The University of the West Indies, St. Augustine
COMP 2603 Object Oriented Programming 1

Week 8 Lab

In this lab, we will build a more sophisticated GUI using the Netbeans GUI Builder. The GUI will feature simple components such as textfields, labels, and buttons. It will also have advanced components such as combo boxes, radio buttons, check boxes and lists. Multiple panels will be used to organise these components. Action listeners and mouse listeners will then be used to control how the GUI responds to user-entered data and actions. Finally, we will connect the GUI to a domain object in keeping with the three-tier architecture for object-oriented applications.

Download link for the Netbeans IDE (Java SE): https://netbeans.org/downloads/

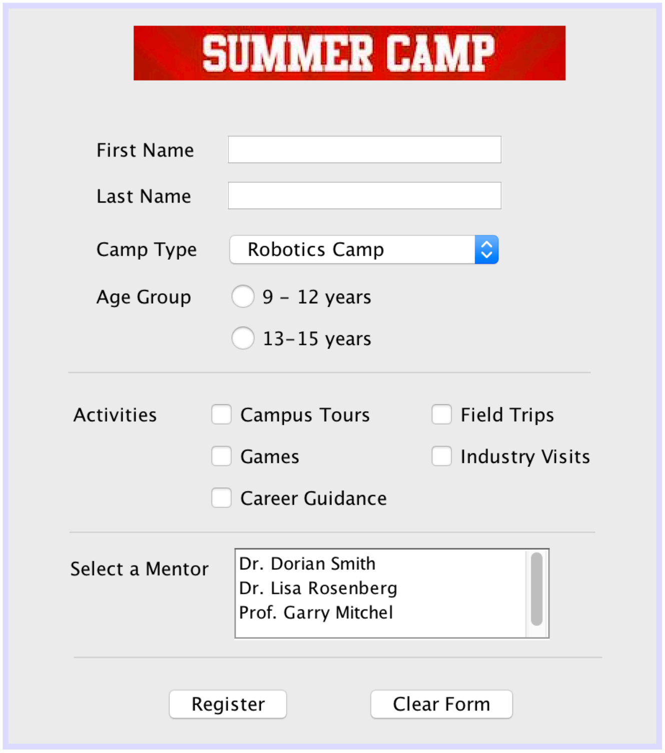


Figure 1

The lab focuses on creating a GUI for collecting data from persons interested in attending a summer camp. Figure 1 shows a mockup of what all of the GUI components looks like.

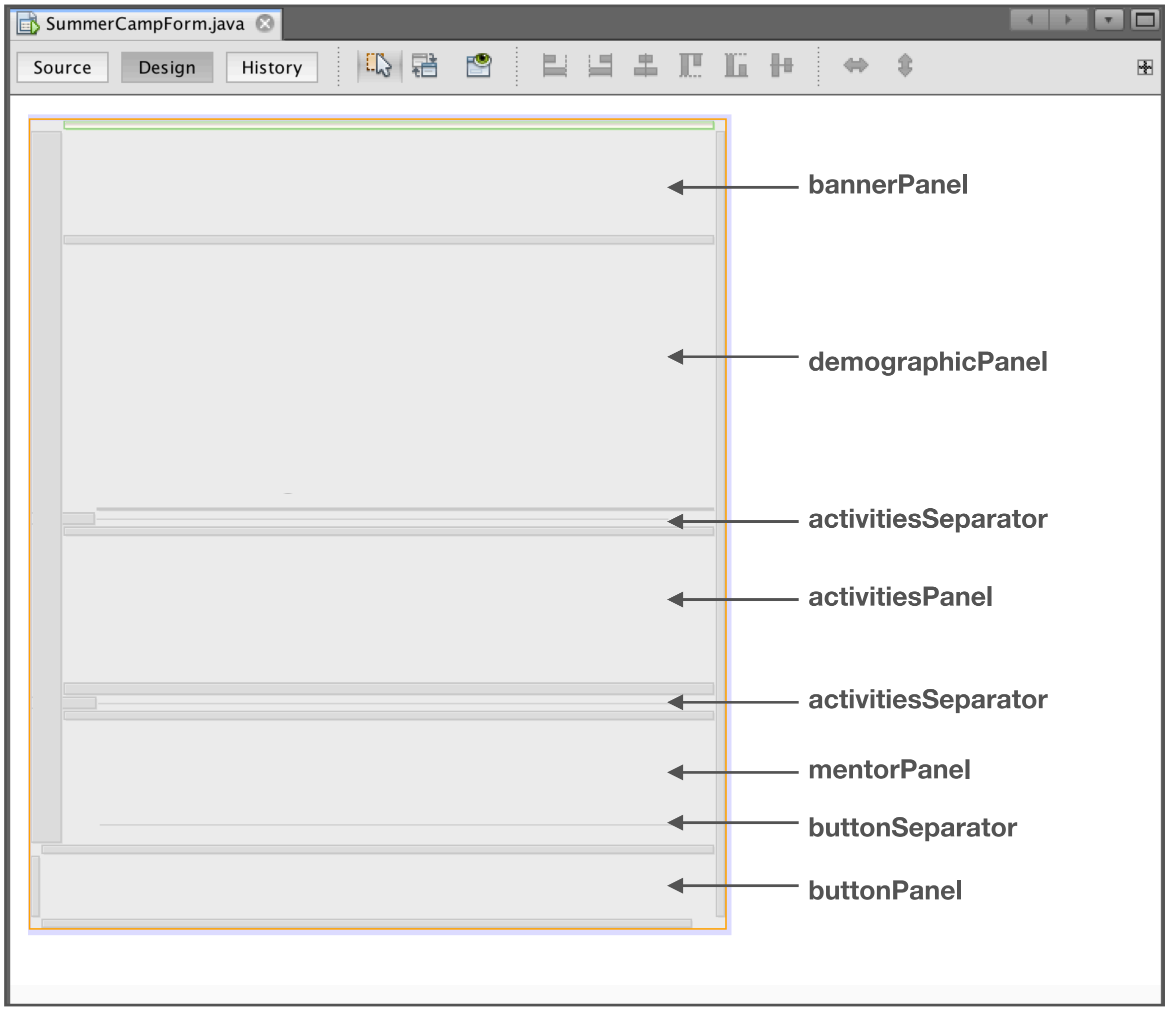
Part 1: Creating the JFrame and Panels

1. Open the Netbeans IDE and create a new Java Project called Lab8.
2. Create a new **JFrame** class called SummerCampForm with the following specifications: JFrame size: 420 x 425

3. Switch to Design view and add the following JPanel objects from the Swing Containers Palette:
    a. bannerPanel
    b. demographicPanel
    c. activitiesPanel
    d. mentorPanel
    e. buttonPanel

   We will add components to each JPanel later in the lab.
4. Add the following JSeparator objects from the Swing Controls Palette:
    a. activitiesSeparator
    b. mentorSeparator
    c. buttonSeparator

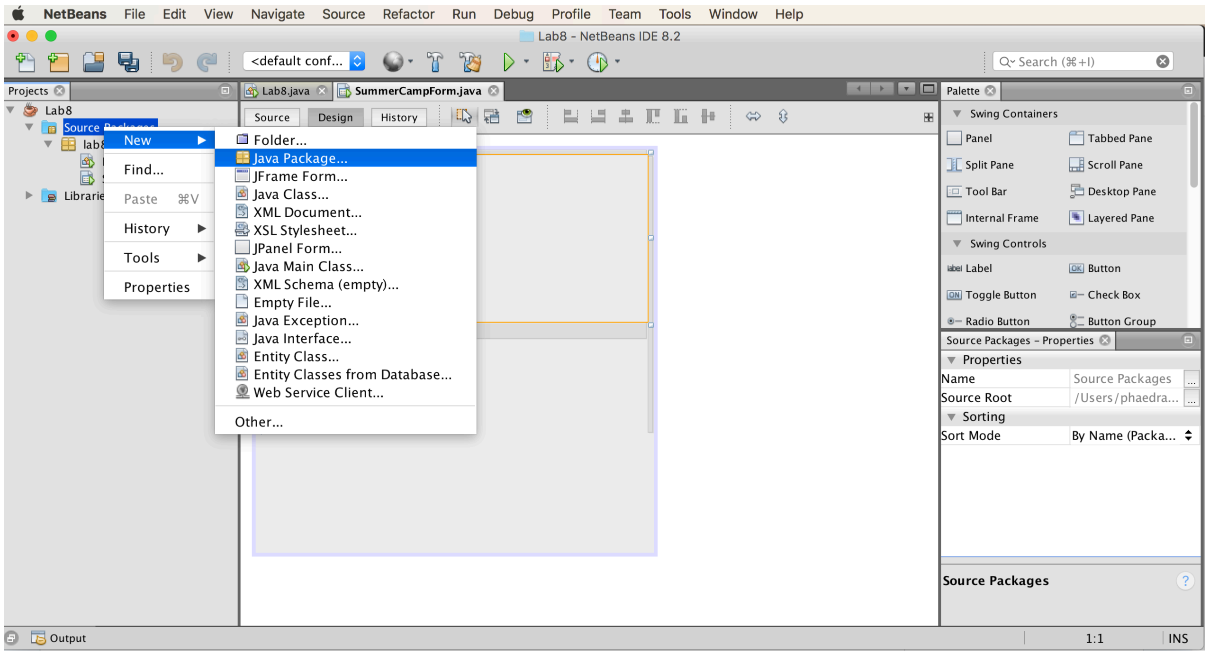
   Ensure that the JSeparator objects are between the respective panels on the Frame itself and not actually within any JPanel. The JSeparator objects are purely for aesthetic purposes.



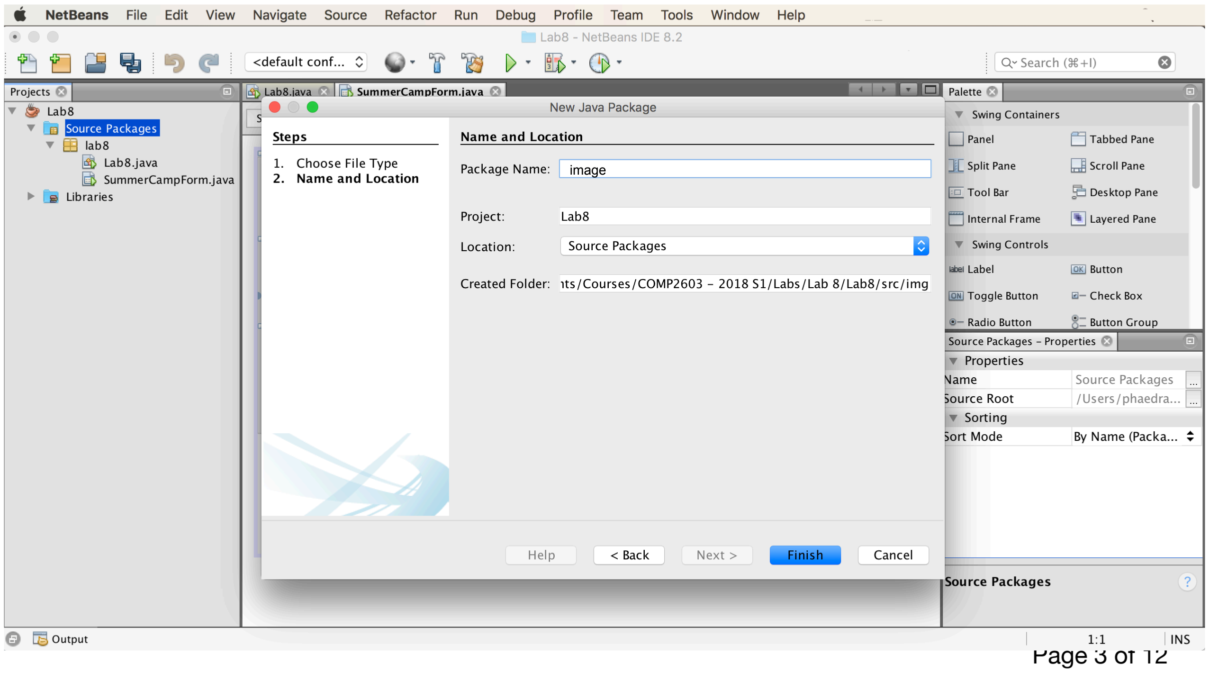
Your design does not have to be exactly to scale as the one shown above. It only needs to have the components specified.

