POS Cafe System

Marissa Bueno

Overview

I created a system for a Cafe that handles

- Transactions
- Inventory
- Customer Loyalty Program

Tools

- Java 8
- SDK 11
- VS Code

Motivation

This system was motivated by my love for coffee and frustration towards a crowded coffee bar. I wanted to create a system that would alert customers the status of their orders while also handling business logic for the Cafe.

Object Oriented Design

Factory

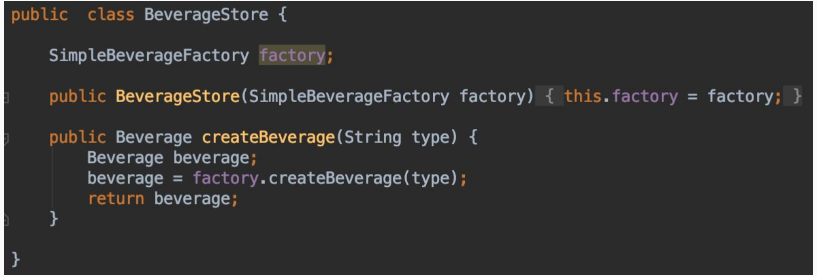
Template

Decorator

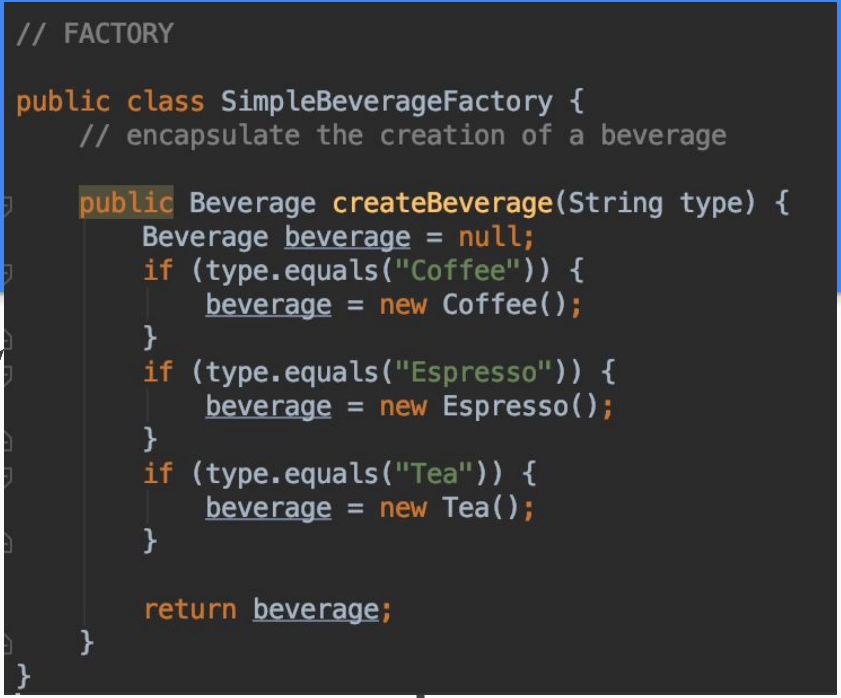
Command

Factory

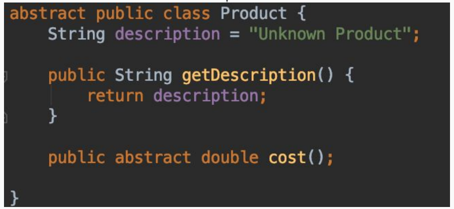
beverageStore.createBeverage(order.getBeverageOrder().get(i));



```java
public  class BeverageStore {

    SimpleBeverageFactory factory;

    public BeverageStore(SimpleBeverageFactory factory) { this.factory = factory; }

    public Beverage createBeverage(String type) {
        Beverage beverage;
        beverage = factory.createBeverage(type);
        return beverage;
    }

}
```

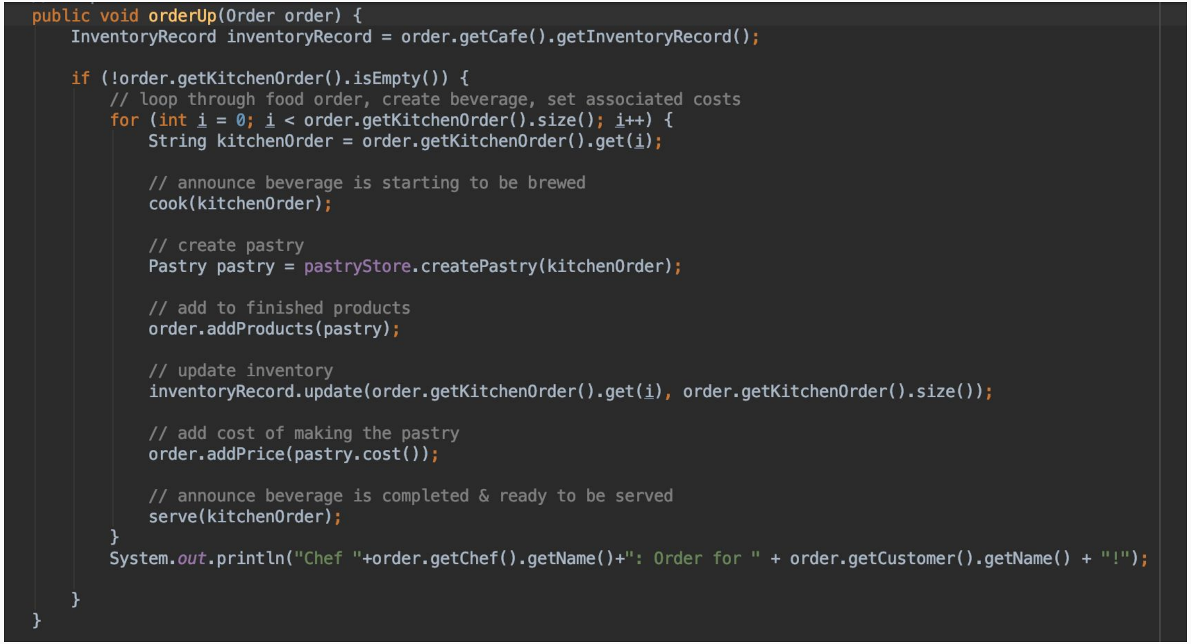


```java
// FACTORY

public class SimpleBeverageFactory {
    // encapsulate the creation of a beverage

    public Beverage createBeverage(String type) {
        Beverage beverage = null;
        if (type.equals("Coffee")) {
            beverage = new Coffee();
        }
        if (type.equals("Espresso")) {
            beverage = new Espresso();
        }
        if (type.equals("Tea")) {
            beverage = new Tea();
        }

        return beverage;
    }
}
```

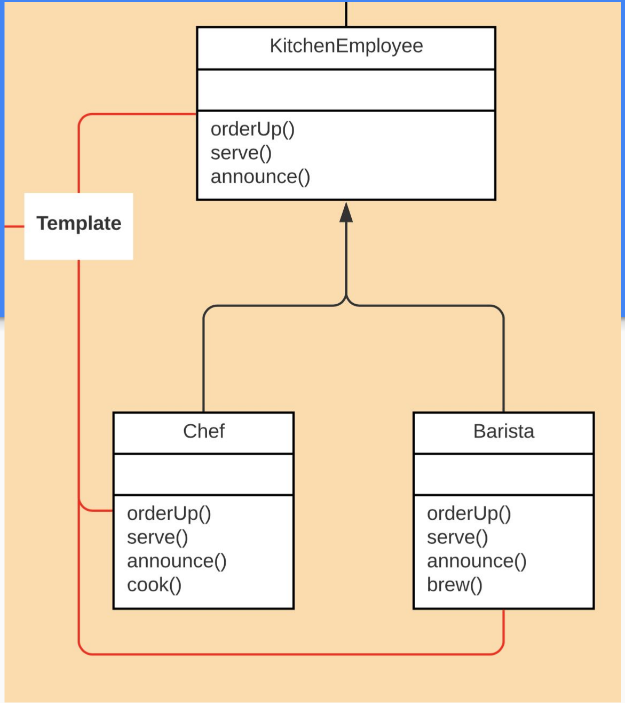


```java
abstract public class Product {
    String description = "Unknown Product";

    public String getDescription() {
        return description;
    }

    public abstract double cost();
}
```

Template

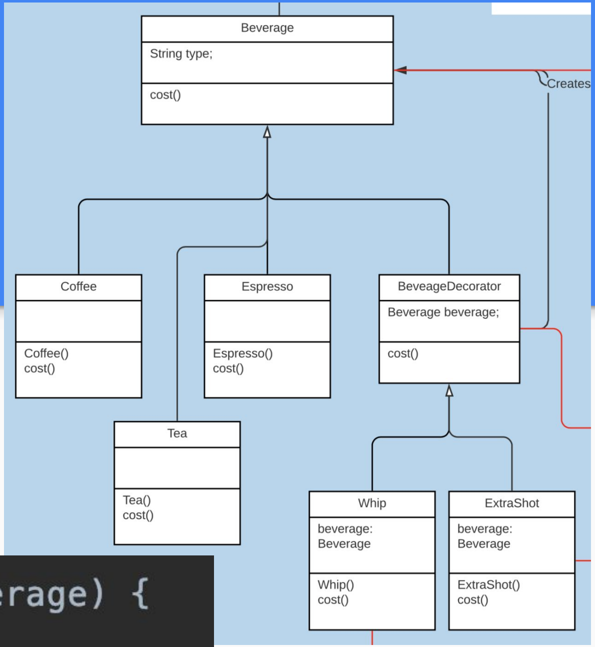


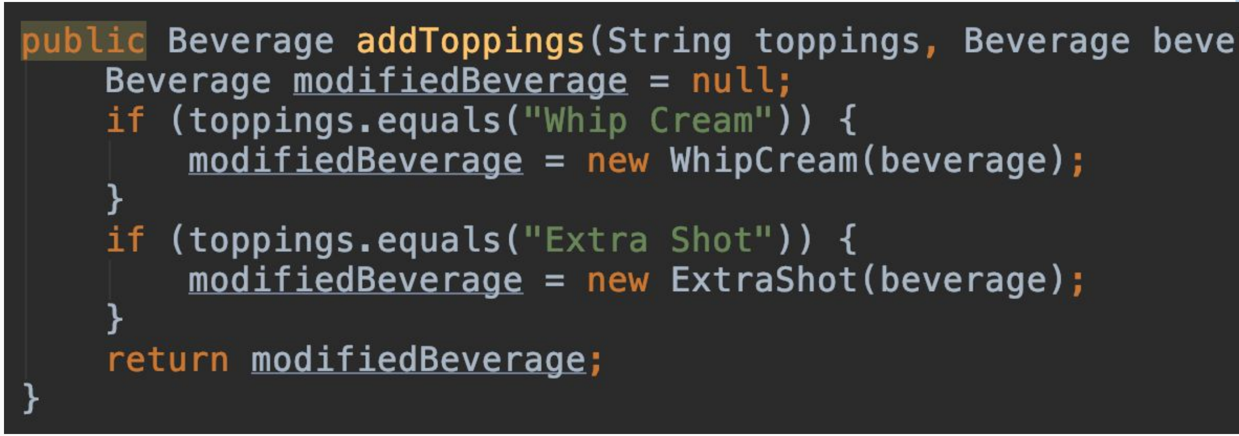
```java
public void orderUp(Order order) {
    InventoryRecord inventoryRecord = order.getCafe().getInventoryRecord();

    if (!order.getKitchenOrder().isEmpty()) {
        // loop through food order, create beverage, set associated costs
        for (int i = 0; i < order.getKitchenOrder().size(); i++) {
            String kitchenOrder = order.getKitchenOrder().get(i);

            // announce beverage is starting to be brewed
            cook(kitchenOrder);

            // create pastry
            Pastry pastry = pastryStore.createPastry(kitchenOrder);

            // add to finished products
            order.addProducts(pastry);

            // update inventory
            inventoryRecord.update(order.getKitchenOrder().get(i), order.getKitchenOrder().size());

            // add cost of making the pastry
            order.addPrice(pastry.cost());

            // announce beverage is completed & ready to be served
            serve(kitchenOrder);
        }
        System.out.println("Chef "+order.getChef().getName()+": Order for " + order.getCustomer().getName() + "!");

    }
}
```



