Gigabrot_C++

Generated by Doxygen 1.9.2

1 Gigabrot 1

1.1 About The Project . . . 1

1.1.1 Note: . . . 2

1.1.1.1 Built With: . . . 2

1.2 Getting Started . . . 2

1.2.1 Prerequisites . . . 2

1.2.2 Cloning . . . 2

1.3 Usage . . . 2

1.4 Contact . . . 3

1.4.1 Demonstrated C++ Fundamentals . . . 3

1.4.1.1 For Reference . . . 3

2 Hierarchical Index 5

2.1 Class Hierarchy . . . 5

3 Class Index 7

3.1 Class List . . . 7

4 File Index 9

4.1 File List . . . 9

5 Class Documentation 11

5.1 Colorization Class Reference . . . 11

5.1.1 Detailed Description . . . 12

5.1.2 Constructor & Destructor Documentation . . . 12

5.1.2.1 ∼Colorization() . . . 13

5.1.2.2 Colorization() . . . 13

5.1.3 Member Function Documentation . . . 13

5.1.3.1 get_max_color_value() . . . 13

5.1.3.2 get_min_color_value() . . . 13

5.1.3.3 get_type() . . . 13

5.1.4 Member Data Documentation . . . 14

5.1.4.1 maxColorValue . . . 14

5.1.4.2 minColorValue . . . 14

5.1.4.3 type . . . 14

5.2 InsideColor Class Reference . . . 15

5.2.1 Detailed Description . . . 17

5.2.2 Constructor & Destructor Documentation . . . 17

5.2.2.1 InsideColor() . . . 17

5.2.2.2 ∼InsideColor() . . . 17

5.2.3 Member Function Documentation . . . 17

5.2.3.1 calculate_b() . . . 17

5.2.3.2 calculate_bw() . . . 18

5.2.3.3 calculate_g() . . . 18

5.2.3.4 calculate_r() 18
5.3 LineColor Class Reference 19
5.3.1 Detailed Description 21
5.3.2 Constructor & Destructor Documentation 21
5.3.2.1 LineColor() 21
5.3.2.2 ∼LineColor() 21
5.3.3 Member Function Documentation 21
5.3.3.1 calculate_b() 21
5.3.3.2 calculate_bw() 22
5.3.3.3 calculate_g() 22
5.3.3.4 calculate_r() 22
5.4 Mandelbrot Class Reference 23
5.4.1 Detailed Description 25
5.4.2 Constructor & Destructor Documentation 25
5.4.2.1 Mandelbrot() [1/3] 25
5.4.2.2 Mandelbrot() [2/3] 25
5.4.2.3 Mandelbrot() [3/3] 26
5.4.2.4 ∼Mandelbrot() 26
5.4.3 Member Function Documentation 27
5.4.3.1 average() 27
5.4.3.2 colorize_bw() 28
5.4.3.3 current_pixel() 28
5.4.3.4 describe_border() 29
5.4.3.5 get_c() 30
5.4.3.6 get_t() 30
5.4.3.7 in_border() 31
5.4.3.8 in_set() 31
5.4.3.9 interpolate() 32
5.4.3.10 iterate() 32
5.4.3.11 reset() 33
5.4.3.12 set_border() 34
5.4.3.13 set_image() 34
5.4.3.14 set_iSkip() 34
5.4.3.15 set_plane() 34
5.4.3.16 set_stripe_density() 35
5.4.3.17 shape_check() 35
5.4.4 Friends And Related Function Documentation 35
5.4.4.1 operator<< 35
5.4.5 Member Data Documentation 36
5.4.5.1 a 36
5.4.5.2 bulb 36
5.4.5.3 c 36

5.4.5.4 cardioid . . . 36
5.4.5.5 cxMax . . . 36
5.4.5.6 cxMin . . . 37
5.4.5.7 cyMax . . . 37
5.4.5.8 cyMin . . . 37
5.4.5.9 d . . . 37
5.4.5.10 dC . . . 37
5.4.5.11 de . . . 37
5.4.5.12 escapeRadius . . . 38
5.4.5.13 height . . . 38
5.4.5.14 iSkip . . . 38
5.4.5.15 iter . . . 38
5.4.5.16 iterMax . . . 38
5.4.5.17 pixHeight . . . 38
5.4.5.18 pixWidth . . . 39
5.4.5.19 prevA . . . 39
5.4.5.20 pX . . . 39
5.4.5.21 pY . . . 39
5.4.5.22 q . . . 39
5.4.5.23 r . . . 39
5.4.5.24 shade . . . 40
5.4.5.25 stripeDensity . . . 40
5.4.5.26 thin . . . 40
5.4.5.27 width . . . 40
5.4.5.28 z . . . 40
5.5 Neumorphic Class Reference . . . 41
5.5.1 Detailed Description . . . 43
5.5.2 Constructor & Destructor Documentation . . . 43
5.5.2.1 Neumorphic() . . . 43
5.5.3 Member Function Documentation . . . 43
5.5.3.1 calculate() . . . 43
5.5.3.2 get_angle() . . . 44
5.5.3.3 get_heightFactor() . . . 44
5.5.3.4 get_reflection() . . . 45
5.5.4 Member Data Documentation . . . 45
5.5.4.1 angle . . . 45
5.5.4.2 dC . . . 45
5.5.4.3 heightFactor . . . 45
5.5.4.4 reflection . . . 45
5.5.4.5 u . . . 46
5.5.4.6 v . . . 46
5.5.4.7 z . . . 46

5.6 NormalMap Class Reference 46
5.6.1 Detailed Description 48
5.6.2 Constructor & Destructor Documentation 48
5.6.2.1 ∼NormalMap() 49
5.6.2.2 NormalMap() 49
5.6.3 Member Function Documentation 49
5.6.3.1 calculate() 49
5.6.3.2 dot_product() 49
5.6.3.3 get_max_val() 50
5.6.3.4 get_min_val() 50
5.6.3.5 get_type() 50
5.6.4 Member Data Documentation 50
5.6.4.1 maxMapVal 50
5.6.4.2 minMapVal 51
5.6.4.3 type 51
5.7 PPM Class Reference 51
5.7.1 Detailed Description 53
5.7.2 Constructor & Destructor Documentation 53
5.7.2.1 PPM() `[1/3]` 53
5.7.2.2 PPM() `[2/3]` 53
5.7.2.3 PPM() `[3/3]` 54
5.7.3 Member Function Documentation 54
5.7.3.1 close() 54
5.7.3.2 init_stream() 55
5.7.3.3 operator=() 55
5.7.3.4 set_comment() 56
5.7.3.5 set_height() 56
5.7.3.6 set_outputDirectory() 56
5.7.3.7 set_width() 57
5.7.3.8 write_header() 57
5.7.3.9 write_row() 58
5.7.4 Member Data Documentation 58
5.7.4.1 comment 58
5.7.4.2 fileName 58
5.7.4.3 header 59
5.7.4.4 height 59
5.7.4.5 image 59
5.7.4.6 magic 59
5.7.4.7 outputDirectory 59
5.7.4.8 pixMaxVal 59
5.7.4.9 subPixel 60
5.7.4.10 width 60

5.8 Shading Class Reference 60
5.8.1 Detailed Description 62
5.8.2 Constructor & Destructor Documentation 62
5.8.2.1 Shading() 62
5.8.2.2 ∼Shading() 62
5.8.3 Member Function Documentation 62
5.8.3.1 calculate_b() 62
5.8.3.2 calculate_bw() 62
5.8.3.3 calculate_g() 63
5.8.3.4 calculate_r() 63
5.8.3.5 get_max_color_value() 63
5.8.3.6 get_min_color_value() 63
5.8.4 Member Data Documentation 63
5.8.4.1 type 63
5.9 Striping Class Reference 64
5.9.1 Detailed Description 66
5.9.2 Constructor & Destructor Documentation 66
5.9.2.1 Striping() 66
5.9.2.2 ∼Striping() 66
5.9.3 Member Function Documentation 66
5.9.3.1 calculate_b() 66
5.9.3.2 calculate_bw() 67
5.9.3.3 calculate_g() 67
5.9.3.4 calculate_r() 67
5.9.3.5 get_average() 68
5.9.4 Member Data Documentation 68
5.9.4.1 average 68
5.9.4.2 reflection 68
5.10 ThreadPool Class Reference 68
5.10.1 Detailed Description 70
5.10.2 Member Typedef Documentation 70
5.10.2.1 process 70
5.10.2.2 queue 70
5.10.2.3 queueVec 70
5.10.2.4 Threads 70
5.10.3 Constructor & Destructor Documentation 70
5.10.3.1 ThreadPool() 71
5.10.3.2 ∼ThreadPool() 71
5.10.4 Member Function Documentation 72
5.10.4.1 enqueue_task() 72
5.10.4.2 enqueue_work() 72
5.10.5 Member Data Documentation 73

5.10.5.1 count . . . 73
5.10.5.2 countMult . . . 73
5.10.5.3 index . . . 73
5.10.5.4 queues . . . 73
5.10.5.5 threads . . . 74
5.11 UnboundedQueue< T > Class Template Reference . . . 74
5.11.1 Detailed Description . . . 75
5.11.2 Member Typedef Documentation . . . 75
5.11.2.1 queue_t . . . 75
5.11.3 Constructor & Destructor Documentation . . . 76
5.11.3.1 UnboundedQueue() . . . 76
5.11.3.2 ∼UnboundedQueue() . . . 76
5.11.4 Member Function Documentation . . . 76
5.11.4.1 block() . . . 76
5.11.4.2 blocking() . . . 77
5.11.4.3 emplace() . . . 77
5.11.4.4 empty() . . . 77
5.11.4.5 pop() . . . 78
5.11.4.6 push() [1/2] . . . 78
5.11.4.7 push() [2/2] . . . 78
5.11.4.8 size() . . . 79
5.11.4.9 try_pop() . . . 79
5.11.4.10 try_push() [1/2] . . . 79
5.11.4.11 try_push() [2/2] . . . 80
5.11.4.12 unblock() . . . 80
5.11.5 Member Data Documentation . . . 80
5.11.5.1 condition . . . 80
5.11.5.2 is_block . . . 81
5.11.5.3 queue . . . 81
5.11.5.4 queueLock . . . 81

6 File Documentation 83

6.1 D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/CMakeFiles/3.21.1/CompilerIdC/←
CMakeCCompilerId.c File Reference . . . 83
6.1.1 Macro Definition Documentation . . . 83
6.1.1.1 __has_include . . . 84
6.1.1.2 ARCHITECTURE_ID . . . 84
6.1.1.3 C_DIALECT . . . 84
6.1.1.4 COMPILER_ID . . . 84
6.1.1.5 DEC . . . 84
6.1.1.6 HEX . . . 85
6.1.1.7 PLATFORM_ID . . . 85
6.1.1.8 STRINGIFY . . . 85

6.1.1.9 STRINGIFY_HELPER 85
6.1.2 Function Documentation 85
6.1.2.1 main() 86
6.1.3 Variable Documentation 86
6.1.3.1 info_arch 86
6.1.3.2 info_compiler 86
6.1.3.3 info_language_dialect_default 86
6.1.3.4 info_platform 87
6.2 CMakeCCompilerId.c 87
6.3 D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/CMakeFiles/3.21.1/CompilerId↩ CXX/CMakeCXXCompilerId.cpp File Reference 96
6.3.1 Macro Definition Documentation 97
6.3.1.1 __has_include 97
6.3.1.2 ARCHITECTURE_ID 97
6.3.1.3 COMPILER_ID 97
6.3.1.4 CXX_STD 97
6.3.1.5 DEC 98
6.3.1.6 HEX 98
6.3.1.7 PLATFORM_ID 98
6.3.1.8 STRINGIFY 98
6.3.1.9 STRINGIFY_HELPER 99
6.3.2 Function Documentation 99
6.3.2.1 main() 99
6.3.3 Variable Documentation 99
6.3.3.1 info_arch 99
6.3.3.2 info_compiler 100
6.3.3.3 info_language_dialect_default 100
6.3.3.4 info_platform 100
6.4 CMakeCXXCompilerId.cpp 100
6.5 D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/CMakeFiles/FindOpenMP/Open↩ MPCheckVersion.c File Reference 109
6.5.1 Function Documentation 110
6.5.1.1 main() 110
6.5.2 Variable Documentation 110
6.5.2.1 ompver_str 111
6.6 OpenMPCheckVersion.c 111
6.7 D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/CMakeFiles/FindOpenMP/Open↩ MPCheckVersion.cpp File Reference 111
6.7.1 Function Documentation 112
6.7.1.1 main() 112
6.7.2 Variable Documentation 112
6.7.2.1 ompver_str 112
6.8 OpenMPCheckVersion.cpp 112

6.9 D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/CMakeFiles/FindOpenMP/Open↩ MPTryFlag.c File Reference 113
6.9.1 Function Documentation 113
6.9.1.1 main() 113
6.10 OpenMPTryFlag.c 113
6.11 D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/CMakeFiles/FindOpenMP/Open↩ MPTryFlag.cpp File Reference 114
6.11.1 Function Documentation 114
6.11.1.1 main() 114
6.12 OpenMPTryFlag.cpp 114
6.13 D:/Documents/Final/Mandelbrot_Cpp_Final/README.md File Reference 115
6.14 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Colorization.cpp File Reference 115
6.15 Colorization.cpp 115
6.16 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Colorization.h File Reference 116
6.17 Colorization.h 116
6.18 D:/Documents/Final/Mandelbrot_Cpp_Final/src/InsideColor.cpp File Reference 117
6.19 InsideColor.cpp 117
6.20 D:/Documents/Final/Mandelbrot_Cpp_Final/src/InsideColor.h File Reference 118
6.21 InsideColor.h 119
6.22 D:/Documents/Final/Mandelbrot_Cpp_Final/src/LineColor.cpp File Reference 120
6.23 LineColor.cpp 120
6.24 D:/Documents/Final/Mandelbrot_Cpp_Final/src/LineColor.h File Reference 121
6.25 LineColor.h 122
6.26 D:/Documents/Final/Mandelbrot_Cpp_Final/src/main.cpp File Reference 122
6.26.1 Function Documentation 123
6.26.1.1 main() 123
6.27 main.cpp 124
6.28 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Mandelbrot.cpp File Reference 126
6.28.1 Function Documentation 126
6.28.1.1 operator<<() 126
6.29 Mandelbrot.cpp 127
6.30 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Mandelbrot.h File Reference 130
6.30.1 Macro Definition Documentation 131
6.30.1.1 M_PI 131
6.31 Mandelbrot.h 131
6.32 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Neumorphic.cpp File Reference 133
6.33 Neumorphic.cpp 133
6.34 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Neumorphic.h File Reference 134
6.35 Neumorphic.h 135
6.36 D:/Documents/Final/Mandelbrot_Cpp_Final/src/NormalMap.cpp File Reference 136
6.37 NormalMap.cpp 136
6.38 D:/Documents/Final/Mandelbrot_Cpp_Final/src/NormalMap.h File Reference 136
6.39 NormalMap.h 137

6.40 D:/Documents/Final/Mandelbrot_Cpp_Final/src/PPM.cpp File Reference 138
6.41 PPM.cpp 138
6.42 D:/Documents/Final/Mandelbrot_Cpp_Final/src/PPM.h File Reference 139
6.43 PPM.h 140
6.44 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Shading.cpp File Reference 141
6.45 Shading.cpp 142
6.46 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Shading.h File Reference 143
6.47 Shading.h 143
6.48 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Striping.cpp File Reference 144
6.49 Striping.cpp 145
6.50 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Striping.h File Reference 145
6.51 Striping.h 146
6.52 D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPool.cpp File Reference 147
6.53 ThreadPool.cpp 147
6.54 D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPool.h File Reference 148
6.55 ThreadPool.h 149
6.56 D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPoolTest.cpp File Reference 150
6.56.1 Function Documentation 151
6.56.1.1 main() 151
6.57 ThreadPoolTest.cpp 152
6.58 D:/Documents/Final/Mandelbrot_Cpp_Final/src/UnboundedQueue.h File Reference 153
6.59 UnboundedQueue.h 154

Index **157**

Chapter 1

Gigabrot

Gigabrot_C++

This is a C++ implementation of my original Gigabrot project, put into this repo for ease of submission. Its goal is to take in user-specified parameters and efficiently produce a PPM image render of the Mandelbrot set with stripe-average colorization and a pseudo-neumorphic normal mapping effect. This project also demonstrates various fundamental aspects of coding using the C++ language. Sample output image shown above.

Jump to Usage»

· Report Bug / Request Feature ·

1.1 About The Project

The Mandelbrot set is a set of complex numbers named for Benoit B. Mandelbrot for which the function $f_c(z) = z^2 + c$ does not exist when z is iterated from 0. The implications of this set of numbers range far and wide in the world of mathematics, and it is often a subject of interest in computing due to its many notable properties and ease of computing. It is also widely known for its aesthetic appeal as a fractal and the many coloring techniques that can be implemented.

The stripe average colorization method implemented here, described in Jussi Härkönen's "On fractal coloring techniques", is an extension of the Triangle Inequality Average method and highlights various features of the fractal. It is combined here with a normal vector map calculated to simulate the soft, shadowy nature of modern neumorphic UI design.

A key aspect of the set's applications in computing is that the iteration is *embarrassingly parallel*, meaning it can be easily separated into parallel tasks. Thus, the creation of the thread pool in this project. This pool uses only C++ provide libraries and uses an unlocked work queue to allow for job-stealing, inspired by Sean Parent's talk.

1.1.1 Note:

Parallelization of the main Mandelbrot code has not been implemented. The vector used to cache each row of the set to be written to the PPM file stream output throws a wrench in the "embarrassingly parallel" mix and I did not have time to fully debug the corrupted image output. While I removed that part of the code from my `main()`, I have included the test program I used to verify that the thread pool and its exception handling are in fact working.

1.1.1.1 Built With:

- CLion
- Cygwin

1.2 Getting Started

To get a local copy up and running follow these simple steps:

1.2.1 Prerequisites

- C++17
- CMake 3.21

To view the output `.ppm` files and convert to `.png`:

- `Netpbm Viewer`

1.2.2 Cloning

1. Clone the repo into the desired directory
   ```
   git clone https://github.com/c-biancone/Mandelbrot_Cpp_Final.git
   ```
2. Open the directory from within CLion and build

1.3 Usage

1. In CLion, hit the play button to build and run the code. In the terminal, enter the image resolution when prompted. This value is automatically checked and copied so that the image is square for this implementation. Enter the desired filename when prompted.
2. Once the code has finished running, the output can be found within the `output` directory. A 1280 x 1280 image is generated in ∼2.5s on my laptop, and scales as expected according to image size.
3. Drag and drop this file into `Netpbm Viewer`.
4. If this file is to your liking, download the `.png` file.

1.3.0.0.1 Test Thread Pool:

1. Find `CMakeLists.txt` and switch the comment under "executables" from `src/ThreadPoolTest.cpp` to `src/main.cpp`.
2. Build and run code as before. This main function will generate a random number of exceptions thrown from within the threads of the thread pool. Run repeatedly to get different output.

1.4 Contact

Chris Biancone - `email`

Project Link: `https://github.com/pinecone19/Gigabrot`

1.4.1 Demonstrated C++ Fundamentals

1.4.1.1 For Reference

- Class Inheritance
  - 3-level hierarchy
  - Is-a and Has-a relationship
  - Polymorphism
    - [pure] virtual functions
    - Abstract classes
    - Downcasting
- Operator Overloading
  - Copy constructor
  - Assignment and stream insertion operators
- Data Structures
  - (Unbounded) queue
  - Vector
- Exception handling
  - Try/catch
  - Throw/rethrow
- C++ STL
- Custom Templated Classes
- File Procesing
- String Stream Processing

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

- Colorization 11
  - Shading 60
    - InsideColor 15
    - LineColor 19
    - Striping 64
- Mandelbrot 23
- NormalMap 46
  - Neumorphic 41
- PPM 51
- ThreadPool 68
- UnboundedQueue< T > 74

Chapter 3

Class Index

3.1 Class List

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

Colorization 11
InsideColor 15
LineColor 19
Mandelbrot 23
Neumorphic 41
NormalMap 46
PPM 51
Shading 60
Striping 64
ThreadPool 68
UnboundedQueue< T > 74

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/CMakeFiles/3.21.1/CompilerIdC/CMakeCCompilerId.c 83
D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/CMakeFiles/3.21.1/CompilerIdCXX/CMakeCXXCompilerId.cpp 96
D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/CMakeFiles/FindOpenMP/OpenMPCheckVersion.c 109
D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/CMakeFiles/FindOpenMP/OpenMPCheckVersion.cpp 111
D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/CMakeFiles/FindOpenMP/OpenMPTryFlag.c 113
D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/CMakeFiles/FindOpenMP/OpenMPTryFlag.cpp 114
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Colorization.cpp 115
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Colorization.h 116
D:/Documents/Final/Mandelbrot_Cpp_Final/src/InsideColor.cpp 117
D:/Documents/Final/Mandelbrot_Cpp_Final/src/InsideColor.h 118
D:/Documents/Final/Mandelbrot_Cpp_Final/src/LineColor.cpp 120
D:/Documents/Final/Mandelbrot_Cpp_Final/src/LineColor.h 121
D:/Documents/Final/Mandelbrot_Cpp_Final/src/main.cpp 122
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Mandelbrot.cpp 126
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Mandelbrot.h 130
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Neumorphic.cpp 133
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Neumorphic.h 134
D:/Documents/Final/Mandelbrot_Cpp_Final/src/NormalMap.cpp 136
D:/Documents/Final/Mandelbrot_Cpp_Final/src/NormalMap.h 136
D:/Documents/Final/Mandelbrot_Cpp_Final/src/PPM.cpp 138
D:/Documents/Final/Mandelbrot_Cpp_Final/src/PPM.h 139
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Shading.cpp 141
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Shading.h 143
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Striping.cpp 144
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Striping.h 145
D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPool.cpp 147
D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPool.h 148
D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPoolTest.cpp 150
D:/Documents/Final/Mandelbrot_Cpp_Final/src/UnboundedQueue.h 153

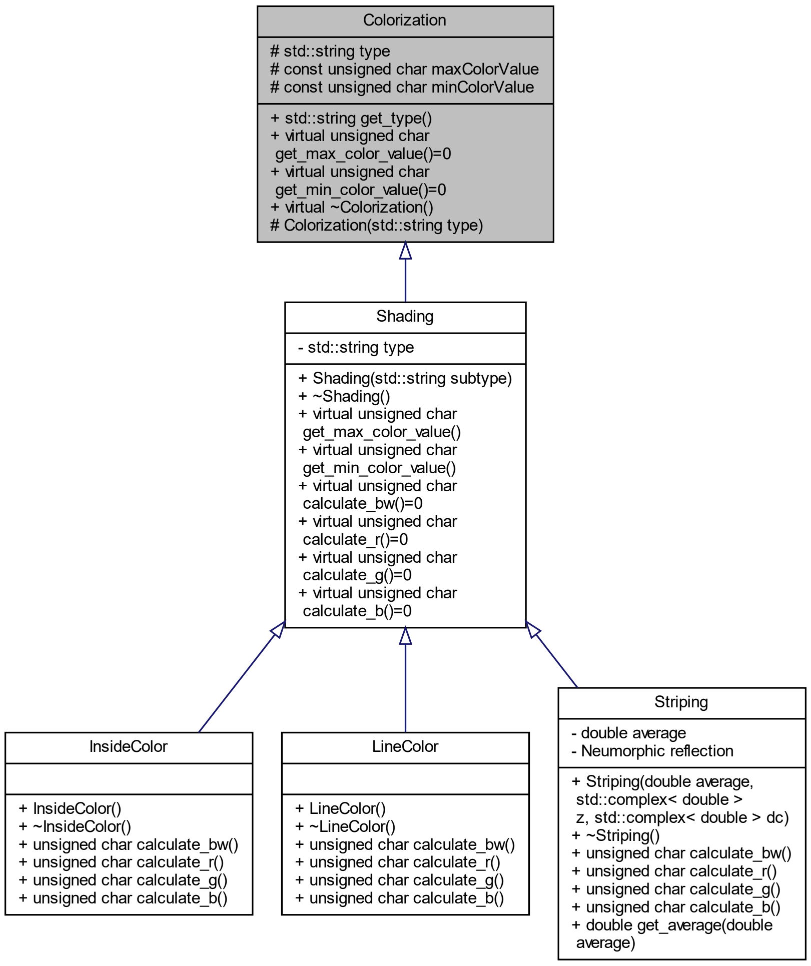
Chapter 5

Class Documentation

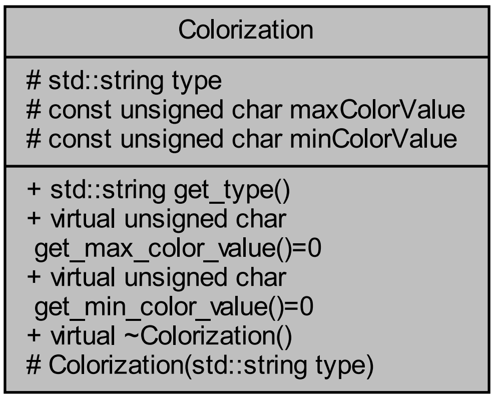
5.1 Colorization Class Reference

`#include <Colorization.h>`

Inheritance diagram for Colorization:


Colorization
# std::string type
# const unsigned char maxColorValue
# const unsigned char minColorValue
+ std::string get_type()
+ virtual unsigned char get_max_color_value()=0
+ virtual unsigned char get_min_color_value()=0
+ virtual ~Colorization()
# Colorization(std::string type)
Shading
- std::string type
+ Shading(std::string subtype)
+ ~Shading()
+ virtual unsigned char get_max_color_value()
+ virtual unsigned char get_min_color_value()
+ virtual unsigned char calculate_bw()=0
+ virtual unsigned char calculate_r()=0
+ virtual unsigned char calculate_g()=0
+ virtual unsigned char calculate_b()=0
InsideColor
+ InsideColor()
+ ~InsideColor()
+ unsigned char calculate_bw()
+ unsigned char calculate_r()
+ unsigned char calculate_g()
+ unsigned char calculate_b()
LineColor
+ LineColor()
+ ~LineColor()
+ unsigned char calculate_bw()
+ unsigned char calculate_r()
+ unsigned char calculate_g()
+ unsigned char calculate_b()
Striping
- double average
- Neumorphic reflection
+ Striping(double average, std::complex< double > z, std::complex< double > dc)
+ ~Striping()
+ unsigned char calculate_bw()
+ unsigned char calculate_r()
+ unsigned char calculate_g()
+ unsigned char calculate_b()
+ double get_average(double average)

Collaboration diagram for Colorization:


Colorization
# std::string type
# const unsigned char maxColorValue
# const unsigned char minColorValue
+ std::string get_type()
+ virtual unsigned char get_max_color_value()=0
+ virtual unsigned char get_min_color_value()=0
+ virtual ~Colorization()
# Colorization(std::string type)

Public Member Functions

- std::string get_type ()
- virtual unsigned char get_max_color_value ()=0
- virtual unsigned char get_min_color_value ()=0
- virtual ∼Colorization ()

Protected Member Functions

- Colorization (std::string type)

Protected Attributes

- std::string type
- const unsigned char maxColorValue = 255
- const unsigned char minColorValue = 0

5.1.1 Detailed Description

Definition at line 6 of file Colorization.h.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 ∼Colorization()

```
Colorization::∼Colorization ( )  [virtual], [default]
```

5.1.2.2 Colorization()

```
Colorization::Colorization (
            std::string type )  [explicit], [protected]
```

Parametrized constructor

Parameters

type	of colorization

Definition at line 7 of file Colorization.cpp.

```
                                      : type(std::move(type))
{}
```

5.1.3 Member Function Documentation

5.1.3.1 get_max_color_value()

```
virtual unsigned char Colorization::get_max_color_value ( )  [pure virtual]
```

Implemented in Shading.

5.1.3.2 get_min_color_value()

```
virtual unsigned char Colorization::get_min_color_value ( )  [pure virtual]
```

Implemented in Shading.

5.1.3.3 get_type()

```
std::string Colorization::get_type ( )
```

Returns

type of colorization

Definition at line 10 of file Colorization.cpp.

```
{
  return type;
}
```

5.1.4 Member Data Documentation

5.1.4.1 maxColorValue

```
const unsigned char Colorization::maxColorValue = 255  [protected]
```

Definition at line 34 of file Colorization.h.

5.1.4.2 minColorValue

```
const unsigned char Colorization::minColorValue = 0  [protected]
```

Definition at line 36 of file Colorization.h.

5.1.4.3 type

```
std::string Colorization::type  [protected]
```

Determines type of colorization

Definition at line 32 of file Colorization.h.

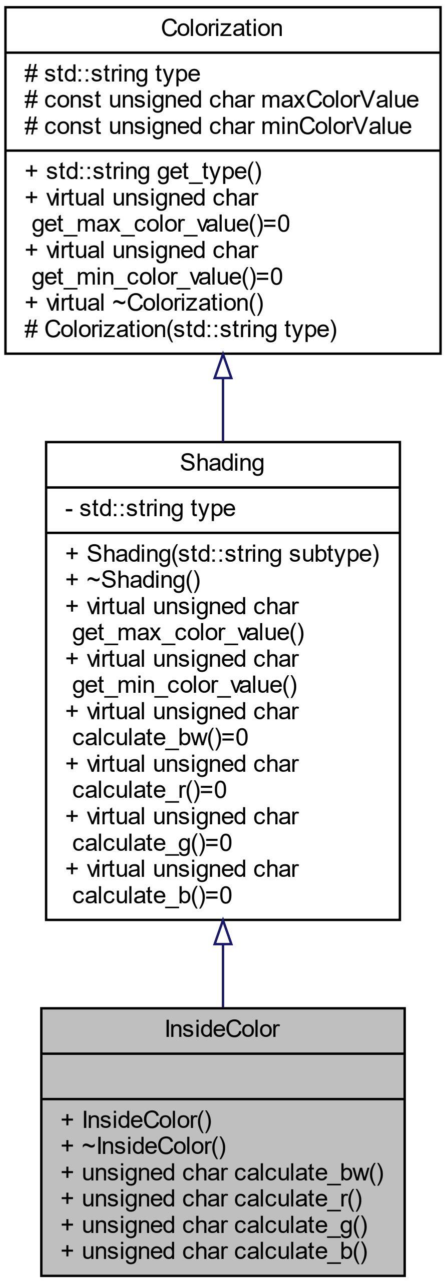
The documentation for this class was generated from the following files:

- D:/Documents/Final/Mandelbrot_Cpp_Final/src/Colorization.h
- D:/Documents/Final/Mandelbrot_Cpp_Final/src/Colorization.cpp

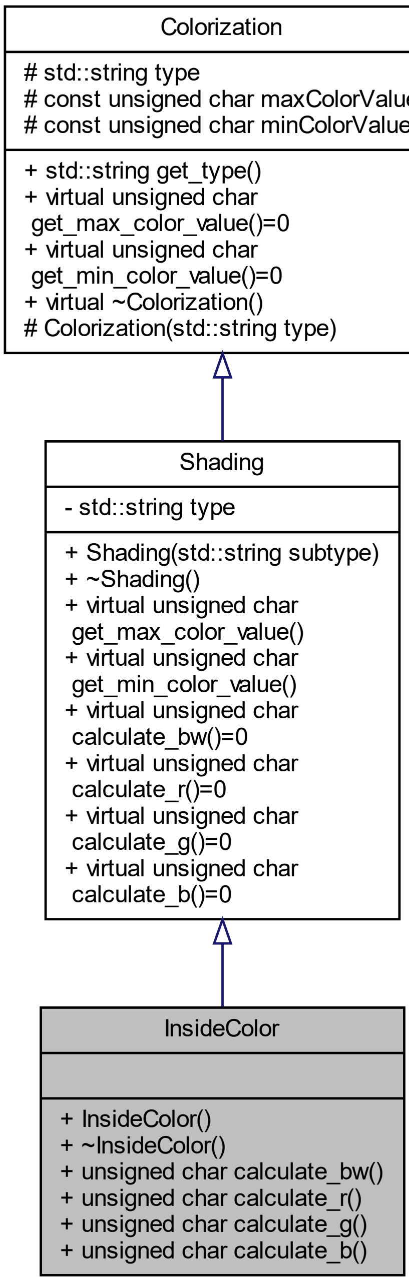
5.2 InsideColor Class Reference

```
#include <InsideColor.h>
```

Inheritance diagram for InsideColor:


Colorization
# std::string type
# const unsigned char maxColorValue
# const unsigned char minColorValue
+ std::string get_type()
+ virtual unsigned char get_max_color_value()=0
+ virtual unsigned char get_min_color_value()=0
+ virtual ~Colorization()
# Colorization(std::string type)
Shading
- std::string type
+ Shading(std::string subtype)
+ ~Shading()
+ virtual unsigned char get_max_color_value()
+ virtual unsigned char get_min_color_value()
+ virtual unsigned char calculate_bw()=0
+ virtual unsigned char calculate_r()=0
+ virtual unsigned char calculate_g()=0
+ virtual unsigned char calculate_b()=0
InsideColor
+ InsideColor()
+ ~InsideColor()
+ unsigned char calculate_bw()
+ unsigned char calculate_r()
+ unsigned char calculate_g()
+ unsigned char calculate_b()

Collaboration diagram for InsideColor:


Colorization
# std::string type
# const unsigned char maxColorValue
# const unsigned char minColorValue
+ std::string get_type()
+ virtual unsigned char get_max_color_value()=0
+ virtual unsigned char get_min_color_value()=0
+ virtual ~Colorization()
# Colorization(std::string type)
Shading
- std::string type
+ Shading(std::string subtype)
+ ~Shading()
+ virtual unsigned char get_max_color_value()
+ virtual unsigned char get_min_color_value()
+ virtual unsigned char calculate_bw()=0
+ virtual unsigned char calculate_r()=0
+ virtual unsigned char calculate_g()=0
+ virtual unsigned char calculate_b()=0
InsideColor
+ InsideColor()
+ ~InsideColor()
+ unsigned char calculate_bw()
+ unsigned char calculate_r()
+ unsigned char calculate_g()
+ unsigned char calculate_b()

Public Member Functions

- InsideColor ()
- ∼InsideColor ()
- unsigned char calculate_bw ()
- unsigned char calculate_r ()
- unsigned char calculate_g ()
- unsigned char calculate_b ()

Additional Inherited Members

5.2.1 Detailed Description

Definition at line 6 of file InsideColor.h.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 InsideColor()

```
InsideColor::InsideColor ( )
```

Definition at line 3 of file InsideColor.cpp.

```
                          : Shading("Inside")
{}
```

5.2.2.2 ∼InsideColor()

```
InsideColor::∼InsideColor ( )  [default]
```

5.2.3 Member Function Documentation

5.2.3.1 calculate_b()

```
unsigned char InsideColor::calculate_b ( )  [virtual]
```

Implements Shading.

Definition at line 24 of file InsideColor.cpp.

```
{
  return minColorValue;
}
```

5.2.3.2 calculate_bw()

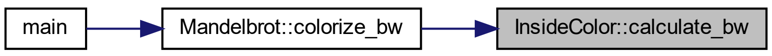
```
unsigned char InsideColor::calculate_bw ( )  [virtual]
```

Implements Shading.

Definition at line 9 of file InsideColor.cpp.

```
00010 {
00011   return minColorValue;
00012 }
```

Here is the caller graph for this function:



5.2.3.3 calculate_g()

```
unsigned char InsideColor::calculate_g ( )  [virtual]
```

Implements Shading.

Definition at line 19 of file InsideColor.cpp.

```
00020 {
00021   return minColorValue;
00022 }
```

5.2.3.4 calculate_r()

```
unsigned char InsideColor::calculate_r ( )  [virtual]
```

Implements Shading.

