

预定类面向对象设计

文泰来 老师



扫描二维码关注微信/微博
获取最新面试题及权威解答

微信: [ninechapter](#)

知乎专栏: <http://zhuanlan.zhihu.com/jiuzhang>

微博: <http://www.weibo.com/ninechapter>

官网: www.jiuzhang.com

Restaurant

- Can you design a restaurant?



Clarify

- What
- How

- What

关键字: Restaurant

- What

关键字: Restaurant

In / Out: ?

- What

关键字: Restaurant

In / Out :

Party / Table



- What

关键字: Restaurant, Party, Table

What

关键字: Restaurant

属性: ?

What

关键字： Restaurant



What

关键字： Restaurant



- 吧台



- 大堂



- 包间

What

关键字：Restaurant



- 桌子的规格不一样，能坐的人数区别
- 吧台里，party的区分
- 收费？

- 针对本题

大堂，所有桌子都一样，暂不考虑人数限制

- How

规则?

- How



- 是否能够预约?

- How



- 是否能够送外卖?

Clarify

- How



- 每个Order需要区分是Dine-in还是Dine-out

- 针对本题

没有Reservation

没有Dine-out

思考模式1:

Party 进入餐馆 -> Host指引到空桌 (find table) ->
一个waiter负责这桌客人 (assign waiter) -> 客人点菜 (take order)
Chef 拿到order, 按顺序做菜 (cook by order) -> Order server
(serve order) -> 客人吃完后付钱 (check out)



思考模式2:

1. 客人进入餐馆，餐馆返回一个Table
2. 客人点菜，餐馆返回一桌菜
3. 客人付账，餐馆清空Table



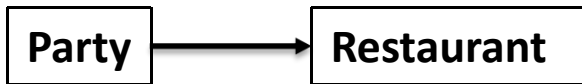
思考模式1:



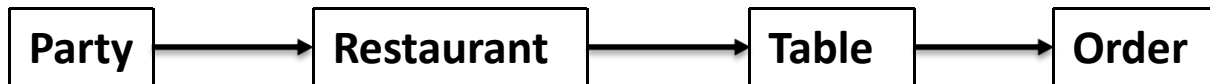
思考模式2:

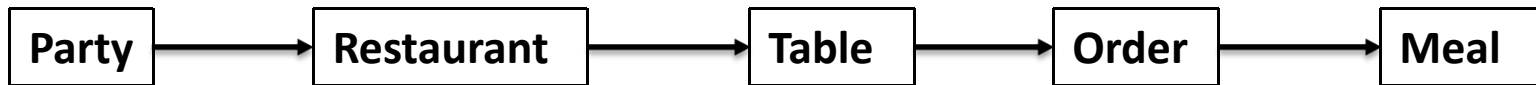


Restaurant









Party

Restaurant
- List<Table> tables

Table

Order

Meal

Party

Restaurant
<ul style="list-style-type: none">- List<Table> tables- List<Meal> menu

Table

Order

Meal

Party

Restaurant
<ul style="list-style-type: none">- List<Table> tables- List<Meal> menu

Table
<ul style="list-style-type: none">- Party p

Order

Meal

Party

Restaurant
<ul style="list-style-type: none">- List<Table> tables- List<Meal> menu

Table

Order

Meal

Party

Restaurant
<ul style="list-style-type: none">- List<Table> tables- List<Meal> menu

Table

Order
<ul style="list-style-type: none">- List<Meal> meals

Meal

- Party

- Party
- Make order ?

- Restaurant

- Restaurant
- Find table

- Restaurant
 - Find table
 - Take Order

- Restaurant
 - Find table
 - Take Order
 - Checkout

- Table

- N/A

- Order

- N/A

- Meal

- N/A

- Management 类常见use case

Reserve:

Serve:

Checkout:

Use case summary



- Mangement 类常见use case

Reserve: 暂不考虑

Use case summary



- Management 类常见use case

Reserve: 暂不考虑

Serve: Find table, Take order

Use case summary



- Management 类常见use case

Reserve: 暂不考虑

Serve: Find table, Take order

Checkout: checkout

Party

Restaurant
- List<Table> tables - List<Meal> menu

Table

Order
- List<Meal> meals

Meal

Use cases
Find table
Take order
Checkout

- Use case: Find table
- Restaurant find an available table, and change the table to be unavailable

Party

Restaurant
- List<Table> tables - List<Meal> menu
+ Table findTable()

Table

Order
- List<Meal> meals

Meal

Use cases
Find table
Take order
Checkout

Party

Restaurant
- List<Table> tables - List<Meal> menu
+ Table findTable()

Table
- Boolean available

Order
- List<Meal> meals

Meal

Use cases
Find table
Take order
Checkout

Party

Restaurant
- List<Table> tables - List<Meal> menu
+ Table findTable()

Table
- Boolean available
+ boolean isAvailable()

Order
- List<Meal> meals

Meal

Use cases
Find table
Take order
Checkout

Party

Restaurant
- List<Table> tables - List<Meal> menu
+ Table findTable()

Table
- Boolean available
+ boolean isAvailable() + void changeStatus()

Order
- List<Meal> meals

Meal

Use cases
Find table
Take order
Checkout

Party

Restaurant
- List<Table> tables - List<Meal> menu
+ Table findTable()

Table
- Boolean available
+ boolean isAvailable() + void changeStatus()

Order
- List<Meal> meals

Meal

Use cases
Find table
Take order
Checkout

Party

Restaurant
- List<Table> tables - List<Meal> menu
+ Table findTable()

Table
- Boolean available
+ boolean isAvailable() + void markUnavailable()

Order
- List<Meal> meals

Meal

Use cases
Find table
Take order
Checkout

- Use case: Take order
- Restaurant takes an order

Party

Restaurant
- List<Table> tables - List<Meal> menu
+ Table findTable() + void takeOrder(Order o)

Table
- Boolean available
+ boolean isAvailable() + void markUnavailable()

Order
- List<Meal> meals

Meal

Use cases
Find table
Take order
Checkout

- Use case: Check out
 - Restaurant checks out a table/order, mark the table available again
 - Calculate order price

Party

Restaurant
<ul style="list-style-type: none">- List<Table> tables- List<Meal> menu
<ul style="list-style-type: none">+ Table findTable()+ void takeOrder(Order o)

Table
<ul style="list-style-type: none">- Boolean available
<ul style="list-style-type: none">+ boolean isAvailable()+ void markUnavailable()

Order
<ul style="list-style-type: none">- List<Meal> meals

Meal

Use cases
Find table
Take order
Checkout

Party

Restaurant
<ul style="list-style-type: none">- List<Table> tables- List<Meal> menu
<ul style="list-style-type: none">+ Table findTable()+ void takeOrder(Order o)

Table
<ul style="list-style-type: none">- Boolean available
<ul style="list-style-type: none">+ boolean isAvailable()+ void markUnavailable()

Order
<ul style="list-style-type: none">- List<Meal> meals- Table t

Meal

Use cases
Find table
Take order
Checkout

Party

Restaurant
<ul style="list-style-type: none">- List<Table> tables- List<Meal> menu
<ul style="list-style-type: none">+ Table findTable()+ void takeOrder(Order o)+ void checkout(Order o)

Table
<ul style="list-style-type: none">- Boolean available
<ul style="list-style-type: none">+ boolean isAvailable()+ void markUnavailable()

Order
<ul style="list-style-type: none">- List<Meal> meals- Table t

Meal

Use cases
Find table
Take order
Checkout

Party

Restaurant
<ul style="list-style-type: none">- List<Table> tables- List<Meal> menu
<ul style="list-style-type: none">+ Table findTable()+ void takeOrder(Order o)+ void checkout(Order o)

Table
<ul style="list-style-type: none">- Boolean available
<ul style="list-style-type: none">+ boolean isAvailable()+ void markUnavailable()+ void markAvailable()

Order
<ul style="list-style-type: none">- List<Meal> meals- Table t

Meal

Use cases
Find table
Take order
Checkout

Party

Restaurant
<ul style="list-style-type: none">- List<Table> tables- List<Meal> menu
<ul style="list-style-type: none">+ Table findTable()+ void takeOrder(Order o)+ void checkout(Order o)

Table
<ul style="list-style-type: none">- Boolean available
<ul style="list-style-type: none">+ boolean isAvailable()+ void markUnavailable()+ void markAvailable()

Order
<ul style="list-style-type: none">- List<Meal> meals- Table t

Meal
<ul style="list-style-type: none">- Float price
<ul style="list-style-type: none">+ float getPrice()

Use cases
Find table
Take order
Checkout

Party

Restaurant
<ul style="list-style-type: none">- List<Table> tables- List<Meal> menu
<ul style="list-style-type: none">+ Table findTable()+ void takeOrder(Order o)+ void checkout(Order o)

Table
<ul style="list-style-type: none">- Boolean available
<ul style="list-style-type: none">+ boolean isAvailable()+ void markUnavailable()+ void markAvailable()

Order
<ul style="list-style-type: none">- List<Meal> meals- Table t
<ul style="list-style-type: none">+ float getPrice()

Meal
<ul style="list-style-type: none">- Float price
<ul style="list-style-type: none">+ float getPrice()

Use cases
Find table
Take order
Checkout

Party

Restaurant
<ul style="list-style-type: none">- List<Table> tables- List<Meal> menu
<ul style="list-style-type: none">+ Table findTable()+ void takeOrder(Order o)+ void checkout(Order o)

Table
<ul style="list-style-type: none">- Boolean available
<ul style="list-style-type: none">+ boolean isAvailable()+ void markUnavailable()+ void markAvailable()

Order
<ul style="list-style-type: none">- List<Meal> meals- Table t- Party p
<ul style="list-style-type: none">+ float getPrice()

Meal
<ul style="list-style-type: none">- Float price
<ul style="list-style-type: none">+ float getPrice()

Use cases
Find table
Take order
Checkout

Challenge



- Share table?

Party

Restaurant
<ul style="list-style-type: none">- List<Table> tables- List<Meal> menu- Map<Table, List<Order>> orders
<ul style="list-style-type: none">+ Table findTable()+ void takeOrder(Order o)+ void checkout(Order o)

Table
<ul style="list-style-type: none">- Boolean available
<ul style="list-style-type: none">+ boolean isAvailable()+ void markUnavailable()+ void markAvailable()

Order
<ul style="list-style-type: none">- List<Meal> meals- Table t- Party p
<ul style="list-style-type: none">+ float getPrice()

Meal
<ul style="list-style-type: none">- Float price
<ul style="list-style-type: none">+ float getPrice()

Use cases
Find table
Take order
Checkout

Party

Restaurant
<ul style="list-style-type: none">- List<Table> tables- List<Meal> menu- Map<Table, List<Order>> orders
<ul style="list-style-type: none">+ Table findTable()+ void takeOrder(Order o)+ void checkout(Order o)

Table
<ul style="list-style-type: none">- Boolean available- Int availableSeats
<ul style="list-style-type: none">+ boolean isAvailable()

Order
<ul style="list-style-type: none">- List<Meal> meals- Table t- Party p
<ul style="list-style-type: none">+ float getPrice()

Meal
<ul style="list-style-type: none">- Float price
<ul style="list-style-type: none">+ float getPrice()

Use cases
Find table
Take order
Checkout

Party
- Int size

Restaurant
<ul style="list-style-type: none">- List<Table> tables- List<Meal> menu- Map<Table, List<Order>> orders
<ul style="list-style-type: none">+ Table findTable()+ void takeOrder(Order o)+ void checkout(Order o)

Table
<ul style="list-style-type: none">- Boolean available- Int availableSeats
+ boolean isAvailable()

Order
<ul style="list-style-type: none">- List<Meal> meals- Table t- Party p
+ float getPrice()

Meal
- Float price
+ float getPrice()

