DDROIDD Summer Internship 2022

Documentation

Martin Maria-Denisa

Content

1. The objective

The main goal was to design a program that would simulate a shopping cart. The customer can create orders, and at the end he will receive a receipt.

2. Problem analysis, modeling, scenarios, use cases.

To model this problem I used the Java language and I used the Lombok library.

Oriented programming allows us to deal with the problem at a higher level, without being constrained, to such an extent, by the technical characteristics. This strategy is also called bottom-up design. It is very advantageous because structures with a direct connection in the real world (objects, actions, etc.) can be found.

How to use the program:

You have to type the following:

-catalog (to see the product catalog)

- create order (to create a new order);

- add (to put a product in the cart);

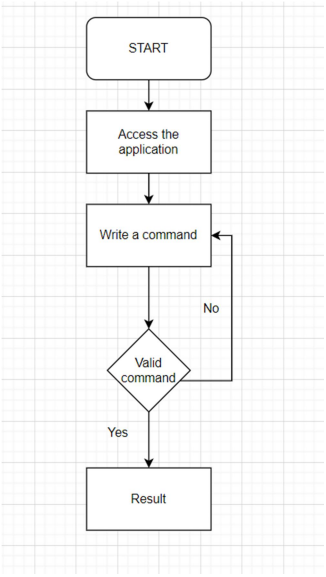
Then the name of the product will be asked: write the name of the desired product and press enter, without other spaces after the name;

-at the end write checkout to see the receipt

-exit (to exit the application);

Success story: data entered correctly.

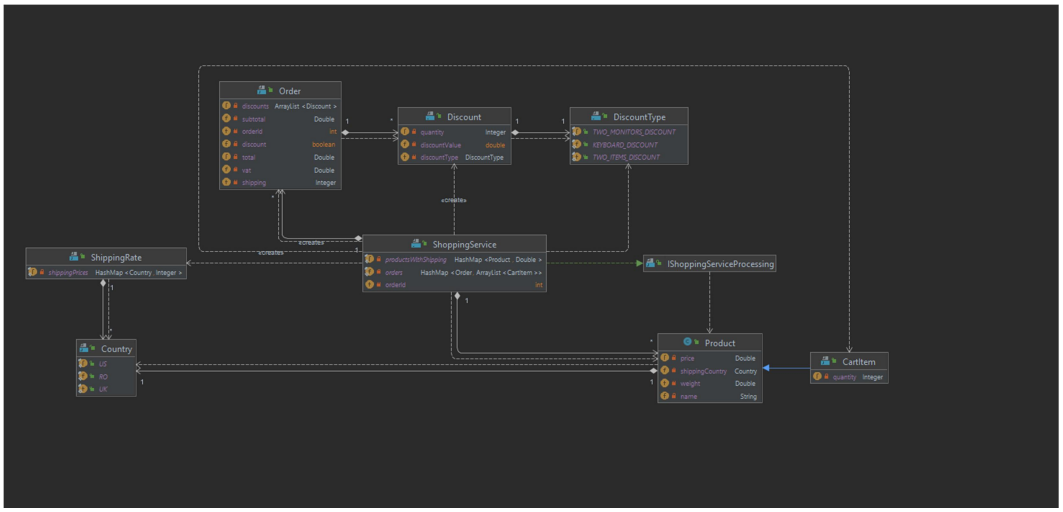
Case of failure: Failure to fill in the required data, incorrect spelling of product names or orders.



3. Design

The architectural model used is model M (V) C (Model-View-Controller). MVC is a widespread concept in Web programming. The purpose of MVC is to keep the business logic and user interface separate, so that those who maintain the application can change one part more easily, without affecting other parts. In MVC, the model contains the information (data) and business rules and is responsible for managing the data in the application. The model is the core of the application. The controller is the part of the application that deals with user interaction. The data entered by the user is read in the controller, sent to the model and the operations are performed.

