CS 161-01 Spring Semester 2018

Lab 9 – Lab 10: Menu and Paints
40 pts

posted: 04/09/2018

due: 04/21/2018

last day of submission 04/28/2018

Learning Outcomes

- Design and implement multi-class solutions to programming problems (b, c, i)
- Apply inheritance and polymorphism (b, c, i)
- Utilize event driven programming and interfaces in advanced GUI applications (b, c, i)

Preliminaries

Schedule your work: do the GUI with menu on the lab of 04/09 – 04/11; code the part of painting figures next week.

1. Create a NetBeans or Eclipse Java project. Name the project as **<your-ipfw-id>_CS161_lab9-10**.
2. Add the Java classes named **Menus** and **Test** to your project, the last will be the main class.
3. Add the following comment block to the beginning of your Java classes (necessary for getting a grade):

```
/*
 * <your name>
 * CS161
 * Spring 2018
 * Lab 9-10
 */
```

4. Add another comment block shortly describing what task your class is responsible for (see the description of the assignments below).

Exercises.

1. In this lab you implement a GUI (the Menus class) as **Figure 1** shows. Note that for the sake of saving space in this document the vertical size is about 1/3 of the actual window.
2. The basic task of the GUI is as follows:
    - The user selects an Item from the 'Figures' menu, say 'oval' is selected, see **Figure 2**
    - Next the user selects a color from the 'Colors' menu, say the radio button with color blue, see **Figure 3**
    - The user clicks the button 'PAINTBUTTON' which makes the colored shape appear in the center region, see **Figure 4**.

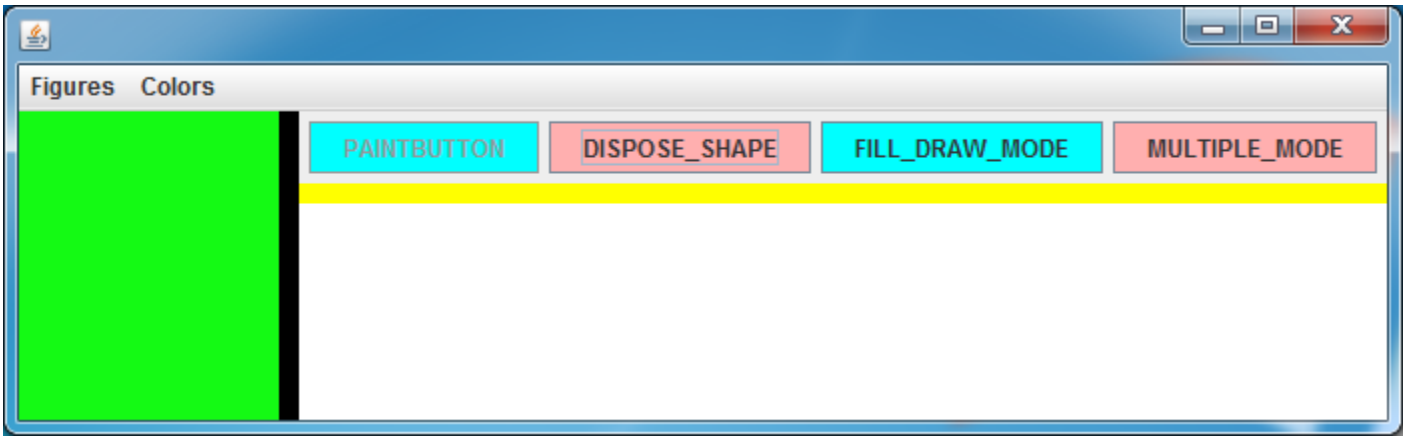

Figure 1

Figure 2

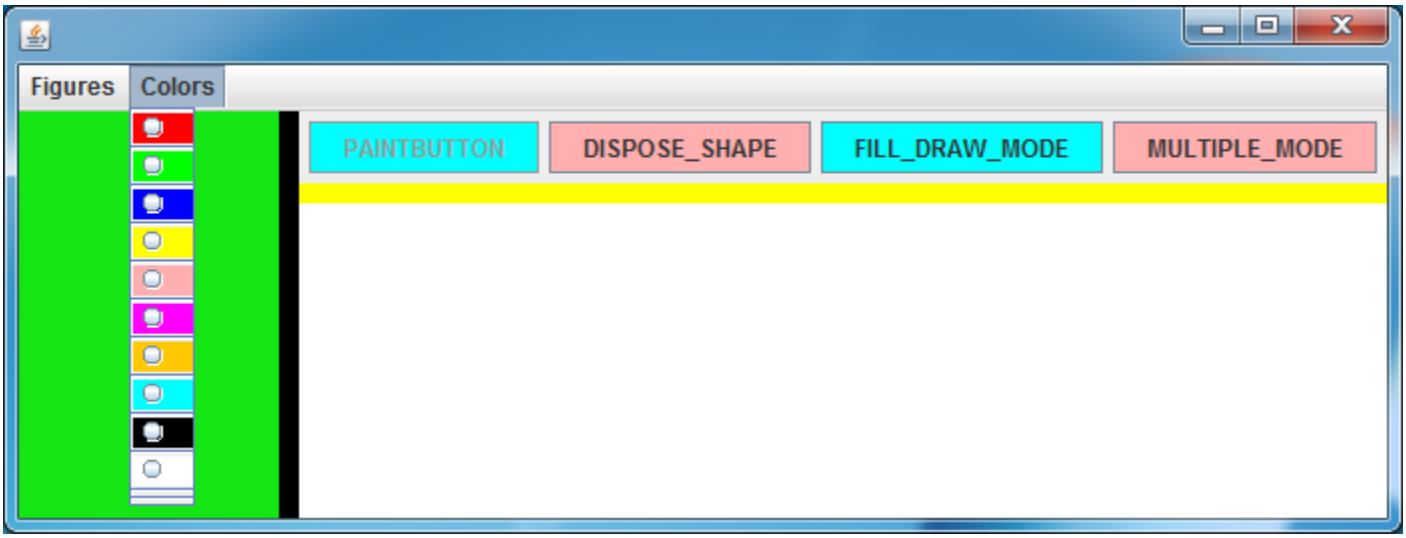

Figure 3

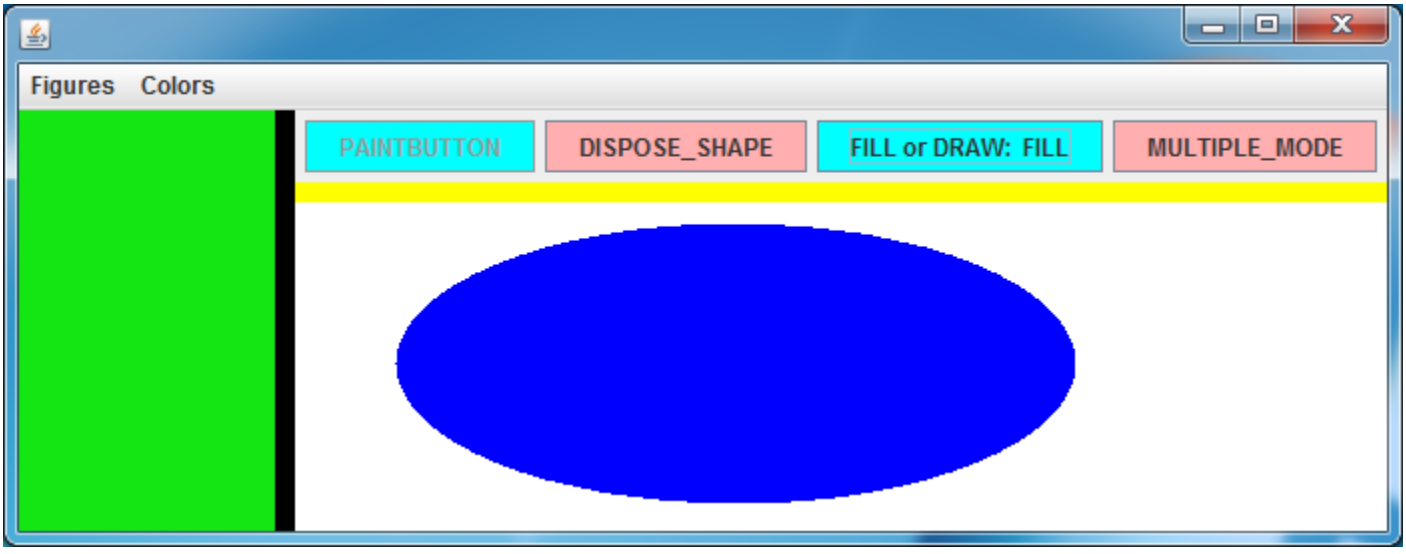

Figure 4

3. Components as shown in the Figures above and their functionalities
    - The MenuBar is the top of the frame
    - The bar contains the Menus named Figures and Colors; the menus dropped down are shown in Figure 2 and Figure 3.
    - The CENTER region and the EAST region of the content pane is used only. The background of the pane is black, the BorderLayout is reset with gaps.
    - The green panel on the center region is just a space holder for the two Menus, it has no other task.
    - The east panel sets a nested BorderLayout with gaps, the NORTH region and the CENTER region is in use only
    - The subpanel in the north carries four control buttons, each has its useful action in the painting process. The buttons with their texts also control the horizontal size of the east panel, relative to the size of the center panel.
    - The central subpanel of the east panel is the one subject to carry the painted shapes, see Figure 4.
    - The button functionalities are detailed below in the Implementation section.

4. **Menus** class implementation hints and requirements
    - (2 pts) Field declarations and initializations
      -- Using initializer lists define two arrays:
      -- a String array **figureItems** for listing the names in the Figures menu, these will be **JMenuItem** objects
      -- a Color array **colors**, listing the color constants shown in the Colors menu. The items in Colors will be **JRadioButtonMenuItem** objects and those carry the color as their **background** values
      -- Declare a String field **selectedShape** and a Color field **selectedColor** to store the menu selections
      -- Declare a JMenuBar reference and two JMenu references as fields
      -- Declare **and instantiate** a listener field from both inner classes

    - (4 pts) Define two inner classes **FigureListener** and **ColorListener.** Each implements ActionListener and defines its actionPerformed( ) method.
      Note that in these methods you do not need if statements, since the same action applies for any selection: retrieve the **value** from the selected item and forward it to the fields **selectedShape** and **selectedColor**, respectively. Use the **getText()** method for the item in the Figures menu, and here is the code to retrieve the color from the selected item of the Colors menu::
      `((JRadioButtonMenuItem) e.getSource()).getBackground()`
      Observe that type cast is needed for all selected items in either menu.

    - (2 pt) Define a helper method to build the Figures menu. Instantiate the menu with its name and run a for loop to instantiate a JMenuItem item for **each** name in the figureItems array. Register with the item its listener (see the previous bullet) and add the items to the corresponding menu

    - (2 pt) Do the same as above for the Colors menu

- (2 pt) Define a helper method to build the JMenuBar object and using the setter add the bar to the frame.
- (12 pts) Define an inner class named PaintOnPanel that extends JPanel. Therefore a PaintOnPanel object is a JPanel by inheritance and this will be the painted center subpanel to be added to the east panel. Define the library paintComponent(Graphics g) method such that it runs a **switch** selection structure to execute the painting as required by the menu selections. The control is the string selectedShape, the cases are the menu items in Figures. Set the color of the Graphics object to the selected color.

- (1 pt) The PAINTBUTTON is disabled by default (it is in the wait for the menu selections). After the color selection the actionPerformed( ) method enables this button.

- (1 pt) The FILL_DRAW_MODE button just flips a boolean data field of **Menus** to its opposite. The **switch** structure in the paintComponent method applies an **if** selection in the appropriate cases to choose shape filling or shape drawing

- (1 pt) The MULTIPLE_MODE button also flips a boolean data field which decides to apply (true) or ignore (false) the break statement in the switch cases. Without break, all or many cases execute and we get a rich canvas, see **Figure 5 below**. Note that the wider horizontal line is a triple drawing with 1 pixel vertical change.

- The MODE buttons above change their own title when clicked according to the boolean selection

- (1 pt) The DISPOSE_SHAPE button "eliminates" all the paintings by repainting everything to the background color WHITE.

- (12 pts) Build the frame with the menus as described above. Decide the frame dimensions and the coordinates for the painted shapes by experimental runs of the program. My frame is built to size 700, 650. Make your class extend JFrame and implement ActionListener. In order to activate you buttons define a selective actionPerformed( ) in Menus with the usual if statements for button identification. Register with all buttons **`this`** for their listener.

