# 建造者模式

1. **基本概念**

### **1、定义：**

建造者（BuilderMethod）模式的定义：建造者模式是当创建复杂对象的算法应该应该独立于该对象的组成部分以及他们的装配方式的适用的模式。

简单来说，建造者方法主要用来创建一些复杂的对象，这些对象的内部构建间的建造顺序通常是稳定的，但对象内部的构建通常面临着复杂的变化。

### 2、组成环节：

指挥类（Director）：是为了用来根据用户的需求来进行对象的实现。

抽象建造类（Builder）：是为了创建一个具体建造类对象的各个部分指定的抽象接口。

具体建造类（ConcreteBuilder）：具体创造者，是抽象创造类的子类，实现Builder接口，构造和装配各个配件，具体的对象。

## 二、go语言中实现

##### 1、首先定义一个Builder的建造类

type builder interface {  
 StepOne()  
 StepTwo()  
 StepLast()  
}

##### 2、首先定义一个Director的指挥类

type director interface {  
 build(b builder)  
}  
  
type directors struct{}

##### 3、为指挥类进行build方法的实现

func (d directors) build(b builder) {  
 b.StepOne()  
 b.StepTwo()  
 b.StepLast()  
}

##### 4、定义两个ConcreteBuild的具体方法类

type concreteBuilderA struct {  
 isStepOneFinish bool  
 isStepTwoFinish bool  
}

type concreteBuilderB struct {  
 isStepOneFinish bool  
 isStepTwoFinish bool  
}

##### 4、为这两个ConcreteBuild的具体方法类的方法进行实现

func (cb \*concreteBuilderB) StepOne() {  
 cb.isStepOneFinish = true  
 fmt.Println("concreteBuilderB step one is finish.")  
}  
  
func (cb \*concreteBuilderB) StepTwo() {  
 cb.isStepTwoFinish = true  
 fmt.Println("concreteBuilderB step two is finish.")  
}  
  
func (cb \*concreteBuilderB) StepLast() {  
 if cb.isStepTwoFinish && cb.isStepOneFinish {  
 fmt.Println("concreteBuilderB is finish!")  
 } else {  
 fmt.Println("concreteBuilderB build unsuccess.")  
 }  
}

func (ca \*concreteBuilderA) StepOne() {  
 ca.isStepOneFinish = true  
 fmt.Println("concreteBuilderA step one is finish.")  
}  
  
func (ca \*concreteBuilderA) StepTwo() {  
 ca.isStepTwoFinish = true  
 fmt.Println("concreteBuilderA step two is finish.")  
}  
  
func (ca \*concreteBuilderA) StepLast() {  
 if ca.isStepTwoFinish && ca.isStepOneFinish {  
 fmt.Println("concreteBuilderA is finish!")  
 } else {  
 fmt.Println("concreteBuilderA build unsuccess.")  
 }  
}

##### 5、通过main函数实现

func main() {  
 b := directors{}  
 ca := concreteBuilderA{}  
 b.build(&ca)  
 cb := concreteBuilderB{}  
 b.build(&cb)  
}

##### 6、输出结果

**concreteBuilderA step one is finish.**

**concreteBuilderA step two is finish.**

**concreteBuilderA is finish!**

**concreteBuilderB step one is finish.**

**concreteBuilderB step two is finish.**

**concreteBuilderB is finish!**