Keenan McElwee

OOPDA

Professor Myers

7/3/16

Della’s House of Bagels Write-Up

**Driver:** instantiates an instance of the class OrderCalculatorGUI.

**CostPanel:** extends JPanel; contains all information regarding the costs of an order (sub, tax, and total prices) as well as buttons to calculate and update the presented costs and to exit the application. In the constructor Two vertical Boxes contain all elements (one for the JLabels that display the costs, the other for the JButtons that operate the calculate and exit functions). The constructor instantiates three JLabels (for subtotal, tax, and total), two JButtons (for the calculate and exit functions), and their respective Boxes to which they are added; then it adds an inner class that extends ActionListener (ExitButtonListener) which, when the Exit button is pressed, calls a method that ends the program. Its layout is set to new BorderLayout, with the buttonBox to the east and totalsBox to the west, so that they may appear side-by-side without interfering with one another. CostPanel also contains private fields to contain the JButtons and JLabels, as well as getters and setters for those fields.

**ToppingPanel:** extends JPanel; contains the display for, information on, and functions of selecting toppings for a bagel. In the constructor the class’s private fields for toppings (JCheckBoxes) are instantiated with strings that label what the JCheckBox is for (Cream cheese, Butter, etc) and adds them to the panel after it sets itself a Border with a call to BorderFactory.createTitledBorder() with the passed parameter “Toppings” so that the panel is clearly labeled for the user. The class also contains the method getToppingCost() that returns a double of value equal to the sum of the class’s final doubles corresponding to the selected JCheckBoxes.

**GreetingPanel:** extends JPanel; acts as a header for window. The constructor instantiates a JLabel with the parameter “Welcome to Della’s Limited House of Bagels”, then adds this JLabel to itself.

**BagelPanel:** extends JPanel; contains the display for, information on, and functions of selecting a type of bagel. The constructor calls setLayout on the overall class to set to new GridLayout(2,1) so that the two JRadioButtons are placed evenly vertically in the panel. The class’s private fields for white- and wheatBagel JRadioButtons are instantiated with the parameters (“White”, true) and (“Wheat”) respectively to label clearly the JRadioButtons for the user; the former passing true so that it is selected when the panel is first instantiated. The JRadioButtons are added to a newly instantiated ButtonGroup so that they know to interact with only each other, and the JRadioButtons are added to the panel after it sets itself a Border with a call to BorderFactory.createTitledBorder() with the passed parameter “Bagel” so that the panel is clearly labeled for the user. The class also contains the method getBagelCost() that returns a double of value equal to the final double corresponding to the selected JRadioButton.

**CoffeePanel:** extends JPanel; contains the display for, information on, and functions of selecting a type of coffee. The constructor calls setLayout on the overall class to set to new GridLayout(4,1) so that the four JRadioButtons are placed evenly vertically in the panel. The class’s private fields for no-, regular-, and decafCoffee as well as cappuccino JRadioButtons are instantiated with the parameters (“None”), (“Regular”,true), (“Decaf Coffee”), and (“Cappuccino”) respectively to label clearly the JRadioButtons for the user; the one passing true so that it is selected when the panel is first instantiated. The JRadioButtons are added to a newly instantiated ButtonGroup so that they know to interact with only each other, and the JRadioButtons are added to the panel after it sets itself a Border with a call to BorderFactory.createTitledBorder() with the passed parameter “Coffee” so that the panel is clearly labeled for the user. The class also contains the method getCoffeeCost() that returns a double of value equal to the final double corresponding to the selected JRadioButton.

**OrderCalculatorGUI:** extends JFrame; contains all panels, information, displays, and functionality required to order from Della’s House of Bagels. The constructor first calls to super() with the parameter “Order Calculator” to title the window the passed string, then calls setLayout(new BorderLayout()) to give order to the window and setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE) so that the user clicking on the window’s exit button will end the program. The five private fields of type JPanel (one for each JPanel subclass previously described) are instantiated and then added to the JFrame’s inherent panel; GreetingPanel to the north, BagelPanel the west, ToppingPanel the center, CoffeePanel the east, and CostPanel the south. The constructor then passes an inner class fimplementing ActionListener as a parameter on costPanel’s calcButton JButton’s addActionListener() method; this type of ActionListener instantiates a local variable of type double (subtotal) by calling the respective get\_\_\_\_Cost() methods of bagelPanel, toppingsPanel, and coffeePanel; a second local variable of type double (tax) by multiplying subtotal by the class’s final double field TAX\_RATE; and a third local variable of type double (total) by adding those two doubles together; finally after instantiating a DecimalFormat object (dollar) it calls the methods from costPanel getLabSub().getText(), getLabTax().getText(), and getLabTax().getText() by passing the respective type of cost the JLabels represent (“Subtotal: $”, “Tax: $”,”Total: $”) with concatenated on the respective local double variable after being passed as a parameter for dollar.format(). OrderCalculatorGUI also contains an inner class ExitButtonListener that implements ActionListener and is identical to the previously described ExitButtonListener but is not used locally.

**Issues in OOD Principal:**

* ExitButtonListener and CalcButtonListener should be in the CostPanel class, not OrderCalculatorGUI, so to reduce coupling between the two classes. Also, the latter’s copy of ExitButtonListener is private yet never called locally, so it does nothing there anyhow.
* BagelPanel and CoffeePanel are almost identical, with ToppingPanel marginally different: they all share similar constructors and get\_\_\_\_Cost() methods. They could easily be made subclasses of a common abstract superclass that contains a concrete or at least abstact method to cover the getCost() commonality as well as constructor choices.
* OrderCalculatorGUI’s private fields buttonPanel, calcButton, and exitButton are never used nor even instantiated so they further couple the class with CostPanel unneccessarily.