My Project

Generated by Doxygen 1.9.3

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

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

Chapter 3

Class Documentation

3.1 Date Class Reference

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. Default Date is 1/1/1900.
- bool operator< (Date other) const

  checks if this Date is chronologically before other Date
- bool **operator<=** (Date other) const

  see operator<
- bool operator> (Date other) const

  checks if this Date is chronologically after other Date
- bool **operator>=** (Date other) const

  see operator>
- bool operator== (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 incremention operator for Date

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

3.1.1 Constructor & Destructor Documentation

3.1.1.1 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

d	day
m	month
y	year

3.1.2 Member Function Documentation

3.1.2.1 getToday()

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

Get the today date.

Returns

Date

3.1.2.2 operator()()

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

records this Date in buffer in format YYYY-MM-DD

Parameters

buf	buffer where Date is recorded

Returns

const char∗ pointer to beginning of buf

3.1.2.3 operator++()

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

overloaded prefix incremention operator for Date

Returns

Date& reference to this Date

3.1.2.4 operator-()

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

Parameters

other	Date

Returns

int difference between of this Date and other

3.1.2.5 operator<()

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

checks if this Date is chronologically before other Date

Parameters

other	compared Date

Returns

true this Date is chronologically before other

false this Date is not chronologically before other

3.1.2.6 operator==()

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

checks if two dates are identical

Parameters

other	compared Date

Returns

true the dates are identical

false the dates are not identical

3.1.2.7 operator>()

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

checks if this Date is chronologically after other Date

Parameters

other	compared Date

Returns

true this Date is chronologically after other

false this Date is not chronologically after other

3.1.3 Friends And Related Function Documentation

3.1.3.1 operator<<

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

overloaded operator for outputing Date

Parameters

os	output stream
d	Date to be output

Returns

std::ostream& reference to the output stream

3.1.3.2 operator>>

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

overloaded operator for inputing Date

Parameters

is	input stream
d	Date to be input

Returns

std::istream& reference to the input stream

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

- Date.hpp
- Date.cpp

3.2 DatePeriod Struct Reference

Public Member Functions

- unsigned **length** () const

Public Attributes

- Date **from**
- Date **to**

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

- Date.hpp

3.3 Hotel Class Reference

Public Member Functions

- Hotel (std::string hotelDataFile)

  Construct a new Hotel object.
- **Hotel** (const Hotel &)=delete
- Hotel & **operator=** (const Hotel &)=delete
- ∼**Hotel** ()

  Destroy the Hotel object.
- std::string getName () const

  get the name of this Hotel
- void **nextDay** ()

  advance to the nextDay
- bool reserveRoom (unsigned number, DatePeriod period, std::string name="-", std::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. Report written in file named "report-YYYY-←↩ MM-DD.txt" where YYYY-MM-DD is the beginning of the period.
- void **searchRoom** (unsigned minBeds, DatePeriod period) const
- bool serviceRoom (unsigned number, DatePeriod period, std::string note)

  plans maintenance for particular Room and period leaving note for the service

Static Public Member Functions

- static Date today ()

  get today's Date according to all Hotels

3.3.1 Constructor & Destructor Documentation

3.3.1.1 Hotel()

```
Hotel::Hotel (
            std::string hotelDataFile )
```

Construct a new Hotel object.

Parameters

hotelDataFile	path to file where rooms are recorded

3.3.2 Member Function Documentation

3.3.2.1 freeRoom()

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

tries to free Room with particular ID

Parameters

number	Room's ID

Returns

true room is now free

false room not found

3.3.2.2 getName()

```
std::string Hotel::getName ( ) const  [inline]
```

get the name of this Hotel

Returns

std::string

3.3.2.3 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

period	desired period of time

Returns

Hotel& this Hotel

3.3.2.4 reserveRoom()

```
bool Hotel::reserveRoom (
             unsigned number,
             DatePeriod period,
             std::string name = "-",
             std::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

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

Returns

true successfull reservation

false failed reservation (not made)

3.3.2.5 serviceRoom()

