

Hotel

Generated by Doxygen 1.9.3

1 Hotel Project Documentation	1
1.1 Overview	1
1.2 Achitecture	1
1.2.1 Supporting classes	1
1.2.2 Example	2
2 Bibliography	3
2.1 GitHub	3
2.2 Used materials	3
3 User commands	5
3.1 The UI works with the following commands:	5
3.2 Adding new reservation	5
3.3 Free rooms on particular date	5
3.4 Free particular room	5
3.5 Create report about the usage of the rooms in particular period	5
3.6 Suggest room for a group of guests and partcular period	6
3.7 Close a room for maintenance	6
3.8 See current state of all the rooms	6
3.9 See when a room is free for certain number of days	6
4 Class Index	7
4.1 Class List	7
5 File Index	9
5.1 File List	9
6 Class Documentation	11
6.1 Date Class Reference	11
6.1.1 Detailed Description	12
6.1.2 Constructor & Destructor Documentation	12
6.1.2.1 Date()	12
6.1.3 Member Function Documentation	12
6.1.3.1 getToday()	12
6.1.3.2 operator>()	13
6.1.3.3 operator++()	13
6.1.3.4 operator-()	13
6.1.3.5 operator<()	13
6.1.3.6 operator<=()	14
6.1.3.7 operator==()	14
6.1.3.8 operator>()	14
6.1.3.9 operator>=()	15
6.1.3.10 readDataFromBinary()	15
6.1.3.11 writeToBinaryFile()	15

6.1.4 Friends And Related Function Documentation	15
6.1.4.1 operator<<	15
6.1.4.2 operator>>	16
6.2 DatePeriod Struct Reference	16
6.2.1 Detailed Description	17
6.2.2 Member Function Documentation	17
6.2.2.1 length()	17
6.2.2.2 operator++()	18
6.2.2.3 readProper()	18
6.2.3 Member Data Documentation	18
6.2.3.1 from	18
6.2.3.2 to	18
6.3 Hotel Class Reference	18
6.3.1 Detailed Description	19
6.3.2 Constructor & Destructor Documentation	19
6.3.2.1 Hotel() [1/3]	20
6.3.2.2 Hotel() [2/3]	20
6.3.2.3 Hotel() [3/3]	20
6.3.2.4 ~Hotel()	20
6.3.3 Member Function Documentation	20
6.3.3.1 freeRoom()	20
6.3.3.2 getName()	21
6.3.3.3 getReport()	21
6.3.3.4 nextDay()	21
6.3.3.5 operator=()	22
6.3.3.6 reserveRoom()	22
6.3.3.7 searchRoom()	22
6.3.3.8 seeRoomForNights()	23
6.3.3.9 serviceRoom()	23
6.3.3.10 showAvailableRooms()	23
6.3.3.11 showToday()	24
6.3.3.12 today()	24
6.3.3.13 workDay()	24
6.4 HotelBuilding Class Reference	24
6.4.1 Detailed Description	25
6.4.2 Constructor & Destructor Documentation	25
6.4.2.1 HotelBuilding() [1/2]	25
6.4.2.2 HotelBuilding() [2/2]	26
6.4.2.3 ~HotelBuilding()	26
6.4.3 Member Function Documentation	26
6.4.3.1 createReport()	26
6.4.3.2 getRoomCount()	26

6.4.3.3 newDate()	27
6.4.3.4 operator=()	27
6.4.3.5 operator[]()	27
6.4.3.6 readDataFromBinary()	27
6.4.3.7 showAvailableRooms()	28
6.4.3.8 showRoomForNights()	28
6.4.3.9 showRoomsStatesToday()	28
6.4.3.10 suggestRoom()	29
6.4.3.11 writeToBinaryFile()	29
6.4.4 Friends And Related Function Documentation	29
6.4.4.1 RoomAnalyzer	29
6.5 HotelInterface Class Reference	30
6.5.1 Detailed Description	30
6.5.2 Member Function Documentation	30
6.5.2.1 beginDay()	30
6.5.2.2 createHeader()	30
6.6 Reservation Class Reference	31
6.6.1 Detailed Description	31
6.6.2 Constructor & Destructor Documentation	31
6.6.2.1 Reservation() [1/2]	31
6.6.2.2 Reservation() [2/2]	32
6.6.3 Member Function Documentation	32
6.6.3.1 getFrom()	32
6.6.3.2 getNights()	32
6.6.3.3 getNote()	33
6.6.3.4 getTo()	33
6.6.3.5 isActive()	33
6.6.3.6 isPast()	33
6.6.3.7 isServiced()	34
6.6.3.8 LeavingInAdvance()	34
6.6.3.9 onDate()	34
6.6.3.10 operator=()	34
6.6.3.11 readDataFromBinary()	35
6.6.3.12 stateOnDate()	35
6.6.3.13 writeToBinaryFile()	35
6.7 Room Class Reference	35
6.7.1 Detailed Description	36
6.7.2 Constructor & Destructor Documentation	36
6.7.2.1 Room() [1/2]	36
6.7.2.2 Room() [2/2]	37
6.7.2.3 ~Room()	37
6.7.3 Member Function Documentation	37

6.7.3.1 addReservation()	37
6.7.3.2 closeForService()	38
6.7.3.3 freeRoom()	38
6.7.3.4 getBedCount()	38
6.7.3.5 getNumber()	38
6.7.3.6 isFreeInPeriod()	38
6.7.3.7 isFreeNow()	39
6.7.3.8 isFreeOnDate()	39
6.7.3.9 newDate()	39
6.7.3.10 operator=()	39
6.7.3.11 readDataFromBinary()	39
6.7.3.12 showActivity()	40
6.7.3.13 showReservationsInPeriod()	40
6.7.3.14 writeToBinaryFile()	40
6.8 RoomAnalyzer Class Reference	41
6.8.1 Detailed Description	41
6.8.2 Member Function Documentation	41
6.8.2.1 soonestFreePeriod()	41
6.8.2.2 suggest()	42
6.9 String Class Reference	42
6.9.1 Detailed Description	43
6.9.2 Constructor & Destructor Documentation	43
6.9.2.1 String() [1/2]	43
6.9.2.2 String() [2/2]	43
6.9.2.3 ~String()	43
6.9.3 Member Function Documentation	43
6.9.3.1 c_str()	44
6.9.3.2 capacity()	44
6.9.3.3 operator const char *()	44
6.9.3.4 operator+()	44
6.9.3.5 operator+=()	44
6.9.3.6 operator=()	44
6.9.3.7 shrink_to_fit()	45
6.9.3.8 size()	45
7 File Documentation	47
7.1 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/3.23.0-rc2/CompilerIdC/↵ CMakeCCompilerId.c File Reference	47
7.1.1 Macro Definition Documentation	47
7.1.1.1 __has_include	48
7.1.1.2 ARCHITECTURE_ID	48
7.1.1.3 C_VERSION	48
7.1.1.4 COMPILER_ID	48

7.1.1.5 DEC	48
7.1.1.6 HEX	48
7.1.1.7 PLATFORM_ID	49
7.1.1.8 STRINGIFY	49
7.1.1.9 STRINGIFY_HELPER	49
7.1.2 Function Documentation	49
7.1.2.1 main()	49
7.1.3 Variable Documentation	49
7.1.3.1 info_arch	49
7.1.3.2 info_compiler	49
7.1.3.3 info_language_extensions_default	50
7.1.3.4 info_language_standard_default	50
7.1.3.5 info_platform	50
7.2 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/3.23.0-rc2/CompilerIdCXX/CMakeCXXCompilerId.cpp File Reference ↵	50
7.2.1 Macro Definition Documentation	51
7.2.1.1 __has_include	51
7.2.1.2 ARCHITECTURE_ID	51
7.2.1.3 COMPILER_ID	51
7.2.1.4 CXX_STD	51
7.2.1.5 DEC	51
7.2.1.6 HEX	52
7.2.1.7 PLATFORM_ID	52
7.2.1.8 STRINGIFY	52
7.2.1.9 STRINGIFY_HELPER	52
7.2.2 Function Documentation	52
7.2.2.1 main()	52
7.2.3 Variable Documentation	52
7.2.3.1 info_arch	53
7.2.3.2 info_compiler	53
7.2.3.3 info_language_extensions_default	53
7.2.3.4 info_language_standard_default	53
7.2.3.5 info_platform	53
7.3 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/Date.cpp.obj.d File Reference	54
7.4 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/Hotel.cpp.obj.d File Reference	54
7.5 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/HotelBuilding.cpp.obj.d File Reference ↵	54
7.6 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/HotelInterface.cpp.obj.d File Reference ↵	54
7.7 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/main.cpp.obj.d File Reference	54

7.8 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/Reservation.cpp.↵ obj.d File Reference	54
7.9 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/Room.cpp.obj.d File Reference	54
7.10 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/RoomAnalyzer.cpp.↵ obj.d File Reference	54
7.11 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/String.cpp.obj.d File Reference	54
7.12 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Constants.hpp File Reference	54
7.12.1 Variable Documentation	56
7.12.1.1 cmdArr	56
7.12.1.2 COMMANDS	56
7.12.1.3 daysFromBeginning	56
7.12.1.4 DISPLAY	57
7.12.1.5 DISPLAY_WIDTH	57
7.12.1.6 INIT_CAPACITY	57
7.12.1.7 STRING_MAX_LENGTH	57
7.13 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Constants.hpp	57
7.14 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Date.cpp File Reference	58
7.14.1 Function Documentation	58
7.14.1.1 operator<<()	58
7.14.1.2 operator>>() [1/2]	59
7.14.1.3 operator>>() [2/2]	59
7.15 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Date.hpp File Reference	59
7.15.1 Function Documentation	61
7.15.1.1 operator>>()	61
7.16 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Date.hpp	61
7.17 markdowns/bibliography.md File Reference	62
7.18 markdowns/commands.md File Reference	62
7.19 markdowns/mainpage.md File Reference	62
7.20 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Hotel.cpp File Reference	62
7.20.1 Function Documentation	63
7.20.1.1 readFromIlfstream()	63
7.21 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Hotel.hpp File Reference	63
7.21.1 Function Documentation	64
7.21.1.1 readFromIlfstream()	64
7.22 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Hotel.hpp	65
7.23 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/HotelBuilding.cpp File Reference	66
7.24 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/HotelBuilding.hpp File Reference	66
7.25 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/HotelBuilding.hpp	67
7.26 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/HotelInterface.cpp File Reference	68
7.27 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/HotelInterface.hpp File Reference	69
7.28 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/HotelInterface.hpp	70

7.29 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/main.cpp File Reference	70
7.29.1 Function Documentation	70
7.29.1.1 main()	71
7.30 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Reservation.cpp File Reference	71
7.31 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Reservation.hpp File Reference	71
7.31.1 Enumeration Type Documentation	73
7.31.1.1 ReservationState	73
7.31.2 Function Documentation	73
7.31.2.1 operator<<()	73
7.32 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Reservation.hpp	73
7.33 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Room.cpp File Reference	75
7.33.1 Function Documentation	75
7.33.1.1 operator<<()	75
7.34 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Room.hpp File Reference	76
7.34.1 Function Documentation	77
7.34.1.1 operator<<()	77
7.35 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Room.hpp	77
7.36 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/RoomAnalyzer.cpp File Reference	79
7.37 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/RoomAnalyzer.hpp File Reference	79
7.38 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/RoomAnalyzer.hpp	81
7.39 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/String.cpp File Reference	81
7.39.1 Function Documentation	82
7.39.1.1 operator<<()	82
7.40 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/String.hpp File Reference	82
7.40.1 Function Documentation	83
7.40.1.1 operator<<()	83
7.41 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/String.hpp	84
7.42 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Types.hpp File Reference	85
7.43 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Types.hpp	85

Chapter 1

Hotel Project Documentation

[Commands](#)

[Bibliography](#)

1.1 Overview

This is information system, serving a hotel. The system allows the user to make reservations, receive up-to-date information about the rooms in the hotel, make suggestion which room is most suitable for the guest. The program runs in console mode with several instructions, being printed on startup. The data for the rooms of the hotel is written in file named "h.rooms" in same directory as the executable. Then if data has already been recorded about the hotel (recognized by its name) the data is being read from a binary file called "<hotel name>.dat". On closing, the data about the hotel is being written again.

1.2 Achitecture

The main class is [Hotel](#), having name, static date keeping track of today's date and a hotel building. The hotel building is a list of rooms, each of which has unique ID and number of beds as long as lists of reservations in future and past. The reservation is object having strings for the reserver's name, a note and period of reservation. The state of a reservation is determined through the hotel's today date.

1.2.1 Supporting classes

I developed several classes to ease the work with the main classes.

- [String](#) - dynamic string used for the name of the hotel and name of guest and note in a reservation;
- [Date](#) - class consisting of day, month and year. Used widely in the project to know how the reservations are positioned in time relative to today's date (for the hotel). For these purposes many operators are overloaded;
- [RoomAnalyzer](#) - utility class, knowing algorithms to be performed on rooms for some results (sorting them or finding best one by criteria);
- [HotellInterface](#) - utility class, taking care of the UI of the console program. Can show the user available commands.

1.2.2 Example

reserve: 102 15/7/2022 22/7/2022 Ivan; has dog

Reservation successfully made!

rooms:

101: No reservations.

102: Next reservation is from 15/7/2022 to 22/7/2022. (*answer depends on current date*)

103: No reservations.

...

report: 1/3/2022 1/4/2022

report "report-2022-03-01.txt" created successfully.

free: 101

Room is already free!

free: 102

Room freed successfully! (*answer depends on current date*)

request: 4 1/8/2022 7/8/2022

1. Number: *<best room>* Beds: *<best room's beds>* : *<available | NOT available>*
between 1/8/2022 and 7/8/2022

1. Number: *<second best room>* Beds: *<second best room's beds>* : *<available | NOT available>* between 1/8/2022 and 7/8/2022

2. Number: *<third best room>* Beds: *<third best room's beds>* : *<available | NOT available>* between 1/8/2022 and 7/8/2022

3. Number: *<fourth best room>* Beds: *<fourth best room's beds>* : *<available | NOT available>* between 1/8/2022 and 7/8/2022

4. Number: *<fifth best room>* Beds: *<fifth best room's beds>* : *<available | NOT available>* between 1/8/2022 and 7/8/2022

maintenance: 103 <today> 20/8/2022 cleaning

Maintenance planned successfully.

rooms:

...

103: Next maintenance is from <today> to 20/8/2022.

...

