

routeplannerx

Generated by Doxygen 1.8.7

Mon May 19 2014 13:00:52

Contents

1	Data Structure Index	1
1.1	Data Structures	1
2	File Index	3
2.1	File List	3
3	Data Structure Documentation	5
3.1	coord Struct Reference	5
3.1.1	Detailed Description	5
3.1.2	Field Documentation	5
3.1.2.1	x	5
3.1.2.2	y	5
3.2	Node Struct Reference	5
3.2.1	Detailed Description	6
3.2.2	Field Documentation	6
3.2.2.1	checkpoint	6
3.2.2.2	coords	6
3.2.2.3	east	6
3.2.2.4	mark	6
3.2.2.5	name	6
3.2.2.6	north	6
3.2.2.7	past	7
3.2.2.8	south	7
3.2.2.9	west	7
4	File Documentation	9
4.1	src/data.h File Reference	9
4.1.1	Detailed Description	9
4.1.2	Typedef Documentation	9
4.1.2.1	node	9
	Index	10

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

coord	A coordinate struct. A structure to represent the coordinates of a point	5
Node	5

Chapter 2

File Index

2.1 File List

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

src/ data.h	Header file to describe necessary data structures	9
src/ functions.c	??
src/ functions.h	??
src/ main.c	??
src/ main.h	??

Chapter 3

Data Structure Documentation

3.1 coord Struct Reference

A coordinate struct. A structure to represent the coordinates of a point.

```
#include <data.h>
```

Data Fields

- int [x](#)
- int [y](#)

3.1.1 Detailed Description

A coordinate struct. A structure to represent the coordinates of a point.

Definition at line 11 of file data.h.

3.1.2 Field Documentation

3.1.2.1 int coord::x

The x-coordinate of the point.

Definition at line 12 of file data.h.

3.1.2.2 int coord::y

The y-coordinate of the point.

Definition at line 13 of file data.h.

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

- src/[data.h](#)

3.2 Node Struct Reference

```
#include <data.h>
```

Data Fields

- char `name` [3]
- coord `coords`
- int `checkpoint`
- int `mark`
- int `past`
- struct `Node` * `north`
- struct `Node` * `south`
- struct `Node` * `east`
- struct `Node` * `west`

3.2.1 Detailed Description

Structure representing a node.

Definition at line 17 of file `data.h`.

3.2.2 Field Documentation

3.2.2.1 int `Node::checkpoint`

The number of the checkpoint the node is connected to. Is -1 when not connected.

Definition at line 20 of file `data.h`.

3.2.2.2 coord `Node::coords`

A coord to represent the coordinates of the node.

Definition at line 19 of file `data.h`.

3.2.2.3 struct `Node*` `Node::east`

Pointer to the eastern neighbour node.

Definition at line 25 of file `data.h`.

3.2.2.4 int `Node::mark`

Mark used by Lee algorithm.

Definition at line 21 of file `data.h`.

3.2.2.5 char `Node::name`[3]

The name of the node in the form "<x><y>", e.g. "03".

Definition at line 18 of file `data.h`.

3.2.2.6 struct `Node*` `Node::north`

Pointer to the northern neighbour node.

Definition at line 23 of file `data.h`.

3.2.2.7 int Node::past

Used by Lee algorithm: 0 means not yet visited, 1 means already visited.

Definition at line 22 of file data.h.

3.2.2.8 struct Node* Node::south

Pointer to the southern neighbour node.

Definition at line 24 of file data.h.

3.2.2.9 struct Node* Node::west

Pointer to the western neighbour node.

Definition at line 26 of file data.h.

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

- [src/data.h](#)

Chapter 4

File Documentation

4.1 src/data.h File Reference

Header file to describe necessary data structures.

Data Structures

- struct [coord](#)
A coordinate struct. A structure to represent the coordinates of a point.
- struct [Node](#)

Typedefs

- typedef struct [Node](#) [node](#)

4.1.1 Detailed Description

Header file to describe necessary data structures.

Definition in file [data.h](#).

4.1.2 Typedef Documentation

4.1.2.1 typedef struct **Node** **node**

Structure representing a node.

Index

- checkpoint
 - Node, [6](#)
- coord, [5](#)
 - x, [5](#)
 - y, [5](#)
- coords
 - Node, [6](#)
- east
 - Node, [6](#)
- mark
 - Node, [6](#)
- name
 - Node, [6](#)
- Node, [5](#)
 - checkpoint, [6](#)
 - coords, [6](#)
 - east, [6](#)
 - mark, [6](#)
 - name, [6](#)
 - north, [6](#)
 - past, [6](#)
 - south, [7](#)
 - west, [7](#)
- north
 - Node, [6](#)
- past
 - Node, [6](#)
- south
 - Node, [7](#)
- west
 - Node, [7](#)
- x
 - coord, [5](#)
- y
 - coord, [5](#)