Use cases
Find table
Take order
Checkout

Party
- Int size

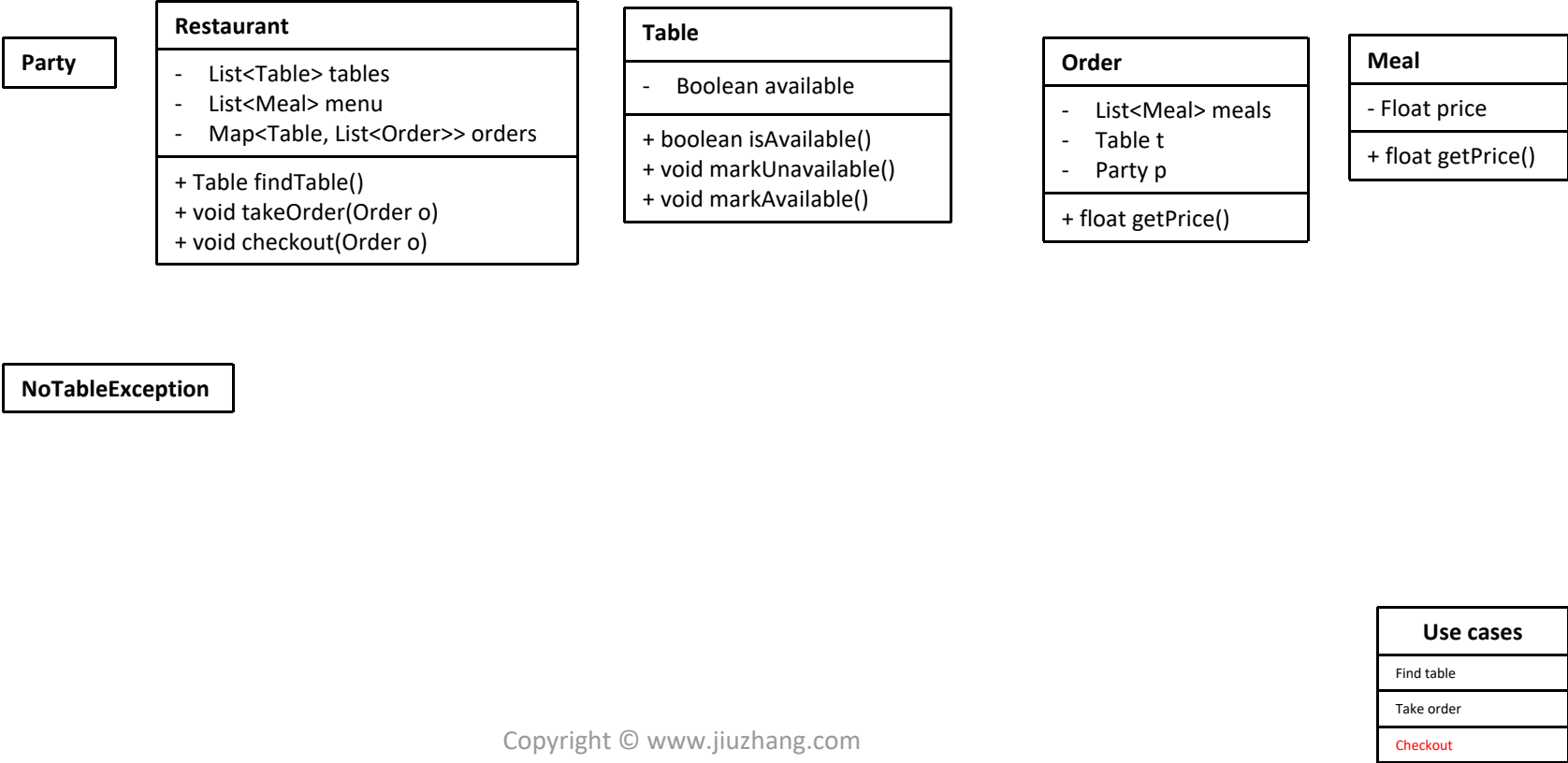
Restaurant
<ul style="list-style-type: none">- List<Table> tables- List<Meal> menu- Map<Table, List<Order>> orders
<ul style="list-style-type: none">+ Table findTable(Party p)+ void takeOrder(Order o)+ void checkout(Order o)

Table
<ul style="list-style-type: none">- Boolean available- Int availableSeats
+ boolean isAvailable()

Order
<ul style="list-style-type: none">- List<Meal> meals- Table t- Party p
+ float getPrice()

Meal
- Float price
+ float getPrice()

Use cases
Find table
Take order
Checkout



Challenge

- How can you change your design to support reservation in your restaurant?



- 预定类OOD题型
- 预定类OOD解题思路
- Hotel reservation system
- Booking.com

- Restaurant reservation system

- Restaurant reservation system
- Hotel reservation system

- Restaurant reservation system
- Hotel reservation system
- Flight/Bus/Train reservation system

- 频率：中

- 频率：中
- 难度：高

- What
 - 考虑预定的东西

- What
 - 考虑预定的东西

例子：机票

- What
 - 考虑预定的东西

例子：机票

机舱/座位号/...

- Use case
 - Search
 - Select
 - Cancel

- Use case

Search criteria -> Search() -> List<Result> -> Select() -> Receipt

- 有哪些需要和面试官统一的contract?

- 有哪些需要和面试官统一的contract?

Search criteria -> Search() -> List<Result> -> Select() -> Receipt

- Search criteria

- Search criteria

Make a reservation

2 people	▼	Oct 10, 2017	▼	7:00 PM	▼	Find a Table
----------	---	--------------	---	---------	---	--------------

- Search criteria

Make a reservation

2 people	▼	Oct 10, 2017	▼	7:00 PM	▼	Find a Table
----------	---	--------------	---	---------	---	--------------

- 人数： 无拼桌，每张桌子大小相同，不会有超过桌子大小的人数

- Search criteria

Make a reservation

2 people	▼	Oct 10, 2017	▼	7:00 PM	▼	Find a Table
----------	---	--------------	---	---------	---	--------------

- 人数：无拼桌，每张桌子大小相同，不会有超过桌子大小的人数
- 日期：是否允许预定多日以后的？

- Search criteria

Make a reservation

2 people	▼	Oct 10, 2017	▼	7:00 PM	▼	Find a Table
----------	---	--------------	---	---------	---	--------------

- 人数：无拼桌，每张桌子大小相同，不会有超过桌子大小的人数
- 日期：是否允许预定多日以后的？
- 时间：是否所有时间都允许预定？

- Search criteria

Make a reservation

2 people	▼	Oct 10, 2017	▼	7:00 PM	▼	Find a Table
----------	---	--------------	---	---------	---	--------------

- 人数：无拼桌，每张桌子大小相同，不会有超过桌子大小的人数
- 日期：是否允许预定多日以后的？ - 允许
- 时间：是否所有时间都允许预定？ - 24/7

- Search criteria

Make a reservation

2 people	▼	Oct 10, 2017	▼	7:00 PM	▼	Find a Table
----------	---	--------------	---	---------	---	--------------

- 人数：无拼桌，每张桌子大小相同，不会有超过桌子大小的人数
- 日期：是否允许预定多日以后的？ - 允许
- 时间：是否所有时间都允许预定？ - 24/7
- Design: FindTableForReservation(Timeslot t)

- Search criteria

Make a reservation

2 people	▼	Oct 10, 2017	▼	7:00 PM	▼	Find a Table
----------	---	--------------	---	---------	---	--------------

- 人数：无拼桌，每张桌子大小相同，不会有超过桌子大小的人数
- 日期：是否允许预定多日以后的？ - 允许
- 时间：是否所有时间都允许预定？ - 24/7
- Design: FindTableForReservation(Timeslot t)
- Timeslot contains Date and time

- List<Result>

- List<Result>

- 当选择的时间段可以/不行时，系统应该给出什么反馈？

Reservation



- 做法一:

Make a reservation

2 people	Nov 2, 2017	7:30 PM	Find a Table
----------	-------------	---------	--------------

Booked 22 times today

5:30 PM	6:00 PM	7:30 PM	8:00 PM	8:30 PM
---------	---------	---------	---------	---------

[See affiliated restaurants](#)

Make a reservation

2 people	Oct 10, 2017	7:00 PM	Find a Table
----------	--------------	---------	--------------

Booked 22 times today

You're in luck! We still have 1 timeslot left

			9:30 PM	
--	--	--	---------	--

[See affiliated restaurants](#)

- List<Result>


Result == Timeslot

- List<Result>


Result == Timeslot

- Design: List<Timeslot> findTableForReservation(Timeslot t)
- Possible Challenge: 跟面试官讨论如何获得这个List

- 做法二:
 - 可以预定：直接进入Confirm阶段
 - 不能预定：Throw exception / Show message






GUESTS	DATE	TIME	RESTAURANT
2 people	Tue, Oct 10, 2017	9:30 PM	Tosceno Harvard Square

 Hurry! We can only hold this table for you for 4:34 minutes

POINTS
Not a member? Join OpenTable and receive 100 points upon dining.

Already a member? [Sign in](#)

<input type="text" value="First Name"/>	<input type="text" value="Last Name"/>
<div> Phone Number</div>	<input type="text" value="Email"/>
<div>Select an Occasion (optional) </div>	<div>Add a special request (optional) </div>

☐ Yes, I want to receive email messages from this restaurant.
☐ Yes, I want to get text updates and reminders about my reservations*

Complete Reservation

- 做法二：
 - 可以预定：直接进入Confirm阶段
 - 不能预定：Throw exception / Show message

- Design: `Pair<Table, Timeslot> findTableForReservation(Time slot)`
throws `NoTableForReservationException`

- 做法二：
 - 可以预定：直接进入Confirm阶段
 - 不能预定：Throw exception / Show message
- Design: `Pair<Table, Timeslot> findTableForReservation(Time slot)`
throws `NoTableForReservationException`
- Design: `void confirmReservation(Pair<Table, Timeslot> reservation)`

- 为什么我们可以跳过List<Result>这个步骤？

- 为什么我们可以跳过List<Result>这个步骤?

因为Table是一样的，用户不用选择也不会知道是订1号桌还是2号桌

Party
- Int capacity
+ int getCapacity()

Restaurant
- List<Table> tables - List<Meal> menu - Map<Table, List<Order>> orders
+ Table findTable() + void takeOrder(Order o) + void checkout(Order o)

Table
- Boolean available - Int capacity
+ boolean isAvailable() + void markUnavailable() + void markAvailable() + get capacity()

Order
- List<Meal> meals - Table t - Party p
+ float getPrice()

Meal
- Float price
+ float getPrice()

NoTableException

Use cases
Find table
Take order
Checkout

Party
- Int capacity
+ int getCapacity()

Restaurant
- List<Table> tables - List<Meal> menu - Map<Table, List<Order>> orders
+ Table findTable() + void takeOrder(Order o) + void checkout(Order o)

Table
- Boolean available - Int capacity
+ boolean isAvailable() + void markUnavailable() + void markAvailable() + get capacity()

Order
- List<Meal> meals - Table t - Party p
+ float getPrice()

Meal
- Float price
+ float getPrice()

Reservation
- Table table - Timeslot timeslot

NoTableException

Use cases
Find table
Take order
Checkout

Party
- Int capacity
+ int getCapacity()

Restaurant
- List<Table> tables - List<Meal> menu - Map<Table, List<Order>> orders
+ Table findTable() + void takeOrder(Order o) + void checkout(Order o) + Reservation findTableForReservation(Timeslot t)

Table
- Boolean available - Int capacity
+ boolean isAvailable() + void markUnavailable() + void markAvailable() + get capacity()

Reservation
- Table table - Timeslot timeslot

Order
- List<Meal> meals - Table t - Party p
+ float getPrice()

Meal
- Float price
+ float getPrice()

NoTableException

Use cases
Find table
Take order
Checkout

Party
- Int capacity
+ int getCapacity()

Restaurant
- List<Table> tables - List<Meal> menu - Map<Table, List<Order>> orders
+ Table findTable() + void takeOrder(Order o) + void checkout(Order o) + Reservation findTableForReservation(Timeslot t)

Table
- Boolean available - Int capacity
+ boolean isAvailable() + void markUnavailable() + void markAvailable() + get capacity()

Reservation
- Table table - Timeslot timeslot

Order
- List<Meal> meals - Table t - Party p
+ float getPrice()

Meal
- Float price
+ float getPrice()

NoTableException

NoTableForReservationException

Use cases
Find table
Take order
Checkout

Party
- Int capacity
+ int getCapacity()

Restaurant
- List<Table> tables - List<Meal> menu - Map<Table, List<Order>> orders
+ Table findTable() + void takeOrder(Order o) + void checkout(Order o) + Reservation findTableForReservation(Timeslot t) + void confirmReservation(Reservation r)

Table
- Boolean available - Int capacity
+ boolean isAvailable() + void markUnavailable() + void markAvailable() + get capacity()

Reservation
- Table table - Timeslot timeslot

Order
- List<Meal> meals - Table t - Party p
+ float getPrice()

Meal
- Float price
+ float getPrice()

NoTableException

NoTableForReservationException

Use cases
Find table
Take order
Checkout

Any problems?