Definition at line 14 of file InsideColor.cpp.

```
00015 {
00016   return minColorValue;
00017 }
```

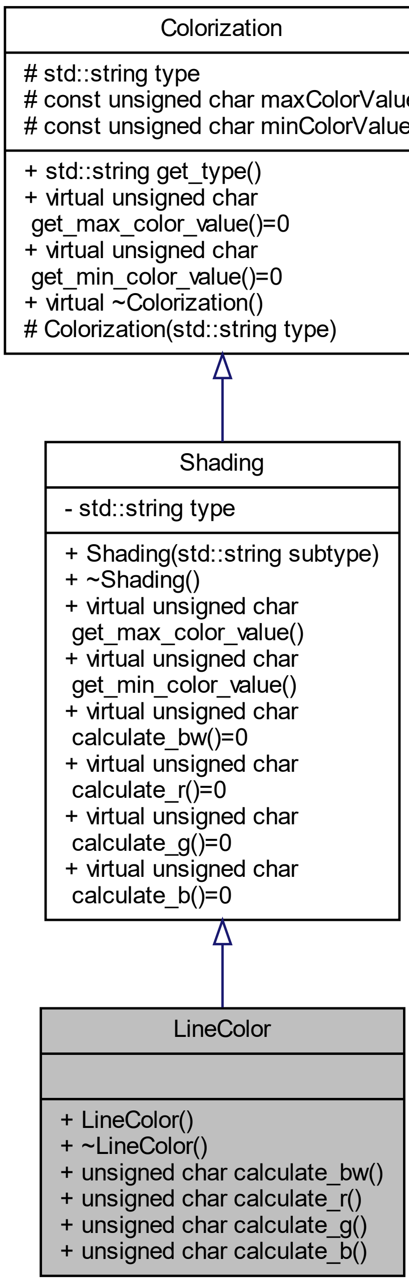
The documentation for this class was generated from the following files:

- D:/Documents/Final/Mandelbrot_Cpp_Final/src/InsideColor.h
- D:/Documents/Final/Mandelbrot_Cpp_Final/src/InsideColor.cpp

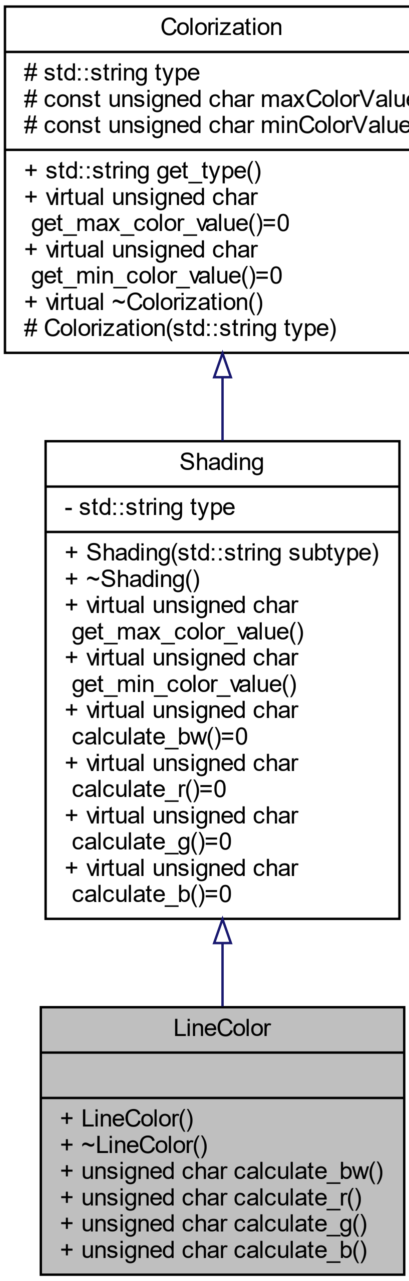
5.3 LineColor Class Reference

```
#include <LineColor.h>
```

Inheritance diagram for LineColor:


Colorization
# std::string type
# const unsigned char maxColorValue
# const unsigned char minColorValue
+ std::string get_type()
+ virtual unsigned char get_max_color_value()=0
+ virtual unsigned char get_min_color_value()=0
+ virtual ~Colorization()
# Colorization(std::string type)
Shading
- std::string type
+ Shading(std::string subtype)
+ ~Shading()
+ virtual unsigned char get_max_color_value()
+ virtual unsigned char get_min_color_value()
+ virtual unsigned char calculate_bw()=0
+ virtual unsigned char calculate_r()=0
+ virtual unsigned char calculate_g()=0
+ virtual unsigned char calculate_b()=0
LineColor
+ LineColor()
+ ~LineColor()
+ unsigned char calculate_bw()
+ unsigned char calculate_r()
+ unsigned char calculate_g()
+ unsigned char calculate_b()

Collaboration diagram for LineColor:


Colorization
# std::string type
# const unsigned char maxColorValue
# const unsigned char minColorValue
+ std::string get_type()
+ virtual unsigned char get_max_color_value()=0
+ virtual unsigned char get_min_color_value()=0
+ virtual ~Colorization()
# Colorization(std::string type)
Shading
- std::string type
+ Shading(std::string subtype)
+ ~Shading()
+ virtual unsigned char get_max_color_value()
+ virtual unsigned char get_min_color_value()
+ virtual unsigned char calculate_bw()=0
+ virtual unsigned char calculate_r()=0
+ virtual unsigned char calculate_g()=0
+ virtual unsigned char calculate_b()=0
LineColor
+ LineColor()
+ ~LineColor()
+ unsigned char calculate_bw()
+ unsigned char calculate_r()
+ unsigned char calculate_g()
+ unsigned char calculate_b()

Public Member Functions

- LineColor ()
- ∼LineColor ()
- unsigned char calculate_bw ()
- unsigned char calculate_r ()
- unsigned char calculate_g ()
- unsigned char calculate_b ()

Additional Inherited Members

5.3.1 Detailed Description

Definition at line 6 of file LineColor.h.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 LineColor()

```
LineColor::LineColor ( )
```

Definition at line 3 of file LineColor.cpp.

```
00003                         : Shading("Line")
00004 {}
```

5.3.2.2 ∼LineColor()

```
LineColor::∼LineColor ( )  [default]
```

5.3.3 Member Function Documentation

5.3.3.1 calculate_b()

```
unsigned char LineColor::calculate_b ( )  [virtual]
```

Implements Shading.

Definition at line 24 of file LineColor.cpp.

```
00025 {
00026   return maxColorValue;
00027 }
```

5.3.3.2 calculate_bw()

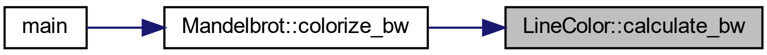
```
unsigned char LineColor::calculate_bw ( )  [virtual]
```

Implements Shading.

Definition at line 9 of file LineColor.cpp.

```
{
  return maxColorValue;
}
```

Here is the caller graph for this function:



5.3.3.3 calculate_g()

```
unsigned char LineColor::calculate_g ( )  [virtual]
```

Implements Shading.

Definition at line 19 of file LineColor.cpp.

```
{
  return maxColorValue;
}
```

5.3.3.4 calculate_r()

```
unsigned char LineColor::calculate_r ( )  [virtual]
```

Implements Shading.

Definition at line 14 of file LineColor.cpp.

```
{
  return maxColorValue;
}
```

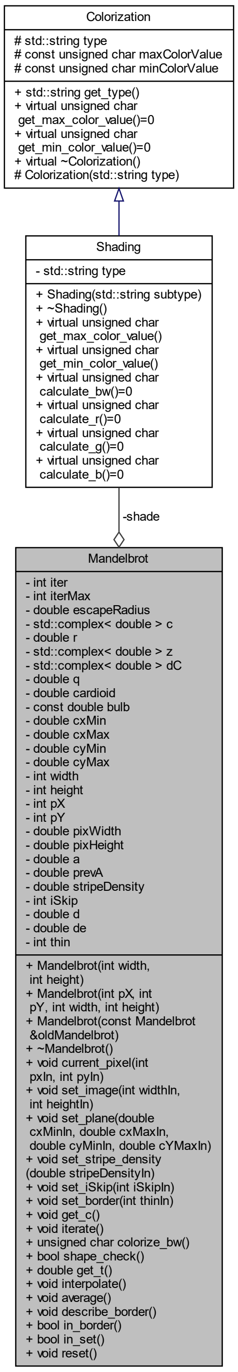
The documentation for this class was generated from the following files:

- D:/Documents/Final/Mandelbrot_Cpp_Final/src/LineColor.h
- D:/Documents/Final/Mandelbrot_Cpp_Final/src/LineColor.cpp

5.4 Mandelbrot Class Reference

```
#include <Mandelbrot.h>
```

Collaboration diagram for Mandelbrot:


Colorization
# std::string type
# const unsigned char maxColorValue
# const unsigned char minColorValue
+ std::string get_type()
+ virtual unsigned char get_max_color_value()=0
+ virtual unsigned char get_min_color_value()=0
+ virtual ~Colorization()
# Colorization(std::string type)
Shading
- std::string type
+ Shading(std::string subtype)
+ ~Shading()
+ virtual unsigned char get_max_color_value()
+ virtual unsigned char get_min_color_value()
+ virtual unsigned char calculate_bw()=0
+ virtual unsigned char calculate_r()=0
+ virtual unsigned char calculate_g()=0
+ virtual unsigned char calculate_b()=0
-shade
Mandelbrot
- int iter
- int iterMax
- double escapeRadius
- std::complex< double > c
- double r
- std::complex< double > z
- std::complex< double > dC
- double q
- double cardioid
- const double bulb
- double cxMin
- double cxMax
- double cyMin
- double cyMax
- int width
- int height
- int pX
- int pY
- double pixWidth
- double pixHeight
- double a
- double prevA
- double stripeDensity
- int iSkip
- double d
- double de
- int thin
+ Mandelbrot(int width, int height)
+ Mandelbrot(int pX, int pY, int width, int height)
+ Mandelbrot(const Mandelbrot &oldMandelbrot)
+ ~Mandelbrot()
+ void current_pixel(int pxIn, int pyIn)
+ void set_image(int widthIn, int heightIn)
+ void set_plane(double cxMinIn, double cxMaxIn, double cyMinIn, double cYMaxIn)
+ void set_stripe_density (double stripeDensityIn)
+ void set_iSkip(int iSkipIn)
+ void set_border(int thinIn)
+ void get_c()
+ void iterate()
+ unsigned char colorize_bw()
+ bool shape_check()
+ double get_t()
+ void interpolate()
+ void average()
+ void describe_border()
+ bool in_border()
+ bool in_set()
+ void reset()

Public Member Functions

- Mandelbrot (int width, int height)

- Mandelbrot (int pX, int pY, int width, int height)
- Mandelbrot (const Mandelbrot &oldMandelbrot)
- ∼Mandelbrot ()
- void current_pixel (int pxIn, int pyIn)
- void set_image (int widthIn, int heightIn)
- void set_plane (double cxMinIn, double cxMaxIn, double cyMinIn, double cYMaxIn)
- void set_stripe_density (double stripeDensityIn)
- void set_iSkip (int iSkipIn)
- void set_border (int thinIn)
- void get_c ()
- void iterate ()
- unsigned char colorize_bw ()
- bool shape_check ()
- double get_t ()
- void interpolate ()
- void average ()
- void describe_border ()
- bool in_border ()
- bool in_set ()
- void reset ()

Private Attributes

- int iter
- int iterMax
- double escapeRadius
- std::complex$<$ double $>$ c
- double r
- std::complex$<$ double $>$ z
- std::complex$<$ double $>$ dC
- double q
- double cardioid
- const double bulb = 0.0625
- double cxMin
- double cxMax
- double cyMin
- double cyMax
- int width
- int height
- int pX
- int pY
- double pixWidth
- double pixHeight
- double a
- double prevA
- double stripeDensity
- int iSkip
- double d
- double de
- int thin
- Shading $*$ shade

Friends

- std::ostream & operator<< (std::ostream &os, const Mandelbrot &mandelbrot)

5.4.1 Detailed Description

Definition at line 16 of file Mandelbrot.h.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 Mandelbrot() [1/3]

```
Mandelbrot::Mandelbrot (
              int width,
              int height )
```

Default parametrized constructor

Parameters

width	- image width
height	- image height

Definition at line 5 of file Mandelbrot.cpp.

```
                                             : width(width), height(height)
{
  pX = 0;
  pY = 0;
  iter = 0;
  iterMax = 1000;
  escapeRadius = 1000000.0;
  // lnER = log(escapeRadius);
  c = 0.0;
  r = 0.0;
  z = 0.0;
  dC = 0.0;
  q = 0.0;
  cardioid = 0.0;
  a = 0.0;
  prevA = 0.0;
  stripeDensity = 7.0;
  d = 0.0;
  de = 0.0;
  cxMin = -2.2;
  cxMax = 0.8;
  cyMin = -1.5;
  cyMax = 1.5;
  pixWidth = 0.0;
  pixHeight = 0.0;
  iSkip = 1;
  thin = 3;
  shade = nullptr; // avoid calling "new" more than once per pixel
}
```

5.4.2.2 Mandelbrot() [2/3]

```
Mandelbrot::Mandelbrot (
            int pX,
            int pY,
            int width,
            int height )
```

Minimal parametrized constructor used for calling within pixel loops

Parameters

pX	- current pixel x
pY	- current pixel y
width	- image width
height	- image height

Definition at line 35 of file Mandelbrot.cpp.

```
                                                        : pX(pX), pY(pY), width(width),
height(height)
{
  iter = 0;
  iterMax = 1000;
  escapeRadius = 1000000.0;
  // lnER = log(escapeRadius);
  c = 0.0;
  r = 0.0;
  z = 0.0;
  dC = 0.0;
  q = 0.0;
  cardioid = 0.0;
  a = 0.0;
  prevA = 0.0;
  stripeDensity = 7.0;
  d = 0.0;
  de = 0.0;
  cxMin = -2.2;
  cxMax = 0.8;
  cyMin = -1.5;
  cyMax = 1.5;
  pixWidth = 0.0;
  pixHeight = 0.0;
  iSkip = 1;
  thin = 3;
  shade = nullptr; // avoid calling "new" more than once per pixel
}
```

5.4.2.3 Mandelbrot() [3/3]

```
Mandelbrot::Mandelbrot (
            const Mandelbrot & oldMandelbrot )
```

Definition at line 256 of file Mandelbrot.cpp.

```
                                                  : Mandelbrot(oldMandelbrot.width,
                                                               oldMandelbrot.height)
{}
```

5.4.2.4 ∼Mandelbrot()

```
Mandelbrot::∼Mandelbrot ( )
```

Definition at line 64 of file Mandelbrot.cpp.

```
{
  delete shade;
}
```

5.4.3 Member Function Documentation

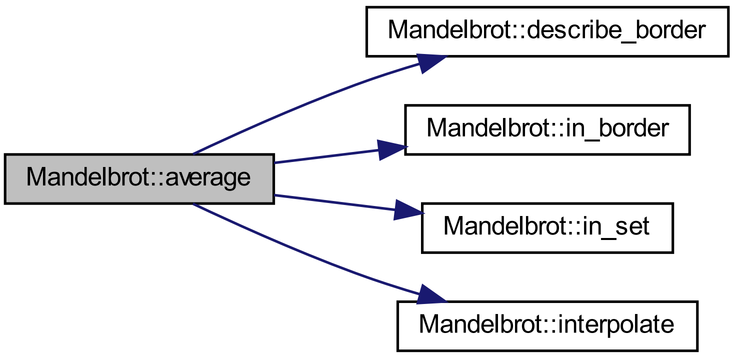
5.4.3.1 average()

```
void Mandelbrot::average ( )
```

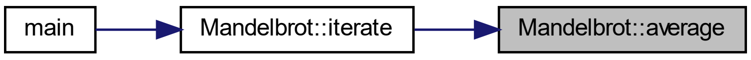
Definition at line 187 of file Mandelbrot.cpp.

```
{
  if (in_set())
  {
    a = -1.0;
  } else {
    describe_border();
    if (in_border()) // in border
    {
      a = FP_ZERO;
    } else {
      a /= static_cast<double>((iter - iSkip)); // A(n)
      prevA /= static_cast<double>((iter - iSkip - 1)); // A(n-1)
      this->interpolate();
      a = (d * a) + ((1.0 - d) * prevA);
    }
  }
}
```

Here is the call graph for this function:



Here is the caller graph for this function:



5.4.3.2 colorize_bw()

```
unsigned char Mandelbrot::colorize_bw ( )
```

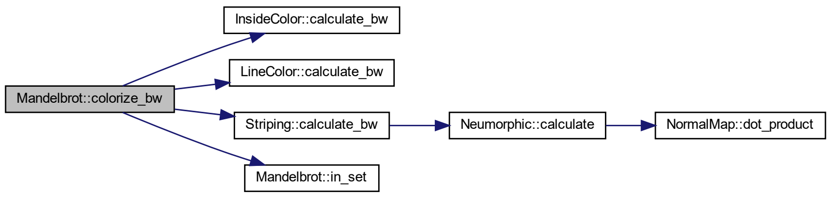
Returns

single output pixel value

Definition at line 140 of file Mandelbrot.cpp.

```
{
  if (in_set())
  {
    shade = new InsideColor();
    InsideColor *color;
    color = dynamic_cast<InsideColor*>(shade);
    return color->calculate_bw();
  } else if (a == FP_ZERO) {
    shade = new LineColor();
    LineColor *color;
    color = dynamic_cast<LineColor*>(shade);
    return color->calculate_bw();
  } else {
    shade = new Striping(a, z, dC);
    Striping *color;
    color = dynamic_cast<Striping*>(shade);
    return color->calculate_bw();
  }
}
```

Here is the call graph for this function:



Here is the caller graph for this function:

main
Mandelbrot::colorize_bw

5.4.3.3 current_pixel()

```
void Mandelbrot::current_pixel (
             int pxIn,
             int pyIn )
```

Set current position within image

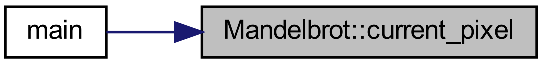
Parameters

px↩ In	
py↩ In	

Definition at line 75 of file Mandelbrot.cpp.

```
{
  pX = pxIn;
  pY = pyIn;
}
```

Here is the caller graph for this function:



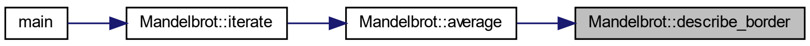
5.4.3.4 describe_border()

```
void Mandelbrot::describe_border ( )
```

Definition at line 206 of file Mandelbrot.cpp.

```
{
  de = 2.0 * r * log(r) / abs(dC);
}
```

Here is the caller graph for this function:



5.4.3.5 get_c()

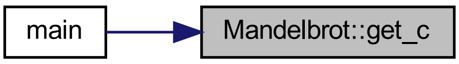
```
void Mandelbrot::get_c ( )
```

Determine where pixel lies in complex plane

Definition at line 104 of file Mandelbrot.cpp.

```
{
  pixWidth = (cxMax-cxMin) / static_cast<double>(width);
  pixHeight = (cyMax-cyMin) / static_cast<double>(height);
  c = (cxMin + static_cast<double>(pX) * pixWidth) + ((cyMax - static_cast<double>(pY) * pixHeight) *
 1i);
}
```

Here is the caller graph for this function:



5.4.3.6 get_t()

```
double Mandelbrot::get_t ( )
```

addend function

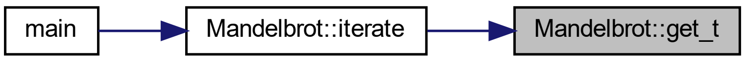
Returns

mapped real number t

Definition at line 174 of file Mandelbrot.cpp.

```
{
  return 0.5 + 0.5 * sin(stripeDensity * arg(z));
}
```

Here is the caller graph for this function:



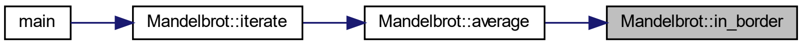
5.4.3.7 in_border()

```
bool Mandelbrot::in_border ( )
```

Definition at line 211 of file Mandelbrot.cpp.

```
{
  if (de < (pixWidth / static_cast<double>(thin)))
  {
    return true;
  } else {
    return false;
  }
}
```

Here is the caller graph for this function:



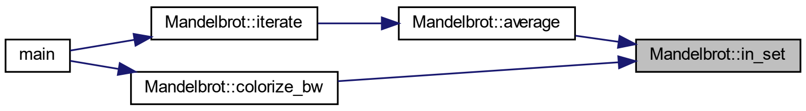
5.4.3.8 in_set()

```
bool Mandelbrot::in_set ( )
```

Definition at line 221 of file Mandelbrot.cpp.

```
{
  if (iter == iterMax)
  {
    return true;
  } else {
    return false;
  }
}
```

Here is the caller graph for this function:



5.4.3.9 interpolate()

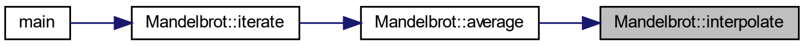
```
void Mandelbrot::interpolate ( )
```

Removes level sets of escape time

Definition at line 179 of file Mandelbrot.cpp.

```
{
  // smooth iteration count
  d = static_cast<double>(iter + 1) + log(log(escapeRadius) / log(r)) / M_LN2;
  d = d - static_cast<double>(static_cast<int>(d)); // only fractional part = interpolation
  // coefficient
}
```

Here is the caller graph for this function:



5.4.3.10 iterate()

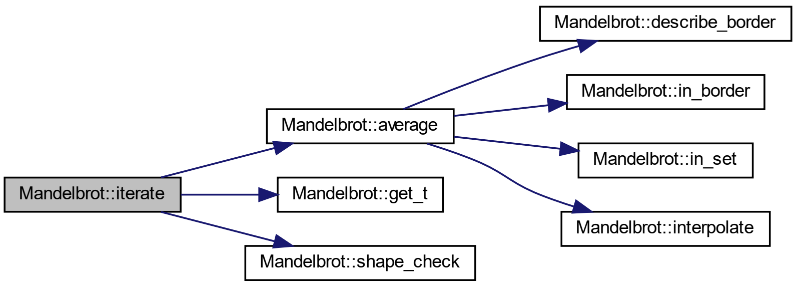
```
void Mandelbrot::iterate ( )
```

Main Mandelbrot function

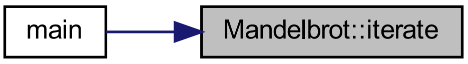
Definition at line 111 of file Mandelbrot.cpp.

```
{
  if (!this->shape_check())
  {
    for (iter = 0; iter < iterMax; iter++)
    {
      // mandelbrot set formula
      dC = 2.0 * dC * z + 1.0;
      z = z * z + c;

      // compute average
      if (iter > iSkip)
      {
        a += get_t();
      }

      r = abs(z);
      if (r > escapeRadius)
      {
        break;
      }

      prevA = a;
    }

    average();
  }
}
```

Here is the call graph for this function:



Here is the caller graph for this function:



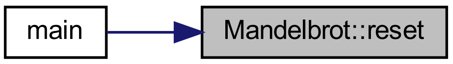
5.4.3.11 reset()

```
void Mandelbrot::reset ( )
```

Definition at line 242 of file Mandelbrot.cpp.

```
{
  c = 0.0;
  r = 0.0;
  z = 0.0;
  dC = 0.0;
  q = 0.0;
  cardioid = 0.0;
  a = 0.0;
  prevA = 0.0;
  d = 0.0;
  shade = nullptr; // avoid calling "new" more than once per pixel
}
```

Here is the caller graph for this function:



5.4.3.12 set_border()

```
void Mandelbrot::set_border (
            int thinIn )
```

Definition at line 99 of file Mandelbrot.cpp.

```
00100 {
00101   thin = thinIn;
00102 }
```

5.4.3.13 set_image()

```
void Mandelbrot::set_image (
            int widthIn,
            int heightIn )
```

Definition at line 69 of file Mandelbrot.cpp.

```
00070 {
00071   width = widthIn;
00072   height = heightIn;
00073 }
```

5.4.3.14 set_iSkip()

```
void Mandelbrot::set_iSkip (
            int iSkipIn )
```

Definition at line 94 of file Mandelbrot.cpp.

```
00095 {
00096   iSkip = iSkipIn;
00097 }
```

5.4.3.15 set_plane()

```
void Mandelbrot::set_plane (
            double cxMinIn,
            double cxMaxIn,
            double cyMinIn,
            double cYMaxIn )
```

Definition at line 81 of file Mandelbrot.cpp.

```
00082 {
00083   cxMin = cxMinIn;
00084   cxMax = cxMaxIn;
00085   cyMin = cyMinIn;
00086   cyMax = cYMaxIn;
00087 }
```

5.4.3.16 set_stripe_density()

```
void Mandelbrot::set_stripe_density (
            double stripeDensityIn )
```

Definition at line 89 of file Mandelbrot.cpp.

```
{
  stripeDensity = stripeDensityIn;
}
```

5.4.3.17 shape_check()

```
bool Mandelbrot::shape_check ( )
```

Shape checking algorithm - determines if point is within main cardioid or secondary bulb. Removes about 91% of the set from being iterated. Should not be implemented for a render that does not include these parts, will add unnecessary computing

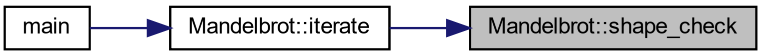
Returns

TRUE if within the main shapes

Definition at line 161 of file Mandelbrot.cpp.

```
{
    q = ((real(c) - 0.25) * (real(c) - 0.25)) + (imag(c) * imag(c));
    cardioid = 0.25 * imag(c) * imag(c);
    if ((real(c) * real(c) + 2.0 * real(c) + 1.0 + imag(c) * imag(c)) < bulb ||
        (q * (q +(real(c) - 0.25)) < cardioid))
    {
        return true;
    } else {
        return false;
    }
}
```

Here is the caller graph for this function:



5.4.4 Friends And Related Function Documentation

5.4.4.1 operator<<

```
std::ostream & operator<< (
            std::ostream & os,
            const Mandelbrot & mandelbrot )   [friend]
```

5.4.5 Member Data Documentation

5.4.5.1 a

```
double Mandelbrot::a  [private]
```

Definition at line 156 of file Mandelbrot.h.

5.4.5.2 bulb

```
const double Mandelbrot::bulb = 0.0625  [private]
```

independent

Definition at line 131 of file Mandelbrot.h.

5.4.5.3 c

```
std::complex<double> Mandelbrot::c  [private]
```

Definition at line 116 of file Mandelbrot.h.

5.4.5.4 cardioid

```
double Mandelbrot::cardioid  [private]
```

Definition at line 126 of file Mandelbrot.h.

5.4.5.5 cxMax

```
double Mandelbrot::cxMax  [private]
```

Definition at line 136 of file Mandelbrot.h.

5.4.5.6 cxMin

```
double Mandelbrot::cxMin  [private]
```

Definition at line 134 of file Mandelbrot.h.

5.4.5.7 cyMax

```
double Mandelbrot::cyMax  [private]
```

Definition at line 140 of file Mandelbrot.h.

5.4.5.8 cyMin

```
double Mandelbrot::cyMin  [private]
```

Definition at line 138 of file Mandelbrot.h.

5.4.5.9 d

```
double Mandelbrot::d  [private]
```

Definition at line 171 of file Mandelbrot.h.

5.4.5.10 dC

```
std::complex<double> Mandelbrot::dC  [private]
```

Definition at line 122 of file Mandelbrot.h.

5.4.5.11 de

```
double Mandelbrot::de  [private]
```

Definition at line 174 of file Mandelbrot.h.

5.4.5.12 escapeRadius

```
double Mandelbrot::escapeRadius  [private]
```

Definition at line 112 of file Mandelbrot.h.

5.4.5.13 height

```
int Mandelbrot::height  [private]
```

Definition at line 145 of file Mandelbrot.h.

5.4.5.14 iSkip

```
int Mandelbrot::iSkip  [private]
```

Exclude iSkip+1 elements from average

Definition at line 168 of file Mandelbrot.h.

5.4.5.15 iter

```
int Mandelbrot::iter  [private]
```

Definition at line 108 of file Mandelbrot.h.

5.4.5.16 iterMax

```
int Mandelbrot::iterMax  [private]
```

Definition at line 110 of file Mandelbrot.h.

5.4.5.17 pixHeight

```
double Mandelbrot::pixHeight  [private]
```

Definition at line 153 of file Mandelbrot.h.

5.4.5.18 pixWidth

```
double Mandelbrot::pixWidth  [private]
```

Definition at line 151 of file Mandelbrot.h.

5.4.5.19 prevA

```
double Mandelbrot::prevA  [private]
```

Definition at line 158 of file Mandelbrot.h.

5.4.5.20 pX

```
int Mandelbrot::pX  [private]
```

Definition at line 147 of file Mandelbrot.h.

5.4.5.21 pY

```
int Mandelbrot::pY  [private]
```

Definition at line 149 of file Mandelbrot.h.

5.4.5.22 q

```
double Mandelbrot::q  [private]
```

Definition at line 124 of file Mandelbrot.h.

5.4.5.23 r

```
double Mandelbrot::r  [private]
```

Definition at line 118 of file Mandelbrot.h.

5.4.5.24 shade

```
Shading* Mandelbrot::shade [private]
```

Definition at line 177 of file Mandelbrot.h.

5.4.5.25 stripeDensity

```
double Mandelbrot::stripeDensity [private]
```

Higher is more dense

Definition at line 163 of file Mandelbrot.h.

5.4.5.26 thin

```
int Mandelbrot::thin [private]
```

Definition at line 175 of file Mandelbrot.h.

5.4.5.27 width

```
int Mandelbrot::width [private]
```

Definition at line 143 of file Mandelbrot.h.

5.4.5.28 z

```
std::complex<double> Mandelbrot::z [private]
```

Definition at line 120 of file Mandelbrot.h.

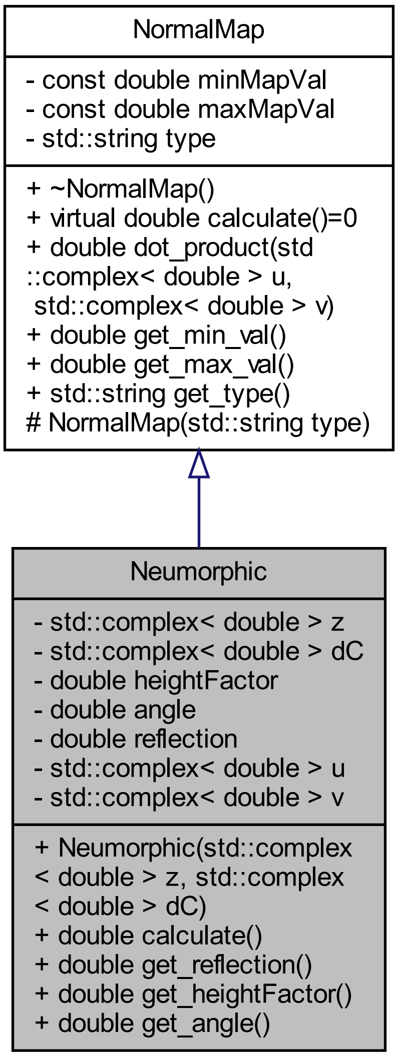
The documentation for this class was generated from the following files:

- D:/Documents/Final/Mandelbrot_Cpp_Final/src/Mandelbrot.h
- D:/Documents/Final/Mandelbrot_Cpp_Final/src/Mandelbrot.cpp

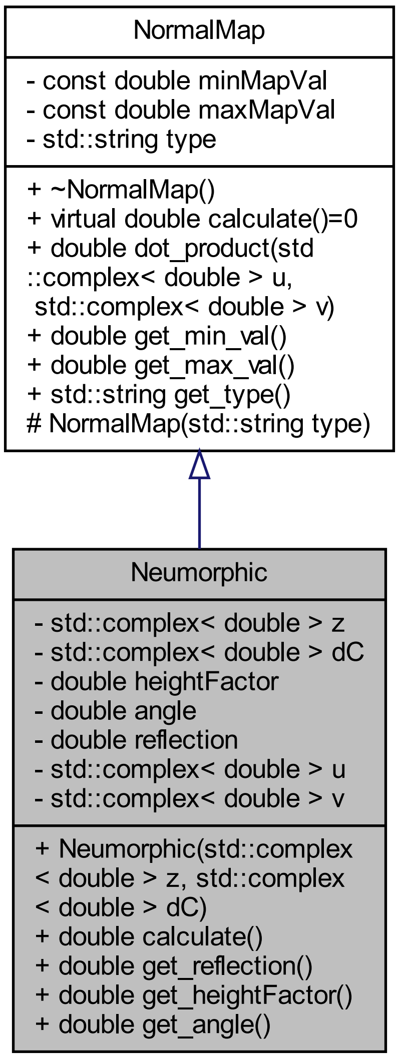
5.5 Neumorphic Class Reference

`#include <Neumorphic.h>`

Inheritance diagram for Neumorphic:


NormalMap
- const double minMapVal
- const double maxMapVal
- std::string type
+ ~NormalMap()
+ virtual double calculate()=0
+ double dot_product(std::complex< double > u, std::complex< double > v)
+ double get_min_val()
+ double get_max_val()
+ std::string get_type()
# NormalMap(std::string type)
Neumorphic
- std::complex< double > z
- std::complex< double > dC
- double heightFactor
- double angle
- double reflection
- std::complex< double > u
- std::complex< double > v
+ Neumorphic(std::complex< double > z, std::complex< double > dC)
+ double calculate()
+ double get_reflection()
+ double get_heightFactor()
+ double get_angle()

Collaboration diagram for Neumorphic:


NormalMap
- const double minMapVal
- const double maxMapVal
- std::string type
+ ~NormalMap()
+ virtual double calculate()=0
+ double dot_product(std ::complex< double > u, std::complex< double > v)
+ double get_min_val()
+ double get_max_val()
+ std::string get_type()
# NormalMap(std::string type)
Neumorphic
- std::complex< double > z
- std::complex< double > dC
- double heightFactor
- double angle
- double reflection
- std::complex< double > u
- std::complex< double > v
+ Neumorphic(std::complex < double > z, std::complex < double > dC)
+ double calculate()
+ double get_reflection()
+ double get_heightFactor()
+ double get_angle()

Public Member Functions

- Neumorphic (std::complex< double > z, std::complex< double > dC)
- double calculate ()
- double get_reflection ()
- double get_heightFactor ()
- double get_angle ()

Private Attributes

- std::complex< double > z
- std::complex< double > dC

- double heightFactor
- double angle
- double reflection
- std::complex< double > u
- std::complex< double > v

Additional Inherited Members

5.5.1 Detailed Description

Definition at line 6 of file Neumorphic.h.

5.5.2 Constructor & Destructor Documentation

5.5.2.1 Neumorphic()

```
Neumorphic::Neumorphic (
              std::complex< double > z,
              std::complex< double > dC )
```

Default parametrized constructor

Parameters

z	
dC	

Definition at line 5 of file Neumorphic.cpp.

```
00005                                                              : NormalMap("Neumorphic"), z(z), dC
00006 (dC)
00007 {
00008   heightFactor = 1.5;
00009   angle = 45.0 / 360.0;
00010   reflection = FP_ZERO;
00011   v = exp(2.0 * angle * M_PI * 1i);
00012 }
```

5.5.3 Member Function Documentation

5.5.3.1 calculate()

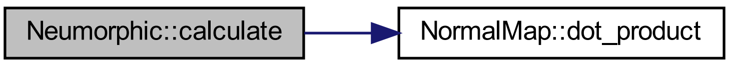
```
double Neumorphic::calculate ( )   [virtual]
```

Implements NormalMap.

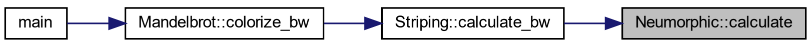
Definition at line 14 of file Neumorphic.cpp.

```
{
  u = z / dC;
  u = u / abs(u); // normalize
  reflection = dot_product(u, v) + heightFactor;
  reflection = reflection / (1.0 + heightFactor); // rescale so that it does not get bigger than 1
  if (reflection < 0.0)
  {
    reflection = 0.0;
  } else {}
  return reflection;
}
```

Here is the call graph for this function:



Here is the caller graph for this function:



5.5.3.2 get_angle()

```
double Neumorphic::get_angle ( )
```

Definition at line 37 of file Neumorphic.cpp.

```
{
  return angle;
}
```

5.5.3.3 get_heightFactor()

```
double Neumorphic::get_heightFactor ( )
```

Definition at line 32 of file Neumorphic.cpp.

```
{
  return heightFactor;
}
```

5.5.3.4 get_reflection()

```
double Neumorphic::get_reflection ( )
```

Definition at line 27 of file Neumorphic.cpp.

```
{
  return reflection;
}
```

5.5.4 Member Data Documentation

5.5.4.1 angle

```
double Neumorphic::angle  [private]
```

incoming direction of light WRT +x-axis (degrees) change first number

Definition at line 40 of file Neumorphic.h.

5.5.4.2 dC

```
std::complex<double> Neumorphic::dC  [private]
```

Definition at line 29 of file Neumorphic.h.

5.5.4.3 heightFactor

```
double Neumorphic::heightFactor  [private]
```

of pseudo incoming light vector

Definition at line 34 of file Neumorphic.h.

5.5.4.4 reflection

```
double Neumorphic::reflection  [private]
```

normalized normal vector

Definition at line 45 of file Neumorphic.h.

5.5.4.5 u

```
std::complex<double> Neumorphic::u [private]
```

Definition at line 47 of file Neumorphic.h.

5.5.4.6 v

```
std::complex<double> Neumorphic::v [private]
```

unit vector in direction of this.angle

Definition at line 52 of file Neumorphic.h.