Part 2: Adding a JLabel with a Picture

1. Retrieve the image file called banner.jpg from the Lab 8 on myElearning. Save it on your Desktop.
2. In the Design View, add a **JLabel** called **bannerLabel** to the **bannerPanel.** Leave the default value as jLabel1 for now.
   - Before we can set the content of the bannerLabel object to the banner.jpg image file, we need to create a folder in the Netbeans project that will be used to store the image file.
3. Create a new java package by clicking Source Package -> New -> Java Package



4. Set the package name to **image**


New Java Package

Steps
1. Choose File Type
2. Name and Location

Name and Location

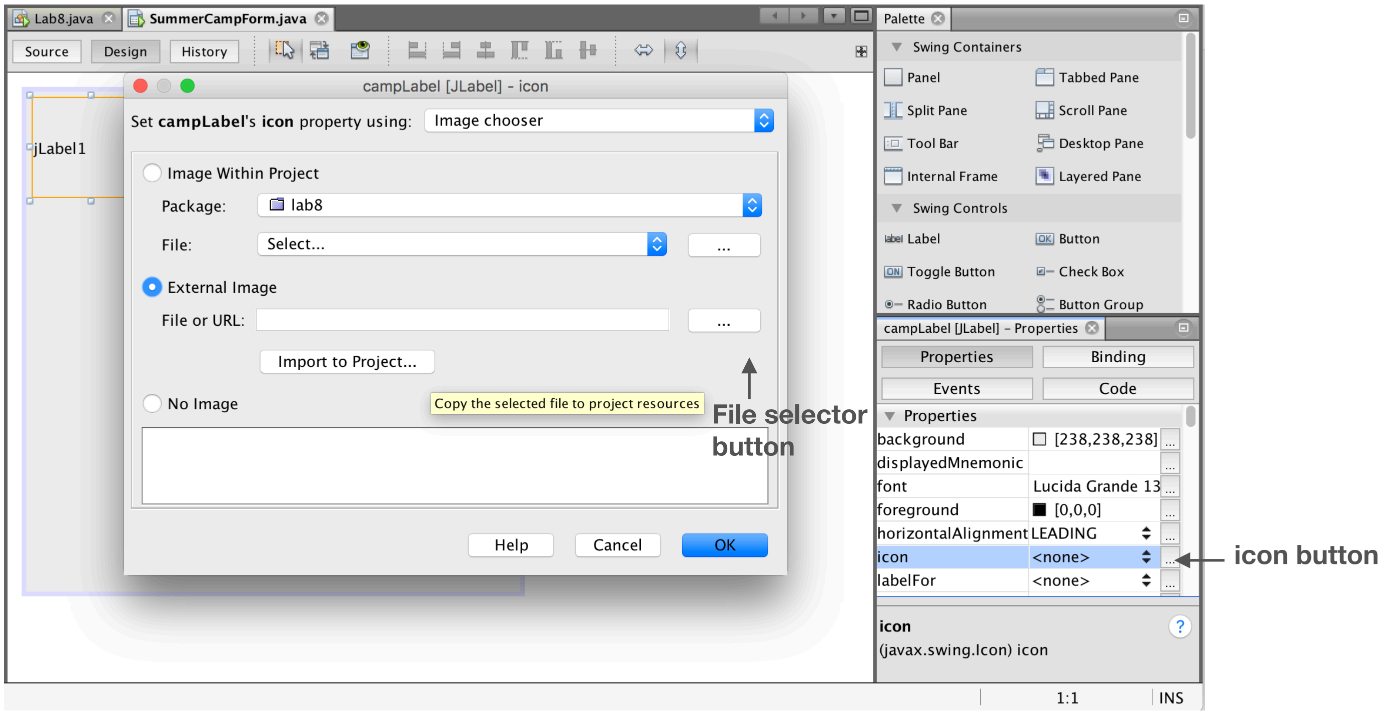
Package Name: image

Project: Lab8

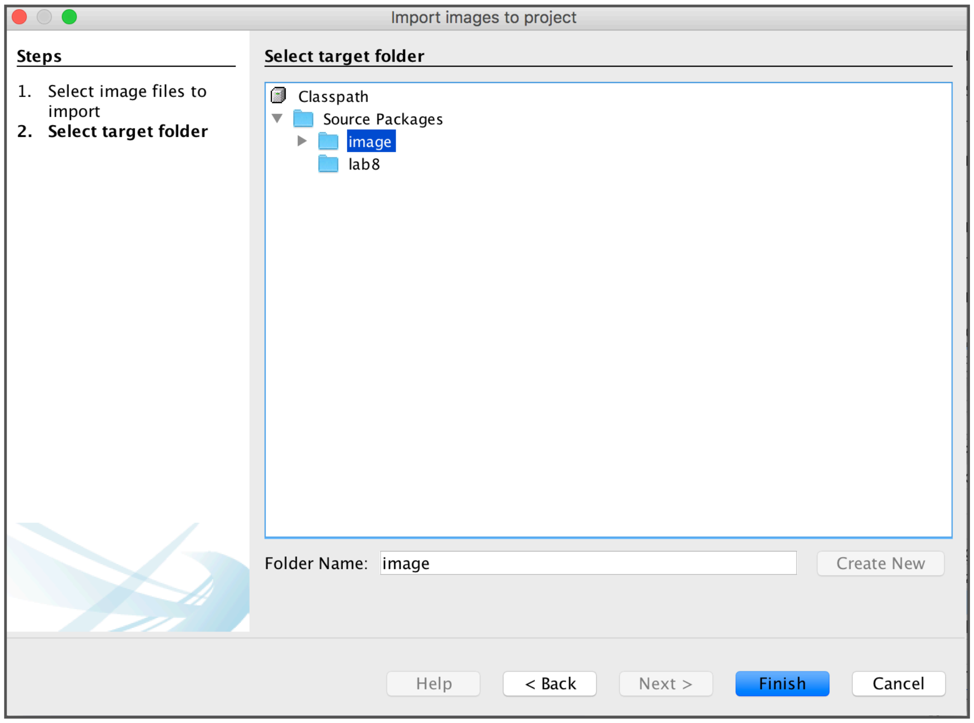
Location: Source Packages

Created Folder: nts/Courses/COMP2603 – 2018 S1/Labs/Lab 8/Lab8/src/img

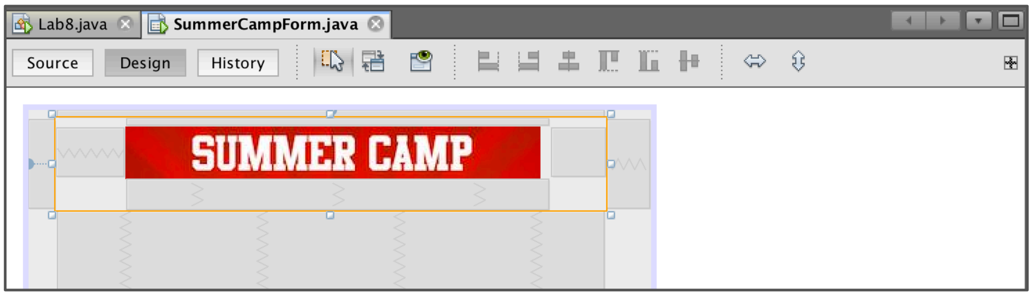
5. Select the **bannerLabel** object and then click on the **icon** button in the **Properties** window.
    a. Select the **External image** option
    b. Click the **File selector** button and navigate to the location of the banner.jpg file on your Desktop.



c. Ensure that the target folder is the **image** folder. Click Finish. Then click on the **Import to Project** button when done.