Any problems?

- How to know if a table is open for reservation for a timeslot?

Any problems?

- How to know if a table is open for reservation for a timeslot?

要知道每张桌子的预定情况

Table
<ul style="list-style-type: none">- Boolean available- Int capacity- List<Timeslot> reservations
<ul style="list-style-type: none">+ boolean isAvailable()+ void markUnavailable()+ void markAvailable()+ get capacity()

Any problems?

- How to know if a table is open for reservation for a timeslot?

把时间点转换为时间段

Table
<ul style="list-style-type: none">- Boolean available- Int capacity- List<TimePeriod> reservations
<ul style="list-style-type: none">+ boolean isAvailable()+ void markUnavailable()+ void markAvailable()+ get capacity()

Party
- Int capacity
+ int getCapacity()

Restaurant
- List<Table> tables - List<Meal> menu - Map<Table, List<Order>> orders
+ Table findTable() + void takeOrder(Order o) + void checkout(Order o) + Reservation findTableForReservation(Timeslot t) + void confirmReservation(Reservation r)

Table
- Boolean available - Int capacity
+ boolean isAvailable() + void markUnavailable() + void markAvailable() + get capacity()

Reservation
- Table table - Timeslot timeslot

Order
- List<Meal> meals - Table t - Party p
+ float getPrice()

Meal
- Float price
+ float getPrice()

NoTableException

NoTableForReservationException

Use cases
Find table
Take order
Checkout

Party
- Int capacity
+ int getCapacity()

TimePeroid
- Time start
- Time end

Restaurant
- List<Table> tables
- List<Meal> menu
- Map<Table, List<Order>> orders
+ Table findTable()
+ void takeOrder(Order o)
+ void checkout(Order o)
+ Reservation findTableForReservation(Timeslot t)
+ void confirmReservation(Reservation r)

Table
- Boolean available
- Int capacity
+ boolean isAvailable()
+ void markUnavailable()
+ void markAvailable()
+ get capacity()

Reservation
- Table table
- Timeslot timeslot

Order
- List<Meal> meals
- Table t
- Party p
+ float getPrice()

Meal
- Float price
+ float getPrice()

NoTableException

NoTableForReservationException

Use cases
Find table
Take order
Checkout

Party
- Int capacity
+ int getCapacity()

TimePeroid
- Time start
- Time end

Restaurant
- List<Table> tables
- List<Meal> menu
- Map<Table, List<Order>> orders
+ Table findTable()
+ void takeOrder(Order o)
+ void checkout(Order o)
+ Reservation findTableForReservation(Timeslot t)
+ void confirmReservation(Reservation r)

Table
- Boolean available
- Int capacity
+ boolean isAvailable()
+ void markUnavailable()
+ void markAvailable()
+ get capacity()

Reservation
- Table table
- TimePeroid timePeriod

Order
- List<Meal> meals
- Table t
- Party p
+ float getPrice()

Meal
- Float price
+ float getPrice()

NoTableException

NoTableForReservationException

Use cases
Find table
Take order
Checkout

Party
- Int capacity
+ int getCapacity()

TimePeroid
- Time start
- Time end

Table
- Boolean available
- Int capacity
- List<TimePeroid> reservations
+ boolean isAvailable()
+ void markUnavailable()
+ void markAvailable()
+ get capacity()

Order
- List<Meal> meals
- Table t
- Party p
+ float getPrice()

Meal
- Float price
+ float getPrice()

Restaurant
- List<Table> tables
- List<Meal> menu
- Map<Table, List<Order>> orders
+ Table findTable()
+ void takeOrder(Order o)
+ void checkout(Order o)
+ Reservation findTableForReservation(Timeslot t)
+ void confirmReservation(Reservation r)

Reservation
- Table table
- TimePeroid timePeriod

NoTableException

NoTableForReservationException

Use cases
Find table
Take order
Checkout

Party
- Int capacity
+ int getCapacity()

TimePeroid
- Time start
- Time end

Table
- Boolean available
- Int capacity
+ boolean isAvailable()
+ void markUnavailable()
+ void markAvailable()
+ get capacity()

Order
- List<Meal> meals
- Table t
- Party p
+ float getPrice()

Meal
- Float price
+ float getPrice()

Restaurant
- List<Table> tables
- List<Meal> menu
- Map<Table, List<Order>> orders
- Map<Table, List<TimePeroid>> reservation
+ Table findTable()
+ void takeOrder(Order o)
+ void checkout(Order o)
+ Reservation findTableForReservation(Timeslot t)
+ void confirmReservation(Reservation r)

Reservation
- Table table
- TimePeroid timePeroid

NoTableException

NoTableForReservationException

Use cases
Find table
Take order
Checkout

- Restaurant takes a cancel reservation request, and ask the table to free that timeslot.

Party
- Int capacity
+ int getCapacity()

TimePeroid
- Time start
- Time end

Table
- Boolean available
- Int capacity
+ boolean isAvailable()
+ void markUnavailable()
+ void markAvailable()
+ get capacity()

Order
- List<Meal> meals
- Table t
- Party p
+ float getPrice()

Meal
- Float price
+ float getPrice()

Restaurant
- List<Table> tables
- List<Meal> menu
- Map<Table, List<Order>> orders
- Map<Table, List<TimePeroid>> reservation
+ Table findTable()
+ void takeOrder(Order o)
+ void checkout(Order o)
+ Reservation findTableForReservation(Timeslot t)
+ void confirmReservation(Reservation r)
+ void cancelReservation(Reservation r)

Reservation
- Table table
- TimePeroid timePeroid

NoTableException

NoTableForReservationException

Use cases
Find table
Take order
Checkout

- Can you design a hotel reservation system?

Hotel reservation system



- Can you design a hotel reservation system?

Search

Destination/Property Name:
New York City

Are you traveling for work?
☐ Yes ☒ No

Check-in
Friday, November 17, 20...

Check-out
Saturday, November 18, 20...

1-night stay

Rooms
1 room

Adults
2 adults

Children
No children

Search

Welcome to New York City, the Big Apple
433 properties found – including 112 value deals!

3 Reasons to Visit:

Central Park
Sightseeing

Metropolitan Museum Art
Fine Art Museums

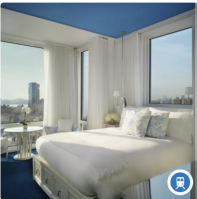
Our Top Picks

Lowest Price First

Value for Money

Recommended for You:

MODERN Sleek design, contemporary comfort.



NoMo SoHo

SoHo, New York City, NY – Show on map

Subway Access

45 people are looking right now

Booked 41 times in the last 24 hours


Great Value Today

Double Room

In high demand!


Risk Free: You can cancel later, so lock in

VS.



漫趣乐园-如家上海浦东机场店

★★★★★ 5.0 分 28 点评



上海市浦东新区川沙路4518号

入住10月10日 / 离店 10月11日

小魔仙大床房A

详情

会员价 > 390元起 预订

超级飞侠大床房A

详情

会员价 > 390元起 预订

Copyright © www.jiuzhang.com

- What

是为一间酒店设计预定房间系统，还是先选择酒店的系统？

Design上会有哪些区别？

- What

是为一间酒店设计预定房间系统，还是先选择酒店的系统？

- 搜索条件区别

人数 + 时间 VS. 人数 + 时间 + 地址

- What

是为一间酒店设计预定房间系统，还是先选择酒店的系统？

- 返回结果区别

Rooms VS. Hotels

- What

是为一间酒店设计预定房间系统，还是先选择酒店的系统？

- 针对本题：

先设计一间酒店，再设计选择酒店的系统

- What

Search criteria -> Search() -> List<Result> -> Select() -> Receipt

- What

Search criteria -> Search() -> List<Result> -> Select() -> Receipt

除了考虑题目中的名词之外，还需要从上述的三处考虑，What类型的提问主要针对List<Result>

- What

关键字: Room

• What

关键字: Room

Room Type	Sleeps	Today's Price	Your Choices	Select Rooms	Confirm Your Reservation
<p>▶ Classic Room with King Bed in high demand!</p> <p>1 king bed </p> <p>This room includes 1 extra large double bed, cable TV and a work area.</p> <p> Private bathroom</p> <p> Free WiFi</p> <p>• Iron • Free toiletries</p> <p>• Bathroom</p> <p>• Telephone • Towels</p> <p>• Linens</p>	2	<p>\$259</p> <p>Excludes: Taxes and US\$ 21.78 Property service charge per night</p> <p> There's an even lower price available! Sign in to see it</p>	<p>• Low rate -- no money back</p> <p>• Service charge</p>	0	<p>I'll reserve</p> <p>Confirmation is immediate</p> <p>101 other people looking now</p>
	2	<p>\$344</p> <p>Excludes: Taxes and US\$ 21.78 Property service charge per night</p> <p> There's an even lower price available! Sign in to see it</p>	<p>✓ FREE cancellation before Oct 19, 2017</p> <p>✓ No advance payment</p> <p>• Service charge</p>	0	
	2	<p>\$354</p> <p>Excludes: Taxes and US\$ 21.78 Property service charge per night</p> <p> There's an even lower price available! Sign in to see it</p>	<p> Breakfast included</p> <p>✓ FREE cancellation before Oct 19, 2017</p> <p>✓ No advance payment</p> <p>• Service charge</p>	0	
<p>▶ Penn 5000 Room with Double Bed</p> <p>● Only 1 room left!</p> <p>1 full bed </p> <p>This room includes a refrigerator and a flat-screen TV.</p> <p> Private bathroom</p> <p> Free WiFi</p> <p>More</p>	2	<p>\$269</p> <p>Excludes: Taxes and US\$ 21.78 Property service charge per night</p> <p> There's an even lower price available! Sign in to see it</p>	<p>• Low rate -- no money back</p> <p>• Service charge</p>	0	
	2	<p>\$354</p> <p>Excludes: Taxes and US\$ 21.78 Property service charge per night</p> <p> There's an even lower price available! Sign in to see it</p>	<p>✓ FREE cancellation before Oct 19, 2017</p> <p>✓ No advance payment</p> <p>• Service charge</p>	0	
	2	<p>\$364</p> <p>Excludes: Taxes and US\$ 21.78 Property service charge per night</p> <p> There's an even lower price available! Sign in to see it</p>	<p> Breakfast included</p> <p>✓ FREE cancellation before Oct 19, 2017</p> <p>✓ No advance payment</p> <p>• Service charge</p>	0	

- What

针对本题：房间的人数和价格可能会不同

Challenge

针对本题：房间的人数和价格可能会不同。

如何设计房间类？

Challenge

如何设计房间类?

Room
<ul style="list-style-type: none">- int capacity- float price

Challenge

如何设计房间类?

Room
<ul style="list-style-type: none">- int capacity- float price

List<Result> -> List<Room>

Room_1
Capacity : 2 Price: 198

Room_2
Capacity : 2 Price: 198

Room_3
Capacity : 2 Price: 198

Room_4
Capacity : 2 Price: 198

Room_5
Capacity : 1 Price: 128

Room_6
Capacity : 1 Price: 128

Challenge



如何设计房间类?

