06 状态模式、适配器模式

C-3 创建: 张林伟, 最后修改: 张林伟 2018-10-22 12:26

状态模式

当一个对象的内在状态改变时允许改变其行为，这个对象看起来像是改变了其类。

状态模式主要解决的是当控制一个对象状态转换的条件表达式过于复杂时的情况。把状态的判断逻辑转移到表示不同状态的一系列类当中，可以把复杂的判断逻辑简化。

Demo

状态接口

代码块 Java

```java
public interface State {
    void handle(Context context);
}
```

状态具体类

代码块 Java

```java
// ConcreteStateA.java
public class ConcreteStateA implements State {
    @Override public void handle(Context context) {
        System.out.println("当前状态为：状态B");
        context.setState(new ConcreteStateB());
    }
}

// ConcreteStateB.java
public class ConcreteStateB implements State {
    @Override public void handle(Context context) {
        System.out.println("当前状态为：状态A");
        context.setState(new ConcreteStateA());
    }
}
```

Context

代码块 Java

```java
@Data
@AllArgsConstructor
public class Context {

    private State state;

    public void request() {
        state.handle(this);
    }
}
```

客户端

代码块 Java

```java
public class Client {

    public static void main(String[] args) {
        Context context = new Context(new ConcreteStateA());

        context.request();
        context.request();
        context.request();
        context.request();
    }
```

```
}
```

适配器模式

将一个类的接口转换成客户希望的另外一个接口。适配器模式使得原本由于接口不兼容而不能一起工作的那些类可以一起工作。

Demo

适配器

代码块 Java

```java
// Target.java
public interface Target {
    void request();
}

// Adapter.java
public class Adapter implements Target {
    @Setter
    private Adaptee adaptee;

    public Adapter(Adaptee adaptee) {
        this.adaptee = adaptee;
    }

    @Override public void request() {
        adaptee.specificRequest();
    }
}
```

被适配者

代码块 Java

```java
// Adaptee.java
public interface Adaptee {
    void specificRequest();
}

// ConcreteAdaptee1.java
public class ConcreteAdaptee1 implements Adaptee {
    @Override public void specificRequest() {
        System.out.println("执行业务代码1");
    }
}

// ConcreteAdaptee2.java
public class ConcreteAdaptee2 implements Adaptee {
    @Override public void specificRequest() {
        System.out.println("执行业务代码2");
    }
}
```

客户端

代码块 Java

```java
public class Client {
    public static void main(String[] args) {
        Target target = new Adapter(new ConcreteAdaptee1());
        target.request();

        ((Adapter) target).setAdaptee(new ConcreteAdaptee2());
        target.request();
    }
}
```