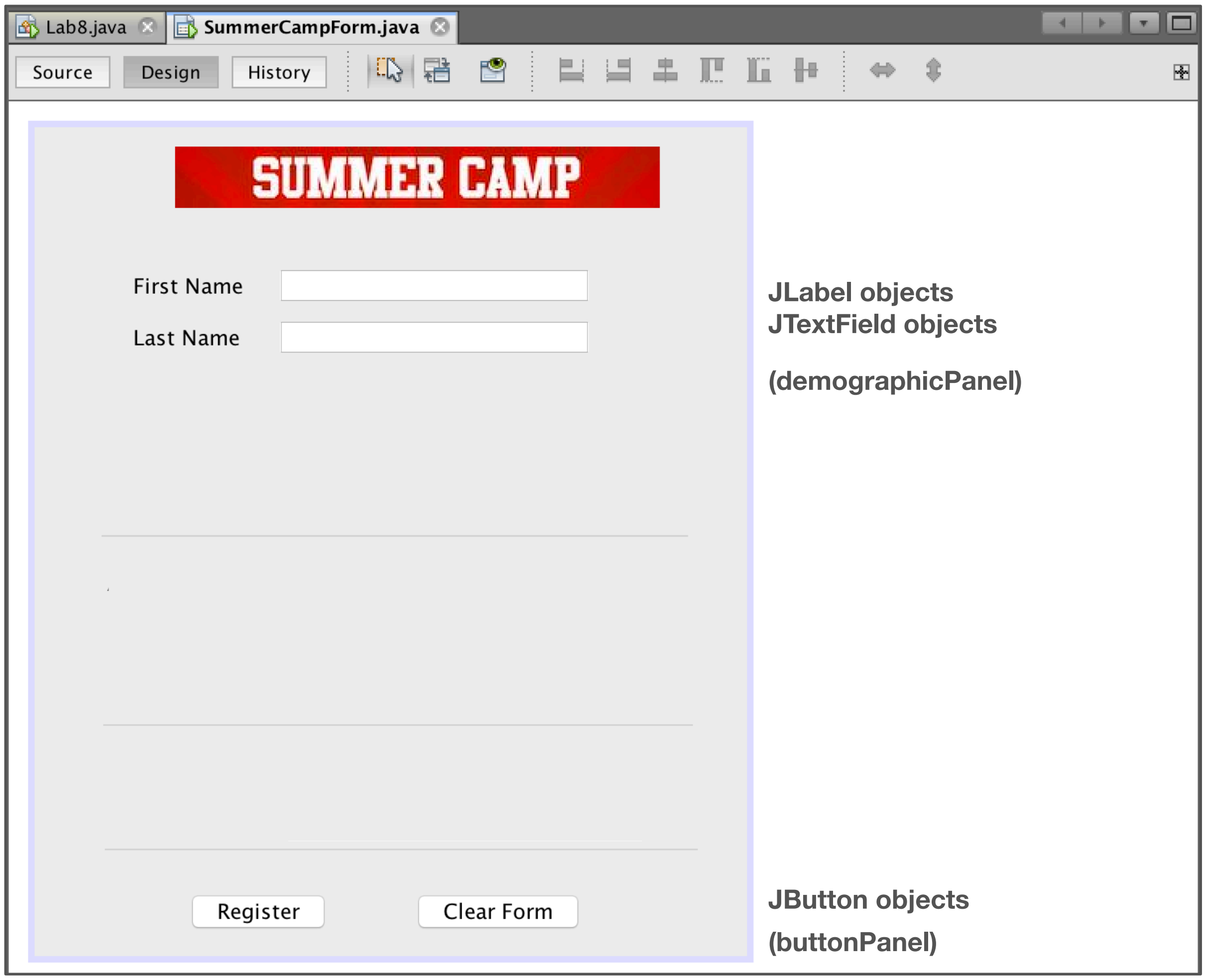


6. Resize and center the JLabel object as necessary within the bannerPanel.



Part 3: Adding Simple Components

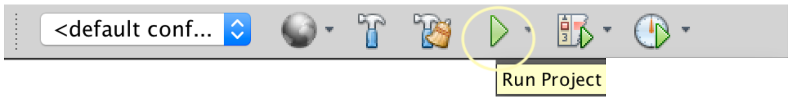
1. Add the following components to the demographicPanel and the buttonPanel.



2. Switch to the Source View of the **Lab8.java** class. Type the following code in the main method of the class to generate a GUI instance:

