

# 面向对象设计入门

第一讲(免费试听)

我是班主任嘎嘎，  
加我领取课程福利哦

讲师：文泰来



加班主任，进班级答疑群  
快速获取面试资料/课程福利

关注公众号，了解大厂资讯

+ 微信study322  
获取全新2022版

## OOD(面向对象)的三大特性

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 三大特性

+ 微信study322  
获取全新2022版

- 封装
- 继承
- 多态

2021免费分享 请勿倒卖  
加微信study322 获取2022版

+ 微信study322  
获取全新2022版

## 封装

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 封装

+ 微信study322  
获取全新2022版

九章算法 e5d850

- Class
- Object

```
11 class Animal {}  
12  
13 Animal a = new Animal();
```

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 封装

+ 微信study322  
获取全新2022版

```
12 class Employee {  
13  
14 }  
15  
16 String name;  
17 float salary;  
18 int level;  
19  
20 void raiseSalary();  
21 void printName();  
22 void promoteLevel();
```

```
12 class Employee {  
13     String name;  
14     float salary;  
15     int level;  
16  
17     void raiseSalary();  
18     void printName();  
19     void promoteLevel();  
20 }
```

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 封装

+ 微信study322  
获取全新2022版

```
12 class Employee {  
13     private String name;  
14     private float salary;  
15     private int level;  
16  
17     public void raiseSalary(float amount);  
18     public void printName();  
19     public void promoteLevel();  
20 }
```

2021免费分享 请勿倒卖  
加微信study322 获取2022版

+ 微信study322  
获取全新2022版

## 继承

2021免费分享 请勿倒卖  
加微信study322 获取2022版



- 继承

加微信study322 获取全新2022版

```
12 // 继承 - Inheritance
13
14 class Animal
15 {
16     public void description()
17     {
18         System.out.println("This is a general animal object");
19     }
20
21     protected String name;
22     public int id;
23     private String privacy;
24 }
25
26 class Dog extends Animal
27 {
28 }
29
30 Dog dog = new Dog();
31 dog.description();
```

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 继承

+ 微信study322

获取全新2022版

```
12 // 继承 - Inheritance
13
14 class Animal
15 {
16     public void description()
17     {
18         System.out.println("This is a general animal object");
19     }
20
21     protected String name;
22     public int id;
23     private String privacy;
24 }
25
26 class Dog extends Animal
27 {
28     // override
29     public void description()
30     {
31         System.out.println("This is a dog object");
32         System.out.println("Name -> " + name);
33         System.out.println("Id -> " + id);
34         System.out.println("Privacy -> " + privacy); // This is WRONG!
35     }
36 }
```

2021 免费分享 请勿倒卖  
加微信study322 获取2022版

- 继承

```
12 // 继承 - Inheritance
13
14 class Animal
15 {
16     public void description()
17     {
18         System.out.println("This is a general animal object");
19     }
20
21     protected String name;
22     public int id;
23     private String privacy;
24 }
25
26 class Dog extends Animal
27 {
28     // override
29     public void description()
30     {
31         System.out.println("This is a dog object");
32     }
33
34     // overload
35     public void description(String type)
36     {
37         System.out.println("This is a " + type);
38     }
39 }
40
41 Dog dog = new Dog();
42 dog.description();
```

- 继承

+ 微信study322  
获取全新2022版

```
12 // 继承 - Inheritance
13 class Animal
14 {
15     public void description()
16     {
17         System.out.println("This is a general animal object");
18     }
19
20     protected String name;
21     public int id;
22     private String privacy;
23 }
24
25 class Dog extends Animal
26 {
27     public void description()
28     {
29         super(); // This will call Base class's description
30     }
31 }
```

2021免费分享 请勿倒卖  
+ 微信study322 获取2022版

- 继承

+ 微信study322  
获取全新2022版

```
12 // 继承 - Inheritance
13 abstract class Animal
14 {
15     public void description()
16     {
17         System.out.println("This is a general animal object");
18     }
19
20     protected String name;
21     public int id;
22     private String privacy;
23 }
24
25 class Dog extends Animal
26 {
27     public void description()
28     {
29         super(); // This will call Base class's description
30     }
31 }
32
33 Animal animal = new Animal(); // This is WRONG
34 Animal animal = new Dog(); // This is CORRECT
```

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 继承

```
12 // 继承 - Inheritance
13 abstract class Animal
14 {
15     public void description()
16     {
17         System.out.println("This is a general animal object");
18     }
19
20     public abstract void makeSound();
21
22     protected String name;
23     public int id;
24     private String privacy;
25 }
26
27 class Dog extends Animal
28 {
29     public void makeSound()
30     {
31         System.out.println("Woof !");
32     }
33 }
34
35 class Cat extends Animal
36 {
37     public void makeSound()
38     {
39         System.out.println("Meow !");
40     }
41 }
42
43 abstract class Mammal extends Animal
44 {
45 }
```

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 继承

```
14 // 继承 - Inheritance
15
16 interface Service
17 {
18     // No constructor
19     public void serve();
20     public void retire();
21 }
22
23 class Dog implements Service
24 {
25     public void serve()
26     {
27         // dog in service
28     }
29
30     public void retire()
31     {
32         // dog retire from service
33     }
34 }
```

- 继承

```
14 // 继承 - Inheritance
15
16 interface Service
17 {
18     // No constructor
19     public void serve();
20     public void retire();
21 }
22
23 interface Circus
24 {
25     public void perform();
26 }
27
28
29 class Dog implements Service, Circus
30 {
31     public void serve()
32     {
33         // dog in service
34     }
35
36     public void retire()
37     {
38         // dog retire from service
39     }
40
41     public void perform()
42     {
43         // dog perform in circus
44     }
45 }
```

+ 微信study322  
获取全新2022版

九章算法 e5d850

2021免费分享 请勿倒卖  
加微信study322 获取2022版



- 继承

```
14 // 继承 - Inheritance
15 interface Service
16 {
17     // No constructor
18     public void serve();
19     public void retire();
20 }
21
22 interface Circus
23 {
24     public void perform();
25 }
26
27 class Animal
28 {
29     public void description()
30     {
31         System.out.println("This is a general animal object");
32     }
33
34     protected String name;
35     public int id;
36     private String privacy;
37 }
38
39 class Dog extends Animal implements Service, Circus
40 {
41     public void serve()
42     {
43         // dog in service
44     }
45
46     public void retire()
47     {
48         // dog retire from service
49     }
50
51     public void perform()
52     {
53         // dog perform in circus
54     }
55 }
```

+ 微信study322  
获取全新2022版

九章算法 e5d850

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 继承

```
14 // 继承 - Inheritance
15
16 interface Service
17 {
18     // No constructor
19     public void serve();
20     public void retire();
21 }
22
23 interface Circus
24 {
25     public void perform();
26 }
27
28
29 class Dog implements Service, Circus
30 {
31     public void serve()
32     {
33         // dog in service
34     }
35
36     public void retire()
37     {
38         // dog retire from service
39     }
40
41     public void perform()
42     {
43         // dog perform in circus
44     }
45 }
```

2021免费分享 请勿倒卖  
加微信study322 获取2022版

+ 微信study322  
获取全新2022版

## 多态

九章算法 e5d850

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 多态

```
14 // 多态 - Polymorphism
15 abstract class Animal
16 {
17     public abstract void makeSound();
18 }
19
20 final class Dog extends Animal
21 {
22     public void makeSound()
23     {
24         System.out.println("Woof !");
25     }
26 }
27
28 class Cat extends Animal
29 {
30     public void makeSound()
31     {
32         System.out.println("Meeow !");
33     }
34 }
35
36 Animal animal1 = new Dog(); // This is CORRECT
37 Animal animal2 = new Cat(); // This is CORRECT
38
39 animal1.makeSound();
40 animal2.makeSound();
```

+ 微信 study322  
获取全新2022版

## Exception

2021免费分享 请勿倒卖  
加微信 study322 获取2022版

- 异常

+ 微信study322  
获取全新2022版

Exception in JAVA:

- Checked Exception (IO Exception, Compile time exception)
- Unchecked Exception (Runtime Exception, NPE)

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 异常

+ 微信study322

获取全新2022版

```
23 // 异常 - Exception
24 class MyException extends Exception
25 {
26     public MyException(String s)
27     {
28         super(s);
29     }
30 }
31
32 public class Testing
33 {
34     public void test()
35     {
36         try
37         {
38             throw new MyException("My exception");
39         }
40         catch (MyException ex)
41         {
42             System.out.println(ex.getMessage()); // you will get "My exception"
43         }
44     }
45 }
```

2021免费分享 请勿倒卖

+ 微信study322 获取2022版

- 异常

```
37 public class Testing
38 {
39     public void test() throws MyException
40     {
41         if(true)
42         {
43             throw new MyException("My exception");
44         }
45     }
46
47     public void test1()
48     {
49         test(); // Wrong, since test throws unchecked exception
50     }
51
52     public void test2() throws MyException
53     {
54         test(); // CORRECT, since test2 throws MyException
55     }
56
57     public void test3()
58     {
59         try
60         {
61             test(); // Also correct, since we wrap test by try-catch block
62         }
63         catch(MyException ex)
64         {
65             System.out.println(ex.getMessage());
66         }
67     }
68 }
```



+ 微信 study322  
获取全新2022版

## Enum

九章算法 e5d850

2021免费分享 请勿倒卖  
加微信 study322 获取2022版

- 枚举变量

+ 微信study322  
获取全新2022版

```
11 // 枚举变量 - Enum
12 public enum TrafficSignal
13 {
14     // Defined in compile time
15     RED, YELLOW, GREEN
16 }
17
18 public class Testing
19 {
20     TrafficSignal signal = TrafficSignal.RED;
21 }
```

九章算法 e5d850

加微信study322

请勿倒卖  
获取2022版

- 我不太能够区分OOD和系统设计；想要能够系统的学习OOD的知识点

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 我不太能够区分OOD和系统设计；想要能够系统的学习OOD的知识点
- 我是在读的学生，还没有面试经验；想要学习如何准备OOD的面试

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 我不太能够区分OOD和系统设计；想要能够系统的学习OOD的知识点
- 我是在读的学生，还没有面试经验；想要学习如何准备OOD的面试
- 我经常被OOD题型的面试题难住，不知道应该从何下手；想要学习解题方法和技巧

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# What will you be learning from lesson 1



九章算法

- 什么是OOD，他和系统设计有什么区别？
- OOD经常在面试中出现吗？它重要吗？
- 怎么样的设计才算是好的设计？
- 如何解答OOD的题目 – 5C解题法
- 这门课有什么要求吗？

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# What will you be learning after lesson 1



九章算法

- 什么是OOD，他和系统设计有什么区别？
- OOD经常在面试中出现吗？它重要吗？
- 怎么样的设计才算是好的设计？
- 如何解答OOD的题目 – 5C解题法
- 这门课有什么要求吗？
  
- OOD的题目类型划分：
  - 管理类 / 预定类 / 实物类 / 游戏类

2021免费分享 请勿倒卖  
加微信study322 获取2022版

+ 微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版



# OOD vs. System Design



	Object Oriented Design	System Design
面试者		
出题目的		
常见公司		
关键字		
例题		

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# OOD vs. System Design



	Object Oriented Design	System Design
面试者	应届毕业生, SDE I -	有经验的面试者, SDE I +
出题目的		
常见公司		
关键字		
例题		

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# OOD vs. System Design



	Object Oriented Design	System Design
面试者	应届毕业生, SDE I -	有经验的面试者, SDE I +
出题目的	OOD常被当做考察面试者综合素质的标准	需要处理大量数据, 提供Service的部门
常见公司		
关键字		
例题		

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# OOD vs. System Design



	Object Oriented Design	System Design
面试者	应届毕业生, SDE I -	有经验的面试者, SDE I +
出题目的	OOD常被当做考察面试者综合素质的标准	需要处理大量数据, 提供Service的部门
常见公司	Amazon, Uber,...	Facebook, Twitter,...
关键字		
例题		

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# OOD vs. System Design



	Object Oriented Design	System Design
面试者	应届毕业生, SDE I -	有经验的面试者, SDE I +
出题目的	OOD常被当做考察面试者综合素质的标准	需要处理大量数据, 提供Service的部门
常见公司	Amazon, Uber,...	Facebook, Twitter,...
关键字	Viability	Scalability
例题		

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# OOD vs. System Design



	Object Oriented Design	System Design
面试者	应届毕业生, SDE I -	有经验的面试者, SDE I +
出题目的	OOD常被当做考察面试者综合素质的标准	需要处理大量数据, 提供Service的部门
常见公司	Amazon, Uber,...	Facebook, Twitter,...
关键字	Viability	Scalability
例题	Design Elevator System	Design Twitter

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 面试频率：
  - Phone interview 低
  - Onsite interview 中高频

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 面试频率：
  - Phone interview 低
  - Onsite interview 中高频
- 面试重要性：
  - 考察作为程序员的基础和大局观
  - 在一些公司拥有一票否决权

2021免费分享 请勿倒卖  
加微信study322 获取2022版



- 面试频率：
  - Phone interview 低
  - Onsite interview 中高频
- 面试重要性：
  - 考察作为程序员的基础和大局观
  - 在一些公司拥有一票否决权
- 高频公司：
  - Amazon, Bloomberg, TripAdvisor, EMC, Uber...

- Coding skill
  - Java entry level, 有基本的Java知识, 了解基本的data structure如Array, List, HashMap等

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Coding skill
  - Java entry level, 有基本的Java知识, 了解基本的data structure如Array, List, HashMap等
- Design pattern
  - 不需要design pattern的基础, 我们将会在课程中讲解如何运用常见的design pattern来为面试加分

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Coding skill
  - Java entry level, 有基本的Java知识, 了解基本的data structure如Array, List, HashMap等
- Design pattern
  - 不需要design pattern的基础, 我们将会在课程中讲解如何运用常见的design pattern来为面试加分
- Time commitment
  - 每节课时2小时, 一周两节课, 2022共五节课10小时
  - Lintcode 做题, 每周一小时

# 如何评判一轮OOD面试



+ 微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- S – Single responsibility principle
- O – Open close principle
- L – Liskov substitution principle
- I – Interface segregation principle
- D – Dependency inversion principle

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Single responsibility principle 单一责任原则

一个类应该有且只有一个去改变他的理由，这意味着一个类应该只有一项工作。

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Single responsibility principle 单一责任原则

一个类应该有且只有一个去改变他的理由，这意味着一个类应该只有一项工作。

```
public class AreaCalculator
{
    private float result;

    public float getResult()
    {
        return this.result;
    }

    public float calculateArea(Triangle t)
    {
        this.result = h * b / 2;
    }
}
```

2021免费分享 请勿倒卖  
加微信study322 获取2022版



- Single responsibility principle 单一责任原则

```
public class AreaCalculator
{
    private float result;

    public float getResult()
    {
        return this.result;
    }

    public float calculateArea(Triangle t)
    {
        this.result = h * b / 2;
    }
}
```

```
public class AreaCalculator
{
    private float result;

    public float getResult()
    {
        return this.result;
    }

    public float calculateArea(Triangle t)
    {
        this.result = h * b / 2;
    }

    public void printResultInJson()
    {
        jsonPrinter.initialize();
        jsonPrinter.print(this.result);
        jsonPrinter.close();
    }
}
```

- Single responsibility principle 单一责任原则

```
public class AreaCalculator
{
    private float result;

    public float getResult()
    {
        return this.result;
    }

    public float calculateArea(Triangle t)
    {
        this.result = h * b / 2;
    }
}

public class Printer
{
    public printInJson(float number)
    {
        jsonPrinter.initialize();
        jsonPrinter.print(this.result);
        jsonPrinter.close();
    }
}
```

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Open close principle 开放封闭原则

对象或实体应该对扩展开放，对修改封闭 (Open to extension, close to modification)。

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Open close principle 开放封闭原则

对象或实体应该对扩展开放，对修改封闭 (Open to extension, close to modification)。

```
public class AreaCalculator
{
    public float calculateArea(Triangle t)
    {
        //calculates the area for triangle
    }

    public float calculateArea(Rectangle r)
    {
        //calculates the area for rectangle
    }
}
```

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Open close principle 开放封闭原则

对象或实体应该对扩展开放，对修改封闭 (Open to extension, close to modification)。

```
public interface Shape
{
    public float getArea();
}

public class Triangle implements Shape
{
    public float getArea()
    {
        return b * h / 2;
    }
}
```

```
public class AreaCalculator
{
    private float result;

    public float getResult()
    {
        return this.result;
    }

    public float calculateArea(Shape s)
    {
        this.result = s.getArea();
    }
}
```

- Liskov substitution principle 里氏替换原则

任何一个子类或派生类应该可以替换它们的基类或父类

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Interface segregation principle 接口分离原则

不应该强迫一个类实现它用不上的接口

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Liskov substitution principle 里氏替换原则

任何一个子类或派生类应该可以替换它们的基类或父类

```
public class Shape
{
    abstract public float calculateVolumn();
    abstract public float calculateArea();
}

public class Rectangle extends Shape
{
    //...
}

public class Cube extends Shape
{
    //...
}
```

2021免费分享 请勿倒卖  
加微信study322 获取2022版



```
public interface Shape
{
    public float calculateVolumn();
    public float calculateArea();
}

public class Rectangle implements Shape
{
    //...
}

public class Cube implements Shape
{
    //...
}
```

+ 微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Dependency inversion principle 依赖反转原则

抽象不应该依赖于具体实现，具体实现应该依赖于抽象

High-level的实体不应该依赖于low-level的实体

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Dependency inversion principle 依赖反转原则

