00 简单工厂模式、策略模式、工厂方法模式

C-3 创建: 张林伟, 最后修改: 张林伟 2018-09-30 17:07

简单工厂模式和策略模式区别

- 用途：简单工厂模式是创建型模式，它的作用就是创建对象；策略模式是行为型模式，它的作用是让一个对象在许多行为中选择一种行为
- 行为：一个关注对象创建；一个关注行为封装
- 解决问题：简单工厂模式主要解决的是资源的统一分发，将对象的创建完全独立出来，让对象的创建和具体的使用客户无关。主要应用在多数据库选择，类库文件加载等；策略模式是为了解决策略的切换与扩展，让策略的变化独立于使用策略的客户

简单工厂模式

接口（或父类）

代码块 Java

```java
public interface IShape {
    /**
     * 形状名字
     * @return
     */
    String shapeName();
}
```

实现类（或子类）

代码块 Java

```java
// Circle.java
public class Circle implements IShape {
    @Override public String shapeName() {
        return "Circle";
    }
}

// Square.java
public class Square implements IShape{
    @Override public String shapeName() {
        return "Square";
    }
}

// Rectangle.java
public class Rectangle implements IShape{
    @Override public String shapeName() {
        return "Rectangle";
    }
}
```

工厂类

代码块 Java

```java
public class ShapeFactory {
    private static IShape shape;
    public static IShape getShape(String shapeName) throws Exception {
        switch (shapeName) {
            case "Rectangle":
                shape =  new Rectangle();
                break;
            case "Square":
                shape = new Square();
                break;
            case "Circle":
                shape = new Circle();
```

```
                break;
            default:
                throw new Exception("shapeName错误");
        }
        return shape;
    }
}
```

Demo

代码块 Java

```
public class SimpleFactoryPatternDemo {
    public static void main(String[] args) {
        try {
            IShape shape = ShapeFactory.getShape("Circle");
            System.out.println(shape.shapeName());
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

策略模式

策略模式

工厂方法模式

定义一个用于创建对象的接口，让子类决定实例化哪一个类。工厂方法使一个类的实例化延迟到其子类。

Demo

工厂接口

代码块 Java

```
public interface MathFactory {
    double operate(double x, double y);
}
```

工厂实现类

代码块 Java

```
// AddFactory.java
public class AddFactory implements MathFactory {
    @Override public double operate(double x, double y) {
        return x + y;
    }
}

// SubFactory.java
public class SubFactory implements MathFactory{
    @Override public double operate(double x, double y) {
        return x - y;
    }
}
```

客户端

代码块 Java

```
public class Client {

    public static void main(String[] args) {
        MathFactory addFactory = new AddFactory();
```

```
        System.out.println(addFactory.operate(1,2));

        MathFactory subFactory = new SubFactory();
        System.out.println(subFactory.operate(1,2));
    }
}
```

简单工厂 vs 工厂方法：

简单工厂模式的最大优点在于工厂类中包含了必要的逻辑判断，根据客户端的选择条件动态实例化相关的类，对于客户端来说，去除了与具体产品的依赖。

工厂方法模式实现时，客户端需要决定实例化哪一个工厂。也就是说，工厂方法把简单工厂的内部逻辑判断移到了客户端代码来进行。