Room
<ul style="list-style-type: none">- int capacity- float price

List<Result> -> List<Room>

Room_1
Capacity : 2 Price: 198

Room_2
Capacity : 2 Price: 198

Room_3
Capacity : 2 Price: 198

Room_4
Capacity : 2 Price: 198

Room_5
Capacity : 1 Price: 128

Room_6
Capacity : 1 Price: 128

Room Type	Sleeps	Today's Price	Your Choices	Select Rooms	Confirm Your Reservation
Classic Room with King Bed High demand! 1 king bed This room includes 1 extra large double bed, cable TV and a work area.	2	\$259 Excludes: Taxes and USD 21.78 Property service charge per night There's an even lower price available. See it to save it	<ul style="list-style-type: none">Low rate -- no money backService charge	0 2	Fill reserve <small>Confirmation is irreversible</small> 101 other people looking now
Private bathroom Free WiFi • Iron • Free toiletries • Bathroom • Telephone • Towels • Linens	2	\$344 Excludes: Taxes and USD 21.78 Property service charge per night There's an even lower price available. See it to save it	<ul style="list-style-type: none">FREE cancellation before Oct 19, 2017No advance paymentService charge	0 2	
Private bathroom Free WiFi More	2	\$354 Excludes: Taxes and USD 21.78 Property service charge per night There's an even lower price available. See it to save it	<ul style="list-style-type: none">Breakfast includedFREE cancellation before Oct 19, 2017No advance paymentService charge	0 2	
Queen 5000 Room with Double Bed Only 1 room left! 1 full bed This room includes a refrigerator and a flat-screen TV.	2	\$269 Excludes: Taxes and USD 21.78 Property service charge per night There's an even lower price available. See it to save it	<ul style="list-style-type: none">Low rate -- no money backService charge	0 2	
Private bathroom Free WiFi More	2	\$354 Excludes: Taxes and USD 21.78 Property service charge per night There's an even lower price available. See it to save it	<ul style="list-style-type: none">FREE cancellation before Oct 19, 2017No advance paymentService charge	0 2	
	2	\$364 Excludes: Taxes and USD 21.78 Property service charge per night There's an even lower price available. See it to save it	<ul style="list-style-type: none">Breakfast includedFREE cancellation before Oct 19, 2017No advance paymentService charge	0 2	

- How

规则?

- 预定类规则:

Search criteria

- 预定类规则:

Search criteria

Search

Destination/Property Name:

Are you traveling for work? ?

☐ Yes ☐ No

Check-in

Check-out

1-night stay

Rooms

Adults

Children

Welcome to New York City, the Big Apple

433 properties found – including 112 value deals!

3 Reasons to Visit:

Central Park
Sightseeing

Metropolitan Museum Art
Fine Art Museums

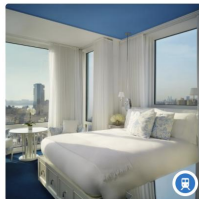
Our Top Picks

Lowest Price First

Value for Money

Recommended for You:

MODERN Sleek design, contemporary comfort.



NoMo SoHo

[SoHo, New York City, NY – Show on map](#)
Subway Access

45 people are looking right now
Booked 41 times in the last 24 hours

Great Value Today

Double Room

In high demand!

Risk Free: You can cancel later, so lock in

Hotel

Core object



Request

Hotel

Core object



Request

Hotel

RoomType

Core object



Request

Hotel

RoomType

Room

Core object

Request

Hotel

RoomType

Reservation

Room

Core object



Request

Hotel

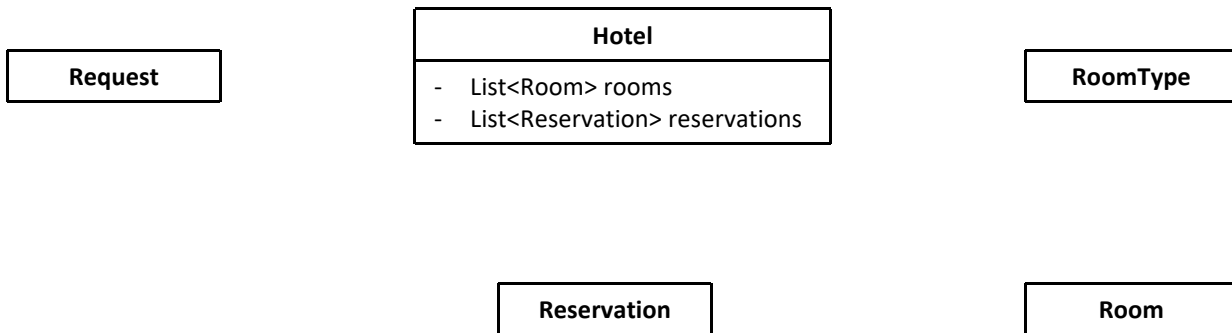
- List<Room> rooms

RoomType

Reservation

Room

Core object



Core object

Request

Hotel
<ul style="list-style-type: none">- List<Room> rooms- List<Reservation> reservations

RoomType

Reservation

Room
<ul style="list-style-type: none">- RoomType type

Core object



Request

Hotel
<ul style="list-style-type: none">- List<Room> rooms- List<Reservation> reservations

RoomType

Reservation
<ul style="list-style-type: none">- List<Room> rooms

Room
<ul style="list-style-type: none">- RoomType type

Hotel

Hotel:

- Search for available rooms

Hotel:

- Search for available rooms
- Make reservation

Hotel:

- Search for available rooms
- Make reservation
- Cancel reservation

Classes

Request

Hotel
<ul style="list-style-type: none">- List<Room> rooms- List<Reservation> reservations

RoomType

Room
<ul style="list-style-type: none">- RoomType type

Reservation
<ul style="list-style-type: none">- List<Room> rooms

Use cases
Search for available rooms
Make reservation
Cancel reservation

Search for available rooms

- 1: Based on search criteria
- 2: Go through rooms to check availability
- 3: list available room types and available count

Search for available rooms

1: Based on search criteria

Classes

Request
- Date startDate

Hotel
- List<Room> rooms - List<Reservation> reservations

RoomType

Room
- RoomType type

Reservation
- List<Room> rooms

Use cases
Search for available rooms
Make reservation
Cancel reservation

Classes

Request
- Date startDate
- Date endDate

Hotel
- List<Room> rooms
- List<Reservation> reservations

RoomType

Room
- RoomType type

Reservation
- List<Room> rooms

Use cases
Search for available rooms
Make reservation
Cancel reservation

Classes



Request
- Date startDate
- Date endDate

Hotel
- List<Room> rooms
- List<Reservation> reservations
+ Map<RoomType, int> handleSearchRequest(Request r)

RoomType

Room
- RoomType type

Reservation
- List<Room> rooms

Use cases
Search for available rooms
Make reservation
Cancel reservation

Search for available rooms

2: Go through rooms to check availability

Request
<ul style="list-style-type: none">- Date startDate- Date endDate

Hotel
<ul style="list-style-type: none">- List<Room> rooms- List<Reservation> reservations
+ Map<RoomType, int> handleSearchRequest(Request r)

RoomType

Room
<ul style="list-style-type: none">- RoomType type- List<Date> reservations

Reservation
- List<Room> rooms

Use cases
Search for available rooms
Make reservation
Cancel reservation

Classes

Request
- Date startDate
- Date endDate

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
+ Map<RoomType, int> handleSearchRequest(Request r)

RoomType

Room
- RoomType type

Reservation
- List<Room> rooms

Use cases
Search for available rooms
Make reservation
Cancel reservation

Classes

Request
- Date startDate
- Date endDate

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)

RoomType

Room
- RoomType type

Reservation
- List<Room> rooms

Use cases
Search for available rooms
Make reservation
Cancel reservation

Classes

Request
- Date startDate
- Date endDate

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)

RoomType

Room
- RoomType type
- Boolean available

Reservation
- List<Room> rooms

Use cases
Search for available rooms
Make reservation
Cancel reservation

Classes



Request
<ul style="list-style-type: none">- Date startDate- Date endDate

Hotel
<ul style="list-style-type: none">- List<Room> rooms- List<Reservation> reservations- Map<Room, List<Date>> roomReservations
<ul style="list-style-type: none">+ Map<RoomType, int> handleSearchRequest(Request r)- boolean isRequestAvailable(Request r, List<Date> dates)

RoomType

Room
<ul style="list-style-type: none">- RoomType type- Boolean available
<ul style="list-style-type: none">+ boolean isAvailable()

Reservation
<ul style="list-style-type: none">- List<Room> rooms

Use cases
Search for available rooms
Make reservation
Cancel reservation

Search for available rooms

3: list available room types and counts

Classes



Request
- Date startDate
- Date endDate

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)

RoomType
SINGLE
DOUBLE

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

Reservation
- List<Room> rooms

Use cases
Search for available rooms
Make reservation
Cancel reservation

Request
<ul style="list-style-type: none">- Date startDate- Date endDate

Hotel
<ul style="list-style-type: none">- List<Room> rooms- List<Reservation> reservations- Map<Room, List<Date>> roomReservations
<ul style="list-style-type: none">+ Map<RoomType, int> handleSearchRequest(Request r)- boolean isRequestAvailable(Request r, List<Date> dates)

<<Enumeration>> RoomType
SINGLE
DOUBLE

Room
<ul style="list-style-type: none">- RoomType type- Boolean available
+ boolean isAvailable()

Reservation
- List<Room> rooms

Use cases
Search for available rooms
Make reservation
Cancel reservation

Make reservation

- 1: Add RoomType and number of rooms in a request
- 2: Send request to Hotel
- 3: If there is enough room left, confirm the reservation
- 4: If there isn't enough room left, throw exception

Make reservation

1: Add RoomType and number of rooms in a request

Request
<ul style="list-style-type: none">- Date startDate- Date endDate

Hotel
<ul style="list-style-type: none">- List<Room> rooms- List<Reservation> reservations- Map<Room, List<Date>> roomReservations
<ul style="list-style-type: none">+ Map<RoomType, int> handleSearchRequest(Request r)- boolean isRequestAvailable(Request r, List<Date> dates)

<<Enumeration>> RoomType
<ul style="list-style-type: none">SINGLEDOUBLE

Room
<ul style="list-style-type: none">- RoomType type- Boolean available
<ul style="list-style-type: none">+ boolean isAvailable()

Reservation
<ul style="list-style-type: none">- List<Room> rooms

Use cases
Search for available rooms
Make reservation
Cancel reservation

Request
- Date startDate
- Date endDate

Reservation
- List<Room> rooms

ReservationRequest

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)

<<Enumeration>> RoomType
SINGLE
DOUBLE

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

Use cases
Search for available rooms
Make reservation
Cancel reservation

Request
<ul style="list-style-type: none">- Date startDate- Date endDate

Reservation
<ul style="list-style-type: none">- List<Room> rooms

ReservationRequest
<ul style="list-style-type: none">- Date startDate- Date endDate

Hotel
<ul style="list-style-type: none">- List<Room> rooms- List<Reservation> reservations- Map<Room, List<Date>> roomReservations
<ul style="list-style-type: none">+ Map<RoomType, int> handleSearchRequest(Request r)- boolean isRequestAvailable(Request r, List<Date> dates)

<<Enumeration>> RoomType
<ul style="list-style-type: none">SINGLEDOUBLE

Room
<ul style="list-style-type: none">- RoomType type- Boolean available
<ul style="list-style-type: none">+ boolean isAvailable()

Use cases
Search for available rooms
Make reservation
Cancel reservation

Classes

Request
- Date startDate
- Date endDate

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)

<<Enumeration>> RoomType
SINGLE
DOUBLE

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

Reservation
- List<Room> rooms

ReservationRequest
- Date startDate
- Date endDate
- Map<RoomType, int> roomsNeeded

Use cases
Search for available rooms
Make reservation
Cancel reservation

Make reservation

2: Send request to Hotel

Classes

Request
- Date startDate
- Date endDate

Reservation
- List<Room> rooms

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)

<<Enumeration>> RoomType
SINGLE
DOUBLE

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

ReservationRequest
- Date startDate
- Date endDate
- Map<RoomType, int> roomsNeeded

