Daniel Schroeder

Project 2 writeup

Abstract

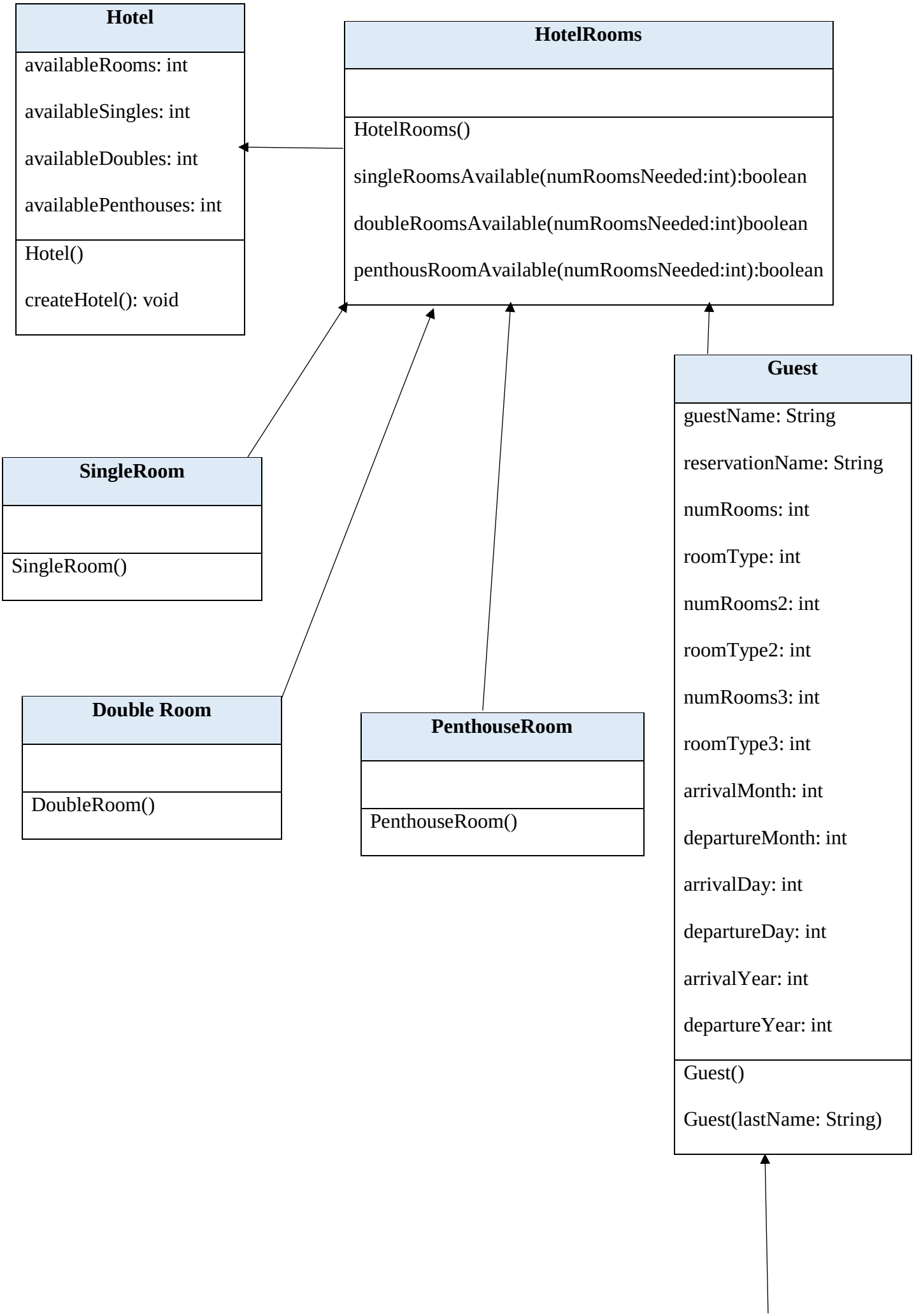
This paper goes over my final product. It goes into details about the motivation behind the project, a detailed description of the product, the requirements of the product, a literature survey, a user manual, and a conclusion.

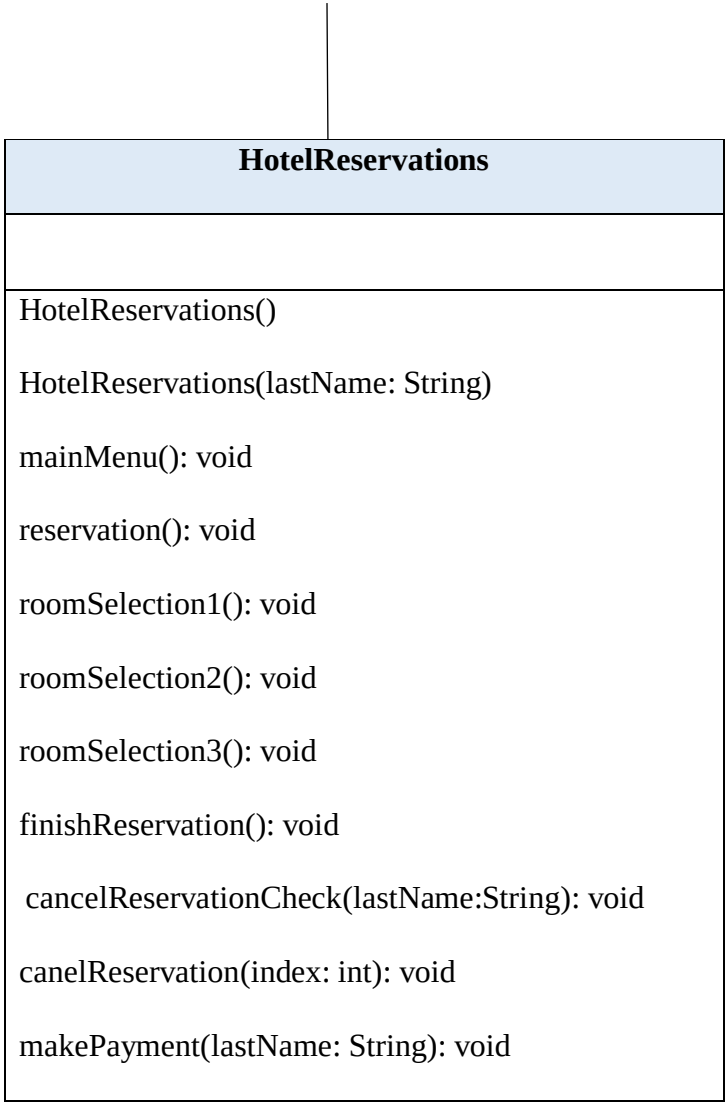
Introduction

This project was first thought of when I was at a hotel and wondered how hard it would be to make a guest management system. It turns out to be pretty difficult. However I continued to try to work out the issues in my code until I had something that worked. In the rest of this writeup I will go over how the product works and how to use it.

Detailed System Description

The classes I needed included a Hotel class, a Guest class, a class HotelReservations, a class HotelRooms, and a class for each of the three types of rooms. The class Hotel is a super class of both HotelReservations and HotelRooms, and HotelRooms is a super class of the three classes for each type of room. The project uses the command line to ask the user different questions about what type of reservation they would like to make. Some of what is asked of the user is the arrival and departure dates, the floor the guest would like to be on, the type of room preferred, and how many of that type of room the guest would like.


Hotel
availableRooms: int
availableSingles: int
availableDoubles: int
availablePenthouses: int
Hotel()
createHotel(): void
HotelRooms
HotelRooms()
singleRoomsAvailable(numRoomsNeeded:int):boolean
doubleRoomsAvailable(numRoomsNeeded:int)boolean
penthousRoomAvailable(numRoomsNeeded:int):boolean
Guest
guestName: String
reservationName: String
numRooms: int
roomType: int
numRooms2: int
roomType2: int
numRooms3: int
roomType3: int
arrivalMonth: int
departureMonth: int
arrivalDay: int
departureDay: int
arrivalYear: int
departureYear: int
Guest()
Guest(lastName: String)
SingleRoom
SingleRoom()
Double Room
DoubleRoom()
PenthouseRoom
PenthouseRoom()



Requirements

This system requires a database server to store the information of the guests as well as the availability of the hotel. The database server would not have to be big as this hotel is only 60 rooms big. For bigger hotels or chains of hotels, this database server would need to be able to hold more information.

Literature Survey

There are many other ways to store the guest information and manage reservations. Every hotel have this system implemented and many different hotels use a different system. My system is different because it does not require an employee to enter it into the system. It only requires the users information and everything is done automatically.

User Manual

To use this system is quite simple. Start by running the Hotel class. This is the driver class for the system. Then the user needs to follow the instructions on the screen. If a user inputs a value that the system doesn't understand, the program may crash or quit unexpectedly. Many of the inputs that the user must enter are ints. If the user enters and other data type the program will crash. If the user enters an int that is not listed, the program may have unexpected results such as quitting prematurely. It is also important that the user sticks to the form of entry shown by the program. Otherwise, the program could crash.

Conclusion

In conclusion, this paper went over the main features of the system, how to use the system, and the requirements of the system. It also acknowledged that there are other systems out there like this and discussed the differences between them. This project was significant to me because it taught me a lot about debugging code and how hard it can be. Up until this project I did not think

that coding took very long because I had always done it with relative ease. This project showed me that it is otherwise.