Submit: Zip your project folder and upload the zip file on Blackboard at your **lab** section

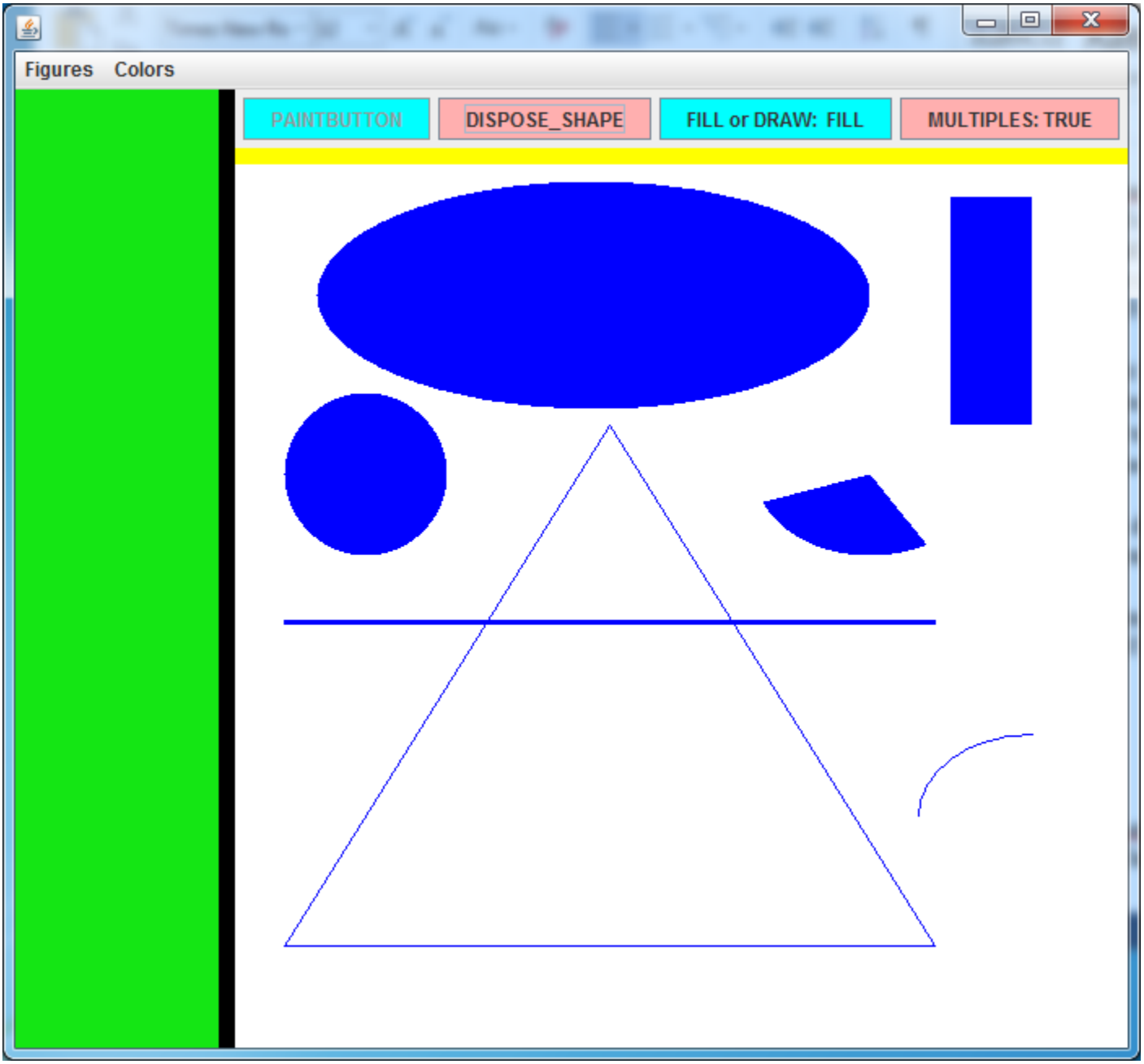
Figures
Colors
PAINTBUTTON
DISPOSE_SHAPE
FILL or DRAW: FILL
MULTIPLES: TRUE

Figure 5