Use cases
Search for available rooms
Make reservation
Cancel reservation

Request
<ul style="list-style-type: none">- Date startDate- Date endDate

Reservation
<ul style="list-style-type: none">- List<Room> rooms

ReservationRequest
<ul style="list-style-type: none">- Date startDate- Date endDate- Map<RoomType, int> roomsNeeded

Hotel
<ul style="list-style-type: none">- List<Room> rooms- List<Reservation> reservations- Map<Room, List<Date>> roomReservations
<ul style="list-style-type: none">+ Map<RoomType, int> handleSearchRequest(Request r)- boolean isRequestAvailable(Request r, List<Date> dates)+ Reservation makeReservation(ReservationRequest r)

<<Enumeration>> RoomType
<ul style="list-style-type: none">SINGLEDOUBLE

Room
<ul style="list-style-type: none">- RoomType type- Boolean available
<ul style="list-style-type: none">+ boolean isAvailable()

Use cases
Search for available rooms
Make reservation
Cancel reservation

Make reservation

3: If there is enough room left, confirm the reservation

Request
<ul style="list-style-type: none">- Date startDate- Date endDate

Reservation
<ul style="list-style-type: none">- List<Room> rooms- Date startDate- Date endDate

ReservationRequest
<ul style="list-style-type: none">- Date startDate- Date endDate- Map<RoomType, int> roomsNeeded

Hotel
<ul style="list-style-type: none">- List<Room> rooms- List<Reservation> reservations- Map<Room, List<Date>> roomReservations
<ul style="list-style-type: none">+ Map<RoomType, int> handleSearchRequest(Request r)- boolean isRequestAvailable(Request r, List<Date> dates)+ Reservation makeReservation(ReservationRequest r)

<<Enumeration>> RoomType
<ul style="list-style-type: none">SINGLEDOUBLE

Room
<ul style="list-style-type: none">- RoomType type- Boolean available
+ boolean isAvailable()

Use cases
Search for available rooms
Make reservation
Cancel reservation

Make reservation

4: If there isn't enough room left, throw exception

Request
<ul style="list-style-type: none">- Date startDate- Date endDate

Reservation
<ul style="list-style-type: none">- List<Room> rooms- Date startDate- Date endDate

ReservationRequest
<ul style="list-style-type: none">- Date startDate- Date endDate- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

Hotel
<ul style="list-style-type: none">- List<Room> rooms- List<Reservation> reservations- Map<Room, List<Date>> roomReservations
<ul style="list-style-type: none">+ Map<RoomType, int> handleSearchRequest(Request r)- boolean isRequestAvailable(Request r, List<Date> dates)+ Reservation makeReservation(ReservationRequest r)

<<Enumeration>> RoomType
<ul style="list-style-type: none">SINGLEDOUBLE

Room
<ul style="list-style-type: none">- RoomType type- Boolean available
+ boolean isAvailable()

Use cases
Search for available rooms
Make reservation
Cancel reservation

Search for available rooms

- 1: Based on search criteria
- 2: Go through rooms to check availability
- 3: list available room types and room count

Make reservation

- 1: Add RoomType and number of rooms in a request
- 2: Send request to Hotel
- 3: If there is enough room left, confirm the reservation
- 4: If there isn't enough room left, throw exception

Challenge



九章算法

```
Map<RoomType, List<Room>> map = new HashMap<>();

for(Entry<Room, List<Date>> entry : roomReservations.entrySet())
{
    Room r = entry.getKey();
    List<Date> roomBooked = entry.getValue();

    if(isRequestAvailable(roomBooked))
    {
        if(map.containsKey(r.getRoomType()))
        {
            List<Room> roomList = map.get(r.getRoomType());
            roomList.add(r);
            map.put(r.getRoomType(), roomList);
        }
        else
        {
            List<Room> roomList = new ArrayList<>();
            roomList.add(r);
            map.put(r.getRoomType(), roomList);
        }
    }
}
```

Challenge

- Map<RoomType, List<Room>> map
 - Go through rooms to check availability
 - If there is enough room left, confirm the reservation
 - If there isn't enough room left, throw exception

Challenge

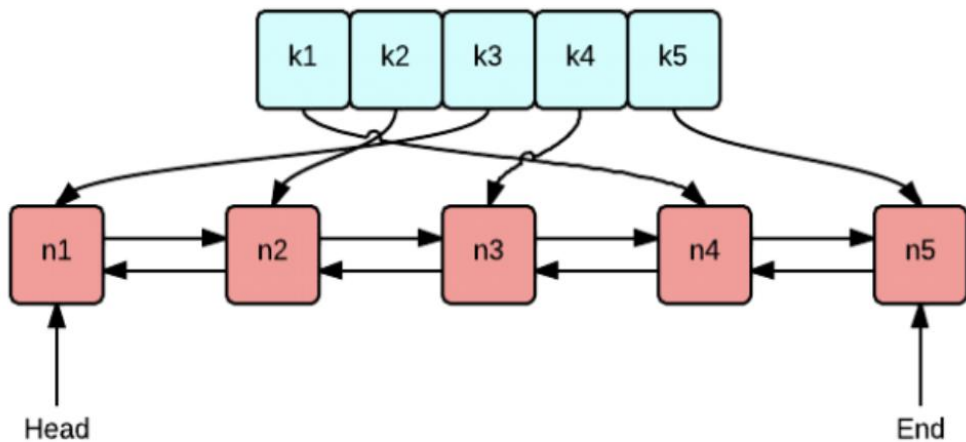
- LRU Cache

The access sequence for the below example is A B C D E D F.



Challenge

- LRU Cache



Challenge

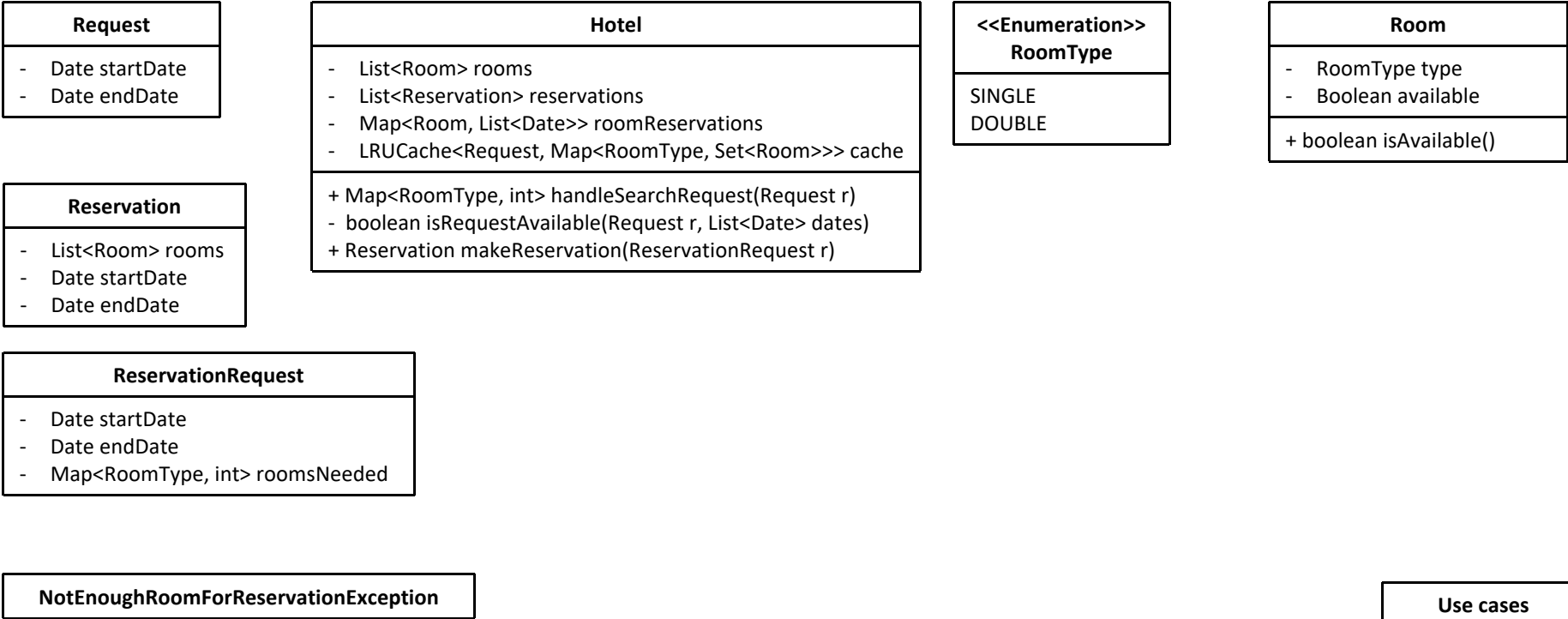


九章算法

```
class LRUCache extends LinkedHashMap<Request, Map<RoomType, List<Room>>>
{
    private int capacity;

    public LRUCache(int capacity)
    {
        super(capacity);
        this.capacity = capacity;
    }

    @Override
    protected boolean removeEldestEntry(Map.Entry<Request, Map<RoomType, List<Room>>> eldest){
        return size() > this.capacity;
    }
}
```



Cancel reservation

1: Hotel system takes a reservation, and cancel it.

Request
<ul style="list-style-type: none">- Date startDate- Date endDate

Reservation
<ul style="list-style-type: none">- List<Room> rooms- Date startDate- Date endDate

ReservationRequest
<ul style="list-style-type: none">- Date startDate- Date endDate- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

Hotel
<ul style="list-style-type: none">- List<Room> rooms- List<Reservation> reservations- Map<Room, List<Date>> roomReservations- LRU<Request, Map<RoomType, Set<Room>>> cache
<ul style="list-style-type: none">+ Map<RoomType, int> handleSearchRequest(Request r)- boolean isRequestAvailable(Request r, List<Date> dates)+ Reservation makeReservation(ReservationRequest r)+ void cancelReservation(Reservation r)

<<Enumeration>> RoomType
<ul style="list-style-type: none">SINGLEDOUBLE

Room
<ul style="list-style-type: none">- RoomType type- Boolean available
+ boolean isAvailable()

Use cases
Search for available rooms
Make reservation
Cancel reservation

Classes – Final view



Request
- Date startDate
- Date endDate

Reservation
- List<Room> rooms
- Date startDate
- Date endDate

ReservationRequest
- Date startDate
- Date endDate
- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
- LRUCache<Request, Map<RoomType, Set<Room>>> cache
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)
+ Reservation makeReservation(ReservationRequest r)
+ void cancelReservation(Reservation r)

<<Enumeration>> RoomType
SINGLE
DOUBLE

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

Use cases
Search for available rooms
Make reservation
Cancel reservation

Challenge

面试官：

Can you extend your design to a booking.com like system?

Challenge

Search criteria -> Search() -> List<Result> -> Select() -> Receipt

Challenge

Search criteria -> Search() -> List<Result> -> Select() -> Receipt

Search criteria A -> Search() -> List<Result_A> -> Select() -> Search
criteria B -> Search() -> List<Result_B> -> Select() -> Receipt

Challenge

Search criteria -> Search() -> List<Result> -> Select() -> Receipt

Search criteria A -> Search() -> List<Result_A> -> Select() -> Search
criteria B -> Search() -> List<Result_B> -> Select() -> Receipt

Search criteria A : 10/30 -11/5 Boston

Challenge

Search criteria -> Search() -> List<Result> -> Select() -> Receipt

Search criteria A -> Search() -> List<Result_A> -> Select() -> Search
criteria B -> Search() -> List<Result_B> -> Select() -> Receipt

Search criteria A : 10/30 -11/5 Boston

BookingSystem.Search()

Challenge

Search criteria -> Search() -> List<Result> -> Select() -> Receipt

Search criteria A -> Search() -> List<Result_A> -> Select() -> Search
criteria B -> Search() -> List<Result_B> -> Select() -> Receipt

Search criteria A : 10/30 -11/5 Boston

BookingSystem.Search()

List<Result_A>: Hotel_1, Hotel_2,...

Challenge

Search criteria -> Search() -> List<Result> -> Select() -> Receipt

Search criteria A -> Search() -> List<Result_A> -> Select() -> Search
criteria B -> Search() -> List<Result_B> -> Select() -> Receipt

Search criteria A : 10/30 -11/5 Boston

BookingSystem.Search()

List<Result_A>: Hotel_1, Hotel_2,...

