

HPAir Flight Map

Generated by Doxygen 1.8.13

Contents

1	Class Index	1
1.1	Class List	1
2	File Index	3
2.1	File List	3
3	Class Documentation	5
3.1	City Struct Reference	5
3.2	Flight Struct Reference	5
3.3	FlightMap Class Reference	5
3.3.1	Member Function Documentation	6
3.3.1.1	addCity()	6
3.3.1.2	getCity()	6
3.3.1.3	getPath()	6
4	File Documentation	9
4.1	PA03/FlightMapV1.cpp File Reference	9
4.1.1	Detailed Description	9
4.1.2	Function Documentation	9
4.1.2.1	log()	9
4.2	PA03/FlightMapV1.h File Reference	10
4.2.1	Detailed Description	10
4.2.2	Function Documentation	10
4.2.2.1	log()	10
4.3	PA03/PA03.cpp File Reference	11

4.3.1	Detailed Description	11
4.3.2	Function Documentation	11
4.3.2.1	handleRequests()	11
4.3.2.2	loadCities()	12
4.3.2.3	loadFlights()	12
4.3.2.4	printFlightPlan()	13
4.4	PA03/PA03.h File Reference	13
4.4.1	Detailed Description	14
4.4.2	Function Documentation	14
4.4.2.1	handleRequests()	14
4.4.2.2	loadCities()	14
4.4.2.3	loadFlights()	15
4.4.2.4	printFlightPlan()	15
Index		17

Chapter 1

Class Index

1.1 Class List

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

City	5
Flight	5
FlightMap	5

Chapter 2

File Index

2.1 File List

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

PA03/ FlightMapV1.cpp	
FlightMap class implementation	9
PA03/ FlightMapV1.h	
Header for FlightMap implementation	10
PA03/ PA03.cpp	
Program driver implementation file	11
PA03/ PA03.h	
Program driver header file	13

Chapter 3

Class Documentation

3.1 City Struct Reference

Public Attributes

- `std::string` **name**
- `std::vector< Flight >` **flights**
- `bool` **visited**

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

- [PA03/FlightMapV1.h](#)

3.2 Flight Struct Reference

Public Attributes

- `City *` **from**
- `City *` **to**
- `unsigned int` **id**
- `unsigned int` **cost**

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

- [PA03/FlightMapV1.h](#)

3.3 FlightMap Class Reference

Public Member Functions

- `City *` [addCity](#) (`std::string &name`)
Adds a new city to the map.
- `City *` [getCity](#) (`std::string &name`)
Gets a city from the map with a specific name.
- `std::stack< Flight * >` [getPath](#) (`City *`from, `City *`to)
Finds path of flights to get from one city to another.

3.3.1 Member Function Documentation

3.3.1.1 addCity()

```
City * FlightMap::addCity (
    std::string & name )
```

Adds a new city to the map.

Parameters

in	<i>name</i>	Name of the new city
----	-------------	----------------------

Returns

Pointer to the new city

3.3.1.2 getCity()

```
City * FlightMap::getCity (
    std::string & name )
```

Gets a city from the map with a specific name.

Parameters

in	<i>name</i>	The name of the city to search for
----	-------------	------------------------------------

Returns

Pointer to the found city, or nullptr if not found

3.3.1.3 getPath()

```
std::stack< Flight * > FlightMap::getPath (
    City * from,
    City * to )
```

Finds path of flights to get from one city to another.

Parameters

in	<i>from</i>	Pointer to the city to start searching from
in	<i>to</i>	Pointer to the city to search for

Algorithm

Starting with *from*, searches through each adjacent unvisited city and then repeats until it either finds *to* or runs out of adjacent cities

Returns

A stack of flights from the origin city to the destination city Or an empty stack if no path could be found

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

- [PA03/FlightMapV1.h](#)
- [PA03/FlightMapV1.cpp](#)

Chapter 4

File Documentation

4.1 PA03/FlightMapV1.cpp File Reference

[FlightMap](#) class implementation.

```
#include "FlightMapV1.h"
```

Functions

- void [log](#) (const std::string &text, bool newLine)
Logs information to file defined in LOG_FILE.

4.1.1 Detailed Description

[FlightMap](#) class implementation.

