软件体系结构作业 14

姓　名：洪伟鑫

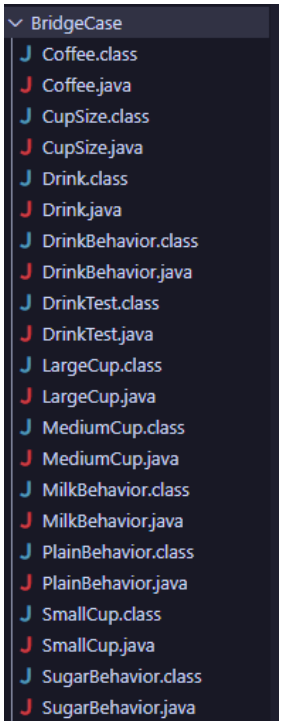
专　业：软件工程

年　级：2022 级

学　号：37220222203612

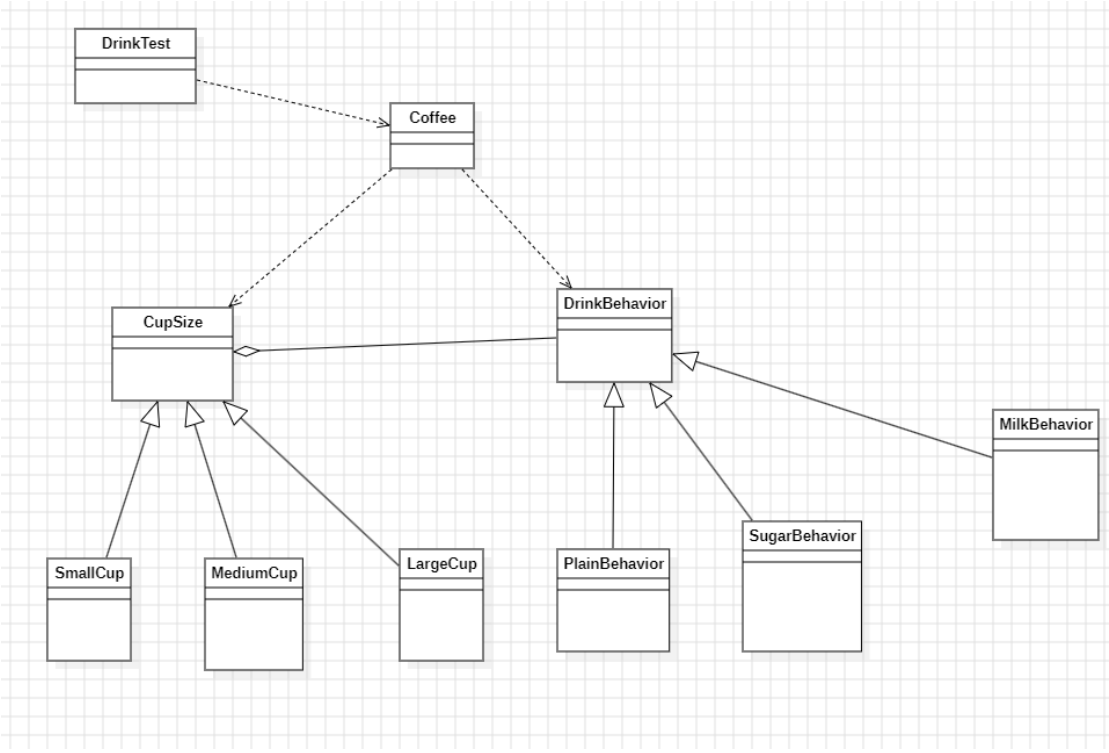
2025 年 4 月 22 日

1、 试用 Bridge 模式完成下列事情：饮料的杯子有大、中、小；行为有：加奶，加糖，啥都不加。



具体包含的类和接口如下:

1. Drink 接口：定义了饮料的基本行为
2. Coffee 类：实现了 Drink 接口，是具体的饮料类
3. CupSize 抽象类：定义了杯子大小的抽象
4. LargeCup、MediumCup、SmallCup：具体的杯子大小实现
5. DrinkBehavior 接口：定义了饮料行为的抽象
6. MilkBehavior、SugarBehavior、PlainBehavior：具体的行为实现



代码:

```java
package BridgeCase;

public class DrinkTest {
    Run | Debug
    public static void main(String[] args) {
        // 大杯加奶咖啡
        Drink largeMilkCoffee = new Coffee(new LargeCup(), new MilkBehavior());
        System.out.println("=== 大杯加奶咖啡 ===");
        largeMilkCoffee.prepare();
        largeMilkCoffee.serve();
        System.out.println();

        // 中杯加糖咖啡
        Drink mediumSugarCoffee = new Coffee(new MediumCup(), new SugarBehavior());
        System.out.println("=== 中杯加糖咖啡 ===");
        mediumSugarCoffee.prepare();
        mediumSugarCoffee.serve();
        System.out.println();

        // 小杯原味咖啡
        Drink smallPlainCoffee = new Coffee(new SmallCup(), new PlainBehavior());
        System.out.println("=== 小杯原味咖啡 ===");
        smallPlainCoffee.prepare();
        smallPlainCoffee.serve();
        System.out.println();

        // 大杯加糖咖啡
        Drink largeSugarCoffee = new Coffee(new LargeCup(), new SugarBehavior());
        System.out.println("=== 大杯加糖咖啡 ===");
        largeSugarCoffee.prepare();
        largeSugarCoffee.serve();
        System.out.println();

        // 中杯加奶咖啡
        Drink mediumMilkCoffee = new Coffee(new MediumCup(), new MilkBehavior());
        System.out.println("=== 中杯加奶咖啡 ===");
        mediumMilkCoffee.prepare();
        mediumMilkCoffee.serve();
    }
}
```

以大杯加奶咖啡为例

```java
package BridgeCase;

public class LargeCup extends CupSize {
    public LargeCup() {
        this.size = "大";
    }
}
```

```java
package BridgeCase;

public class MilkBehavior implements DrinkBehavior {
    @Override
    public void addBehavior() {
        System.out.println("加奶");
    }
}
```

桥接器中分别实现了三种 Size 和三种行为（不加、加糖、加奶）
只需要在 Coffee 类实例化时传入参数即可，无需每种组合都进行实现。

```java
package BridgeCase;

public class Coffee implements Drink {
    private CupSize cupSize;
    private DrinkBehavior behavior;

    public Coffee(CupSize cupSize, DrinkBehavior behavior) {
        this.cupSize = cupSize;
        this.behavior = behavior;
    }

    @Override
    public void prepare() {
        System.out.println("准备" + cupSize.getSize() + "杯咖啡");
        behavior.addBehavior();
    }

    @Override
    public void serve() {
        System.out.println("服务" + cupSize.getSize() + "杯咖啡");
    }
}
```

Coffee 类中可以直接准备对应大小和采用对应行为地实现具体咖啡制作。

运行结果:

```
PS E:\大三资料\大三下课程资料\体系结构\PPT\Code\Bridge> javac -encoding UTF-8 BridgeCase/*.java
PS E:\大三资料\大三下课程资料\体系结构\PPT\Code\Bridge> java BridgeCase.DrinkTest
=== 大杯加奶咖啡 ===
准备大杯咖啡
加奶
服务大杯咖啡

=== 中杯加糖咖啡 ===
准备中杯咖啡
加糖
服务中杯咖啡

=== 小杯原味咖啡 ===
准备小杯咖啡
原味
服务小杯咖啡

=== 大杯加糖咖啡 ===
准备大杯咖啡
加糖
服务大杯咖啡

=== 中杯加奶咖啡 ===
准备中杯咖啡
加奶
服务中杯咖啡
```