BookingSystem.Select()

Challenge

Search criteria -> Search() -> List<Result> -> Select() -> Receipt

Search criteria A -> Search() -> List<Result_A> -> Select() -> Search
criteria B -> Search() -> List<Result_B> -> Select() -> Receipt

Search criteria A : 10/30 -11/5 Boston

Search criteria B : 10/30 -11/5

BookingSystem.Search()

List<Result_A>: Hotel_1, Hotel_2,...

BookingSystem.Select()

Challenge

Search criteria -> Search() -> List<Result> -> Select() -> Receipt

Search criteria A -> Search() -> List<Result_A> -> Select() -> Search
criteria B -> Search() -> List<Result_B> -> Select() -> Receipt

Search criteria A : 10/30 -11/5 Boston

Search criteria B : 10/30 -11/5

BookingSystem.Search()

Hotel_1.Search()

List<Result_A>: Hotel_1, Hotel_2,...

BookingSystem.Select()

Challenge

Search criteria -> Search() -> List<Result> -> Select() -> Receipt

Search criteria A -> Search() -> List<Result_A> -> Select() -> Search
criteria B -> Search() -> List<Result_B> -> Select() -> Receipt

Search criteria A : 10/30 -11/5 Boston

Search criteria B : 10/30 -11/5

BookingSystem.Search()

Hotel_1.Search()

List<Result_A>: Hotel_1, Hotel_2,...

List<Result_A>: RoomType_1, ...

BookingSystem.Select()

Challenge

Search criteria -> Search() -> List<Result> -> Select() -> Receipt

Search criteria A -> Search() -> List<Result_A> -> Select() -> Search
criteria B -> Search() -> List<Result_B> -> Select() -> Receipt

Search criteria A : 10/30 -11/5 Boston

Search criteria B : 10/30 -11/5

BookingSystem.Search()

Hotel_1.Search()

List<Result_A>: Hotel_1, Hotel_2,...

List<Result_A>: RoomType_1, ...

BookingSystem.Select()

BookingSystem.Select()

What

What

Search criteria A

List<Result_A>

Search criteria B

List<Result_B>

Receipt

What

Search criteria A


List<Result_A>

Search criteria B





List<Result_B>


Receipt




What

 **Find Deals for Any Season**
From cozy country homes to funky city apartments

Destination, property name or address:





Check-in   Check-out  

Are you traveling for work? ☐ Yes ☐ No 

Rooms  Adults 
Children 

Search

Ashburn, VA

 Fri 10/13	 Sat 10/14
 1 room 	 2 guests 

What are the search criteria for booking system ?

针对本题:

用Start Date, End Date, Group size, City

How

Search()

Select()

Search()

Select()

How

Search()

Select()

Search()

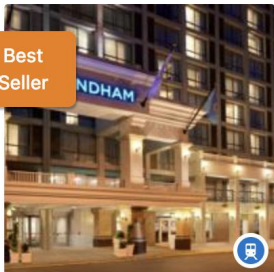
Select()

How

Our Top Picks	Lowest Price First	Stars ▼	Distance From Downtown	Review Score
---------------	--------------------	---------	------------------------	--------------

Recommended for You:

Best Seller






Wyndham Boston Beacon Hill

[West End, Boston, MA – Show on map](#) (2,800 feet from center) – Subway Access

The Museum of Science and TD Bank Garden are located 900 metres from this Boston city centre hotel, featuring completely non-smoking rooms, an on-site restaurant and many modern amenities.

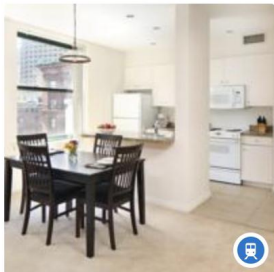
In high demand! Booked 64 times in the last 24 hours

Good 7.8
1,830 reviews

Show prices

\$\$\$



Apartments **Oakwood Boston**

[Financial District, Boston, MA – Show on map](#) (1,000 feet from center) – Subway Access

Located in Financial District of Boston, Oakwood Boston offers a fitness centre and free WiFi. The property is 100 metres from Faneuil Hall, 100 metres from Quincy Market and 4 km from Fenway Park.

Booked 2 times in the last 48 hours

Excellent 8.7
73 reviews

Location 9.5

Show prices

\$\$\$

- 哪些Hotel可以被放进List里?
- 按照什么顺序排序?

How

针对本题：

- 在同样的City就能放进List里
- Hotel在List里的顺序不重要

How

针对本题:

- 在同样的City就能放进List里
- Hotel在List里的顺序不重要

Core object



Request
<ul style="list-style-type: none">- Date startDate- Date endDate

Reservation
<ul style="list-style-type: none">- List<Room> rooms- Date startDate- Date endDate

ReservationRequest
<ul style="list-style-type: none">- Date startDate- Date endDate- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

Hotel
<ul style="list-style-type: none">- List<Room> rooms- List<Reservation> reservations- Map<Room, List<Date>> roomReservations- LRUCache<Request, Map<RoomType, Set<Room>>> cache
<ul style="list-style-type: none">+ Map<RoomType, int> handleSearchRequest(Request r)- boolean isRequestAvailable(Request r, List<Date> dates)+ Reservation makeReservation(ReservationRequest r)+ void cancelReservation(Reservation r)

<<Enumeration>> RoomType
<ul style="list-style-type: none">SINGLEDOUBLE

Room
<ul style="list-style-type: none">- RoomType type- Boolean available
+ boolean isAvailable()

Use cases
Search for available rooms
Make reservation
Cancel reservation

Core object



Request
<ul style="list-style-type: none">- Date startDate- Date endDate

Reservation
<ul style="list-style-type: none">- List<Room> rooms- Date startDate- Date endDate

ReservationRequest
<ul style="list-style-type: none">- Date startDate- Date endDate- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

Hotel
<ul style="list-style-type: none">- List<Room> rooms- List<Reservation> reservations- Map<Room, List<Date>> roomReservations- LRUCache<Request, Map<RoomType, Set<Room>>> cache
<ul style="list-style-type: none">+ Map<RoomType, int> handleSearchRequest(Request r)- boolean isRequestAvailable(Request r, List<Date> dates)+ Reservation makeReservation(ReservationRequest r)+ void cancelReservation(Reservation r)

<<Enumeration>> RoomType
<ul style="list-style-type: none">SINGLEDOUBLE

BookingSystem

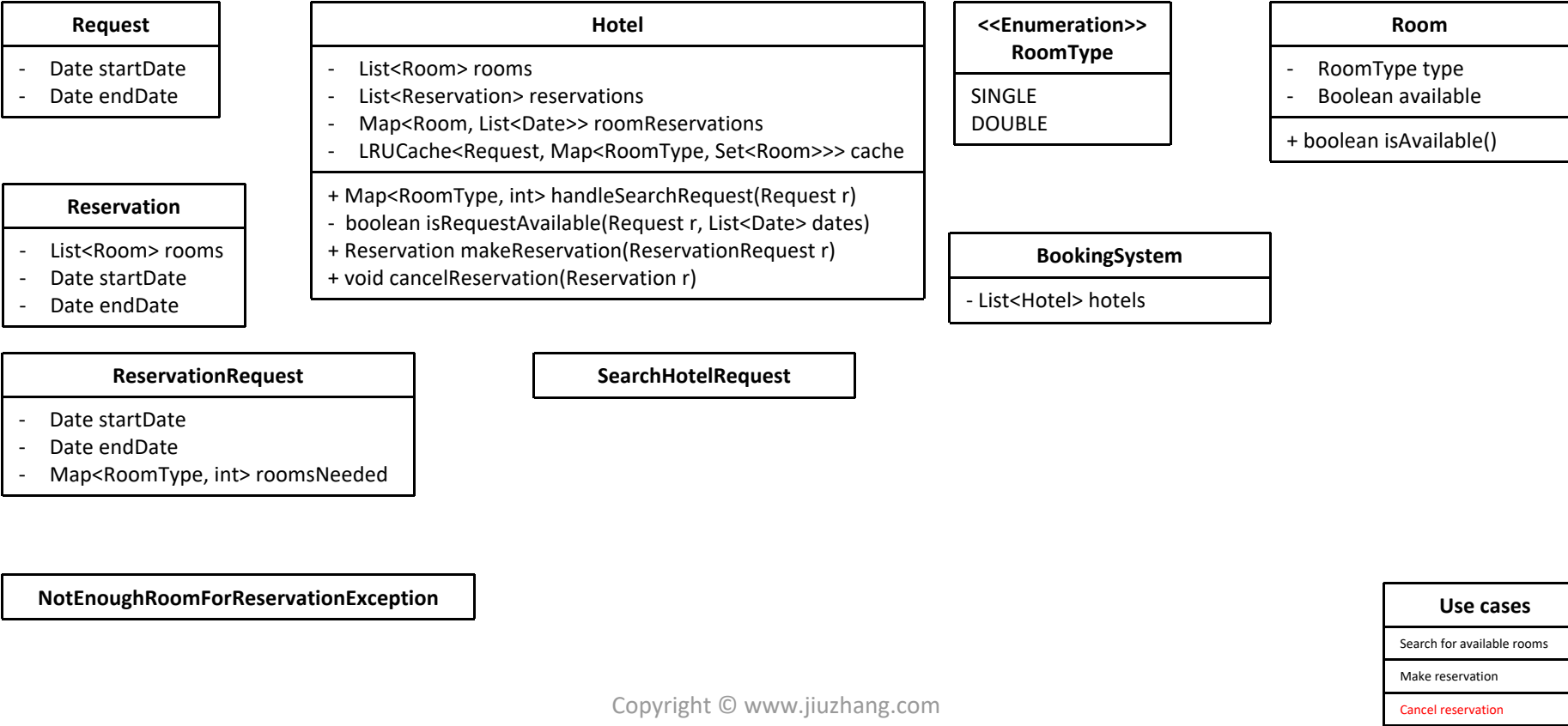
Room
<ul style="list-style-type: none">- RoomType type- Boolean available
+ boolean isAvailable()