Which steps are standard and which steps are specific to each employee

Decorator





```
public Beverage addToppings(String toppings, Beverage beverage) {
    Beverage modifiedBeverage = null;
    if (toppings.equals("Whip Cream")) {
        modifiedBeverage = new WhipCream(beverage);
    }
    if (toppings.equals("Extra Shot")) {
        modifiedBeverage = new ExtraShot(beverage);
    }
    return modifiedBeverage;
}
```

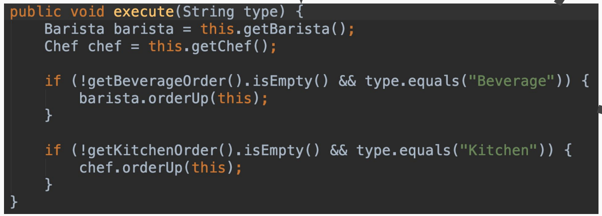
Command

Cashier takes an Order

invokes

order.execute("Beverage");

order.execute("Kitchen");



```
public void execute(String type) {
    Barista barista = this.getBarista();
    Chef chef = this.getChef();

    if (!getBeverageOrder().isEmpty() && type.equals("Beverage")) {
        barista.orderUp(this);
    }

    if (!getKitchenOrder().isEmpty() && type.equals("Kitchen")) {
        chef.orderUp(this);
    }
}
```

receivers

Noteworthy Changes

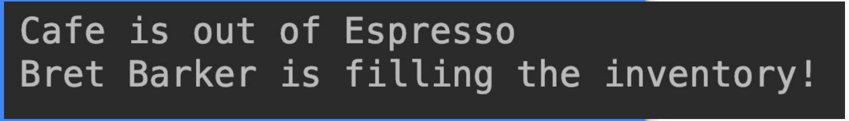
Decouple Simulator from the Cafe

- Originally I had many lists getting passed into the Cafe as parameters.
- I opted to have .txt files be parameters such that establishing a new Cafe is as easy as laying out everything in a .txt file. The Cafe will read the files in and set up the Cafe accordingly & be ready for operation!

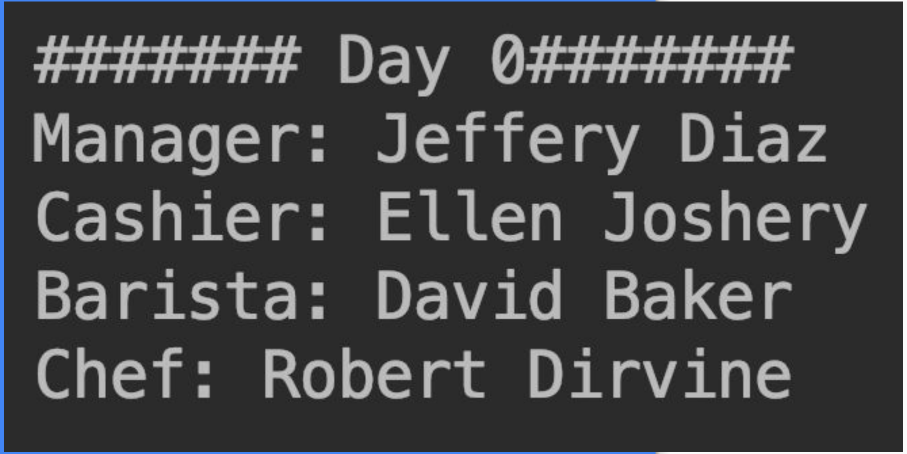
Removing the Singleton Pattern for the Manager

- I realized that this pattern wasn’t going to be very useful and could break the system if I had more than one Manager work throughout X amount of time or if I had multiple cafes.

