
nephtys Documentation

Release 0.0.1

Sztergbaum Roman

May 01, 2019

CONTENTS

1	api reference manual	1
2	Indices and tables	5

API REFERENCE MANUAL

1.1 nephtys::client::config

struct config

This is the game configuration.

Public Functions

bool **operator==** (const *config* &*rhs_config*) const

Return true if the configurations are equal, false otherwise.

Parameters

- *rhs_config*: the config that you want to compare.

bool **operator!=** (const *config* &*rhs_config*) const

Return true if the configurations are different, false otherwise.

Parameters

- *rhs_config*: the config that you want to compare.

Public Members

window::win_cfg **window**

window data information

Below is an example of a valid configuration for the nephtys client:

```
{
  "window": {
    "size": {
      "height": 1200,
      "width": 800
    },
    "title": "nephtys",
    "fullscreen": false
  }
}
```

See also *nephtys::utils*, *nephtys::window::win_cfg*.

1.2 nephtys::window::win_cfg

struct win_cfg

The configuration of the game window contains all the data necessary for its manipulation: **size**, **name**, **full screen** [...].

Public Functions

bool **operator==**(const win_cfg &rhs_win) const

Return true if the configuration of the windows of the game are identical, false otherwise.

Parameters

- rhs_win: the window configuration that you want to compare.

bool **operator!=**(const win_cfg &rhs_win) const

Return true if the configuration of the windows of the game are different, false otherwise.

Parameters

- rhs_win: the window configuration that you want to compare

Public Members

st::height **height** = {1200}

the height of the game window

st::width **width** = {800}

the width of the game window

std::string **title** = {"nephtys client"}

the title of the game window

bool **is_fullscreen** = {false}

is the window full screen?

1.3 nephtys::utils

template<typename TConfig>

TConfig nephtys::utils::load_configuration(std::filesystem::path &&config_path, std::string filename)

This function allows us to load a configuration through a path and filename. There are three different behaviors in this function:

- if the parameter path does not exist the function will attempt to create the directories of the given path.
- if the configuration does not exist a default one will be **created**.
- if the path and the name of the file exists, the contents of the configuration will be **loaded**.

Example:

```
auto cfg = utils::load_configuration<client::config>(std::filesystem::current_
↳ path() / "assets/config", "nephtys_client.config.json");
```

Return a loaded/created configuration.

Template Parameters

- `TConfig`: the type of template you want to load

Parameters

- `config_path`: the path to the configuration you want to load
- `filename`: the name of the configuration you want to load.

INDICES AND TABLES

- search

N

nephtys::client::config (C++ *class*), 1
 nephtys::client::config::operator!=
 (C++ *function*), 1
 nephtys::client::config::operator==
 (C++ *function*), 1
 nephtys::client::config::window (C++
 member), 1
 nephtys::utils::load_configuration (C++
 function), 2
 nephtys::window::win_cfg (C++ *class*), 2
 nephtys::window::win_cfg::height (C++
 member), 2
 nephtys::window::win_cfg::is_fullscreen
 (C++ *member*), 2
 nephtys::window::win_cfg::operator!=
 (C++ *function*), 2
 nephtys::window::win_cfg::operator==
 (C++ *function*), 2
 nephtys::window::win_cfg::title (C++
 member), 2
 nephtys::window::win_cfg::width (C++
 member), 2