Use cases
Search for available rooms
Make reservation
Cancel reservation

九章算法

Use cases
Search for available rooms
Make reservation
Cancel reservation

Core object

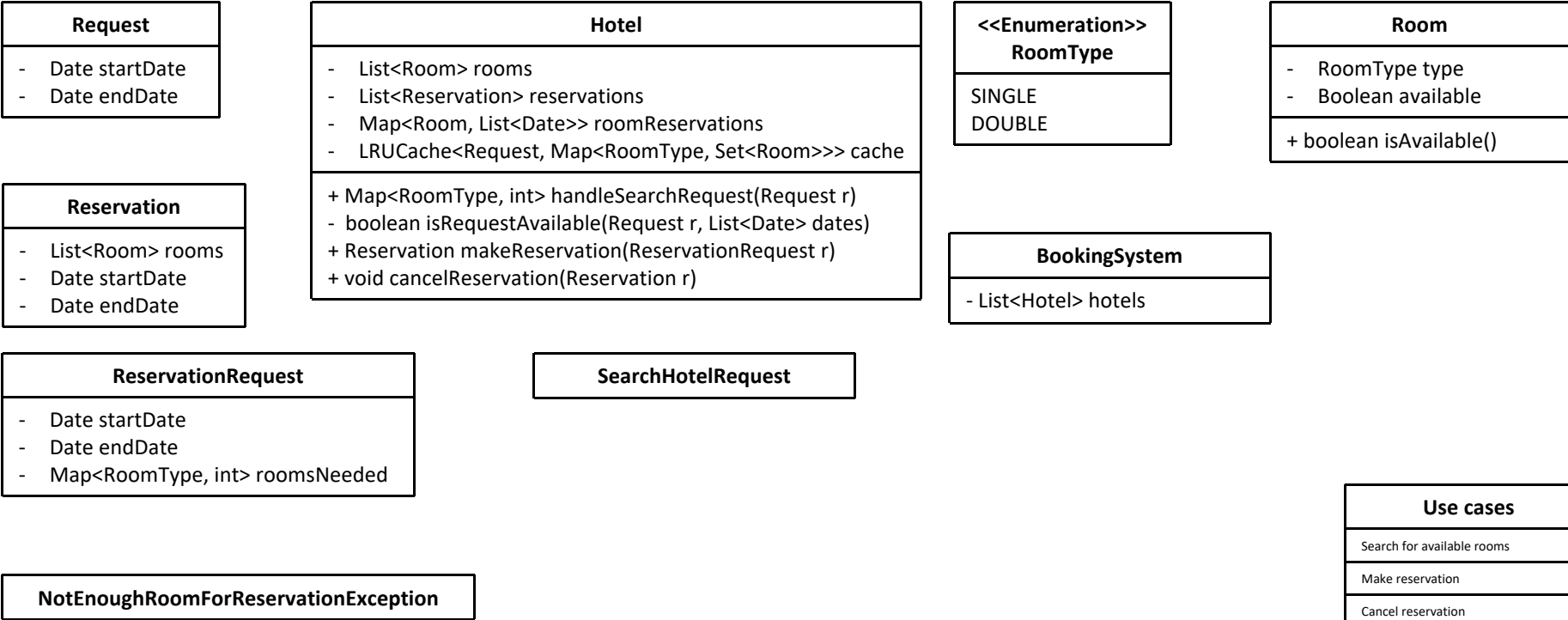


- BookingSystem

- BookingSystem
 - Search for hotels

- BookingSystem
 - Search for hotels
 - Make reservation

- BookingSystem
 - Search for hotels
 - Make reservation
 - Cancel reservation



- Search for hotels
 - Based on request (start date + end date + city + group size)
 - Check for all hotels in this city
 - For such hotels, if this is enough capacity for group size during dates
 - List all hotels that satisfy above criteria

- Search for hotels
- Based on request (start date + end date + city + group size)

Class

Request
- Date startDate
- Date endDate

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
- LRUCache<Request, Map<RoomType, Set<Room>>> cache
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)
+ Reservation makeReservation(ReservationRequest r)
+ void cancelReservation(Reservation r)

<<Enumeration>> RoomType
SINGLE
DOUBLE

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

Reservation
- List<Room> rooms
- Date startDate
- Date endDate

BookingSystem
- List<Hotel> hotels

ReservationRequest
- Date startDate
- Date endDate
- Map<RoomType, int> roomsNeeded

SearchHotelRequest
- Date startDate
- Date endDate
- String city
- Int groupSize

NotEnoughRoomForReservationException

Use cases
Search for available rooms
Make reservation
Cancel reservation
Search for hotels
Make reservation (Booking System)
Cancel reservation (Booking System)

- Search for hotels
 - Based on request (start date + end date + city + group size)
 - Check for all hotels in this city

Class

Request
- Date startDate
- Date endDate

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
- LRUCache<Request, Map<RoomType, Set<Room>>> cache
- String city
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)
+ Reservation makeReservation(ReservationRequest r)
+ void cancelReservation(Reservation r)

<<Enumeration>> RoomType
SINGLE
DOUBLE

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

Reservation
- List<Room> rooms
- Date startDate
- Date endDate

BookingSystem
- List<Hotel> hotels

ReservationRequest
- Date startDate
- Date endDate
- Map<RoomType, int> roomsNeeded

SearchHotelRequest
- Date startDate
- Date endDate
- String city
- Int groupSize

NotEnoughRoomForReservationException

Use cases
Search for available rooms
Make reservation
Cancel reservation
Search for hotels
Make reservation (Booking System)
Cancel reservation (Booking System)

Request
- Date startDate
- Date endDate

Reservation
- List<Room> rooms
- Date startDate
- Date endDate

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
- LRUCache<Request, Map<RoomType, Set<Room>>> cache
- String city
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)
+ Reservation makeReservation(ReservationRequest r)
+ void cancelReservation(Reservation r)
+ boolean isSameCity(string city)

<<Enumeration>> RoomType
SINGLE
DOUBLE

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

BookingSystem
- List<Hotel> hotels

ReservationRequest
- Date startDate
- Date endDate
- Map<RoomType, int> roomsNeeded

SearchHotelRequest
- Date startDate
- Date endDate
- String city
- Int groupSize

NotEnoughRoomForReservationException

Use cases
Search for available rooms
Make reservation
Cancel reservation
Search for hotels
Make reservation (Booking System)
Cancel reservation (Booking System)

- Search for hotels
 - Based on request (start date + end date + city + group size)
 - Check for all hotels in this city
 - For such hotels, if this is enough capacity for group size during dates

Class



Request
- Date startDate
- Date endDate

Reservation
- List<Room> rooms
- Date startDate
- Date endDate

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
- LRUCache<Request, Map<RoomType, Set<Room>>> cache
- String city
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)
+ Reservation makeReservation(ReservationRequest r)
+ void cancelReservation(Reservation r)
+ boolean isSameCity(string city)

ReservationRequest
- Date startDate
- Date endDate
- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

SearchHotelRequest
- Date startDate
- Date endDate
- String city
- Int groupSize

<<Enumeration>> RoomType
SINGLE
DOUBLE
- Int capacity

BookingSystem
- List<Hotel> hotels

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

Use cases
Search for available rooms
Make reservation
Cancel reservation
Search for hotels
Make reservation (Booking System)
Cancel reservation (Booking System)

Class



Request
- Date startDate
- Date endDate

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
- LRUCache<Request, Map<RoomType, Set<Room>>> cache
- String city
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)
+ Reservation makeReservation(ReservationRequest r)
+ void cancelReservation(Reservation r)
+ boolean isSameCity(string city)
+ boolean isValidRequest(SearchHotelRequest r)

<<Enumeration>> RoomType
SINGLE
DOUBLE
- Int capacity

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

Reservation
- List<Room> rooms
- Date startDate
- Date endDate

BookingSystem
- List<Hotel> hotels

ReservationRequest
- Date startDate
- Date endDate
- Map<RoomType, int> roomsNeeded

SearchHotelRequest
- Date startDate
- Date endDate
- String city
- Int groupSize

NotEnoughRoomForReservationException

Use cases
Search for available rooms
Make reservation
Cancel reservation
Search for hotels
Make reservation (Booking System)
Cancel reservation (Booking System)

- Search for hotels
 - Based on request (start date + end date + city + group size)
 - Check for all hotels in this city
 - For such hotels, if this is enough capacity for group size during dates
 - List all hotels that satisfy above criteria

Request
- Date startDate
- Date endDate

Reservation
- List<Room> rooms
- Date startDate
- Date endDate

ReservationRequest
- Date startDate
- Date endDate
- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
- LRUCache<Request, Map<RoomType, Set<Room>>> cache
- String city
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)
+ Reservation makeReservation(ReservationRequest r)
+ void cancelReservation(Reservation r)
+ boolean isSameCity(string city)
+ boolean isValidRequest(SearchHotelRequest r)

SearchHotelRequest
- Date startDate
- Date endDate
- String city
- Int groupSize

<<Enumeration>> RoomType
SINGLE
DOUBLE
- Int capacity

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

BookingSystem
- List<Hotel> hotels
+ List<Hotel> searchHotel(SearchHotelRequest r)

Use cases
Search for available rooms
Make reservation
Cancel reservation
Search for hotels
Make reservation (Booking System)
Cancel reservation (Booking System)

- Make reservation
 - Pick a hotel
 - Send reservation request to that hotel

Request
- Date startDate
- Date endDate

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
- LRUCache<Request, Map<RoomType, Set<Room>>> cache
- String city
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)
+ Reservation makeReservation(ReservationRequest r)
+ void cancelReservation(Reservation r)
+ boolean isSameCity(string city)
+ boolean isValidRequest(SearchHotelRequest r)

Reservation
- List<Room> rooms
- Date startDate
- Date endDate

ReservationRequest
- Date startDate
- Date endDate
- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

SearchHotelRequest
- Date startDate
- Date endDate
- String city
- Int groupSize

<<Enumeration>> RoomType
SINGLE
DOUBLE
- Int capacity

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

BookingSystem
- List<Hotel> hotels
+ List<Hotel> searchHotel(SearchHotelRequest r)
+ Reservation makeReservation(Hotel h, ReservationRequest r)

Use cases
Search for available rooms
Make reservation
Cancel reservation
Search for hotels
Make reservation (Booking System)
Cancel reservation (Booking System)

- Cancel reservation
- Cancel reservation for a hotel

Request
- Date startDate
- Date endDate

Reservation
- Hotel hotel
- List<Room> rooms
- Date startDate
- Date endDate

ReservationRequest
- Date startDate
- Date endDate
- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
- LRUCache<Request, Map<RoomType, Set<Room>>> cache
- String city
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)
+ Reservation makeReservation(ReservationRequest r)
+ void cancelReservation(Reservation r)
+ boolean isSameCity(string city)
+ boolean isValidRequest(SearchHotelRequest r)

SearchHotelRequest
- Date startDate
- Date endDate
- String city
- Int groupSize

<<Enumeration>> RoomType
SINGLE
DOUBLE
- Int capacity

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

BookingSystem
- List<Hotel> hotels
+ List<Hotel> searchHotel(SearchHotelRequest r)
+ Reservation makeReservation(Hotel h, ReservationRequest r)
+ void cancelReservation(Reservation r)

Use cases
Search for available rooms
Make reservation
Cancel reservation
Search for hotels
Make reservation (Booking System)
Cancel reservation (Booking System)

Class – Final view



Request
- Date startDate
- Date endDate

Reservation
- Hotel hotel
- List<Room> rooms
- Date startDate
- Date endDate

ReservationRequest
- Date startDate
- Date endDate
- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
- LRUCache<Request, Map<RoomType, Set<Room>>> cache
- String city
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)
+ Reservation makeReservation(ReservationRequest r)
+ void cancelReservation(Reservation r)
+ boolean isSameCity(string city)
+ boolean isValidRequest(SearchHotelRequest r)

SearchHotelRequest
- Date startDate
- Date endDate
- String city
- Int groupSize

<<Enumeration>> RoomType
SINGLE
DOUBLE
- Int capacity

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

BookingSystem
- List<Hotel> hotels
+ List<Hotel> searchHotel(SearchHotelRequest r)
+ Reservation makeReservation(Hotel h, ReservationRequest r)
+ void cancelReservation(Reservation r)

Use cases
Search for available rooms
Make reservation
Cancel reservation
Search for hotels
Make reservation (Booking System)
Cancel reservation (Booking System)

Challenge



- Can you add payment in your system?