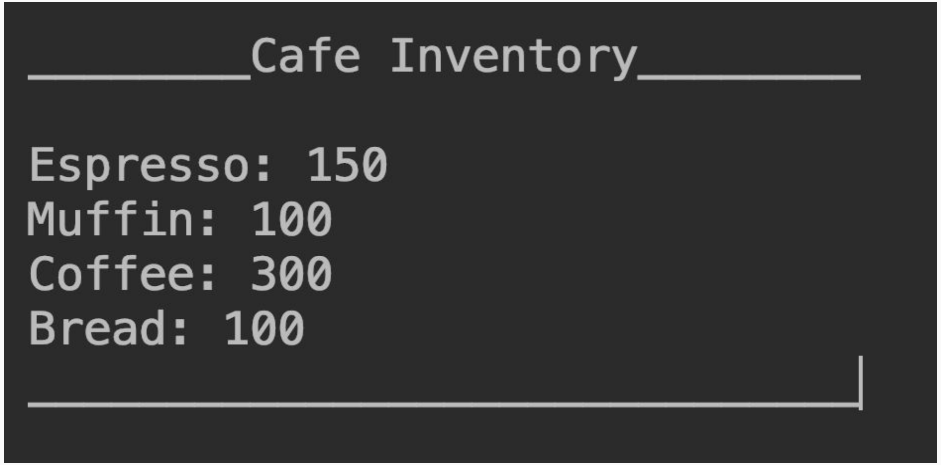
Screen Grabs



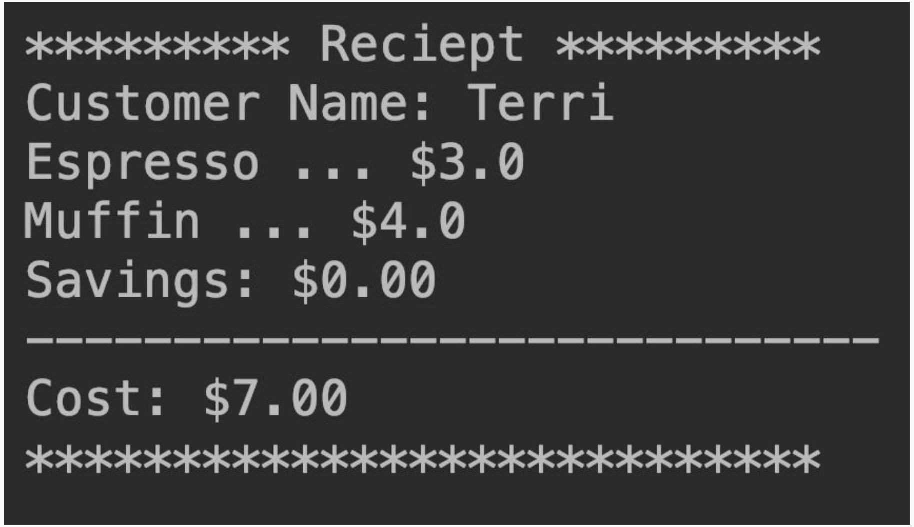
```
Cafe is out of Espresso
Bret Barker is filling the inventory!
```



```
####### Day 0#######
Manager: Jeffery Diaz
Cashier: Ellen Joshery
Barista: David Baker
Chef: Robert Dirvine
```