5.5.4.7 z

```
std::complex<double> Neumorphic::z [private]
```

Definition at line 27 of file Neumorphic.h.

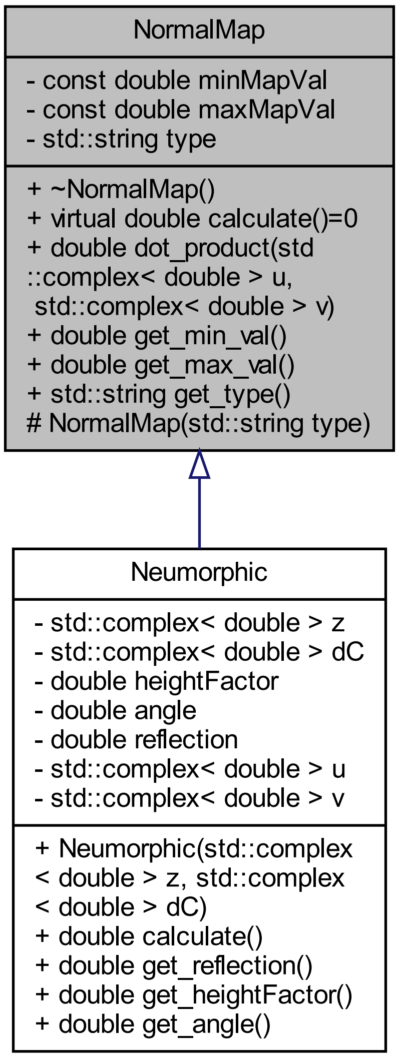
The documentation for this class was generated from the following files:

- D:/Documents/Final/Mandelbrot_Cpp_Final/src/Neumorphic.h
- D:/Documents/Final/Mandelbrot_Cpp_Final/src/Neumorphic.cpp

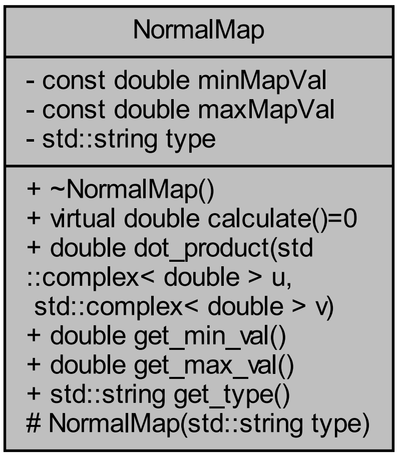
5.6 NormalMap Class Reference

```
#include <NormalMap.h>
```

Inheritance diagram for NormalMap:


NormalMap
- const double minMapVal
- const double maxMapVal
- std::string type
+ ~NormalMap()
+ virtual double calculate()=0
+ double dot_product(std::complex< double > u, std::complex< double > v)
+ double get_min_val()
+ double get_max_val()
+ std::string get_type()
# NormalMap(std::string type)
Neumorphic
- std::complex< double > z
- std::complex< double > dC
- double heightFactor
- double angle
- double reflection
- std::complex< double > u
- std::complex< double > v
+ Neumorphic(std::complex< double > z, std::complex< double > dC)
+ double calculate()
+ double get_reflection()
+ double get_heightFactor()
+ double get_angle()

Collaboration diagram for NormalMap:



Public Member Functions

- ∼NormalMap ()
- virtual double calculate ()=0
- double dot_product (std::complex< double > u, std::complex< double > v)
- double get_min_val ()
- double get_max_val ()
- std::string get_type ()

Protected Member Functions

- NormalMap (std::string type)

Private Attributes

- const double minMapVal = 0.0
- const double maxMapVal = 1.0
- std::string type

5.6.1 Detailed Description

Definition at line 8 of file NormalMap.h.

5.6.2 Constructor & Destructor Documentation

5.6.2.1 ∼NormalMap()

```
NormalMap::∼NormalMap ( )  [default]
```

5.6.2.2 NormalMap()

```
NormalMap::NormalMap (
            std::string type )  [explicit], [protected]
```

Definition at line 5 of file NormalMap.cpp.

```
{}
```

5.6.3 Member Function Documentation

5.6.3.1 calculate()

```
virtual double NormalMap::calculate ( )  [pure virtual]
```

Implemented in Neumorphic.

5.6.3.2 dot_product()

```
double NormalMap::dot_product (
            std::complex< double > u,
            std::complex< double > v )
```

Complex number dot product

Parameters

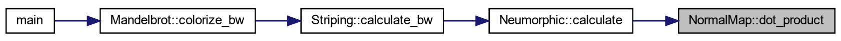
u	
v	

Returns

Definition at line 11 of file NormalMap.cpp.

```
{
  return real(u) * real(v) + imag(u) * imag(v);
}
```

Here is the caller graph for this function:



5.6.3.3 get_max_val()

```
double NormalMap::get_max_val ( )
```

Definition at line 21 of file NormalMap.cpp.

```
{
  return maxMapVal;
}
```

5.6.3.4 get_min_val()

```
double NormalMap::get_min_val ( )
```

Definition at line 16 of file NormalMap.cpp.

```
{
  return minMapVal;
}
```

5.6.3.5 get_type()

```
std::string NormalMap::get_type ( )
```

Definition at line 25 of file NormalMap.cpp.

```
{
  return type;
}
```

5.6.4 Member Data Documentation

5.6.4.1 maxMapVal

```
const double NormalMap::maxMapVal = 1.0  [private]
```

Definition at line 38 of file NormalMap.h.

5.6.4.2 minMapVal

```
const double NormalMap::minMapVal = 0.0  [private]
```

Definition at line 36 of file NormalMap.h.

5.6.4.3 type

```
std::string NormalMap::type  [private]
```

Definition at line 40 of file NormalMap.h.

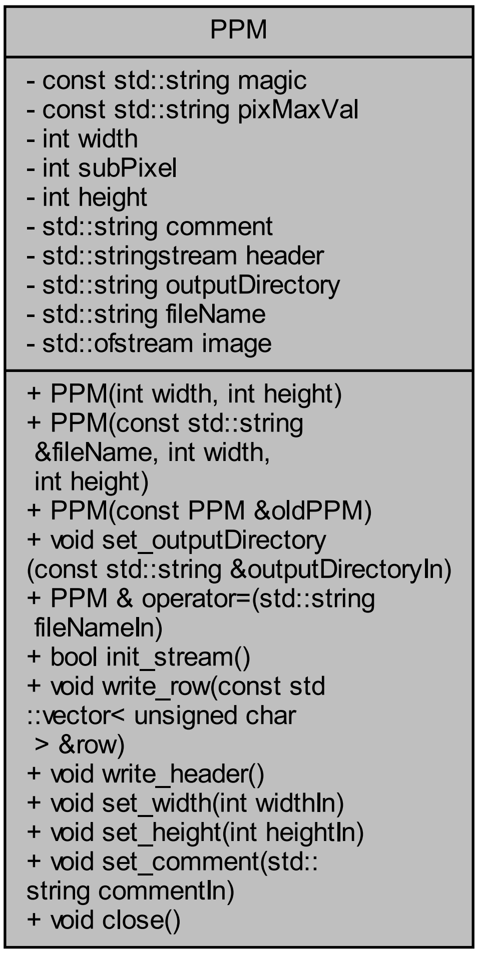
The documentation for this class was generated from the following files:

- D:/Documents/Final/Mandelbrot_Cpp_Final/src/NormalMap.h
- D:/Documents/Final/Mandelbrot_Cpp_Final/src/NormalMap.cpp

5.7 PPM Class Reference

```
#include <PPM.h>
```

Collaboration diagram for PPM:


PPM
- const std::string magic
- const std::string pixMaxVal
- int width
- int subPixel
- int height
- std::string comment
- std::stringstream header
- std::string outputDirectory
- std::string fileName
- std::ofstream image
+ PPM(int width, int height)
+ PPM(const std::string &fileName, int width, int height)
+ PPM(const PPM &oldPPM)
+ void set_outputDirectory (const std::string &outputDirectoryIn)
+ PPM & operator=(std::string fileNameIn)
+ bool init_stream()
+ void write_row(const std ::vector< unsigned char > &row)
+ void write_header()
+ void set_width(int widthIn)
+ void set_height(int heightIn)
+ void set_comment(std:: string commentIn)
+ void close()

Public Member Functions

- PPM (int width, int height)
- PPM (const std::string &fileName, int width, int height)
- PPM (const PPM &oldPPM)
- void set_outputDirectory (const std::string &outputDirectoryIn)
- PPM & operator= (std::string fileNameIn)
- bool init_stream ()
- void write_row (const std::vector$<$ unsigned char $>$ &row)
- void write_header ()
- void set_width (int widthIn)
- void set_height (int heightIn)
- void set_comment (std::string commentIn)
- void close ()

Private Attributes

- const std::string magic = "P6\n"
- const std::string pixMaxVal = "255\n"
- int width
- int subPixel
- int height
- std::string comment
- std::stringstream header
- std::string outputDirectory
- std::string fileName
- std::ofstream image

5.7.1 Detailed Description

Definition at line 11 of file PPM.h.

5.7.2 Constructor & Destructor Documentation

5.7.2.1 PPM() [1/3]

```
PPM::PPM (
            int width,
            int height )
```

Default parametrized constructor contains output directory and file name

Parameters

width	
height	

Definition at line 7 of file PPM.cpp.

```
                                 : width(width), height(height)
{
  outputDirectory = "..\\output\\";
  fileName = outputDirectory + "gigabrot_default.ppm";
  subPixel = width * 3;
}
```

5.7.2.2 PPM() [2/3]

```
PPM::PPM (
            const std::string & fileName,
            int width,
            int height )
```

Minimal parametrized constructor Only needs file name, contains output directory

Parameters

fileName	
width	
height	

Definition at line 14 of file PPM.cpp.

```
                                                          : width(width), height(height)
{
  outputDirectory = "..\\output\\";
  this->fileName = outputDirectory + fileName;
  subPixel = width * 3;
}
```

5.7.2.3 PPM() [3/3]

```
PPM::PPM (
               const PPM & oldPPM )
```

Copy constructor

Parameters

oldPPM	- dereferenced old object

Definition at line 21 of file PPM.cpp.

```
                              : magic(oldPPM.magic), pixMaxVal(oldPPM.pixMaxVal),
    width(oldPPM.width), height(oldPPM.height), subPixel(oldPPM.subPixel), comment(oldPPM.comment)
{}
```

5.7.3 Member Function Documentation

5.7.3.1 close()

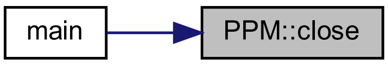
```
void PPM::close ( )
```

Close ofstream for image

Definition at line 72 of file PPM.cpp.

```
{
  image.close();
  cout « "File " « fileName « " saved\n";
}
```

Here is the caller graph for this function:



5.7.3.2 init_stream()

```
bool PPM::init_stream ( )
```

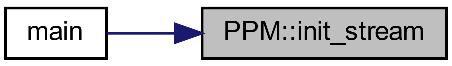
Initialize ofstream

Returns

Definition at line 36 of file PPM.cpp.

```
{
  image.open(fileName, ios::binary);

  if (image.is_open())
  {
    return true;
  } else
  {
    return false;
  }
}
```

Here is the caller graph for this function:



5.7.3.3 operator=()

```
PPM & PPM::operator= (
             std::string fileNameIn )
```

Overloaded assignment operator used to override output directory and file name

Parameters

file←NameIn	

Returns

Definition at line 30 of file PPM.cpp.

```
{
  this->fileName = std::move(fileNameIn);
  return *this;
}
```

5.7.3.4 set_comment()

```
void PPM::set_comment (
            std::string commentIn )
```

Set custom comment for PPM image

Parameters

comment↩ In	

Definition at line 67 of file PPM.cpp.

```
{
  comment = std::move(commentIn);
}
```

5.7.3.5 set_height()

```
void PPM::set_height (
            int heightIn )
```

Set height of PPM image

Parameters

height↩ In	

Definition at line 62 of file PPM.cpp.

```
{
  height = heightIn;
}
```

5.7.3.6 set_outputDirectory()

```
void PPM::set_outputDirectory (
            const std::string & outputDirectoryIn )
```

Sets output directory destination

Parameters

file↩NameIn	

Definition at line 25 of file PPM.cpp.

```
{
  outputDirectory = outputDirectoryIn;
}
```

5.7.3.7 set_width()

```
void PPM::set_width (
              int widthIn )
```

Set width of PPM image

Parameters

width↩In	

Definition at line 57 of file PPM.cpp.

```
{
  width = widthIn;
}
```

5.7.3.8 write_header()

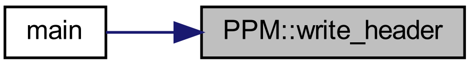
```
void PPM::write_header ( )
```

Print header data to PPM file

Definition at line 49 of file PPM.cpp.

```
{
  string widthStr = to_string(this->width);
  string lengthStr = to_string(this->height);
  header « magic « widthStr « " " « lengthStr « "\n" « comment « "\n" « pixMaxVal;
  image « header.rdbuf();
}
```

Here is the caller graph for this function:



5.7.3.9 write_row()

```
void PPM::write_row (
              const std::vector< unsigned char > & row )  [inline]
```

Print row of pixels to PPM file - templated for size of array

Template Parameters

N	- std::array size

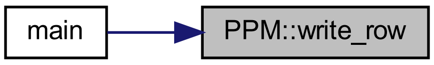
Parameters

row	- std::array of pixels

Definition at line 63 of file PPM.h.

```
{
  image.write((char const *) row.data(), row.size());
}
```

Here is the caller graph for this function:



5.7.4 Member Data Documentation

5.7.4.1 comment

```
std::string PPM::comment  [private]
```

Definition at line 112 of file PPM.h.

5.7.4.2 fileName

```
std::string PPM::fileName  [private]
```

Definition at line 121 of file PPM.h.

5.7.4.3 header

```
std::stringstream PPM::header [private]
```

Contains full header info

Definition at line 117 of file PPM.h.

5.7.4.4 height

```
int PPM::height [private]
```

Definition at line 110 of file PPM.h.

5.7.4.5 image

```
std::ofstream PPM::image [private]
```

Definition at line 123 of file PPM.h.

5.7.4.6 magic

```
const std::string PPM::magic = "P6\n" [private]
```

Determines type of PPM

Definition at line 102 of file PPM.h.

5.7.4.7 outputDirectory

```
std::string PPM::outputDirectory [private]
```

Definition at line 119 of file PPM.h.

5.7.4.8 pixMaxVal

```
const std::string PPM::pixMaxVal = "255\n" [private]
```

Definition at line 104 of file PPM.h.

5.7.4.9 subPixel

`int PPM::subPixel [private]`

Definition at line 108 of file PPM.h.

5.7.4.10 width

`int PPM::width [private]`

Definition at line 106 of file PPM.h.

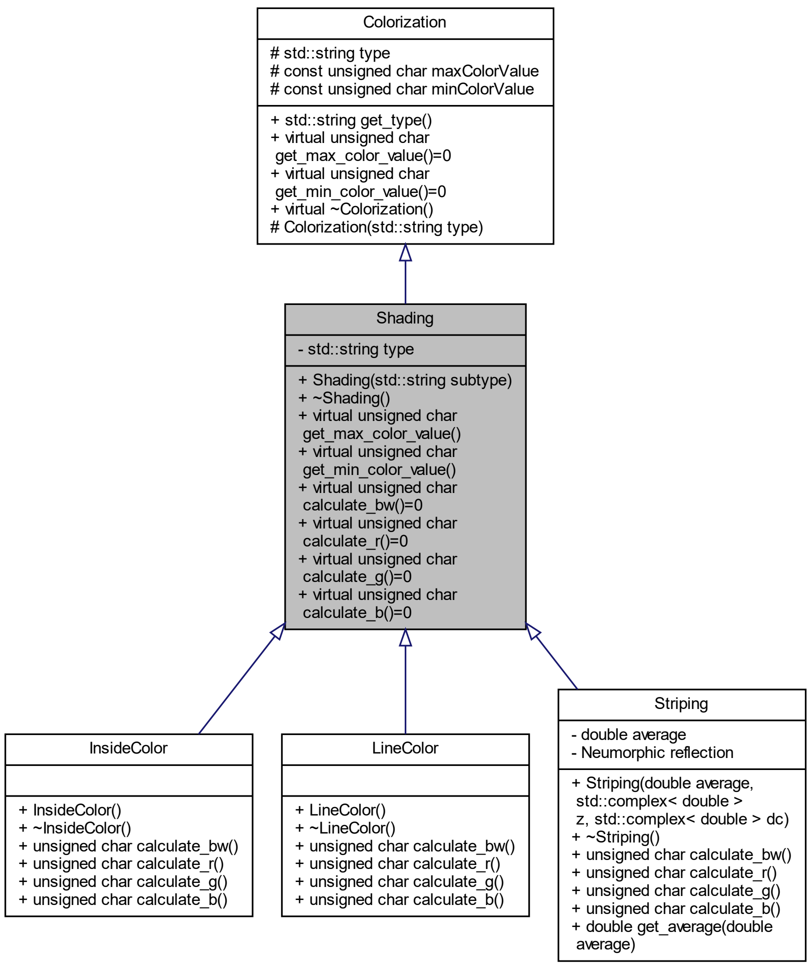
The documentation for this class was generated from the following files:

- D:/Documents/Final/Mandelbrot_Cpp_Final/src/PPM.h
- D:/Documents/Final/Mandelbrot_Cpp_Final/src/PPM.cpp

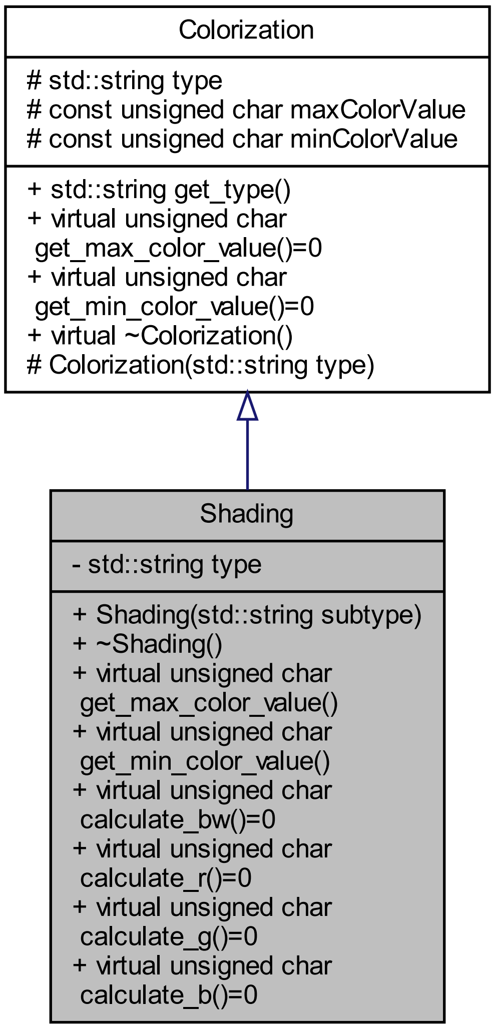
5.8 Shading Class Reference

`#include <Shading.h>`

Inheritance diagram for Shading:


Colorization
# std::string type
# const unsigned char maxColorValue
# const unsigned char minColorValue
+ std::string get_type()
+ virtual unsigned char get_max_color_value()=0
+ virtual unsigned char get_min_color_value()=0
+ virtual ~Colorization()
# Colorization(std::string type)
Shading
- std::string type
+ Shading(std::string subtype)
+ ~Shading()
+ virtual unsigned char get_max_color_value()
+ virtual unsigned char get_min_color_value()
+ virtual unsigned char calculate_bw()=0
+ virtual unsigned char calculate_r()=0
+ virtual unsigned char calculate_g()=0
+ virtual unsigned char calculate_b()=0
InsideColor
+ InsideColor()
+ ~InsideColor()
+ unsigned char calculate_bw()
+ unsigned char calculate_r()
+ unsigned char calculate_g()
+ unsigned char calculate_b()
LineColor
+ LineColor()
+ ~LineColor()
+ unsigned char calculate_bw()
+ unsigned char calculate_r()
+ unsigned char calculate_g()
+ unsigned char calculate_b()
Striping
- double average
- Neumorphic reflection
+ Striping(double average, std::complex< double > z, std::complex< double > dc)
+ ~Striping()
+ unsigned char calculate_bw()
+ unsigned char calculate_r()
+ unsigned char calculate_g()
+ unsigned char calculate_b()
+ double get_average(double average)

Collaboration diagram for Shading:


Colorization
# std::string type
# const unsigned char maxColorValue
# const unsigned char minColorValue
+ std::string get_type()
+ virtual unsigned char get_max_color_value()=0
+ virtual unsigned char get_min_color_value()=0
+ virtual ~Colorization()
# Colorization(std::string type)
Shading
- std::string type
+ Shading(std::string subtype)
+ ~Shading()
+ virtual unsigned char get_max_color_value()
+ virtual unsigned char get_min_color_value()
+ virtual unsigned char calculate_bw()=0
+ virtual unsigned char calculate_r()=0
+ virtual unsigned char calculate_g()=0
+ virtual unsigned char calculate_b()=0

Public Member Functions

- Shading (std::string subtype)
- ∼Shading ()
- virtual unsigned char get_max_color_value ()
- virtual unsigned char get_min_color_value ()
- virtual unsigned char calculate_bw ()=0
- virtual unsigned char calculate_r ()=0
- virtual unsigned char calculate_g ()=0
- virtual unsigned char calculate_b ()=0

Private Attributes

- std::string type = "Shading"

Additional Inherited Members

5.8.1 Detailed Description

Definition at line 6 of file Shading.h.

5.8.2 Constructor & Destructor Documentation

5.8.2.1 Shading()

```
Shading::Shading (
            std::string subtype )  [explicit]
```

Definition at line 5 of file Shading.cpp.

```
00005                               : Colorization(std::move(type))
00006 {}
```

5.8.2.2 ∼Shading()

```
Shading::∼Shading ( )  [default]
```

5.8.3 Member Function Documentation

5.8.3.1 calculate_b()

```
virtual unsigned char Shading::calculate_b ( )  [pure virtual]
```

Implemented in InsideColor, LineColor, and Striping.

5.8.3.2 calculate_bw()

```
virtual unsigned char Shading::calculate_bw ( )  [pure virtual]
```

Implemented in InsideColor, LineColor, and Striping.

5.8.3.3 calculate_g()

```
virtual unsigned char Shading::calculate_g ( )  [pure virtual]
```

Implemented in InsideColor, LineColor, and Striping.

5.8.3.4 calculate_r()

```
virtual unsigned char Shading::calculate_r ( )  [pure virtual]
```

Implemented in InsideColor, LineColor, and Striping.

5.8.3.5 get_max_color_value()

```
unsigned char Shading::get_max_color_value ( )  [virtual]
```

Implements Colorization.

Definition at line 11 of file Shading.cpp.

```
{
  return maxColorValue;
}
```

5.8.3.6 get_min_color_value()

```
unsigned char Shading::get_min_color_value ( )  [virtual]
```

Implements Colorization.

Definition at line 16 of file Shading.cpp.

```
{
  return minColorValue;
}
```

5.8.4 Member Data Documentation

5.8.4.1 type

```
std::string Shading::type = "Shading"  [private]
```

Definition at line 29 of file Shading.h.

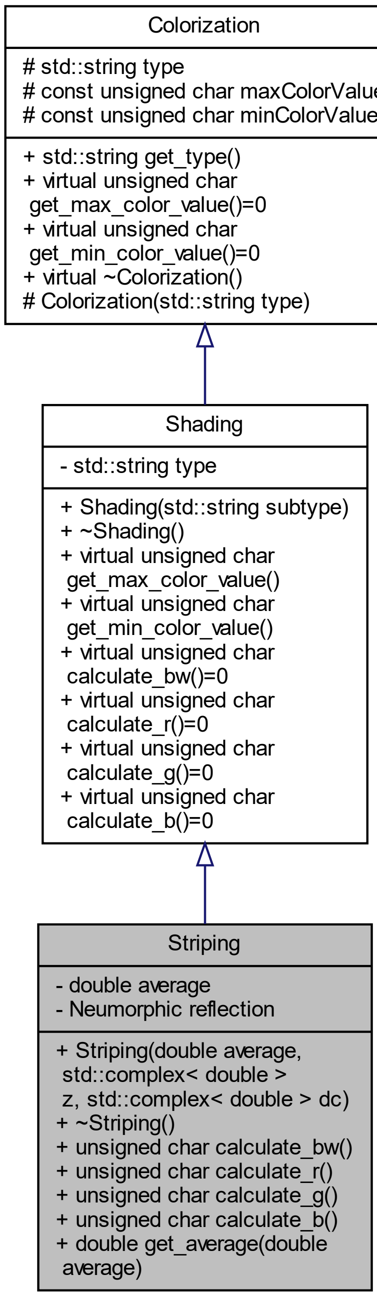
The documentation for this class was generated from the following files:

- D:/Documents/Final/Mandelbrot_Cpp_Final/src/Shading.h
- D:/Documents/Final/Mandelbrot_Cpp_Final/src/Shading.cpp

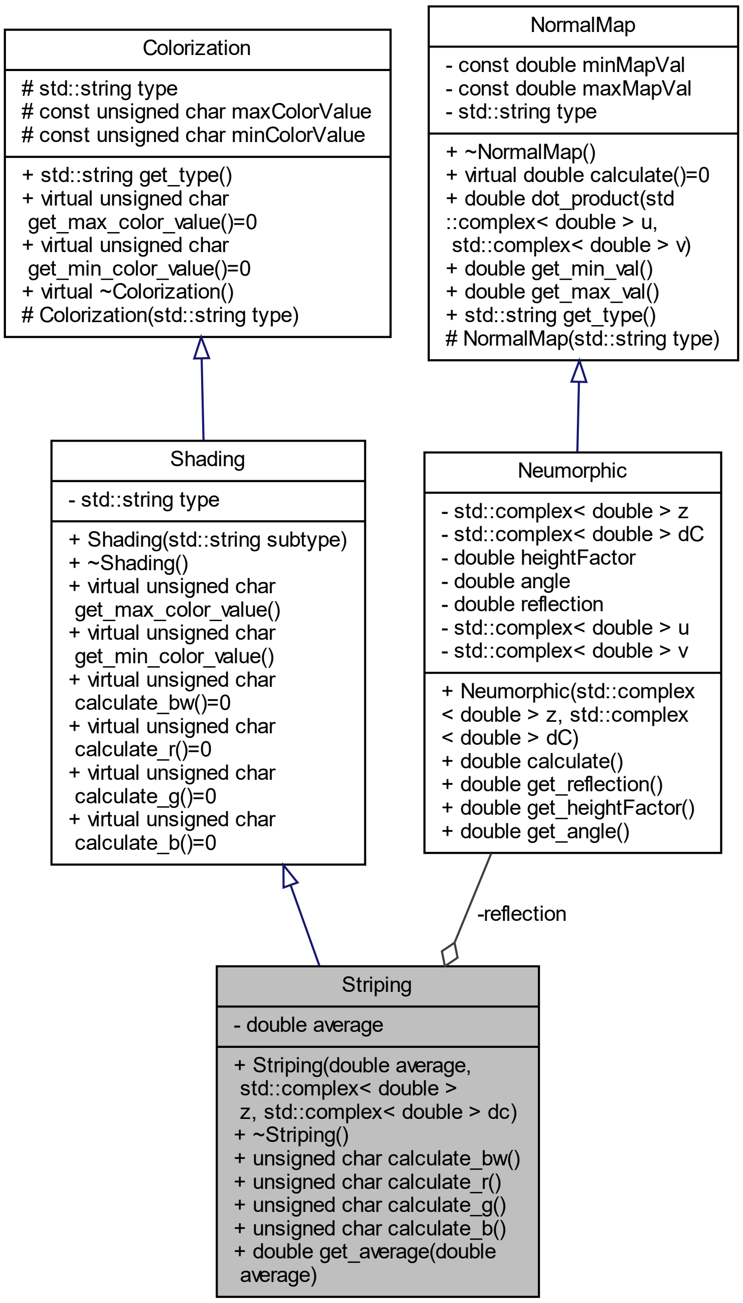
5.9 Striping Class Reference

```
#include <Striping.h>
```

Inheritance diagram for Striping:


Colorization
# std::string type
# const unsigned char maxColorValue
# const unsigned char minColorValue
+ std::string get_type()
+ virtual unsigned char get_max_color_value()=0
+ virtual unsigned char get_min_color_value()=0
+ virtual ~Colorization()
# Colorization(std::string type)
Shading
- std::string type
+ Shading(std::string subtype)
+ ~Shading()
+ virtual unsigned char get_max_color_value()
+ virtual unsigned char get_min_color_value()
+ virtual unsigned char calculate_bw()=0
+ virtual unsigned char calculate_r()=0
+ virtual unsigned char calculate_g()=0
+ virtual unsigned char calculate_b()=0
Striping
- double average
- Neumorphic reflection
+ Striping(double average, std::complex< double > z, std::complex< double > dc)
+ ~Striping()
+ unsigned char calculate_bw()
+ unsigned char calculate_r()
+ unsigned char calculate_g()
+ unsigned char calculate_b()
+ double get_average(double average)

Collaboration diagram for Striping:


Colorization
# std::string type
# const unsigned char maxColorValue
# const unsigned char minColorValue
+ std::string get_type()
+ virtual unsigned char get_max_color_value()=0
+ virtual unsigned char get_min_color_value()=0
+ virtual ~Colorization()
# Colorization(std::string type)
NormalMap
- const double minMapVal
- const double maxMapVal
- std::string type
+ ~NormalMap()
+ virtual double calculate()=0
+ double dot_product(std::complex< double > u, std::complex< double > v)
+ double get_min_val()
+ double get_max_val()
+ std::string get_type()
# NormalMap(std::string type)
Shading
- std::string type
+ Shading(std::string subtype)
+ ~Shading()
+ virtual unsigned char get_max_color_value()
+ virtual unsigned char get_min_color_value()
+ virtual unsigned char calculate_bw()=0
+ virtual unsigned char calculate_r()=0
+ virtual unsigned char calculate_g()=0
+ virtual unsigned char calculate_b()=0
Neumorphic
- std::complex< double > z
- std::complex< double > dC
- double heightFactor
- double angle
- double reflection
- std::complex< double > u
- std::complex< double > v
+ Neumorphic(std::complex < double > z, std::complex < double > dC)
+ double calculate()
+ double get_reflection()
+ double get_heightFactor()
+ double get_angle()
-reflection
Striping
- double average
+ Striping(double average, std::complex< double > z, std::complex< double > dc)
+ ~Striping()
+ unsigned char calculate_bw()
+ unsigned char calculate_r()
+ unsigned char calculate_g()
+ unsigned char calculate_b()
+ double get_average(double average)

Public Member Functions

- Striping (double average, std::complex< double > z, std::complex< double > dc)
- ∼Striping ()
- unsigned char calculate_bw ()
- unsigned char calculate_r ()
- unsigned char calculate_g ()

- unsigned char calculate_b ()
- double get_average (double average)

Private Attributes

- double average
- Neumorphic reflection

Additional Inherited Members

5.9.1 Detailed Description

Definition at line 7 of file Striping.h.

5.9.2 Constructor & Destructor Documentation

5.9.2.1 Striping()

```
Striping::Striping (
            double average,
            std::complex< double > z,
            std::complex< double > dc )
```

Definition at line 5 of file Striping.cpp.

```
00005                                                                   : Shading("Striping"),
00006 average(average), reflection(z, dc)
00007 {}
```

5.9.2.2 ∼Striping()

```
Striping::∼Striping ( )  [default]
```

5.9.3 Member Function Documentation

5.9.3.1 calculate_b()

```
unsigned char Striping::calculate_b ( )  [virtual]
```

Implements Shading.

Definition at line 28 of file Striping.cpp.

```
00029 {
00030   return 0;
00031 }
```

5.9.3.2 calculate_bw()

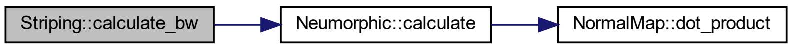
```
unsigned char Striping::calculate_bw ( )  [virtual]
```

Implements Shading.

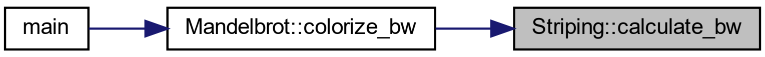
Definition at line 12 of file Striping.cpp.

```
{
  return static_cast<unsigned char>((static_cast<double>((maxColorValue - 1)) - (100.0 *
  average)) * reflection.calculate()); // explicit casting
}
```

Here is the call graph for this function:



Here is the caller graph for this function:



5.9.3.3 calculate_g()

```
unsigned char Striping::calculate_g ( )  [virtual]
```

Implements Shading.

Definition at line 23 of file Striping.cpp.

```
{
  return 0;
}
```

5.9.3.4 calculate_r()

```
unsigned char Striping::calculate_r ( )  [virtual]
```

Implements Shading.

Definition at line 18 of file Striping.cpp.

```
{
  return 0;
}
```

5.9.3.5 get_average()

```
double Striping::get_average (
            double average )
```

Definition at line 33 of file Striping.cpp.

```
00034 {
00035   return average;
00036 }
```

5.9.4 Member Data Documentation

5.9.4.1 average

```
double Striping::average  [private]
```

Definition at line 27 of file Striping.h.

5.9.4.2 reflection

```
Neumorphic Striping::reflection  [private]
```

Definition at line 29 of file Striping.h.

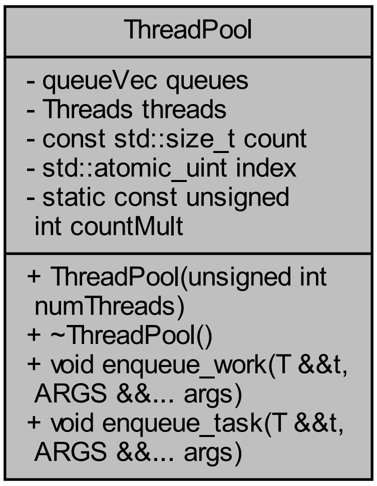
The documentation for this class was generated from the following files:

- D:/Documents/Final/Mandelbrot_Cpp_Final/src/Striping.h
- D:/Documents/Final/Mandelbrot_Cpp_Final/src/Striping.cpp

5.10 ThreadPool Class Reference

```
#include <ThreadPool.h>
```

Collaboration diagram for ThreadPool:


ThreadPool
- queueVec queues
- Threads threads
- const std::size_t count
- std::atomic_uint index
- static const unsigned int countMult
+ ThreadPool(unsigned int numThreads)
+ ~ThreadPool()
+ void enqueue_work(T &&t, ARGS &&... args)
+ void enqueue_task(T &&t, ARGS &&... args)

Public Member Functions

- ThreadPool (unsigned int numThreads)
- ∼ThreadPool ()
- template<typename T , typename... ARGS>
  void enqueue_work (T &&t, ARGS &&... args)
- template<typename T , typename... ARGS>
  void enqueue_task (T &&t, ARGS &&... args)

Private Types

- using process = std::function< void(void)>
- using queue = UnboundedQueue< process >
- using queueVec = std::vector< queue >
- using Threads = std::vector< std::thread >

Private Attributes

- queueVec queues
- Threads threads
- const std::size_t count
- std::atomic_uint index = 0

Static Private Attributes

- static const unsigned int countMult = 2

5.10.1 Detailed Description

Only using C++ intrinsics

Definition at line 16 of file ThreadPool.h.

5.10.2 Member Typedef Documentation

5.10.2.1 process

```
using ThreadPool::process = std::function<void(void)> [private]
```

Definition at line 70 of file ThreadPool.h.

5.10.2.2 queue

```
using ThreadPool::queue = UnboundedQueue<process> [private]
```

Definition at line 72 of file ThreadPool.h.

5.10.2.3 queueVec

```
using ThreadPool::queueVec = std::vector<queue> [private]
```

Definition at line 74 of file ThreadPool.h.

5.10.2.4 Threads

```
using ThreadPool::Threads = std::vector<std::thread> [private]
```

Definition at line 79 of file ThreadPool.h.