Chapter 2

Bibliography

2.1 GitHub

My GitHub repository is located at:

https://github.com/KaloyanTs/OOP_Project1

2.2 Used materials

- [String](#) library:
<https://www.cplusplus.com/reference/string/string/>
- Quick sort algorithm:
[https://www.tutorialspoint.com/data_structures_algorithms/quick_sort↵_algorithm.htm](https://www.tutorialspoint.com/data_structures_algorithms/quick_sort_algorithm.htm)
- Working with dates:
<http://www.sunshine2k.de/articles/coding/datediffindays/calcdiffofdatsindates.↵html>
- OOP lectures
- Get current date and time:
https://www.tutorialspoint.com/cplusplus/cpp_date_time.htm

Chapter 3

User commands

3.1 The UI works with the following commands:

(They are displayed on startup in the console)

Note

all dates must be input in format D/M/Y
parameters with [] are mandatory and these with {} are optional

3.2 Adding new reservation

reserve: [Room number] [Accommodation date] [Departure date] {Guest name[:]} {Note}

3.3 Free rooms on particular date

available: [date]

3.4 Free particular room

free: [Room number]

3.5 Create report about the usage of the rooms in particular period

report: [From date] [To date]

3.6 Suggest room for a group of guests and particular period

request: [minimal number of beds] [Accommodation date] [Departure date]

3.7 Close a room for maintenance

maintenance: [room number] [From date] [To date] [Note]

3.8 See current state of all the rooms

rooms:

3.9 See when a room is free for certain number of days

plan: [[Room](#) number] [Number of nights]

Chapter 4

Class Index

4.1 Class List

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

Date	Class representing date with day, month and year	11
DatePeriod	Class containing two dates forming a period of time from Date to Date	16
Hotel	Class representing hotel with name, current Date and a building (list of rooms)	18
HotelBuilding	Class representing list of rooms	24
HotelInterface	Utility Class taking care of the UI of the program	30
Reservation	Class representing information about a reservation	31
Room	Class representing a room in hotel	35
RoomAnalyzer	Utility class to perform algorithms on the rooms in a building	41
String	Class representing dynamic string (array of characters)	42

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Constants.hpp	54
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Date.cpp	58
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Date.hpp	59
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Hotel.cpp	62
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Hotel.hpp	63
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/HotelBuilding.cpp	66
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/HotelBuilding.hpp	66
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/HotelInterface.cpp	68
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/HotelInterface.hpp	69
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/main.cpp	70
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Reservation.cpp	71
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Reservation.hpp	71
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Room.cpp	75
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Room.hpp	76
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/RoomAnalyzer.cpp	79
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/RoomAnalyzer.hpp	79
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/String.cpp	81
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/String.hpp	82
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Types.hpp	85
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/3.23.0-rc2/CompilerIdC/CMakeCCompilerId.c	47
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/3.23.0-rc2/CompilerIdCXX/CMakeCXXCompilerId.cpp	50
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/Date.cpp.obj.d	54
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/Hotel.cpp.obj.d	54
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/HotelBuilding.cpp.obj.d	54
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/HotelInterface.cpp.obj.d	54
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/main.cpp.obj.d	54
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/Reservation.cpp.obj.d	54
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/Room.cpp.obj.d	54
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/RoomAnalyzer.cpp.obj.d	54
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/String.cpp.obj.d	54

Chapter 6

Class Documentation

6.1 Date Class Reference

Class representing date with day, month and year.

```
#include <Date.hpp>
```

Public Member Functions

- [Date](#) (unsigned short d=1, unsigned short m=1, unsigned short y=1900)
Construct a new [Date](#) object from day, month and year.
- bool [operator<](#) (const [Date](#) other) const
checks if this [Date](#) is chronologically before other [Date](#)
- bool [operator<=](#) (const [Date](#) other) const
see operator<
- bool [operator>](#) (const [Date](#) other) const
checks if this [Date](#) is chronologically after other [Date](#)
- bool [operator>=](#) (const [Date](#) other) const
see operator>
- bool [operator==](#) (const [Date](#) other) const
checks if two dates are identical
- const char * [operator\(\)](#) (char *buf) const
records this [Date](#) in buffer in format YYYY-MM-DD
- int [operator-](#) ([Date](#) other) const
- [Date](#) & [operator++](#) ()
overloaded prefix incrementation operator for [Date](#)
- void [writeToBinaryFile](#) (std::ofstream &ofs)
write the [Room](#) data into binary file opened by ofstream
- void [readDataFromBinary](#) (std::ifstream &ifstream)
read the [Room](#) data from binary file opened by ifstream

Static Public Member Functions

- static [Date](#) [getToday](#) ()
Get the today date.

Friends

- `std::istream & operator>> (std::istream &is, Date &d)`
overloaded operator for inputing [Date](#)
- `std::ostream & operator<< (std::ostream &os, const Date &d)`
overloaded operator for outputing [Date](#)

6.1.1 Detailed Description

Class representing date with day, month and year.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 [Date\(\)](#)

```
Date::Date (
    unsigned short d = 1,
    unsigned short m = 1,
    unsigned short y = 1900 ) [inline]
```

Construct a new [Date](#) object from day, month and year.

Default [Date](#) is 1/1/1900

Parameters

<i>d</i>	day
<i>m</i>	month
<i>y</i>	year

6.1.3 Member Function Documentation

6.1.3.1 [getToday\(\)](#)

```
Date Date::getToday ( ) [static]
```

Get the today date.

Returns

[Date](#)

6.1.3.2 operator()

```
const char * Date::operator() (
    char * buf ) const
```

records this [Date](#) in buffer in format YYYY-MM-DD

Parameters

<i>buf</i>	buffer where Date is recorded
------------	---

Returns

const char* pointer to beginning of buf

6.1.3.3 operator++()

```
Date & Date::operator++ ( )
```

overloaded prefix incrementation operator for [Date](#)

Returns

[Date](#)& reference to this [Date](#)

6.1.3.4 operator-()

```
int Date::operator- (
    Date other ) const
```

Parameters

<i>other</i>	Date
--------------	----------------------

Returns

int difference between of this [Date](#) and other

6.1.3.5 operator<()

```
bool Date::operator< (
    const Date other ) const
```

checks if this [Date](#) is chronologically before other [Date](#)

Parameters

<i>other</i>	compared Date
--------------	-------------------------------

Returns

true this [Date](#) is chronologically before other
false this [Date](#) is not chronologically before other

6.1.3.6 operator<=()

```
bool Date::operator<= (
    const Date other ) const
```

see operator<

6.1.3.7 operator==()

```
bool Date::operator== (
    const Date other ) const
```

checks if two dates are identical

Parameters

<i>other</i>	compared Date
--------------	-------------------------------

Returns

true the dates are identical
false the dates are not identical

6.1.3.8 operator>()

```
bool Date::operator> (
    const Date other ) const
```

checks if this [Date](#) is chronologically after other [Date](#)

Parameters

<i>other</i>	compared Date
--------------	-------------------------------

Returns

true this [Date](#) is chronologically after other
false this [Date](#) is not chronologically after other

6.1.3.9 operator>=()

```
bool Date::operator>= (
    const Date other ) const
```

see [operator>](#)

6.1.3.10 readDataFromBinary()

```
void Date::readDataFromBinary (
    std::ifstream & ifs )
```

read the [Room](#) data from binary file opened by ifstream

Parameters

<i>ofs</i>	input stream connected to binary file
------------	---------------------------------------

6.1.3.11 writeToBinaryFile()

```
void Date::writeToBinaryFile (
    std::ofstream & ofs )
```

write the [Room](#) data into binary file opened by ofstream

Parameters

<i>ofs</i>	output stream connected to binary file
------------	--

6.1.4 Friends And Related Function Documentation**6.1.4.1 operator<<**

```
std::ostream & operator<< (
    std::ostream & os,
```

```
const Date & d ) [friend]
```

overloaded operator for outputting [Date](#)

Parameters

<i>os</i>	output stream
<i>d</i>	Date to be output

Returns

std::ostream& reference to the output stream

6.1.4.2 operator>>

```
std::istream & operator>> (
    std::istream & is,
    Date & d ) [friend]
```

overloaded operator for inputting [Date](#)

Parameters

<i>is</i>	input stream
<i>d</i>	Date to be input

Returns

std::istream& reference to the input stream

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

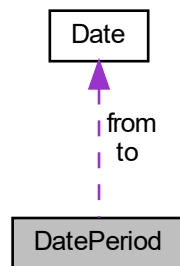
- C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/[Date.hpp](#)
- C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/[Date.cpp](#)

6.2 DatePeriod Struct Reference

Class containing two dates forming a period of time from [Date](#) to [Date](#).

```
#include <Date.hpp>
```

Collaboration diagram for DatePeriod:



Public Member Functions

- unsigned [length](#) () const
distance in days of the period
- [DatePeriod](#) & [operator++](#) ()
moving period one day forward
- void [readProper](#) ()
method to read from stdin a proper period of time (from is before to)

Public Attributes

- [Date](#) [from](#)
beginning [Date](#) of the period
- [Date](#) [to](#)
end [Date](#) of the period

6.2.1 Detailed Description

Class containing two dates forming a period of time from [Date](#) to [Date](#).

6.2.2 Member Function Documentation

6.2.2.1 [length\(\)](#)

```
unsigned DatePeriod::length ( ) const [inline]
```

distance in days of the period

Returns

unsigned days between beginning and end

6.2.2.2 operator++()

```
DatePeriod & DatePeriod::operator++ ( )
```

moving period one day forward

Returns

[DatePeriod](#)& this [DatePeriod](#)

6.2.2.3 readProper()

```
void DatePeriod::readProper ( )
```

method to read from stdin a proper period of time (from is before to)

6.2.3 Member Data Documentation

6.2.3.1 from

```
Date DatePeriod::from
```

beginning [Date](#) of the period

6.2.3.2 to

```
Date DatePeriod::to
```

end [Date](#) of the period

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

- C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/[Date.hpp](#)
- C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/[Date.cpp](#)

6.3 Hotel Class Reference

Class representing hotel with name, current [Date](#) and a building (list of rooms)

```
#include <Hotel.hpp>
```

Public Member Functions

- [Hotel](#) ()=delete
- [Hotel](#) ([String](#) hotelDataFile)
Construct a new [Hotel](#) object.
- [Hotel](#) (const [Hotel](#) &)=delete
- [Hotel](#) & operator= (const [Hotel](#) &)=delete
- [~Hotel](#) ()
Destroy the [Hotel](#) object.
- [String](#) getName () const
get the name of this [Hotel](#)
- void [nextDay](#) ()
advance to the nextDay
- bool [reserveRoom](#) (unsigned number, const [DatePeriod](#) &period, [String](#) name="-", [String](#) note="None\n")
makes a new [Reservation](#) for particular [Room](#) and period with options for name of guest and notes to the [Reservation](#)
- [Hotel](#) & [showAvailableRooms](#) (std::ostream &, [Date](#))
output to stream all available rooms for a particular [Date](#)
- bool [freeRoom](#) (unsigned number)
tries to free [Room](#) with particular ID
- [Hotel](#) & [getReport](#) ([DatePeriod](#) &period)
Creates report for the usage of this [Hotel](#)'s rooms in the period from-to.
- void [searchRoom](#) (unsigned minBeds, const [DatePeriod](#) &period) const
given minimum number of beds and a desired period to print most suitable rooms for accomodation
- bool [serviceRoom](#) (unsigned number, const [DatePeriod](#) &period, [String](#) note)
plans maintenance for particular [Room](#) and period leaving note for the service
- [Hotel](#) & [showToday](#) ()
print status of the building rooms
- [Hotel](#) & [seeRoomForNights](#) (unsigned number, unsigned nights)
print soonest period of particular number of days when particular room is free
- bool [workDay](#) ()
work with this [Hotel](#) for a whole day

Static Public Member Functions

- static [Date](#) today ()
get today's [Date](#) according to all [Hotels](#)

6.3.1 Detailed Description

Class representing hotel with name, current [Date](#) and a building (list of rooms)

6.3.2 Constructor & Destructor Documentation

6.3.2.1 Hotel() [1/3]

```
Hotel::Hotel ( ) [delete]
```

6.3.2.2 Hotel() [2/3]

```
Hotel::Hotel (
    String hotelDataFile )
```

Construct a new [Hotel](#) object.

Parameters

<i>hotelDataFile</i>	path to file where rooms are recorded
----------------------	---------------------------------------

6.3.2.3 Hotel() [3/3]

```
Hotel::Hotel (
    const Hotel & ) [delete]
```

6.3.2.4 ~Hotel()

```
Hotel::~~Hotel ( )
```

Destroy the [Hotel](#) object.

6.3.3 Member Function Documentation

6.3.3.1 freeRoom()

```
bool Hotel::freeRoom (
    unsigned number )
```

tries to free [Room](#) with particular ID

Parameters

<i>number</i>	Room's ID
---------------	---------------------------

Returns

true room is now free
false room not found

6.3.3.2 getName()

```
String Hotel::getName ( ) const [inline]
```

get the name of this [Hotel](#)

Returns

[String](#)

6.3.3.3 getReport()

```
Hotel & Hotel::getReport (
    DatePeriod & period )
```

Creates report for the usage of this [Hotel](#)'s rooms in the period from-to.

Report written in file named "report-YYYY-MM-DD.txt" where YYYY-MM-DD is the beginning of the period

Parameters

<i>period</i>	desired period of time
---------------	------------------------

Returns

[Hotel](#)& this [Hotel](#)

6.3.3.4 nextDay()

```
void Hotel::nextDay ( )
```

advance to the nextDay

6.3.3.5 operator=()

```
Hotel & Hotel::operator= (
    const Hotel & ) [delete]
```

6.3.3.6 reserveRoom()

```
bool Hotel::reserveRoom (
    unsigned number,
    const DatePeriod & period,
    String name = "-",
    String note = "None\n" )
```

makes a new [Reservation](#) for particular [Room](#) and period with options for name of guest and notes to the [Reservation](#)

Parameters

<i>number</i>	of the desired Room
<i>from</i>	accomodation Date
<i>to</i>	leaving Date
<i>name</i>	guest's name
<i>note</i>	note to the reservation

Returns

true successfull reservation
false failed reservation (not made)

6.3.3.7 searchRoom()

```
void Hotel::searchRoom (
    unsigned minBeds,
    const DatePeriod & period ) const
```

given minimum number of beds and a desired period to print most suitable rooms for accomodation

Parameters

<i>minBeds</i>	minimum number of beds
<i>period</i>	desired time period

6.3.3.8 seeRoomForNights()

```
Hotel & Hotel::seeRoomForNights (
    unsigned number,
    unsigned nights )
```

print soonest period of particular number of days when particular room is free

Parameters

<i>number</i>	ID of a room
<i>nights</i>	number of nights to stay in the Hotel for

Returns

[Hotel](#)& this [Hotel](#)

6.3.3.9 serviceRoom()

```
bool Hotel::serviceRoom (
    unsigned number,
    const DatePeriod & period,
    String note )
```

plans maintenance for particular [Room](#) and period leaving note for the service

Parameters

<i>number</i>	Room 's ID
<i>period</i>	desired period of time
<i>note</i>	any notes to the service

Returns

true service planned successfully

false service planning failed (room not found or is reserved for the period)

6.3.3.10 showAvailableRooms()

```
Hotel & Hotel::showAvailableRooms (
    std::ostream & os,
    Date d )
```

output to stream all available rooms for a particular [Date](#)

Returns

[Hotel](#)& this [Hotel](#)

6.3.3.11 showToday()

```
Hotel & Hotel::showToday ( )
```

print status of the building rooms

Returns

[Hotel](#)& this [Hotel](#)

6.3.3.12 today()

```
static Date Hotel::today ( ) [inline], [static]
```

get today's [Date](#) according to all Hotels

Returns

[Date](#)

6.3.3.13 workDay()

```
bool Hotel::workDay ( )
```

work with this [Hotel](#) for a whole day

Returns

true day ended with the [Hotel](#) still working

false the [Hotel](#) was closed and its data written to "<name>.dat"

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

- C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/[Hotel.hpp](#)
- C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/[Hotel.cpp](#)

6.4 HotelBuilding Class Reference

Class representing list of rooms.

```
#include <HotelBuilding.hpp>
```

Public Member Functions

- [HotelBuilding](#) (std::ifstream &ifs)
Construct a new [HotelBuilding](#) object from a text file containing rooms info Format of the file:
- [HotelBuilding](#) (const [HotelBuilding](#) &other)=delete
- [HotelBuilding](#) & operator= ([HotelBuilding](#) &other)=delete
- [~HotelBuilding](#) ()
Destroy the [HotelBuilding](#) object.
- size_t [getRoomCount](#) () const
Get the room count.
- [Room](#) * [operator\[\]](#) (unsigned roomNumber) const
seek for a room with particular number
- void [newDate](#) ([Date](#) d)
update this [HotelBuilding](#) rooms data on a new [Date](#)
- void [showAvailableRooms](#) (std::ostream &os, [Date](#) d) const
show available rooms on a particular [Date](#)
- void [createReport](#) ([DatePeriod](#) &period) const
Create a report for the usage of rooms for a particular period of time (ending before the today [Date](#)) in folder reports
Format of the report:
- void [suggestRoom](#) (unsigned beds, const [DatePeriod](#) &period)
show top DISPLAY (or all rooms if less than DISPLAY + 1) sorted by suitability for the guest, given minimal number of beds and particular period of time for an eventual reservation
- void [showRoomsStatesToday](#) ([Date](#) today) const
prints to stdout all rooms together with up to today [Date](#) info about their Availability (now, in future or in the past)
- void [showRoomForNights](#) (unsigned number, unsigned nights, [Date](#) today) const
show soonest period of particular nights when particular room is free
- void [writeToBinaryFile](#) (std::ofstream &ofs)
write the [HotelBuilding](#) data into binary file opened by ofstream
- void [readDataFromBinary](#) (std::ifstream &ifs)
read the [HotelBuilding](#) data from binary file opened by ifstream

Friends

- class [RoomAnalyzer](#)
using [RoomAnalyzer](#) to perform algorithms for the room list (database)

6.4.1 Detailed Description

Class representing list of rooms.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 [HotelBuilding](#)() [1/2]

```
HotelBuilding::HotelBuilding (
    std::ifstream & ifs )
```

Construct a new [HotelBuilding](#) object from a text file containing rooms info Format of the file:

1. <size> ... n(>1). <room #(n-2) number> <room #(n-2) count of beds>

Parameters

<i>ifs</i>	input file stream to text file, containing rooms data
------------	---

6.4.2.2 HotelBuilding() [2/2]

```
HotelBuilding::HotelBuilding (
    const HotelBuilding & other ) [delete]
```

6.4.2.3 ~HotelBuilding()

```
HotelBuilding::~~HotelBuilding ( )
```

Destroy the [HotelBuilding](#) object.

