UNIVERSITI TEKNIKAL MALAYSIA MELAKA

Fakulti Teknologi Maklumat dan Komunikasi

BITP 3113
OBJECT ORIENTED PROGRAMMING

LAB:
WEEK 11 – JAVA GUI (SECTION 1)

LECTURER NAME:
DR EMALIANA BINTI KASMURI

STUDENT NAME:
MUHAMMAD AFIQ MUHAIMIN BIN MOHD ZAINI

MATRIC NUMBER:
B032410648

SEMESTER: 2-2024/25

Section 2: Understanding Common Java GUI Components

a. Download the **Week 11 - Section 02 - Supplementary Codes.zip** file from ulearn.
b. Unzip the file.
c. Execute the MainGUIApp.java. The initial output shall be similar as shown in Figure 1.

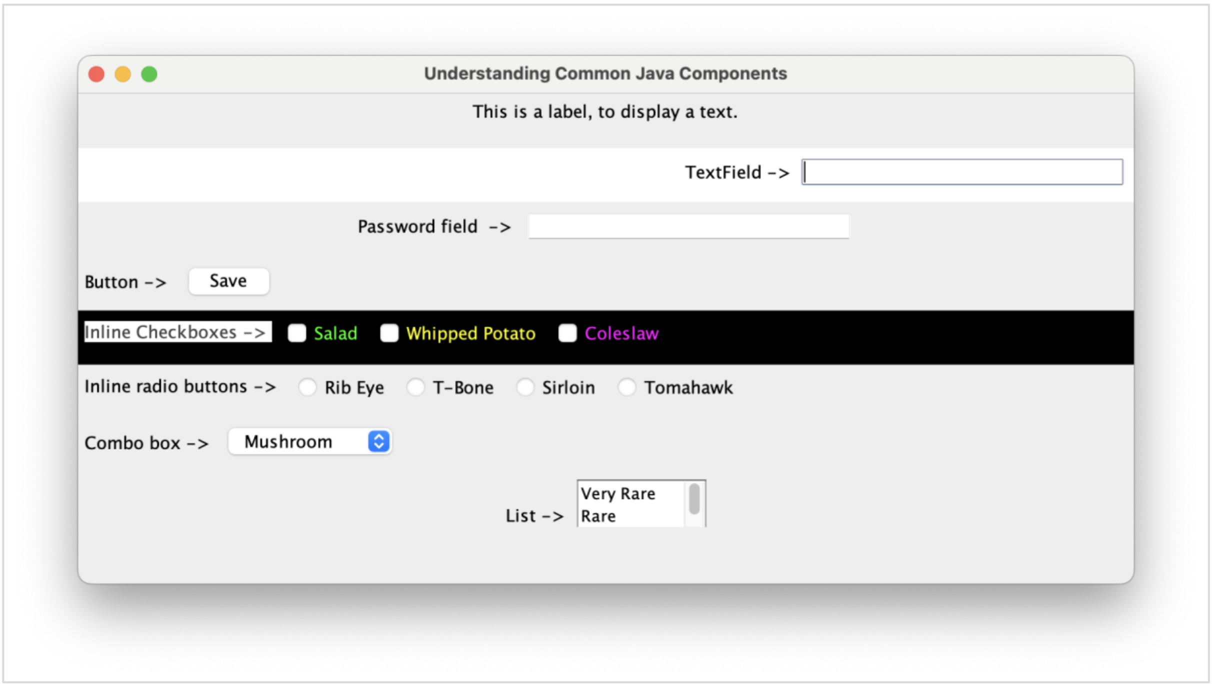


Figure 4: Common GUI components for Java desktop application

Exercise 5: GUI Component Identification

a. Open the **ComponentScreen.java** file.
b. Observe the code and the output from the execution.
c. Identify the object and its constructor for each of the GUI components loaded on the screen from MainGUIApp.java execution. Record the answer in Table 2.

