SOFTWARE ENGINEERING INSTITUTE
华东师范大学软件学院

Object oriented Analysis &Design
面向对象分析与设计

Lecture_17 Design Pattern-Command

主讲: 陈小红

日期:

GoF设计模式的分类

（1）Creational（创建型）5个

（2）Structural（结构型）7个

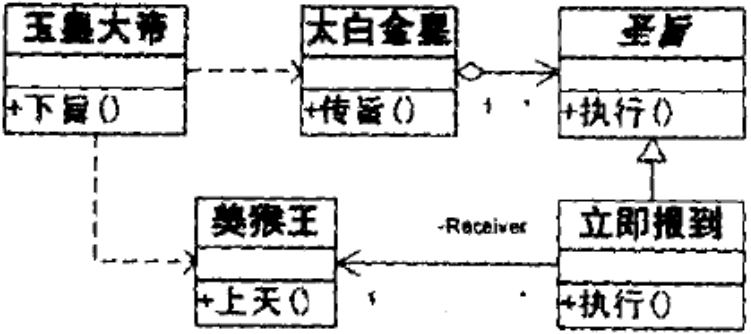
（3）Behavioral（行为型）11个

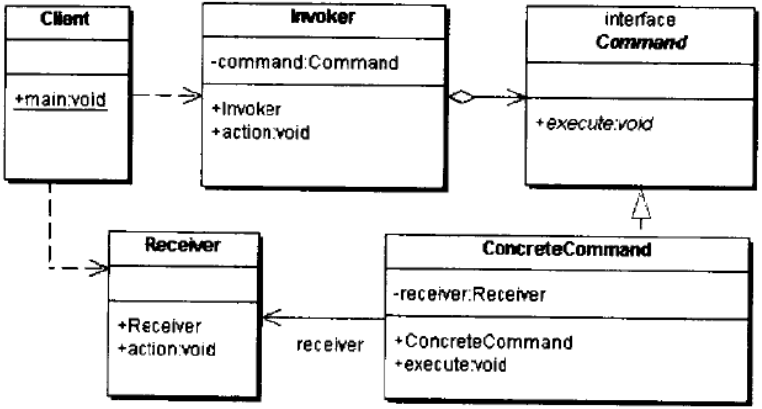
	创建型	结构型	行为型
类	Factory Method	Adapter_Class	Interpreter Template Method
对象	Abstract Factory Builder Prototype Singleton	Adapter_Object Bridge Composite Decorator Facade Flyweight Proxy	Chain of Responsibility Command Iterator Mediator Memento Observer State Strategy Visitor

引子----玉帝传美猴王上天

- 玉皇大帝命令太白金星召美猴王上天：“金星径入（水帘洞）当中，面南立定道：“我是西方太白金星，奉玉帝招安圣旨，下界请你上天，拜受仙录””。

玉帝传美猴王上天





命令模式涉及到五个角色，它们分别是：

- 客户（Client）角色：创建了一个具体命令（ConcreteCommand）对象并确定其接收者。
- 命令（Command）角色：声明了一个给所有具体命令类的抽象接口。这是一个抽象角色，通常由一个 Java 接口或 Java 抽象类实现。
- 具体命令（ConcreteCommand）角色：定义一个接收者和行为之间的弱耦合；实现 execute()方法，负责调用接收者的相应操作。execute()方法通常叫做执行方法。
- 请求者（Invoker）角色：负责调用命令对象执行请求，相关的方法叫做行动方法。
- 接收者（Receiver）角色：负责具体实施和执行一个请求。任何一个类都可以成为接收者，实施和执行请求的方法叫做行动方法。

```
public class Client
{
    public static void main(String[] args)
    {
        Receiver receiver = new Receiver();
        Command command = new ConcreteCommand(receiver);
        Invoker invoker = new Invoker( command );
        invoker.action();
    }
}
```

```
public class Invoker
{
    private Command command;
    /**
     *  构造子
     */
    public Invoker(Command command)
    {
        this.command = command;
    }
    /**
     *  行动方法
     */
    public void action()
    {
        command.execute();
    }
}
```

```
public class Receiver
{
    /**
     *  构造子
     */
     public Receiver()
     {
          //write code here
     }
    /**
     *  行动方法
     */
     public void action()
     {
          System.out.println("Action has been taken.");
     }
}
```

```
public interface Command
{
    /**
     *  执行方法
     */
     void execute();
}
```

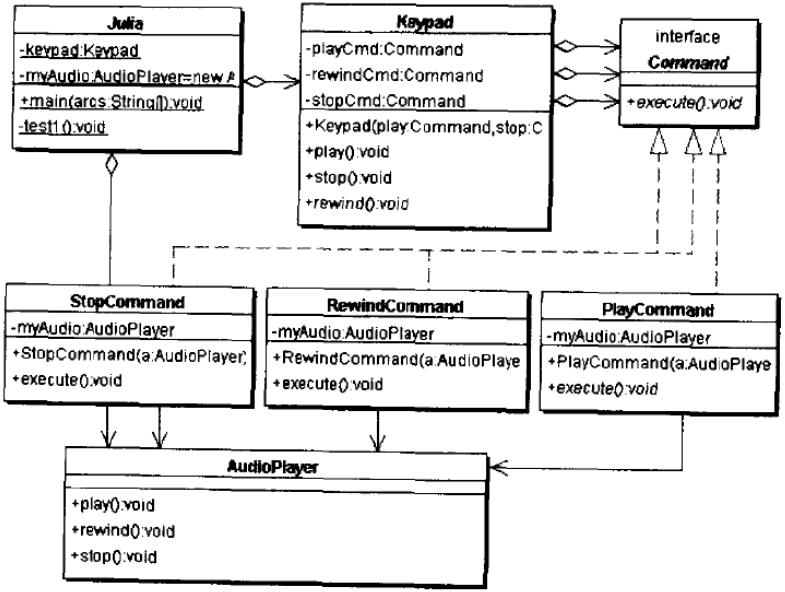
```
public class ConcreteCommand implements Command
{
    private Receiver receiver;
   /**
    *  构造子
    */
    public ConcreteCommand(Receiver receiver)
    {
        this.receiver = receiver;
    }
   /**
    *  执行方法
    */
    public void execute()
    {
        receiver.action();
    }
}
```

案例

- 烧烤摊VS. 烧烤店

案例-一个Audio Player的例子(自学)

- 小女孩Julia有一个盒式录音机，怎么实现这个录音机的模拟系统。
- 该系统有播音(play)、倒带(rewind)和停止(stop)功能。


Julia
-keypad:Keypad
-myAudio:AudioPlayer=new A
+main(args:String[]):void
-test1():void
Keypad
-playCmd:Command
-rewindCmd:Command
-stopCmd:Command
+Keypad(play:Command,stop:C
+play():void
+stop():void
+rewind():void
interface
Command
+execute():void
StopCommand
-myAudio:AudioPlayer
+StopCommand(a:AudioPlayer)
+execute():void
RewindCommand
-myAudio:AudioPlayer
+RewindCommand(a:AudioPlaye
+execute():void
PlayCommand
-myAudio:AudioPlayer
+PlayCommand(a:AudioPlaye
+execute():void
AudioPlayer
+play():void
+rewind():void
+stop():void

The end