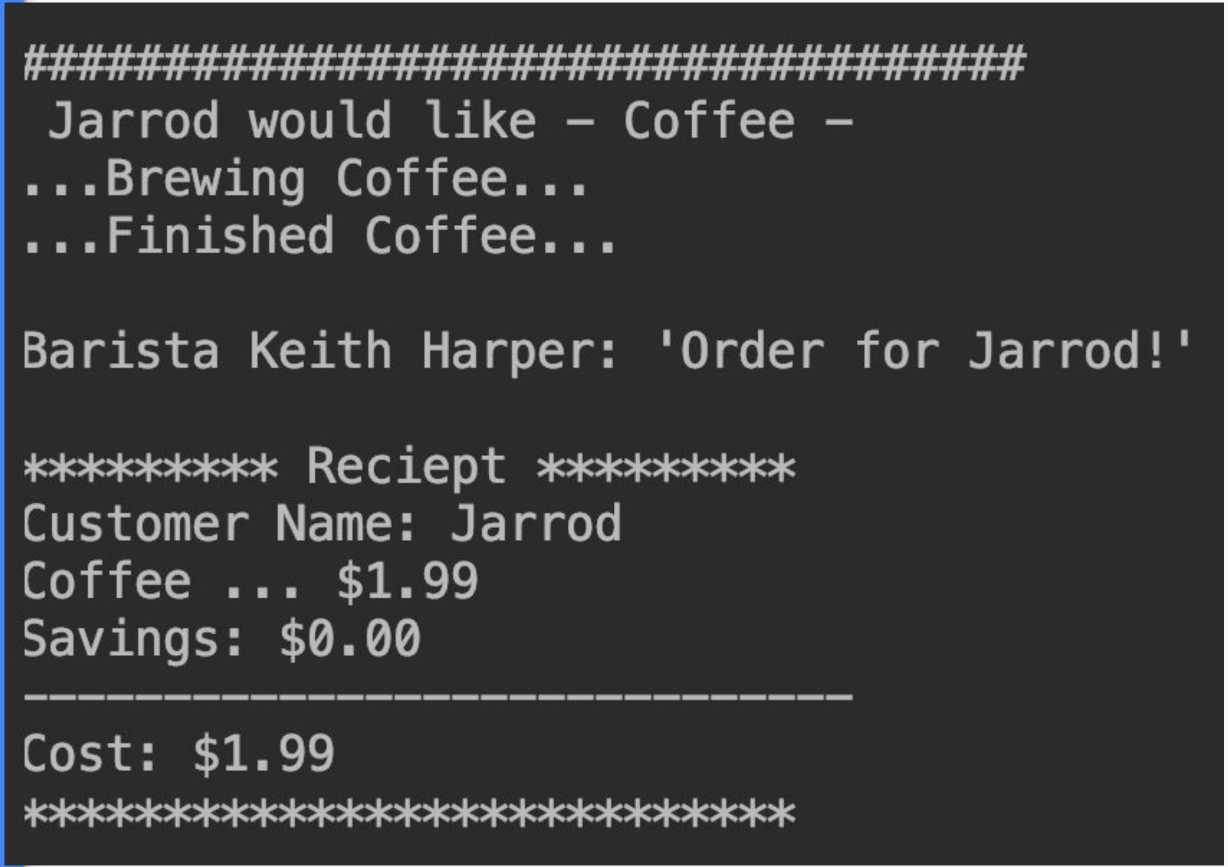


```
________Cafe Inventory________

Espresso: 150
Muffin: 100
Coffee: 300
Bread: 100
                              |
______________________________|
```



```
********** Reciept **********
Customer Name: Terri
Espresso ... $3.0
Muffin ... $4.0
Savings: $0.00
-------------------------------
Cost: $7.00
*****************************
```

Typical Order



```
###################################
 Jarrod would like - Coffee -
...Brewing Coffee...
...Finished Coffee...

Barista Keith Harper: 'Order for Jarrod!'

********* Reciept *********
Customer Name: Jarrod
Coffee ... $1.99
Savings: $0.00
-----------------------------
Cost: $1.99
***************************
```