Help – Project Week 11

Our project is to modify an example program that uses the Swing GUI Library. You must read the assigned textbook sections and do the participation exercises before you can modify the program.

Preliminary Steps

1. Create a NetBeans project called ProjectWeek11 and name the main class "SeatReservationFrame". Copy all of the source code from the "SeatReservationFrame" Class in the textbook into the editor – without deleting the package statement. Create a new Java Class file in the same package and call it "SeatInfo". Copy the source code from the "SeatInfo" Class in the textbook into the editor and again do not erase the package statement. Compile and run the program.
2. Carefully review the program and follow the code.
3. Consult the Help/Support document called "Help-ReadLinesFromTextFiles". You will incorporate the code from the help document into the "SeatReservationFrame" class. The main() method will become "fillSeatsFromFile()" and include the "rightString()" method as well just so you can get the program to read in the data from the "ReservationData.txt" file. Change the name of the file in the code to "ReservationData.txt". Set the Run properties of the Project Properties so the Working Directory is set to the package folder of the project folder. Copy the "ReservationData.txt" file into the package folder. Call the "fillSeatsFromFile()" at the end of the "SeatReservationFrame" Constructor. Compile and run the program. The file contents of "ReservationData.txt" should be output to the Output window (not the GUI) and the GUI should work as it did in Step 1.

Analyze Program

When you analyze the program, you will see that the main() method is used to create an object variable of the Class that has the main() method in it (SeatReservationFrame).

```
SeatReservationFrame myFrame = new SeatReservationFrame();
```

Looking at the code in constructor, we see that is where text fields and buttons are defined and built. The Class extends the JFrame Class so that is how the main container is defined by making a new SeatReservationFrame object called myFrame and then the Constructor places the GUI objects into the JFrame window.

The data is processed in two ways in this problem:

1. The storage of reservations is done using the ArrayList "seatResArr".
2. The display of the reservations is done using a JTable which was defined to hold a two dimensional array of type Object called tableVals.

   At the top of the SeatReservationFrame Constructor, tableVals is declared.

   ```
   Object[][] tableVals = new Object[5][4];
   ```

In the middle of the Constructor, the JTable is defined using the column headings variable and the tableVals array.

```
seatStatusTable = new JTable(tableVals, columnHeadings);
```

The problem called for us to change the number of total reservations from 5 to 20. The Authors gave us a constant called "NUM_SEATS". We need to change the assignment from 5 to 20. But the misstep the Authors made was not to use "NUM_SEATS" to define the "tableVals" array (they used the literal value of 5 in the declaration). Change the 5 to "NUM_SEATS" and you will complete the ability to record 20 reservations rather than 5 just by changing the value of "NUM_SEATS".

The actions for the buttons is processed with the method "actionPerformed()". This is an overridden method since the Class implements the "ActionListener" Interface. When you add a delete button, you will add code to process the delete in the "actionPerformed()" method. The code that set traps the action of the button clicks are defined as follows:

```
reserveButton.addActionListener(this);
```

and

```
quitButton.addActionListener(this);
```

Process Each File Record

We need to look at what happens when the "reserveButton" is clicked to know how to process each line of the "ReservationData.txt" file. The user is expected to have placed a seat #, a first name, a last name, and an amount in the GUI form fields. The routine extracts the values out of the form fields and then updates the "seatResArr" ArrayList with the values.

```
firstName = firstNameField.getText();
lastName = lastNameField.getText();
amtPaid = ((Number) amountPaidField.getValue()).intValue();

seatElement = new SeatInfo();          // Create new Seat object
seatElement.reserveSeat(firstName, lastName, amtPaid);
seatResArr.set(seatNum, seatElement); // Add seat to ArrayList
```

The display is then updated by updating the JTable with the updated "seatResArr" ArrayList.

```
updateTable();                          // Synchronize table with sts ArrayList
```

We can't call the "actionPerformed()" method to process each record of the file because of the specific data that is extracted from GUI form and we need to reference the array elements from the String's "split()" method. So for each record read in and then split into the fields, we can add the 6 lines to update the "seatResArr" shown above to the "fillSeatsFromFile()" method in place of the output of the data. The 3 assignment statements will instead reference the "fields" Array elements.

Add the "updateTable()" call (invocation) as the last line of the " SeatReservationFrame " Constructor … just after the "fillSeatsFromFile()" invocation.

Data from ReservationData.txt:

0,Martha,Hobson,45
5,John,Boraxle,60
8,Gene,Baxter,7000
11,Javier,Georges,65
12,Kyle,Ferrani,57
19,Shaniqua,James,200

The Modified App Interface: