknown"
- #define KNRM "\x1B[0m"

  Normal ASCII Terminal color.
- #define KRED "\x1B[31m"

  Red ASCII Terminal color.
- #define KGRN "\x1B[32m"

  Green ASCII Terminal color.
- #define KYEL "\x1B[33m"

  Yellow ASCII Terminal color.
- #define KBLU "\x1B[34m"

  Blue ASCII Terminal color.
- #define KMAG "\x1B[35m"

  Magenta ASCII Terminal color.
- #define KCYN "\x1B[36m"

Cyan ASCII Terminal color.

- #define KWHT "\x1B[37m"

  White ASCII Terminal color.
- #define KRESET "\033[0m"

  Resets any ASCII terminal color.
- #define CLEARMASK_NOT 0x000Fu

  Mask to be binded to the opposite of another color mask for byte extraction.
- #define HIGHLIGHTMASK 0x0777u

  Mask to be used to form highlight mode.
- #define NEXTCOLOR(x) (x = (x << 4))

  Shifts the current byte to the left, so another component is added.
- #define MIXCOLOR(x, y) (x = (x | y))

  Sets the last byte of the mask to the byte component value to be set.
- #define SHADOWMODE(x) (x >> 1)

  Sets a color to its shadowed mode.
- #define HIGHLIGHTMODE(x) ((x >> 1) + HIGHLIGHTMASK)

  Sets a color to its highlighted mode.
- #define GETRCOLOR(x) ((x ^ ~(~x | (CLEARMASK_NOT << 8))) >> 8)

  Gets red component of a color.
- #define GETGCOLOR(x) ((x ^ ~(~x | (CLEARMASK_NOT << 4))) >> 4)

  Gets green component of a color.
- #define GETBCOLOR(x) (x ^ ~(~x | CLEARMASK_NOT))

  Gets blue component of a color.
- #define GETACOLOR(x) ((x ^ ~(~x | (CLEARMASK_NOT << 12))) >> 12)

  Gets alpha component of a color.
- #define MASKTOBYTE(x) (x * 8)

  Transforms a recently extracted color component into a byte to be used by renderer.

Typedefs

- typedef uint8_t byte

  8-bit unsigned type.
- typedef uint16_t word

  16-bit unsigned type.
- typedef uint32_t dword

  32-bit unsigned type.
- typedef uint64_t qword

  64-bit unsigned type.
- typedef int8_t byte_s

  8-bit signed type.
- typedef int16_t word_s

  16-bit signed type.
- typedef int32_t dword_s

  32-bit signed type.
- typedef int64_t qword_s

  64-bit signed type.
- typedef word ColorM

  The definition of a color mask. Defined by byte format 0xARGB, being:
- typedef vec2t< double > vec2d

  A class representing a point in 2D space, with double precision.

- typedef vec2t< byte > vec2b

  A class representing a point in 2D space, with byte precision.
- typedef vec2t< word > vec2w

  A class representing a point in 2D space, with word precision.
- typedef vec2t< dword > vec2dw

  A class representing a point in 2D space, with double word precision.
- typedef vec2t< qword > vec2qw

  A class representing a point in 2D space, with quad word precision.
- typedef vec2t< byte_s > vec2sb

  A class representing a point in 2D space, with signed byte precision.
- typedef vec2t< word_s > vec2sw

  A class representing a point in 2D space, with signed word precision.
- typedef vec2t< dword_s > vec2sdw

  A class representing a point in 2D space, with signed double word precision.
- typedef vec2t< qword_s > vec2sqw

  A class representing a point in 2D space, with signed quad word precision.

Enumerations

- enum ColorDef {
  TRANSP = 0x0000u, BLACK = 0xF000u, RED = 0xFF00u, GREEN = 0xF0F0u,
  BLUE = 0xF00Fu, YELLOW = 0xFFF0u, CYAN = 0xF0FFu, MAGENTA = 0xFF0Fu,
  GREY10 = 0xFCCCu, GREY25 = 0xFAAAu, GREY50 = 0xF888u, GREY75 = 0xF666u,
  GREY80 = 0xF444u, GREY90 = 0xF222u, WHITE = 0xFFFFu, CORNFLOWERBLUE = 0xF68Eu,
  DARKBLUE = 0xF008u, ORANGE = 0xFFA0u, ORANGERED = 0xFF40u, BLUEVIOLET = 0xF82Eu }

  Color mask premade definitions.
- enum OF_LogLevel {
  OF_LOG_LVL_CRITICAL = 0, OF_LOG_LVL_ERROR = 1, OF_LOG_LVL_WARNING = 2, OF_LOG_LV↩
  L_INFO = 3,
  OF_LOG_LVL_NONE = 4 }

  Debug log levels for Oficina.

Functions

- float MASKTOFLOAT (ColorM c)

  Transforms a recently extracted color component into a float to be used by renderer.
- float degtorad (float angle)

  Converts an angle from degrees to radians.
- float radtodeg (float angle)

  Converts an angle from radians to degrees.
- float absolute (float val)

  Gives back the absolute value of another value.
- void clamp (float &value, float min, float max)

  Clamps a value to minimum and maximum values.
- int OF_Log (OF_LogLevel level, const char *fmt,...)

  Log function (printf-like) for Oficina's Debug target. This will only print onscreen if DEBUG_ENABLED is defined, or under a critical error.
- void OF_SetMinimalLogLevel (OF_LogLevel level)

  Sets the minimum log level of Oficina output.

13.12.1 Macro Definition Documentation

13.12.1.1 ARCHITECTURE

```
#define ARCHITECTURE "unknown"
```

13.12.1.2 M_HALF_PI

```
#define M_HALF_PI 1.57079632679489661923
```

13.12.1.3 M_PI

```
#define M_PI 3.14159265358979323846
```

13.12.1.4 M_QUARTER_PI

```
#define M_QUARTER_PI 0.78539816339744830961
```

13.12.1.5 M_TAU

```
#define M_TAU 6.28318530717958647692
```

13.12.1.6 M_THREEQUARTERS_PI

```
#define M_THREEQUARTERS_PI 2.35619449019234492883
```

13.12.1.7 M_TWO_PI

```
#define M_TWO_PI M_TAU
```

13.12.1.8 OF_ARCHITECTURE_UNKNOWN

```
#define OF_ARCHITECTURE_UNKNOWN
```

13.12.1.9 OF_PLATFORM_UNKNOWN

```
#define OF_PLATFORM_UNKNOWN
```

13.12.2 Typedef Documentation

13.12.2.1 ColorM

```
typedef word ColorM
```

The definition of a color mask. Defined by byte format 0xARGB, being:

- A: Alpha
- R: Red
- G: Green
- B: Blue

13.12.3 Enumeration Type Documentation

13.12.3.1 ColorDef

```
enum ColorDef
```

Color mask premade definitions.

Enumerator

TRANSP	Transparent color.
BLACK	Black color.
RED	Red color.
GREEN	Green color.
BLUE	Blue color.
YELLOW	Yellow color.
CYAN	Cyan color.
MAGENTA	Magenta color.
GREY10	Grey 10% color.
GREY25	Grey 25% color.
GREY50	Grey 50% color.
GREY75	Grey 75% color.
GREY80	Grey 80% color.
GREY90	Grey 90% color.
WHITE	White color.
CORNFLOWERBLUE	Cornflower Blue color.
DARKBLUE	Dark Blue color.
ORANGE	Orange color.
ORANGERED	Orange Red color.
BLUEVIOLET	Blue Violet color.

13.12.3.2 OF_LogLevel

```
enum OF_LogLevel
```

Debug log levels for Oficina.