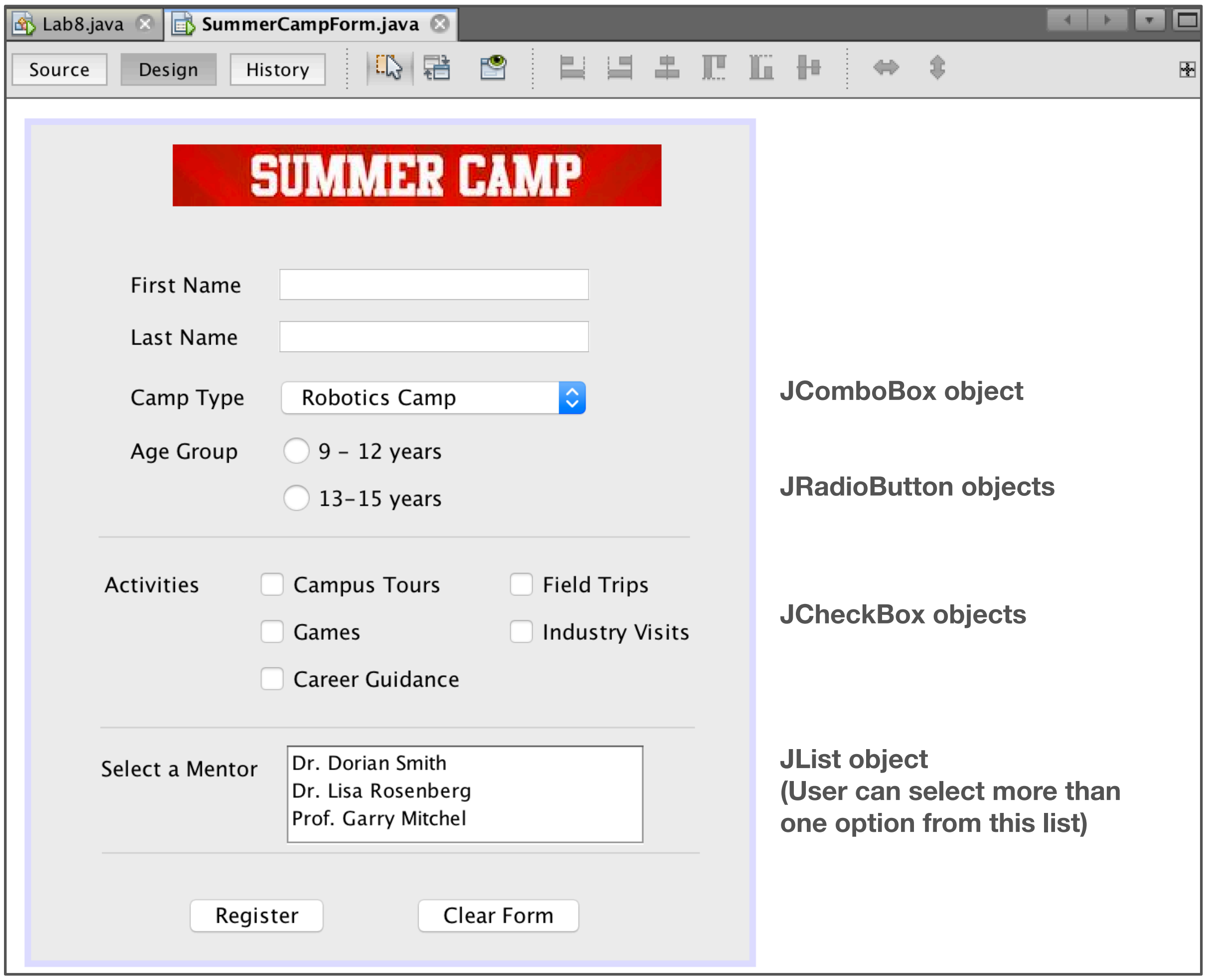
```
SummerCampForm gui = new SummerCampForm();
gui.setVisible(true);
```

3. From the **Lab8** class, run the Project by clicking on the green arrow.



Part 4: Adding Advanced Components

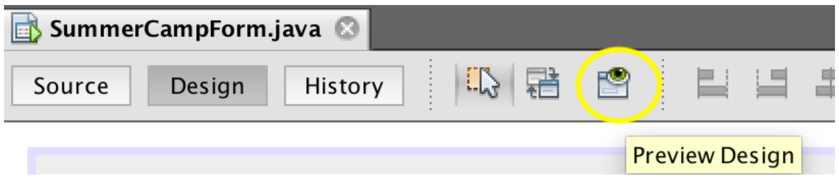
1. Add the advanced components shown below to the remaining **JPanel** objects.



2. The **JComboBox** object should have the following three options in its data model:
   - Robotics Camp, Web Design Camp, Algorithms Camp

   The data can be set using the **model** option in the Properties Window.
3. The **JList** object should have the following three mentors in its data model for now:
   - Dr Dorian Smith, Dr. Lisa Rosenberg, Prof. Garry Mitchel

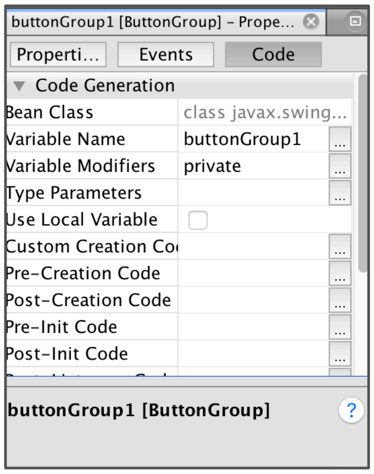
   The data can be set using the **model** option in the Properties Window.
4. Preview how the GUI will look by clicking on the Preview Design button.



Are you restricted from selecting both options for the age group? Why not?

Answer:

5. The **JRadioButton** objects should be added to a **ButtonGroup** object so that users are allowed to select only one of the two options for age group.
   a. Add a **ButtonGroup** object to the **JFrame** from the **Swing Controls Palette.** You will not see a visual component on the **JFrame.**
   b. Click on the **Code** button in the **Properties** Window.
   c. Change the variable name of the **ButtonGroup** object to **ageButtonGroup**.


buttonGroup1 [ButtonGroup] – Prope...

