SC LBM

Generated by Doxygen 1.8.13

Contents

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 3

Class Documentation

3.1 Geometry Class Reference

Public Member Functions

- Geometry (const size_t Nx, const size_t Ny)

  Creates a geometry object with nodes $N_x \times N_y$ nodes.
- Geometry (const Geometry &gm)

  Copy constructor.
- Geometry & operator= (const Geometry &rhs)

  Copy assignment operator.
- Geometry (Geometry &&gm) noexcept

  Move constructor.
- Geometry && operator= (Geometry &&rhs)

  Move assignment operator.
- void add_walls (const size_t dH=1, const std::string where="y")

  Create a segment of solid walls.
- void **add_rectangle** (const std::vector< size_t >)
- void **add_ellipse** (const std::vector< size_t >)
- void **add_array** (const std::vector< size_t >, const std::vector< size_t >, const std::string, const size_t alpha=0)
- const bool **operator()** (const size_t i, const size_t j) const
- void **write** (const std::string) const
- void **read** (const std::string)
- const bool * get_geom () const

  Get pointer to constant geom object (can't change it!)
- size_t Nx () const

  Get number of nodes in x (horizontal) direction.
- size_t Ny () const

  Get number of nodes in y (vertical) direction.
- ~Geometry ()

  Destructor.

3.1.1 Constructor & Destructor Documentation

3.1.1.1 Geometry() [1/3]

```
Geometry::Geometry (
            const size_t Nx,
            const size_t Ny )  [inline]
```

Creates a geometry object with nodes $N_x \times N_y$ nodes.

Underlying array is initialized to 0

Parameters

Nx	- number of nodes in x direction
Ny	- number of nodes in y direction

3.1.1.2 Geometry() [2/3]

```
Geometry::Geometry (
            const Geometry & gm )
```

Copy constructor.

Creates a new Geometry object through a deep copy of another Geometry object

Parameters

gm	[in] - Geometry object to be copy from

3.1.1.3 Geometry() [3/3]

```
Geometry::Geometry (
            Geometry && gm )  [noexcept]
```

Move constructor.

Creates a new Geometry object by direct moving of another Geometry object

Parameters

gm	[in] - object to move from

3.1.2 Member Function Documentation

3.1.2.1 add_walls()

```
void Geometry::add_walls (
            const size_t dH = 1,
            const std::string where = "y" )
```

Create a segment of solid walls.

Parameters

dH	[in] - wall thickness,
where	[in] - wall direction

3.1.2.2 operator=() [1/2]

```
Geometry & Geometry::operator= (
            const Geometry & rhs )
```

Copy assignment operator.

Copies the content of a Geometry object to another geometry object. Objects must have the same dimensions.

Parameters

rhs	[in] - Geometry object to be copied/assigned from

3.1.2.3 operator=() [2/2]

```
Geometry&& Geometry::operator= (
            Geometry && rhs )
```

Move assignment operator.

Moves content of Geometry object rhs to another geometry object. Objects must have the same dimensions.

Parameters

rhs	[in] - object to move from

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

- include/geometry.h
- src/geometry.cpp

Chapter 4

File Documentation

4.1 tests/common/test_utils.h File Reference

```
#include <typeinfo>
#include "../../include/common.h"
```

Functions

- void tst_pass (const bool val, const std::string msg)

  Print if the test passed or failed.
- template<typename R , typename... Args>
  bool exception_test (bool verbose, const std::exception *expected, R(*function)(Args...), Args... args)

  Wrapper for testing if a function correctly raises exceptions.

Variables

- const auto print_msg

  Print message in specified color.
- const auto equal_doubles

  Equality comparison with doubles.

4.1.1 Function Documentation

4.1.1.1 exception_test()

```
template<typename R , typename...  Args>
bool exception_test (
            bool verbose,
            const std::exception * expected,
            R(*)(Args...)  function,
            Args...  args )
```

Wrapper for testing if a function correctly raises exceptions.

Executes given functions and returns true if the target exception happened and false if a different exception occurred. If nullptr specified instead of exception object, no exception will return true and exception will return false. All exceptions are handled.

Parameters

verbose	[in] - prints exception messages
expected	[in] - expected exception type or nullptr for no exception
function	[in] - function to be called by the wrapper
args	[in] - arguments to that function as a parameter pack

Returns

True or fals if test passed/failed under given conditions

4.1.1.2 tst_pass()

```
void tst_pass (
            const bool val,
            const std::string msg )
```

Print if the test passed or failed.

Hardcoded color scheme, depending on val it will print a pass/fail notification following the test name

Parameters

val	[in] - test outcome
tname	[in] - test name

4.1.2 Variable Documentation

4.1.2.1 equal_doubles

```
const auto equal_doubles
```

Initial value:

```
= [](const double v1, const double v2, const double eps=1e-5)
                            { return !(std::abs((v1 - v2)/v1) > eps); }
```

Equality comparison with doubles.

Parameters

v1	[in]
v2	[in] - two doubles to check for equality
eps	[in] - relative tolerance, default 1e-5

4.1.2.2 print_msg

```
const auto print_msg
```

Initial value:

```
= [](const std::string msg)
                            { std::cout << msg << std::endl; }
```

Print message in specified color.

Parameters

in	*msg*	- content to print

Index