Challenge

- Can you add payment in your system?
 - Need to get price for each reservation
 - Need to take a payment method

Challenge

- Can you add payment in your system?
- Need to get price for each reservation

Request
- Date startDate
- Date endDate

Reservation
- Hotel hotel
- List<Room> rooms
- Date startDate
- Date endDate

ReservationRequest
- Date startDate
- Date endDate
- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
- LRUCache<Request, Map<RoomType, Set<Room>>> cache
- String city
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)
+ Reservation makeReservation(ReservationRequest r)
+ void cancelReservation(Reservation r)
+ boolean isSameCity(string city)
+ boolean isValidRequest(SearchHotelRequest r)

SearchHotelRequest
- Date startDate
- Date endDate
- String city
- Int groupSize

<<Enumeration>> RoomType
SINGLE
DOUBLE
- Int capacity
- Float price

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

BookingSystem
- List<Hotel> hotels
+ List<Hotel> searchHotel(SearchHotelRequest r)
+ Reservation makeReservation(Hotel h, ReservationRequest r)
+ void cancelReservation(Reservation r)

Use cases
Search for available rooms
Make reservation
Cancel reservation
Search for hotels
Make reservation (Booking System)
Cancel reservation (Booking System)

Enum code sample



九章算法

```
public enum RoomType
{
    SINGLE(1, 129),
    DOUBLE(2, 199);

    private int capacity;
    private float price;

    RoomType(int capacity, float price)
    {
        this.capacity = capacity;
        this.price = price;
    }

    public int getCapacity()
    {
        return capacity;
    }

    public float getPrice()
    {
        return price;
    }
}
```

Request
<ul style="list-style-type: none">- Date startDate- Date endDate

Reservation
<ul style="list-style-type: none">- Hotel hotel- List<Room> rooms- Date startDate- Date endDate

ReservationRequest
<ul style="list-style-type: none">- Date startDate- Date endDate- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

Hotel
<ul style="list-style-type: none">- List<Room> rooms- List<Reservation> reservations- Map<Room, List<Date>> roomReservations- LRUCache<Request, Map<RoomType, Set<Room>>> cache- String city
<ul style="list-style-type: none">+ Map<RoomType, int> handleSearchRequest(Request r)- boolean isRequestAvailable(Request r, List<Date> dates)+ Reservation makeReservation(ReservationRequest r)+ void cancelReservation(Reservation r)+ boolean isSameCity(string city)+ boolean isValidRequest(SearchHotelRequest r)

SearchHotelRequest
<ul style="list-style-type: none">- Date startDate- Date endDate- String city- Int groupSize

<<Enumeration>> RoomType
SINGLE DOUBLE
<ul style="list-style-type: none">- Int capacity- Float price

Room
<ul style="list-style-type: none">- RoomType type- Boolean available
+ boolean isAvailable()

BookingSystem
<ul style="list-style-type: none">- List<Hotel> hotels
<ul style="list-style-type: none">+ List<Hotel> searchHotel(SearchHotelRequest r)+ Reservation makeReservation(Hotel h, ReservationRequest r)+ void cancelReservation(Reservation r)+ void payByPaypal (Reservation r)

Use cases
Search for available rooms
Make reservation
Cancel reservation
Search for hotels
Make reservation (Booking System)
Cancel reservation (Booking System)

Request
- Date startDate
- Date endDate

Reservation
- Hotel hotel
- List<Room> rooms
- Date startDate
- Date endDate

ReservationRequest
- Date startDate
- Date endDate
- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
- LRUCache<Request, Map<RoomType, Set<Room>>> cache
- String city
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)
+ Reservation makeReservation(ReservationRequest r)
+ void cancelReservation(Reservation r)
+ boolean isSameCity(string city)
+ boolean isValidRequest(SearchHotelRequest r)

SearchHotelRequest
- Date startDate
- Date endDate
- String city
- Int groupSize

<<Enumeration>> RoomType
SINGLE
DOUBLE
- Int capacity
- Float price

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

BookingSystem
- List<Hotel> hotels
+ List<Hotel> searchHotel(SearchHotelRequest r)
+ Reservation makeReservation(Hotel h, ReservationRequest r)
+ void cancelReservation(Reservation r)
+ void payByPaypal (Reservation r)
+ void payByCreditCard (Reservation r)

Use cases
Search for available rooms
Make reservation
Cancel reservation
Search for hotels
Make reservation (Booking System)
Cancel reservation (Booking System)

Good practice:



- Strategy Pattern

Request
- Date startDate
- Date endDate

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
- LRUCache<Request, Map<RoomType, Set<Room>>> cache
- String city
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)
+ Reservation makeReservation(ReservationRequest r)
+ void cancelReservation(Reservation r)
+ boolean isSameCity(string city)
+ boolean isValidRequest(SearchHotelRequest r)

Reservation
- Hotel hotel
- List<Room> rooms
- Date startDate
- Date endDate

ReservationRequest
- Date startDate
- Date endDate
- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

SearchHotelRequest
- Date startDate
- Date endDate
- String city
- Int groupSize

<<Enumeration>> RoomType
SINGLE
DOUBLE
- Int capacity
- Float price

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

BookingSystem
- List<Hotel> hotels
+ List<Hotel> searchHotel(SearchHotelRequest r)
+ Reservation makeReservation(Hotel h, ReservationRequest r)
+ void cancelReservation(Reservation r)
+ void payByPaypal (Reservation r)
+ void payByCreditCard (Reservation r)

Use cases
Search for available rooms
Make reservation
Cancel reservation
Search for hotels
Make reservation (Booking System)
Cancel reservation (Booking System)

Request
<ul style="list-style-type: none">- Date startDate- Date endDate

Hotel
<ul style="list-style-type: none">- List<Room> rooms- List<Reservation> reservations- Map<Room, List<Date>> roomReservations- LRUCache<Request, Map<RoomType, Set<Room>>> cache- String city
<ul style="list-style-type: none">+ Map<RoomType, int> handleSearchRequest(Request r)- boolean isRequestAvailable(Request r, List<Date> dates)+ Reservation makeReservation(ReservationRequest r)+ void cancelReservation(Reservation r)+ boolean isSameCity(string city)+ boolean isValidRequest(SearchHotelRequest r)

Reservation
<ul style="list-style-type: none">- Hotel hotel- List<Room> rooms- Date startDate- Date endDate

ReservationRequest
<ul style="list-style-type: none">- Date startDate- Date endDate- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

SearchHotelRequest
<ul style="list-style-type: none">- Date startDate- Date endDate- String city- Int groupSize

<<Enumeration>> RoomType
SINGLE DOUBLE
<ul style="list-style-type: none">- Int capacity- Float price

Room
<ul style="list-style-type: none">- RoomType type- Boolean available
+ boolean isAvailable()

BookingSystem
<ul style="list-style-type: none">- List<Hotel> hotels
<ul style="list-style-type: none">+ List<Hotel> searchHotel(SearchHotelRequest r)+ Reservation makeReservation(Hotel h, ReservationRequest r)+ void cancelReservation(Reservation r)

<<interface>> PaymentStrategy

Use cases
Search for available rooms
Make reservation
Cancel reservation
Search for hotels
Make reservation (Booking System)
Cancel reservation (Booking System)

Request
<ul style="list-style-type: none">- Date startDate- Date endDate

Hotel
<ul style="list-style-type: none">- List<Room> rooms- List<Reservation> reservations- Map<Room, List<Date>> roomReservations- LRUCache<Request, Map<RoomType, Set<Room>>> cache- String city
<ul style="list-style-type: none">+ Map<RoomType, int> handleSearchRequest(Request r)- boolean isRequestAvailable(Request r, List<Date> dates)+ Reservation makeReservation(ReservationRequest r)+ void cancelReservation(Reservation r)+ boolean isSameCity(string city)+ boolean isValidRequest(SearchHotelRequest r)

Reservation
<ul style="list-style-type: none">- Hotel hotel- List<Room> rooms- Date startDate- Date endDate

ReservationRequest
<ul style="list-style-type: none">- Date startDate- Date endDate- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

SearchHotelRequest
<ul style="list-style-type: none">- Date startDate- Date endDate- String city- Int groupSize

<<Enumeration>> RoomType
SINGLE DOUBLE
<ul style="list-style-type: none">- Int capacity- Float price

Room
<ul style="list-style-type: none">- RoomType type- Boolean available
+ boolean isAvailable()

BookingSystem
<ul style="list-style-type: none">- List<Hotel> hotels
<ul style="list-style-type: none">+ List<Hotel> searchHotel(SearchHotelRequest r)+ Reservation makeReservation(Hotel h, ReservationRequest r)+ void cancelReservation(Reservation r)

<<interface>> PaymentStrategy
+ void pay(Reservation r)

Use cases
Search for available rooms
Make reservation
Cancel reservation
Search for hotels
Make reservation (Booking System)
Cancel reservation (Booking System)

Request
<ul style="list-style-type: none">- Date startDate- Date endDate

Hotel
<ul style="list-style-type: none">- List<Room> rooms- List<Reservation> reservations- Map<Room, List<Date>> roomReservations- LRUCache<Request, Map<RoomType, Set<Room>>> cache- String city
<ul style="list-style-type: none">+ Map<RoomType, int> handleSearchRequest(Request r)- boolean isRequestAvailable(Request r, List<Date> dates)+ Reservation makeReservation(ReservationRequest r)+ void cancelReservation(Reservation r)+ boolean isSameCity(string city)+ boolean isValidRequest(SearchHotelRequest r)

Reservation
<ul style="list-style-type: none">- Hotel hotel- List<Room> rooms- Date startDate- Date endDate

ReservationRequest
<ul style="list-style-type: none">- Date startDate- Date endDate- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

SearchHotelRequest
<ul style="list-style-type: none">- Date startDate- Date endDate- String city- Int groupSize

<<Enumeration>> RoomType
SINGLE DOUBLE
<ul style="list-style-type: none">- Int capacity- Float price

Room
<ul style="list-style-type: none">- RoomType type- Boolean available
+ boolean isAvailable()

BookingSystem
<ul style="list-style-type: none">- List<Hotel> hotels- PaymentStrategy strategy
<ul style="list-style-type: none">+ List<Hotel> searchHotel(SearchHotelRequest r)+ Reservation makeReservation(Hotel h, ReservationRequest r)+ void cancelReservation(Reservation r)

<<interface>> PaymentStrategy
+ void pay(Reservation r)

Use cases
Search for available rooms
Make reservation
Cancel reservation
Search for hotels
Make reservation (Booking System)
Cancel reservation (Booking System)

Class



Request
- Date startDate
- Date endDate

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
- LRUCache<Request, Map<RoomType, Set<Room>>> cache
- String city
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)
+ Reservation makeReservation(ReservationRequest r)
+ void cancelReservation(Reservation r)
+ boolean isSameCity(string city)
+ boolean isValidRequest(SearchHotelRequest r)

Reservation
- Hotel hotel
- List<Room> rooms
- Date startDate
- Date endDate

ReservationRequest
- Date startDate
- Date endDate
- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

SearchHotelRequest
- Date startDate
- Date endDate
- String city
- Int groupSize

<<Enumeration>> RoomType
SINGLE
DOUBLE
- Int capacity
- Float price

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

BookingSystem
- List<Hotel> hotels
- PaymentStrategy strategy
+ List<Hotel> searchHotel(SearchHotelRequest r)
+ Reservation makeReservation(Hotel h, ReservationRequest r)
+ void cancelReservation(Reservation r)
+ void setStrategy(Payment strategy)

<<interface>> PaymentStrategy
+ void pay(Reservation r)

Use cases
Search for available rooms
Make reservation
Cancel reservation
Search for hotels
Make reservation (Booking System)
Cancel reservation (Booking System)

Class



Request
- Date startDate
- Date endDate

Reservation
- Hotel hotel
- List<Room> rooms
- Date startDate
- Date endDate

ReservationRequest
- Date startDate
- Date endDate
- Map<RoomType, int> roomsNeeded

NotEnoughRoomForReservationException

Hotel
- List<Room> rooms
- List<Reservation> reservations
- Map<Room, List<Date>> roomReservations
- LRUCache<Request, Map<RoomType, Set<Room>>> cache
- String city
+ Map<RoomType, int> handleSearchRequest(Request r)
- boolean isRequestAvailable(Request r, List<Date> dates)
+ Reservation makeReservation(ReservationRequest r)
+ void cancelReservation(Reservation r)
+ boolean isSameCity(string city)
+ boolean isValidRequest(SearchHotelRequest r)

SearchHotelRequest
- Date startDate
- Date endDate
- String city
- Int groupSize

<<Enumeration>> RoomType
SINGLE
DOUBLE
- Int capacity
- Float price

Room
- RoomType type
- Boolean available
+ boolean isAvailable()

BookingSystem
- List<Hotel> hotels
- PaymentStrategy strategy
+ List<Hotel> searchHotel(SearchHotelRequest r)
+ Reservation makeReservation(Hotel h, ReservationRequest r)
+ void cancelReservation(Reservation r)
+ void setStrategy(Payment strategy)
+ void makePayment(Reservation r)

<<interface>> PaymentStrategy
+ void pay(Reservation r)

Use cases
Search for available rooms
Make reservation
Cancel reservation
Search for hotels
Make reservation (Booking System)
Cancel reservation (Booking System)



扫描二维码关注微信/微博
获取最新面试题及权威解答

微信: [ninechapter](#)

知乎专栏: <http://zhuankan.zhihu.com/jiuzhang>

微博: <http://www.weibo.com/ninechapter>

官网: www.jiuzhang.com