Enumerator

OF_LOG_LVL_CRITICAL	Critical output. Usually used on critical errors that WILL affect performance. **Warning** Critical output can be seen on console regardless of debug mode or not, unlike the others.
OF_LOG_LVL_ERROR	Error output. Used in case of errors that will not affect performance too much.
OF_LOG_LVL_WARNING	Warning output. Used in case of problems that will not affect performance, but may cause issues.
OF_LOG_LVL_INFO	Information output. Used when outputting information that only has monitoring purposes.
OF_LOG_LVL_NONE	Out-of-scope output. Will be printed regardless on the console, but only under Debug mode.

13.12.4 Function Documentation

13.12.4.1 OF_Log()

```
int OF_Log (
            OF_LogLevel level,
            const char * fmt,
             ...  )
```

Log function (printf-like) for Oficina's Debug target. This will only print onscreen if DEBUG_ENABLED is defined, or under a critical error.

Warning

Will not print anything and also return -1 if the log level is greated than the current level.

Parameters

level	Level of the current output.
fmt	Format for the output.
...	Complementary values for the format output.

Returns

Whether log was printed or not, or if the printing operation had a problem.

13.12.4.2 OF_SetMinimalLogLevel()

```
void OF_SetMinimalLogLevel (
            OF_LogLevel level )
```

Sets the minimum log level of Oficina output.

Parameters

level	New minimal log level to be given.

Warning

This will inhibit any logs under the current log level, except if OF_LOG_LVL_NONE was chosen.

13.13 RenderingSystem.hpp File Reference

```
#include <SDL2/SDL.h>
#include <GL/glew.h>
#include <SDL2/SDL_opengl.h>
#include <OficinaFramework/OficinaTypes.hpp>
#include <map>
#include <vector>
```

Classes

- class OficinaFramework::RenderingSystem

  Groups rendering-related controls. Use this to allocate and deallocate textures accelerated by GPU, and also for drawing textures or primitives onscreen.
- class OficinaFramework::RenderingSystem::Texture

  Represents a texture. Use RenderingSystem::TexturePool to allocate a new Texture.
- class OficinaFramework::RenderingSystem::Font

  Represents a Font, a texture with monospace characters to be used to draw text onscreen.
- class OficinaFramework::RenderingSystem::SpriteSheet

  Represents a Sprite Sheet, a texture containing frames used for animating objects such as characters.
- class OficinaFramework::RenderingSystem::Animation

  Represents an Animation, a set of controls for animating objects using SpriteSheets.
- struct OficinaFramework::RenderingSystem::Animation::AnimationSpecs

  A struct representing the specs of a single animation.
- class OficinaFramework::RenderingSystem::TexturePool

  Represents a structure that can manage the allocation and deallocation of textures.
- class OficinaFramework::RenderingSystem::IRendererObject

  Interface for GPU-related objects.
- class OficinaFramework::RenderingSystem::RenderBuffer
- class OficinaFramework::RenderingSystem::FrameBuffer

  Describes a Frame Buffer object.

Namespaces

- OficinaFramework

13.14 ScreenSystem.hpp File Reference

```
#include <OficinaFramework/OficinaTypes.hpp>
#include <OficinaFramework/RenderingSystem.hpp>
#include <SDL2/SDL.h>
#include <SDL2/SDL_opengl.h>
#include <SDL2/SDL_image.h>
#include <list>
#include <queue>
#include <stack>
#include <sstream>
```

Classes

- class OficinaFramework::ScreenSystem

  Groups screen management controls. Use this class to add/remove screens and set them active or inactive.
- class OficinaFramework::ScreenSystem::Screen

  A class representing a Screen to be rendered on the screen manager.

Namespaces

- OficinaFramework

13.15 TimingSystem.hpp File Reference

```
#include <OficinaFramework/OficinaTypes.hpp>
```

Classes

- class OficinaFramework::TimingSystem

  Groups framerate and in-game time controls. Use this class for accurate movement according to framerate, as well as setting it to an unlimited, time-based framerate.
- class OficinaFramework::TimingSystem::TimeSpan

  Class designed to count a period of time.

Namespaces

- OficinaFramework

Index