抽象不应该依赖于具体实现，具体实现应该依赖于抽象

High-level的实体不应该依赖于low-level的实体

```
public class AreaCalculator
{
    private float result;
    private Triangle t;

    public float getResult()
    {
        return this.result;
    }

    public float calculateArea()
    {
        this.result = t.h * t.b / 2;
    }
}
```

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Dependency inversion principle 依赖反转原则

抽象不应该依赖于具体实现，具体实现应该依赖于抽象

High-level的实体不应该依赖于low-level的实体

```
public interface Shape
{
    public float getArea();
}

public class Triangle implements Shape
{
    public float getArea()
    {
        return b * h / 2;
    }
}
```

```
public class AreaCalculator
{
    private float result;

    public float getResult()
    {
        return this.result;
    }

    public float calculateArea(Shape s)
    {
        this.result = s.getArea();
    }
}
```

# 面试中应该怎么做？

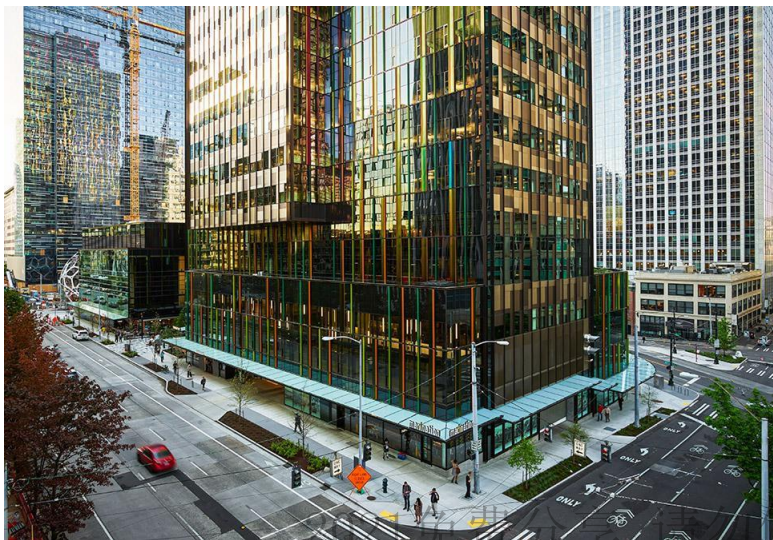


- 实战演练      + 微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Elevator

- Can you design an elevator system for this building?



2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Clarify
- Core objects
- Cases
- Classes
- Correctness

+ 微信study322  
获取全新2022版



2021免费分享 请勿倒卖  
加微信study322 获取2022版

## Clarify

+ 微信study322

说人话：通过和面试官交流，去除题目中的歧义，确定答题范围

## Core objects

说人话：确定题目所涉及的类，以及类之间的映射关系

## Cases

说人话：确定题目中所需要实现的场景和功能

## Classes

说人话：通过类图的方式，具体填充题目中涉及的类

## Correctness

说人话：检查自己的设计，是否满足关键点

2021免费分享 请勿倒卖

+ 微信study322 获取2022版



Example: Glass of water?

加微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版

Example: Glass of water?



2021免费分享 请勿倒卖  
加微信study322 获取2022版

Example: Glass of water?



2021免费分享 请勿倒卖  
加微信study322 获取2022版

Example: Glass of water ?



2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Clarify

Example: Glass of water ?



2021免费分享 请勿倒卖  
加微信study322 获取2022版



- What
- How

+ 微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- What                      + 微信study322  
                                    获取全新2022版

针对题目中的关键字来提问，帮助自己更好的确定答题范围。

\*大多数的关键字为名词，通过名词的属性来考虑

2021免费分享 请勿倒卖  
加微信study322 获取2022版

关键字1: Elevator

+ 微信study322  
获取全新2022版

属性?

2021免费分享 请勿倒卖  
加微信study322 获取2022版



关键字1: Elevator

+ 微信study322

获取全新2022版

- 可能需要考虑获取每辆电梯的目前重量



2021免费分享 请勿倒卖  
加微信study322 获取2022版

## 关键字1: Elevator

+ 微信study322  
获取全新2022版



- 可能需要考虑获取每辆电梯的目前重量

- What's the weight limit of the elevator ?

- Do we need to consider overweight for our elevator system ?

2021免费分享 请勿倒卖  
加微信study322 获取2022版

关键字1: Elevator

+ 微信study322  
获取全新2022版



- 是否需要设计两种类，如果需要它们之间是什么关系？
- 客梯和货梯有什么区别？

2021免费分享 请勿倒卖  
加微信study322 获取2022版

关键字1: Elevator    + 微信study322  
获取全新2022版

针对本题：所有电梯厢均为相同规格

2021免费分享 请勿倒卖  
加微信study322 获取2022版

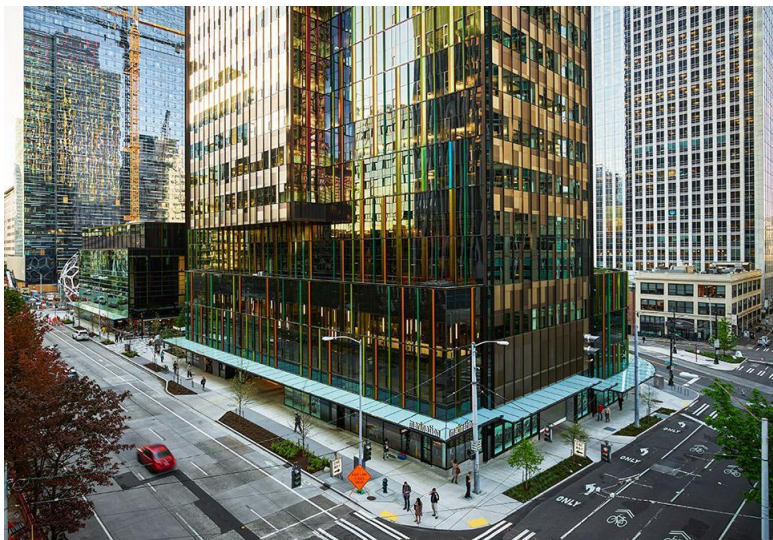
关键字2: Building

+ 微信study322  
获取全新2022版

属性?

2021免费分享 请勿倒卖  
加微信study322 获取2022版

关键字2: Building + 微信study322  
获取全新2022版



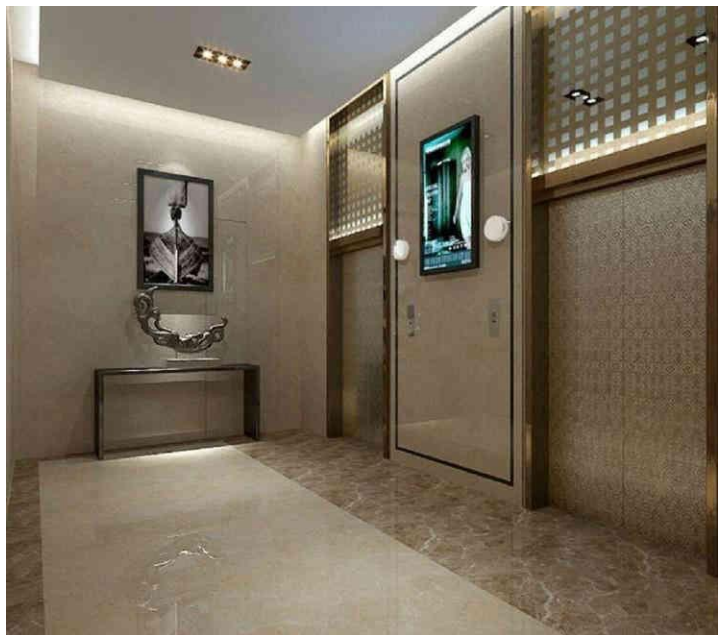
楼有多大/楼有多高/楼内能容纳多少人?

- 通用属性，对于题目帮助不大

2021免费分享 请勿倒卖  
加微信study322 获取2022版

关键字2: Building

+ 微信study322  
获取全新2022版



是否有多处能搭乘的电梯口?

- 当收到一个搭乘电梯的请求时, 有多少电梯能够响应?

2021免费分享 请勿倒卖  
加微信study322 获取2022版

关键字2: Building    + 微信study322  
获取全新2022版

针对本题：每层仅一处能搭乘，所有电梯均可响应

2021免费分享 请勿倒卖  
加微信study322 获取2022版



## - How

+ 微信study322  
获取全新2022版

针对问题主题的规则来提问，帮助自己明确解题方向。

\*此类问题没有标准答案，你可以提出一些解决方法，通过面试官的反应，选择一个你比较有信心（简单）的方案

2021免费分享 请勿倒卖  
加微信study322 获取2022版

电梯有哪些规则？

+ 微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版

+ 微信study322  
获取全新2022版



如何判断电梯是否超重？

2021免费分享 请勿倒卖  
加微信study322 获取2022版

+ 微信study322  
获取全新2022版



如何判断电梯是否超重？

- Passenger class包含重量
- 电梯能够自动感应当前重量

2021免费分享 请勿倒卖  
加微信study322 获取2022版

+ 微信study322  
获取全新2022版



当按下按钮时，哪一台电梯会相应？

- 同方向 > 静止 > 反向
- 一半负责奇数楼层，一半负责偶数楼层
- ...

2021免费分享 请勿倒卖  
加微信study322 获取2022版

+ 微信study322  
获取全新2022版



当电梯在运行时，哪些按键可以响应？

- 是否能按下反向的楼层

2021免费分享 请勿倒卖  
加微信study322 获取2022版

规则：  
+ 微信study322  
获取全新2022版

对于本题：同向 > 静止 > 反向，当运行时不能按下反向的楼层

信息：电梯至少需要三种状态，并且要知道当前在哪一层

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 什么是Core Object
- 为什么要定义Core Object ?
- 如何定义Core Object ?

2021免费分享 请勿倒卖  
加微信study322 获取2022版



- 什么是Core Object

+ 微信study322  
获取全新2022版

为了完成设计，需要哪些类？

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 为什么要定义Core Object ?
  - 这是和面试官初步的纸面contract
  - 承上启下，来自于Clarify的结果，成为Use case的依据
  - 为画类图打下基础

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 如何定义Core Object?  
+ 微信study322  
获取全新2022版
- 以一个Object作为基础，线性思考
- 确定Objects之间的映射关系

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 如何定义Core Object?

ElevatorSystem

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 如何定义Core Object?



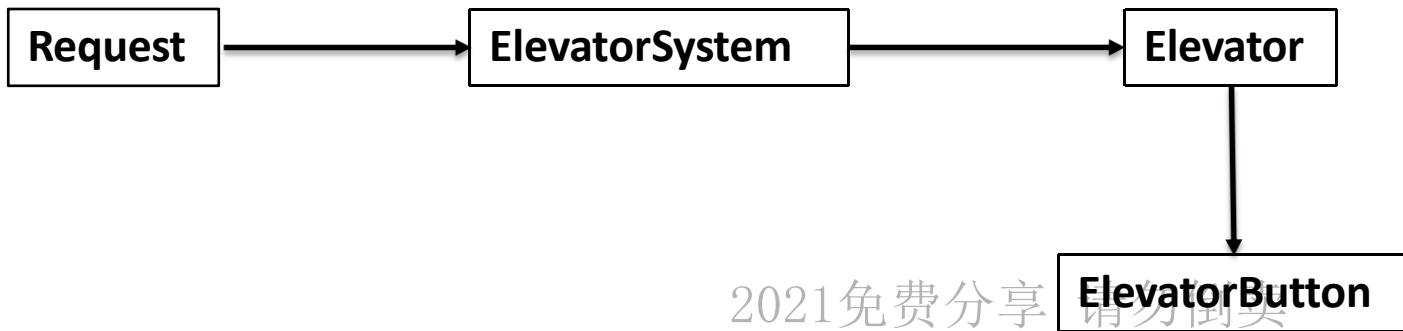
2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 如何定义Core Object?



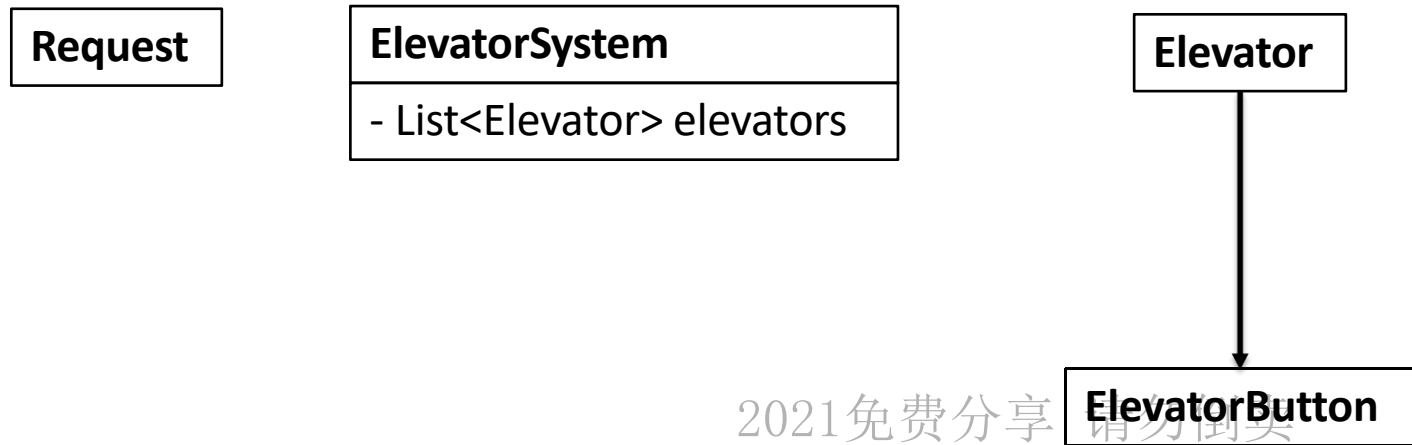
2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 如何定义Core Object?



2021免费分享  
加微信study322 获取2022版

- 如何定义Core Object?



2021免费分享  
加微信study322 获取2022版



- 如何定义Core Object?

**Request**

**ElevatorSystem**

- List<Elevator> elevators

**Elevator**

- List<ElevatorButton> buttons

**ElevatorButton**

2021免费分享  
加微信study322 获取2022版

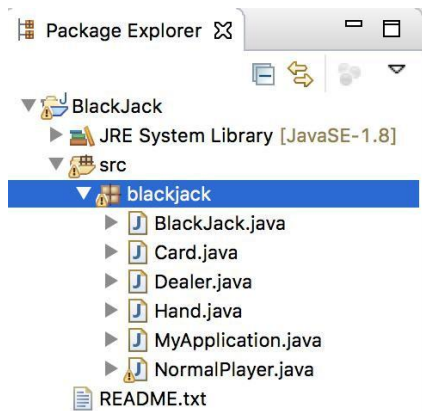
- Access modifier
  - + 微信study322  
获取全新2022版
- package
- public
- private
- protected

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- package
- + 微信study322  
获取全新2022版

如果什么都不声明，变量和函数都是package level visible的，在同一个package内的其他类都可以访问

Example:



```
public class BlackJack
{
    String name = "test";
}

public class Card
{
    public void printName()
    {
        System.out.println(name);
    }
}
```

在类图中，避免使用default的package level access

- public + 微信study322  
获取全新2022版

如果声明为public，变量和函数都是public level visible的，任何其他的类都可以访问

Example:

```
public static void main(String[] arguments)
{
    //...
}
```

在类图中，用“+”表示一个变量或者函数为public

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- private
- + 微信study322  
获取全新2022版

如果声明为**private**，变量和函数都是**class level visible**的，这是所有**access modifier**中限制最多的一个。仅有定义这些变量和函数的类自己可以访问。

**private**也是OOD当中实现封装的重要手段。 Example:

```
public class AreaCalculator()  
{  
    private Logger log;  
}
```

2021免费分享 请勿倒卖  
加微信study322 获取2022版

在类图中，用“-”表示一个变量或者函数为**private**

- protected + 微信study322  
获取全新2022版

如果声明为protected，变量和函数在能被定义他们的类访问的基础上，还能够被该类的子类所访问。

protected也是OOD当中实现继承的重要手段。

Example:

```
class AudioPlayer
{
    protected Speaker speaker;
}

class StreamingAudioPlayer extends AudioPlayer
{
    public void openSpeaker()
    {
        speaker.open();
    }
}
```

在类图中，用“#”表示一个变量或者函数为protected

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 什么是Use case ?
- 为什么要写Use cases ?
- 如何写Use cases ?

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 什么是Use case ?  
+ 微信study322  
获取全新2022版

在你设计的系统中，需要支持哪些功能？

2021免费分享 请勿倒卖  
加微信study322 获取2022版



- 为什么要写Use cases?  
+ 微信study322 获取全新2022版
- 这是你和面试官白纸黑字达成的第二份共识，把你将要实现的功能列在白板上
- 帮助你在解题过程中，理清条例，一个一个Case实现
- 作为检查的标准

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 怎么写Use cases ? 加微信study322  
获取全新2022版
- 利用定义的Core Object, 列举出每个Object对应产生的use case.
- 每个use case只需要先用一句简单的话来描述即可

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- ElevatorSystem    + 微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- ElevatorSystem    + 微信study322  
                             获取全新2022版
- Handle request

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Request      + 微信study322  
                    获取全新2022版

N/A

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Elevator

+ 微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Elevator
  - + 微信study322  
获取全新2022版
- Take external request

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Elevator
  - + 微信study322  
获取全新2022版
- Take external request
- Take internal request

2021免费分享 请勿倒卖  
加微信study322 获取2022版



- Elevator
  - + 微信study322  
获取全新2022版
- Take external request
- Take internal request
- Open gate

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Elevator
  - + 微信study322  
获取全新2022版
- Take external request
- Take internal request
- Open gate
- Close gate

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Elevator
  - + 微信study322  
获取全新2022版
- Take external request
- Take internal request
- Open gate
- Close gate
- Check weight

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Elevator
  - Take external request
  - Take internal request
  - Open gate
  - Close gate
  - Check weight

What about single responsibility principle?

- ElevatorButton      + 微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- ElevatorButton      + 微信study322  
                             获取全新2022版
- Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 什么是类图?
- 为什么要画类图?
- 怎么画类图?

+ 微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Class diagram (类图) 加微信study322  
获取全新2022版

Class Name
Attributes
Functions

2021免费分享 请勿倒卖  
加微信study322 获取2022版



- 为什么要画类图? + 微信study322  
获取全新2022版
- 可交付, **Minimal Viable Product**
- 节省时间, 不容易在**Coding**上挣扎
- 建立在**Use case**上, 和之前的步骤层层递进, 条例清晰, 便于交流和修改
- 如果时间允许/面试官要求, 便于转化成**Code**

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 怎么画类图?      + 微信study322  
                         获取全新2022版
- 遍历你所列出的use cases
- 对于每一个use case, 更加详细的描述这个use case在做什么事情  
    (例如: take external request -> ElevatorSystem takes an external request, and decide to push this request to an appropriate elevator)
- 针对这个描述, 在已有的Core objects里填充进所需要的信息

2021免费分享 请勿倒卖  
加微信study322 获取2022版

Request

**ElevatorSystem**

- List<Elevator> elevators

**Elevator**

- List<ElevatorButton> buttons

**ElevatorButton**

**Use cases**

Handle request

Take external request

Take internal request

Open gate

Close gate

Check weight

Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Use case: Handle request

**ElevatorSystem** takes an **external request**, and decide to push this request to an appropriate **elevator**

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Class

**ExternalRequest**

**ElevatorSystem**

- List<Elevator> elevators

+ void handleRequest(ExternalRequest r)

**Elevator**

- List<ElevatorButton> buttons

**ElevatorButton**

**Use cases**

Handle request

Take external request

Take internal request

Open gate

Close gate

Check weight

Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 如何知道一个函数，是否成功完成任务？

加微信study322  
获取全新2022版

地下一层电梯关闭，这时有人在地下一层按了向上的按钮，会发生什么？

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 如何知道一个函数，是否成功完成任务？

加微信study322  
获取全新2022版

- Use boolean instead of void

成功的话返回true, 否则返回false

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 如何知道一个函数，是否成功完成任务？

加微信study322  
获取全新2022版

- Use boolean instead of void

成功的话返回true, 否则返回false

- 如何知道是什么地方出错？

2021免费分享 请勿倒卖  
加微信study322 获取2022版



- 如何知道一个函数，是否成功完成任务？

加微信study322  
获取全新2022版

- Use exceptions

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Class



**ExternalRequest**

**ElevatorSystem**

- List<Elevator> elevators

+ void handleRequest(ExternalRequest r)

**Elevator**

- List<ElevatorButton> buttons

**InvalidExternalRequestException**

**ElevatorButton**

**Use cases**

**Handle request**

Take external request

Take internal request

Open gate

Close gate

Check weight

Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Use case: Take external request

An **elevator** takes an external **request**, inserts in its stop list.

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Class



## ExternalRequest

- Direction d
- int level

## ElevatorSystem

- List<Elevator> elevators
- + void handleRequest(ExternalRequest r)

## Elevator

- List<ElevatorButton> buttons

## InvalidExternalRequestException

## ElevatorButton

## Use cases

Handle request

Take external request

Take internal request

Open gate

Close gate

Check weight

Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Class



## ExternalRequest

- Direction d
- int level

## ElevatorSystem

- List<Elevator> elevators
- + void handleRequest(ExternalRequest r)

## Elevator

- List<ElevatorButton> buttons

## InvalidExternalRequestException

<<enumeration>>  
Direction

Up Down

## ElevatorButton

## Use cases

Handle request

Take external request

Take internal request

Open gate

Close gate

Check weight

Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Class



## ExternalRequest

- Direction d
- int level

## ElevatorSystem

- List<Elevator> elevators
- + void handleRequest(ExternalRequest r)

## Elevator

- List<ElevatorButton> buttons
- + void handleExternalRequest(ExternalRequest r)

## InvalidExternalRequestException

<<enumeration>>  
Direction

Up Down

## ElevatorButton

## Use cases

Handle request

Take external request

Take internal request

Open gate

Close gate

Check weight

Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Class



## ExternalRequest

- Direction d
- int level

## ElevatorSystem

- List<Elevator> elevators
- + void handleRequest(ExternalRequest r)

## Elevator

- List<ElevatorButton> buttons
- List<Integer> stops
- + void handleExternalRequest(ExternalRequest r)

## InvalidExternalRequestException

<<enumeration>>  
Direction

Up Down

## ElevatorButton

## Use cases

Handle request

Take external request

Take internal request

Open gate

Close gate

Check weight

Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Challenge

- 如果电梯目前在1L，有人按下了5L向上，之后又有人按下了3L向上，电梯会怎样行动？

stops will be {5,3} Expected is: {3,5}

2021免费分享 请勿倒卖  
加微信study322 获取2022版



# Challenge

- 如果电梯目前在1L，有人按下了5L向上，之后又有人按下了3L向上，电梯会怎样行动？

stops will be {5,3} Expected is: {3,5}

Solution1: sort stops every time we add to it.

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Challenge

- 如果电梯目前在1L，有人按下了5L向上，之后又有人按下了3L向上，电梯会怎样行动？

stops will be {5,3} Expected is: {3,5}

Solution2: use priority queue instead of list

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Challenge

- 如果电梯目前在1L，有人按下了5L向上，之后又有人按下了3L向上，紧接着这台电梯又被分配了一个2L向下的request。这台电梯会如何行动？

stops will be {2, 3, 5} Expected is: {3, 5, 2}

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Challenge

- 如果电梯目前在1L，有人按下了5L向上，之后又有人按下了3L向上，紧接着这台电梯又被分配了一个2L向下的request。这台电梯会如何行动？

stops will be {2, 3, 5} Expected is: {3, 5, 2}

Solution: keep 2 lists for different direction

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Class



ExternalRequest
- Direction d - int level

ElevatorSystem
- List<Elevator> elevators  + void handleRequest(ExternalRequest r)

Elevator
- List<ElevatorButton> buttons - List<Integer> upStops - List<Integer> downStops  + void handleExternalRequest(ExternalRequest r)

InvalidExternalRequestException
---------------------------------

<<enumeration>> Direction
Up Down

ElevatorButton
----------------

Use cases
Handle request
Take external request
Take internal request
Open gate
Close gate
Check weight
Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Challenge



- How do you handle an external request?
- What if I want to apply different ways to handle external requests during different time of a day?
- Can you implement it in code?

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Challenge

- How do you handle an external request?

如我们最早和面试官讨论的结果： 同方向

> 静止 > 反向

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Challenge



- What if I want to apply different ways to handle external requests during different time of a day?

2021免费分享 请勿倒卖  
加微信study322 获取2022版



# Challenge

- What if I want to apply different ways to handle external requests during different time of a day?
- Solution 1: if - else

```
public void handleRequest(ExternalRequest r)
{
    if(time == TIME.PEAK)
    {
        // use peak hour handler
    }

    else if(time == TIME.NORMAL)
    {
        // use normal hour handler
    }
}
```

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Challenge

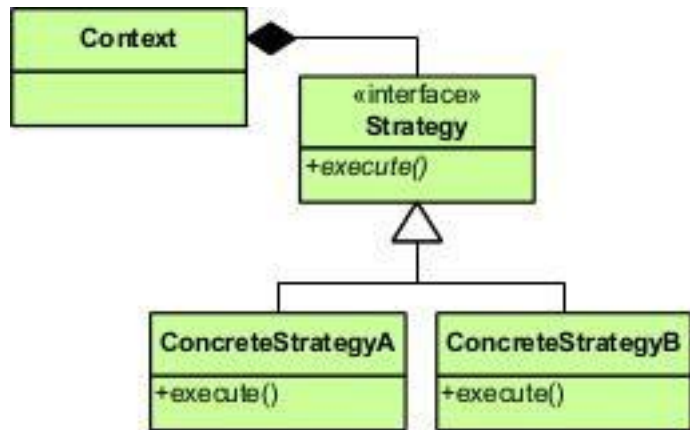
- What if I want to apply different ways to handle external requests during different time of a day?

