

labo\_07\_schaufelberger\_yannick\_gallay\_david

Generated by Doxygen 1.8.13



# Contents

<b>1</b>	<b>File Index</b>	<b>1</b>
1.1	File List	1
<b>2</b>	<b>File Documentation</b>	<b>3</b>
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
2.3	interface.h File Reference	5
2.3.1	Function Documentation	5
2.3.1.1	ask_for_restart()	5
2.3.2	Variable Documentation	5
2.3.2.1	RESTART_CHAR	6
2.3.2.2	STOP_CHAR	6
2.4	utilities.h File Reference	6
2.4.1	Macro Definition Documentation	6
2.4.1.1	CLEAR_BUFFER	6
2.4.2	Function Documentation	7
2.4.2.1	check_date_order()	7
2.4.2.2	days_between_dates()	7
2.4.2.3	get_days_since_reference_day()	8
2.4.2.4	is_date_valid()	8
2.4.2.5	is_leap_year()	9
	<b>Index</b>	<b>11</b>



# Chapter 1

## File Index

### 1.1 File List

Here is a list of all files with brief descriptions:

<a href="#">constants.h</a>	.....	3
<a href="#">delta_date.h</a>	.....	5
<a href="#">interface.h</a>	.....	5
<a href="#">utilities.h</a>	.....	6



# 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:

## 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 startDay, int startMonth, int startYear, int endDay, int endMonth, int endYear)
 

*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 heavily inspired by the Julian Day calculation found on this page : <http://www.cs.utsa.edu/~cs1063/projects/Spring2011/Project1/jdn-explanation.html> It has been slightly modified to fit the current project.*

### 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 startDay,
    int startMonth,
    int startYear,
    int endDay,
    int endMonth,
    int endYear )
```

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 heavily inspired by the Julian Day calculation found on this page : <http://www.cs.utsa.edu/~cs1063/projects/Spring2011/Project1/jdn-explanation.html> It has been slightly modified to fit the current project.

**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\_for\_restart  
interface.h, 5

CLEAR\_BUFFER  
utilities.h, 6

check\_date\_order  
utilities.h, 7

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, 7

delta\_date.h, 5

get\_days\_since\_reference\_day  
utilities.h, 8

interface.h, 5  
ask\_for\_restart, 5  
RESTART\_CHAR, 5  
STOP\_CHAR, 6

is\_date\_valid  
utilities.h, 8

is\_leap\_year  
utilities.h, 8

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, 5

STOP\_CHAR  
interface.h, 6

utilities.h, 6  
CLEAR\_BUFFER, 6  
check\_date\_order, 7  
days\_between\_dates, 7  
get\_days\_since\_reference\_day, 8  
is\_date\_valid, 8  
is\_leap\_year, 8