Properti... | Events | Code

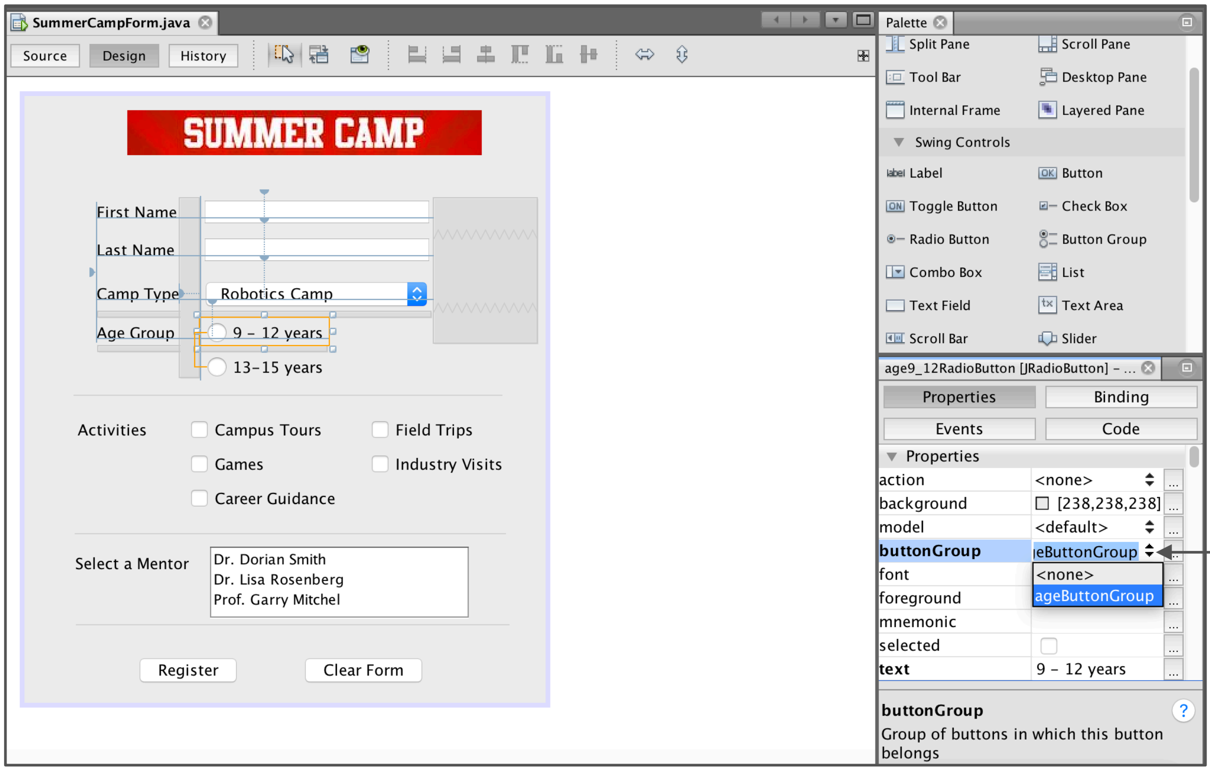
Code Generation

Bean Class	class javax.swing...
Variable Name	buttonGroup1
Variable Modifiers	private
Type Parameters	
Use Local Variable	☐
Custom Creation Co	
Pre-Creation Code	
Post-Creation Code	
Pre-Init Code	
Post-Init Code	

buttonGroup1 [ButtonGroup]

change variable name

6. We need to add the **JRadioButton** objects to the **ageButtonGroup** object.
   a. Select the first **JRadioButton** object. In the **Properties** window, set the **buttonGroup** to **ageButtonGroup.**
   b. Repeat for the second **JRadioButton** object.



age9_12RadioButton [JRadioButton] – ...

Properties | Binding | Events | Code

Properties

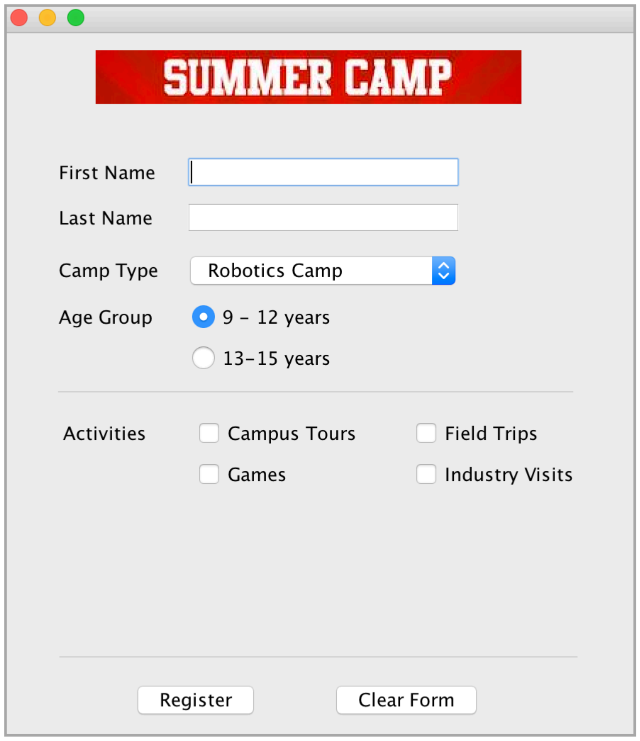
action	<none>
background	☐ [238,238,238]
model	<default>
buttonGroup	ageButtonGroup
font	<none>
foreground	ageButtonGroup
mnemonic	
selected	☐
text	9 – 12 years

buttonGroup
Group of buttons in which this button belongs

set button group

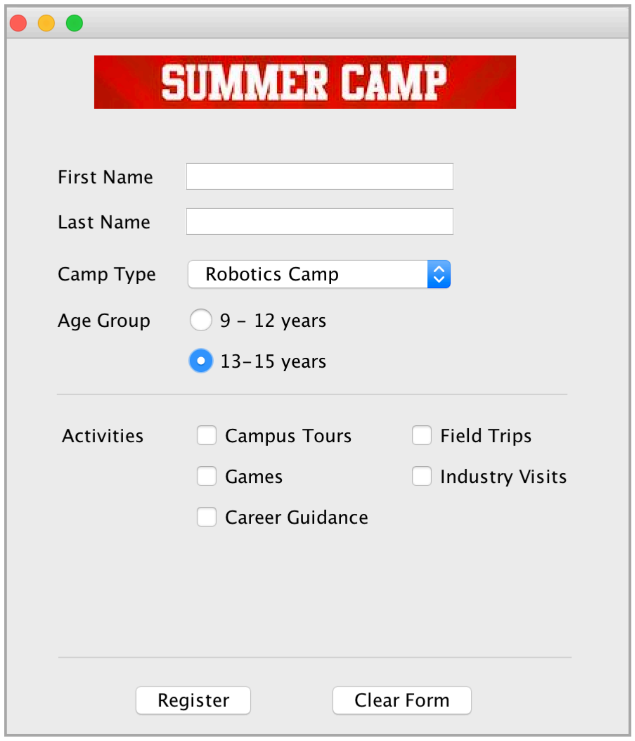
Part 5: Adding Mouse Listeners

A **MouseListener** allows a **MouseEvent**, initiated by the user's manipulation of the mouse, to be tracked. Examples include: **mouseClicked, mouseEntered, mouseExited, mousePressed, mouseReleased.**



First Name:
Last Name:
Camp Type: Robotics Camp
Age Group: [✓ 9 – 12 years] 13–15 years
Activities: ☐ Campus Tours ☐ Field Trips ☐ Games ☐ Industry Visits

Figure 2: Default Screen



First Name:
Last Name:
Camp Type: Robotics Camp
Age Group: 9 – 12 years [✓ 13–15 years]
Activities: ☐ Campus Tours ☐ Field Trips ☐ Games ☐ Industry Visits ☐ Career Guidance

Figure 3: Additional Check-box (Career Guidance) is displayed when 13-15 years Age Group option is selected.

