

Raytracing Playground

0.01

Generated by Doxygen 1.9.8

1 Atividade01	1
2 Learning the basics Raytracing	3
2.1 How it is organized	3
2.2 Dependencies	3
2.3 Atividades	3
2.3.1 Atividade01	3
3 Class Index	5
3.1 Class List	5
4 File Index	7
4.1 File List	7
5 Class Documentation	9
5.1 RGBv Struct Reference	9
5.1.1 Member Data Documentation	9
5.1.1.1 blue	9
5.1.1.2 green	9
5.1.1.3 red	9
6 File Documentation	11
6.1 Atividade01/image_utils.cpp File Reference	11
6.1.1 Function Documentation	11
6.1.1.1 load_image()	11
6.1.1.2 save_image()	12
6.2 Atividade01/image_utils.h File Reference	13
6.2.1 Function Documentation	13
6.2.1.1 load_image()	13
6.2.1.2 save_image()	13
6.3 image_utils.h	14
6.4 Atividade01/main.cpp File Reference	14
6.4.1 Function Documentation	14
6.4.1.1 main()	14
6.5 Atividade01/README.md File Reference	14
6.6 README.md File Reference	14
Index	15

Chapter 1

Atividade01

For this assignment, I wrote a function to manage saving images on the disk. I chose primarily PNG encoding when dealing with images, thus using the `libpng` library. To test the save function, I represented some images in a 2D array of 256x256 and used the function `save_image()` to save them into .png files.

A load function is yet to be written for use on future assignments.

Chapter 2

Learning the basics Raytracing

This repository is an attempt to learn the concepts and fundamentals of raytracing. The code presented here follows the assignments from the class 1001315 – COMPUTAÇÃO GRÁFICA, lectured by the professor **Mario A. S. Lizier** on the **Universidade Federal de Sao Carlos - Campus Sorocaba**. The class structure bases itself on the series of books **Raytracing in One Weekend**, written by Peter Shirley, Trevor David Black, and Steve Hollasch.

2.1 How it is organized

I divided this repository into sections where each section corresponds to one assignment. These divisions follow the labeling pattern `AtividadeXX` in which the **XX** corresponds to the assignment number.

2.2 Dependencies

- `libpng`

2.3 Atividades

In a short text format, I describe the work done and project decisions made. For each assignment, there is a corresponding subsection with such descriptions.

2.3.1 Atividade01

For this assignment, I wrote a function to manage saving images on the disk. I chose primarily PNG encoding when dealing with images, thus using the `libpng` library. To test the save function, I represented some images in a 2D array of 256x256 and used the function `save_image()` to save them into .png files.

A load function is yet to be written for use on future assignments.

Chapter 3

Class Index

3.1 Class List

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

RGBv	9
--------------------------------	---

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Atividade01/ image_utils.cpp	11
Atividade01/ image_utils.h	13
Atividade01/ main.cpp	14

Chapter 5

Class Documentation

5.1 RGBv Struct Reference

```
#include <image_utils.h>
```

Public Attributes

- int [red](#)
- int [green](#)
- int [blue](#)

5.1.1 Member Data Documentation

5.1.1.1 blue

```
int RGBv::blue
```

5.1.1.2 green

```
int RGBv::green
```

5.1.1.3 red

```
int RGBv::red
```

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

- Atividade01/[image_utils.cpp](#)
- Atividade01/[image_utils.h](#)

Chapter 6

File Documentation

6.1 Atividade01/image_utils.cpp File Reference

```
#include <iostream>
#include <math.h>
#include <malloc.h>
#include <png.h>
```

Classes

- struct [RGBv](#)

Functions

- int [save_image](#) (char *path, [RGBv](#) **m, int height, int width)
Saves a 2D array in a image file.
- int [load_image](#) (char *path, [RGBv](#) **m, int height, int width)

6.1.1 Function Documentation

6.1.1.1 load_image()

```
int load_image (
    char * path,
    RGBv ** m,
    int height,
    int width )
```

6.1.1.2 save_image()

```
int save_image (
    char * path,
    RGBv ** m,
    int height,
    int width )
```

Saves a 2D array in a image file.