Solution 2: Strategy design pattern

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Challenge

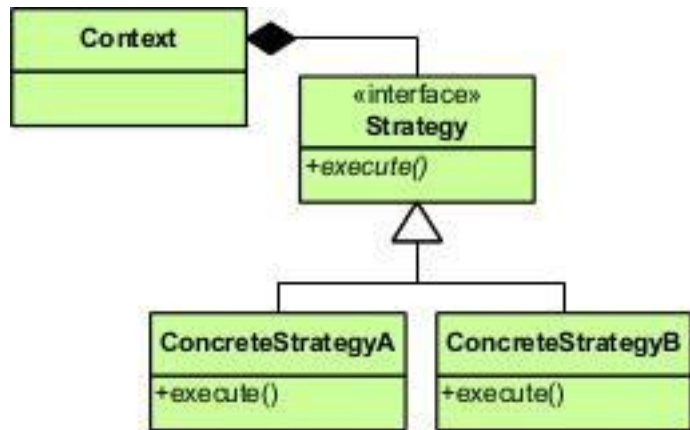
- Strategy Pattern + 微信study322  
获取全新2022版



2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Challenge

- Strategy Pattern + 微信study322  
获取全新2022版

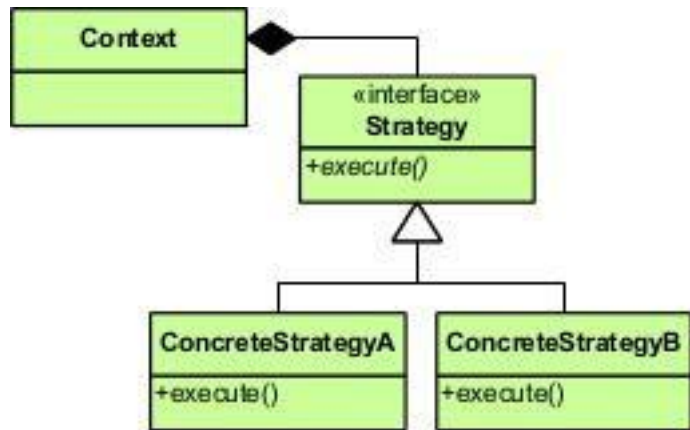


- 封装了多种 算法/策略

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Challenge

- Strategy Pattern + 微信study322  
获取全新2022版

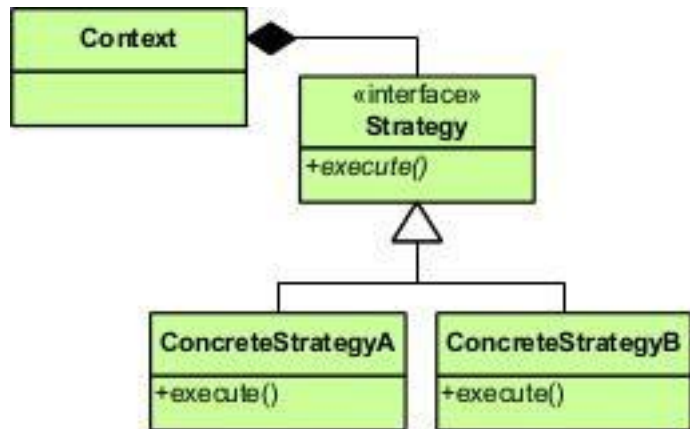


- 封装了多种 算法/策略
- 使得算法/策略之间能够互相替换

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Challenge

- Strategy Pattern



ElevatorSystem
List<Elevator> elevators - HandleRequestStrategy strategy
+ void handleRequest(ExternalRequest r) + void setStrategy(HandleRequestStrategy s)

《interface》 HandleRequestStaregy
+ void handleRequest(Request r, List<Elevator> elevators)

PeakHourHandleRequestStaregy
+ void handleRequest(Request r, List<Elevator> elevators)

NormalHourHandleRequestStaregy
+ void handleRequest(Request r, List<Elevator> elevators)

# Challenge

- Strategy design pattern

加微信study322  
获取全新2022版

```
interface HandleRequestStrategy
{
    public void handleRequest(ExternalRequest request, List<Elevator> elevators);
}

class RandomHandleRequestStrategy implements HandleRequestStrategy
{
    public void handleRequest(ExternalRequest request, List<Elevator> elevators)
    {
        Random rand = new Random();

        int n = rand.nextInt(elevators.size());

        elevators.get(n).handleExternalRequest(request);
    }
}

class AlwaysOneElevatorHandleRequestStrategy implements HandleRequestStrategy
{
    public void handleRequest(ExternalRequest request, List<Elevator> elevators)
    {
        elevators.get(0).handleExternalRequest(request);
    }
}
```

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Challenge

- Strategy design pattern

加微信study322  
获取全新2022版

```
class MyJavaApplication
{
    ElevatorSystem system = new ElevatorSystem();

    system.setStrategy(new RandomHandleRequestStrategy());

    ExternalRequest request = new ExternalRequest(Direction.UP, 3);

    system.handleRequest(request);
}

class ElevatorSystem
{
    private HandleRequestStrategy strategy = new HandleRequestStrategy();
    private List<Elevator> elevators = new ArrayList<>();

    public void setStrategy(HandleRequestStrategy strategy)
    {
        this.strategy = strategy;
    }

    public void handleRequest(ExternalRequest request)
    {
        strategy.handleRequest(request, elevators);
    }
}
```

```
interface HandleRequestStrategy
{
    public void handleRequest(ExternalRequest request, List<Elevator> elevators);
}

class RandomHandleRequestStrategy implements HandleRequestStrategy
{
    public void handleRequest(ExternalRequest request, List<Elevator> elevators)
    {
        Random rand = new Random();

        int n = rand.nextInt(elevators.size());

        elevators.get(n).handleExternalRequest(request);
    }
}

class AlwaysOneElevatorHandleRequestStrategy implements HandleRequestStrategy
{
    public void handleRequest(ExternalRequest request, List<Elevator> elevators)
    {
        elevators.get(0).handleExternalRequest(request);
    }
}
```

2021免费分享，请勿倒卖  
加微信study322 获取2022版



- Use case: Take internal request

加微信study322  
获取全新2022版

An **elevator** takes an internal **request**, determine if it's valid, inserts in its stop list.

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Class



## ExternalRequest

- Direction d
- int level

## InternalRequest

- int level

## ElevatorSystem

- List<Elevator> elevators
- + void handleRequest(ExternalRequest r)

## Elevator

- List<ElevatorButton> buttons
- List<Integer> upStops
- List<Integer> downStops
- + void handleExternalRequest(ExternalRequest r)

## InvalidExternalRequestException

## <<enumeration>> Direction

Up Down

## ElevatorButton

## Use cases

Handle request

Take external request

Take internal request

Open gate

Close gate

Check weight

Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Class



ExternalRequest
- Direction d - int level

InternalRequest
- int level

ElevatorButton
----------------

ElevatorSystem
- List<Elevator> elevators
+ void handleRequest(ExternalRequest r)

Elevator
- List<ElevatorButton> buttons - List<Integer> upStops - List<Integer> downStops
+ void handleExternalRequest(ExternalRequest r) + void handleInternalRequest(InternalRequest r)

InvalidExternalRequestException
---------------------------------

<<enumeration>> Direction
Up Down

Use cases
Handle request
Take external request
Take internal request
Open gate
Close gate
Check weight
Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Class



## ExternalRequest

- Direction d
- int level

## InternalRequest

- int level

## ElevatorSystem

- List<Elevator> elevators
- + void handleRequest(ExternalRequest r)

## Elevator

- List<ElevatorButton> buttons
- List<Integer> upStops
- List<Integer> downStops
- + void handleExternalRequest(ExternalRequest r)
- + void handleInternalRequest(InternalRequest r)
- boolean isValidRequest(InternalRequest r)

## InvalidExternalRequestException

## <<enumeration>> Direction

Up Down

## ElevatorButton

## Use cases

Handle request

Take external request

Take internal request

Open gate

Close gate

Check weight

Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Challenge



- 如何判断一个Internal request 是否为Valid?

加微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Challenge

- 如何判断一个Internal request 是否为Valid?

Solution:

If elevator going up

requested level lower than current level

invalid

If elevator going down

requested level higher than current level

invalid

# Challenge

- 如何判断一个 **Internal request** 是否为 **Valid**?

Solution:

If elevator **going up**

requested level lower than **current level**

invalid

If elevator **going down**

requested level higher than **current level**

invalid

# Class



ExternalRequest
- Direction d - int level

InternalRequest
- int level

ElevatorButton
----------------

ElevatorSystem
- List<Elevator> elevators
+ void handleRequest(ExternalRequest r)

Elevator
- List<ElevatorButton> buttons - List<Integer> upStops - List<Integer> downStops - int currentLevel
+ void handleExternalRequest(ExternalRequest r) + void handleInternalRequest(InternalRequest r) - boolean isRequestValid(InternalRequest r)

InvalidExternalRequestException
---------------------------------

<<enumeration>> Direction
Up Down

Use cases
Handle request
Take external request
Take internal request
Open gate
Close gate
Check weight
Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版



# Class



ExternalRequest
- Direction d - int level

InternalRequest
- int level

ElevatorButton
----------------

ElevatorSystem
- List<Elevator> elevators
+ void handleRequest(ExternalRequest r)

Elevator
- List<ElevatorButton> buttons - List<Integer> upStops - List<Integer> downStops - int currentLevel - Status status
+ void handleExternalRequest(ExternalRequest r) + void handleInternalRequest(InternalRequest r) - boolean isValidRequest(InternalRequest r)

InvalidExternalRequestException
---------------------------------

<<enumeration>> Direction
Up Down

<<enumeration>> Status
Up Down Idle

Use cases
Handle request
Take external request
Take internal request
Open gate
Close gate
Check weight
Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Use case: Open gate 加微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Use case: Open gate
- 加微信study322  
获取全新2022版

并行 VS 串行

单线程 VS 多线程

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Use case: Open gate

加微信study322  
获取全新2022版

单线程:

{3, 5, 2} -> {5, 2} -> {2} -> {}

(1, Up) -> Open gate -> (3, Up) -> Close gate -> (3, Up) -> Open Gate ->  
(5, Up) -> Close gate -> (5, Down) -> Open gate -> (2, Down) -> Close  
Gate -> (2, Idle)

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Use case: Open gate 加微信study322  
获取全新2022版

多线程:

{3, 5, 2} -> {5, 2} -> {2} -> {} Critical Data

```
public class Elevator implements Runnable
{
    @Override
    public void run()
    {
        while(true)
        {
            if(thereIsSomethingLeftInStop())
            {
                operating();
            }
            else
            {
                Thread.sleep();
            }
        }
    }
}
```

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Class



## ExternalRequest

- Direction d
- int level

## InternalRequest

- int level

## ElevatorButton

## ElevatorSystem

- List<Elevator> elevators
- + void handleRequest(ExternalRequest r)

## Elevator

- List<ElevatorButton> buttons
- List<Integer> upStops
- List<Integer> downStops
- int currentLevel
- Status status
- boolean gateOpen
- + void handleExternalRequest(ExternalRequest r)
- + void handleInternalRequest(InternalRequest r)
- + void openGate()
- boolean isRequestValid(InternalRequest r)

## InvalidExternalRequestException

<<enumeration>>  
Direction

Up Down

<<enumeration>>  
Status

Up Down Idle

## Use cases

Handle request

Take external request

Take internal request

Open gate

Close gate

Check weight

Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Use case: Close gate

## An **elevator**

checks if overweight; close the door;  
then check stops corresponds to current status;  
if no stops left, check the reserve direction stops;  
change status to reserve direction or idle.

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Use case: check **weight**

An **elevator** checks its **current weight** and compare with **limit** to see if overweight

2021免费分享 请勿倒卖  
加微信study322 获取2022版



# Class



## ExternalRequest

- Direction d
- int level

## InternalRequest

- int level

## ElevatorButton

## ElevatorSystem

- List<Elevator> elevators
- + void handleRequest(ExternalRequest r)

## Elevator

- List<ElevatorButton> buttons
- List<Integer> upStops
- List<Integer> downStops
- int currentLevel
- Status status
- boolean gateOpen
- + void handleExternalRequest(ExternalRequest r)
- + void handleInternalRequest(InternalRequest r)
- + void openGate()
- float getCurrentWeight()
- boolean isValidRequest(InternalRequest r)

## InvalidExternalRequestException

<<enumeration>>  
Direction

Up Down

<<enumeration>>  
Status

Up Down Idle

## Use cases

Handle request

Take external request

Take internal request

Open gate

Close gate

Check weight

Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Class



## ExternalRequest

- Direction d
- int level

## InternalRequest

- int level

## ElevatorButton

## ElevatorSystem

- List<Elevator> elevators
- + void handleRequest(ExternalRequest r)

## Elevator

- List<ElevatorButton> buttons
- List<Integer> upStops
- List<Integer> downStops
- int currentLevel
- Status status
- boolean gateOpen
- float weightLimit
- + void handleExternalRequest(ExternalRequest r)
- + void handleInternalRequest(InternalRequest r)
- + void openGate()
- float getCurrentWeight()
- boolean isValidRequest(InternalRequest r)

## InvalidExternalRequestException

<<enumeration>>  
Direction

Up Down

<<enumeration>>  
Status

Up Down Idle

## Use cases

Handle request

Take external request

Take internal request

Open gate

Close gate

Check weight

Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Class



## ExternalRequest

- Direction d
- int level

## InternalRequest

- int level

## ElevatorButton

## ElevatorSystem

- List<Elevator> elevators
- + void handleRequest(ExternalRequest r)

## Elevator

- List<ElevatorButton> buttons
- List<Integer> upStops
- List<Integer> downStops
- int currentLevel
- Status status
- boolean gateOpen
- float weightLimit
- + void handleExternalRequest(ExternalRequest r)
- + void handleInternalRequest(InternalRequest r)
- + void openGate()
- float getCurrentWeight()
- boolean isValidRequest(InternalRequest r)

## InvalidExternalRequestException

## OverWeightException

<<enumeration>>  
Direction

Up Down

<<enumeration>>  
Status

Up Down Idle

## Use cases

Handle request

Take external request

Take internal request

Open gate

Close gate

Check weight

Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Class



## ExternalRequest

- Direction d
- int level

## InternalRequest

- int level

## ElevatorButton

## ElevatorSystem

- List<Elevator> elevators
- + void handleRequest(ExternalRequest r)

## Elevator

- List<ElevatorButton> buttons
- List<Integer> upStops
- List<Integer> downStops
- int currentLevel
- Status status
- boolean gateOpen
- float weightLimit
- + void handleExternalRequest(ExternalRequest r)
- + void handleInternalRequest(InternalRequest r)
- + void openGate()
- + void closeGate()
- float getCurrentWeight()
- boolean isRequestValid(InternalRequest r)

## InvalidExternalRequestException

## OverWeightException

<<enumeration>>  
Direction

Up Down

<<enumeration>>  
Status

Up Down Idle

## Use cases

Handle request

Take external request

Take internal request

Open gate

Close gate

Check weight

Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- Use case: press **button**

A **button** inside elevator is pressed, will generate an **internal request** and send to the **elevator**.

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Class



## ExternalRequest

- Direction d
- int level

## InternalRequest

- int level

## ElevatorButton

- int level

## ElevatorSystem

- List<Elevator> elevators
- + void handleRequest(ExternalRequest r)

## Elevator

- List<ElevatorButton> buttons
- List<Integer> upStops
- List<Integer> downStops
- int currentLevel
- Status status
- boolean gateOpen
- float weightLimit
- + void handleExternalRequest(ExternalRequest r)
- + void handleInternalRequest(InternalRequest r)
- + void openGate()
- + void closeGate()
- float getCurrentWeight()
- boolean isRequestValid(InternalRequest r)

## InvalidExternalRequestException

## OverWeightException

<<enumeration>>  
Direction

Up Down

<<enumeration>>  
Status

Up Down Idle

## Use cases

Handle request

Take external request

Take internal request

Open gate

Close gate

Check weight

Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Class



## ExternalRequest

- Direction d
- int level

## InternalRequest

- int level

## ElevatorSystem

- List<Elevator> elevators
- + void handleRequest(ExternalRequest r)

## ElevatorButton

- int level
- + boolean pressButton()

## Elevator

- List<ElevatorButton> buttons
- List<Integer> upStops
- List<Integer> downStops
- int currentLevel
- Status status
- boolean gateOpen
- float weightLimit
- + void handleExternalRequest(ExternalRequest r)
- + void handleInternalRequest(InternalRequest r)
- + void openGate()
- + void closeGate()
- float getCurrentWeight()
- boolean isRequestValid(InternalRequest r)

## InvalidExternalRequestException

## OverWeightException

<<enumeration>>  
Direction

Up Down

<<enumeration>>  
Status

Up Down Idle

## Use cases

Handle request

Take external request

Take internal request

Open gate

Close gate

Check weight

Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Class



## ExternalRequest

- Direction d
- int level

## InternalRequest

- int level

## ElevatorSystem

- List<Elevator> elevators
- + void handleRequest(ExternalRequest r)

## ElevatorButton

- int level
- Elevator elevator

+ InternalRequest pressButton()

## Elevator

- List<ElevatorButton> buttons
- List<Integer> upStops
- List<Integer> downStops
- int currentLevel
- Status status
- boolean gateOpen
- float weightLimit

- + void handleExternalRequest(ExternalRequest r)
- + void handleInternalRequest(InternalRequest r)
- + void openGate()
- + void closeGate()
- float getCurrentWeight()
- boolean isRequestValid(InternalRequest r)