5.10.3 Constructor & Destructor Documentation

5.10.3.1 ThreadPool()

```
ThreadPool::ThreadPool (
              unsigned int numThreads )  [explicit]
```

Definition at line 3 of file ThreadPool.cpp.

```
                                              : queues(numThreads), count(numThreads)
{
  if (!numThreads)
  {
    throw std::invalid_argument("thread count must be nonzero!\n");
  } else if (numThreads < 0) {
    throw std::invalid_argument("thread count must be positive! how did this happen??");
  }

  auto worker = [this] (auto i) {
    while (true)
    {
      process proc;
      for (auto j = 0; j < count * countMult; j++)
      {
        if (queues[(i + j) % count].try_pop(proc))
        {
          break;
        }
      }
      if (!proc && !queues[i].pop(proc))
      {
        break;
      }
      proc();
    }
  };

  threads.reserve(numThreads);

  for (auto i = 0; i < numThreads; i++)
  {
    threads.emplace_back(worker, i);
  }
}
```

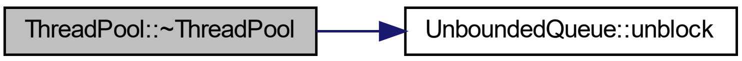
5.10.3.2 ∼ThreadPool()

```
ThreadPool::∼ThreadPool ( )
```

Definition at line 39 of file ThreadPool.cpp.

```
{
  for (auto& queue: queues)
  {
    queue.unblock();
  }
  for (auto& thread : threads)
  {
    thread.join();
  }
}
```

Here is the call graph for this function:



5.10.4 Member Function Documentation

5.10.4.1 enqueue_task()

```
template<typename T , typename... ARGS>
void ThreadPool::enqueue_task (
            T && t,
            ARGS &&... args ) [inline]
```

Definition at line 44 of file ThreadPool.h.

```
{
  using taskReturnType = std::invoke_result<T, ARGS...>;
  using taskType = std::packaged_task<taskReturnType()>;

  auto task = std::make_shared<taskType>(std::bind(std::forward<T>(t),
std::forward<ARGS>(args)...));
  auto work = [=] () {(*task)();};
  auto result = task->get_future();

  auto i = index++;

  for (auto j = 0; j < count * countMult; j++)
  {
    if (queues[(i + j) % count].try_push(work))
    {
      return result;
    }
  }

  queues[i % count].push(std::move(work));

  return result;
}
```

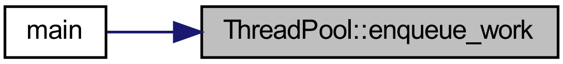
5.10.4.2 enqueue_work()

```
template<typename T , typename... ARGS>
void ThreadPool::enqueue_work (
            T && t,
            ARGS &&... args ) [inline]
```

Definition at line 25 of file ThreadPool.h.

```
{
  auto work = [proc = std::forward<T>(t), tuple = std::make_tuple(std::forward<ARGS>(args)...)]
      () {std::apply(proc, tuple);};

  auto i = index++;

  for (auto j = 0; j < count * countMult; j++)
  {
    if (queues[(i + j) % count].try_push(work))
    {
      return;
    }
  }

  queues[i % count].push(std::move(work));
}
```

Here is the caller graph for this function:



5.10.5 Member Data Documentation

5.10.5.1 count

```
const std::size_t ThreadPool::count  [private]
```

Definition at line 84 of file ThreadPool.h.

5.10.5.2 countMult

```
const unsigned int ThreadPool::countMult = 2  [inline], [static], [private]
```

Definition at line 88 of file ThreadPool.h.

5.10.5.3 index

```
std::atomic_uint ThreadPool::index = 0  [private]
```

Definition at line 86 of file ThreadPool.h.

5.10.5.4 queues

```
queueVec ThreadPool::queues  [private]
```

Definition at line 76 of file ThreadPool.h.

5.10.5.5 threads

```
Threads ThreadPool::threads  [private]
```

Definition at line 81 of file ThreadPool.h.

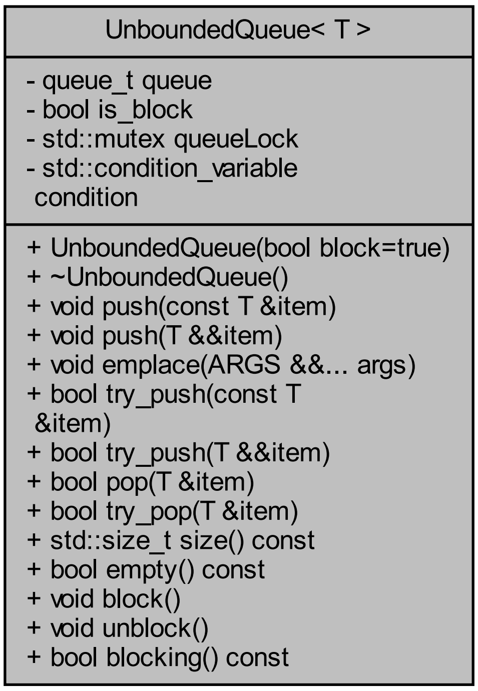
The documentation for this class was generated from the following files:

- D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPool.h
- D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPool.cpp

5.11 UnboundedQueue< T > Class Template Reference

```
#include <UnboundedQueue.h>
```

Collaboration diagram for UnboundedQueue< T >:



Public Member Functions

- UnboundedQueue (bool block=true)
- ∼UnboundedQueue ()
- void push (const T &item)
- void push (T &&item)
- template<typename... ARGS>
  void emplace (ARGS &&... args)
- bool try_push (const T &item)
- bool try_push (T &&item)
- bool pop (T &item)
- bool try_pop (T &item)
- std::size_t size () const
- bool empty () const
- void block ()
- void unblock ()
- bool blocking () const

Private Types

- using queue_t = std::queue< T >

Private Attributes

- queue_t queue
- bool is_block
- std::mutex queueLock
- std::condition_variable condition

5.11.1 Detailed Description

template<typename T>
class UnboundedQueue< T >

Definition at line 11 of file UnboundedQueue.h.

5.11.2 Member Typedef Documentation

5.11.2.1 queue_t

```
template<typename T >
using UnboundedQueue< T >::queue_t = std::queue<T>  [private]
```

Definition at line 65 of file UnboundedQueue.h.

5.11.3 Constructor & Destructor Documentation

5.11.3.1 UnboundedQueue()

```
template<typename T >
UnboundedQueue< T >::UnboundedQueue (
            bool block = true )  [explicit]
```

Default Parametrized constructor

Parameters

block	

Definition at line 77 of file UnboundedQueue.h.

```
00077                                            : is_block(block)
00078 {}
```

5.11.3.2 ∼UnboundedQueue()

```
template<typename T >
UnboundedQueue< T >::∼UnboundedQueue ( )  [default]
```

Default deconstructor

5.11.4 Member Function Documentation

5.11.4.1 block()

```
template<typename T >
void UnboundedQueue< T >::block
```

Definition at line 183 of file UnboundedQueue.h.

```
00184 {
00185   std::scoped_lock guard(queueLock);
00186   is_block = true;
00187 }
```

5.11.4.2 blocking()

```
template<typename T >
bool UnboundedQueue< T >::blocking
```

Definition at line 200 of file UnboundedQueue.h.

```
{
  std::scoped_lock guard(queueLock);
  return is_block;
}
```

5.11.4.3 emplace()

```
template<typename T >
template<typename...  ARGS>
void UnboundedQueue< T >::emplace (
             ARGS &&...  args )
```

Emplace

Template Parameters

Args	

Parameters

args	

Definition at line 102 of file UnboundedQueue.h.

```
{
  {
    std::scoped_lock guard(queueLock);
    queue.emplace(std::forward<>(args)...);
  }
  condition.notify_one();
}
```

5.11.4.4 empty()

```
template<typename T >
bool UnboundedQueue< T >::empty
```

Definition at line 176 of file UnboundedQueue.h.

```
{
  std::scoped_lock guard(queueLock);
  return queue.empty();
}
```

5.11.4.5 pop()

```
template<typename T >
bool UnboundedQueue< T >::pop (
              T & item )
```

Definition at line 142 of file UnboundedQueue.h.

```
{
  std::unique_lock guard(queueLock);
  condition.wait(guard, [&] () {return !queue.empty() || !is_block;});
  if (queue.empty())
  {
    return false;
  }
  item = std::move(queue.front());
  queue.pop();
  return true;
}
```

5.11.4.6 push() [1/2]

```
template<typename T >
void UnboundedQueue< T >::push (
              const T & item )
```

Parameters

item	- const reference

Definition at line 81 of file UnboundedQueue.h.

```
{
  {
    std::scoped_lock guard(queueLock);
    queue.push(item);
  }
  condition.notify_one();
}
```

5.11.4.7 push() [2/2]

```
template<typename T >
void UnboundedQueue< T >::push (
              T && item )
```

Parameters

item	- double reference

Definition at line 91 of file UnboundedQueue.h.

```
{
  {
    std::scoped_lock guard(queueLock);
    queue.push(std::move(item));
  }
  condition.notify_one();
}
```

5.11.4.8 size()

```
template<typename T >
std::size_t UnboundedQueue< T >::size
```

Definition at line 169 of file UnboundedQueue.h.

```
{
  std::scoped_lock guard(queueLock);
  return queue.size();
}
```

5.11.4.9 try_pop()

```
template<typename T >
bool UnboundedQueue< T >::try_pop (
              T & item )
```

Definition at line 156 of file UnboundedQueue.h.

```
{
  std::unique_lock guard(queueLock, std::try_to_lock);
  if (!guard || queue.empty())
  {
    return false;
  }
  item = std::move(queue.front());
  queue.pop();
  return true;
}
```

5.11.4.10 try_push() [1/2]

```
template<typename T >
bool UnboundedQueue< T >::try_push (
              const T & item )
```

Definition at line 112 of file UnboundedQueue.h.

```
{
  {
    std::unique_lock guard(queueLock, std::try_to_lock);
    if (!guard)
    {
      return false;
    }
    queue.push(item);
  }
  condition.notify_one();
  return true;
}
```

5.11.4.11 try_push() [2/2]

```
template<typename T >
bool UnboundedQueue< T >::try_push (
            T && item )
```

Definition at line 127 of file UnboundedQueue.h.

```
{
  {
    std::unique_lock guard(queueLock, std::try_to_lock);
    if (!guard)
    {
      return false;
    }
    queue.push(std::move(item));
  }
  condition.notify_one();
  return true;
}
```

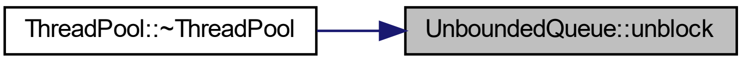
5.11.4.12 unblock()

```
template<typename T >
void UnboundedQueue< T >::unblock
```

Definition at line 190 of file UnboundedQueue.h.

```
{
  {
    std::scoped_lock guard(queueLock);
    is_block = false;
  }
  condition.notify_all();
}
```

Here is the caller graph for this function:



5.11.5 Member Data Documentation

5.11.5.1 condition

```
template<typename T >
std::condition_variable UnboundedQueue< T >::condition  [private]
```

Definition at line 72 of file UnboundedQueue.h.

5.11.5.2 is_block

```
template<typename T >
bool UnboundedQueue< T >::is_block  [private]
```

Definition at line 68 of file UnboundedQueue.h.

5.11.5.3 queue

```
template<typename T >
queue_t UnboundedQueue< T >::queue  [private]
```

Definition at line 66 of file UnboundedQueue.h.

5.11.5.4 queueLock

```
template<typename T >
std::mutex UnboundedQueue< T >::queueLock  [mutable], [private]
```

Definition at line 70 of file UnboundedQueue.h.

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

- D:/Documents/Final/Mandelbrot_Cpp_Final/src/UnboundedQueue.h

Chapter 6

File Documentation

6.1 D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/↩ CMakeFiles/3.21.1/CompilerIdC/CMakeCCompilerId.c File Reference

Macros

- #define __has_include(x) 0
- #define COMPILER_ID ""
- #define STRINGIFY_HELPER(X) #X
- #define STRINGIFY(X) STRINGIFY_HELPER(X)
- #define PLATFORM_ID
- #define ARCHITECTURE_ID
- #define DEC(n)
- #define HEX(n)
- #define C_DIALECT

Functions

- int main (int argc, char ∗argv[ ])

Variables

- char const ∗ info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
- char const ∗ info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
- char const ∗ info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
- const char ∗ info_language_dialect_default

6.1.1 Macro Definition Documentation

6.1.1.1 __has_include

```
#define __has_include(
              x ) 0
```

Definition at line 17 of file CMakeCCompilerId.c.

6.1.1.2 ARCHITECTURE_ID

```
#define ARCHITECTURE_ID
```

Definition at line 668 of file CMakeCCompilerId.c.

6.1.1.3 C_DIALECT

```
#define C_DIALECT
```

Definition at line 757 of file CMakeCCompilerId.c.

6.1.1.4 COMPILER_ID

```
#define COMPILER_ID ""
```

Definition at line 412 of file CMakeCCompilerId.c.

6.1.1.5 DEC

```
#define DEC(
              n )
```

Value:

```
  ('0' + (((n) / 10000000)%10)), \
  ('0' + (((n) / 1000000)%10)),  \
  ('0' + (((n) / 100000)%10)),   \
  ('0' + (((n) / 10000)%10)),    \
  ('0' + (((n) / 1000)%10)),     \
  ('0' + (((n) / 100)%10)),      \
  ('0' + (((n) / 10)%10)),       \
  ('0' +  ((n) % 10))
```

Definition at line 672 of file CMakeCCompilerId.c.

6.1.1.6 HEX

```
#define HEX(
              n )
```

Value:

```
('0' + ((n)»28 & 0xF)), \
('0' + ((n)»24 & 0xF)), \
('0' + ((n)»20 & 0xF)), \
('0' + ((n)»16 & 0xF)), \
('0' + ((n)»12 & 0xF)), \
('0' + ((n)»8  & 0xF)), \
('0' + ((n)»4  & 0xF)), \
('0' + ((n)     & 0xF))
```

Definition at line 683 of file CMakeCCompilerId.c.

6.1.1.7 PLATFORM_ID

```
#define PLATFORM_ID
```

Definition at line 540 of file CMakeCCompilerId.c.

6.1.1.8 STRINGIFY

```
#define STRINGIFY(
              X ) STRINGIFY_HELPER(X)
```

Definition at line 433 of file CMakeCCompilerId.c.

6.1.1.9 STRINGIFY_HELPER

```
#define STRINGIFY_HELPER(
              X ) #X
```

Definition at line 432 of file CMakeCCompilerId.c.

6.1.2 Function Documentation

6.1.2.1 main()

```
int main (
            int argc,
            char * argv[ ] )
```

Definition at line 781 of file CMakeCCompilerId.c.

```
{
  int require = 0;
  require += info_compiler[argc];
  require += info_platform[argc];
  require += info_arch[argc];
#ifdef COMPILER_VERSION_MAJOR
  require += info_version[argc];
#endif
#ifdef COMPILER_VERSION_INTERNAL
  require += info_version_internal[argc];
#endif
#ifdef SIMULATE_ID
  require += info_simulate[argc];
#endif
#ifdef SIMULATE_VERSION_MAJOR
  require += info_simulate_version[argc];
#endif
#if defined(__CRAYXT_COMPUTE_LINUX_TARGET)
  require += info_cray[argc];
#endif
  require += info_language_dialect_default[argc];
  (void)argv;
  return require;
}
```

6.1.3 Variable Documentation

6.1.3.1 info_arch

```
char const* info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
```

Definition at line 749 of file CMakeCCompilerId.c.

6.1.3.2 info_compiler

```
char const* info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
```

Definition at line 419 of file CMakeCCompilerId.c.

6.1.3.3 info_language_dialect_default

```
const char* info_language_dialect_default
```

Initial value:

```
=
  "INFO" ":" "dialect_default[" C_DIALECT "]"
```

Definition at line 770 of file CMakeCCompilerId.c.

6.1.3.4 info_platform

```
char const* info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
```

Definition at line 748 of file CMakeCCompilerId.c.

6.2 CMakeCCompilerId.c

Go to the documentation of this file.

```
#ifdef __cplusplus
# error "A C++ compiler has been selected for C."
#endif

#if defined(__18CXX)
# define ID_VOID_MAIN
#endif
#if defined(__CLASSIC_C__)
/* cv-qualifiers did not exist in K&R C */
# define const
# define volatile
#endif

#if !defined(__has_include)
/* If the compiler does not have __has_include, pretend the answer is
   always no.  */
#  define __has_include(x) 0
#endif


/* Version number components: V=Version, R=Revision, P=Patch
   Version date components:   YYYY=Year, MM=Month,   DD=Day  */

#if defined(__INTEL_COMPILER) || defined(__ICC)
# define COMPILER_ID "Intel"
# if defined(_MSC_VER)
#  define SIMULATE_ID "MSVC"
# endif
# if defined(__GNUC__)
#  define SIMULATE_ID "GNU"
# endif
  /* __INTEL_COMPILER = VRP prior to 2021, and then VVVV for 2021 and later,
     except that a few beta releases use the old format with V=2021.  */
# if __INTEL_COMPILER < 2021 || __INTEL_COMPILER == 202110 || __INTEL_COMPILER == 202111
#  define COMPILER_VERSION_MAJOR DEC(__INTEL_COMPILER/100)
#  define COMPILER_VERSION_MINOR DEC(__INTEL_COMPILER/10 % 10)
#  if defined(__INTEL_COMPILER_UPDATE)
#   define COMPILER_VERSION_PATCH DEC(__INTEL_COMPILER_UPDATE)
#  else
#   define COMPILER_VERSION_PATCH DEC(__INTEL_COMPILER   % 10)
#  endif
# else
#  define COMPILER_VERSION_MAJOR DEC(__INTEL_COMPILER)
#  define COMPILER_VERSION_MINOR DEC(__INTEL_COMPILER_UPDATE)
   /* The third version component from --version is an update index,
      but no macro is provided for it.  */
#  define COMPILER_VERSION_PATCH DEC(0)
# endif
# if defined(__INTEL_COMPILER_BUILD_DATE)
   /* __INTEL_COMPILER_BUILD_DATE = YYYYMMDD */
#  define COMPILER_VERSION_TWEAK DEC(__INTEL_COMPILER_BUILD_DATE)
# endif
# if defined(_MSC_VER)
   /* _MSC_VER = VVRR */
#  define SIMULATE_VERSION_MAJOR DEC(_MSC_VER / 100)
#  define SIMULATE_VERSION_MINOR DEC(_MSC_VER % 100)
# endif
# if defined(__GNUC__)
#  define SIMULATE_VERSION_MAJOR DEC(__GNUC__)
# elif defined(__GNUG__)
#  define SIMULATE_VERSION_MAJOR DEC(__GNUG__)
# endif
# if defined(__GNUC_MINOR__)
#  define SIMULATE_VERSION_MINOR DEC(__GNUC_MINOR__)
# endif
# if defined(__GNUC_PATCHLEVEL__)
#  define SIMULATE_VERSION_PATCH DEC(__GNUC_PATCHLEVEL__)
# endif

```

```
#elif (defined(__clang__) && defined(__INTEL_CLANG_COMPILER)) || defined(__INTEL_LLVM_COMPILER)
# define COMPILER_ID "IntelLLVM"
#if defined(_MSC_VER)
# define SIMULATE_ID "MSVC"
#endif
#if defined(__GNUC__)
# define SIMULATE_ID "GNU"
#endif
/* __INTEL_LLVM_COMPILER = VVVVRP prior to 2021.2.0, VVVVRRPP for 2021.2.0 and
 * later.  Look for 6 digit vs. 8 digit version number to decide encoding.
 * VVVV is no smaller than the current year when a version is released.
 */
#if __INTEL_LLVM_COMPILER < 1000000L
# define COMPILER_VERSION_MAJOR DEC(__INTEL_LLVM_COMPILER/100)
# define COMPILER_VERSION_MINOR DEC(__INTEL_LLVM_COMPILER/10 % 10)
# define COMPILER_VERSION_PATCH DEC(__INTEL_LLVM_COMPILER    % 10)
#else
# define COMPILER_VERSION_MAJOR DEC(__INTEL_LLVM_COMPILER/10000)
# define COMPILER_VERSION_MINOR DEC(__INTEL_LLVM_COMPILER/100 % 100)
# define COMPILER_VERSION_PATCH DEC(__INTEL_LLVM_COMPILER     % 100)
#endif
#if defined(_MSC_VER)
  /* _MSC_VER = VVRR */
# define SIMULATE_VERSION_MAJOR DEC(_MSC_VER / 100)
# define SIMULATE_VERSION_MINOR DEC(_MSC_VER % 100)
#endif
#if defined(__GNUC__)
# define SIMULATE_VERSION_MAJOR DEC(__GNUC__)
#elif defined(__GNUG__)
# define SIMULATE_VERSION_MAJOR DEC(__GNUG__)
#endif
#if defined(__GNUC_MINOR__)
# define SIMULATE_VERSION_MINOR DEC(__GNUC_MINOR__)
#endif
#if defined(__GNUC_PATCHLEVEL__)
# define SIMULATE_VERSION_PATCH DEC(__GNUC_PATCHLEVEL__)
#endif

#elif defined(__PATHCC__)
# define COMPILER_ID "PathScale"
# define COMPILER_VERSION_MAJOR DEC(__PATHCC__)
# define COMPILER_VERSION_MINOR DEC(__PATHCC_MINOR__)
# if defined(__PATHCC_PATCHLEVEL__)
#  define COMPILER_VERSION_PATCH DEC(__PATHCC_PATCHLEVEL__)
# endif

#elif defined(__BORLANDC__) && defined(__CODEGEARC_VERSION__)
# define COMPILER_ID "Embarcadero"
# define COMPILER_VERSION_MAJOR HEX(__CODEGEARC_VERSION__»24 & 0x00FF)
# define COMPILER_VERSION_MINOR HEX(__CODEGEARC_VERSION__»16 & 0x00FF)
# define COMPILER_VERSION_PATCH DEC(__CODEGEARC_VERSION__     & 0xFFFF)

#elif defined(__BORLANDC__)
# define COMPILER_ID "Borland"
  /* __BORLANDC__ = 0xVRR */
# define COMPILER_VERSION_MAJOR HEX(__BORLANDC__»8)
# define COMPILER_VERSION_MINOR HEX(__BORLANDC__ & 0xFF)

#elif defined(__WATCOMC__) && __WATCOMC__ < 1200
# define COMPILER_ID "Watcom"
   /* __WATCOMC__ = VVRR */
# define COMPILER_VERSION_MAJOR DEC(__WATCOMC__ / 100)
# define COMPILER_VERSION_MINOR DEC((__WATCOMC__ / 10) % 10)
# if (__WATCOMC__ % 10) > 0
#  define COMPILER_VERSION_PATCH DEC(__WATCOMC__ % 10)
# endif

#elif defined(__WATCOMC__)
# define COMPILER_ID "OpenWatcom"
   /* __WATCOMC__ = VVRP + 1100 */
# define COMPILER_VERSION_MAJOR DEC((__WATCOMC__ - 1100) / 100)
# define COMPILER_VERSION_MINOR DEC((__WATCOMC__ / 10) % 10)
# if (__WATCOMC__ % 10) > 0
#  define COMPILER_VERSION_PATCH DEC(__WATCOMC__ % 10)
# endif

#elif defined(__SUNPRO_C)
# define COMPILER_ID "SunPro"
# if __SUNPRO_C >= 0x5100
   /* __SUNPRO_C = 0xVRRP */
#  define COMPILER_VERSION_MAJOR HEX(__SUNPRO_C»12)
#  define COMPILER_VERSION_MINOR HEX(__SUNPRO_C»4 & 0xFF)
#  define COMPILER_VERSION_PATCH HEX(__SUNPRO_C    & 0xF)
# else
   /* __SUNPRO_CC = 0xVRP */
#  define COMPILER_VERSION_MAJOR HEX(__SUNPRO_C»8)
#  define COMPILER_VERSION_MINOR HEX(__SUNPRO_C»4 & 0xF)
#  define COMPILER_VERSION_PATCH HEX(__SUNPRO_C    & 0xF)
# endif

#elif defined(__HP_cc)
# define COMPILER_ID "HP"
  /* __HP_cc = VVRRPP */
# define COMPILER_VERSION_MAJOR DEC(__HP_cc/10000)
# define COMPILER_VERSION_MINOR DEC(__HP_cc/100 % 100)
# define COMPILER_VERSION_PATCH DEC(__HP_cc     % 100)

#elif defined(__DECC)
# define COMPILER_ID "Compaq"
  /* __DECC_VER = VVRRTPPPP */
# define COMPILER_VERSION_MAJOR DEC(__DECC_VER/10000000)
# define COMPILER_VERSION_MINOR DEC(__DECC_VER/100000  % 100)
# define COMPILER_VERSION_PATCH DEC(__DECC_VER         % 10000)

#elif defined(__IBMC__) && defined(__COMPILER_VER__)
# define COMPILER_ID "zOS"
  /* __IBMC__ = VRP */
# define COMPILER_VERSION_MAJOR DEC(__IBMC__/100)
# define COMPILER_VERSION_MINOR DEC(__IBMC__/10 % 10)
# define COMPILER_VERSION_PATCH DEC(__IBMC__    % 10)

#elif defined(__ibmxl__) && defined(__clang__)
# define COMPILER_ID "XLClang"
# define COMPILER_VERSION_MAJOR DEC(__ibmxl_version__)
# define COMPILER_VERSION_MINOR DEC(__ibmxl_release__)
# define COMPILER_VERSION_PATCH DEC(__ibmxl_modification__)
# define COMPILER_VERSION_TWEAK DEC(__ibmxl_ptf_fix_level__)


#elif defined(__IBMC__) && !defined(__COMPILER_VER__) && __IBMC__ >= 800
# define COMPILER_ID "XL"
  /* __IBMC__ = VRP */
# define COMPILER_VERSION_MAJOR DEC(__IBMC__/100)
# define COMPILER_VERSION_MINOR DEC(__IBMC__/10 % 10)
# define COMPILER_VERSION_PATCH DEC(__IBMC__    % 10)

#elif defined(__IBMC__) && !defined(__COMPILER_VER__) && __IBMC__ < 800
# define COMPILER_ID "VisualAge"
  /* __IBMC__ = VRP */
# define COMPILER_VERSION_MAJOR DEC(__IBMC__/100)
# define COMPILER_VERSION_MINOR DEC(__IBMC__/10 % 10)
# define COMPILER_VERSION_PATCH DEC(__IBMC__    % 10)

#elif defined(__NVCOMPILER)
# define COMPILER_ID "NVHPC"
# define COMPILER_VERSION_MAJOR DEC(__NVCOMPILER_MAJOR__)
# define COMPILER_VERSION_MINOR DEC(__NVCOMPILER_MINOR__)
# if defined(__NVCOMPILER_PATCHLEVEL__)
#  define COMPILER_VERSION_PATCH DEC(__NVCOMPILER_PATCHLEVEL__)
# endif

#elif defined(__PGI)
# define COMPILER_ID "PGI"
# define COMPILER_VERSION_MAJOR DEC(__PGIC__)
# define COMPILER_VERSION_MINOR DEC(__PGIC_MINOR__)
# if defined(__PGIC_PATCHLEVEL__)
#  define COMPILER_VERSION_PATCH DEC(__PGIC_PATCHLEVEL__)
# endif

#elif defined(_CRAYC)
# define COMPILER_ID "Cray"
# define COMPILER_VERSION_MAJOR DEC(_RELEASE_MAJOR)
# define COMPILER_VERSION_MINOR DEC(_RELEASE_MINOR)

#elif defined(__TI_COMPILER_VERSION__)
# define COMPILER_ID "TI"
  /* __TI_COMPILER_VERSION__ = VVVRRRPPP */
# define COMPILER_VERSION_MAJOR DEC(__TI_COMPILER_VERSION__/1000000)
# define COMPILER_VERSION_MINOR DEC(__TI_COMPILER_VERSION__/1000   % 1000)
# define COMPILER_VERSION_PATCH DEC(__TI_COMPILER_VERSION__        % 1000)

#elif defined(__CLANG_FUJITSU)
# define COMPILER_ID "FujitsuClang"
# define COMPILER_VERSION_MAJOR DEC(__FCC_major__)
# define COMPILER_VERSION_MINOR DEC(__FCC_minor__)
# define COMPILER_VERSION_PATCH DEC(__FCC_patchlevel__)
# define COMPILER_VERSION_INTERNAL_STR __clang_version__


#elif defined(__FUJITSU)
# define COMPILER_ID "Fujitsu"
# if defined(__FCC_version__)
#   define COMPILER_VERSION __FCC_version__
# elif defined(__FCC_major__)
#   define COMPILER_VERSION_MAJOR DEC(__FCC_major__)
#   define COMPILER_VERSION_MINOR DEC(__FCC_minor__)
#   define COMPILER_VERSION_PATCH DEC(__FCC_patchlevel__)
# endif
# if defined(__fcc_version)
#   define COMPILER_VERSION_INTERNAL DEC(__fcc_version)
# elif defined(__FCC_VERSION)
#   define COMPILER_VERSION_INTERNAL DEC(__FCC_VERSION)
# endif


#elif defined(__ghs__)
# define COMPILER_ID "GHS"
/* __GHS_VERSION_NUMBER = VVVVRP */
# ifdef __GHS_VERSION_NUMBER
# define COMPILER_VERSION_MAJOR DEC(__GHS_VERSION_NUMBER / 100)
# define COMPILER_VERSION_MINOR DEC(__GHS_VERSION_NUMBER / 10 % 10)
# define COMPILER_VERSION_PATCH DEC(__GHS_VERSION_NUMBER      % 10)
# endif

#elif defined(__TINYC__)
# define COMPILER_ID "TinyCC"

#elif defined(__BCC__)
# define COMPILER_ID "Bruce"

#elif defined(__SCO_VERSION__)
# define COMPILER_ID "SCO"

#elif defined(__ARMCC_VERSION) && !defined(__clang__)
# define COMPILER_ID "ARMCC"
#if __ARMCC_VERSION >= 1000000
  /* __ARMCC_VERSION = VRRPPPP */
  # define COMPILER_VERSION_MAJOR DEC(__ARMCC_VERSION/1000000)
  # define COMPILER_VERSION_MINOR DEC(__ARMCC_VERSION/10000 % 100)
  # define COMPILER_VERSION_PATCH DEC(__ARMCC_VERSION     % 10000)
#else
  /* __ARMCC_VERSION = VRPPPP */
  # define COMPILER_VERSION_MAJOR DEC(__ARMCC_VERSION/100000)
  # define COMPILER_VERSION_MINOR DEC(__ARMCC_VERSION/10000 % 10)
  # define COMPILER_VERSION_PATCH DEC(__ARMCC_VERSION    % 10000)
#endif


#elif defined(__clang__) && defined(__apple_build_version__)
# define COMPILER_ID "AppleClang"
# if defined(_MSC_VER)
#  define SIMULATE_ID "MSVC"
# endif
# define COMPILER_VERSION_MAJOR DEC(__clang_major__)
# define COMPILER_VERSION_MINOR DEC(__clang_minor__)
# define COMPILER_VERSION_PATCH DEC(__clang_patchlevel__)
# if defined(_MSC_VER)
   /* _MSC_VER = VVRR */
#  define SIMULATE_VERSION_MAJOR DEC(_MSC_VER / 100)
#  define SIMULATE_VERSION_MINOR DEC(_MSC_VER % 100)
# endif
# define COMPILER_VERSION_TWEAK DEC(__apple_build_version__)

#elif defined(__clang__) && defined(__ARMCOMPILER_VERSION)
# define COMPILER_ID "ARMClang"
  # define COMPILER_VERSION_MAJOR DEC(__ARMCOMPILER_VERSION/1000000)
  # define COMPILER_VERSION_MINOR DEC(__ARMCOMPILER_VERSION/10000 % 100)
  # define COMPILER_VERSION_PATCH DEC(__ARMCOMPILER_VERSION     % 10000)
# define COMPILER_VERSION_INTERNAL DEC(__ARMCOMPILER_VERSION)

#elif defined(__clang__) && __has_include(<hip/hip_version.h>)
# define COMPILER_ID "ROCMClang"
# if defined(_MSC_VER)
#  define SIMULATE_ID "MSVC"
# elif defined(__clang__)
#  define SIMULATE_ID "Clang"
# elif defined(__GNUC__)
#  define SIMULATE_ID "GNU"
# endif
# if defined(__clang__) && __has_include(<hip/hip_version.h>)
#  include <hip/hip_version.h>
#  define COMPILER_VERSION_MAJOR DEC(HIP_VERSION_MAJOR)
#  define COMPILER_VERSION_MINOR DEC(HIP_VERSION_MINOR)
#  define COMPILER_VERSION_PATCH DEC(HIP_VERSION_PATCH)
# endif

#elif defined(__clang__)
# define COMPILER_ID "Clang"
# if defined(_MSC_VER)
#  define SIMULATE_ID "MSVC"
# endif
# define COMPILER_VERSION_MAJOR DEC(__clang_major__)
# define COMPILER_VERSION_MINOR DEC(__clang_minor__)
# define COMPILER_VERSION_PATCH DEC(__clang_patchlevel__)
# if defined(_MSC_VER)
   /* _MSC_VER = VVRR */
#  define SIMULATE_VERSION_MAJOR DEC(_MSC_VER / 100)
#  define SIMULATE_VERSION_MINOR DEC(_MSC_VER % 100)
# endif

#elif defined(__GNUC__)
# define COMPILER_ID "GNU"
# define COMPILER_VERSION_MAJOR DEC(__GNUC__)
# if defined(__GNUC_MINOR__)
#  define COMPILER_VERSION_MINOR DEC(__GNUC_MINOR__)
# endif
# if defined(__GNUC_PATCHLEVEL__)
#  define COMPILER_VERSION_PATCH DEC(__GNUC_PATCHLEVEL__)
# endif

#elif defined(_MSC_VER)
# define COMPILER_ID "MSVC"
  /* _MSC_VER = VVRR */
# define COMPILER_VERSION_MAJOR DEC(_MSC_VER / 100)
# define COMPILER_VERSION_MINOR DEC(_MSC_VER % 100)
# if defined(_MSC_FULL_VER)
#  if _MSC_VER >= 1400
    /* _MSC_FULL_VER = VVRRPPPPP */
#   define COMPILER_VERSION_PATCH DEC(_MSC_FULL_VER % 100000)
#  else
    /* _MSC_FULL_VER = VVRRPPPP */
#   define COMPILER_VERSION_PATCH DEC(_MSC_FULL_VER % 10000)
#  endif
# endif
# if defined(_MSC_BUILD)
#  define COMPILER_VERSION_TWEAK DEC(_MSC_BUILD)
# endif

#elif defined(__VISUALDSPVERSION__) || defined(__ADSPBLACKFIN__) || defined(__ADSPTS__) || defined(__ADSP21000__)
# define COMPILER_ID "ADSP"
#if defined(__VISUALDSPVERSION__)
  /* __VISUALDSPVERSION__ = 0xVVRRPP00 */
# define COMPILER_VERSION_MAJOR HEX(__VISUALDSPVERSION__»24)
# define COMPILER_VERSION_MINOR HEX(__VISUALDSPVERSION__»16 & 0xFF)
# define COMPILER_VERSION_PATCH HEX(__VISUALDSPVERSION__»8  & 0xFF)
#endif

#elif defined(__IAR_SYSTEMS_ICC__) || defined(__IAR_SYSTEMS_ICC)
# define COMPILER_ID "IAR"
# if defined(__VER__) && defined(__ICCARM__)
#  define COMPILER_VERSION_MAJOR DEC((__VER__) / 1000000)
#  define COMPILER_VERSION_MINOR DEC(((__VER__) / 1000) % 1000)
#  define COMPILER_VERSION_PATCH DEC((__VER__) % 1000)
#  define COMPILER_VERSION_INTERNAL DEC(__IAR_SYSTEMS_ICC__)
# elif defined(__VER__) && (defined(__ICCAVR__) || defined(__ICCRX__) || defined(__ICCRH850__) || defined(__ICCRL78__) || defined(__ICC430__) || defined(__ICCRISCV__) || defined(__ICCV850__) || defined(__ICC8051__) || defined(__ICCSTM8__))
#  define COMPILER_VERSION_MAJOR DEC((__VER__) / 100)
#  define COMPILER_VERSION_MINOR DEC((__VER__) - (((__VER__) / 100)*100))
#  define COMPILER_VERSION_PATCH DEC(__SUBVERSION__)
#  define COMPILER_VERSION_INTERNAL DEC(__IAR_SYSTEMS_ICC__)
# endif

#elif defined(__SDCC_VERSION_MAJOR) || defined(SDCC)
# define COMPILER_ID "SDCC"
# if defined(__SDCC_VERSION_MAJOR)
#  define COMPILER_VERSION_MAJOR DEC(__SDCC_VERSION_MAJOR)
#  define COMPILER_VERSION_MINOR DEC(__SDCC_VERSION_MINOR)
#  define COMPILER_VERSION_PATCH DEC(__SDCC_VERSION_PATCH)
# else
  /* SDCC = VRP */
#  define COMPILER_VERSION_MAJOR DEC(SDCC/100)
#  define COMPILER_VERSION_MINOR DEC(SDCC/10 % 10)
#  define COMPILER_VERSION_PATCH DEC(SDCC    % 10)
# endif


/* These compilers are either not known or too old to define an
  identification macro.  Try to identify the platform and guess that
  it is the native compiler.  */
#elif defined(__hpux) || defined(__hpua)
# define COMPILER_ID "HP"

#else /* unknown compiler */
# define COMPILER_ID ""
#endif

```

```
/* Construct the string literal in pieces to prevent the source from
   getting matched.  Store it in a pointer rather than an array
   because some compilers will just produce instructions to fill the
   array rather than assigning a pointer to a static array.  */
char const* info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]";
#ifdef SIMULATE_ID
char const* info_simulate = "INFO" ":" "simulate[" SIMULATE_ID "]";
#endif

#ifdef __QNXNTO__
char const* qnxnto = "INFO" ":" "qnxnto[]";
#endif

#if defined(__CRAYXT_COMPUTE_LINUX_TARGET)
char const *info_cray = "INFO" ":" "compiler_wrapper[CrayPrgEnv]";
#endif

#define STRINGIFY_HELPER(X) #X
#define STRINGIFY(X) STRINGIFY_HELPER(X)

/* Identify known platforms by name.  */
#if defined(__linux) || defined(__linux__) || defined(linux)
# define PLATFORM_ID "Linux"

#elif defined(__MSYS__)
# define PLATFORM_ID "MSYS"

#elif defined(__CYGWIN__)
# define PLATFORM_ID "Cygwin"

#elif defined(__MINGW32__)
# define PLATFORM_ID "MinGW"

#elif defined(__APPLE__)
# define PLATFORM_ID "Darwin"

#elif defined(_WIN32) || defined(__WIN32__) || defined(WIN32)
# define PLATFORM_ID "Windows"

#elif defined(__FreeBSD__) || defined(__FreeBSD)
# define PLATFORM_ID "FreeBSD"

#elif defined(__NetBSD__) || defined(__NetBSD)
# define PLATFORM_ID "NetBSD"

#elif defined(__OpenBSD__) || defined(__OPENBSD)
# define PLATFORM_ID "OpenBSD"

#elif defined(__sun) || defined(sun)
# define PLATFORM_ID "SunOS"

#elif defined(_AIX) || defined(__AIX) || defined(__AIX__) || defined(__aix) || defined(__aix__)
# define PLATFORM_ID "AIX"

#elif defined(__hpux) || defined(__hpux__)
# define PLATFORM_ID "HP-UX"

#elif defined(__HAIKU__)
# define PLATFORM_ID "Haiku"

#elif defined(__BeOS) || defined(__BEOS__) || defined(_BEOS)
# define PLATFORM_ID "BeOS"

#elif defined(__QNX__) || defined(__QNXNTO__)
# define PLATFORM_ID "QNX"

#elif defined(__tru64) || defined(_tru64) || defined(__TRU64__)
# define PLATFORM_ID "Tru64"

#elif defined(__riscos) || defined(__riscos__)
# define PLATFORM_ID "RISCos"

#elif defined(__sinix) || defined(__sinix__) || defined(__SINIX__)
# define PLATFORM_ID "SINIX"

#elif defined(__UNIX_SV__)
# define PLATFORM_ID "UNIX_SV"

#elif defined(__bsdos__)
# define PLATFORM_ID "BSDOS"

#elif defined(_MPRAS) || defined(MPRAS)
# define PLATFORM_ID "MP-RAS"

#elif defined(__osf) || defined(__osf__)
# define PLATFORM_ID "OSF1"

```

```
#elif defined(_SCO_SV) || defined(SCO_SV) || defined(sco_sv)
# define PLATFORM_ID "SCO_SV"

#elif defined(__ultrix) || defined(__ultrix__) || defined(_ULTRIX)
# define PLATFORM_ID "ULTRIX"

#elif defined(__XENIX__) || defined(_XENIX) || defined(XENIX)
# define PLATFORM_ID "Xenix"

#elif defined(__WATCOMC__)
# if defined(__LINUX__)
#  define PLATFORM_ID "Linux"

# elif defined(__DOS__)
#  define PLATFORM_ID "DOS"

# elif defined(__OS2__)
#  define PLATFORM_ID "OS2"

# elif defined(__WINDOWS__)
#  define PLATFORM_ID "Windows3x"

# elif defined(__VXWORKS__)
#  define PLATFORM_ID "VxWorks"

# else /* unknown platform */
#  define PLATFORM_ID
# endif

#elif defined(__INTEGRITY)
# if defined(INT_178B)
#  define PLATFORM_ID "Integrity178"

# else /* regular Integrity */
#  define PLATFORM_ID "Integrity"
# endif

#else /* unknown platform */
# define PLATFORM_ID

#endif

/* For windows compilers MSVC and Intel we can determine
   the architecture of the compiler being used.  This is because
   the compilers do not have flags that can change the architecture,
   but rather depend on which compiler is being used
*/
#if defined(_WIN32) && defined(_MSC_VER)
# if defined(_M_IA64)
#  define ARCHITECTURE_ID "IA64"

# elif defined(_M_ARM64EC)
#  define ARCHITECTURE_ID "ARM64EC"

# elif defined(_M_X64) || defined(_M_AMD64)
#  define ARCHITECTURE_ID "x64"

# elif defined(_M_IX86)
#  define ARCHITECTURE_ID "X86"

# elif defined(_M_ARM64)
#  define ARCHITECTURE_ID "ARM64"

# elif defined(_M_ARM)
#  if _M_ARM == 4
#   define ARCHITECTURE_ID "ARMV4I"
#  elif _M_ARM == 5
#   define ARCHITECTURE_ID "ARMV5I"
#  else
#   define ARCHITECTURE_ID "ARMV" STRINGIFY(_M_ARM)
#  endif

# elif defined(_M_MIPS)
#  define ARCHITECTURE_ID "MIPS"

# elif defined(_M_SH)
#  define ARCHITECTURE_ID "SHx"

# else /* unknown architecture */
#  define ARCHITECTURE_ID ""
# endif

#elif defined(__WATCOMC__)
# if defined(_M_I86)
#  define ARCHITECTURE_ID "I86"

# elif defined(_M_IX86)
#  define ARCHITECTURE_ID "X86"

# else /* unknown architecture */
#  define ARCHITECTURE_ID ""
# endif

#elif defined(__IAR_SYSTEMS_ICC__) || defined(__IAR_SYSTEMS_ICC)
# if defined(__ICCARM__)
#  define ARCHITECTURE_ID "ARM"

# elif defined(__ICCRX__)
#  define ARCHITECTURE_ID "RX"

# elif defined(__ICCRH850__)
#  define ARCHITECTURE_ID "RH850"

# elif defined(__ICCRL78__)
#  define ARCHITECTURE_ID "RL78"

# elif defined(__ICCRISCV__)
#  define ARCHITECTURE_ID "RISCV"

# elif defined(__ICCAVR__)
#  define ARCHITECTURE_ID "AVR"

# elif defined(__ICC430__)
#  define ARCHITECTURE_ID "MSP430"

# elif defined(__ICCV850__)
#  define ARCHITECTURE_ID "V850"

# elif defined(__ICC8051__)
#  define ARCHITECTURE_ID "8051"

# elif defined(__ICCSTM8__)
#  define ARCHITECTURE_ID "STM8"

# else /* unknown architecture */
#  define ARCHITECTURE_ID ""
# endif

#elif defined(__ghs__)
# if defined(__PPC64__)
#  define ARCHITECTURE_ID "PPC64"

# elif defined(__ppc__)
#  define ARCHITECTURE_ID "PPC"

# elif defined(__ARM__)
#  define ARCHITECTURE_ID "ARM"

# elif defined(__x86_64__)
#  define ARCHITECTURE_ID "x64"

# elif defined(__i386__)
#  define ARCHITECTURE_ID "X86"

# else /* unknown architecture */
#  define ARCHITECTURE_ID ""
# endif

#elif defined(__TI_COMPILER_VERSION__)
# if defined(__TI_ARM__)
#  define ARCHITECTURE_ID "ARM"

# elif defined(__MSP430__)
#  define ARCHITECTURE_ID "MSP430"

# elif defined(__TMS320C28XX__)
#  define ARCHITECTURE_ID "TMS320C28x"

# elif defined(__TMS320C6X__) || defined(_TMS320C6X)
#  define ARCHITECTURE_ID "TMS320C6x"

# else /* unknown architecture */
#  define ARCHITECTURE_ID ""
# endif

#else
#  define ARCHITECTURE_ID
#endif

/* Convert integer to decimal digit literals.  */
#define DEC(n)                   \
  ('0' + (((n) / 10000000)%10)), \
  ('0' + (((n) / 1000000)%10)),  \
  ('0' + (((n) / 100000)%10)),   \
  ('0' + (((n) / 10000)%10)),    \
  ('0' + (((n) / 1000)%10)),     \
  ('0' + (((n) / 100)%10)),      \
  ('0' + (((n) / 10)%10)),       \
  ('0' +  ((n) % 10))

/* Convert integer to hex digit literals.  */
#define HEX(n)             \
  ('0' + ((n)»28 & 0xF)), \
  ('0' + ((n)»24 & 0xF)), \
  ('0' + ((n)»20 & 0xF)), \
  ('0' + ((n)»16 & 0xF)), \
  ('0' + ((n)»12 & 0xF)), \
  ('0' + ((n)»8  & 0xF)), \
  ('0' + ((n)»4  & 0xF)), \
  ('0' + ((n)     & 0xF))

/* Construct a string literal encoding the version number. */
#ifdef COMPILER_VERSION
char const* info_version = "INFO" ":" "compiler_version[" COMPILER_VERSION "]";

/* Construct a string literal encoding the version number components. */
#elif defined(COMPILER_VERSION_MAJOR)
char const info_version[] = {
  'I', 'N', 'F', 'O', ':',
  'c','o','m','p','i','l','e','r','_','v','e','r','s','i','o','n','[',
  COMPILER_VERSION_MAJOR,
# ifdef COMPILER_VERSION_MINOR
  '.', COMPILER_VERSION_MINOR,
#  ifdef COMPILER_VERSION_PATCH
   '.', COMPILER_VERSION_PATCH,
#   ifdef COMPILER_VERSION_TWEAK
    '.', COMPILER_VERSION_TWEAK,
#   endif
#  endif
# endif
  ']','\0'};
#endif

/* Construct a string literal encoding the internal version number. */
#ifdef COMPILER_VERSION_INTERNAL
char const info_version_internal[] = {
  'I', 'N', 'F', 'O', ':',
  'c','o','m','p','i','l','e','r','_','v','e','r','s','i','o','n','_',
  'i','n','t','e','r','n','a','l','[',
  COMPILER_VERSION_INTERNAL,']','\0'};
#elif defined(COMPILER_VERSION_INTERNAL_STR)
char const* info_version_internal = "INFO" ":" "compiler_version_internal["
 COMPILER_VERSION_INTERNAL_STR "]";
#endif

/* Construct a string literal encoding the version number components. */
#ifdef SIMULATE_VERSION_MAJOR
char const info_simulate_version[] = {
  'I', 'N', 'F', 'O', ':',
  's','i','m','u','l','a','t','e','_','v','e','r','s','i','o','n','[',
  SIMULATE_VERSION_MAJOR,
# ifdef SIMULATE_VERSION_MINOR
  '.', SIMULATE_VERSION_MINOR,
#  ifdef SIMULATE_VERSION_PATCH
   '.', SIMULATE_VERSION_PATCH,
#   ifdef SIMULATE_VERSION_TWEAK
    '.', SIMULATE_VERSION_TWEAK,
#   endif
#  endif
# endif
  ']','\0'};
#endif

/* Construct the string literal in pieces to prevent the source from
   getting matched.  Store it in a pointer rather than an array
   because some compilers will just produce instructions to fill the
   array rather than assigning a pointer to a static array.  */
char const* info_platform = "INFO" ":" "platform[" PLATFORM_ID "]";
char const* info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]";



#if !defined(__STDC__) && !defined(__clang__)
# if defined(_MSC_VER) || defined(__ibmxl__) || defined(__IBMC__)
#  define C_DIALECT "90"
# else
#  define C_DIALECT
# endif
#elif __STDC_VERSION__ > 201710L
# define C_DIALECT "23"
#elif __STDC_VERSION__ >= 201710L
```

```
# define C_DIALECT "17"
#elif __STDC_VERSION__ >= 201000L
# define C_DIALECT "11"
#elif __STDC_VERSION__ >= 199901L
# define C_DIALECT "99"
#else
# define C_DIALECT "90"
#endif
const char* info_language_dialect_default =
  "INFO" ":" "dialect_default[" C_DIALECT "]";

/*--------------------------------------------------------------------------*/

#ifdef ID_VOID_MAIN
void main() {}
#else
# if defined(__CLASSIC_C__)
int main(argc, argv) int argc; char *argv[];
# else
int main(int argc, char* argv[])
# endif
{
  int require = 0;
  require += info_compiler[argc];
  require += info_platform[argc];
  require += info_arch[argc];
#ifdef COMPILER_VERSION_MAJOR
  require += info_version[argc];
#endif
#ifdef COMPILER_VERSION_INTERNAL
  require += info_version_internal[argc];
#endif
#ifdef SIMULATE_ID
  require += info_simulate[argc];
#endif
#ifdef SIMULATE_VERSION_MAJOR
  require += info_simulate_version[argc];
#endif
#if defined(__CRAYXT_COMPUTE_LINUX_TARGET)
  require += info_cray[argc];
#endif
  require += info_language_dialect_default[argc];
  (void)argv;
  return require;
}
#endif
```

6.3 D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/↩ CMakeFiles/3.21.1/CompilerIdCXX/CMakeCXXCompilerId.cpp File Reference

Macros

- #define __has_include(x) 0
- #define COMPILER_ID ""
- #define STRINGIFY_HELPER(X) #X
- #define STRINGIFY(X) STRINGIFY_HELPER(X)
- #define PLATFORM_ID
- #define ARCHITECTURE_ID
- #define DEC(n)
- #define HEX(n)
- #define CXX_STD __cplusplus

Functions

- int main (int argc, char ∗argv[ ])

Variables

- char const ∗ info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
- char const ∗ info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
- char const ∗ info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
- const char ∗ info_language_dialect_default

6.3.1 Macro Definition Documentation

6.3.1.1 __has_include

```
#define __has_include(
              x ) 0
```

Definition at line 11 of file CMakeCXXCompilerId.cpp.

6.3.1.2 ARCHITECTURE_ID

```
#define ARCHITECTURE_ID
```

Definition at line 653 of file CMakeCXXCompilerId.cpp.

6.3.1.3 COMPILER_ID

```
#define COMPILER_ID ""
```

Definition at line 397 of file CMakeCXXCompilerId.cpp.

6.3.1.4 CXX_STD

```
#define CXX_STD __cplusplus
```

Definition at line 751 of file CMakeCXXCompilerId.cpp.

6.3.1.5 DEC

```
#define DEC(
              n )
```

Value:

```
  ('0' + (((n) / 10000000)%10)), \
  ('0' + (((n) / 1000000)%10)),  \
  ('0' + (((n) / 100000)%10)),   \
  ('0' + (((n) / 10000)%10)),    \
  ('0' + (((n) / 1000)%10)),     \
  ('0' + (((n) / 100)%10)),      \
  ('0' + (((n) / 10)%10)),       \
  ('0' +  ((n) % 10))
```

Definition at line 657 of file CMakeCXXCompilerId.cpp.

6.3.1.6 HEX

```
#define HEX(
              n )
```

Value:

```
  ('0' + ((n)»28 & 0xF)), \
  ('0' + ((n)»24 & 0xF)), \
  ('0' + ((n)»20 & 0xF)), \
  ('0' + ((n)»16 & 0xF)), \
  ('0' + ((n)»12 & 0xF)), \
  ('0' + ((n)»8  & 0xF)), \
  ('0' + ((n)»4  & 0xF)), \
  ('0' + ((n)     & 0xF))
```

Definition at line 668 of file CMakeCXXCompilerId.cpp.

6.3.1.7 PLATFORM_ID

```
#define PLATFORM_ID
```

Definition at line 525 of file CMakeCXXCompilerId.cpp.

6.3.1.8 STRINGIFY

```
#define STRINGIFY(
              X ) STRINGIFY_HELPER(X)
```

Definition at line 418 of file CMakeCXXCompilerId.cpp.

6.3.1.9 STRINGIFY_HELPER

```
#define STRINGIFY_HELPER(
              X ) #X
```

Definition at line 417 of file CMakeCXXCompilerId.cpp.

6.3.2 Function Documentation

6.3.2.1 main()

```
int main (
             int argc,
             char * argv[ ] )
```

Definition at line 772 of file CMakeCXXCompilerId.cpp.

```
{
  int require = 0;
  require += info_compiler[argc];
  require += info_platform[argc];
#ifdef COMPILER_VERSION_MAJOR
  require += info_version[argc];
#endif
#ifdef COMPILER_VERSION_INTERNAL
  require += info_version_internal[argc];
#endif
#ifdef SIMULATE_ID
  require += info_simulate[argc];
#endif
#ifdef SIMULATE_VERSION_MAJOR
  require += info_simulate_version[argc];
#endif
#if defined(__CRAYXT_COMPUTE_LINUX_TARGET)
  require += info_cray[argc];
#endif
  require += info_language_dialect_default[argc];
  (void)argv;
  return require;
}
```

6.3.3 Variable Documentation

6.3.3.1 info_arch

```
char const* info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
```

Definition at line 734 of file CMakeCXXCompilerId.cpp.

6.3.3.2 info_compiler

```
char const* info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
```

Definition at line 404 of file CMakeCXXCompilerId.cpp.

6.3.3.3 info_language_dialect_default

```
const char* info_language_dialect_default
```

Initial value:
```
= "INFO" ":" "dialect_default["
  "98"
"]"
```

Definition at line 754 of file CMakeCXXCompilerId.cpp.

6.3.3.4 info_platform

```
char const* info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
```

Definition at line 733 of file CMakeCXXCompilerId.cpp.

6.4 CMakeCXXCompilerId.cpp

Go to the documentation of this file.
```
/* This source file must have a .cpp extension so that all C++ compilers
   recognize the extension without flags.  Borland does not know .cxx for
   example.  */
#ifndef __cplusplus
# error "A C compiler has been selected for C++."
#endif

#if !defined(__has_include)
/* If the compiler does not have __has_include, pretend the answer is
   always no.  */
#  define __has_include(x) 0
#endif


/* Version number components: V=Version, R=Revision, P=Patch
   Version date components:   YYYY=Year, MM=Month,   DD=Day  */

#if defined(__COMO__)
# define COMPILER_ID "Comeau"
  /* __COMO_VERSION__ = VRR */
# define COMPILER_VERSION_MAJOR DEC(__COMO_VERSION__ / 100)
# define COMPILER_VERSION_MINOR DEC(__COMO_VERSION__ % 100)

#elif defined(__INTEL_COMPILER) || defined(__ICC)
# define COMPILER_ID "Intel"
# if defined(_MSC_VER)
#  define SIMULATE_ID "MSVC"
# endif
# if defined(__GNUC__)
#  define SIMULATE_ID "GNU"
# endif
  /* __INTEL_COMPILER = VRP prior to 2021, and then VVVV for 2021 and later,
     except that a few beta releases use the old format with V=2021.  */
# if __INTEL_COMPILER < 2021 || __INTEL_COMPILER == 202110 || __INTEL_COMPILER == 202111
#  define COMPILER_VERSION_MAJOR DEC(__INTEL_COMPILER/100)
#  define COMPILER_VERSION_MINOR DEC(__INTEL_COMPILER/10 % 10)
#  if defined(__INTEL_COMPILER_UPDATE)
#   define COMPILER_VERSION_PATCH DEC(__INTEL_COMPILER_UPDATE)
#  else
#   define COMPILER_VERSION_PATCH DEC(__INTEL_COMPILER   % 10)
#  endif
# else
#  define COMPILER_VERSION_MAJOR DEC(__INTEL_COMPILER)
#  define COMPILER_VERSION_MINOR DEC(__INTEL_COMPILER_UPDATE)
   /* The third version component from --version is an update index,
      but no macro is provided for it.  */
#  define COMPILER_VERSION_PATCH DEC(0)
# endif
# if defined(__INTEL_COMPILER_BUILD_DATE)
   /* __INTEL_COMPILER_BUILD_DATE = YYYYMMDD */
#  define COMPILER_VERSION_TWEAK DEC(__INTEL_COMPILER_BUILD_DATE)
# endif
# if defined(_MSC_VER)
   /* _MSC_VER = VVRR */
#  define SIMULATE_VERSION_MAJOR DEC(_MSC_VER / 100)
#  define SIMULATE_VERSION_MINOR DEC(_MSC_VER % 100)
# endif
# if defined(__GNUC__)
#  define SIMULATE_VERSION_MAJOR DEC(__GNUC__)
# elif defined(__GNUG__)
#  define SIMULATE_VERSION_MAJOR DEC(__GNUG__)
# endif
# if defined(__GNUC_MINOR__)
#  define SIMULATE_VERSION_MINOR DEC(__GNUC_MINOR__)
# endif
# if defined(__GNUC_PATCHLEVEL__)
#  define SIMULATE_VERSION_PATCH DEC(__GNUC_PATCHLEVEL__)
# endif

#elif (defined(__clang__) && defined(__INTEL_CLANG_COMPILER)) || defined(__INTEL_LLVM_COMPILER)
# define COMPILER_ID "IntelLLVM"
#if defined(_MSC_VER)
# define SIMULATE_ID "MSVC"
#endif
#if defined(__GNUC__)
# define SIMULATE_ID "GNU"
#endif
/* __INTEL_LLVM_COMPILER = VVVVRP prior to 2021.2.0, VVVVRRPP for 2021.2.0 and
 * later.  Look for 6 digit vs. 8 digit version number to decide encoding.
 * VVVV is no smaller than the current year when a version is released.
 */
#if __INTEL_LLVM_COMPILER < 1000000L
# define COMPILER_VERSION_MAJOR DEC(__INTEL_LLVM_COMPILER/100)
# define COMPILER_VERSION_MINOR DEC(__INTEL_LLVM_COMPILER/10 % 10)
# define COMPILER_VERSION_PATCH DEC(__INTEL_LLVM_COMPILER    % 10)
#else
# define COMPILER_VERSION_MAJOR DEC(__INTEL_LLVM_COMPILER/10000)
# define COMPILER_VERSION_MINOR DEC(__INTEL_LLVM_COMPILER/100 % 100)
# define COMPILER_VERSION_PATCH DEC(__INTEL_LLVM_COMPILER     % 100)
#endif
#if defined(_MSC_VER)
  /* _MSC_VER = VVRR */
# define SIMULATE_VERSION_MAJOR DEC(_MSC_VER / 100)
# define SIMULATE_VERSION_MINOR DEC(_MSC_VER % 100)
#endif
#if defined(__GNUC__)
# define SIMULATE_VERSION_MAJOR DEC(__GNUC__)
#elif defined(__GNUG__)
# define SIMULATE_VERSION_MAJOR DEC(__GNUG__)
#endif
#if defined(__GNUC_MINOR__)
# define SIMULATE_VERSION_MINOR DEC(__GNUC_MINOR__)
#endif
#if defined(__GNUC_PATCHLEVEL__)
# define SIMULATE_VERSION_PATCH DEC(__GNUC_PATCHLEVEL__)
#endif

#elif defined(__PATHCC__)
# define COMPILER_ID "PathScale"
# define COMPILER_VERSION_MAJOR DEC(__PATHCC__)
# define COMPILER_VERSION_MINOR DEC(__PATHCC_MINOR__)
# if defined(__PATHCC_PATCHLEVEL__)
#  define COMPILER_VERSION_PATCH DEC(__PATHCC_PATCHLEVEL__)
# endif

#elif defined(__BORLANDC__) && defined(__CODEGEARC_VERSION__)
# define COMPILER_ID "Embarcadero"
# define COMPILER_VERSION_MAJOR HEX(__CODEGEARC_VERSION__»24 & 0x00FF)
# define COMPILER_VERSION_MINOR HEX(__CODEGEARC_VERSION__»16 & 0x00FF)
# define COMPILER_VERSION_PATCH DEC(__CODEGEARC_VERSION__     & 0xFFFF)

```

```
#elif defined(__BORLANDC__)
# define COMPILER_ID "Borland"
  /* __BORLANDC__ = 0xVRR */
# define COMPILER_VERSION_MAJOR HEX(__BORLANDC__»8)
# define COMPILER_VERSION_MINOR HEX(__BORLANDC__ & 0xFF)

#elif defined(__WATCOMC__) && __WATCOMC__ < 1200
# define COMPILER_ID "Watcom"
   /* __WATCOMC__ = VVRR */
# define COMPILER_VERSION_MAJOR DEC(__WATCOMC__ / 100)
# define COMPILER_VERSION_MINOR DEC((__WATCOMC__ / 10) % 10)
# if (__WATCOMC__ % 10) > 0
#  define COMPILER_VERSION_PATCH DEC(__WATCOMC__ % 10)
# endif

#elif defined(__WATCOMC__)
# define COMPILER_ID "OpenWatcom"
   /* __WATCOMC__ = VVRP + 1100 */
# define COMPILER_VERSION_MAJOR DEC((__WATCOMC__ - 1100) / 100)
# define COMPILER_VERSION_MINOR DEC((__WATCOMC__ / 10) % 10)
# if (__WATCOMC__ % 10) > 0
#  define COMPILER_VERSION_PATCH DEC(__WATCOMC__ % 10)
# endif

#elif defined(__SUNPRO_CC)
# define COMPILER_ID "SunPro"
# if __SUNPRO_CC >= 0x5100
   /* __SUNPRO_CC = 0xVRRP */
#  define COMPILER_VERSION_MAJOR HEX(__SUNPRO_CC»12)
#  define COMPILER_VERSION_MINOR HEX(__SUNPRO_CC»4 & 0xFF)
#  define COMPILER_VERSION_PATCH HEX(__SUNPRO_CC    & 0xF)
# else
   /* __SUNPRO_CC = 0xVRP */
#  define COMPILER_VERSION_MAJOR HEX(__SUNPRO_CC»8)
#  define COMPILER_VERSION_MINOR HEX(__SUNPRO_CC»4 & 0xF)
#  define COMPILER_VERSION_PATCH HEX(__SUNPRO_CC    & 0xF)
# endif

#elif defined(__HP_aCC)
# define COMPILER_ID "HP"
  /* __HP_aCC = VVRRPP */
# define COMPILER_VERSION_MAJOR DEC(__HP_aCC/10000)
# define COMPILER_VERSION_MINOR DEC(__HP_aCC/100 % 100)
# define COMPILER_VERSION_PATCH DEC(__HP_aCC     % 100)

#elif defined(__DECCXX)
# define COMPILER_ID "Compaq"
  /* __DECCXX_VER = VVRRTPPPP */
# define COMPILER_VERSION_MAJOR DEC(__DECCXX_VER/10000000)
# define COMPILER_VERSION_MINOR DEC(__DECCXX_VER/100000  % 100)
# define COMPILER_VERSION_PATCH DEC(__DECCXX_VER         % 10000)

#elif defined(__IBMCPP__) && defined(__COMPILER_VER__)
# define COMPILER_ID "zOS"
  /* __IBMCPP__ = VRP */
# define COMPILER_VERSION_MAJOR DEC(__IBMCPP__/100)
# define COMPILER_VERSION_MINOR DEC(__IBMCPP__/10 % 10)
# define COMPILER_VERSION_PATCH DEC(__IBMCPP__    % 10)

#elif defined(__ibmxl__) && defined(__clang__)
# define COMPILER_ID "XLClang"
# define COMPILER_VERSION_MAJOR DEC(__ibmxl_version__)
# define COMPILER_VERSION_MINOR DEC(__ibmxl_release__)
# define COMPILER_VERSION_PATCH DEC(__ibmxl_modification__)
# define COMPILER_VERSION_TWEAK DEC(__ibmxl_ptf_fix_level__)


#elif defined(__IBMCPP__) && !defined(__COMPILER_VER__) && __IBMCPP__ >= 800
# define COMPILER_ID "XL"
  /* __IBMCPP__ = VRP */
# define COMPILER_VERSION_MAJOR DEC(__IBMCPP__/100)
# define COMPILER_VERSION_MINOR DEC(__IBMCPP__/10 % 10)
# define COMPILER_VERSION_PATCH DEC(__IBMCPP__    % 10)

#elif defined(__IBMCPP__) && !defined(__COMPILER_VER__) && __IBMCPP__ < 800
# define COMPILER_ID "VisualAge"
  /* __IBMCPP__ = VRP */
# define COMPILER_VERSION_MAJOR DEC(__IBMCPP__/100)
# define COMPILER_VERSION_MINOR DEC(__IBMCPP__/10 % 10)
# define COMPILER_VERSION_PATCH DEC(__IBMCPP__    % 10)

#elif defined(__NVCOMPILER)
# define COMPILER_ID "NVHPC"
# define COMPILER_VERSION_MAJOR DEC(__NVCOMPILER_MAJOR__)
# define COMPILER_VERSION_MINOR DEC(__NVCOMPILER_MINOR__)
# if defined(__NVCOMPILER_PATCHLEVEL__)
#  define COMPILER_VERSION_PATCH DEC(__NVCOMPILER_PATCHLEVEL__)
# endif

#elif defined(__PGI)
# define COMPILER_ID "PGI"
# define COMPILER_VERSION_MAJOR DEC(__PGIC__)
# define COMPILER_VERSION_MINOR DEC(__PGIC_MINOR__)
# if defined(__PGIC_PATCHLEVEL__)
#  define COMPILER_VERSION_PATCH DEC(__PGIC_PATCHLEVEL__)
# endif

#elif defined(_CRAYC)
# define COMPILER_ID "Cray"
# define COMPILER_VERSION_MAJOR DEC(_RELEASE_MAJOR)
# define COMPILER_VERSION_MINOR DEC(_RELEASE_MINOR)

#elif defined(__TI_COMPILER_VERSION__)
# define COMPILER_ID "TI"
  /* __TI_COMPILER_VERSION__ = VVVRRRPPP */
# define COMPILER_VERSION_MAJOR DEC(__TI_COMPILER_VERSION__/1000000)
# define COMPILER_VERSION_MINOR DEC(__TI_COMPILER_VERSION__/1000   % 1000)
# define COMPILER_VERSION_PATCH DEC(__TI_COMPILER_VERSION__        % 1000)

#elif defined(__CLANG_FUJITSU)
# define COMPILER_ID "FujitsuClang"
# define COMPILER_VERSION_MAJOR DEC(__FCC_major__)
# define COMPILER_VERSION_MINOR DEC(__FCC_minor__)
# define COMPILER_VERSION_PATCH DEC(__FCC_patchlevel__)
# define COMPILER_VERSION_INTERNAL_STR __clang_version__


#elif defined(__FUJITSU)
# define COMPILER_ID "Fujitsu"
# if defined(__FCC_version__)
#   define COMPILER_VERSION __FCC_version__
# elif defined(__FCC_major__)
#   define COMPILER_VERSION_MAJOR DEC(__FCC_major__)
#   define COMPILER_VERSION_MINOR DEC(__FCC_minor__)
#   define COMPILER_VERSION_PATCH DEC(__FCC_patchlevel__)
# endif
# if defined(__fcc_version)
#   define COMPILER_VERSION_INTERNAL DEC(__fcc_version)
# elif defined(__FCC_VERSION)
#   define COMPILER_VERSION_INTERNAL DEC(__FCC_VERSION)
# endif


#elif defined(__ghs__)
# define COMPILER_ID "GHS"
/* __GHS_VERSION_NUMBER = VVVVRP */
# ifdef __GHS_VERSION_NUMBER
# define COMPILER_VERSION_MAJOR DEC(__GHS_VERSION_NUMBER / 100)
# define COMPILER_VERSION_MINOR DEC(__GHS_VERSION_NUMBER / 10 % 10)
# define COMPILER_VERSION_PATCH DEC(__GHS_VERSION_NUMBER      % 10)
# endif

#elif defined(__SCO_VERSION__)
# define COMPILER_ID "SCO"

#elif defined(__ARMCC_VERSION) && !defined(__clang__)
# define COMPILER_ID "ARMCC"
#if __ARMCC_VERSION >= 1000000
  /* __ARMCC_VERSION = VRRPPPP */
  # define COMPILER_VERSION_MAJOR DEC(__ARMCC_VERSION/1000000)
  # define COMPILER_VERSION_MINOR DEC(__ARMCC_VERSION/10000 % 100)
  # define COMPILER_VERSION_PATCH DEC(__ARMCC_VERSION     % 10000)
#else
  /* __ARMCC_VERSION = VRPPPP */
  # define COMPILER_VERSION_MAJOR DEC(__ARMCC_VERSION/100000)
  # define COMPILER_VERSION_MINOR DEC(__ARMCC_VERSION/10000 % 10)
  # define COMPILER_VERSION_PATCH DEC(__ARMCC_VERSION    % 10000)
#endif


#elif defined(__clang__) && defined(__apple_build_version__)
# define COMPILER_ID "AppleClang"
# if defined(_MSC_VER)
#  define SIMULATE_ID "MSVC"
# endif
# define COMPILER_VERSION_MAJOR DEC(__clang_major__)
# define COMPILER_VERSION_MINOR DEC(__clang_minor__)
# define COMPILER_VERSION_PATCH DEC(__clang_patchlevel__)
# if defined(_MSC_VER)
   /* _MSC_VER = VVRR */
#  define SIMULATE_VERSION_MAJOR DEC(_MSC_VER / 100)
#  define SIMULATE_VERSION_MINOR DEC(_MSC_VER % 100)
# endif
# define COMPILER_VERSION_TWEAK DEC(__apple_build_version__)
```

```

#elif defined(__clang__) && defined(__ARMCOMPILER_VERSION)
# define COMPILER_ID "ARMClang"
  # define COMPILER_VERSION_MAJOR DEC(__ARMCOMPILER_VERSION/1000000)
  # define COMPILER_VERSION_MINOR DEC(__ARMCOMPILER_VERSION/10000 % 100)
  # define COMPILER_VERSION_PATCH DEC(__ARMCOMPILER_VERSION     % 10000)
# define COMPILER_VERSION_INTERNAL DEC(__ARMCOMPILER_VERSION)

#elif defined(__clang__) && __has_include(<hip/hip_version.h>)
# define COMPILER_ID "ROCMClang"
# if defined(_MSC_VER)
#  define SIMULATE_ID "MSVC"
# elif defined(__clang__)
#  define SIMULATE_ID "Clang"
# elif defined(__GNUC__)
#  define SIMULATE_ID "GNU"
# endif
# if defined(__clang__) && __has_include(<hip/hip_version.h>)
#  include <hip/hip_version.h>
#  define COMPILER_VERSION_MAJOR DEC(HIP_VERSION_MAJOR)
#  define COMPILER_VERSION_MINOR DEC(HIP_VERSION_MINOR)
#  define COMPILER_VERSION_PATCH DEC(HIP_VERSION_PATCH)
# endif

#elif defined(__clang__)
# define COMPILER_ID "Clang"
# if defined(_MSC_VER)
#  define SIMULATE_ID "MSVC"
# endif
# define COMPILER_VERSION_MAJOR DEC(__clang_major__)
# define COMPILER_VERSION_MINOR DEC(__clang_minor__)
# define COMPILER_VERSION_PATCH DEC(__clang_patchlevel__)
# if defined(_MSC_VER)
   /* _MSC_VER = VVRR */
#  define SIMULATE_VERSION_MAJOR DEC(_MSC_VER / 100)
#  define SIMULATE_VERSION_MINOR DEC(_MSC_VER % 100)
# endif

#elif defined(__GNUC__) || defined(__GNUG__)
# define COMPILER_ID "GNU"
# if defined(__GNUC__)
#  define COMPILER_VERSION_MAJOR DEC(__GNUC__)
# else
#  define COMPILER_VERSION_MAJOR DEC(__GNUG__)
# endif
# if defined(__GNUC_MINOR__)
#  define COMPILER_VERSION_MINOR DEC(__GNUC_MINOR__)
# endif
# if defined(__GNUC_PATCHLEVEL__)
#  define COMPILER_VERSION_PATCH DEC(__GNUC_PATCHLEVEL__)
# endif

#elif defined(_MSC_VER)
# define COMPILER_ID "MSVC"
  /* _MSC_VER = VVRR */
# define COMPILER_VERSION_MAJOR DEC(_MSC_VER / 100)
# define COMPILER_VERSION_MINOR DEC(_MSC_VER % 100)
# if defined(_MSC_FULL_VER)
#  if _MSC_VER >= 1400
    /* _MSC_FULL_VER = VVRRPPPPP */
#   define COMPILER_VERSION_PATCH DEC(_MSC_FULL_VER % 100000)
#  else
    /* _MSC_FULL_VER = VVRRPPPP */
#   define COMPILER_VERSION_PATCH DEC(_MSC_FULL_VER % 10000)
#  endif
# endif
# if defined(_MSC_BUILD)
#  define COMPILER_VERSION_TWEAK DEC(_MSC_BUILD)
# endif

#elif defined(__VISUALDSPVERSION__) || defined(__ADSPBLACKFIN__) || defined(__ADSPTS__) ||
 defined(__ADSP21000__)
# define COMPILER_ID "ADSP"
#if defined(__VISUALDSPVERSION__)
  /* __VISUALDSPVERSION__ = 0xVVRRPP00 */
# define COMPILER_VERSION_MAJOR HEX(__VISUALDSPVERSION__»24)
# define COMPILER_VERSION_MINOR HEX(__VISUALDSPVERSION__»16 & 0xFF)
# define COMPILER_VERSION_PATCH HEX(__VISUALDSPVERSION__»8  & 0xFF)
#endif

#elif defined(__IAR_SYSTEMS_ICC__) || defined(__IAR_SYSTEMS_ICC)
# define COMPILER_ID "IAR"
# if defined(__VER__) && defined(__ICCARM__)
#  define COMPILER_VERSION_MAJOR DEC((__VER__) / 1000000)
#  define COMPILER_VERSION_MINOR DEC(((__VER__) / 1000) % 1000)
#  define COMPILER_VERSION_PATCH DEC((__VER__) % 1000)
#  define COMPILER_VERSION_INTERNAL DEC(__IAR_SYSTEMS_ICC__)
# elif defined(__VER__) && (defined(__ICCAVR__) || defined(__ICCRX__) || defined(__ICCRH850__) ||
 defined(__ICCRL78__) || defined(__ICC430__) || defined(__ICCRISCV__) || defined(__ICCV850__) ||
 defined(__ICC8051__) || defined(__ICCSTM8__))
#  define COMPILER_VERSION_MAJOR DEC((__VER__) / 100)
#  define COMPILER_VERSION_MINOR DEC((__VER__) - (((__VER__) / 100)*100))
#  define COMPILER_VERSION_PATCH DEC(__SUBVERSION__)
#  define COMPILER_VERSION_INTERNAL DEC(__IAR_SYSTEMS_ICC__)
# endif


/* These compilers are either not known or too old to define an
  identification macro.  Try to identify the platform and guess that
  it is the native compiler.  */
#elif defined(__hpux) || defined(__hpua)
# define COMPILER_ID "HP"

#else /* unknown compiler */
# define COMPILER_ID ""
#endif

/* Construct the string literal in pieces to prevent the source from
   getting matched.  Store it in a pointer rather than an array
   because some compilers will just produce instructions to fill the
   array rather than assigning a pointer to a static array.  */
char const* info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]";
#ifdef SIMULATE_ID
char const* info_simulate = "INFO" ":" "simulate[" SIMULATE_ID "]";
#endif

#ifdef __QNXNTO__
char const* qnxnto = "INFO" ":" "qnxnto[]";
#endif

#if defined(__CRAYXT_COMPUTE_LINUX_TARGET)
char const *info_cray = "INFO" ":" "compiler_wrapper[CrayPrgEnv]";
#endif

#define STRINGIFY_HELPER(X) #X
#define STRINGIFY(X) STRINGIFY_HELPER(X)

/* Identify known platforms by name.  */
#if defined(__linux) || defined(__linux__) || defined(linux)
# define PLATFORM_ID "Linux"

#elif defined(__MSYS__)
# define PLATFORM_ID "MSYS"

#elif defined(__CYGWIN__)
# define PLATFORM_ID "Cygwin"

#elif defined(__MINGW32__)
# define PLATFORM_ID "MinGW"

#elif defined(__APPLE__)
# define PLATFORM_ID "Darwin"

#elif defined(_WIN32) || defined(__WIN32__) || defined(WIN32)
# define PLATFORM_ID "Windows"

#elif defined(__FreeBSD__) || defined(__FreeBSD)
# define PLATFORM_ID "FreeBSD"

#elif defined(__NetBSD__) || defined(__NetBSD)
# define PLATFORM_ID "NetBSD"

#elif defined(__OpenBSD__) || defined(__OPENBSD)
# define PLATFORM_ID "OpenBSD"

#elif defined(__sun) || defined(sun)
# define PLATFORM_ID "SunOS"

#elif defined(_AIX) || defined(__AIX) || defined(__AIX__) || defined(__aix) || defined(__aix__)
# define PLATFORM_ID "AIX"

#elif defined(__hpux) || defined(__hpux__)
# define PLATFORM_ID "HP-UX"

#elif defined(__HAIKU__)
# define PLATFORM_ID "Haiku"

#elif defined(__BeOS) || defined(__BEOS__) || defined(_BEOS)
# define PLATFORM_ID "BeOS"

#elif defined(__QNX__) || defined(__QNXNTO__)
# define PLATFORM_ID "QNX"

#elif defined(__tru64) || defined(_tru64) || defined(__TRU64__)
```

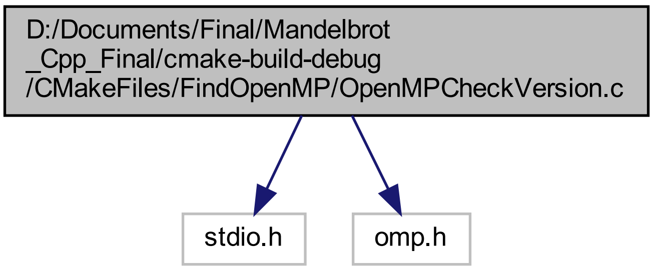
```
# define PLATFORM_ID "Tru64"

#elif defined(__riscos) || defined(__riscos__)
# define PLATFORM_ID "RISCos"

#elif defined(__sinix) || defined(__sinix__) || defined(__SINIX__)
# define PLATFORM_ID "SINIX"

#elif defined(__UNIX_SV__)
# define PLATFORM_ID "UNIX_SV"

#elif defined(__bsdos__)
# define PLATFORM_ID "BSDOS"

#elif defined(_MPRAS) || defined(MPRAS)
# define PLATFORM_ID "MP-RAS"

#elif defined(__osf) || defined(__osf__)
# define PLATFORM_ID "OSF1"

#elif defined(_SCO_SV) || defined(SCO_SV) || defined(sco_sv)
# define PLATFORM_ID "SCO_SV"

#elif defined(__ultrix) || defined(__ultrix__) || defined(_ULTRIX)
# define PLATFORM_ID "ULTRIX"

#elif defined(__XENIX__) || defined(_XENIX) || defined(XENIX)
# define PLATFORM_ID "Xenix"

#elif defined(__WATCOMC__)
# if defined(__LINUX__)
#  define PLATFORM_ID "Linux"

# elif defined(__DOS__)
#  define PLATFORM_ID "DOS"

# elif defined(__OS2__)
#  define PLATFORM_ID "OS2"

# elif defined(__WINDOWS__)
#  define PLATFORM_ID "Windows3x"

# elif defined(__VXWORKS__)
#  define PLATFORM_ID "VxWorks"

# else /* unknown platform */
#  define PLATFORM_ID
# endif

#elif defined(__INTEGRITY)
# if defined(INT_178B)
#  define PLATFORM_ID "Integrity178"

# else /* regular Integrity */
#  define PLATFORM_ID "Integrity"
# endif

#else /* unknown platform */
# define PLATFORM_ID

#endif

/* For windows compilers MSVC and Intel we can determine
   the architecture of the compiler being used.  This is because
   the compilers do not have flags that can change the architecture,
   but rather depend on which compiler is being used
*/
#if defined(_WIN32) && defined(_MSC_VER)
# if defined(_M_IA64)
#  define ARCHITECTURE_ID "IA64"

# elif defined(_M_ARM64EC)
#  define ARCHITECTURE_ID "ARM64EC"

# elif defined(_M_X64) || defined(_M_AMD64)
#  define ARCHITECTURE_ID "x64"

# elif defined(_M_IX86)
#  define ARCHITECTURE_ID "X86"

# elif defined(_M_ARM64)
#  define ARCHITECTURE_ID "ARM64"

# elif defined(_M_ARM)
#  if _M_ARM == 4
#   define ARCHITECTURE_ID "ARMV4I"
#  elif _M_ARM == 5
#   define ARCHITECTURE_ID "ARMV5I"
#  else
#   define ARCHITECTURE_ID "ARMV" STRINGIFY(_M_ARM)
#  endif

# elif defined(_M_MIPS)
#  define ARCHITECTURE_ID "MIPS"

# elif defined(_M_SH)
#  define ARCHITECTURE_ID "SHx"

# else /* unknown architecture */
#  define ARCHITECTURE_ID ""
# endif

#elif defined(__WATCOMC__)
# if defined(_M_I86)
#  define ARCHITECTURE_ID "I86"

# elif defined(_M_IX86)
#  define ARCHITECTURE_ID "X86"

# else /* unknown architecture */
#  define ARCHITECTURE_ID ""
# endif

#elif defined(__IAR_SYSTEMS_ICC__) || defined(__IAR_SYSTEMS_ICC)
# if defined(__ICCARM__)
#  define ARCHITECTURE_ID "ARM"

# elif defined(__ICCRX__)
#  define ARCHITECTURE_ID "RX"

# elif defined(__ICCRH850__)
#  define ARCHITECTURE_ID "RH850"

# elif defined(__ICCRL78__)
#  define ARCHITECTURE_ID "RL78"

# elif defined(__ICCRISCV__)
#  define ARCHITECTURE_ID "RISCV"

# elif defined(__ICCAVR__)
#  define ARCHITECTURE_ID "AVR"

# elif defined(__ICC430__)
#  define ARCHITECTURE_ID "MSP430"

# elif defined(__ICCV850__)
#  define ARCHITECTURE_ID "V850"

# elif defined(__ICC8051__)
#  define ARCHITECTURE_ID "8051"

# elif defined(__ICCSTM8__)
#  define ARCHITECTURE_ID "STM8"

# else /* unknown architecture */
#  define ARCHITECTURE_ID ""
# endif

#elif defined(__ghs__)
# if defined(__PPC64__)
#  define ARCHITECTURE_ID "PPC64"

# elif defined(__ppc__)
#  define ARCHITECTURE_ID "PPC"

# elif defined(__ARM__)
#  define ARCHITECTURE_ID "ARM"

# elif defined(__x86_64__)
#  define ARCHITECTURE_ID "x64"

# elif defined(__i386__)
#  define ARCHITECTURE_ID "X86"

# else /* unknown architecture */
#  define ARCHITECTURE_ID ""
# endif

#elif defined(__TI_COMPILER_VERSION__)
# if defined(__TI_ARM__)
#  define ARCHITECTURE_ID "ARM"

# elif defined(__MSP430__)
#  define ARCHITECTURE_ID "MSP430"

# elif defined(__TMS320C28XX__)
#  define ARCHITECTURE_ID "TMS320C28x"

# elif defined(__TMS320C6X__) || defined(_TMS320C6X)
#  define ARCHITECTURE_ID "TMS320C6x"

# else /* unknown architecture */
#  define ARCHITECTURE_ID ""
# endif

#else
#  define ARCHITECTURE_ID
#endif

/* Convert integer to decimal digit literals.  */
#define DEC(n)                   \
  ('0' + (((n) / 10000000)%10)), \
  ('0' + (((n) / 1000000)%10)),  \
  ('0' + (((n) / 100000)%10)),   \
  ('0' + (((n) / 10000)%10)),    \
  ('0' + (((n) / 1000)%10)),     \
  ('0' + (((n) / 100)%10)),      \
  ('0' + (((n) / 10)%10)),       \
  ('0' +  ((n) % 10))

/* Convert integer to hex digit literals.  */
#define HEX(n)             \
  ('0' + ((n)»28 & 0xF)), \
  ('0' + ((n)»24 & 0xF)), \
  ('0' + ((n)»20 & 0xF)), \
  ('0' + ((n)»16 & 0xF)), \
  ('0' + ((n)»12 & 0xF)), \
  ('0' + ((n)»8  & 0xF)), \
  ('0' + ((n)»4  & 0xF)), \
  ('0' + ((n)     & 0xF))

/* Construct a string literal encoding the version number. */
#ifdef COMPILER_VERSION
char const* info_version = "INFO" ":" "compiler_version[" COMPILER_VERSION "]";

/* Construct a string literal encoding the version number components. */
#elif defined(COMPILER_VERSION_MAJOR)
char const info_version[] = {
  'I', 'N', 'F', 'O', ':',
  'c','o','m','p','i','l','e','r','_','v','e','r','s','i','o','n','[',
  COMPILER_VERSION_MAJOR,
# ifdef COMPILER_VERSION_MINOR
  '.', COMPILER_VERSION_MINOR,
#  ifdef COMPILER_VERSION_PATCH
   '.', COMPILER_VERSION_PATCH,
#   ifdef COMPILER_VERSION_TWEAK
    '.', COMPILER_VERSION_TWEAK,
#   endif
#  endif
# endif
  ']','\0'};
#endif

/* Construct a string literal encoding the internal version number. */
#ifdef COMPILER_VERSION_INTERNAL
char const info_version_internal[] = {
  'I', 'N', 'F', 'O', ':',
  'c','o','m','p','i','l','e','r','_','v','e','r','s','i','o','n','_',
  'i','n','t','e','r','n','a','l','[',
  COMPILER_VERSION_INTERNAL,']','\0'};
#elif defined(COMPILER_VERSION_INTERNAL_STR)
char const* info_version_internal = "INFO" ":" "compiler_version_internal["
 COMPILER_VERSION_INTERNAL_STR "]";
#endif

/* Construct a string literal encoding the version number components. */
#ifdef SIMULATE_VERSION_MAJOR
char const info_simulate_version[] = {
  'I', 'N', 'F', 'O', ':',
  's','i','m','u','l','a','t','e','_','v','e','r','s','i','o','n','[',
  SIMULATE_VERSION_MAJOR,
# ifdef SIMULATE_VERSION_MINOR
  '.', SIMULATE_VERSION_MINOR,
#  ifdef SIMULATE_VERSION_PATCH
   '.', SIMULATE_VERSION_PATCH,
#   ifdef SIMULATE_VERSION_TWEAK
    '.', SIMULATE_VERSION_TWEAK,
#   endif
#  endif
# endif
  ']','\0'};
```

```
#endif

/* Construct the string literal in pieces to prevent the source from
   getting matched.  Store it in a pointer rather than an array
   because some compilers will just produce instructions to fill the
   array rather than assigning a pointer to a static array.  */
char const* info_platform = "INFO" ":" "platform[" PLATFORM_ID "]";
char const* info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]";



#if defined(__INTEL_COMPILER) && defined(_MSVC_LANG) && _MSVC_LANG < 201403L
#  if defined(__INTEL_CXX11_MODE__)
#    if defined(__cpp_aggregate_nsdmi)
#      define CXX_STD 201402L
#    else
#      define CXX_STD 201103L
#    endif
#  else
#    define CXX_STD 199711L
#  endif
#elif defined(_MSC_VER) && defined(_MSVC_LANG)
#  define CXX_STD _MSVC_LANG
#else
#  define CXX_STD __cplusplus
#endif

const char* info_language_dialect_default = "INFO" ":" "dialect_default["
#if CXX_STD > 202002L
  "23"
#elif CXX_STD > 201703L
  "20"
#elif CXX_STD >= 201703L
  "17"
#elif CXX_STD >= 201402L
  "14"
#elif CXX_STD >= 201103L
  "11"
#else
  "98"
#endif
"]";

/*--------------------------------------------------------------------------*/

int main(int argc, char* argv[])
{
  int require = 0;
  require += info_compiler[argc];
  require += info_platform[argc];
#ifdef COMPILER_VERSION_MAJOR
  require += info_version[argc];
#endif
#ifdef COMPILER_VERSION_INTERNAL
  require += info_version_internal[argc];
#endif
#ifdef SIMULATE_ID
  require += info_simulate[argc];
#endif
#ifdef SIMULATE_VERSION_MAJOR
  require += info_simulate_version[argc];
#endif
#if defined(__CRAYXT_COMPUTE_LINUX_TARGET)
  require += info_cray[argc];
#endif
  require += info_language_dialect_default[argc];
  (void)argv;
  return require;
}
```

6.5 D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/↵ CMakeFiles/FindOpenMP/OpenMPCheckVersion.c File Reference

```
#include <stdio.h>
#include <omp.h>
```

Include dependency graph for OpenMPCheckVersion.c:


D:/Documents/Final/Mandelbrot
_Cpp_Final/cmake-build-debug
/CMakeFiles/FindOpenMP/OpenMPCheckVersion.c
stdio.h
omp.h

Functions

- int main (void)

Variables

- const char ompver_str [ ]

6.5.1 Function Documentation

6.5.1.1 main()

```
int main (
            void )
```

Definition at line 13 of file OpenMPCheckVersion.c.

```
00014 {
00015   puts(ompver_str);
00016   return 0;
00017 }
```

6.5.2 Variable Documentation

6.5.2.1 ompver_str

```
const char ompver_str[]
```

Initial value:

```
= { 'I', 'N', 'F', 'O', ':', 'O', 'p', 'e', 'n', 'M',
                                'P', '-', 'd', 'a', 't', 'e', '[',
                                ('0' + ((_OPENMP/100000)%10)),
                                ('0' + ((_OPENMP/10000)%10)),
                                ('0' + ((_OPENMP/1000)%10)),
                                ('0' + ((_OPENMP/100)%10)),
                                ('0' + ((_OPENMP/10)%10)),
                                ('0' + ((_OPENMP/1)%10)),
                                ']', '\0' }
```

Definition at line 4 of file OpenMPCheckVersion.c.

6.6 OpenMPCheckVersion.c

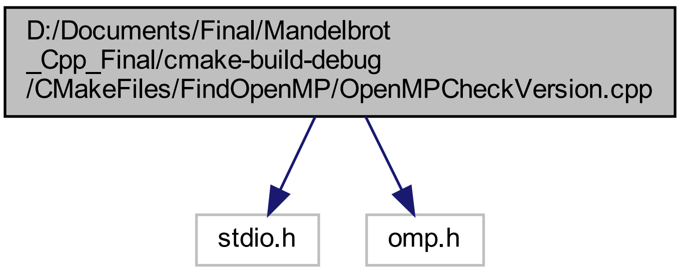
Go to the documentation of this file.

```

#include <stdio.h>
#include <omp.h>
const char ompver_str[] = { 'I', 'N', 'F', 'O', ':', 'O', 'p', 'e', 'n', 'M',
                            'P', '-', 'd', 'a', 't', 'e', '[',
                            ('0' + ((_OPENMP/100000)%10)),
                            ('0' + ((_OPENMP/10000)%10)),
                            ('0' + ((_OPENMP/1000)%10)),
                            ('0' + ((_OPENMP/100)%10)),
                            ('0' + ((_OPENMP/10)%10)),
                            ('0' + ((_OPENMP/1)%10)),
                            ']', '\0' };
int main(void)
{
  puts(ompver_str);
  return 0;
}
```

6.7 D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/↩ CMakeFiles/FindOpenMP/OpenMPCheckVersion.cpp File Reference

```
#include <stdio.h>
#include <omp.h>
```

Include dependency graph for OpenMPCheckVersion.cpp:


D:/Documents/Final/Mandelbrot
_Cpp_Final/cmake-build-debug
/CMakeFiles/FindOpenMP/OpenMPCheckVersion.cpp
stdio.h
omp.h

Functions

- int main (void)

Variables

- const char ompver_str [ ]

6.7.1 Function Documentation

6.7.1.1 main()

```
int main (
              void  )
```

Definition at line 13 of file OpenMPCheckVersion.cpp.

```
00014 {
00015   puts(ompver_str);
00016   return 0;
00017 }
```

6.7.2 Variable Documentation

6.7.2.1 ompver_str

```
const char ompver_str[]
```

Initial value:

```
= { 'I', 'N', 'F', 'O', ':', 'O', 'p', 'e', 'n', 'M',
                            'P', '-', 'd', 'a', 't', 'e', '[',
                            ('0' + ((_OPENMP/100000)%10)),
                            ('0' + ((_OPENMP/10000)%10)),
                            ('0' + ((_OPENMP/1000)%10)),
                            ('0' + ((_OPENMP/100)%10)),
                            ('0' + ((_OPENMP/10)%10)),
                            ('0' + ((_OPENMP/1)%10)),
                            ']', '\0' }
```

Definition at line 4 of file OpenMPCheckVersion.cpp.

6.8 OpenMPCheckVersion.cpp

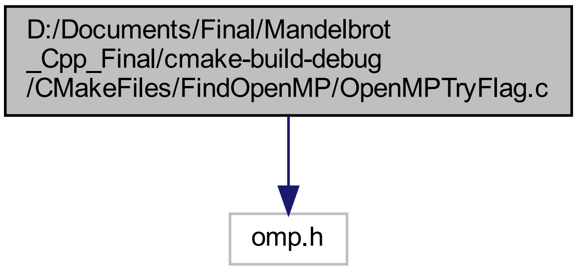
Go to the documentation of this file.

```
00001
00002 #include <stdio.h>
00003 #include <omp.h>
00004 const char ompver_str[] = { 'I', 'N', 'F', 'O', ':', 'O', 'p', 'e', 'n', 'M',
00005                             'P', '-', 'd', 'a', 't', 'e', '[',
00006                             ('0' + ((_OPENMP/100000)%10)),
00007                             ('0' + ((_OPENMP/10000)%10)),
00008                             ('0' + ((_OPENMP/1000)%10)),
00009                             ('0' + ((_OPENMP/100)%10)),
00010                             ('0' + ((_OPENMP/10)%10)),
00011                             ('0' + ((_OPENMP/1)%10)),
00012                             ']', '\0' };
00013 int main(void)
00014 {
00015   puts(ompver_str);
00016   return 0;
00017 }
```

6.9 D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/↩ CMakeFiles/FindOpenMP/OpenMPTryFlag.c File Reference

```
#include <omp.h>
```

Include dependency graph for OpenMPTryFlag.c:


D:/Documents/Final/Mandelbrot
_Cpp_Final/cmake-build-debug
/CMakeFiles/FindOpenMP/OpenMPTryFlag.c
omp.h

Functions

- int main (void)

6.9.1 Function Documentation

6.9.1.1 main()

```
int main (
            void )
```

Definition at line 3 of file OpenMPTryFlag.c.

```
                  {
#ifdef _OPENMP
  omp_get_max_threads();
  return 0;
#elif defined(__HIP_DEVICE_COMPILE__)
  return 0;
#else
  breaks_on_purpose
#endif
}
```

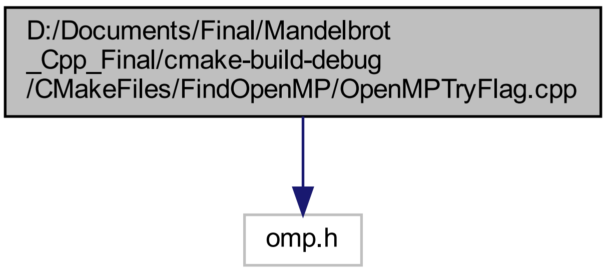
6.10 OpenMPTryFlag.c

Go to the documentation of this file.

```

#include <omp.h>
int main(void) {
#ifdef _OPENMP
  omp_get_max_threads();
  return 0;
#elif defined(__HIP_DEVICE_COMPILE__)
  return 0;
#else
  breaks_on_purpose
#endif
}
```

6.11 D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/↩ CMakeFiles/FindOpenMP/OpenMPTryFlag.cpp File Reference

`#include <omp.h>`
Include dependency graph for OpenMPTryFlag.cpp:


D:/Documents/Final/Mandelbrot
_Cpp_Final/cmake-build-debug
/CMakeFiles/FindOpenMP/OpenMPTryFlag.cpp
omp.h

Functions

- int main (void)

6.11.1 Function Documentation

6.11.1.1 main()

```
int main (
            void  )
```

Definition at line 3 of file OpenMPTryFlag.cpp.

```
00003                {
00004 #ifdef _OPENMP
00005   omp_get_max_threads();
00006   return 0;
00007 #elif defined(__HIP_DEVICE_COMPILE__)
00008   return 0;
00009 #else
00010   breaks_on_purpose
00011 #endif
00012 }
```

6.12 OpenMPTryFlag.cpp

Go to the documentation of this file.

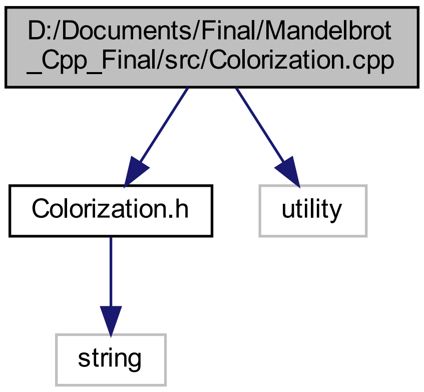
```
00001
00002 #include <omp.h>
00003 int main(void) {
00004 #ifdef _OPENMP
00005   omp_get_max_threads();
00006   return 0;
00007 #elif defined(__HIP_DEVICE_COMPILE__)
00008   return 0;
00009 #else
00010   breaks_on_purpose
00011 #endif
00012 }
```

6.13 D:/Documents/Final/Mandelbrot_Cpp_Final/README.md File Reference

6.14 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Colorization.cpp File Reference

```
#include "Colorization.h"
#include <utility>
```

Include dependency graph for Colorization.cpp:


D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Colorization.cpp
Colorization.h
utility
string

6.15 Colorization.cpp

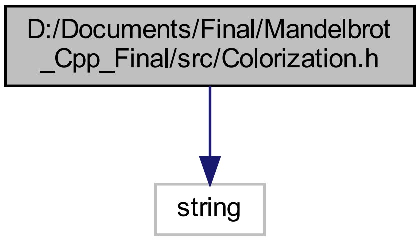
Go to the documentation of this file.

```
#include "Colorization.h"

#include <utility>

using namespace std;

Colorization::Colorization(string type) : type(std::move(type))
{}

std::string Colorization::get_type()
{
  return type;
}

Colorization::~Colorization() = default;
```

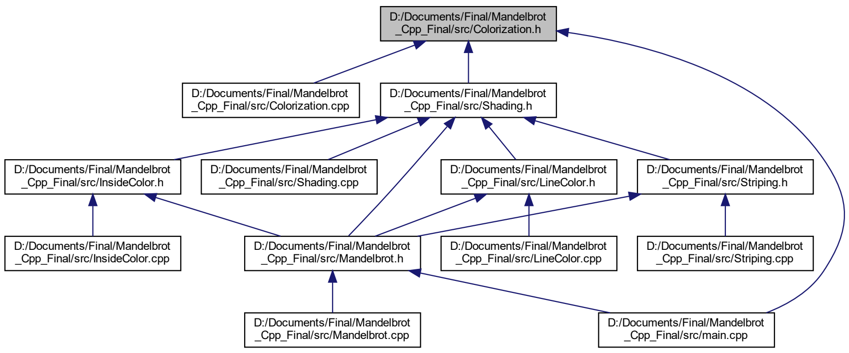
6.16 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Colorization.h File Reference

`#include <string>`

Include dependency graph for Colorization.h:



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


D:/Documents/Final/Mandelbrot_Cpp_Final/src/Colorization.h
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Colorization.cpp
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Shading.h
D:/Documents/Final/Mandelbrot_Cpp_Final/src/InsideColor.h
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Shading.cpp
D:/Documents/Final/Mandelbrot_Cpp_Final/src/LineColor.h
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Striping.h
D:/Documents/Final/Mandelbrot_Cpp_Final/src/InsideColor.cpp
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Mandelbrot.h
D:/Documents/Final/Mandelbrot_Cpp_Final/src/LineColor.cpp
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Striping.cpp
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Mandelbrot.cpp
D:/Documents/Final/Mandelbrot_Cpp_Final/src/main.cpp

Classes

- class Colorization

6.17 Colorization.h

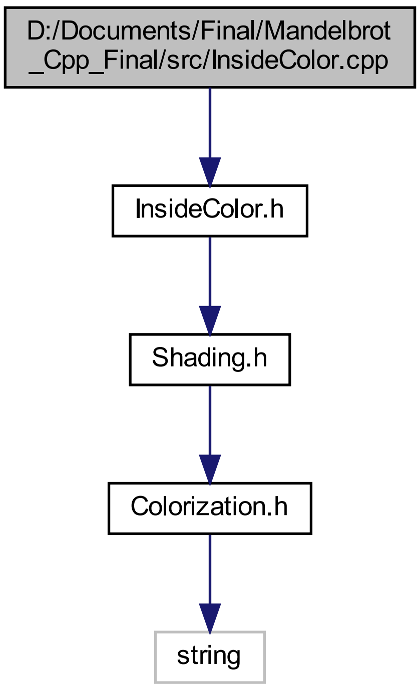
Go to the documentation of this file.

```
#ifndef C___COLORIZATION_H
#define C___COLORIZATION_H

#include <string>

class Colorization
{
 public:
    std::string get_type();

    virtual unsigned char get_max_color_value() = 0;

    virtual unsigned char get_min_color_value() = 0;

    virtual ~Colorization();

protected:

    explicit Colorization(std::string type);

    std::string type;

    const unsigned char maxColorValue = 255;

    const unsigned char minColorValue = 0;

};

#endif  // C___COLORIZATION_H
```

6.18 D:/Documents/Final/Mandelbrot_Cpp_Final/src/InsideColor.cpp File Reference

`#include "InsideColor.h"`

Include dependency graph for InsideColor.cpp:



6.19 InsideColor.cpp

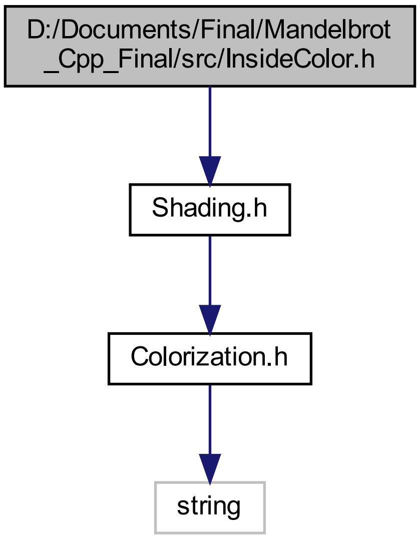
Go to the documentation of this file.

```
#include "InsideColor.h"

InsideColor::InsideColor() : Shading("Inside")
{}

InsideColor::~InsideColor()
= default;

unsigned char InsideColor::calculate_bw()
{
  return minColorValue;
}

unsigned char InsideColor::calculate_r()
{
  return minColorValue;
}

unsigned char InsideColor::calculate_g()
{
  return minColorValue;
}

unsigned char InsideColor::calculate_b()
{
  return minColorValue;
}
```

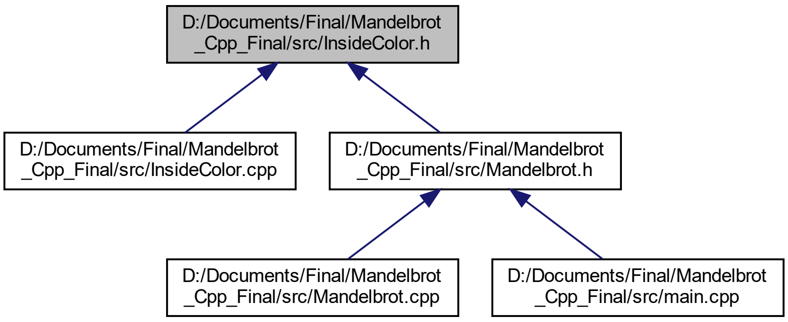
6.20 D:/Documents/Final/Mandelbrot_Cpp_Final/src/InsideColor.h File Reference

```
#include "Shading.h"
```

Include dependency graph for InsideColor.h:



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


D:/Documents/Final/Mandelbrot_Cpp_Final/src/InsideColor.h
D:/Documents/Final/Mandelbrot_Cpp_Final/src/InsideColor.cpp
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Mandelbrot.h
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Mandelbrot.cpp
D:/Documents/Final/Mandelbrot_Cpp_Final/src/main.cpp

Classes

- class InsideColor

6.21 InsideColor.h

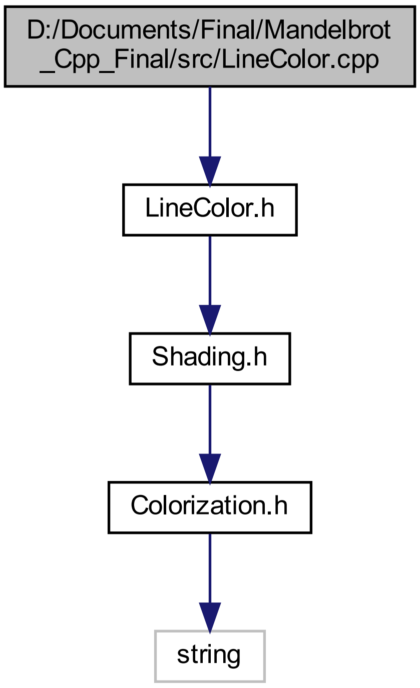
Go to the documentation of this file.

```
#ifndef C___INSIDECOLOR_H
#define C___INSIDECOLOR_H

#include "Shading.h"

class InsideColor : public Shading
{
 public:

  InsideColor();

  ~InsideColor();

  unsigned char calculate_bw();

  unsigned char calculate_r();

  unsigned char calculate_g();

  unsigned char calculate_b();
};

#endif //C___INSIDECOLOR_H
```

6.22 D:/Documents/Final/Mandelbrot_Cpp_Final/src/LineColor.cpp File Reference

`#include "LineColor.h"`

Include dependency graph for LineColor.cpp:



6.23 LineColor.cpp

Go to the documentation of this file.

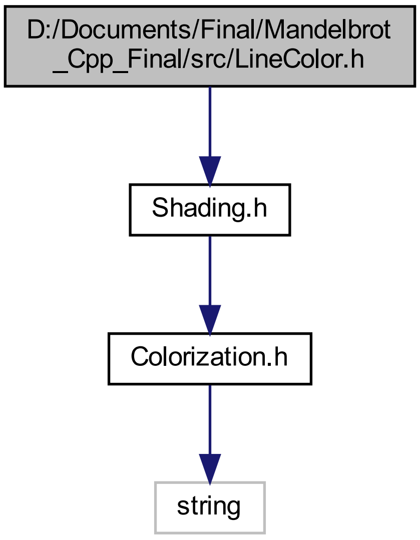
```
#include "LineColor.h"

LineColor::LineColor() : Shading("Line")
{}

LineColor::~LineColor()
= default;

unsigned char LineColor::calculate_bw()
{
  return maxColorValue;
}

unsigned char LineColor::calculate_r()
{
  return maxColorValue;
}

unsigned char LineColor::calculate_g()
{
  return maxColorValue;
}

```

```
unsigned char LineColor::calculate_b()
{
  return maxColorValue;
}
```

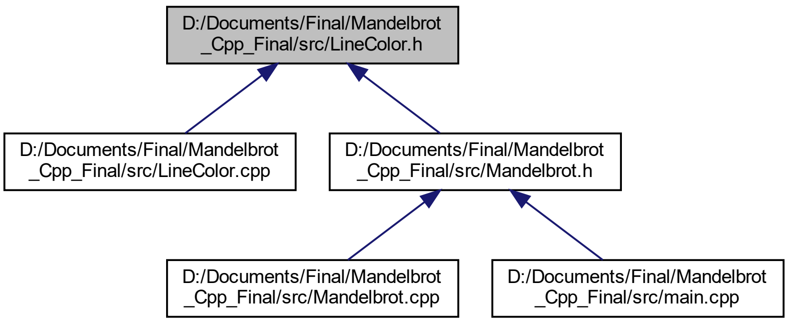
6.24 D:/Documents/Final/Mandelbrot_Cpp_Final/src/LineColor.h File Reference

`#include "Shading.h"`

Include dependency graph for LineColor.h:



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


D:/Documents/Final/Mandelbrot
_Cpp_Final/src/LineColor.h
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/LineColor.cpp
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Mandelbrot.h
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Mandelbrot.cpp
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/main.cpp

Classes

- class LineColor

6.25 LineColor.h

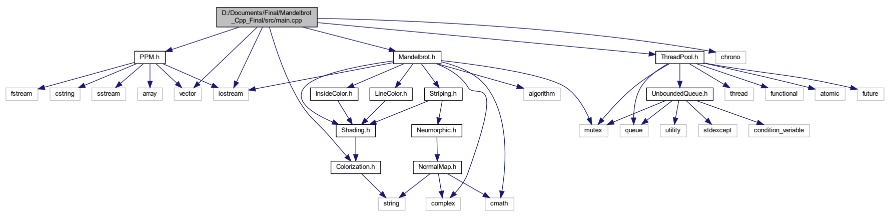
Go to the documentation of this file.

```
#ifndef C___LINECOLOR_H
#define C___LINECOLOR_H

#include "Shading.h"

class LineColor : public Shading
{
public:

    LineColor();

    ~LineColor();

    unsigned char calculate_bw();

    unsigned char calculate_r();

    unsigned char calculate_g();

    unsigned char calculate_b();
};

#endif //C___LINECOLOR_H
```

6.26 D:/Documents/Final/Mandelbrot_Cpp_Final/src/main.cpp File Reference

```
#include "PPM.h"
#include "Colorization.h"
#include "Mandelbrot.h"
#include "ThreadPool.h"
#include <chrono>
#include <iostream>
#include <vector>
```

Include dependency graph for main.cpp:



Functions

- int main ()

6.26.1 Function Documentation

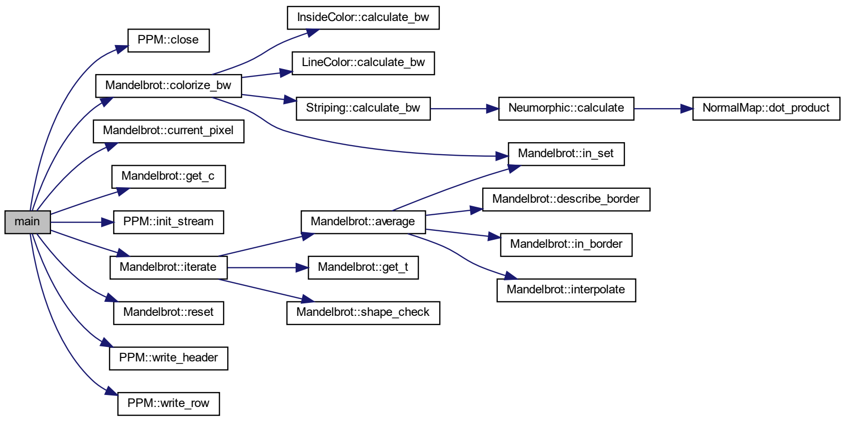
6.26.1.1 main()

```
int main (
            void  )
```

Definition at line 12 of file main.cpp.

```
{
  int width;
  int height;

  // set up input handling to prevent creation of image with distortion for
  // given default Mandelbrot parameters
  bool invalidInput = true;
  while (invalidInput)
  {
    cout « "Enter square image size in pixels:\n";
    string line;
    getline(cin, line);
    istringstream is(line);

    char dummy;
    if (!(is » width) || (is » ws && is.get(dummy)) || !(width > 0))
    {
      cout « "Invalid input. Try again:\n";
    } else {
      invalidInput = false;
    }
  }
  height = width;


  // handle invalid filename inputs
  string fileName;
  bool invalidChar = true;
  while (invalidChar)
  {
    cout « "Enter output file name in the form 'name.ppm':\n";
    cin » fileName;
    for (const char c : fileName)
    {
      if (!isalnum(c) && !ispunct(c))
      {
        cout « "Invalid character found. Try again:\n";
        break;
      } else {
        invalidChar = false;
        break;
      }
    }
  }

  auto begin = chrono::steady_clock::now();

  // set up image stream for writing
  PPM pgm(fileName, width, height);
  // PPM pgm(width, height); // for testing
  if (!pgm.init_stream())
  {
    cout « "Could not open ofstream for image\n";
    return -1;
  }
  pgm.write_header();

  // set up container for image row data
  vector<unsigned char> row{}; // array needs compile-time const length
  row.resize(width * 3); // avoid constant resizing by allocating up front

  cout « "Rendering row by row:\n";

  // using basic constructor since I only want this image
  // call member functions here to set specific parameters
  Mandelbrot gigabrot(width, height);
  cout « gigabrot;

  // unsigned int numThreads = thread::hardware_concurrency();
  // cout « "numThreads: " « numThreads « "\n";

  for (size_t pY = 0; pY < height; pY++)
  {
    for (size_t pX = 0; pX < width; pX++)
    {
      size_t subPixel = 3 * pX;
      gigabrot.current_pixel(pX, pY);
      gigabrot.get_c();
      gigabrot.iterate();
      row[subPixel + 2] = row[subPixel + 1] = row[subPixel] =
          gigabrot.colorize_bw();
      gigabrot.reset();
    }
    {
      // implemented due to possibility of having huge image, keep memory usage low
      // might be causing the issues with parallelization, ruining the embarrassingly parallel
      // aspect of the Mandelbrot set
      pgm.write_row(row);
    }
  }

  pgm.close();

  auto end = chrono::steady_clock::now();
  cout « "Time elapsed: "
       « static_cast<float>(chrono::duration_cast<chrono::milliseconds>
           (end - begin).count()) / 1000.F
       « " sec\n";

  return 0;
}
```

Here is the call graph for this function:



6.27 main.cpp

Go to the documentation of this file.

```
#include "PPM.h"
#include "Colorization.h"
#include "Mandelbrot.h"
#include "ThreadPool.h"

#include <chrono>
#include <iostream>
#include <vector>

using namespace std;

int main()
{
  int width;
  int height;
```

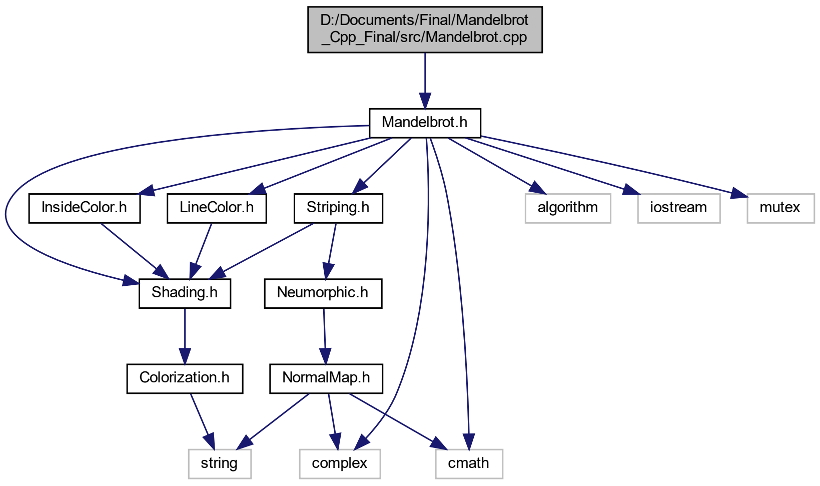
```

// set up input handling to prevent creation of image with distortion for
// given default Mandelbrot parameters
bool invalidInput = true;
while (invalidInput)
{
  cout « "Enter square image size in pixels:\n";
  string line;
  getline(cin, line);
  istringstream is(line);

  char dummy;
  if (!(is » width) || (is » ws && is.get(dummy)) || !(width > 0))
  {
    cout « "Invalid input. Try again:\n";
  } else {
    invalidInput = false;
  }
}
height = width;


// handle invalid filename inputs
string fileName;
bool invalidChar = true;
while (invalidChar)
{
  cout « "Enter output file name in the form 'name.ppm':\n";
  cin » fileName;
  for (const char c : fileName)
  {
    if (!isalnum(c) && !ispunct(c))
    {
      cout « "Invalid character found. Try again:\n";
      break;
    } else {
      invalidChar = false;
      break;
    }
  }
}

auto begin = chrono::steady_clock::now();

// set up image stream for writing
PPM pgm(fileName, width, height);
// PPM pgm(width, height); // for testing
if (!pgm.init_stream())
{
  cout « "Could not open ofstream for image\n";
  return -1;
}
pgm.write_header();

// set up container for image row data
vector<unsigned char> row{}; // array needs compile-time const length
row.resize(width * 3); // avoid constant resizing by allocating up front

cout « "Rendering row by row:\n";

// using basic constructor since I only want this image
// call member functions here to set specific parameters
Mandelbrot gigabrot(width, height);
cout « gigabrot;

// unsigned int numThreads = thread::hardware_concurrency();
// cout « "numThreads: " « numThreads « "\n";

for (size_t pY = 0; pY < height; pY++)
{
  for (size_t pX = 0; pX < width; pX++)
  {
    size_t subPixel = 3 * pX;
    gigabrot.current_pixel(pX, pY);
    gigabrot.get_c();
    gigabrot.iterate();
    row[subPixel + 2] = row[subPixel + 1] = row[subPixel] =
        gigabrot.colorize_bw();
    gigabrot.reset();
  }
  {
    // implemented due to possibility of having huge image, keep memory usage low
    // might be causing the issues with parallelization, ruining the embarrassingly parallel
    // aspect of the Mandelbrot set
    pgm.write_row(row);
  }
}
```

```
  pgm.close();

  auto end = chrono::steady_clock::now();
  cout « "Time elapsed: "
       « static_cast<float>(chrono::duration_cast<chrono::milliseconds>(
           (end - begin).count()) / 1000.F
       « " sec\n";

  return 0;
}
```

6.28 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Mandelbrot.cpp File Reference

`#include "Mandelbrot.h"`

Include dependency graph for Mandelbrot.cpp:



Functions

- std::ostream & operator<< (ostream &os, const Mandelbrot &mandelbrot)

6.28.1 Function Documentation

6.28.1.1 operator<<()

```
std::ostream & operator<< (
              ostream & os,
              const Mandelbrot & mandelbrot )
```

Definition at line 231 of file Mandelbrot.cpp.

```
{
  double pixAspectRatio = (static_cast<double>(mandelbrot.width) / static_cast<double>(mandelbrot
      .height));
  double worldAspectRatio = (mandelbrot.cxMax - mandelbrot.cxMin) / (mandelbrot.cyMax -
      mandelbrot.cyMin);
  double distortion = pixAspectRatio - worldAspectRatio;
  os « "Distortion (should be 0): " « distortion « "\n";
  return os;
}
```

6.29 Mandelbrot.cpp

Go to the documentation of this file.

```
#include "Mandelbrot.h"

using namespace std;

Mandelbrot::Mandelbrot(int width, int height) : width(width), height(height)
{
  pX = 0;
  pY = 0;
  iter = 0;
  iterMax = 1000;
  escapeRadius = 1000000.0;
  // lnER = log(escapeRadius);
  c = 0.0;
  r = 0.0;
  z = 0.0;
  dC = 0.0;
  q = 0.0;
  cardioid = 0.0;
  a = 0.0;
  prevA = 0.0;
  stripeDensity = 7.0;
  d = 0.0;
  de = 0.0;
  cxMin = -2.2;
  cxMax = 0.8;
  cyMin = -1.5;
  cyMax = 1.5;
  pixWidth = 0.0;
  pixHeight = 0.0;
  iSkip = 1;
  thin = 3;
  shade = nullptr; // avoid calling "new" more than once per pixel
}

Mandelbrot::Mandelbrot(int pX, int pY, int width, int height) : pX(pX), pY(pY), width(width),
height(height)
{
  iter = 0;
  iterMax = 1000;
  escapeRadius = 1000000.0;
  // lnER = log(escapeRadius);
  c = 0.0;
  r = 0.0;
  z = 0.0;
  dC = 0.0;
  q = 0.0;
  cardioid = 0.0;
  a = 0.0;
  prevA = 0.0;
  stripeDensity = 7.0;
  d = 0.0;
  de = 0.0;
  cxMin = -2.2;
  cxMax = 0.8;
  cyMin = -1.5;
  cyMax = 1.5;
  pixWidth = 0.0;
  pixHeight = 0.0;
  iSkip = 1;
  thin = 3;
  shade = nullptr; // avoid calling "new" more than once per pixel
}

Mandelbrot::~Mandelbrot()
{
  delete shade;
}

```

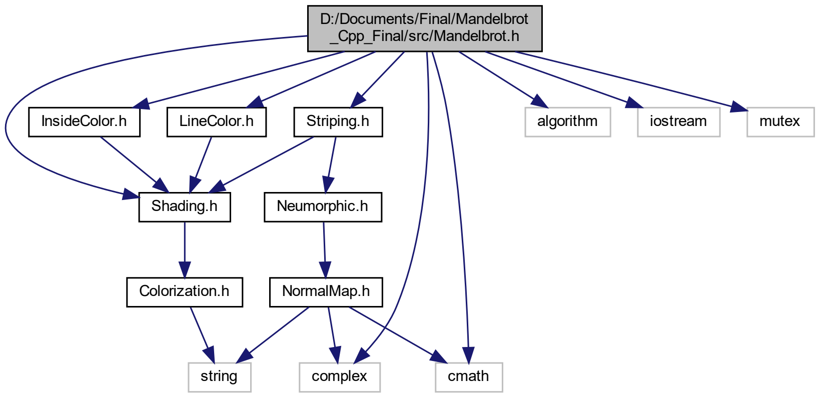
```
void Mandelbrot::set_image(int widthIn, int heightIn)
{
  width = widthIn;
  height = heightIn;
}

void Mandelbrot::current_pixel(int pxIn, int pyIn)
{
  pX = pxIn;
  pY = pyIn;
}

void Mandelbrot::set_plane(double cxMinIn, double cxMaxIn, double cyMinIn, double cYMaxIn)
{
  cxMin = cxMinIn;
  cxMax = cxMaxIn;
  cyMin = cyMinIn;
  cyMax = cYMaxIn;
}

void Mandelbrot::set_stripe_density(double stripeDensityIn)
{
  stripeDensity = stripeDensityIn;
}

void Mandelbrot::set_iSkip(int iSkipIn)
{
  iSkip = iSkipIn;
}

void Mandelbrot::set_border(int thinIn)
{
  thin = thinIn;
}

void Mandelbrot::get_c()
{
  pixWidth = (cxMax-cxMin) / static_cast<double>(width);
  pixHeight = (cyMax-cyMin) / static_cast<double>(height);
  c = (cxMin + static_cast<double>(pX) * pixWidth) + ((cyMax - static_cast<double>(pY) * pixHeight) *
 1i);
}

void Mandelbrot::iterate()
{
  if (!this->shape_check())
  {
    for (iter = 0; iter < iterMax; iter++)
    {
      // mandelbrot set formula
      dC = 2.0 * dC * z + 1.0;
      z = z * z + c;

      // compute average
      if (iter > iSkip)
      {
        a += get_t();
      }

      r = abs(z);
      if (r > escapeRadius)
      {
        break;
      }

      prevA = a;
    }

    average();
  }
}

unsigned char Mandelbrot::colorize_bw()
{
  if (in_set())
  {
    shade = new InsideColor();
    InsideColor *color;
    color = dynamic_cast<InsideColor*>(shade);
    return color->calculate_bw();
  } else if (a == FP_ZERO) {
    shade = new LineColor();
    LineColor *color;
    color = dynamic_cast<LineColor*>(shade);
    return color->calculate_bw();
  } else {
    shade = new Striping(a, z, dC);
    Striping *color;
    color = dynamic_cast<Striping*>(shade);
    return color->calculate_bw();
  }
}

bool Mandelbrot::shape_check()
{
    q = ((real(c) - 0.25) * (real(c) - 0.25)) + (imag(c) * imag(c));
    cardioid = 0.25 * imag(c) * imag(c);
    if ((real(c) * real(c) + 2.0 * real(c) + 1.0 + imag(c) * imag(c)) < bulb ||
        (q * (q +(real(c) - 0.25)) < cardioid))
    {
        return true;
    } else {
        return false;
    }
}

double Mandelbrot::get_t()
{
  return 0.5 + 0.5 * sin(stripeDensity * arg(z));
}

void Mandelbrot::interpolate()
{
  // smooth iteration count
  d = static_cast<double>(iter + 1) + log(log(escapeRadius) / log(r)) / M_LN2;
  d = d - static_cast<double>(static_cast<int>(d)); // only fractional part = interpolation
  // coefficient
}

void Mandelbrot::average()
{
  if (in_set())
  {
    a = -1.0;
  } else {
    describe_border();
    if (in_border()) // in border
    {
      a = FP_ZERO;
    } else {
      a /= static_cast<double>((iter - iSkip)); // A(n)
      prevA /= static_cast<double>((iter - iSkip - 1)); // A(n-1)
      this->interpolate();
      a = (d * a) + ((1.0 - d) * prevA);
    }
  }
}

void Mandelbrot::describe_border()
{
  de = 2.0 * r * log(r) / abs(dC);
}

bool Mandelbrot::in_border()
{
  if (de < (pixWidth / static_cast<double>(thin)))
  {
    return true;
  } else {
    return false;
  }
}

bool Mandelbrot::in_set()
{
  if (iter == iterMax)
  {
    return true;
  } else {
    return false;
  }
}

std::ostream &operator«(ostream &os, const Mandelbrot& mandelbrot)
{
  double pixAspectRatio = (static_cast<double>(mandelbrot.width) / static_cast<double>(mandelbrot
      .height));
  double worldAspectRatio = (mandelbrot.cxMax - mandelbrot.cxMin) / (mandelbrot.cyMax -
      mandelbrot.cyMin);
  double distortion = pixAspectRatio - worldAspectRatio;
  os « "Distortion (should be 0): " « distortion « "\n";
  return os;
}

```

```
void Mandelbrot::reset()
{
  c = 0.0;
  r = 0.0;
  z = 0.0;
  dC = 0.0;
  q = 0.0;
  cardioid = 0.0;
  a = 0.0;
  prevA = 0.0;
  d = 0.0;
  shade = nullptr; // avoid calling "new" more than once per pixel
}

Mandelbrot::Mandelbrot(const Mandelbrot &oldMandelbrot) : Mandelbrot(oldMandelbrot.width,
                                                                       oldMandelbrot.height)
{}


```

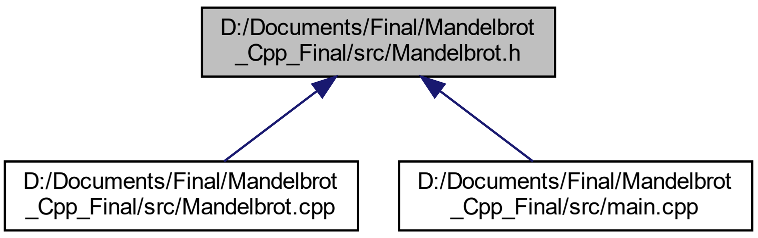
6.30 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Mandelbrot.h File Reference

```
#include "Shading.h"
#include "InsideColor.h"
#include "LineColor.h"
#include "Striping.h"
#include <cmath>
#include <complex>
#include <algorithm>
#include <iostream>
#include <mutex>
```

Include dependency graph for Mandelbrot.h:



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


D:/Documents/Final/Mandelbrot _Cpp_Final/src/Mandelbrot.h
D:/Documents/Final/Mandelbrot _Cpp_Final/src/Mandelbrot.cpp
D:/Documents/Final/Mandelbrot _Cpp_Final/src/main.cpp

Classes

- class Mandelbrot

Macros

- #define M_PI 3.14159265358979323846

6.30.1 Macro Definition Documentation

6.30.1.1 M_PI

```
#define M_PI 3.14159265358979323846
```

Definition at line 14 of file Mandelbrot.h.

6.31 Mandelbrot.h

Go to the documentation of this file.

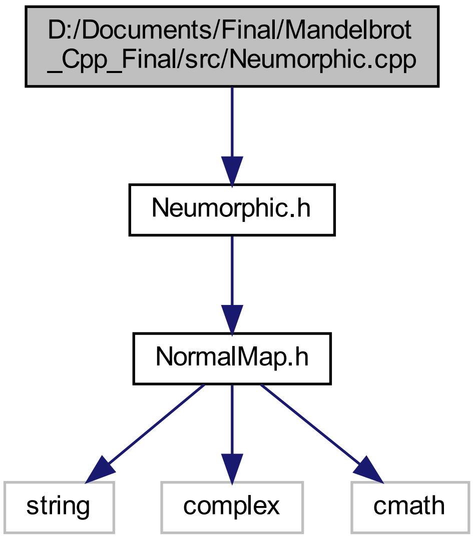
```
#ifndef C___MANDELBROT_H
#define C___MANDELBROT_H

#include "Shading.h"
#include "InsideColor.h"
#include "LineColor.h"
#include "Striping.h"
#include <cmath>
#include <complex>
#include <algorithm>
#include <iostream>
#include <mutex>

#define M_PI 3.14159265358979323846

class Mandelbrot
{
```

```
public:

 Mandelbrot(int width, int height);

 Mandelbrot(int pX, int pY, int width, int height);

 Mandelbrot(const Mandelbrot &oldMandelbrot);

 ~Mandelbrot();

 void current_pixel(int pxIn, int pyIn);

 void set_image(int widthIn, int heightIn);

 void set_plane(double cxMinIn, double cxMaxIn, double cyMinIn, double cYMaxIn);

 void set_stripe_density(double stripeDensityIn);

 void set_iSkip(int iSkipIn);

 void set_border(int thinIn);

 void get_c();

 void iterate();

 unsigned char colorize_bw();

 bool shape_check();

 double get_t();

 void interpolate();

 void average();

 void describe_border();

 bool in_border();

 bool in_set(); // for readability

 friend std::ostream& operator«(std::ostream& os, const Mandelbrot& mandelbrot);

 void reset();


private:

 int iter;

 int iterMax;

 double escapeRadius;

 // double lnER;

 std::complex<double> c;

 double r;

 std::complex<double> z;

 std::complex<double> dC;

 double q;

 double cardioid;

 const double bulb = 0.0625;

 // coordinate plane
 double cxMin;

 double cxMax;

 double cyMin;

 double cyMax;

 // image
 int width;

 int height;

 int pX;

```

```
    int pY;

    double pixWidth;

    double pixHeight;

    // average
    double a;

    double prevA;

    double stripeDensity;

    int iSkip;

    // interpolated
    double d;

    // boundary descriptor
    double de;
    int thin;

    Shading *shade;
};

#endif //C___MANDELBROT_H
```

6.32 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Neumorphic.cpp File Reference

`#include "Neumorphic.h"`
Include dependency graph for Neumorphic.cpp:



6.33 Neumorphic.cpp

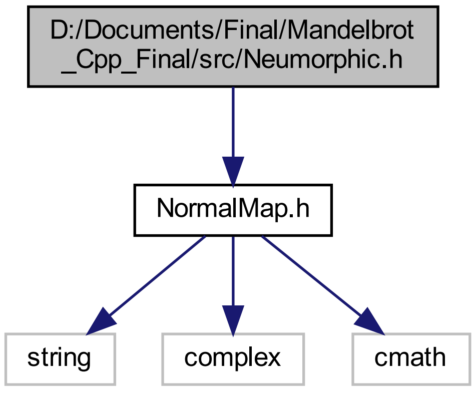
Go to the documentation of this file.

```
#include "Neumorphic.h"

using namespace std;

Neumorphic::Neumorphic(complex<double> z, complex<double> dC) : NormalMap("Neumorphic"), z(z), dC
(dC)
{
  heightFactor = 1.5;
  angle = 45.0 / 360.0;
  reflection = FP_ZERO;
  v = exp(2.0 * angle * M_PI * 1i);
}

double Neumorphic::calculate()
{
  u = z / dC;
  u = u / abs(u); // normalize
  reflection = dot_product(u, v) + heightFactor;
  reflection = reflection / (1.0 + heightFactor); // rescale so that it does not get bigger than 1
  if (reflection < 0.0)
  {
    reflection = 0.0;
  } else {}
  return reflection;
}

double Neumorphic::get_reflection()
{
  return reflection;
}

double Neumorphic::get_heightFactor()
{
  return heightFactor;
}

double Neumorphic::get_angle()
{
  return angle;
}
```

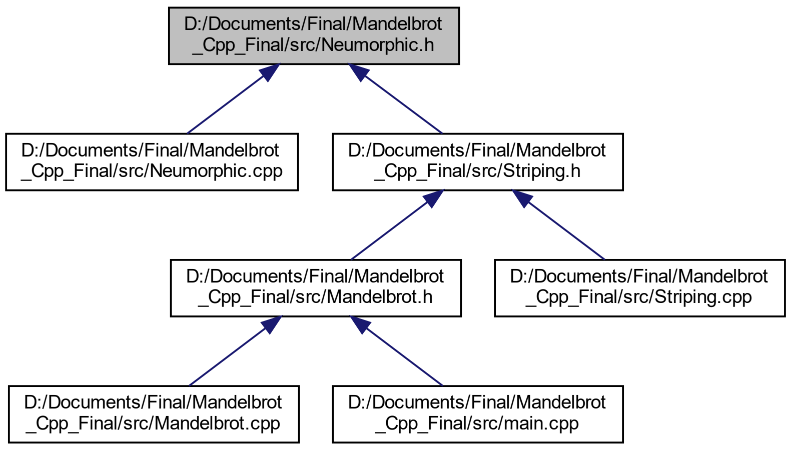
6.34 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Neumorphic.h File Reference

`#include "NormalMap.h"`

Include dependency graph for Neumorphic.h:



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


D:/Documents/Final/Mandelbrot_Cpp_Final/src/Neumorphic.h
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Neumorphic.cpp
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Striping.h
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Mandelbrot.h
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Striping.cpp
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Mandelbrot.cpp
D:/Documents/Final/Mandelbrot_Cpp_Final/src/main.cpp

Classes

- class Neumorphic

6.35 Neumorphic.h

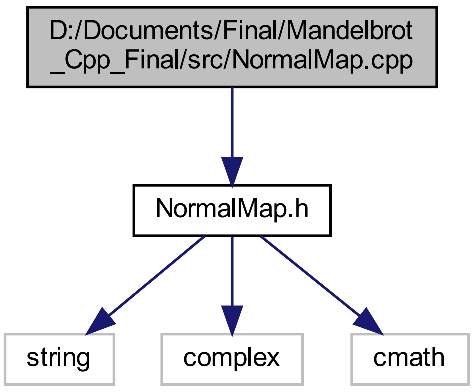
Go to the documentation of this file.

```
#ifndef C____NEUMORPHIC_H_
#define C____NEUMORPHIC_H_

#include "NormalMap.h"

class Neumorphic : public NormalMap
{
 public:

  Neumorphic(std::complex<double> z, std::complex<double> dC);

  double calculate();

  double get_reflection();

  double get_heightFactor();

  double get_angle();

 private:

  std::complex<double> z;

  std::complex<double> dC;

  double heightFactor;

  double angle;

  double reflection;

  std::complex<double> u;

  std::complex<double> v;

};

#endif //C____NEUMORPHIC_H_
```

6.36 D:/Documents/Final/Mandelbrot_Cpp_Final/src/NormalMap.cpp File Reference

`#include "NormalMap.h"`

Include dependency graph for NormalMap.cpp:



6.37 NormalMap.cpp

Go to the documentation of this file.

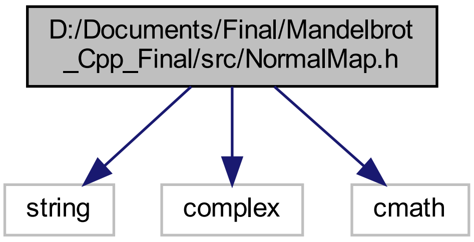
```
#include "NormalMap.h"

using namespace std;

NormalMap::NormalMap(string type)
{}

NormalMap::~NormalMap()
= default;

double NormalMap::dot_product(std::complex<double> u, std::complex<double> v)
{
  return real(u) * real(v) + imag(u) * imag(v);
}

double NormalMap::get_min_val()
{
  return minMapVal;
}

double NormalMap::get_max_val()
{
  return maxMapVal;
}
std::string NormalMap::get_type()
{
  return type;
}
```

6.38 D:/Documents/Final/Mandelbrot_Cpp_Final/src/NormalMap.h File Reference

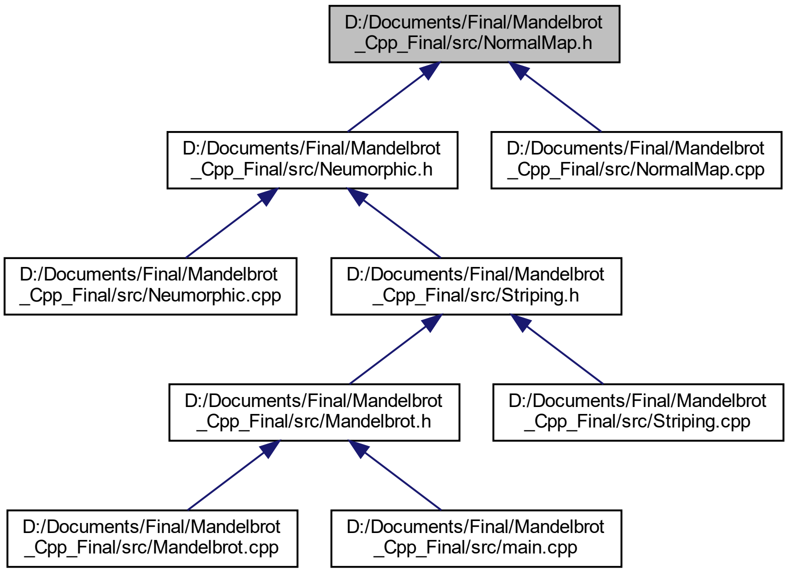
```
#include <string>
#include <complex>
```

```
#include <cmath>
```

Include dependency graph for NormalMap.h:



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


D:/Documents/Final/Mandelbrot
_Cpp_Final/src/NormalMap.h
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Neumorphic.h
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/NormalMap.cpp
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Neumorphic.cpp
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Striping.h
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Mandelbrot.h
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Striping.cpp
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Mandelbrot.cpp
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/main.cpp

Classes

- class NormalMap

6.39 NormalMap.h

Go to the documentation of this file.

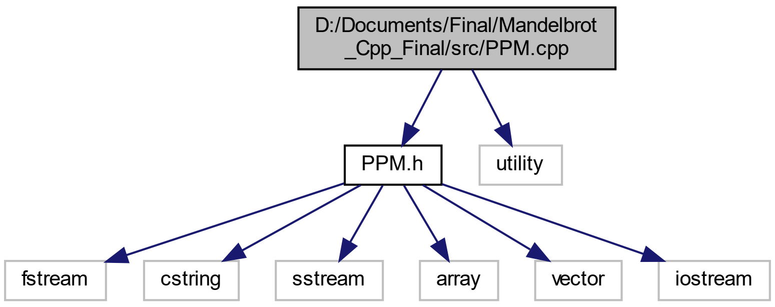
```
00001 #ifndef C___NORMALMAP_H
00002 #define C___NORMALMAP_H
```

```
#include <string>
#include <complex>
#include <cmath>

class NormalMap
{
 public:

  ~NormalMap();

  virtual double calculate() = 0;

  double dot_product(std::complex<double> u, std::complex<double> v);

  double get_min_val();

  double get_max_val();

  std::string get_type();

 protected:

  explicit NormalMap(std::string type);

 private:

  const double minMapVal = 0.0;

  const double maxMapVal = 1.0;

  std::string type;
};

#endif //C___NORMALMAP_H
```

6.40 D:/Documents/Final/Mandelbrot_Cpp_Final/src/PPM.cpp File Reference

`#include "PPM.h"`
`#include <utility>`

Include dependency graph for PPM.cpp:



6.41 PPM.cpp

Go to the documentation of this file.

```
#include "PPM.h"

```

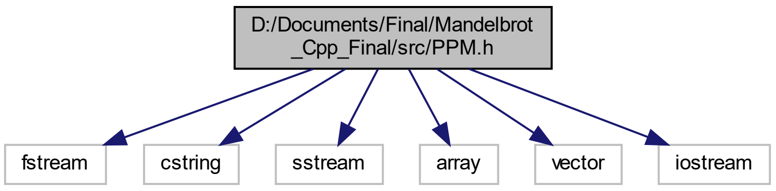
```
#include <utility>

using namespace std;

PPM::PPM(int width, int height) : width(width), height(height)
{
  outputDirectory = "..\\output\\";
  fileName = outputDirectory + "gigabrot_default.ppm";
  subPixel = width * 3;
}

PPM::PPM(const std::string &fileName, int width, int height) : width(width), height(height)
{
  outputDirectory = "..\\output\\";
  this->fileName = outputDirectory + fileName;
  subPixel = width * 3;
}

PPM::PPM(const PPM& oldPPM) : magic(oldPPM.magic), pixMaxVal(oldPPM.pixMaxVal),
    width(oldPPM.width), height(oldPPM.height), subPixel(oldPPM.subPixel), comment(oldPPM.comment)
{}

void PPM::set_outputDirectory(const std::string &outputDirectoryIn)
{
  outputDirectory = outputDirectoryIn;
}

PPM& PPM::operator=(string fileNameIn)
{
  this->fileName = std::move(fileNameIn);
  return *this;
}

bool PPM::init_stream()
{
  image.open(fileName, ios::binary);

  if (image.is_open())
  {
    return true;
  } else
  {
    return false;
  }
}

void PPM::write_header()
{
  string widthStr = to_string(this->width);
  string lengthStr = to_string(this->height);
  header << magic << widthStr << " " << lengthStr << "\n" << comment << "\n" << pixMaxVal;
  image << header.rdbuf();
}

void PPM::set_width(int widthIn)
{
  width = widthIn;
}

void PPM::set_height(int heightIn)
{
  height = heightIn;
}

void PPM::set_comment(string commentIn)
{
  comment = std::move(commentIn);
}

void PPM::close()
{
  image.close();
  cout << "File " << fileName << " saved\n";
}
```

6.42 D:/Documents/Final/Mandelbrot_Cpp_Final/src/PPM.h File Reference

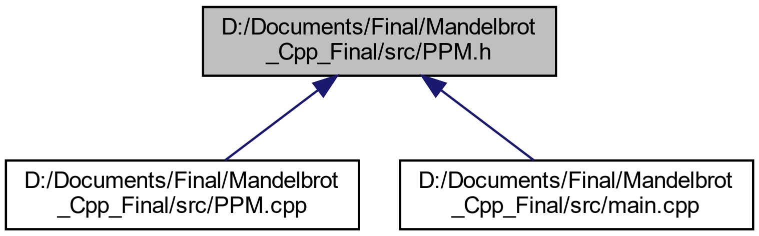
```
#include <fstream>
#include <cstring>
```

```
#include <sstream>
#include <array>
#include <vector>
#include <iostream>
```

Include dependency graph for PPM.h:



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


D:/Documents/Final/Mandelbrot
_Cpp_Final/src/PPM.h
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/PPM.cpp
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/main.cpp

Classes

- class PPM

6.43 PPM.h

Go to the documentation of this file.

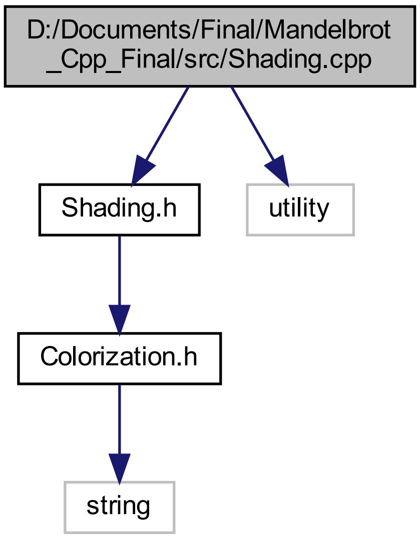
```
#ifndef C___PPM_H
#define C___PPM_H

#include <fstream>
#include <cstring>
#include <sstream>
#include <array>
#include <vector>
#include <iostream>

class PPM
{
 public:

  PPM(int width, int height);

```

```
    PPM(const std::string& fileName, int width, int height);

    PPM(const PPM& oldPPM);

    void set_outputDirectory(const std::string& outputDirectoryIn);

    PPM& operator=(std::string fileNameIn);

    bool init_stream();

    //template<size_t N>
    void write_row(const std::vector<unsigned char> &row)
    {
      image.write((char const *) row.data(), row.size());
    }

    void write_header();

    void set_width(int widthIn);

    void set_height(int heightIn);

    void set_comment(std::string commentIn);

    void close();


  private:

    const std::string magic = "P6\n";

    const std::string pixMaxVal = "255\n";

    int width;

    int subPixel;

    int height;

    std::string comment;

    std::stringstream header;

    std::string outputDirectory;

    std::string fileName;

    std::ofstream image;
};

#endif //C___PPM_H
```

6.44 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Shading.cpp File Reference

```
#include "Shading.h"
#include <utility>
```

Include dependency graph for Shading.cpp:



6.45 Shading.cpp

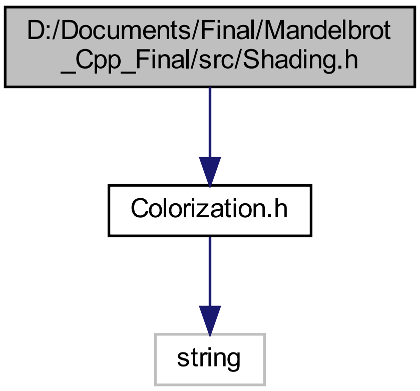
Go to the documentation of this file.

```
#include "Shading.h"

#include <utility>

Shading::Shading(std::string type) : Colorization(std::move(type))
{}

Shading::~Shading()
= default;

unsigned char Shading::get_max_color_value()
{
  return maxColorValue;
}

unsigned char Shading::get_min_color_value()
{
  return minColorValue;
}
```

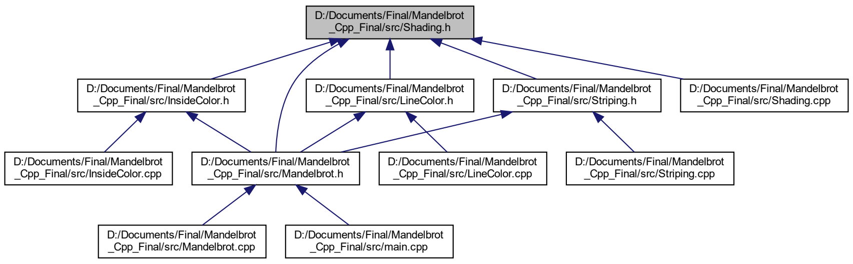
6.46 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Shading.h File Reference

`#include "Colorization.h"`

Include dependency graph for Shading.h:



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


D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Shading.h
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/InsideColor.h
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/LineColor.h
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Striping.h
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Shading.cpp
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/InsideColor.cpp
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Mandelbrot.h
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/LineColor.cpp
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Striping.cpp
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Mandelbrot.cpp
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/main.cpp

Classes

- class Shading

6.47 Shading.h

Go to the documentation of this file.

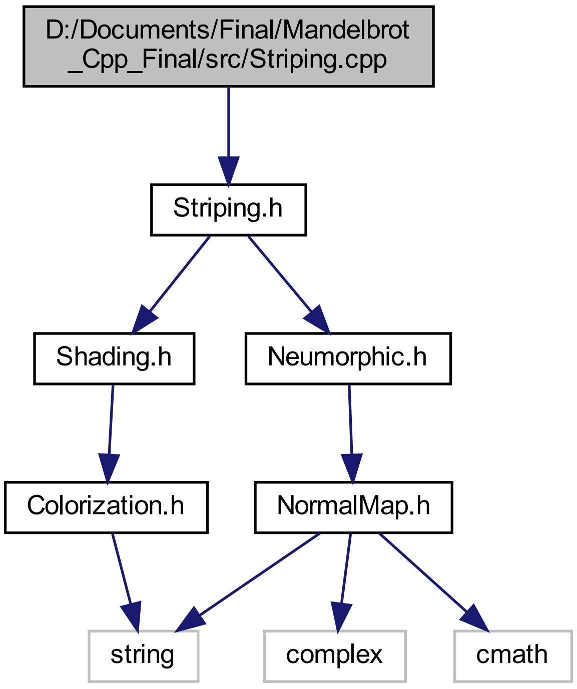
```
#ifndef C___SHADING_H
#define C___SHADING_H

#include "Colorization.h"

class Shading : public Colorization
```

```
{
 public:

  explicit Shading(std::string subtype);

  ~Shading();

  virtual unsigned char get_max_color_value();

  virtual unsigned char get_min_color_value();

  virtual unsigned char calculate_bw() = 0;

  // option for other colorizations
  virtual unsigned char calculate_r() = 0;

  virtual unsigned char calculate_g() = 0;

  virtual unsigned char calculate_b() = 0;

 private:

  std::string type = "Shading";
};

#endif  // C___SHADING_H
```

6.48 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Striping.cpp File Reference

`#include "Striping.h"`

Include dependency graph for Striping.cpp:



6.49 Striping.cpp

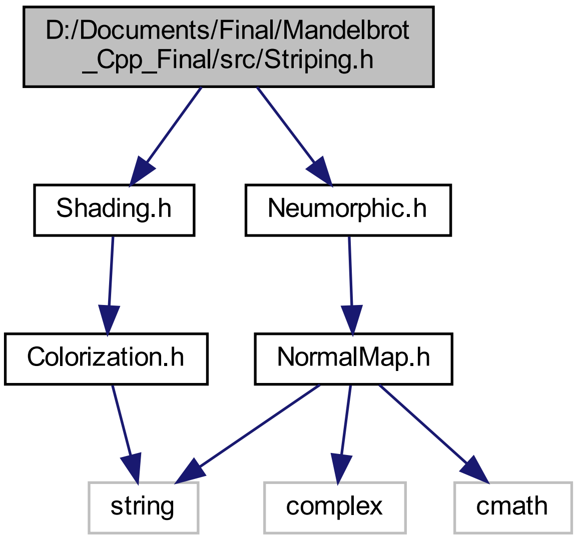
Go to the documentation of this file.

```
#include "Striping.h"

using namespace std;

Striping::Striping(double average, complex<double> z, complex<double> dc) : Shading("Striping"),
average(average), reflection(z, dc)
{}

Striping::~Striping()
= default;

unsigned char Striping::calculate_bw()
{
  return static_cast<unsigned char>((static_cast<double>((maxColorValue - 1)) - (100.0 *
  average)) * reflection.calculate()); // explicit casting
}

unsigned char Striping::calculate_r()
{
  return 0;
}

unsigned char Striping::calculate_g()
{
  return 0;
}

unsigned char Striping::calculate_b()
{
  return 0;
}

double Striping::get_average(double a)
{
  return average;
}
```

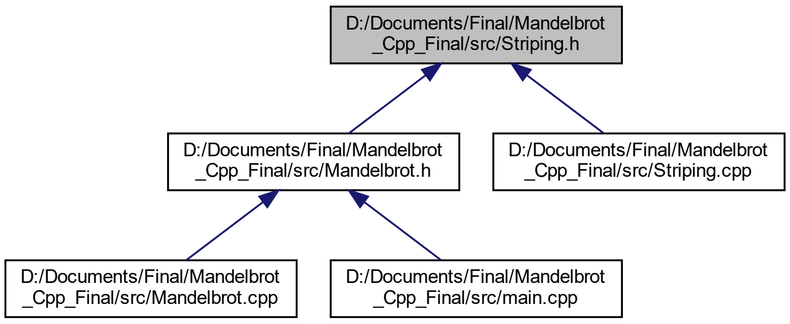
6.50 D:/Documents/Final/Mandelbrot_Cpp_Final/src/Striping.h File Reference

```
#include "Shading.h"
#include "Neumorphic.h"
```

Include dependency graph for Striping.h:



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


D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Striping.h
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Mandelbrot.h
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Striping.cpp
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/Mandelbrot.cpp
D:/Documents/Final/Mandelbrot
_Cpp_Final/src/main.cpp

Classes

- class Striping

6.51 Striping.h

Go to the documentation of this file.

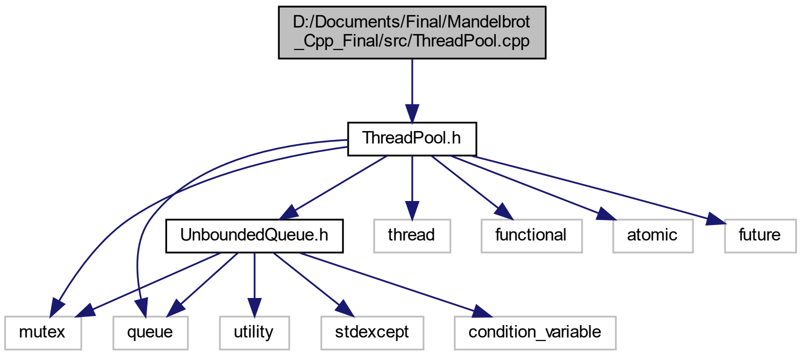
```
00001 #ifndef C___STRIPING_H
```

```
#define C___STRIPING_H

#include "Shading.h"
#include "Neumorphic.h"

class Striping : public Shading
{
 public:

  Striping(double average, std::complex<double> z, std::complex<double> dc);

  ~Striping();

  unsigned char calculate_bw();

  unsigned char calculate_r();

  unsigned char calculate_g();

  unsigned char calculate_b();

  double get_average(double average);

 private:

  double average;

  Neumorphic reflection;
};

#endif //C___STRIPING_H
```

6.52 D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPool.cpp File Reference

`#include "ThreadPool.h"`

Include dependency graph for ThreadPool.cpp:



6.53 ThreadPool.cpp

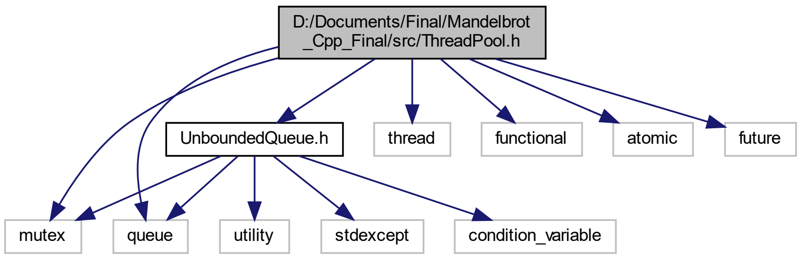
Go to the documentation of this file.

```
#include "ThreadPool.h"

ThreadPool::ThreadPool(unsigned int numThreads) : queues(numThreads), count(numThreads)
{
  if (!numThreads)
  {
    throw std::invalid_argument("thread count must be nonzero!\n");
  } else if (numThreads < 0) {
    throw std::invalid_argument("thread count must be positive! how did this happen??");
  }

  auto worker = [this] (auto i) {
    while (true)
    {
      process proc;
      for (auto j = 0; j < count * countMult; j++)
      {
        if (queues[(i + j) % count].try_pop(proc))
        {
          break;
        }
      }
      if (!proc && !queues[i].pop(proc))
      {
        break;
      }
      proc();
    }
  };

  threads.reserve(numThreads);

  for (auto i = 0; i < numThreads; i++)
  {
    threads.emplace_back(worker, i);
  }
}

ThreadPool::~ThreadPool()
{
  for (auto& queue: queues)
  {
    queue.unblock();
  }
  for (auto& thread : threads)
  {
    thread.join();
  }
}
```

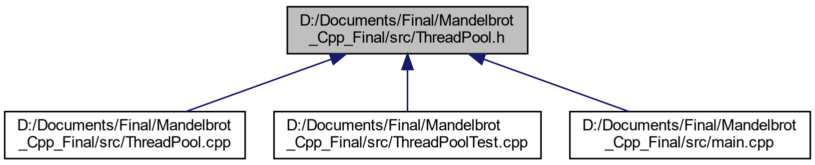
6.54 D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPool.h File Reference

```
#include "UnboundedQueue.h"
#include <thread>
#include <mutex>
#include <queue>
#include <functional>
#include <atomic>
#include <future>
```

Include dependency graph for ThreadPool.h:



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


D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPool.h
D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPool.cpp
D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPoolTest.cpp
D:/Documents/Final/Mandelbrot_Cpp_Final/src/main.cpp

Classes

- class ThreadPool

6.55 ThreadPool.h

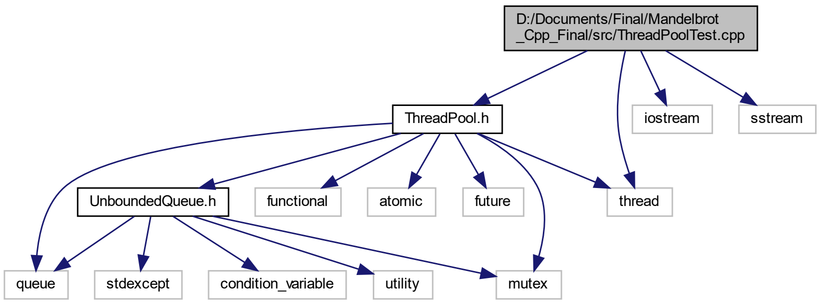
Go to the documentation of this file.

```
#ifndef C____THREADPOOL_H_
#define C____THREADPOOL_H_

#include "UnboundedQueue.h"

#include <thread>
#include <mutex>
#include <queue>
#include <functional>
#include <atomic>
#include <future>

class ThreadPool
{
 public:

  explicit ThreadPool(unsigned int numThreads);

  ~ThreadPool();

  template<typename T, typename... ARGS>
  void enqueue_work(T&& t, ARGS&&... args)
  {
    auto work = [proc = std::forward<T>(t), tuple = std::make_tuple(std::forward<ARGS>(args)...)]
        () {std::apply(proc, tuple);};

    auto i = index++;

    for (auto j = 0; j < count * countMult; j++)
    {
      if (queues[(i + j) % count].try_push(work))
      {
        return;
      }
    }

    queues[i % count].push(std::move(work));
  }

  template<typename T, typename... ARGS>
  void enqueue_task(T&& t, ARGS&&... args)
  {
    using taskReturnType = std::invoke_result<T, ARGS...>;
    using taskType = std::packaged_task<taskReturnType()>;

    auto task = std::make_shared<taskType>(std::bind(std::forward<T>(t),
 std::forward<ARGS>(args)...));
    auto work = [=] () {(*task)();};
    auto result = task->get_future();

    auto i = index++;

    for (auto j = 0; j < count * countMult; j++)
    {
      if (queues[(i + j) % count].try_push(work))
      {
        return result;
      }
    }

    queues[i % count].push(std::move(work));

    return result;
  }

 private:

  using process = std::function<void(void)>;

  using queue = UnboundedQueue<process>;

  using queueVec = std::vector<queue>;

  queueVec queues;


  using Threads = std::vector<std::thread>;

  Threads threads;


  const std::size_t count;

  std::atomic_uint index = 0;

  inline static const unsigned int countMult = 2;

};

#endif //C____THREADPOOL_H_
```

6.56 D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPoolTest.cpp File Reference

```
#include "ThreadPool.h"
#include <thread>
#include <iostream>
#include <sstream>
```

Include dependency graph for ThreadPoolTest.cpp:



Functions

- int main ()

6.56.1 Function Documentation

6.56.1.1 main()

```
int main (
              void  )
```

Definition at line 9 of file ThreadPoolTest.cpp.

```
{
  srand(0);
  auto begin = chrono::high_resolution_clock::now();

  auto numThreads = thread::hardware_concurrency();

  static std::exception_ptr globalExceptionPtr = nullptr;

  try
  {
    ThreadPool pool(numThreads);

    cout « "queueing up some random work\n";
    for (size_t i = 0; i < 1000; i++)
    {
      // pass in lambda of some random work
      pool.enqueue_work([i]()
                        {
                          try
                          {
                            size_t x;
                            size_t repetitions = 10 + (10 * (rand() % 5));
                            for (size_t j = 0; j < repetitions; j++)
                            {
                              x = i + (rand() % 200); // pretty good chance of generating a few
                              // exceptions every few runs
                            }
                            if (x > 1000)
                            {
                              stringstream id;
                              id « this_thread::get_id();
                              throw runtime_error(id.str().c_str());
                            }
                          } catch (...)
                          {
                            globalExceptionPtr = current_exception();
                          }
                        });
      if (globalExceptionPtr)
      {
        try
        {
          rethrow_exception(globalExceptionPtr);
        } catch (const exception &e)
        {
          cout « "caught exception in job from thread: " « e.what() « "\n";
        }
      }
    }
    auto end = chrono::high_resolution_clock::now();
    auto duration = chrono::duration_cast<chrono::milliseconds>(end - begin);
    cout « "pool duration = " « duration.count() / 1000.f « " seconds\n";
  } catch (const exception &ex)
  {
    cout « "caught exception from thread pool: " « ex.what() « "\n";
  }

  return 0;
}
```

Here is the call graph for this function:

main
ThreadPool::enqueue_work

6.57 ThreadPoolTest.cpp

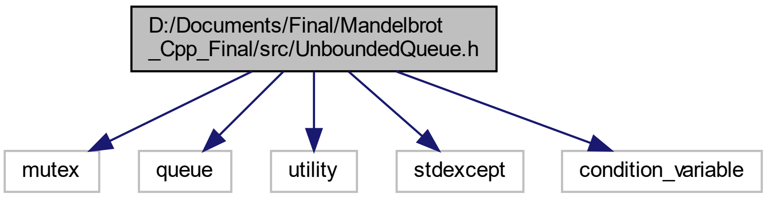
Go to the documentation of this file.

```
#include "ThreadPool.h"

#include <thread>
#include <iostream>
#include <sstream>

using namespace std;

int main()
{
  srand(0);
  auto begin = chrono::high_resolution_clock::now();

  auto numThreads = thread::hardware_concurrency();

  static std::exception_ptr globalExceptionPtr = nullptr;

  try
  {
    ThreadPool pool(numThreads);

    cout « "queueing up some random work\n";
    for (size_t i = 0; i < 1000; i++)
    {
      // pass in lambda of some random work
      pool.enqueue_work([i]()
                         {
                           try
                           {
                             size_t x;
                             size_t repetitions = 10 + (10 * (rand() % 5));
                             for (size_t j = 0; j < repetitions; j++)
                             {
                               x = i + (rand() % 200); // pretty good chance of generating a few
                               // exceptions every few runs
                             }
                             if (x > 1000)
                             {
                               stringstream id;
                               id « this_thread::get_id();
                               throw runtime_error(id.str().c_str());
                             }
                           } catch (...)
                           {
                             globalExceptionPtr = current_exception();
                           }
                         });
      if (globalExceptionPtr)
      {
        try
        {
          rethrow_exception(globalExceptionPtr);
        } catch (const exception &e)
        {
          cout « "caught exception in job from thread: " « e.what() « "\n";
        }
      }
    }
    auto end = chrono::high_resolution_clock::now();
    auto duration = chrono::duration_cast<chrono::milliseconds>(end - begin);
    cout « "pool duration = " « duration.count() / 1000.f « " seconds\n";
  } catch (const exception &ex)
  {
    cout « "caught exception from thread pool: " « ex.what() « "\n";
  }

  return 0;
}
```

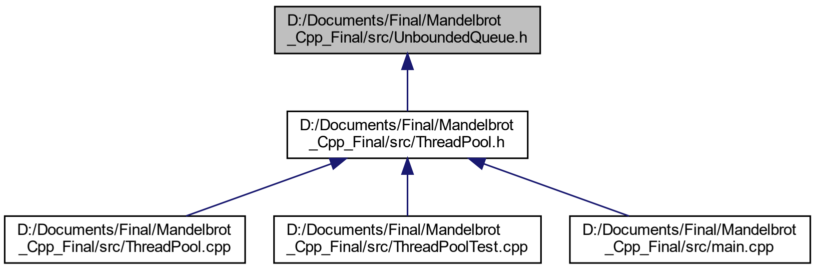
6.58 D:/Documents/Final/Mandelbrot_Cpp_Final/src/UnboundedQueue.h File Reference

```
#include <mutex>
#include <queue>
#include <utility>
#include <stdexcept>
#include <condition_variable>
```

Include dependency graph for UnboundedQueue.h:


D:/Documents/Final/Mandelbrot_Cpp_Final/src/UnboundedQueue.h
mutex
queue
utility
stdexcept
condition_variable

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


D:/Documents/Final/Mandelbrot_Cpp_Final/src/UnboundedQueue.h
D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPool.h
D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPool.cpp
D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPoolTest.cpp
D:/Documents/Final/Mandelbrot_Cpp_Final/src/main.cpp

Classes

- class UnboundedQueue< T >

6.59 UnboundedQueue.h

Go to the documentation of this file.

```
#ifndef C____UNBOUNDEDQUEUE_H_
#define C____UNBOUNDEDQUEUE_H_

#include <mutex>
#include <queue>
#include <utility>
#include <stdexcept>
#include <condition_variable>

template<typename T>
class UnboundedQueue
{
 public:

  explicit UnboundedQueue(bool block = true);

  ~UnboundedQueue();

  void push(const T& item);

  void push(T&& item);

  template<typename... ARGS>
  void emplace(ARGS&&... args);

  bool try_push(const T& item);

  bool try_push(T&& item);

  bool pop(T& item);

  bool try_pop(T& item);

  std::size_t size() const;

  bool empty() const;

  void block();

  void unblock();

  bool blocking() const;


 private:

  using queue_t = std::queue<T>;
  queue_t queue;

  bool is_block;

  mutable std::mutex queueLock;

  std::condition_variable condition;

};

template<typename T>
UnboundedQueue<T>::UnboundedQueue(bool block) : is_block(block)
{}

template<typename T>
void UnboundedQueue<T>::push(const T &item)
{
  {
    std::scoped_lock guard(queueLock);
    queue.push(item);
  }
  condition.notify_one();
}

template<typename T>
void UnboundedQueue<T>::push(T &&item)
{
  {
    std::scoped_lock guard(queueLock);
    queue.push(std::move(item));
  }
  condition.notify_one();
}

template<typename T>
```

```
template<typename... ARGS>
void UnboundedQueue<T>::emplace(ARGS &&... args)
{
  {
    std::scoped_lock guard(queueLock);
    queue.emplace(std::forward<>(args)...);
  }
  condition.notify_one();
}

template<typename T>
bool UnboundedQueue<T>::try_push(const T &item)
{
  {
    std::unique_lock guard(queueLock, std::try_to_lock);
    if (!guard)
    {
      return false;
    }
    queue.push(item);
  }
  condition.notify_one();
  return true;
}

template<typename T>
bool UnboundedQueue<T>::try_push(T &&item)
{
  {
    std::unique_lock guard(queueLock, std::try_to_lock);
    if (!guard)
    {
      return false;
    }
    queue.push(std::move(item));
  }
  condition.notify_one();
  return true;
}

template<typename T>
bool UnboundedQueue<T>::pop(T &item)
{
  std::unique_lock guard(queueLock);
  condition.wait(guard, [&] () {return !queue.empty() || !is_block;});
  if (queue.empty())
  {
    return false;
  }
  item = std::move(queue.front());
  queue.pop();
  return true;
}

template<typename T>
bool UnboundedQueue<T>::try_pop(T &item)
{
  std::unique_lock guard(queueLock, std::try_to_lock);
  if (!guard || queue.empty())
  {
    return false;
  }
  item = std::move(queue.front());
  queue.pop();
  return true;
}

template<typename T>
std::size_t UnboundedQueue<T>::size() const
{
  std::scoped_lock guard(queueLock);
  return queue.size();
}

template<typename T>
bool UnboundedQueue<T>::empty() const
{
  std::scoped_lock guard(queueLock);
  return queue.empty();
}

template<typename T>
void UnboundedQueue<T>::block()
{
  std::scoped_lock guard(queueLock);
  is_block = true;
}
```

```

template<typename T>
void UnboundedQueue<T>::unblock()
{
  {
    std::scoped_lock guard(queueLock);
    is_block = false;
  }
  condition.notify_all();
}

template<typename T>
bool UnboundedQueue<T>::blocking() const
{
  std::scoped_lock guard(queueLock);
  return is_block;
}

template<typename T>
UnboundedQueue<T>::~UnboundedQueue()
= default;



#endif //C____UNBOUNDEDQUEUE_H_
```

Index

__has_include
    CMakeCCompilerId.c, 83
    CMakeCXXCompilerId.cpp, 97
∼Colorization
    Colorization, 12
∼InsideColor
    InsideColor, 17
∼LineColor
    LineColor, 21
∼Mandelbrot
    Mandelbrot, 26
∼NormalMap
    NormalMap, 48
∼Shading
    Shading, 62
∼Striping
    Striping, 66
∼ThreadPool
    ThreadPool, 71
∼UnboundedQueue
    UnboundedQueue< T >, 76

a
    Mandelbrot, 36
angle
    Neumorphic, 45
ARCHITECTURE_ID
    CMakeCCompilerId.c, 84
    CMakeCXXCompilerId.cpp, 97
average
    Mandelbrot, 27
    Striping, 68

block
    UnboundedQueue< T >, 76
blocking
    UnboundedQueue< T >, 76
bulb
    Mandelbrot, 36

c
    Mandelbrot, 36
C_DIALECT
    CMakeCCompilerId.c, 84
calculate
    Neumorphic, 43
    NormalMap, 49
calculate_b
    InsideColor, 17
    LineColor, 21
    Shading, 62
    Striping, 66
calculate_bw
    InsideColor, 17
    LineColor, 21
    Shading, 62
    Striping, 66
calculate_g
    InsideColor, 18
    LineColor, 22
    Shading, 62
    Striping, 67
calculate_r
    InsideColor, 18
    LineColor, 22
    Shading, 63
    Striping, 67
cardioid
    Mandelbrot, 36
close
    PPM, 54
CMakeCCompilerId.c
    __has_include, 83
    ARCHITECTURE_ID, 84
    C_DIALECT, 84
    COMPILER_ID, 84
    DEC, 84
    HEX, 84
    info_arch, 86
    info_compiler, 86
    info_language_dialect_default, 86
    info_platform, 86
    main, 85
    PLATFORM_ID, 85
    STRINGIFY, 85
    STRINGIFY_HELPER, 85
CMakeCXXCompilerId.cpp
    __has_include, 97
    ARCHITECTURE_ID, 97
    COMPILER_ID, 97
    CXX_STD, 97
    DEC, 97
    HEX, 98
    info_arch, 99
    info_compiler, 99
    info_language_dialect_default, 100
    info_platform, 100
    main, 99
    PLATFORM_ID, 98

STRINGIFY, 98
STRINGIFY_HELPER, 98
Colorization, 11
~Colorization, 12
Colorization, 13
get_max_color_value, 13
get_min_color_value, 13
get_type, 13
maxColorValue, 14
minColorValue, 14
type, 14
colorize_bw
Mandelbrot, 27
comment
PPM, 58
COMPILER_ID
CMakeCCompilerId.c, 84
CMakeCXXCompilerId.cpp, 97
condition
UnboundedQueue< T >, 80
count
ThreadPool, 73
countMult
ThreadPool, 73
current_pixel
Mandelbrot, 28
cxMax
Mandelbrot, 36
cxMin
Mandelbrot, 36
CXX_STD
CMakeCXXCompilerId.cpp, 97
cyMax
Mandelbrot, 37
cyMin
Mandelbrot, 37

d
Mandelbrot, 37
D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/CMakeFiles/3.21.1/CompilerIdC/CMakeCCompilerId.c, 83, 87
D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/CMakeFiles/3.21.1/CompilerIdCXX/CMakeCXXCompilerId.cpp, 96, 100
D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/CMakeFiles/FindOpenMP/OpenMPCheckVersion.c, 109, 111
D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/CMakeFiles/FindOpenMP/OpenMPCheckVersion.cpp, 111, 112
D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/CMakeFiles/FindOpenMP/OpenMPTryFlag.c, 113
D:/Documents/Final/Mandelbrot_Cpp_Final/cmake-build-debug/CMakeFiles/FindOpenMP/OpenMPTryFlag.cpp, 114
D:/Documents/Final/Mandelbrot_Cpp_Final/README.md, 115
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Colorization.cpp, 115
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Colorization.h, 116
D:/Documents/Final/Mandelbrot_Cpp_Final/src/InsideColor.cpp, 117
D:/Documents/Final/Mandelbrot_Cpp_Final/src/InsideColor.h, 118, 119
D:/Documents/Final/Mandelbrot_Cpp_Final/src/LineColor.cpp, 120
D:/Documents/Final/Mandelbrot_Cpp_Final/src/LineColor.h, 121, 122
D:/Documents/Final/Mandelbrot_Cpp_Final/src/main.cpp, 122, 124
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Mandelbrot.cpp, 126, 127
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Mandelbrot.h, 130, 131
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Neumorphic.cpp, 133
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Neumorphic.h, 134, 135
D:/Documents/Final/Mandelbrot_Cpp_Final/src/NormalMap.cpp, 136
D:/Documents/Final/Mandelbrot_Cpp_Final/src/NormalMap.h, 136, 137
D:/Documents/Final/Mandelbrot_Cpp_Final/src/PPM.cpp, 138
D:/Documents/Final/Mandelbrot_Cpp_Final/src/PPM.h, 139, 140
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Shading.cpp, 141, 142
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Shading.h, 143
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Striping.cpp, 144, 145
D:/Documents/Final/Mandelbrot_Cpp_Final/src/Striping.h, 145, 146
D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPool.cpp, 147
D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPool.h, 148, 149
D:/Documents/Final/Mandelbrot_Cpp_Final/src/ThreadPoolTest.cpp, 150, 152
D:/Documents/Final/Mandelbrot_Cpp_Final/src/UnboundedQueue.h, 153, 154
dC
Mandelbrot, 37
Neumorphic, 45
de
Mandelbrot, 37
DEC
CMakeCCompilerId.c, 84
CMakeCXXCompilerId.cpp, 97
describe_border
Mandelbrot, 29
dot_product
NormalMap, 49

emplace
UnboundedQueue< T >, 77
empty
UnboundedQueue< T >, 77
enqueue_task
ThreadPool, 72
enqueue_work
ThreadPool, 72
escapeRadius
Mandelbrot, 37

fileName
PPM, 58

get_angle
Neumorphic, 44
get_average
Striping, 67
get_c
Mandelbrot, 29
get_heightFactor
Neumorphic, 44
get_max_color_value
Colorization, 13
Shading, 63
get_max_val
NormalMap, 50
get_min_color_value
Colorization, 13
Shading, 63
get_min_val
NormalMap, 50
get_reflection
Neumorphic, 44
get_t
Mandelbrot, 30
get_type
Colorization, 13
NormalMap, 50

header
PPM, 58
height
Mandelbrot, 38
PPM, 59
heightFactor
Neumorphic, 45
HEX
CMakeCCompilerId.c, 84
CMakeCXXCompilerId.cpp, 98

image
PPM, 59
in_border
Mandelbrot, 30
in_set
Mandelbrot, 31
index
ThreadPool, 73
info_arch
CMakeCCompilerId.c, 86
CMakeCXXCompilerId.cpp, 99
info_compiler
CMakeCCompilerId.c, 86
CMakeCXXCompilerId.cpp, 99
info_language_dialect_default
CMakeCCompilerId.c, 86
CMakeCXXCompilerId.cpp, 100
info_platform
CMakeCCompilerId.c, 86
CMakeCXXCompilerId.cpp, 100
init_stream
PPM, 55
InsideColor, 15
∼InsideColor, 17
calculate_b, 17
calculate_bw, 17
calculate_g, 18
calculate_r, 18
InsideColor, 17
interpolate
Mandelbrot, 31
is_block
UnboundedQueue< T >, 80
iSkip
Mandelbrot, 38
iter
Mandelbrot, 38
iterate
Mandelbrot, 32
iterMax
Mandelbrot, 38

LineColor, 19
∼LineColor, 21
calculate_b, 21
calculate_bw, 21
calculate_g, 22
calculate_r, 22
LineColor, 21

M_PI
Mandelbrot.h, 131
magic
PPM, 59
main
CMakeCCompilerId.c, 85
CMakeCXXCompilerId.cpp, 99
main.cpp, 123
OpenMPCheckVersion.c, 110
OpenMPCheckVersion.cpp, 112
OpenMPTryFlag.c, 113
OpenMPTryFlag.cpp, 114
ThreadPoolTest.cpp, 151
main.cpp
main, 123
Mandelbrot, 23
∼Mandelbrot, 26

a, 36
average, 27
bulb, 36
c, 36
cardioid, 36
colorize_bw, 27
current_pixel, 28
cxMax, 36
cxMin, 36
cyMax, 37
cyMin, 37
d, 37
dC, 37
de, 37
describe_border, 29
escapeRadius, 37
get_c, 29
get_t, 30
height, 38
in_border, 30
in_set, 31
interpolate, 31
iSkip, 38
iter, 38
iterate, 32
iterMax, 38
Mandelbrot, 25, 26
operator<<, 35
pixHeight, 38
pixWidth, 38
prevA, 39
pX, 39
pY, 39
q, 39
r, 39
reset, 33
set_border, 33
set_image, 34
set_iSkip, 34
set_plane, 34
set_stripe_density, 34
shade, 39
shape_check, 35
stripeDensity, 40
thin, 40
width, 40
z, 40
Mandelbrot.cpp
operator<<, 126
Mandelbrot.h
M_PI, 131
maxColorValue
Colorization, 14
maxMapVal
NormalMap, 50
minColorValue
Colorization, 14
minMapVal
NormalMap, 50

Neumorphic, 41
angle, 45
calculate, 43
dC, 45
get_angle, 44
get_heightFactor, 44
get_reflection, 44
heightFactor, 45
Neumorphic, 43
reflection, 45
u, 45
v, 46
z, 46
NormalMap, 46
∼NormalMap, 48
calculate, 49
dot_product, 49
get_max_val, 50
get_min_val, 50
get_type, 50
maxMapVal, 50
minMapVal, 50
NormalMap, 49
type, 51

ompver_str
OpenMPCheckVersion.c, 110
OpenMPCheckVersion.cpp, 112
OpenMPCheckVersion.c
main, 110
ompver_str, 110
OpenMPCheckVersion.cpp
main, 112
ompver_str, 112
OpenMPTryFlag.c
main, 113
OpenMPTryFlag.cpp
main, 114
operator<<
Mandelbrot, 35
Mandelbrot.cpp, 126
operator=
PPM, 55
outputDirectory
PPM, 59

pixHeight
Mandelbrot, 38
pixMaxVal
PPM, 59
pixWidth
Mandelbrot, 38
PLATFORM_ID
CMakeCCompilerId.c, 85
CMakeCXXCompilerId.cpp, 98
pop
UnboundedQueue< T >, 77

PPM, 51
- close, 54
- comment, 58
- fileName, 58
- header, 58
- height, 59
- image, 59
- init_stream, 55
- magic, 59
- operator=, 55
- outputDirectory, 59
- pixMaxVal, 59
- PPM, 53, 54
- set_comment, 56
- set_height, 56
- set_outputDirectory, 56
- set_width, 57
- subPixel, 59
- width, 60
- write_header, 57
- write_row, 57

prevA
- Mandelbrot, 39

process
- ThreadPool, 70

push
- UnboundedQueue< T >, 78

pX
- Mandelbrot, 39

pY
- Mandelbrot, 39

q
- Mandelbrot, 39

queue
- ThreadPool, 70
- UnboundedQueue< T >, 81

queue_t
- UnboundedQueue< T >, 75

queueLock
- UnboundedQueue< T >, 81

queues
- ThreadPool, 73

queueVec
- ThreadPool, 70

r
- Mandelbrot, 39

reflection
- Neumorphic, 45
- Striping, 68

reset
- Mandelbrot, 33

set_border
- Mandelbrot, 33

set_comment
- PPM, 56

set_height
- PPM, 56

set_image
- Mandelbrot, 34

set_iSkip
- Mandelbrot, 34

set_outputDirectory
- PPM, 56

set_plane
- Mandelbrot, 34

set_stripe_density
- Mandelbrot, 34

set_width
- PPM, 57

shade
- Mandelbrot, 39

Shading, 60
- ∼Shading, 62
- calculate_b, 62
- calculate_bw, 62
- calculate_g, 62
- calculate_r, 63
- get_max_color_value, 63
- get_min_color_value, 63
- Shading, 62
- type, 63

shape_check
- Mandelbrot, 35

size
- UnboundedQueue< T >, 79

STRINGIFY
- CMakeCCompilerId.c, 85
- CMakeCXXCompilerId.cpp, 98

STRINGIFY_HELPER
- CMakeCCompilerId.c, 85
- CMakeCXXCompilerId.cpp, 98

stripeDensity
- Mandelbrot, 40

Striping, 64
- ∼Striping, 66
- average, 68
- calculate_b, 66
- calculate_bw, 66
- calculate_g, 67
- calculate_r, 67
- get_average, 67
- reflection, 68
- Striping, 66

subPixel
- PPM, 59

thin
- Mandelbrot, 40

ThreadPool, 68
- ∼ThreadPool, 71
- count, 73
- countMult, 73
- enqueue_task, 72
- enqueue_work, 72
- index, 73

process, 70
queue, 70
queues, 73
queueVec, 70
ThreadPool, 70
Threads, 70
threads, 73
ThreadPoolTest.cpp
main, 151
Threads
ThreadPool, 70
threads
ThreadPool, 73
try_pop
UnboundedQueue< T >, 79
try_push
UnboundedQueue< T >, 79
type
Colorization, 14
NormalMap, 51
Shading, 63

u
Neumorphic, 45
unblock
UnboundedQueue< T >, 80
UnboundedQueue
UnboundedQueue< T >, 76
UnboundedQueue< T >, 74
∼UnboundedQueue, 76
block, 76
blocking, 76
condition, 80
emplace, 77
empty, 77
is_block, 80
pop, 77
push, 78
queue, 81
queue_t, 75
queueLock, 81
size, 79
try_pop, 79
try_push, 79
unblock, 80
UnboundedQueue, 76

v
Neumorphic, 46

width
Mandelbrot, 40
PPM, 60
write_header
PPM, 57
write_row
PPM, 57

z
Mandelbrot, 40
Neumorphic, 46