This function takes a 2D array that represents an image, where each element holds the three color values for an RGB image, and saves them into a PNG file using the library libpng. The height and width of the image needs to be passed to generate the .png file, as well as a path where the file will be store. It is important that the path string contains the image name, not just the folder where you want to save the image.

Parameters

<i>path</i>	File path on which the image will be stored. Includes the name of the image. Ex: <code>"/images/image01.png"</code>
<i>m</i>	2d array of pixels that represents the image to be saved. The type of the array needs to be <code>RGBv</code> , where each element holds the three color values of a pixel.
<i>height</i>	Height in pixels of the image
<i>width</i>	Width in pixels of the image

Returns

The function returns a flag for eventual errors. Return code 1 means that an error occurred during the process. Return code 0 indicates that everything was done successfully.

6.2 Atividade01/image_utils.h File Reference

```
#include <iostream>
```

Classes

- struct `RGBv`

Functions

- int `save_image` (char *path, `RGBv` **m, int height, int width)
Saves a 2D array in a image file.
- int `load_image` (char *path, `RGBv` **m, int height, int width)

6.2.1 Function Documentation

6.2.1.1 load_image()

```
int load_image (
    char * path,
    RGBv ** m,
    int height,
    int width )
```

6.2.1.2 save_image()

```
int save_image (
    char * path,
    RGBv ** m,
    int height,
    int width )
```

Saves a 2D array in a image file.

This function takes a 2D array that represents an image, where each element holds the three color values for an RGB image, and saves them into a PNG file using the library libpng. The height and width of the image needs to be passed to generate the .png file, as well as a path where the file will be store. It is important that the path string contains the image name, not just the folder where you want to save the image.

Parameters

<i>path</i>	File path on which the image will be stored. Includes the name of the image. Ex: "./images/image01.png"
<i>m</i>	2d array of pixels that represents the image to be saved. The type of the array needs to be RGBv , where each element holds the three color values of a pixel.
<i>height</i>	Height in pixels of the image
<i>width</i>	Width in pixels of the image

Returns

The function returns a flag for eventual errors. Return code 1 means that an error occurred during the process. Return code 0 indicates that everything was done successfully.

6.3 image_utils.h

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

```

00001 #ifndef image_utils
00002 #define image_utils
00003
00004 #include <iostream>
00005
00006 using namespace std;
00007
00008 typedef struct {
00009     int red;
00010     int green;
00011     int blue;
00012 } RGBv;
00013
00014 int save_image(char *path, RGBv **m, int height, int width);
00015
00016 int load_image(char *path, RGBv **m, int height, int width);
00017
00018 #endif

```

6.4 Atividade01/main.cpp File Reference

```

#include <iostream>
#include "image_utils.h"

```

Functions

- int [main](#) ()

6.4.1 Function Documentation

6.4.1.1 main()

```
int main ( )
```

6.5 Atividade01/README.md File Reference

6.6 README.md File Reference

Index

Atividade01, [1](#)
Atividade01/image_utils.cpp, [11](#)
Atividade01/image_utils.h, [13](#), [14](#)
Atividade01/main.cpp, [14](#)
Atividade01/README.md, [14](#)

blue
 RGBv, [9](#)

green
 RGBv, [9](#)

image_utils.cpp
 load_image, [11](#)
 save_image, [11](#)
image_utils.h
 load_image, [13](#)
 save_image, [13](#)

Learning the basics Raytracing, [3](#)
load_image
 image_utils.cpp, [11](#)
 image_utils.h, [13](#)

main
 main.cpp, [14](#)
main.cpp
 main, [14](#)

README.md, [14](#)
red

 RGBv, [9](#)
RGBv, [9](#)
 blue, [9](#)
 green, [9](#)
 red, [9](#)

save_image
 image_utils.cpp, [11](#)
 image_utils.h, [13](#)