1. Which GUI component would have to be monitored in order to implement the transition from Figure 2 to Figure 3? Which **MouseEvent**, associated with the GUI component you identified, would trigger the screen transition?

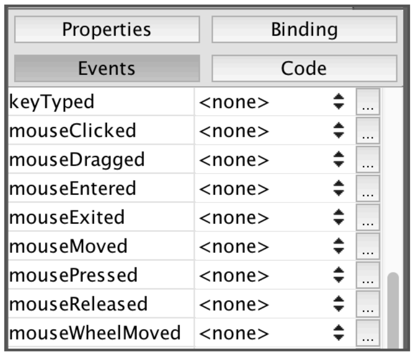
Answer:

2. Switch to Source View of the **SummerCampForm** class. Write code to make the the Default Screen look as presented in Figure 2 when the GUI is launched.
   - The **Career Guidance** check box hidden.
   - The entire mentorPanel (JPanel object) is hidden.
   - The 9-12 age group is selected by default.
   - The Robotics Camp option is selected by default.

TIP: Google the Java API of any class to find out more about the methods that can be used

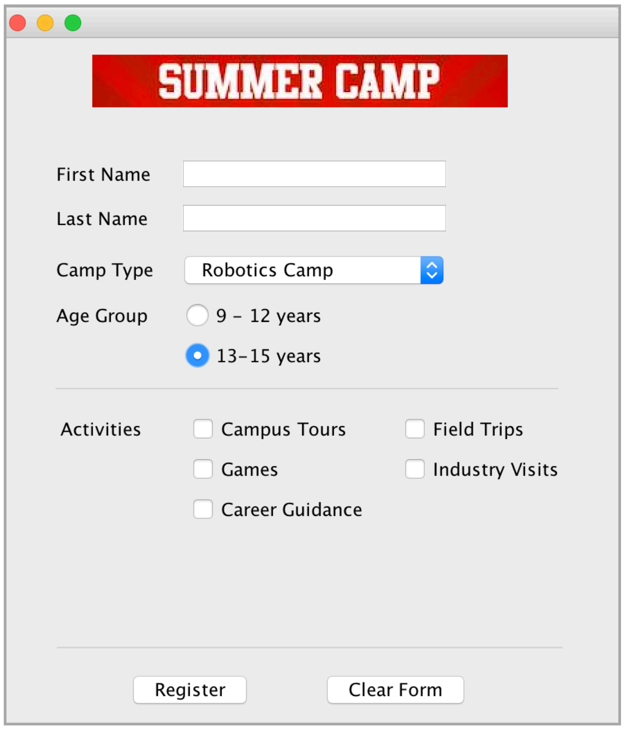
JCheckBox
Panel

3. Create the appropriate MouseListener object for the component you identified in Part 5, Question 1.
   - This can be done in the Netbeans GUI Builder using the Properties Window.
   - Simply activate the appropriate drop-down list corresponding to the Event.
   - The Source View will be automatically presented.
   - Write the code to implement the transition from Figure 2 to 3.



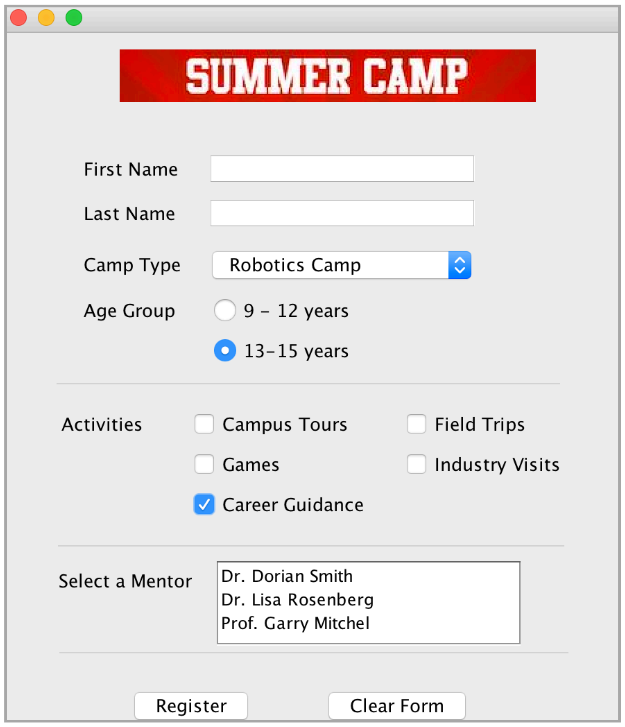
keyTyped	<none>
mouseClicked	<none>
mouseDragged	<none>
mouseEntered	<none>
mouseExited	<none>
mouseMoved	<none>
mousePressed	<none>
mouseReleased	<none>
mouseWheelMoved	<none>

4. Write the code to implement the transition shown in Figure 3 to that of Figure 4.



First Name
Last Name
Camp Type: Robotics Camp
Age Group: ( ) 9 – 12 years [✓ 13–15 years]
Activities: [ ] Campus Tours [ ] Field Trips [ ] Games [ ] Industry Visits [ ] Career Guidance

Figure 3: Screen when 13-15 years is selected
(Repeated for convenience from Page 11)



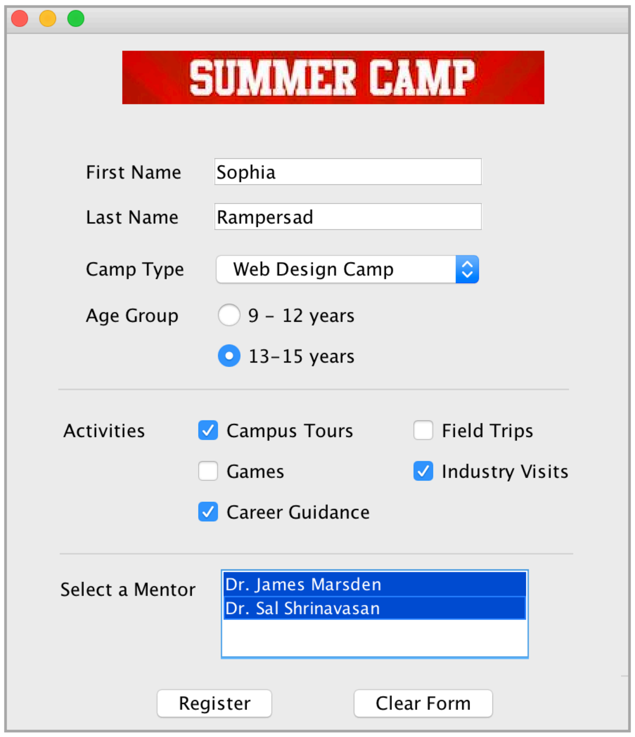
First Name
Last Name
Camp Type: Robotics Camp
Age Group: ( ) 9 – 12 years [✓ 13–15 years]
Activities: [ ] Campus Tours [ ] Field Trips [ ] Games [ ] Industry Visits [✓ Career Guidance]
Select a Mentor: Dr. Dorian Smith / Dr. Lisa Rosenberg / Prof. Garry Mitchel

Figure 4: The entire mentorPanel (JPanel) is made visible when the Career Guidance option is selected

Work out the steps to get this done.