Version

1.00 Alexander Novotny First version

4.1.2 Function Documentation

4.1.2.1 log()

```
void log (
    const std::string & text,
    bool newLine )
```

Logs information to file defined in LOG_FILE.

Parameters

in	<i>text</i>	Text to log
in	<i>newLine</i>	Whether std::endl should be printed at the end. Defaults to true

Note

If LOG_FILE not defined, nothing will happen

4.2 PA03/FlightMapV1.h File Reference

Header for [FlightMap](#) implementation.

```
#include <string>
#include <vector>
#include <stack>
#include <fstream>
```

Classes

- struct [City](#)
- struct [Flight](#)
- class [FlightMap](#)

Macros

- `#define LOG_FILE "log.txt"`

Functions

- void [log](#) (const std::string &, bool=true)
Logs information to file defined in LOG_FILE.

4.2.1 Detailed Description

Header for [FlightMap](#) implementation.

[FlightMap](#) class declaration, as well as [City](#) and [Flight](#) structs and log function used by [FlightMap](#)

Version

1.00 Alexander Novotny First version

4.2.2 Function Documentation

4.2.2.1 log()

```
void log (
    const std::string & text,
    bool newLine )
```

Logs information to file defined in LOG_FILE.

Parameters

in	<i>text</i>	Text to log
in	<i>newLine</i>	Whether std::endl should be printed at the end. Defaults to true

Note

If LOG_FILE not defined, nothing will happen

4.3 PA03/PA03.cpp File Reference

Program driver implementation file.

```
#include "PA03.h"
```

Functions

- int **main** ()
- void **loadCities** ([FlightMap](#) &map, const std::string &fileName)
Loads a list of cities from fileName and adds them to map.
- void **loadFlights** ([FlightMap](#) &map, const std::string &fileName)
Loads a list of flights from fileName and adds them to map.
- void **handleRequests** ([FlightMap](#) &map, const std::string &fileName)
Loads a list of requests from fileName and then processes them.
- void **printFlightPlan** (std::stack< [Flight](#) *> flightPlan)
Takes a flight plan and prints it on screen.

4.3.1 Detailed Description

Program driver implementation file.

Reads in city and flight information, and then processes flight requests

Version

1.00 Alexander Novotny First version

4.3.2 Function Documentation

4.3.2.1 handleRequests()

```
void handleRequests (
    FlightMap & map,
    const std::string & fileName )
```

Loads a list of requests from fileName and then processes them.

Loads a list of requests from fileName, then attempts to find a path for them using map.getPath() and either tells the client that HPAir doesn't fly there or gives them their flight plan

Precondition

fileName contains the name of a valid file
map contains a list of cities filled with flights

Parameters

in	<i>map</i>	The FlightMap to get cities and flights from
in	<i>fileName</i>	The name of the file to load cities from

4.3.2.2 loadCities()

```
void loadCities (
    FlightMap & map,
    const std::string & fileName )
```

Loads a list of cities from fileName and adds them to map.

Precondition

fileName contains the name of a valid file

Postcondition

map will be filled with any new cities found in the file

Parameters

in	<i>map</i>	The FlightMap to add cities to
in	<i>fileName</i>	The name of the file to load cities from

4.3.2.3 loadFlights()

```
void loadFlights (
    FlightMap & map,
    const std::string & fileName )
```

Loads a list of flights from fileName and adds them to map.

Precondition

fileName contains the name of a valid file

Postcondition

Adds any parsed flights from fileName into the cities contained in map

Parameters

in	<i>map</i>	The FlightMap to add flights to
in	<i>fileName</i>	The name of the file to load cities from

4.3.2.4 printFlightPlan()

```
void printFlightPlan (
    std::stack< Flight *> flightPlan )
```

Takes a flight plan and prints it on screen.

Precondition

flightPlan must not be empty

Parameters

in	<i>flightPlan</i>	A stack of flights in order of first to last from top to bottom.
in	<i>fileName</i>	The name of the file to load cities from

4.4 PA03/PA03.h File Reference

Program driver header file.

```
#include <iostream>
#include <fstream>
#include <string>
#include <regex>
#include "FlightMapV1.h"
```

Macros

- `#define CITY_FILE "cityFile.txt"`
- `#define FLIGHT_FILE "flightFile.txt"`
- `#define REQUEST_FILE "requestFile.txt"`

Functions

- void [loadCities](#) ([FlightMap](#) &, const std::string &)
Loads a list of cities from fileName and adds them to map.
- void [loadFlights](#) ([FlightMap](#) &, const std::string &)
Loads a list of flights from fileName and adds them to map.

- void `handleRequest` (`FlightMap` &, const std::string &)
Loads a list of requests from fileName and then processes them.
- void `printFlightPlan` (std::stack< `Flight` *>)
Takes a flight plan and prints it on screen.
- const std::regex `regexFlight` ("^([[:alpha:]]+),\n([[:alpha:]]+)([[:digit:]]+)([[:digit:]]+)\$")
- const std::regex `regexRequest` ("^([[:alpha:]]+),\n([[:alpha:]]+)\$")

4.4.1 Detailed Description

Program driver header file.

Function prototypes, precompiler macros, and includes for program driver

Version

1.00 Alexander Novotny First version

4.4.2 Function Documentation

4.4.2.1 `handleRequest()`

```
void handleRequests (
    FlightMap & map,
    const std::string & fileName )
```

Loads a list of requests from fileName and then processes them.

Loads a list of requests from fileName, then attempts to find a path for them using `map.getPath()` and either tells the client that HPAir doesn't fly there or gives them their flight plan

Precondition

fileName contains the name of a valid file
map contains a list of cities filled with flights

Parameters

in	<i>map</i>	The <code>FlightMap</code> to get cities and flights from
in	<i>fileName</i>	The name of the file to load cities from

4.4.2.2 `loadCities()`

```
void loadCities (
    FlightMap & map,
    const std::string & fileName )
```

Loads a list of cities from fileName and adds them to map.

Precondition

fileName contains the name of a valid file

Postcondition

map will be filled with any new cities found in the file

Parameters

in	<i>map</i>	The FlightMap to add cities to
in	<i>fileName</i>	The name of the file to load cities from

4.4.2.3 loadFlights()

```
void loadFlights (
    FlightMap & map,
    const std::string & fileName )
```

Loads a list of flights from fileName and adds them to map.

Precondition

fileName contains the name of a valid file

Postcondition

Adds any parsed flights from fileName into the cities contained in map

Parameters

in	<i>map</i>	The FlightMap to add flights to
in	<i>fileName</i>	The name of the file to load cities from

4.4.2.4 printFlightPlan()

```
void printFlightPlan (
    std::stack< Flight *> flightPlan )
```

Takes a flight plan and prints it on screen.

Precondition

flightPlan must not be empty

Parameters

in	<i>flightPlan</i>	A stack of flights in order of first to last from top to bottom.
in	<i>fileName</i>	The name of the file to load cities from

Index

- addCity
 - FlightMap, [6](#)
- City, [5](#)
- Flight, [5](#)
- FlightMap, [5](#)
 - addCity, [6](#)
 - getCity, [6](#)
 - getPath, [6](#)
- FlightMapV1.cpp
 - log, [9](#)
- FlightMapV1.h
 - log, [10](#)
- getCity
 - FlightMap, [6](#)
- getPath
 - FlightMap, [6](#)
- handleRequest
 - PA03.cpp, [11](#)
 - PA03.h, [14](#)
- loadCities
 - PA03.cpp, [12](#)
 - PA03.h, [14](#)
- loadFlights
 - PA03.cpp, [12](#)
 - PA03.h, [15](#)
- log
 - FlightMapV1.cpp, [9](#)
 - FlightMapV1.h, [10](#)
- PA03.cpp
 - handleRequest, [11](#)
 - loadCities, [12](#)
 - loadFlights, [12](#)
 - printFlightPlan, [13](#)
- PA03.h
 - handleRequest, [14](#)
 - loadCities, [14](#)
 - loadFlights, [15](#)
 - printFlightPlan, [15](#)
- PA03/FlightMapV1.cpp, [9](#)
- PA03/FlightMapV1.h, [10](#)
- PA03/PA03.cpp, [11](#)
- PA03/PA03.h, [13](#)
- printFlightPlan
 - PA03.cpp, [13](#)
 - PA03.h, [15](#)