UML Diagram:



4. Implementation

I decided to design the classes so that the code was as intuitive and readable as possible.

Package model

CartItem: The class that defines a product that is in the shopping cart. This inherits the Product class.

Discount: The class that defines a discount.

Product: The class that defines a product.

ShippingRate: The class that contains shipping rates.

ShoppingService: Here we deal with the actions on the product basket and orders.

It contains the methods:

-addProduct(Product product) : Add a new product to the productsWithShipping;

-calculateShipping(Product product): Calculate the shipping fee for each product;

-createNewOrder(): Create a new order;

-addToOrder(String product): Add the desired product to the order;

-computeTotals(): Calculate the total, subtotal, VAT and shipping fee;

-checkForDiscounts():Check for discounts that may apply;

-applyDiscount(): Apply the discounts found;

-getMyProduct(String product): Returns a product by name;

-checkForTwoItemsDiscounts(): We check if there is a discount in which we have more than two products;

-checkForKeyboardDiscount(): We check if we have the discount for the keyboard;

-checkForTwoMonitorsDiscount(): We check if we have the discount for two monitors;

DiscountType :Enumeration that contains the types of discounts.

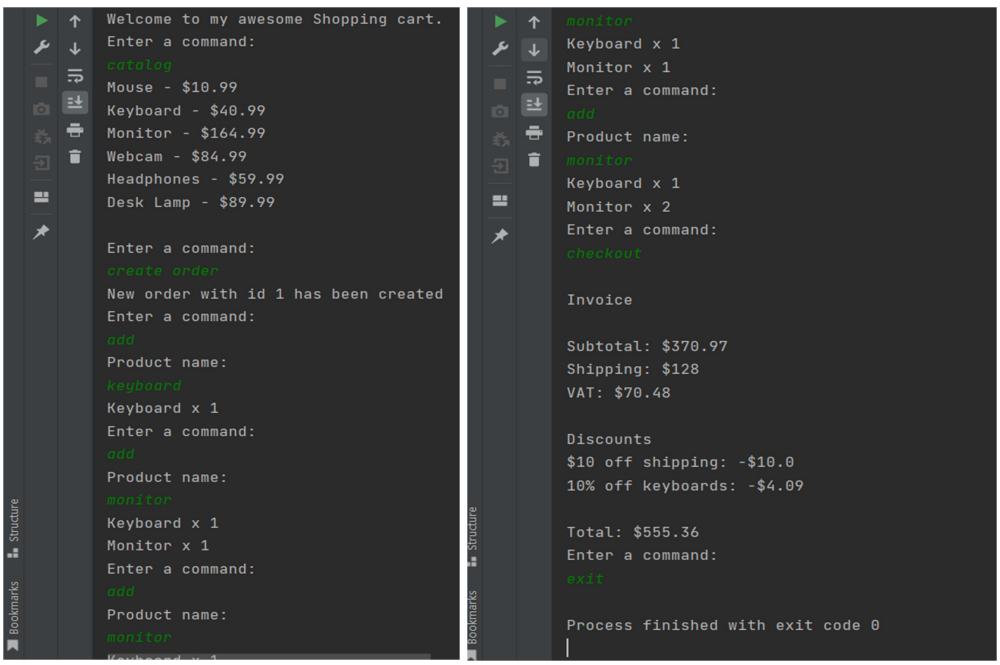
Country: Enumeration that contains the countries from which the products are delivered.

Package controller

Controller: The controller class deals with the given commands.

5. Results

The program runs properly as long as the data is entered correctly. If the user enters characters other than those allowed, the program will not treat this case separately. The results as well as the values are in the grid of those expected. Junit tests and numerous test cases have been introduced manually to test the correctness of the methods as well as the operation of the application.



6. Conclusions

From this challenge I learned the role that the separation of objectives has at the very beginning of the problem and it strengthened and deepened my knowledge.

As further developments, the idea of creating a graphical interface seems very interesting and useful.

Here is a model I created in Paint: