

labo_07_schaufelberger_yannick_gallay_david

Generated by Doxygen 1.8.13

Contents

1	File Index	1
1.1	File List	1
2	File Documentation	3
2.1	constants.h File Reference	3
2.1.1	Enumeration Type Documentation	3
2.1.1.1	MONTHS	3
2.1.2	Variable Documentation	4
2.1.2.1	DATE_SEPARATOR	4
2.1.2.2	MAX_DAY	4
2.1.2.3	MAX_MONTH	4
2.1.2.4	MAX_YEAR	4
2.1.2.5	MIN_DAY	4
2.1.2.6	MIN_MONTH	5
2.1.2.7	MIN_YEAR	5
2.2	delta_date.h File Reference	5
2.2.1	Function Documentation	5
2.2.1.1	ask_and_compute_delta_day_between_two_dates()	6
2.2.1.2	ask_date()	6
2.2.1.3	ask_for_valid_date()	6
2.2.1.4	ignore_date_separator()	7
2.3	interface.h File Reference	7
2.3.1	Function Documentation	7
2.3.1.1	ask_for_restart()	7

2.3.2	Variable Documentation	7
2.3.2.1	RESTART_CHAR	8
2.3.2.2	STOP_CHAR	8
2.4	utilities.h File Reference	8
2.4.1	Macro Definition Documentation	9
2.4.1.1	CLEAR_BUFFER	9
2.4.2	Function Documentation	9
2.4.2.1	check_date_order()	9
2.4.2.2	days_between_dates()	9
2.4.2.3	get_days_since_reference_day()	10
2.4.2.4	is_date_valid()	10
2.4.2.5	is_leap_year()	11
	Index	13

Chapter 1

File Index

1.1 File List

Here is a list of all files with brief descriptions:

constants.h	3
delta_date.h	5
interface.h	7
utilities.h	8

Chapter 2

File Documentation

2.1 constants.h File Reference

Enumerations

- enum MONTHS {
JANUAR = 1, FEBRUAR, MARCH, APRIL,
MAY, JUNE, JULY, AUGUST,
SEPTEMBER, OCTOBER, NOVEMBER, DECEMBER }

Variables

- const int MIN_DAY = 1
- const int MIN_MONTH = JANUAR
- const int MIN_YEAR = 1900
- const int MAX_DAY = 31
- const int MAX_MONTH = DECEMBER
- const int MAX_YEAR = 2300
- const char DATE_SEPARATOR = '-'

2.1.1 Enumeration Type Documentation

2.1.1.1 MONTHS

enum MONTHS

Enumerator

JANUAR	
FEBRUAR	
MARCH	
APRIL	
MAY	

Enumerator

JUNE	
JULY	
AUGUST	
SEPTEMBER	
OCTOBER	
NOVEMBER	
DECEMBER	

2.1.2 Variable Documentation**2.1.2.1 DATE_SEPARATOR**

```
const char DATE_SEPARATOR = '-'
```

2.1.2.2 MAX_DAY

```
const int MAX_DAY = 31
```

2.1.2.3 MAX_MONTH

```
const int MAX_MONTH = DECEMBER
```

2.1.2.4 MAX_YEAR

```
const int MAX_YEAR = 2300
```

2.1.2.5 MIN_DAY

```
const int MIN_DAY = 1
```


2.1.2.6 MIN_MONTH

```
const int MIN_MONTH = JANUAR
```

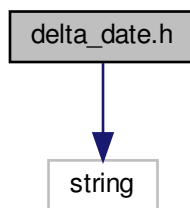
2.1.2.7 MIN_YEAR

```
const int MIN_YEAR = 1900
```

2.2 delta_date.h File Reference

```
#include <string>
```

Include dependency graph for delta_date.h:



Functions

- bool [ignore_date_separator](#) ()
This function clears the next character in the buffer, and returns true if it isn't DATE_SEPARATOR.
- bool [ask_date](#) (const std::string &date, int &day, int &month, int &year)
This function modifies the three referenced arguments to values inputted by the user.
- void [ask_for_valid_date](#) (const std::string &date, int &day, int &month, int &year)
This function asks for a valid date with the format DD-MM-YYYY using ask_date function. As long as the date is invalid, or the format is wrong, it asks for a new date.
- void [ask_and_compute_delta_day_between_two_dates](#) ()
This function asks for a valid dates within a given range of dates in format DD-MM-YYYY and display the number of day between the choosen dates.

2.2.1 Function Documentation

2.2.1.1 ask_and_compute_delta_day_between_two_dates()

```
void ask_and_compute_delta_day_between_two_dates ( )
```

This function asks for a valid dates within a given range of dates in format DD-MM-YYYY and display the number of day between the choosen dates.

2.2.1.2 ask_date()

```
bool ask_date (
    const std::string & date,
    int & day,
    int & month,
    int & year )
```

This function modifies the three referenced arguments to values inputted by the user.

Parameters

<i>date</i>	string telling the user if it's the start date or end date
<i>day</i>	input of the user
<i>month</i>	input of the user
<i>year</i>	input of the user

Returns

true if the date has been retrieve successfully according to DD-MM-YYYY format, else false

2.2.1.3 ask_for_valid_date()

```
void ask_for_valid_date (
    const std::string & date,
    int & day,
    int & month,
    int & year )
```

This function asks for a valid date with the format DD-MM-YYYY using ask_date function. As long as the date is invalid, or the format is wrong, it asks for a new date.

Parameters

<i>date</i>	string telling the user if it's the start date or end date
<i>day</i>	input of the user in function ask_date()
<i>month</i>	input of the user in function ask_date()
<i>year</i>	input of the user in function ask_date()

2.2.1.4 ignore_date_separator()

```
bool ignore_date_separator ( )
```

This function clears the next character in the buffer, and returns true if it isn't DATE_SEPARATOR.

Returns

false if the buffer contains the char DATE_SEPARATOR, true if it doesn't

2.3 interface.h File Reference

Functions

- bool [ask_for_restart](#) ()

This function keeps asking as long as the user enters anything else than RESTART_CHAR or STOP_CHAR.

Variables

- const char [RESTART_CHAR](#) = 'O'
- const char [STOP_CHAR](#) = 'N'

2.3.1 Function Documentation

2.3.1.1 ask_for_restart()

```
bool ask_for_restart ( )
```

This function keeps asking as long as the user enters anything else than RESTART_CHAR or STOP_CHAR.

Returns

true if the user has entered RESTART_CHAR and false if the user has entered STOP_CHAR

2.3.2 Variable Documentation

2.3.2.1 RESTART_CHAR

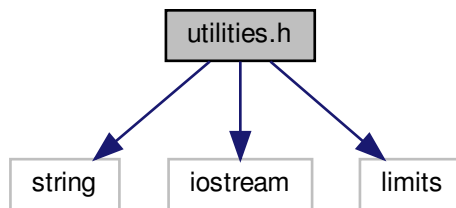
```
const char RESTART_CHAR = 'O'
```

2.3.2.2 STOP_CHAR

```
const char STOP_CHAR = 'N'
```

2.4 utilities.h File Reference

```
#include <string>
#include <iostream>
#include <limits>
Include dependency graph for utilities.h:
```



Macros

- #define [CLEAR_BUFFER](#) `std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n')`

Functions

- bool [is_leap_year](#) (int year)
- bool [is_date_valid](#) (int day, int month, int year)

This function checks that year is between a range of date given by constants, that month is between 1 and 12 and that day is between 1 and 28/29/30/31, depending on the month.
- bool [check_date_order](#) (int start_day, int start_month, int start_year, int end_day, int end_month, int end_year)
- int [days_between_dates](#) (int start_day, int start_month, int start_year, int end_day, int end_month, int end_year)

This returns a negative number if the start date is after the end date.
- int [get_days_since_reference_day](#) (int day, int month, int year)

This function is based on the Julian Day calculation found on this page : <http://www.cs.utsa.edu/~cs1063/projects/Spring2011/Project1/jdn-explanation.html>.

2.4.1 Macro Definition Documentation

2.4.1.1 CLEAR_BUFFER

```
#define CLEAR_BUFFER std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n')
```

2.4.2 Function Documentation

2.4.2.1 check_date_order()

```
bool check_date_order (
    int start_day,
    int start_month,
    int start_year,
    int end_day,
    int end_month,
    int end_year )
```

Parameters

<i>start_day</i>	a number between 1 and 31
<i>start_month</i>	a number between 1 and 12
<i>start_year</i>	a number between 1900 and 2300
<i>end_day</i>	a number between 1 and 31
<i>end_month</i>	a number between 1 and 12
<i>end_year</i>	a number between 1900 and 2300

Returns

true if the date composed of start_day, start_month and start_year is before the date composed of end_day, end_month and end_year

2.4.2.2 days_between_dates()

```
int days_between_dates (
    int start_day,
    int start_month,
    int start_year,
    int end_day,
    int end_month,
    int end_year )
```

This returns a negative number if the start date is after the end date.

Parameters

<i>startDay</i>	
<i>startMonth</i>	
<i>startYear</i>	
<i>endDay</i>	
<i>endMonth</i>	
<i>endYear</i>	

Returns

the number of days between the start date and the end date

2.4.2.3 get_days_since_reference_day()

```
int get_days_since_reference_day (
    int day,
    int month,
    int year )
```

This function is based on the Julian Day calculation found on this page : <http://www.cs.utsa.edu/~cs1063/projects/Spring2011/Project1/jdn-explanation.html>.

Parameters

<i>day</i>	
<i>month</i>	
<i>year</i>	

Returns

the number of days between January 1st 1900 and the date defined by day, month and year

2.4.2.4 is_date_valid()

```
bool is_date_valid (
    int day,
    int month,
    int year )
```

This function checks that year is between a range of date given by constants, that month is between 1 and 12 and that day is between 1 and 28/29/30/31, depending on the month.

Parameters

<i>day</i>	any number
<i>month</i>	any number
<i>year</i>	any number

Returns

true if the date composed of day, month and year is a valid date

2.4.2.5 is_leap_year()

```
bool is_leap_year (
    int year )
```

Parameters

<i>year</i>	any given year
-------------	----------------

Returns

true if year is a leap year, else false

Index

ask_and_compute_delta_day_between_two_dates
 delta_date.h, 5

ask_date
 delta_date.h, 6

ask_for_restart
 interface.h, 7

ask_for_valid_date
 delta_date.h, 6

CLEAR_BUFFER
 utilities.h, 9

check_date_order
 utilities.h, 9

constants.h, 3

 DATE_SEPARATOR, 4

 MAX_DAY, 4

 MAX_MONTH, 4

 MAX_YEAR, 4

 MIN_DAY, 4

 MIN_MONTH, 4

 MIN_YEAR, 5

 MONTHS, 3

DATE_SEPARATOR
 constants.h, 4

days_between_dates
 utilities.h, 9

delta_date.h, 5

 ask_and_compute_delta_day_between_two_dates, 5

 ask_date, 6

 ask_for_valid_date, 6

 ignore_date_separator, 7

get_days_since_reference_day
 utilities.h, 10

ignore_date_separator
 delta_date.h, 7

interface.h, 7

 ask_for_restart, 7

 RESTART_CHAR, 7

 STOP_CHAR, 8

is_date_valid
 utilities.h, 10

is_leap_year
 utilities.h, 11

MAX_DAY
 constants.h, 4

MAX_MONTH
 constants.h, 4

MAX_YEAR
 constants.h, 4

MIN_DAY
 constants.h, 4

MIN_MONTH
 constants.h, 4

MIN_YEAR
 constants.h, 5

MONTHS
 constants.h, 3

RESTART_CHAR
 interface.h, 7

STOP_CHAR
 interface.h, 8

utilities.h, 8

 CLEAR_BUFFER, 9

 check_date_order, 9

 days_between_dates, 9

 get_days_since_reference_day, 10

 is_date_valid, 10

 is_leap_year, 11