Command

Upravte aplikaci „Smart Café“, aby objednávky a platby probíhaly pomocí návrhové vzoru Command. Každá notifikace by tedy nově měla proběhnout v příslušné třídě (pro objednávku nebo platbu).

Nápověda

Vytvořte rozhraní `Command`, které bude předepisovat metodu `execute()`. Následně vytvořte třídy `OrderCommand` a `PaymentCommand`, které rozhraní implementují. V rámci konstruktoru každá je oběma třídám předán příslušný `OrderSubject` (aby bylo možné notifikovat všechny zaregistrované subjekty) a informace potřebné pro objednávku nebo platbu. V rámci metody `execute()` pak proběhne notifikace všech zaregistrovaných subjektů.

Ukázka možné podoby hlavní třídy

```
EmployeeObserver waiter = new EmployeeObserver("Waiter");
EmployeeObserver barista = new EmployeeObserver("Barista");
OrderSubject order = new OrderSubject();
order.addObserver(barista);
OrderSubject payment = new OrderSubject();
payment.addObserver(waiter);

CustomDrink coffee = new CustomDrink.Builder("coffee").milk().sugar().build();
CustomDrink tea = new CustomDrink.Builder("tea").sugar().build();

String cafeName = CafeConfig.getInstance().getCafeName();

Command coffeeOrder = new OrderCommand(
        order,
        coffee + " in " + cafeName
);
Command teaOrder = new OrderCommand(
        order,
        tea + " in " + cafeName
);

coffeeOrder.execute();
teaOrder.execute();

Checkout checkout = new Checkout(new CreditCardPayment());
Command payByCard = new PaymentCommand(
        payment,
        checkout,
        150,
        4
);
checkout.setStrategy(new CashPayment());
Command payByCash = new PaymentCommand(
        payment,
        checkout,
        100,
        2
);
payByCard.execute();
payByCash.execute();
```

Pozn.: Hotová aplikace je taková, která není otestována pouze Vámi, ale především unit testy.