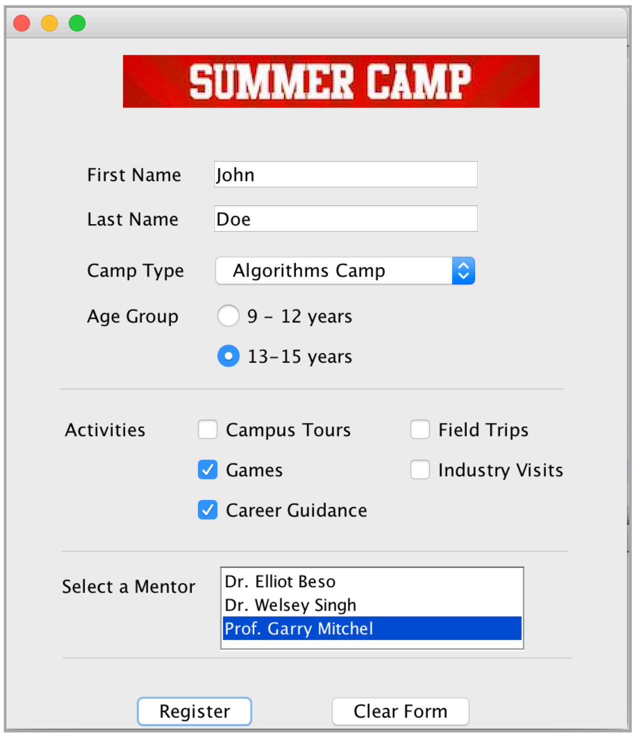
Answer:

5. Write the code to implement the transition shown in Figure 5 and Figure 6 when a different **Camp Type** is selected.



First Name: Sophia
Last Name: Rampersad
Camp Type: Web Design Camp
Age Group: ○ 9 – 12 years [✓ 13–15 years]
Activities: [✓ Campus Tours] Field Trips Games [✓ Industry Visits] [✓ Career Guidance]
Select a Mentor: [✓ Dr. James Marsden] [✓ Dr. Sal Shrinavasan]

Figure 5: The Mentors for the Web Design Camp are Dr. James Marsden, and Dr. Sal Shrinavasan



First Name: John
Last Name: Doe
Camp Type: Algorithms Camp
Age Group: ○ 9 – 12 years [✓ 13–15 years]
Activities: Campus Tours Field Trips [✓ Games] Industry Visits [✓ Career Guidance]
Select a Mentor: Dr. Elliot Beso, Dr. Welsey Singh, [✓ Prof. Garry Mitchel]

Figure 6: The Mentors for the Algorithms Camp are Dr. Elliot Best, Dr. Welsey Singh and Prof. Garry Mitchel

Which GUI component would have to be monitored in order to implement the transition from Figure 5 to Figure 6? Which **ActionEvent**, associated with the GUI component you identified, would trigger the screen transition?

Answer:

Work out the steps to get this done.

Answer:

TIP: Google the Java API of any class to find out more about the methods that can be used

JList

Part 6: Adding Action Listeners

The GUI should have some functionality when the buttons are clicked, namely, data collection for the **Register** button and clearing the components and resetting the form for the Clear button.

1. Attach an **ActionListener** object to the **Register** Button.
2. Write code for the **actionPerformed( )** method of the **Register** Button **ActionListener** that:
    - Collects the data from all of the visible components on the screen.
    - Stores the data in variables

3. Attach an **ActionListener** object to the **Clear** Button.
4. Write code for the **actionPerformed( )** method of the **Clear** Button **ActionListener** that:
    - Resets the form to the default presentation of elements (Figure 2)
    - Clears the data from the visible components.

Part 7: Connecting the GUI to the Domain Object

1. Create a new Java class called **Camper** in your Project. This will be used to create objects that store data collected from the GUI.
2. Paste the following code in the class body. Ensure the package declaration (`package lab8;`) to the top of the class is not altered.

```
public class Camper {
    private String firstName;
    private String lastName;
    private String campType;
    private String ageGroup;
    private java.util.ArrayList<String> activities;
    private java.util.ArrayList<String> mentors;

    public Camper(String fName, String lName, String cType){
        firstName = fName;
        lastName = lName;
        campType = cType;

        activities = new java.util.ArrayList<String>();
        mentors = new java.util.ArrayList<String>();
    }
    public void setAgeGroup(String ageGp){
        ageGroup = ageGp;
    }

    public void setActivities(java.util.List<String> acts){
     if(acts != null)
         activities = new java.util.ArrayList<String>(acts);

    }
```

```
        public void setMentors(java.util.List<String> ments){
        if(ments != null)
             mentors = new java.util.ArrayList<String>(ments);
        }

        public String toString(){
         String s= "";
         s+= firstName + " " + lastName + " " + campType + " " + ageGroup + "\n";
         s+= "Activities: " + activities.toString() + "\n";
         s+= "Mentors: " + mentors.toString() + "\n";
         return s;
        }
    }
```

3. Modify the **actionPerformed( )** method for the **Register** button so that:
    a. All of the data collected from the GUI is used to create a **Camper** object
    b. The **Camper** object is inserted into an **ArrayList** of campers who have registered
    c. The details of all the campers registered so far are printed out to the console

Additional Exercise

Create another class called **CampSystem** that abstracts the GUI away from the domain model and provides a single point of execution from the Lab8 main class. It essentially serves as the business layer in the three-tiered architecture. The Lab8 class may therefore only have one statement in the main method eg. CampSystem.execute( ).