链滴

GOF 设计模式小白教程之建造者

作者：valarchie

原文链接：https://ld246.com/article/1566923751177

来源网站：链滴

建造者模式（Builder）

定义：

将一个复杂对象的构造与它的表示分离，使同样的构建过程可以创建不同的表示，这样的设计模式被为建造者模式。它是将一个复杂的对象分解为多个简单的对象，然后一步一步构建而成。

通俗解释：

有一个包工头负责建房子。房子主要由地基，墙，屋顶构成。他设计了建房的图纸（先打地基，再砌，再盖屋顶）。包工头手下有工人，工人分为石材工和木头工。工人都会打造地基、墙还有屋顶。包头想盖木房子的话，就把木材工叫过来，把建房的图纸给他。同理，想盖石头房子的话，就把石材工来，还是用原来那份建房的图纸给他（意味着不用修改代码即可建不同的房子）。

代码：

包工头：规定了建房子的步骤，这个步骤是不变的。

```
public class Contractor {
    // 工人
    private Builder builder;
    // 建造房子
    public House buildHouse(Builder builder) {

        House house = new House();
        // 1.打地基 2.砌墙 3.盖屋顶
        house.setBase(builder.buildBase());
        house.setWall(builder.buildWall());
        house.setRoof(builder.buildRoof());

        return house;
    }

}
```

工人接口：都会打地基，砌墙，盖屋顶这三个技能

```
public interface Builder {
    // 打地基
    String buildBase();
    // 砌墙
    String buildWall();
    // 盖屋顶
    String buildRoof();

}
```

石材工人实现工人接口

```
public class StoneBuilder implements Builder {
    @Override
```

```
    public String buildBase() {
        return "石材地基";
    }

    @Override
    public String buildWall() {
        return "石材墙";
    }

    @Override
    public String buildRoof() {
        return "石材屋顶";
    }
}
```

木材工人实现工人接口

```
public class WoodBuilder implements Builder {
    @Override
    public String buildBase() {
        return "木材地基";
    }

    @Override
    public String buildWall() {
        return "木材墙";
    }

    @Override
    public String buildRoof() {
        return "木材屋顶";
    }
}
```

测试建房类

```
public class TestContractor {

    public static void main(String[] args) {

        // 包工头
        Contractor contractor = new Contractor();

        // 石材工人
        Builder stoneBuilder = new StoneBuilder();
        // 建石材房子
        House stoneHouse = contractor.buildHouse(stoneBuilder);
        System.out.println(stoneHouse.toString());

        // 木材工人
        WoodBuilder woodBuilder = new WoodBuilder();
        // 建木材房子
        House woodHouse = contractor.buildHouse(woodBuilder);
        System.out.println(woodHouse.toString());
```

```
    }

}
```

解析：

建造者模式其实是将单个零件构建和多个零件的装配进行了解耦。将固定的装配规则封装在Director，再由Director指挥Builder进行单个零件的构建。建造者和模板方法模式区别在于建造者将不变的规封装到另外的类当中，而模板方法将不变的规则封装在父类当中。并且用户完全不需要了解复杂的建规则就可以获得所需的复杂对象。这一模式也是开闭原则的良好例子。