labo_10_comte_emmanuelle_gallay_david

Generated by Doxygen 1.8.13

Contents

Chapter 1

File Index

1.1 File List

Here is a list of all files with brief descriptions:

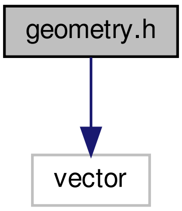
Chapter 2

File Documentation

2.1 geometry.h File Reference

```
#include <vector>
```

Include dependency graph for geometry.h:



Typedefs

- typedef std::size_t Coordinate
- typedef std::vector< Coordinate > Point
- typedef std::vector< Point > PointList
- typedef std::vector< PointState > Line
- typedef std::vector< Line > Map
- typedef std::vector< Map > Space

Enumerations

- enum PointState { EMPTY, EXIST }
- enum MapCode { MAPCODE_XY, MAPCODE_XZ, MAPCODE_YZ }

Functions

- PointList getPointList ()

  Get a point list from the user.
- void displayLine (const Line &line)

  Display a line.
- void displayMap (const Map &map)

  Display a map.
- Space getSpace (PointList list)

  Create a space conatin the three projections.
- void project (PointList list, Map &map, MapCode code)

  Project a point in a map (XY, XZ or YZ)
- void addPoint (Point point, Space &space)

  Add a point to all of projections in space.
- Map getProjection (const Space &space, MapCode code)
- Coordinate getX (Point point)

  Get the coordinate X from a point.
- Coordinate getY (Point point)

  Get the coordinate Y from a point.
- Coordinate getZ (Point point)

  Get the coordinate Z from a point.

2.1.1 Typedef Documentation

2.1.1.1 Coordinate

```
typedef std::size_t Coordinate
```

2.1.1.2 Line

```
typedef std::vector<PointState> Line
```

2.1.1.3 Map

```
typedef std::vector<Line> Map
```

2.1.1.4 Point

```
typedef std::vector<Coordinate> Point
```

2.1.1.5 PointList

```
typedef std::vector<Point> PointList
```

2.1.1.6 Space

```
typedef std::vector<Map> Space
```

2.1.2 Enumeration Type Documentation

2.1.2.1 MapCode

```
enum MapCode
```

Enumerator

MAPCODE_XY	
MAPCODE_XZ	
MAPCODE_YZ	

2.1.2.2 PointState

```
enum PointState
```

Enumerator

EMPTY	
EXIST	

2.1.3 Function Documentation

2.1.3.1 addPoint()

```
void addPoint (
            Point point,
            Space & space )
```

Add a point to all of projections in space.

Parameters

in	*PointList*	list
in		

2.1.3.2 displayLine()

```
void displayLine (
             const Line & line )
```

Display a line.

Parameters

in	*const*	Line& line

2.1.3.3 displayMap()

```
void displayMap (
             const Map & map )
```

Display a map.

Parameters

in	*const*	Map& map

2.1.3.4 getPointList()

```
PointList getPointList ( )
```

Get a point list from the user.

Returns

PointList The point list the user enter

2.1.3.5 getProjection()

```
Map getProjection (
             const Space & space,
             MapCode code )
```

Parameters

in	*const*	Space& space
in	*MapCode*	code The code for the direction of the projection

Returns

Map The map of the projection asked

2.1.3.6 getSpace()

```
Space getSpace (
            PointList list )
```

Create a space conatin the three projections.

Parameters

in	*PointList*	list The list of the points in the space

Returns

Space Contain the three projections

2.1.3.7 getX()

```
Coordinate getX (
            Point point )
```

Get the coordinate X from a point.

Parameters

in	*Point*	point

Returns

Coodrinate The coordinate X of the point

2.1.3.8 getY()

```
Coordinate getY (
            Point point )
```

Get the coordinate Y from a point.

Parameters

in	*Point*	point

Returns

Coodrinate The coordinate Y of the point

2.1.3.9 getZ()

```
Coordinate getZ (
            Point point )
```

Get the coordinate Z from a point.

Parameters

in	*Point*	point

Returns

Coodrinate The coordinate Z of the point

2.1.3.10 project()

```
void project (
            PointList list,
            Map & map,
            MapCode code )
```

Project a point in a map (XY, XZ or YZ)

Parameters

in	*PointList*	list
in		

Index