## InvalidExternalRequestException

## OverWeightException

<<enumeration>>  
Direction

Up Down

<<enumeration>>  
Status

Up Down Idle

## Use cases

Handle request

Take external request

Take internal request

Open gate

Close gate

Check weight

Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版



# Class – Final view



## ExternalRequest

- Direction d
- int level

## InternalRequest

- int level

## ElevatorSystem

- List<Elevator> elevators
- + void handleRequest(ExternalRequest r)

## ElevatorButton

- int level
- Elevator elevator
- + InternalRequest pressButton()

## Elevator

- List<ElevatorButton> buttons
- List<Integer> upStops
- List<Integer> downStops
- int currentLevel
- Status status
- boolean gateOpen
- float weightLimit
- + void handleExternalRequest(ExternalRequest r)
- + void handleInternalRequest(InternalRequest r)
- + void openGate()
- + void closeGate()
- float getCurrentWeight()
- boolean isRequestValid(InternalRequest r)

## InvalidExternalRequestException

## OverWeightException

<<enumeration>>  
Direction

Up Down

<<enumeration>>  
Status

Up Down Idle

## Use cases

Handle request

Take external request

Take internal request

Open gate

Close gate

Check weight

Press button

2021免费分享 请勿倒卖  
加微信study322 获取2022版

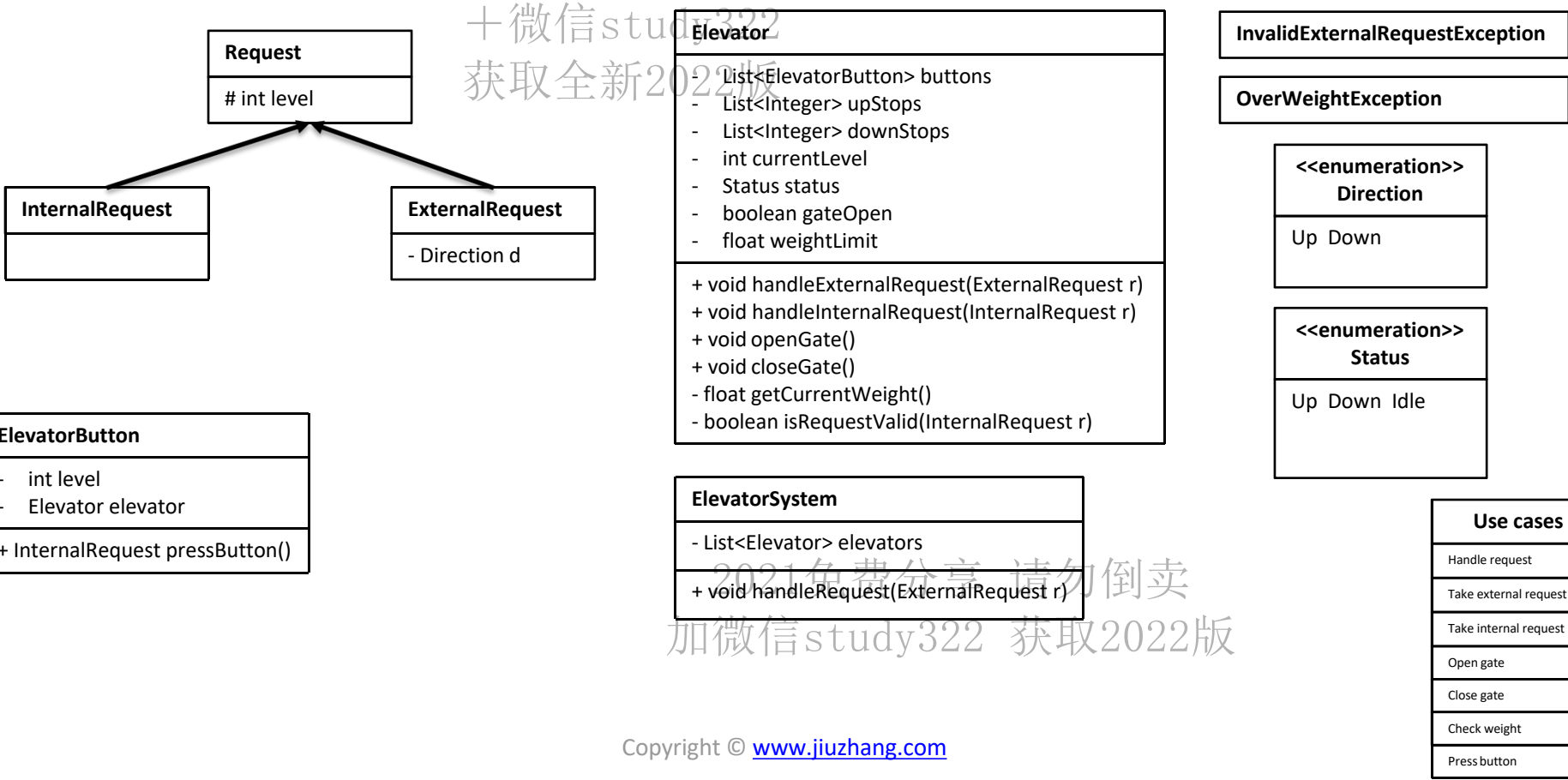
- 从以下几方面检查：  
+ 微信study322  
+ 获取全新2022版
- Validate use cases (检查是否支持所有的use case)
- Follow good practice (面试当中的加分项，展现一个程序员的经验)
- S.O.L.I.D
- Design pattern

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 继承
- + 微信study322  
获取全新2022版

检查你的设计中，是否有重复的类，可以采用继承的方式来表现

2021免费分享 请勿倒卖  
加微信study322 获取2022版



- 什么是OOD

+ 微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 什么是OOD
- SOLID原则

+ 微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 什么是OOD
- SOLID原则
- 5C 解题法

+ 微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 什么是OOD
- SOLID原则
- 5C 解题法
- Good practice: Access modifier

+ 微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版



- 什么是OOD
- SOLID原则
- 5C 解题法
- Good practice: Access modifier
- Good practice: Exception

+ 微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版

- 什么是OOD
- SOLID原则
- 5C 解题法
- Good practice: Access modifier
- Good practice: Exception
- Design pattern: Strategy

+ 微信study322  
获取全新2022版

2021免费分享 请勿倒卖  
加微信study322 获取2022版

+ 微信study322  
获取全新2022版

## 第2章

## 管理类面向对象设计 OOD for Management System

### 本节大纲

- 管理类 OOD 面试题型特点分析
- 实战OOD面试真题：
  - 停车场问题 Parking lot
  - 餐厅管理问题 Restaurant
- 设计模式讲解 Design Pattern: Singleton

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Next...

+ 微信study322  
获取全新2022版

## 第3章

## 预定类面向对象设计 OOD for Reservation System

### 本节大纲

- 预定类面试题型特点分析
- 实战面试真题：
  - 酒店预订系统设计 Hotel Reservation
  - 航空机票预订系统设计 Airline Ticket Reservation

2021免费分享 请勿倒卖  
加微信study322 获取2022版

+ 微信study322  
获取全新2022版

## 第4章

## 实物类面向对象设计 OOD for Real Life Object

### 本节大纲

- 实物类面试题型特点分析
- 实战面试真题：
  - Vending machine
  - Juke box
- 设计模式讲解 Design Pattern: Factory
- 设计模式讲解 Design Pattern: Adaptor

2021免费分享 请勿倒卖  
加微信study322 获取2022版

# Next...

+ 微信study322  
获取全新2022版

## 第5章

## 游戏棋牌类面向对象设计 OOD for Games

### 本节大纲

- 棋牌游戏类面试题特点分析
- 棋牌游戏类面试题特殊技巧讲解
- 实战面试真题：
  - Black Jack
  - Chinese chess
- 课程总结及面试技巧点拨

2021免费分享 请勿倒卖  
加微信study322 获取2022版

+ 微信study322  
获取全新2022版



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](http://www.jiuzhang.com)

2021免费分享 请勿倒卖  
加微信study322 获取2022版

+ 微信study322

**已参加**此次分享活动的同学请私聊人工九妹发送  
福利口令：**面向对象你和我**

【口令有效期：美西时间6月1日-6月8日】

**未参加**此次分享活动的同学请私聊人工九妹发送：  
福利 二字，即可参与此次活动。

福利内容：

1. 【价值 \$300】《面向对象设计专题班》抵价  
券

2. 【价值 ¥249】LintCode VIP 14天

3. 【价值 ¥249】硅谷求职精品讲座VIP 7天

4. ood 推荐书籍系列

5. ood 面试题汇总及参考答案

6. Google Facebook Amazon 秋招求职大礼包



2021免费分享 请勿倒卖 九妹微信号  
加微信study322 获取2022版



新增了一个班主任      + 微信study322  
获取全新2022版

- 督学
- 第二节课开课前2天开班仪式



2021免费分享 请勿倒卖 九妹微信号  
加微信study322 获取2022版