```
bool Hotel::serviceRoom (
             unsigned number,
             DatePeriod period,
             std::string note )
```

plans maintenance for particular Room and period leaving note for the service

Parameters

number	Room's ID
period	desired period of time
note	any notes to the service

Returns

true service planned successfully

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

3.3.2.6 showAvailableRooms()

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

output to stream all available rooms for a particular Date

Returns

Hotel& this Hotel

3.3.2.7 today()

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

get today's Date according to all Hotels

Returns

Date

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

- Hotel.hpp
- Hotel.cpp

3.4 HotelBuilding Class Reference

Public Member Functions

- **HotelBuilding** (std::ifstream &ifs)
- **HotelBuilding** (const HotelBuilding &other)=delete
- HotelBuilding & **operator=** (HotelBuilding &other)=delete
- size_t **getRoomCount** () const
- Room ∗ **operator[ ]** (unsigned roomNumber) const
- void **newDate** (Date d)
- void **showAvailableRooms** (std::ostream &os, Date d) const
- void **createReport** (DatePeriod period) const
- void **suggestRoom** (unsigned beds, DatePeriod period)

Friends

- class **RoomAnalyzer**

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

- HotelBuilding.hpp
- HotelBuilding.cpp

3.5 Reservation Class Reference

Public Member Functions

- Reservation (std::string name, DatePeriod p, std::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
- 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

3.5.1 Constructor & Destructor Documentation

3.5.1.1 Reservation()

```
Reservation::Reservation (
            std::string name,
            DatePeriod p,
            std::string n = "None.\n",
            bool s = false )
```

Construct a new Reservation object.

Parameters

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

3.5.2 Member Function Documentation

3.5.2.1 getFrom()

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

get beginning Date of this Reservation

Returns

Date beginning of the Reservation

3.5.2.2 getNights()

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

get count of nights of this Reservation

Returns

unsigned count of nights of this Reservation

3.5.2.3 getTo()

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

get end Date of this Reservation

Returns

Date end of the Reservation

3.5.2.4 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)

3.5.2.5 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)

3.5.2.6 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

3.5.2.7 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

3.5.2.8 onDate()

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

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

Parameters

d	new today's Date

3.5.2.9 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

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

- Reservation.hpp
- Reservation.cpp

3.6 Room Class Reference

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 ∗&currentRes)

  try to free this room
- void **changeLeaving** (Reservation ∗, Date newDate)
- 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** (DatePeriod period) const
- void **showReservationsInPeriod** (std::ostream &os, DatePeriod period) const
- bool **addReservation** (std::string name, std::string note, DatePeriod period)
- bool **closeForService** (std::string note, DatePeriod period)

3.6.1 Constructor & Destructor Documentation

3.6.1.1 Room()

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

Construct a new Room object.

Parameters

n	number of constructed Room
bC	number of beds in constructed Room

3.6.2 Member Function Documentation

3.6.2.1 freeRoom()

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

try to free this room

Returns

true sucesfully freed room

false room is already free

3.6.2.2 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

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

- Room.hpp
- Room.cpp

3.7 RoomAnalyzer Class Reference

Static Public Member Functions

- static void **suggest** (HotelBuilding &hB, unsigned beds, DatePeriod period)

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

- RoomAnalyzer.hpp
- RoomAnalyzer.cpp

Chapter 4

File Documentation

4.1 Date.hpp

```
#ifndef __DATE_HPP
#define __DATE_HPP
#include <iostream>
#include <ctime>

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

class Date
{
    unsigned short day, month, year;
    bool isVaid() const;
    bool isLeap(unsigned y) const;

public:
    Date(unsigned short d = 1, unsigned short m = 1, unsigned short y = 1900) : day(d), month(m), year(y)
    {}
    bool operator<(Date other) const;
    bool operator<=(Date other) const;
    bool operator>(Date other) const;
    bool operator>=(Date other) const;
    bool operator==(Date other) const;
    const char *operator()(char *buf) const;
    int operator-(Date other) const;
    Date &operator++();

     static Date getToday();

     friend std::istream &operator»(std::istream &is, Date &d);
     friend std::ostream &operator«(std::ostream &os, const Date &d);
};

struct DatePeriod
{
    Date from, to;
    unsigned length() const { return to - from; }
};

std::istream &operator»(std::istream &is, DatePeriod &dP);

#endif
```

4.2 Hotel.hpp

```
#ifndef __HOTEL_HPP
#define __HOTEL_HPP
#include "Types.hpp"
#include "Date.hpp"
#include "Room.hpp"
#include "Reservation.hpp"
#include "HotelBuilding.hpp"
#include <string>

std::string readFromIfstream(std::ifstream &ifs, size_t len);

```

```
class Hotel
{
    std::string name;

    static Date now;

    HotelBuilding *building;

public:
    Hotel() = delete;

    Hotel(std::string hotelDataFile);
    Hotel(const Hotel &) = delete;
    Hotel &operator=(const Hotel &) = delete;

    ~Hotel();

    static Date today() { return now; }

    std::string getName() const { return name; }

    void nextDay();

    bool reserveRoom(unsigned number, DatePeriod period, std::string name = "-", std::string note =
    "None\n");

    Hotel &showAvailableRooms(std::ostream &, Date);

     bool freeRoom(unsigned number);

     Hotel &getReport(DatePeriod period);

     void searchRoom(unsigned minBeds, DatePeriod period) const;

     bool serviceRoom(unsigned number, DatePeriod period, std::string note);
 };

 #endif
```

4.3 HotelBuilding.hpp

```
#ifndef __HOTELBUILDING_HPP
#define __HOTELBUILDING_HPP
#include "Types.hpp"
#include "Room.hpp"
#include "RoomAnalyzer.hpp"
#include <fstream>

class HotelBuilding
{
    Room **rooms;
    size_t size;

public:
    HotelBuilding(std::ifstream &ifs); // todo String
    HotelBuilding(const HotelBuilding &other) = delete;
    HotelBuilding &operator=(HotelBuilding &other) = delete;
    ~HotelBuilding();

    size_t getRoomCount() const { return size; }

    Room *operator[](unsigned roomNumber) const;

    void newDate(Date d);

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

    void createReport(DatePeriod period) const;

    void suggestRoom(unsigned beds, DatePeriod period);

    friend class RoomAnalyzer;
};

#endif
```

4.4 Reservation.hpp

```
#ifndef __RESERVATION_HPP
```

```
#define __RESERVATION_HPP
#include "Types.hpp"
#include "Room.hpp"
#include "Date.hpp"
#include "Hotel.hpp"
#include <cstring>
#include <cassert>
#include <fstream>
#include <string>

enum ReservationState
{
    UNKNOWN = 0,
    PAST,
    ACTIVE,
    FUTURE
};

class Reservation
{
    std::string guestName;
    std::string note;
    DatePeriod period;
    ReservationState state;
    bool service;

public:
    Reservation(std::string name, DatePeriod p, std::string n = "None.\n", bool s = false);
    Reservation(const Reservation &) = delete;
    Reservation &operator=(const Reservation &) = delete;

    bool isActive() const { return state == ACTIVE; }

    bool isPast() const { return state == PAST; }

    bool isServiced() const { return service; }

    Date getFrom() const { return period.from; }

    Date getTo() const { return period.to; }

    unsigned getNights() const { return period.length(); }

    void onDate(Date d);

    ReservationState stateOnDate(Date) const;

    bool LeavingInAdvance(Date);
};

std::ostream &operator«(std::ostream &, const Reservation &);

#endif
```

4.5 Room.hpp

```
#ifndef __ROOM_HPP
#define __ROOM_HPP
#include <iostream>
#include <string>
#include "Types.hpp"
#include "Reservation.hpp"
#include "Hotel.hpp"

const size_t INIT_CAPACITY = 2;

class Room
{
    unsigned number;
    unsigned bedCount;
    Reservation **reservations;
    size_t resCount, resCapacity;

    Reservation **pastReservations;
    size_t pastCount, pastCapacity;

    void expand(Reservation **&arr, size_t &size, size_t &capacity);
    void shrink(Reservation **&arr, size_t &size, size_t &capacity);

    unsigned daysTakenInPeriod(DatePeriod period) const;

    bool newReservation(std::string name, std::string note, DatePeriod period, bool service);

public:
    Room(unsigned n, unsigned bC);
    Room(const Room &) = delete;
    Room &operator=(const Room &) = delete;
    ~Room();

    unsigned getNumber() const { return number; }
    unsigned getBedCount() const { return bedCount; }
    bool isFreeNow() const;

    bool freeRoom(Reservation *&currentRes);
    void changeLeaving(Reservation *, Date newDate); // todo must be private

    void newDate(Date);

    bool isFreeOnDate(Date) const;

    bool isFreeInPeriod(DatePeriod period) const;

    void showReservationsInPeriod(std::ostream &os, DatePeriod period) const;

    bool addReservation(std::string name, std::string note, DatePeriod period);

    bool closeForService(std::string note, DatePeriod period);
};

std::ostream &operator«(std::ostream &os, const Room &R);

#endif
```

4.6 RoomAnalyzer.hpp

```
#ifndef __ROOMANALYZER_HPP
#define __ROOMANALYZER_HPP
#include "Types.hpp"
#include "HotelBuilding.hpp"
#include "Date.hpp"
const size_t DISPLAY = 5;

class RoomAnalyzer
{
    static void sortRooms(HotelBuilding &hB, unsigned *score, size_t size);

public:
    static void suggest(HotelBuilding &hB, unsigned beds, DatePeriod period);
};

#endif
```

4.7 Types.hpp

```
#ifndef __TYPES_HPP
#define __TYPES_HPP

class Date;
class Room;
class HotelBuilding;
class Reservation;
class RoomAnalyzer;
class Hotel;

#endif
```

Index