6.4.3 Member Function Documentation**6.4.3.1 createReport()**

```
void HotelBuilding::createReport (
    DatePeriod & period ) const
```

Create a report for the usage of rooms for a particular period of time (ending before the today [Date](#)) in folder reports
Format of the report:

Report for the usage of the rooms between <beginning of period> and <end of period>: ... [Room #](#) <Room number> between <beginning of period> and <end of period>: <count of nights> nights.

Parameters

<i>period</i>	period of time
---------------	----------------

6.4.3.2 getRoomCount()

```
size_t HotelBuilding::getRoomCount ( ) const [inline]
```

Get the room count.

Returns

size_t count of rooms

6.4.3.3 newDate()

```
void HotelBuilding::newDate (
    Date d )
```

update this [HotelBuilding](#) rooms data on a new [Date](#)

Parameters

<i>d</i>	new Date
----------	--------------------------

6.4.3.4 operator=()

```
HotelBuilding & HotelBuilding::operator= (
    HotelBuilding & other ) [delete]
```

6.4.3.5 operator[]()

```
Room * HotelBuilding::operator[] (
    unsigned roomNumber ) const
```

seek for a room with particular number

Parameters

<i>roomNumber</i>	number if the sought room
-------------------	---------------------------

Returns

Room* if found -> pointer to this [Room](#) else -> nullptr

6.4.3.6 readDataFromBinary()

```
void HotelBuilding::readDataFromBinary (
    std::ifstream & ifs )
```

read the [HotelBuilding](#) data from binary file opened by ifstream

Parameters

<i>ofs</i>	input stream connected to binary file
------------	---------------------------------------

6.4.3.7 showAvailableRooms()

```
void HotelBuilding::showAvailableRooms (
    std::ostream & os,
    Date d ) const
```

show available rooms on a particular [Date](#)

Parameters

<i>os</i>	output stream where the available rooms will be shown in format: Available rooms for <today>: Number: <room number> Bed count: <count of beds> ...
<i>d</i>	Date

6.4.3.8 showRoomForNights()

```
void HotelBuilding::showRoomForNights (
    unsigned number,
    unsigned nights,
    Date today ) const
```

show soonest period of particular nights when particular room is free

Parameters

<i>number</i>	ID of the room
<i>nights</i>	length of the period
<i>today</i>	today's Date

6.4.3.9 showRoomsStatesToday()

```
void HotelBuilding::showRoomsStatesToday (
    Date today ) const
```

prints to stdout all rooms together with up to today [Date](#) info about their Availability (now, in future or in the past)

Parameters

<i>today</i>	Date
--------------	----------------------

6.4.3.10 suggestRoom()

```
void HotelBuilding::suggestRoom (
    unsigned beds,
    const DatePeriod & period )
```

show top DISPLAY (or all rooms if less than DISPLAY + 1) sorted by suitability for the guest, given minimal number of beds and particular period of time for an eventual reservation

Parameters

<i>beds</i>	minimal number of beds insisted in the room
<i>period</i>	period of time

6.4.3.11 writeToBinaryFile()

```
void HotelBuilding::writeToBinaryFile (
    std::ofstream & ofs )
```

write the [HotelBuilding](#) data into binary file opened by ofstream

Parameters

<i>ofs</i>	output stream connected to binary file
------------	--

6.4.4 Friends And Related Function Documentation**6.4.4.1 RoomAnalyzer**

```
friend class RoomAnalyzer [friend]
```

using [RoomAnalyzer](#) to perform algorithms for the room list (database)

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

- C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/[HotelBuilding.hpp](#)
- C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/[HotelBuilding.cpp](#)

6.5 HotelInterface Class Reference

Utility Class taking care of the UI of the program.

```
#include <HotelInterface.hpp>
```

Static Public Member Functions

- static void [createHeader](#) ([Hotel](#) &H)
print to stdout centered name of the [Hotel](#)
- static void [beginDay](#) ()
print the today's date and available commands to use for a [Hotel](#)

6.5.1 Detailed Description

Utility Class taking care of the UI of the program.

6.5.2 Member Function Documentation

6.5.2.1 beginDay()

```
void HotelInterface::beginDay ( ) [static]
```

print the today's date and available commands to use for a [Hotel](#)

6.5.2.2 createHeader()

```
void HotelInterface::createHeader (
    Hotel & H ) [static]
```

print to stdout centered name of the [Hotel](#)

Parameters

<i>H</i>	Hotel whose name is to be printed
----------	---

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

- C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/[HotelInterface.hpp](#)
- C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/[HotelInterface.cpp](#)

6.6 Reservation Class Reference

Class representing information about a reservation.

```
#include <Reservation.hpp>
```

Public Member Functions

- [Reservation](#) ([String](#) name, const [DatePeriod](#) &p, [String](#) n="None.\n", bool s=false)
Construct a new [Reservation](#) object.
- [Reservation](#) (const [Reservation](#) &)=delete
- [Reservation](#) & operator= (const [Reservation](#) &)=delete
- bool [isActive](#) () const
see if this [Reservation](#) is active (today is part of the period)
- bool [isPast](#) () const
see if this [Reservation](#) is past (today is after end of period)
- bool [isServiced](#) () const
see if this [Reservation](#) is a maintenance
- [Date](#) [getFrom](#) () const
get beginning [Date](#) of this [Reservation](#)
- [Date](#) [getTo](#) () const
get end [Date](#) of this [Reservation](#)
- unsigned [getNights](#) () const
get count of nights of this [Reservation](#)
- [String](#) [getNote](#) () const
get the note to this [Reservation](#)
- void [onDate](#) ([Date](#) d)
update the state of the reservation based on new today's [Date](#) (d)
- [ReservationState](#) [stateOnDate](#) ([Date](#)) const
see what would the state of this [Reservation](#) be on particular [Date](#)
- bool [LeavingInAdvance](#) ([Date](#))
try to change end of period for earlier end of this [Reservation](#)
- void [writeToBinaryFile](#) (std::ofstream &ofs)
write the [Reservation](#) data into binary file opened by ofstream
- void [readDataFromBinary](#) (std::ifstream &ifstream)
read the [Reservation](#) data from binary file opened by ifstream

6.6.1 Detailed Description

Class representing information about a reservation.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 [Reservation](#)() [1/2]

```
Reservation::Reservation (
    String name,
    const DatePeriod & p,
    String n = "None.\n",
    bool s = false )
```

Construct a new [Reservation](#) object.

Parameters

<i>name</i>	of the reserver
<i>p</i>	
<i>n</i>	note left for the reservation
<i>s</i>	whether it is reservation or maintenance

6.6.2.2 Reservation() [2/2]

```
Reservation::Reservation (
    const Reservation & ) [delete]
```

6.6.3 Member Function Documentation**6.6.3.1 getFrom()**

```
Date Reservation::getFrom ( ) const [inline]
```

get beginning [Date](#) of this [Reservation](#)

Returns

[Date](#) beginning of the [Reservation](#)

6.6.3.2 getNights()

```
unsigned Reservation::getNights ( ) const [inline]
```

get count of nights of this [Reservation](#)

Returns

unsigned count of nights of this [Reservation](#)

6.6.3.3 getNote()

```
String Reservation::getNote ( ) const [inline]
```

get the note to this [Reservation](#)

Returns

[String](#)

6.6.3.4 getTo()

```
Date Reservation::getTo ( ) const [inline]
```

get end [Date](#) of this [Reservation](#)

Returns

[Date](#) end of the [Reservation](#)

6.6.3.5 isActive()

```
bool Reservation::isActive ( ) const [inline]
```

see if this [Reservation](#) is active (today is part of the period)

Returns

true [Reservation](#) is active

false [Reservation](#) is not active (past or future)

6.6.3.6 isPast()

```
bool Reservation::isPast ( ) const [inline]
```

see if this [Reservation](#) is past (today is after end of period)

Returns

true [Reservation](#) is past

false [Reservation](#) is not past (active or future)

6.6.3.7 isServiced()

```
bool Reservation::isServiced ( ) const [inline]
```

see if this [Reservation](#) is a maintenance

Returns

true this [Reservation](#) is a maintenace

false this [Reservation](#) is for a guest

6.6.3.8 LeavingInAdvance()

```
bool Reservation::LeavingInAdvance (
    Date newTo )
```

try to change end of period for earlier end of this [Reservation](#)

Returns

true end date modified for leaving in advance

false new leaving [Date](#) not appropriate for earlier leaving

6.6.3.9 onDate()

```
void Reservation::onDate (
    Date d )
```

update the state of the reservation based on new today's [Date](#) (d)

Parameters

<i>d</i>	new today's Date
----------	----------------------------------

6.6.3.10 operator=()

```
Reservation & Reservation::operator= (
    const Reservation & ) [delete]
```

6.6.3.11 readDataFromBinary()

```
void Reservation::readDataFromBinary (
    std::ifstream & ifs )
```

read the [Reservation](#) data from binary file opened by ifstream

Parameters

<i>ofs</i>	input stream connected to binary file
------------	---------------------------------------

6.6.3.12 stateOnDate()

```
ReservationState Reservation::stateOnDate (
    Date d ) const
```

see what would the state of this [Reservation](#) be on particular [Date](#)

Returns

ReservationState state on desired [Date](#)

6.6.3.13 writeToBinaryFile()

```
void Reservation::writeToBinaryFile (
    std::ofstream & ofs )
```

write the [Reservation](#) data into binary file opened by ofstream

Parameters

<i>ofs</i>	output stream connected to binary file
------------	--

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

- C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/[Reservation.hpp](#)
- C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/[Reservation.cpp](#)

6.7 Room Class Reference

Class representing a room in hotel.

```
#include <Room.hpp>
```

Public Member Functions

- [Room](#) (unsigned n, unsigned bC)
Construct a new [Room](#) object.
- [Room](#) (const [Room](#) &)=delete
forbidden copying of rooms
- [Room](#) & operator= (const [Room](#) &)=delete
forbidden copying of rooms
- [~Room](#) ()
Destroy the [Room](#) object.
- unsigned [getNumber](#) () const
- unsigned [getBedCount](#) () const
- bool [isFreeNow](#) () const
- bool [freeRoom](#) ([Reservation](#) *¤tRes)
try to free this room
- void [newDate](#) ([Date](#))
apply new [Date](#) to state of all reservations and respectively of the room availability
- bool [isFreeOnDate](#) ([Date](#)) const
see if this room is free in certain date
- bool [isFreeInPeriod](#) (const [DatePeriod](#) &period) const
see if this [Room](#) is free in particular period of time (it is free in all days of the period)
- bool [showReservationsInPeriod](#) (std::ostream &os, const [DatePeriod](#) &period) const
print to output stream info about the number of nights (if positive) in a period this [Room](#) has been taken
- bool [addReservation](#) ([String](#) name, [String](#) note, const [DatePeriod](#) &period)
try to add [Reservation](#) to this [Room](#)
- bool [closeForService](#) ([String](#) note, const [DatePeriod](#) &period)
try to add [Reservation](#) (about a maintenance) to this [Room](#)
- void [showActivity](#) () const
print to stdout information about this [Room](#) latest busyness
- void [writeToBinaryFile](#) (std::ofstream &ofs)
write the [Room](#) data into binary file opened by ofstream
- void [readDataFromBinary](#) (std::ifstream &ifstream)
read the [Room](#) data from binary file opened by ifstream

6.7.1 Detailed Description

Class representing a room in hotel.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 [Room\(\)](#) [1/2]

```
Room::Room (
    unsigned n,
    unsigned bC )
```

Construct a new [Room](#) object.

Parameters

<i>n</i>	number of constructed Room
<i>bC</i>	number of beds in constructed Room

6.7.2.2 Room() [2/2]

```
Room::Room (
    const Room & ) [delete]
```

forbidden copying of rooms

6.7.2.3 ~Room()

```
Room::~~Room ( )
```

Destroy the [Room](#) object.

6.7.3 Member Function Documentation**6.7.3.1 addReservation()**

```
bool Room::addReservation (
    String name,
    String note,
    const DatePeriod & period )
```

try to add [Reservation](#) to this [Room](#)

Parameters

<i>name</i>	name of the guest
<i>note</i>	note to this Reservation
<i>period</i>	period of time

Returns

true successfully added [Reservation](#)

false adding a [Reservation](#) failed (the room is not free in this [DatePeriod](#))

6.7.3.2 closeForService()

```
bool Room::closeForService (
    String note,
    const DatePeriod & period )
```

try to add [Reservation](#) (about a maintenance) to this [Room](#)

Parameters

<i>note</i>	note to this maintenance
<i>period</i>	period of time

Returns

true successfully added maintenance

false adding a maintenance failed (the room is not free in this [DatePeriod](#))

6.7.3.3 freeRoom()

```
bool Room::freeRoom (
    Reservation *& currentRes )
```

try to free this room

Returns

true sucesfully freed room

false room is already free

6.7.3.4 getBedCount()

```
unsigned Room::getBedCount ( ) const [inline]
```

6.7.3.5 getNumber()

```
unsigned Room::getNumber ( ) const [inline]
```

6.7.3.6 isFreeInPeriod()

```
bool Room::isFreeInPeriod (
    const DatePeriod & period ) const
```

see if this [Room](#) is free in particular period of time (it is free in all days of the period)

Parameters

<i>period</i>	period of time
---------------	----------------

Returns

- true the room is free (in all days of the period)
- false the room is not free (there is a day in period when the room is taken)

6.7.3.7 isFreeNow()

```
bool Room::isFreeNow ( ) const
```

6.7.3.8 isFreeOnDate()

```
bool Room::isFreeOnDate (
    Date d ) const
```

see if this room is free in certain date

Returns

- true the room is free
- false the room is taken

6.7.3.9 newDate()

```
void Room::newDate (
    Date newD )
```

apply new [Date](#) to state of all reservations and respectively of the room availability

6.7.3.10 operator=()

```
Room & Room::operator= (
    const Room & ) [delete]
```

forbidden copying of rooms

6.7.3.11 readDataFromBinary()

```
void Room::readDataFromBinary (
    std::ifstream & ifs )
```

read the [Room](#) data from binary file opened by ifstream

Parameters

<i>ofs</i>	input stream connected to binary file
------------	---------------------------------------

6.7.3.12 showActivity()

```
void Room::showActivity ( ) const
```

print to stdout information about this [Room](#) latest busyness

6.7.3.13 showReservationsInPeriod()

```
bool Room::showReservationsInPeriod (
    std::ostream & os,
    const DatePeriod & period ) const
```

print to output stream info about the number of nights (if positive) in a period this [Room](#) has been taken

Parameters

<i>os</i>	output stream
<i>period</i>	period of time

Returns

true there has been taken for at least one night and info has been printed

false the room has been free during this period and no info has been printed

6.7.3.14 writeToBinaryFile()

```
void Room::writeToBinaryFile (
    std::ofstream & ofs )
```

write the [Room](#) data into binary file opened by ofstream

Parameters

<i>ofs</i>	output stream connected to binary file
------------	--

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

- C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Room.hpp
- C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Room.cpp

6.8 RoomAnalyzer Class Reference

Utility class to perform algorithms on the rooms in a building.

```
#include <RoomAnalyzer.hpp>
```

Static Public Member Functions

- static void [suggest](#) ([HotelBuilding](#) &hB, unsigned beds, [DatePeriod](#) period)
print top DISPLAY rooms info based on suitability of a [Room](#) (desired number of beds and period of time)
- static void [soonestFreePeriod](#) (const [HotelBuilding](#) &hB, unsigned number, unsigned nights, [Date](#) today)
print soonest period when a particular room is free for particular number of nights

6.8.1 Detailed Description

Utility class to perform algorithms on the rooms in a building.

6.8.2 Member Function Documentation

6.8.2.1 [soonestFreePeriod\(\)](#)

```
void RoomAnalyzer::soonestFreePeriod (  
    const HotelBuilding & hB,  
    unsigned number,  
    unsigned nights,  
    Date today ) [static]
```

print soonest period when a particular room is free for particular number of nights

Parameters

<i>hB</i>	HotelBuilding
<i>number</i>	ID of the Room
<i>nights</i>	length of a period
<i>today</i>	today's Date

6.8.2.2 suggest()

```
void RoomAnalyzer::suggest (
    HotelBuilding & hB,
    unsigned beds,
    DatePeriod period ) [static]
```

print top DISPLAY rooms info based on suitability of a [Room](#) (desired number of beds and period of time)

Parameters

<i>hB</i>	HotelBuilding
<i>beds</i>	desired number of beds
<i>period</i>	period of time

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

- C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/[RoomAnalyzer.hpp](#)
- C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/[RoomAnalyzer.cpp](#)

6.9 String Class Reference

Class representing dynamic string (array of characters)

```
#include <String.hpp>
```

Public Member Functions

- [String](#) (const char *_str="")
Construct a new [String](#) object by array of chars.
- [String](#) (const [String](#) &other)
copy constructor (makes deep copy of other)
- [String](#) & [operator=](#) (const [String](#) &other)
makes deep copy of other into this
- [~String](#) ()
Destroy the [String](#) object.
- size_t [size](#) () const
get the size of the array (number of symbols before the '\0')
- size_t [capacity](#) () const
get the maximum characters the array can store
- char * [c_str](#) () const
get pointer to the beginning of the array
- [String](#) [operator+](#) (const [String](#) &other) const
concatenation of two strings
- [String](#) & [operator+=](#) (const [String](#) &other)
concatenate another [String](#) to this [String](#)
- [operator const char *](#) () const
*cast this [String](#) to const char * by returning pointer to the beginning*
- [String](#) & [shrink_to_fit](#) ()
make the capacity equal to the size

6.9.1 Detailed Description

Class representing dynamic string (array of characters)

6.9.2 Constructor & Destructor Documentation

6.9.2.1 String() [1/2]

```
String::String (
    const char * _str = "" )
```

Construct a new [String](#) object by array of chars.

Parameters

<code>_str</code>	
-------------------	--

6.9.2.2 String() [2/2]

```
String::String (
    const String & other )
```

copy constructor (makes deep copy of other)

Parameters

<code>other</code>	String
--------------------	------------------------

6.9.2.3 ~String()

```
String::~~String ( )
```

Destroy the [String](#) object.

6.9.3 Member Function Documentation

6.9.3.1 c_str()

```
char * String::c_str ( ) const [inline]
```

get pointer to the beginning of the array

6.9.3.2 capacity()

```
size_t String::capacity ( ) const [inline]
```

get the maximum characters the array can store

6.9.3.3 operator const char *()

```
String::operator const char * ( ) const [inline]
```

cast this [String](#) to const char * by returning pointer to the beginning

6.9.3.4 operator+()

```
String String::operator+ (
    const String & other ) const
```

concatenation of two strings

6.9.3.5 operator+=()

```
String & String::operator+= (
    const String & other )
```

concatenate another [String](#) to this [String](#)

6.9.3.6 operator=()

```
String & String::operator= (
    const String & other )
```

makes deep copy of other into this

Parameters

<i>other</i>	String
--------------	------------------------

Returns

[String](#)& this [String](#)

6.9.3.7 shrink_to_fit()

```
String & String::shrink_to_fit ( )
```

make the capacity equal to the size

6.9.3.8 size()

```
size_t String::size ( ) const [inline]
```

get the size of the array (number of symbols before the '\0')

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

- C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/[String.hpp](#)
- C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/[String.cpp](#)

Chapter 7

File Documentation

7.1 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/3.23.0-rc2/CompilerIdC/CMakeCCompilerId.c File Reference

Macros

- #define `__has_include(x)` 0
- #define `COMPILER_ID` ""
- #define `STRINGIFY_HELPER(X)` #X
- #define `STRINGIFY(X)` `STRINGIFY_HELPER(X)`
- #define `PLATFORM_ID`
- #define `ARCHITECTURE_ID`
- #define `DEC(n)`
- #define `HEX(n)`
- #define `C_VERSION`

Functions

- int `main` (int argc, char *argv[])

Variables

- char const * `info_compiler` = "INFO" ":" "compiler[" `COMPILER_ID` "]"
- char const * `info_platform` = "INFO" ":" "platform[" `PLATFORM_ID` "]"
- char const * `info_arch` = "INFO" ":" "arch[" `ARCHITECTURE_ID` "]"
- const char * `info_language_standard_default`
- const char * `info_language_extensions_default`

7.1.1 Macro Definition Documentation

7.1.1.1 `__has_include`

```
#define __has_include(  
    x ) 0
```

7.1.1.2 `ARCHITECTURE_ID`

```
#define ARCHITECTURE_ID
```

7.1.1.3 `C_VERSION`

```
#define C_VERSION
```

7.1.1.4 `COMPILER_ID`

```
#define COMPILER_ID ""
```

7.1.1.5 `DEC`

```
#define DEC(  
    n )
```

Value:

```
('0' + ((n) / 10000000) % 10), \  
( '0' + ((n) / 1000000) % 10), \  
( '0' + ((n) / 100000) % 10), \  
( '0' + ((n) / 10000) % 10), \  
( '0' + ((n) / 1000) % 10), \  
( '0' + ((n) / 100) % 10), \  
( '0' + ((n) / 10) % 10), \  
( '0' + ((n) % 10))
```

7.1.1.6 `HEX`

```
#define HEX(  
    n )
```

Value:

```
('0' + ((n) >> 28 & 0xF)), \  
( '0' + ((n) >> 24 & 0xF)), \  
( '0' + ((n) >> 20 & 0xF)), \  
( '0' + ((n) >> 16 & 0xF)), \  
( '0' + ((n) >> 12 & 0xF)), \  
( '0' + ((n) >> 8 & 0xF)), \  
( '0' + ((n) >> 4 & 0xF)), \  
( '0' + ((n) & 0xF))
```

7.1.1.7 PLATFORM_ID

```
#define PLATFORM_ID
```

7.1.1.8 STRINGIFY

```
#define STRINGIFY(  
    X ) STRINGIFY_HELPER(X)
```

7.1.1.9 STRINGIFY_HELPER

```
#define STRINGIFY_HELPER(  
    X ) #X
```

7.1.2 Function Documentation

7.1.2.1 main()

```
int main (  
    int argc,  
    char * argv[] )
```

7.1.3 Variable Documentation

7.1.3.1 info_arch

```
char const* info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
```

7.1.3.2 info_compiler

```
char const* info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
```

7.1.3.3 info_language_extensions_default

```
const char* info_language_extensions_default
```

Initial value:

```
= "INFO" ":" "extensions_default["  
  "OFF"  
"]"
```

7.1.3.4 info_language_standard_default

```
const char* info_language_standard_default
```

Initial value:

```
=  
  "INFO" ":" "standard_default[" C_VERSION "]"
```

7.1.3.5 info_platform

```
char const* info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
```

7.2 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/3.23.0-rc2/CompilerIdCXX/CMakeCXXCompilerId.cpp File Reference

Macros

- #define `__has_include(x)` 0
- #define `COMPILER_ID` ""
- #define `STRINGIFY_HELPER(X)` #X
- #define `STRINGIFY(X)` `STRINGIFY_HELPER(X)`
- #define `PLATFORM_ID`
- #define `ARCHITECTURE_ID`
- #define `DEC(n)`
- #define `HEX(n)`
- #define `CXX_STD` __cplusplus

Functions

- int `main` (int argc, char *argv[])

Variables

- char const * [info_compiler](#) = "INFO" ":" "compiler[" COMPILER_ID "]"
- char const * [info_platform](#) = "INFO" ":" "platform[" PLATFORM_ID "]"
- char const * [info_arch](#) = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
- const char * [info_language_standard_default](#)
- const char * [info_language_extensions_default](#)

7.2.1 Macro Definition Documentation

7.2.1.1 __has_include

```
#define __has_include(  
    x ) 0
```

7.2.1.2 ARCHITECTURE_ID

```
#define ARCHITECTURE_ID
```

7.2.1.3 COMPILER_ID

```
#define COMPILER_ID ""
```

7.2.1.4 CXX_STD

```
#define CXX_STD __cplusplus
```

7.2.1.5 DEC

```
#define DEC(  
    n )
```

Value:

```
( '0' + ((n) / 10000000) % 10 ), \  
( '0' + ((n) / 1000000) % 10 ), \  
( '0' + ((n) / 100000) % 10 ), \  
( '0' + ((n) / 10000) % 10 ), \  
( '0' + ((n) / 1000) % 10 ), \  
( '0' + ((n) / 100) % 10 ), \  
( '0' + ((n) / 10) % 10 ), \  
( '0' + ((n) % 10) )
```

7.2.1.6 HEX

```
#define HEX(  
    n )
```

Value:

```
('0' + ((n)>>28 & 0xF)), \  
( '0' + ((n)>>24 & 0xF)), \  
( '0' + ((n)>>20 & 0xF)), \  
( '0' + ((n)>>16 & 0xF)), \  
( '0' + ((n)>>12 & 0xF)), \  
( '0' + ((n)>>8  & 0xF)), \  
( '0' + ((n)>>4  & 0xF)), \  
( '0' + ((n)    & 0xF))
```

7.2.1.7 PLATFORM_ID

```
#define PLATFORM_ID
```

7.2.1.8 STRINGIFY

```
#define STRINGIFY(  
    X ) STRINGIFY_HELPER(X)
```

7.2.1.9 STRINGIFY_HELPER

```
#define STRINGIFY_HELPER(  
    X ) #X
```

7.2.2 Function Documentation

7.2.2.1 main()

```
int main (  
    int argc,  
    char * argv[] )
```

7.2.3 Variable Documentation

7.2.3.1 info_arch

```
char const* info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
```

7.2.3.2 info_compiler

```
char const* info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
```

7.2.3.3 info_language_extensions_default

```
const char* info_language_extensions_default
```

Initial value:

```
= "INFO" ":" "extensions_default["  
  "OFF"  
"]"
```

7.2.3.4 info_language_standard_default

```
const char* info_language_standard_default
```

Initial value:

```
= "INFO" ":" "standard_default["  
  "98"  
"]"
```

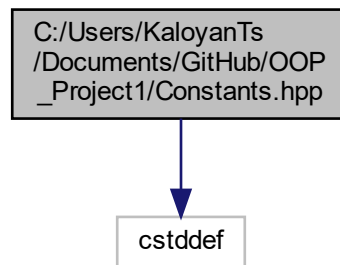
7.2.3.5 info_platform

```
char const* info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
```

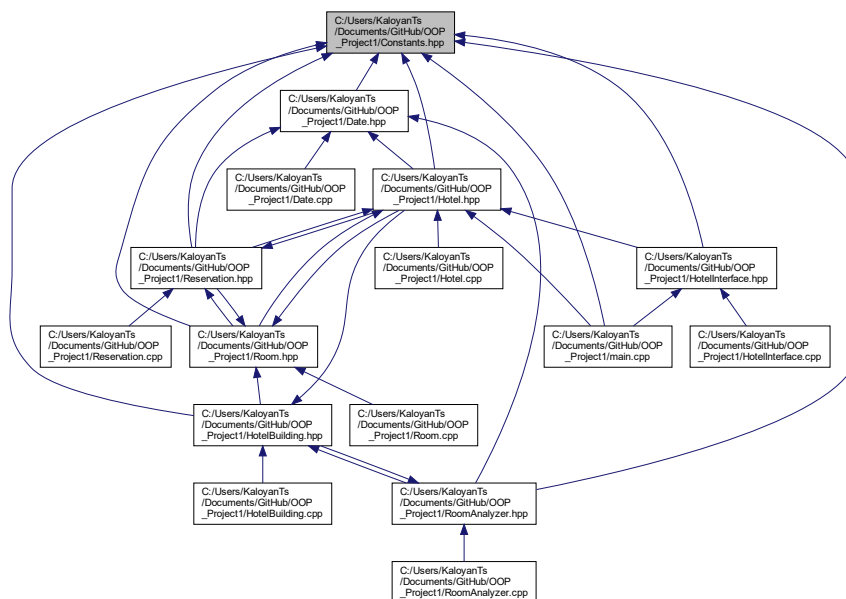
- 7.3 **C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/Date.cpp.obj.d** File Reference
- 7.4 **C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/Hotel.cpp.obj.d** File Reference
- 7.5 **C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/HotelBuilding.cpp.obj.d** File Reference
- 7.6 **C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/HotelInterface.cpp.obj.d** File Reference
- 7.7 **C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/main.cpp.obj.d** File Reference
- 7.8 **C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/Reservation.cpp.obj.d** File Reference
- 7.9 **C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/Room.cpp.obj.d** File Reference
- 7.10 **C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/RoomAnalyzer.cpp.obj.d** File Reference
- 7.11 **C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/String.cpp.obj.d** File Reference
- 7.12 **C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Constants.hpp** File Reference

```
#include <cstdint>
```


Include dependency graph for Constants.hpp:



This graph shows which files directly or indirectly include this file:



Variables

- const unsigned `daysFromBeginning` [] = {0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334}
array keeping days past from January 1st
- const size_t `DISPLAY_WIDTH` = 130
width of the console used for centering the text
- const size_t `COMMANDS` = 8
number of different commands to use for a *Hotel*
- const size_t `STRING_MAX_LENGTH` = 128
max length of strings (char arrays)
- const size_t `INIT_CAPACITY` = 2

- const variable keeping minimal size of the reservations list*
- const size_t `DISPLAY` = 5
max number of rooms to be printed
- const char `cmdArr` [`COMMANDS`][2][`STRING_MAX_LENGTH`]
array of all available commands and instructions how to use them

7.12.1 Variable Documentation

7.12.1.1 cmdArr

```
const char cmdArr[COMMANDS][2][STRING_MAX_LENGTH]
```

Initial value:

```
= {
    {"To make a reservation, enter ",
     {"<reserve: [Room number] [Accommodation date] [Departure date] {Guest name[;]} {Note}>"},
     {"To see list of free rooms for a particular date, enter ",
      {"<available: [date]>"},
     {"To free a room now, enter ",
      {"<free: [Room number]>"},
     {"To get report about the reservations of a room over a period of time, enter ",
      {"<report: [From date] [To date]>"},
     {"To request a room for guests, enter",
      {"<request: [minimal number of beds] [Accommodation date] [Departure date]>"},
     {"To close a room for maintenance, enter",
      {"<maintenance: [room number] [From date] [To date] [Note]>"},
     {"To see activity of all rooms, enter ", {"<rooms:>"},
     {"To see soonest date a room is free for some nights, enter ",
      {"<plan: [Room number] [Number of nights]>"}}
```

array of all available commands and instructions how to use them

7.12.1.2 COMMANDS

```
const size_t COMMANDS = 8
```

number of different commands to use for a [Hotel](#)

7.12.1.3 daysFromBeginning

```
const unsigned daysFromBeginning[] = {0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334}
```

array keeping days past from January 1st

7.12.1.4 DISPLAY

```
const size_t DISPLAY = 5
```

max number of rooms to be printed

7.12.1.5 DISPLAY_WIDTH

```
const size_t DISPLAY_WIDTH = 130
```

width of the console used for centering the text

7.12.1.6 INIT_CAPACITY

```
const size_t INIT_CAPACITY = 2
```

const variable keeping minimal size of the reservations list

7.12.1.7 STRING_MAX_LENGTH

```
const size_t STRING_MAX_LENGTH = 128
```

max length of strings (char arrays)

7.13 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Constants.hpp

[Go to the documentation of this file.](#)

```

1 #ifndef __CONSTANTS_HPP
2 #define __CONSTANTS_HPP
3 #include <cstdint>
4
9 const unsigned daysFromBeginning[] = {0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334};
10
15 const size_t DISPLAY_WIDTH = 130;
16
21 const size_t COMMANDS = 8;
22
27 const size_t STRING_MAX_LENGTH = 128;
28
33 const size_t INIT_CAPACITY = 2;
34
39 const size_t DISPLAY = 5;
40
45 const char cmdArr[COMMANDS][2][STRING_MAX_LENGTH] = {
46     {"To make a reservation, enter "},
47     {"<reserve: [Room number] [Accommodation date] [Departure date] {Guest name[;]} {Note}>"},
48     {"To see list of free rooms for a particular date, enter "},
49     {"<available: [date]>"},
50     {"To free a room now, enter "},
51     {"<free: [Room number]>"},
52     {"To get report about the reservations of a room over a period of time, enter "},
53     {"<report: [From date] [To date]>"},
54     {"To request a room for guests, enter "},
55     {"<request: [minimal number of beds] [Accommodation date] [Departure date]>"},
56     {"To close a room for maintenance, enter "},
57     {"<maintenance: [room number] [From date] [To date] [Note]>"},
58     {"To see activity of all rooms, enter "}, {"<rooms:>"},
59     {"To see soonest date a room is free for some nights, enter "},
60     {"<plan: [Room number] [Number of nights]>"};
61
62 #endif

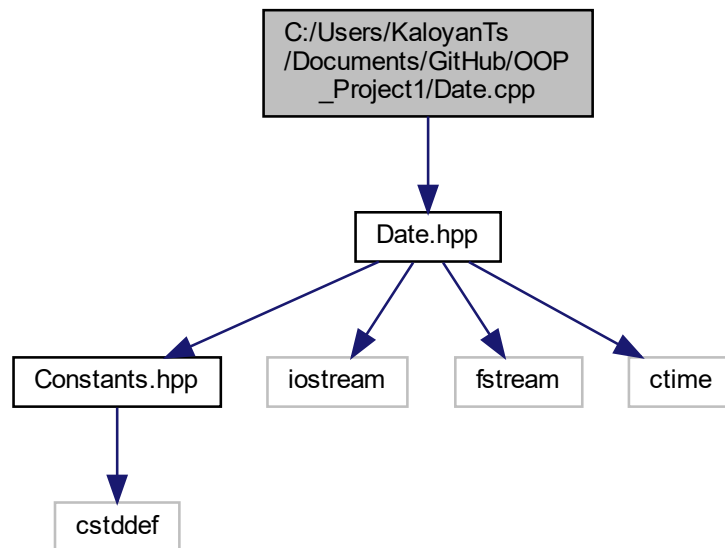
```

7.14 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Date.cpp

File Reference

```
#include "Date.hpp"
```

Include dependency graph for Date.cpp:



Functions

- `std::istream & operator>> (std::istream &is, Date &d)`
- `std::ostream & operator<< (std::ostream &os, const Date &d)`
- `std::istream & operator>> (std::istream &is, DatePeriod &dP)`
overloaded operator>> for [DatePeriod](#) input

7.14.1 Function Documentation

7.14.1.1 operator<<()

```
std::ostream & operator<< (
    std::ostream & os,
    const Date & d )
```

Parameters

<code>os</code>	output stream
<code>d</code>	Date to be output

Returns

std::ostream& reference to the output stream

7.14.1.2 operator>>() [1/2]

```
std::istream & operator>> (
    std::istream & is,
    Date & d )
```

Parameters

<i>is</i>	input stream
<i>d</i>	Date to be input

Returns

std::istream& reference to the input stream

7.14.1.3 operator>>() [2/2]

```
std::istream & operator>> (
    std::istream & is,
    DatePeriod & dP )
```

overloaded operator>> for DatePeriod input

Parameters

<i>is</i>	input stream
<i>dP</i>	DatePeriod to be input

Returns

std::istream& this input stream

7.15 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Date.hpp File Reference

```
#include "Constants.hpp"
#include <iostream>
#include <fstream>
```


Functions

- `std::istream & operator>> (std::istream &is, DatePeriod &dP)`
overloaded `operator>>` for [DatePeriod](#) input

7.15.1 Function Documentation

7.15.1.1 `operator>>()`

```
std::istream & operator>> (
    std::istream & is,
    DatePeriod & dP )
```

overloaded `operator>>` for [DatePeriod](#) input

Parameters

<i>is</i>	input stream
<i>dP</i>	DatePeriod to be input

Returns

`std::istream&` this input stream

7.16 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Date.hpp

[Go to the documentation of this file.](#)

```
1 #ifndef __DATE_HPP
2 #define __DATE_HPP
3 #include "Constants.hpp"
4 #include <iostream>
5 #include <fstream>
6 #include <ctime>
7
12 class Date
13 {
14     unsigned short day, month, year;
25     bool isValid() const;
33     bool isLeap(unsigned y) const;
34
35 public:
43     Date(unsigned short d = 1, unsigned short m = 1, unsigned short y = 1900) : day(d), month(m), year(y)
    {}
51     bool operator<(const Date other) const;
55     bool operator<=(const Date other) const;
63     bool operator>(const Date other) const;
67     bool operator>=(const Date other) const;
75     bool operator==(const Date other) const;
82     const char *operator() (char *buf) const;
89     int operator-(Date other) const;
95     Date &operator++();
96
102     static Date getToday();
103
111     friend std::istream &operator>>(std::istream &is, Date &d);
119     friend std::ostream &operator<<(std::ostream &os, const Date &d);
120
126     void writeToBinaryFile(std::ofstream &ofs);
```

```

127
133     void readDataFromBinary(std::ifstream &ifs);
134 };
135
136 struct DatePeriod
137 {
138     Date from;
139
140     Date to;
141
142     unsigned length() const { return to - from; }
143
144     DatePeriod &operator++();
145
146     void readProper();
147 };
148
149 std::istream &operator>>(std::istream &is, DatePeriod &dP);
150
151 #endif

```

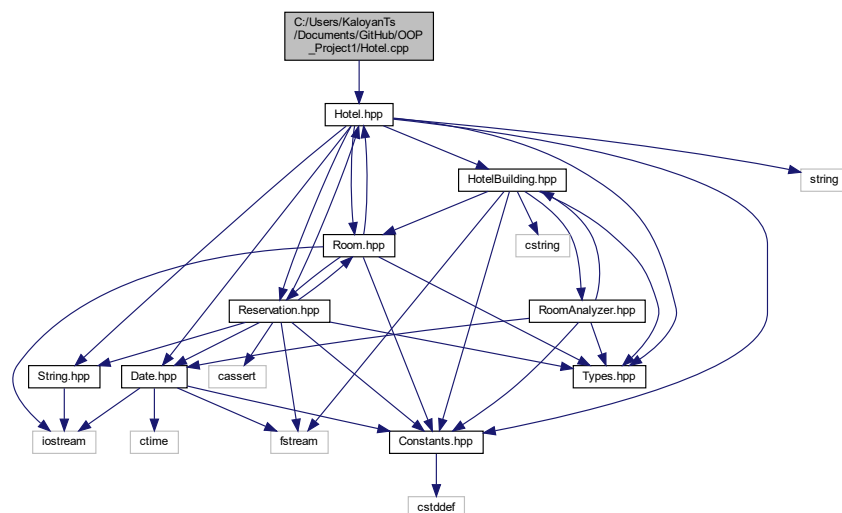
7.17 markdowns/bibliography.md File Reference

7.18 markdowns/commands.md File Reference

7.19 markdowns/mainpage.md File Reference

7.20 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Hotel.cpp File Reference

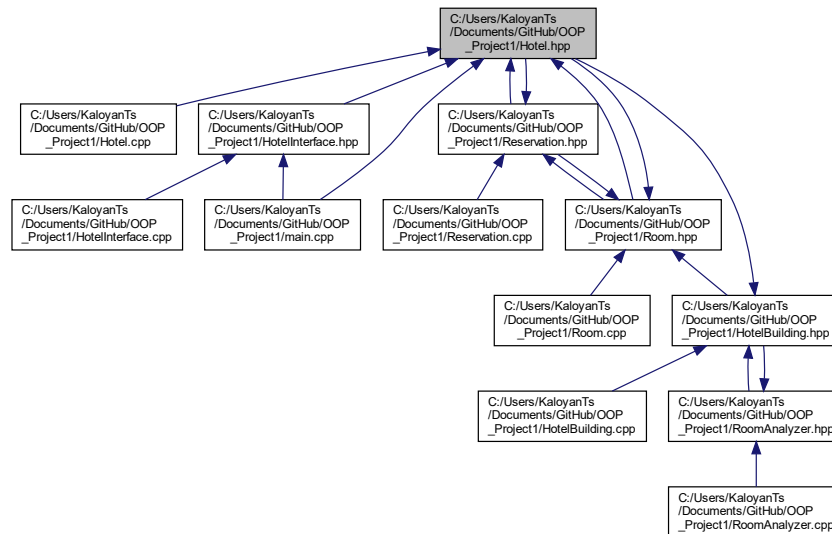
#include "Hotel.hpp"
 Include dependency graph for Hotel.cpp:



Functions

- `String readFromIstream` (std::ifstream &ifs, size_t len)
read string from input file stream with certain length

This graph shows which files directly or indirectly include this file:



Classes

- class [Hotel](#)

Class representing hotel with name, current [Date](#) and a building (list of rooms)

Functions

- [String readFromIfstream](#) (std::ifstream &ifs, size_t len)

read string from input file stream with certain length

7.21.1 Function Documentation

7.21.1.1 readFromIfstream()

```
String readFromIfstream (
    std::ifstream & ifs,
    size_t len )
```

read string from input file stream with certain length

Parameters

<i>ifs</i>	input stream
<i>len</i>	maximum length of the read string

Returns

String

7.22 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Hotel.hpp

[Go to the documentation of this file.](#)

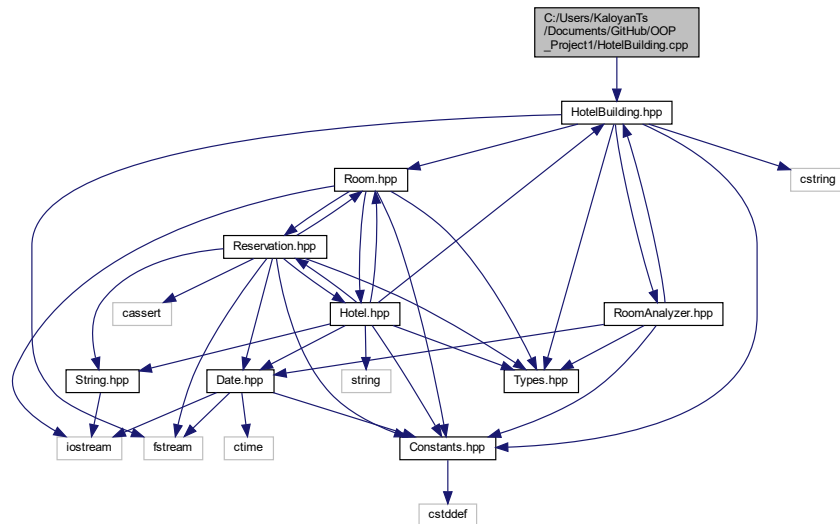
```

1  #ifndef __HOTEL_HPP
2  #define __HOTEL_HPP
3  #include "Types.hpp"
4  #include "Constants.hpp"
5  #include "String.hpp"
6  #include "Date.hpp"
7  #include "Room.hpp"
8  #include "Reservation.hpp"
9  #include "HotelBuilding.hpp"
10 #include <string>
11
19 String readFromIfstream(std::ifstream &ifstream, size_t len);
20
25 class Hotel
26 {
31     String name;
32
37     static Date now;
38
43     HotelBuilding *building;
44
49     void writeToBinaryFile();
50
51 public:
52     Hotel() = delete;
53
59     Hotel(String hotelDataFile);
60     Hotel(const Hotel &) = delete;
61     Hotel &operator=(const Hotel &) = delete;
62
67     ~Hotel();
68
74     static Date today() { return now; }
75
81     String getName() const { return name; }
82
87     void nextDay();
88
100    bool reserveRoom(unsigned number, const DatePeriod &period, String name = "-", String note =
        "None\n");
101
107    Hotel &showAvailableRooms(std::ostream &, Date);
108
116    bool freeRoom(unsigned number);
117
124    Hotel &getReport(DatePeriod &period);
125
132    void searchRoom(unsigned minBeds, const DatePeriod &period) const;
133
143    bool serviceRoom(unsigned number, const DatePeriod &period, String note);
144
150    Hotel &showToday();
151
159    Hotel &seeRoomForNights(unsigned number, unsigned nights);
160
167    bool workDay();
168 };
169
170 #endif

```

7.23 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Hotel Building.cpp File Reference

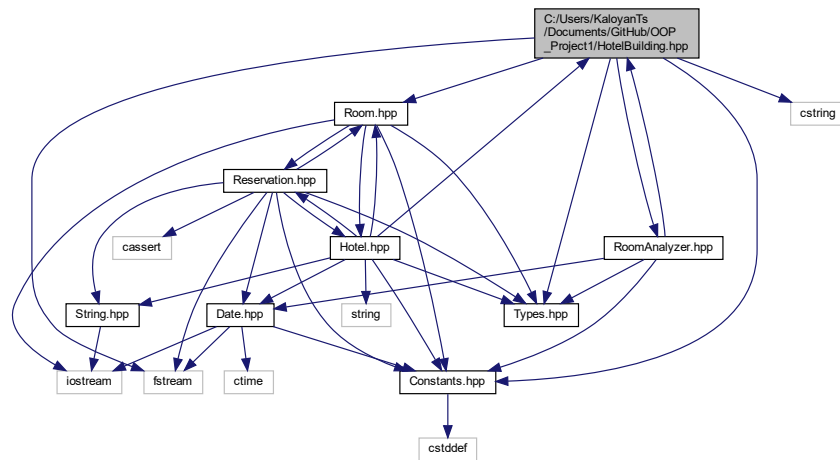
```
#include "HotelBuilding.hpp"
Include dependency graph for HotelBuilding.cpp:
```



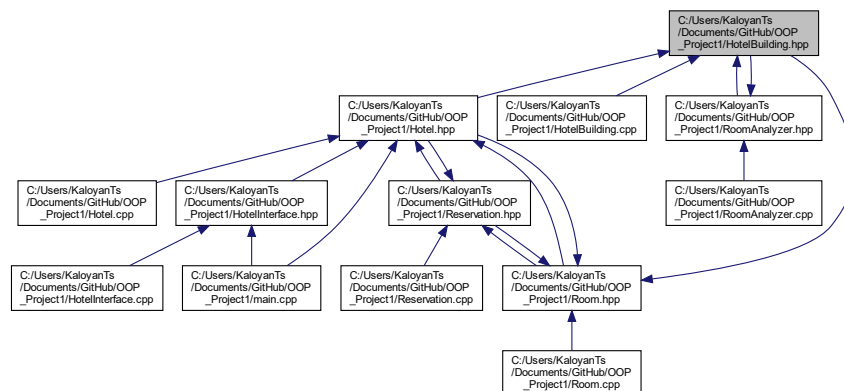
7.24 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Hotel Building.hpp File Reference

```
#include "Types.hpp"
#include "Constants.hpp"
#include "Room.hpp"
#include "RoomAnalyzer.hpp"
#include <fstream>
#include <cstring>
```

Include dependency graph for HotelBuilding.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [HotelBuilding](#)
Class representing list of rooms.

7.25 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/HotelBuilding.hpp

[Go to the documentation of this file.](#)

```

1 #ifndef __HOTELBUILDING_HPP
2 #define __HOTELBUILDING_HPP
3 #include "Types.hpp"
4 #include "Constants.hpp"
5 #include "Room.hpp"
  
```

```

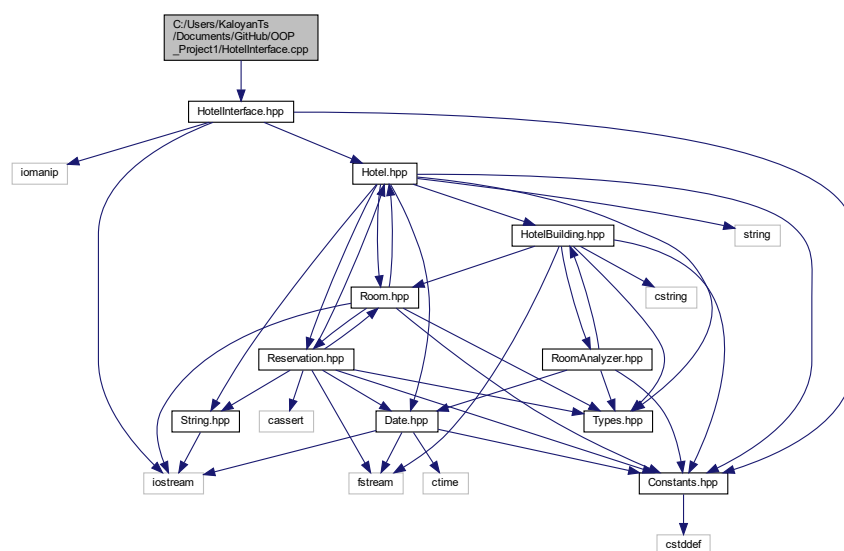
6 #include "RoomAnalyzer.hpp"
7 #include <fstream>
8 #include <cstring>
9
14 class HotelBuilding
15 {
20     Room **rooms;
25     size_t size;
26
27 public:
28     HotelBuilding(std::ifstream &if);
29     HotelBuilding(const HotelBuilding &other) = delete;
30     HotelBuilding &operator=(HotelBuilding &other) = delete;
31     ~HotelBuilding();
32
33     size_t getRoomCount() const { return size; }
34
35     Room *operator[](unsigned roomNumber) const;
36
37     void newDate(Date d);
38
39     void showAvailableRooms(std::ostream &os, Date d) const;
40
41     void createReport(DatePeriod &period) const;
42
43     void suggestRoom(unsigned beds, const DatePeriod &period);
44
45     void showRoomsStatesToday(Date today) const;
46
47     void showRoomForNights(unsigned number, unsigned nights, Date today) const;
48
49     void writeToBinaryFile(std::ofstream &ofs);
50
51     void readDataFromBinary(std::ifstream &if);
52
53     friend class RoomAnalyzer;
54 };
55 #endif

```

7.26 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/HotelInterface.cpp File Reference

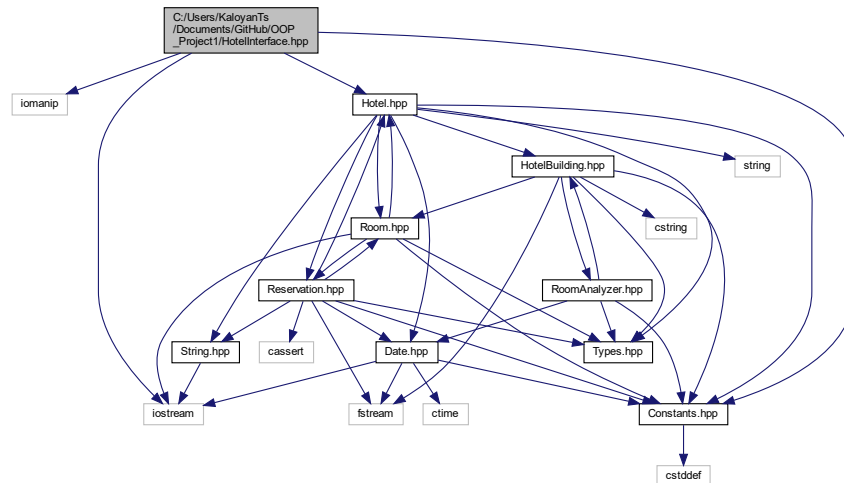
```
#include "HotelInterface.hpp"
```

Include dependency graph for HotelInterface.cpp:

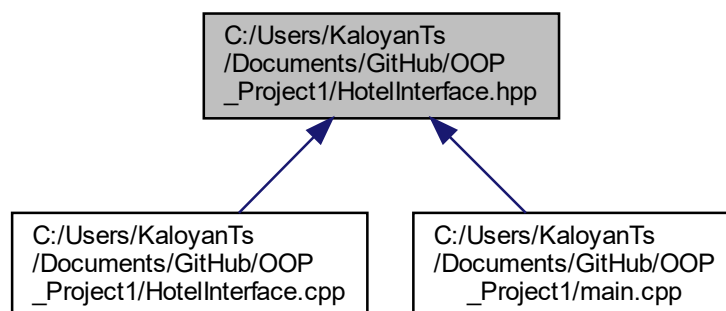


7.27 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/HotelInterface.hpp File Reference

```
#include <iomanip>
#include <iostream>
#include "Constants.hpp"
#include "Hotel.hpp"
Include dependency graph for HotelInterface.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [HotelInterface](#)

Utility Class taking care of the UI of the program.

7.28 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/HotelInterface.hpp

[Go to the documentation of this file.](#)

```

1 #ifndef __HOTELINTERFACE_HPP
2 #define __HOTELINTERFACE_HPP
3 #include <iomanip>
4 #include <iostream>
5 #include "Constants.hpp"
6 #include "Hotel.hpp"
7
9 class HotelInterface
10 {
11 public:
12     static void createHeader(Hotel &H);
13
14     static void beginDay();
15 };
16 #endif

```

7.29 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/main.cpp

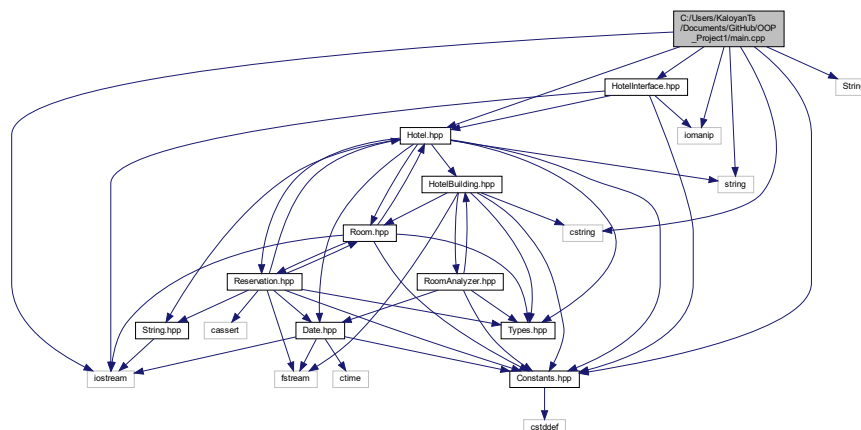
File Reference

```

#include <iostream>
#include <iomanip>
#include <cstring>
#include <string>
#include "Constants.hpp"
#include "String"
#include "HotelInterface.hpp"
#include "Hotel.hpp"

```

Include dependency graph for main.cpp:



Functions

- int [main](#) ()

7.29.1 Function Documentation

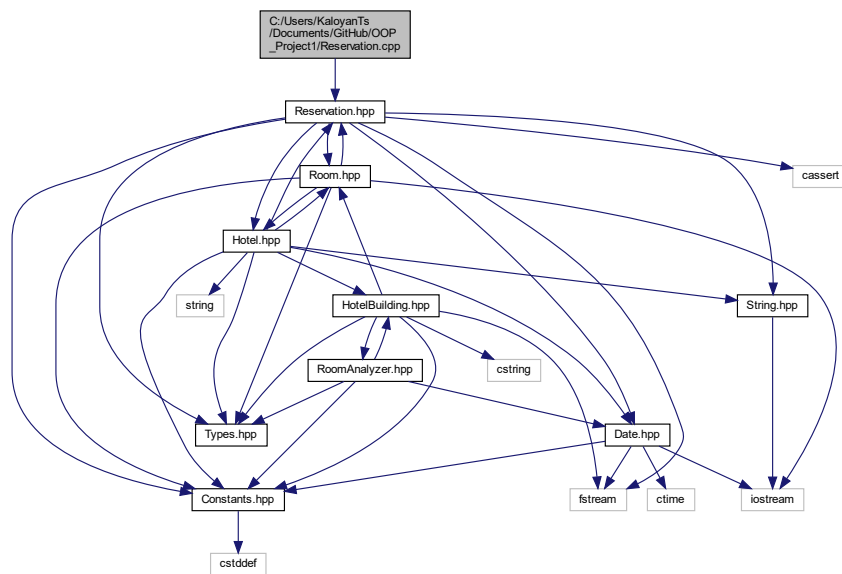
7.29.1.1 main()

```
int main ( )
```

7.30 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/↵ Reservation.cpp File Reference

```
#include "Reservation.hpp"
```

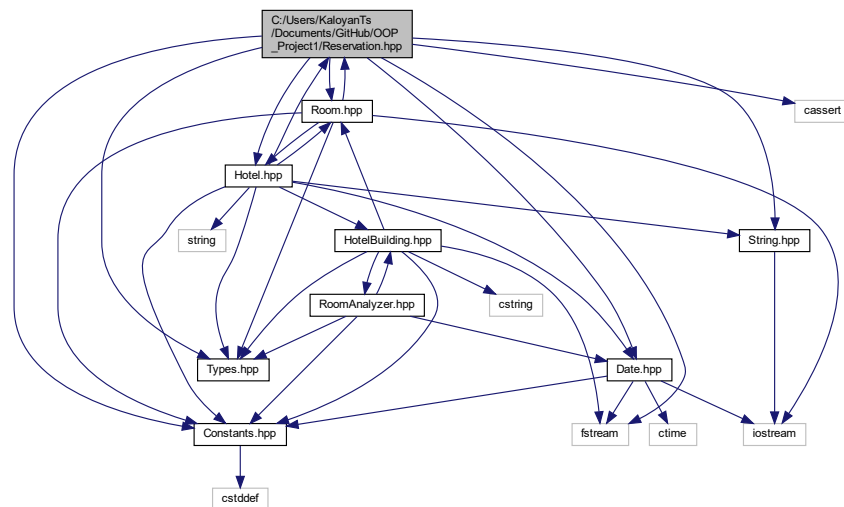
Include dependency graph for Reservation.cpp:



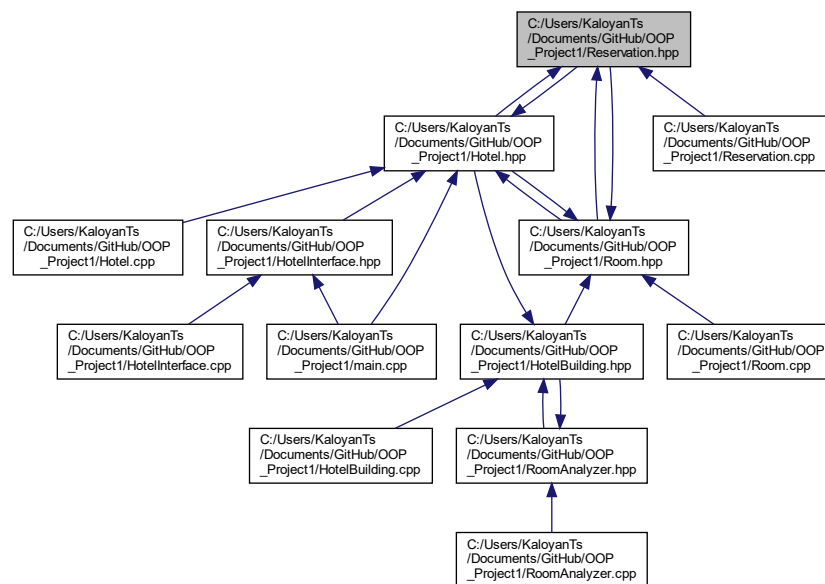
7.31 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/↵ Reservation.hpp File Reference

```
#include "Types.hpp"
#include "Constants.hpp"
#include "String.hpp"
#include "Room.hpp"
#include "Date.hpp"
#include "Hotel.hpp"
#include <cassert>
#include <fstream>
```

Include dependency graph for Reservation.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [Reservation](#)
Class representing information about a reservation.

Enumerations

- enum [ReservationState](#) { [UNKNOWN](#) = 0 , [PAST](#) , [ACTIVE](#) , [FUTURE](#) }
different states in time for a reservation

Functions

- `std::ostream & operator<< (std::ostream &, const Reservation &)`
overloaded operator for ouputting a reservation

7.31.1 Enumeration Type Documentation

7.31.1.1 ReservationState

enum `ReservationState`

different states in time for a reservation

Enumerator

UNKNOWN	
PAST	
ACTIVE	
FUTURE	

7.31.2 Function Documentation

7.31.2.1 operator<<()

```
std::ostream & operator<< (
    std::ostream & ,
    const Reservation & )
```

overloaded operator for ouputting a reservation

Returns

`std::ostream&` ouput stream

7.32 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/↵ Reservation.hpp

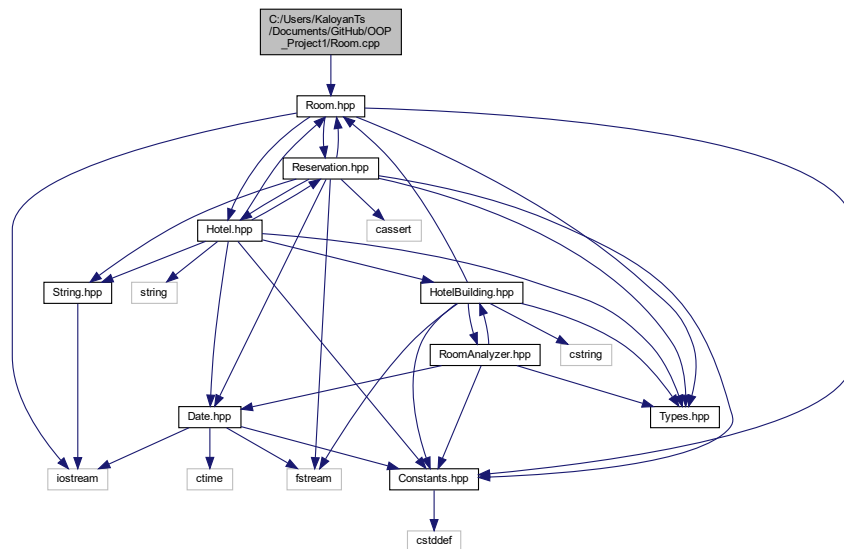
[Go to the documentation of this file.](#)

```
1 #ifndef __RESERVATION_HPP
2 #define __RESERVATION_HPP
3 #include "Types.hpp"
4 #include "Constants.hpp"
5 #include "String.hpp"
```

```
6 #include "Room.hpp"
7 #include "Date.hpp"
8 #include "Hotel.hpp"
9 #include <cassert>
10 #include <fstream>
11
12 enum ReservationState
13 {
14     UNKNOWN = 0,
15     PAST,
16     ACTIVE,
17     FUTURE
18 };
19
20 class Reservation
21 {
22     String guestName;
23     String note;
24     DatePeriod period;
25     ReservationState state;
26     bool service;
27
28 public:
29     Reservation(String name, const DatePeriod &p, String n = "None.\n", bool s = false);
30     Reservation(const Reservation &) = delete;
31     Reservation &operator=(const Reservation &) = delete;
32
33     bool isActive() const { return state == ACTIVE; }
34
35     bool isPast() const { return state == PAST; }
36
37     bool isServiced() const { return service; }
38
39     Date getFrom() const { return period.from; }
40
41     Date getTo() const { return period.to; }
42
43     unsigned getNights() const { return period.length(); }
44
45     String getNote() const { return note; }
46
47     void onDate(Date d);
48
49     ReservationState stateOnDate(Date) const;
50
51     bool LeavingInAdvance(Date);
52
53     void writeToBinaryFile(std::ofstream &ofs);
54
55     void readDataFromBinary(std::ifstream &if);
56 };
57
58 std::ostream &operator<<(std::ostream &, const Reservation &);
59
60 #endif
```

7.33 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Room.cpp File Reference

```
#include "Room.hpp"
Include dependency graph for Room.cpp:
```



Functions

- `std::ostream & operator<< (std::ostream &os, const Room &R)`
prints `Room`'s base info (number and bedCount)

7.33.1 Function Documentation

7.33.1.1 operator<<()

```
std::ostream & operator<< (
    std::ostream & os,
    const Room & R )
```

prints `Room`'s base info (number and bedCount)

Parameters

<code>os</code>	output stream
<code>R</code>	<code>Room</code> to be printed

Classes

- class [Room](#)

Class representing a room in hotel.

Functions

- `std::ostream & operator<< (std::ostream &os, const Room &R)`

prints [Room](#)'s base info (number and bedCount)

7.34.1 Function Documentation

7.34.1.1 operator<<()

```
std::ostream & operator<< (
    std::ostream & os,
    const Room & R )
```

prints [Room](#)'s base info (number and bedCount)

Parameters

<i>os</i>	output stream
<i>R</i>	Room to be printed

Returns

`std::ostream&` same output stream

7.35 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Room.hpp

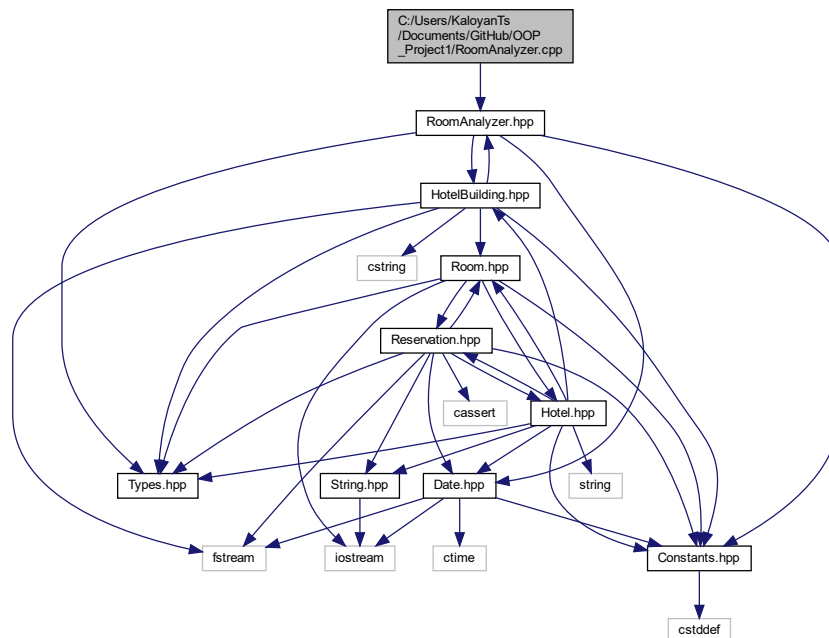
[Go to the documentation of this file.](#)

```
1 #ifndef __ROOM_HPP
2 #define __ROOM_HPP
3 #include <iostream>
4 #include "Types.hpp"
5 #include "Reservation.hpp"
6 #include "Hotel.hpp"
7 #include "Constants.hpp"
8
13 class Room
14 {
19     unsigned number;
24     unsigned bedCount;
29     Reservation **reservations;
34     size_t resCount, resCapacity;
35
40     Reservation **pastReservations;
45     size_t pastCount, pastCapacity;
46
51     void expand(Reservation **&arr, size_t &size, size_t &capacity);
56     void shrink(Reservation **&arr, size_t &size, size_t &capacity);
57
```

```
64     unsigned daysTakenInPeriod(const DatePeriod &period) const;
65
76     bool newReservation(String name, String note, const DatePeriod &period, bool service);
77
82     void moveToPast();
83
84 public:
91     Room(unsigned n, unsigned bC);
96     Room(const Room &) = delete;
101    Room &operator=(const Room &) = delete;
106    ~Room();
107
108    unsigned getNumber() const { return number; }
109    unsigned getBedCount() const { return bedCount; }
110    bool isFreeNow() const;
111
118    bool freeRoom(Reservation *&currentRes);
119
124    void newDate(Date);
125
132    bool isFreeOnDate(Date) const;
133
141    bool isFreeInPeriod(const DatePeriod &period) const;
142
151    bool showReservationsInPeriod(std::ostream &os, const DatePeriod &period) const;
152
162    bool addReservation(String name, String note, const DatePeriod &period);
163
172    bool closeForService(String note, const DatePeriod &period);
173
178    void showActivity() const;
179
185    void writeToBinaryFile(std::ofstream &ofs);
186
192    void readDataFromBinary(std::ifstream &ifb);
193 };
194
202 std::ostream &operator<<(std::ostream &os, const Room &R);
203
204 #endif
```


7.36 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/RoomAnalyzer.cpp File Reference

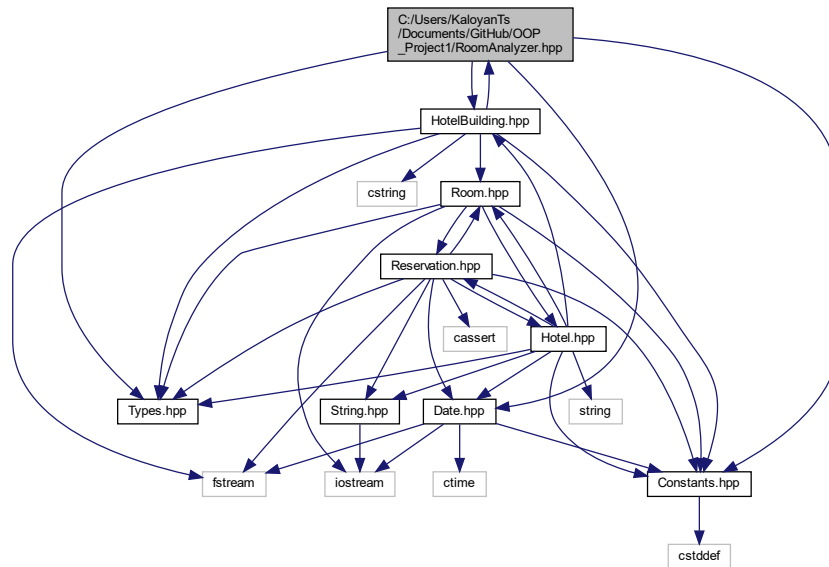
```
#include "RoomAnalyzer.hpp"
Include dependency graph for RoomAnalyzer.cpp:
```



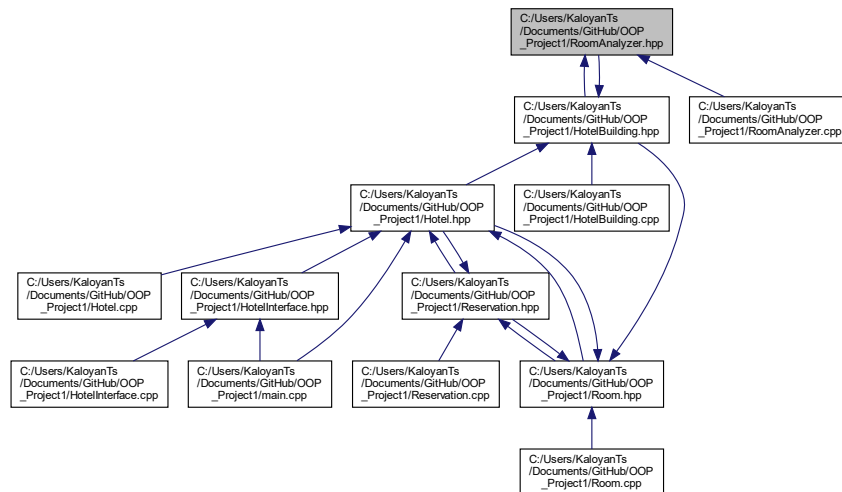
7.37 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/RoomAnalyzer.hpp File Reference

```
#include "Types.hpp"
#include "Constants.hpp"
#include "HotelBuilding.hpp"
#include "Date.hpp"
```

Include dependency graph for RoomAnalyzer.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class [RoomAnalyzer](#)
Utility class to perform algorithms on the rooms in a building.

7.38 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/RoomAnalyzer.hpp↵

[Go to the documentation of this file.](#)

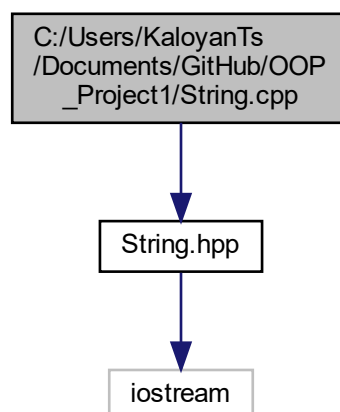
```

1 #ifndef __ROOMANALYZER_HPP
2 #define __ROOMANALYZER_HPP
3 #include "Types.hpp"
4 #include "Constants.hpp"
5 #include "HotelBuilding.hpp"
6 #include "Date.hpp"
7
8
9 class RoomAnalyzer
10 {
11     static void sortRoomsByScore(HotelBuilding &hB, unsigned *score, size_t from, size_t to);
12
13     static void sortRoomsByNumber(HotelBuilding &hB, size_t from, size_t size);
14
15     template <typename T>
16     static void swap(T &a, T &b);
17
18 public:
19     static void suggest(HotelBuilding &hB, unsigned beds, DatePeriod period);
20
21     static void soonestFreePeriod(const HotelBuilding &hB, unsigned number, unsigned nights, Date today);
22 };
23
24 template <typename T>
25 void RoomAnalyzer::swap(T &a, T &b)
26 {
27     T c = a;
28     a = b;
29     b = c;
30 }
31
32 #endif

```

7.39 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/String.cpp File Reference

#include "String.hpp"
 Include dependency graph for String.cpp:



Functions

- `std::ostream & operator<< (std::ostream &os, const String &S)`
print this [String](#) to the given output stream

7.39.1 Function Documentation

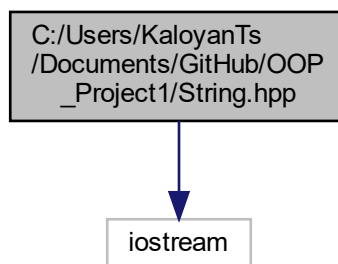
7.39.1.1 `operator<<()`

```
std::ostream & operator<< (  
    std::ostream & os,  
    const String & S )
```

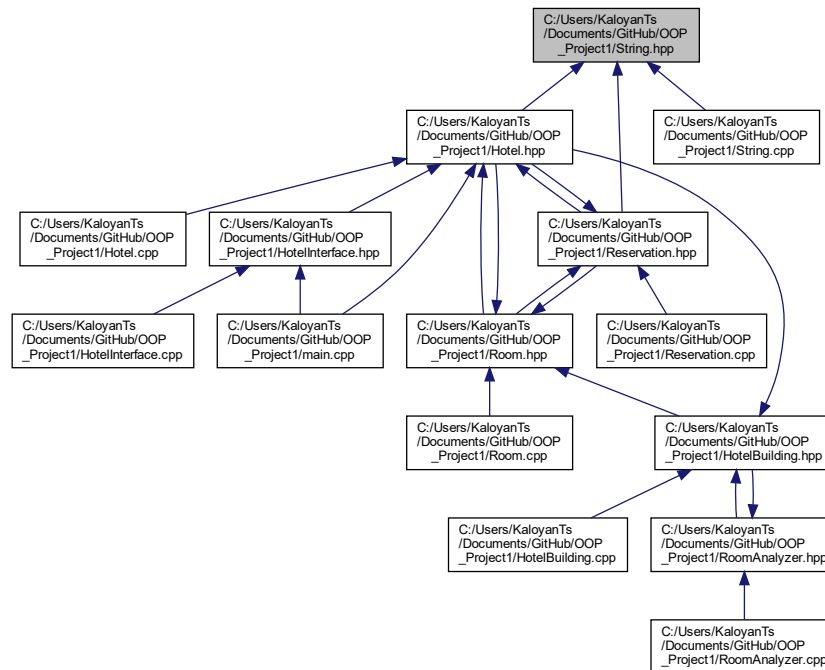
print this [String](#) to the given output stream

7.40 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/String.hpp File Reference

```
#include <iostream>  
Include dependency graph for String.hpp:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [String](#)
Class representing dynamic string (array of characters)

Functions

- `std::ostream & operator<< (std::ostream &os, const String &S)`
print this [String](#) to the given output stream

7.40.1 Function Documentation

7.40.1.1 operator<<()

```
std::ostream & operator<< (
    std::ostream & os,
    const String & S )
```

print this [String](#) to the given output stream

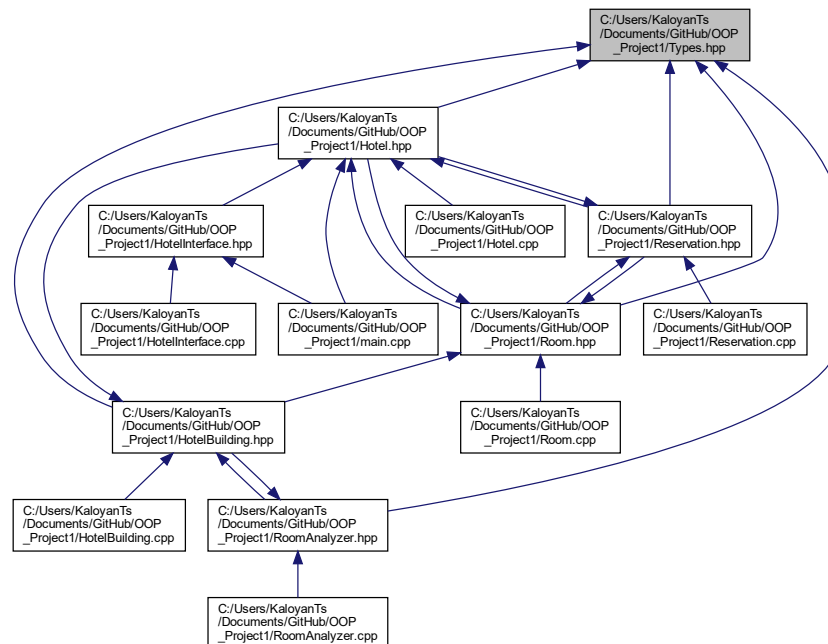
7.41 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/String.hpp

[Go to the documentation of this file.](#)

```
1 #ifndef __STRING_HPP
2 #define __STRING_HPP
3 #include <iostream>
4
5
6 class String
7 {
8     char *str;
9
10     size_t m_size, m_capacity;
11
12     void copy(const String &other);
13
14     void expand(size_t newCap);
15
16 public:
17     String(const char *_str = "");
18     String(const String &other);
19
20     String &operator=(const String &other);
21
22     ~String();
23
24     size_t size() const { return m_size; }
25
26     size_t capacity() const { return m_capacity; }
27
28     char *c_str() const { return str; }
29
30     String operator+(const String &other) const;
31
32     String &operator+=(const String &other);
33
34     operator const char *() const { return c_str(); }
35
36     String &shrink_to_fit();
37 };
38
39 std::ostream &operator<<(std::ostream &os, const String &S);
40
41 #endif
```

7.42 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Types.hpp File Reference

This graph shows which files directly or indirectly include this file:



7.43 C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Types.hpp

[Go to the documentation of this file.](#)

```

1 #ifndef __TYPES_HPP
2 #define __TYPES_HPP
3
4 class String;
5 class Date;
6 class Room;
7 class HotelBuilding;
8 class Reservation;
9 class RoomAnalyzer;
10 class Hotel;
11
12 #endif

```


Index

`__has_include`
 `CMakeCCompilerId.c`, [47](#)
 `CMakeCXXCompilerId.cpp`, [51](#)
~Hotel
 Hotel, [20](#)
~HotelBuilding
 HotelBuilding, [26](#)
~Room
 Room, [37](#)
~String
 String, [43](#)

ACTIVE
 Reservation.hpp, [73](#)
addReservation
 Room, [37](#)
ARCHITECTURE_ID
 `CMakeCCompilerId.c`, [48](#)
 `CMakeCXXCompilerId.cpp`, [51](#)

beginDay
 HotelInterface, [30](#)

C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/3.23.0-rc2/CompilerIdC/CMakeCCompilerId.c, [47](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/3.23.0-rc2/CompilerIdCXX/CMakeCXXCompilerId.cpp, [50](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/Date.cpp.obj.d, [54](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/Hotel.cpp.obj.d, [54](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/HotelBuilding.cpp.obj.d, [54](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/HotelInterface.cpp.obj.d, [54](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/main.cpp.obj.d, [54](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/Reservation.cpp.obj.d, [54](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/Room.cpp.obj.d, [54](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/RoomAnalyzer.cpp.obj.d, [54](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/build/CMakeFiles/hotel.dir/String.cpp.obj.d, [54](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Constants.hpp, [54](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Date.hpp, [58](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Date.hpp, [59](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Hotel.cpp, [62](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Hotel.hpp, [63](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/HotelBuilding.cpp, [66](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/HotelBuilding.hpp, [66](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/HotelInterface.cpp, [68](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/HotelInterface.hpp, [69](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/main.cpp, [70](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Reservation.cpp, [71](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Reservation.hpp, [71](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Room.cpp, [75](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Room.hpp, [76](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/RoomAnalyzer.cpp, [79](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/RoomAnalyzer.hpp, [79](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/String.cpp, [81](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/String.hpp, [81](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Types.hpp, [82](#)
C:/Users/KaloyanTs/Documents/GitHub/OOP_Project1/Types.hpp, [85](#)
C-STR
 String, [43](#)
C-VERSION
 `CMakeCCompilerId.c`, [48](#)
capacity
 String, [44](#)
closeForService
 Room, [37](#)
CMakeCCompilerId.c
 __has_include, [47](#)
 ARCHITECTURE_ID, [48](#)
 C-VERSION, [48](#)
 COMPILER_ID, [48](#)
 DEC, [48](#)
 HEX, [48](#)

- info_arch, 49
- info_compiler, 49
- info_language_extensions_default, 49
- info_language_standard_default, 50
- info_platform, 50
- main, 49
- PLATFORM_ID, 48
- STRINGIFY, 49
- STRINGIFY_HELPER, 49
- CMakeCXXCompilerId.cpp
 - __has_include, 51
 - ARCHITECTURE_ID, 51
 - COMPILER_ID, 51
 - CXX_STD, 51
 - DEC, 51
 - HEX, 51
 - info_arch, 52
 - info_compiler, 53
 - info_language_extensions_default, 53
 - info_language_standard_default, 53
 - info_platform, 53
 - main, 52
 - PLATFORM_ID, 52
 - STRINGIFY, 52
 - STRINGIFY_HELPER, 52
- cmdArr
 - Constants.hpp, 56
- COMMANDS
 - Constants.hpp, 56
- COMPILER_ID
 - CMakeCCompilerId.c, 48
 - CMakeCXXCompilerId.cpp, 51
- Constants.hpp
 - cmdArr, 56
 - COMMANDS, 56
 - daysFromBeginning, 56
 - DISPLAY, 56
 - DISPLAY_WIDTH, 57
 - INIT_CAPACITY, 57
 - STRING_MAX_LENGTH, 57
- createHeader
 - HotelInterface, 30
- createReport
 - HotelBuilding, 26
- CXX_STD
 - CMakeCXXCompilerId.cpp, 51
- Date, 11
 - Date, 12
 - getToday, 12
 - operator<, 13
 - operator<<, 15
 - operator<=, 14
 - operator>, 14
 - operator>>, 16
 - operator>=, 15
 - operator(), 12
 - operator++, 13
 - operator-, 13
 - operator==, 14
 - readDataFromBinary, 15
 - writeToBinaryFile, 15
- Date.cpp
 - operator<<, 58
 - operator>>, 59
- Date.hpp
 - operator>>, 61
- DatePeriod, 16
 - from, 18
 - length, 17
 - operator++, 17
 - readProper, 18
 - to, 18
- daysFromBeginning
 - Constants.hpp, 56
- DEC
 - CMakeCCompilerId.c, 48
 - CMakeCXXCompilerId.cpp, 51
- DISPLAY
 - Constants.hpp, 56
- DISPLAY_WIDTH
 - Constants.hpp, 57
- freeRoom
 - Hotel, 20
 - Room, 38
- from
 - DatePeriod, 18
- FUTURE
 - Reservation.hpp, 73
- getBedCount
 - Room, 38
- getFrom
 - Reservation, 32
- getName
 - Hotel, 21
- getNights
 - Reservation, 32
- getNote
 - Reservation, 32
- getNumber
 - Room, 38
- getReport
 - Hotel, 21
- getRoomCount
 - HotelBuilding, 26
- getTo
 - Reservation, 33
- getToday
 - Date, 12
- HEX
 - CMakeCCompilerId.c, 48
 - CMakeCXXCompilerId.cpp, 51
- Hotel, 18
 - ~Hotel, 20
 - freeRoom, 20

- getName, 21
- getReport, 21
- Hotel, 19, 20
- nextDay, 21
- operator=, 21
- reserveRoom, 22
- searchRoom, 22
- seeRoomForNights, 22
- serviceRoom, 23
- showAvailableRooms, 23
- showToday, 23
- today, 24
- workDay, 24
- Hotel.cpp
 - readFromIstream, 63
- Hotel.hpp
 - readFromIstream, 64
- HotelBuilding, 24
 - ~HotelBuilding, 26
 - createReport, 26
 - getRoomCount, 26
 - HotelBuilding, 25, 26
 - newDate, 27
 - operator=, 27
 - operator[], 27
 - readDataFromBinary, 27
 - RoomAnalyzer, 29
 - showAvailableRooms, 28
 - showRoomForNights, 28
 - showRoomsStatesToday, 28
 - suggestRoom, 29
 - writeToBinaryFile, 29
- HotelInterface, 30
 - beginDay, 30
 - createHeader, 30
- info_arch
 - CMakeCCompilerId.c, 49
 - CMakeCXXCompilerId.cpp, 52
- info_compiler
 - CMakeCCompilerId.c, 49
 - CMakeCXXCompilerId.cpp, 53
- info_language_extensions_default
 - CMakeCCompilerId.c, 49
 - CMakeCXXCompilerId.cpp, 53
- info_language_standard_default
 - CMakeCCompilerId.c, 50
 - CMakeCXXCompilerId.cpp, 53
- info_platform
 - CMakeCCompilerId.c, 50
 - CMakeCXXCompilerId.cpp, 53
- INIT_CAPACITY
 - Constants.hpp, 57
- isActive
 - Reservation, 33
- isFreeInPeriod
 - Room, 38
- isFreeNow
 - Room, 39
- isFreeOnDate
 - Room, 39
- isPast
 - Reservation, 33
- isServiced
 - Reservation, 33
- LeavingInAdvance
 - Reservation, 34
- length
 - DatePeriod, 17
- main
 - CMakeCCompilerId.c, 49
 - CMakeCXXCompilerId.cpp, 52
 - main.cpp, 70
- main.cpp
 - main, 70
- markdowns/bibliography.md, 62
- markdowns/commands.md, 62
- markdowns/mainpage.md, 62
- newDate
 - HotelBuilding, 27
 - Room, 39
- nextDay
 - Hotel, 21
- onDate
 - Reservation, 34
- operator const char *
 - String, 44
- operator<
 - Date, 13
- operator<<
 - Date, 15
 - Date.cpp, 58
 - Reservation.hpp, 73
 - Room.cpp, 75
 - Room.hpp, 77
 - String.cpp, 82
 - String.hpp, 83
- operator<=
 - Date, 14
- operator>
 - Date, 14
- operator>>
 - Date, 16
 - Date.cpp, 59
 - Date.hpp, 61
- operator>=
 - Date, 15
- operator()
 - Date, 12
- operator+
 - String, 44
- operator++
 - Date, 13
 - DatePeriod, 17

- operator+=
 - String, [44](#)
- operator-
 - Date, [13](#)
- operator=
 - Hotel, [21](#)
 - HotelBuilding, [27](#)
 - Reservation, [34](#)
 - Room, [39](#)
 - String, [44](#)
- operator==
 - Date, [14](#)
- operator[]
 - HotelBuilding, [27](#)
- PAST
 - Reservation.hpp, [73](#)
- PLATFORM_ID
 - CMakeCCompilerId.c, [48](#)
 - CMakeCXXCompilerId.cpp, [52](#)
- readDataFromBinary
 - Date, [15](#)
 - HotelBuilding, [27](#)
 - Reservation, [34](#)
 - Room, [39](#)
- readFromIstream
 - Hotel.cpp, [63](#)
 - Hotel.hpp, [64](#)
- readProper
 - DatePeriod, [18](#)
- Reservation, [31](#)
 - getFrom, [32](#)
 - getNights, [32](#)
 - getNote, [32](#)
 - getTo, [33](#)
 - isActive, [33](#)
 - isPast, [33](#)
 - isServiced, [33](#)
 - LeavingInAdvance, [34](#)
 - onDate, [34](#)
 - operator=, [34](#)
 - readDataFromBinary, [34](#)
 - Reservation, [31](#), [32](#)
 - stateOnDate, [35](#)
 - writeToBinaryFile, [35](#)
- Reservation.hpp
 - ACTIVE, [73](#)
 - FUTURE, [73](#)
 - operator<<, [73](#)
 - PAST, [73](#)
 - ReservationState, [73](#)
 - UNKNOWN, [73](#)
- ReservationState
 - Reservation.hpp, [73](#)
- reserveRoom
 - Hotel, [22](#)
- Room, [35](#)
 - ~Room, [37](#)
 - addReservation, [37](#)
 - closeForService, [37](#)
 - freeRoom, [38](#)
 - getBedCount, [38](#)
 - getNumber, [38](#)
 - isFreeInPeriod, [38](#)
 - isFreeNow, [39](#)
 - isFreeOnDate, [39](#)
 - newDate, [39](#)
 - operator=, [39](#)
 - readDataFromBinary, [39](#)
 - Room, [36](#), [37](#)
 - showActivity, [40](#)
 - showReservationsInPeriod, [40](#)
 - writeToBinaryFile, [40](#)
- Room.cpp
 - operator<<, [75](#)
- Room.hpp
 - operator<<, [77](#)
- RoomAnalyzer, [41](#)
 - HotelBuilding, [29](#)
 - soonestFreePeriod, [41](#)
 - suggest, [41](#)
- searchRoom
 - Hotel, [22](#)
- seeRoomForNights
 - Hotel, [22](#)
- serviceRoom
 - Hotel, [23](#)
- showActivity
 - Room, [40](#)
- showAvailableRooms
 - Hotel, [23](#)
 - HotelBuilding, [28](#)
- showReservationsInPeriod
 - Room, [40](#)
- showRoomForNights
 - HotelBuilding, [28](#)
- showRoomsStatesToday
 - HotelBuilding, [28](#)
- showToday
 - Hotel, [23](#)
- shrink_to_fit
 - String, [45](#)
- size
 - String, [45](#)
- soonestFreePeriod
 - RoomAnalyzer, [41](#)
- stateOnDate
 - Reservation, [35](#)
- String, [42](#)
 - ~String, [43](#)
 - c_str, [43](#)
 - capacity, [44](#)
 - operator const char *, [44](#)
 - operator+, [44](#)
 - operator+=, [44](#)
 - operator=, [44](#)

- shrink_to_fit, [45](#)
 - size, [45](#)
 - String, [43](#)
- String.cpp
 - operator<<, [82](#)
- String.hpp
 - operator<<, [83](#)
- STRING_MAX_LENGTH
 - Constants.hpp, [57](#)
- STRINGIFY
 - CMakeCCompilerId.c, [49](#)
 - CMakeCXXCompilerId.cpp, [52](#)
- STRINGIFY_HELPER
 - CMakeCCompilerId.c, [49](#)
 - CMakeCXXCompilerId.cpp, [52](#)
- suggest
 - RoomAnalyzer, [41](#)
- suggestRoom
 - HotelBuilding, [29](#)
- to
 - DatePeriod, [18](#)
- today
 - Hotel, [24](#)
- UNKNOWN
 - Reservation.hpp, [73](#)
- workDay
 - Hotel, [24](#)
- writeToBinaryFile
 - Date, [15](#)
 - HotelBuilding, [29](#)
 - Reservation, [35](#)
 - Room, [40](#)