IST 256
Lab
Week 5 – Thursday, February 13, 2014

1. Writing a GUI Application with a method

In this lab, we will create a project with a GUI to manage the sales of books and magazines. Start a new project and name it something like SalesMethods (for sales with methods). Then create a jFrame Form and name it something like SalesMethodsGUI

In this first part of the lab, we will work together to create the form and the program that allows the user to buy books, and in the next part of the lab, you will add magazines to the form.

Make the Form

This project will have a user interface that assumes that the user can buy two different types of items, books and magazines. We will first create the form, with a user interface that looks something like this:

Enter Number Books at $20 |_______________|
(Get 10% off for 3 or more!)

Enter Number Mags at $3 |_______________|
(Get 5% off for 3 or more!)

|__Checkout __| (button)

Books: |_______________|

Magazines: |_______________|

Total: |_______________|

For the first instructions, you can either create two labels for the two lines of text, or write a multiline label using html tags to put the extra line (as demonstrated in class). (Note: a multi-line label has the text <html> label line 1 <br> label line 2.) To make the additional labels and textfields, note that you can copy and paste the GUI elements.

Form component names

As we noted before for a more complex form, we will rename our components. For each TextField, right click on the TextField and select Change Variable Name. A box will come up where you can type a more meaningful name, for example, you may call the TextFields “numberBooksTF”, “numberMagsTF”. “displayBooksTF”, “displayMagsTF”and “displayTotalTF”. You can also change the name of the button to be “CheckoutButton”.

Now create the Event -> Action -> actionPerformed method for the Checkout button, and notice that the name of the button method contains the new name of the button.
(We won't change the names of these labels because we are not using any of them in the program (with a setText()), but if you were going to use them, you could change the name in the same way.)

Write the a method to compute the price of the orders

Go to the source tab and in the program, before the CheckoutActionPerformed method, we will write a method called computeCost, which will be similar to the one developed in class.

Using the method developed in class for a model, design and write a similar method. How many parameters should it have? The key is to have 3 parameters.

First write the **method header line**, giving the scope keyword "public", the return type, the procedure name, and the formal parameter list. Note that you must have three parameters in the parameter list for this method.

The **body of the procedure** will compute the price of the order. It must compute the cost of the order as the number of items * the price of the items. It then deducts a discount based on the rate given as a parameter.

Write the Checkout Button actionPerformed method

The button method must do the following steps:

1. get the number of books typed into the TextField called numberBooksTF, checking the input for the empty string
2. get the number of magazines typed into the TextField called numberMagsTF, checking the input for the empty string
3. compute the cost of the books by calling the computeCost method,
4. compute the cost of the magazines by calling the computeCost method,
5. compute the total cost as the sum of the books and magazines
6. display the resulting costs for the books, magazines and total in the TextFields at the bottom of the form.

Here is more detail about how to do these steps:

0. Declare variables for the number of books, the cost of the book order, and the total order. (What types are these variables?)

1. For the books and magazines TextFields, if the TextField is empty (has value ""), the number of items is 0, otherwise get the number from the TextField .

2. Next, write a statement that will call the method computeCost to compute the cost of the books order, using appropriate actual parameters for books.
Assign the result of the computeCost method call to the variable for the price of books. (Always make sure that the number and types of actual parameters match the formal parameters in the method heading.)

3. Write a similar statement to get the cost of the magazines and another statement to add the books and magazines to get a total cost

3. Display all three results, formatting as currency.

[Recall how to format currency:
Put the statement "import java.text.NumberFormat;" in your program near the top. It should come after the package statement and before the first public class statement.

Next, put the statement that starts NumberFormat to create the currency formatter into your actionPerformed method; a good spot would be right after the variable declarations.

```
NumberFormat cf = NumberFormat.getCurrencyInstance();
```

Display the cost of books in the textfield named displayBooksTF, using cf.format(costBooks).]

Test your program

Run the program for several values for the number of books and magazines and note the results. For one trial, just leave the textfields empty, i.e. don't type anything. For the next 3 trials, type the books as 2 and the magazines from 0 to 3. For the last 2 trials, try some different numbers of books and magazines of your choice.

Book TextField	Mag TextField	Books	Magazines	Total
2	0			
2	1			
2	2			
2	3			

2. Add a Method: **Design a method** that will read a textField and return a number. If the TextField is empty, return 0 and otherwise return the number typed into the TextField.

In this method, a TextField will be a parameter to the method – here is an example of a method header with a TextField parameter. The type of a TextField can be given as javax.swing.JTextField:

```
private int getNumber(javax.swing.JTextField numberTF)
```

Call this method in your CheckoutButton to get the number from the numberBooksTF and the numberMagsTF.

Hand in your lab sheets with the program for the Checkout Button and with your two methods by the beginning of class next Tuesday, February 18, 2014.