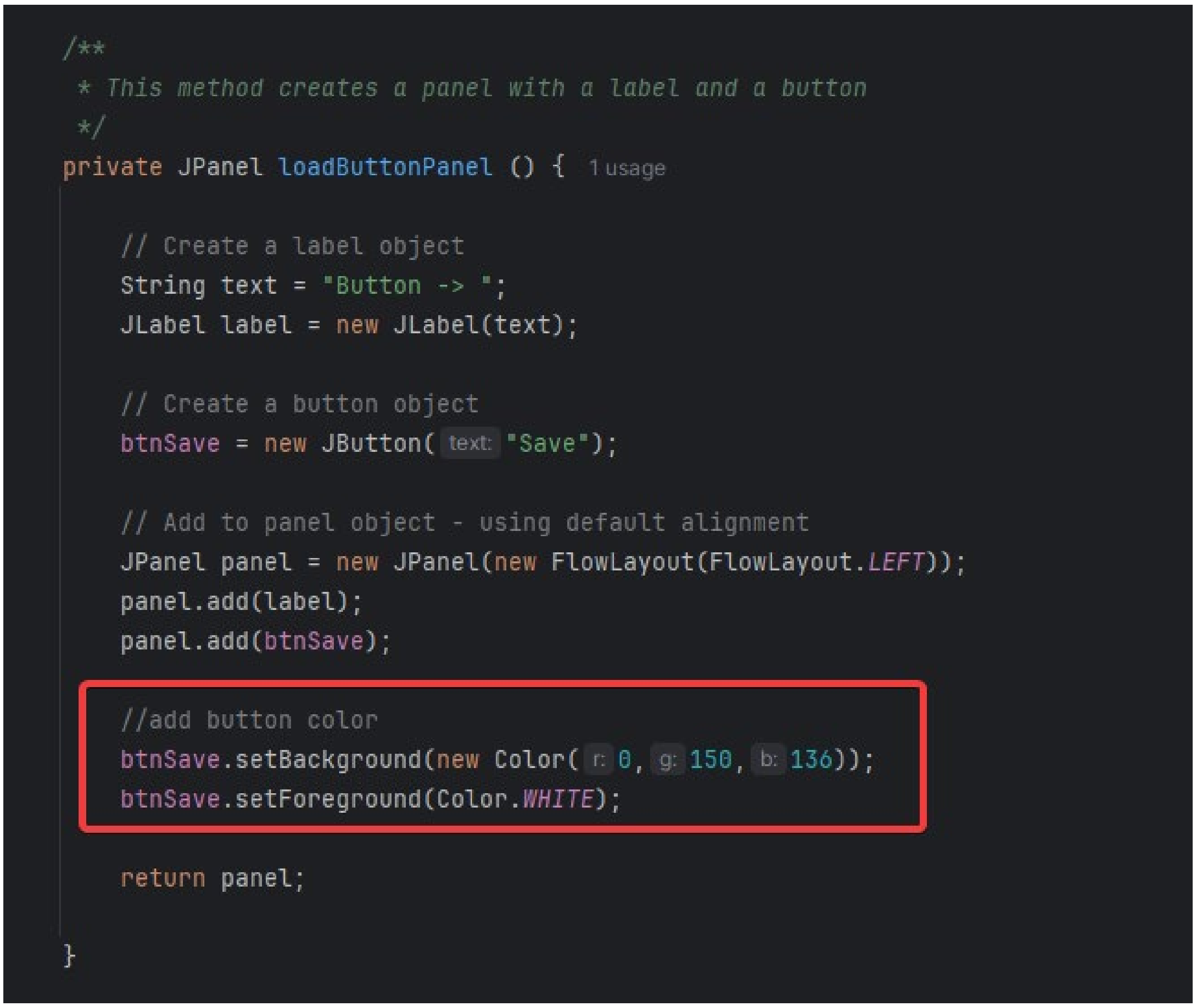
Table 2: GUI component observation results

GUI Component	Object	Constructor
JLabel	Label	new JLabel(String text)
JCheckBox	cbWhippedPotato	new JCheckBox("Whipped Potato")
JRadioButton	rbTomahawk	new JRadioButton("Tomahawk")
JList<String>	lstDoneness	new JList<String>(doneness)

Exercise 6: Overriding GUI Component Color

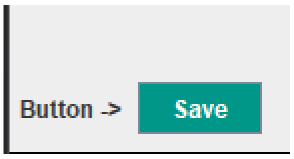
a. **Describe how should a programmer change the color for the GUI components.**
   Programmer can change the color of the GUI using the method ***setBackground(Color color)*** **to change the background color and** ***setForeground(Color color)*** **to change the text or foreground color**

b. Based on your description, change the default color for one of the GUI components to the color of your choice.



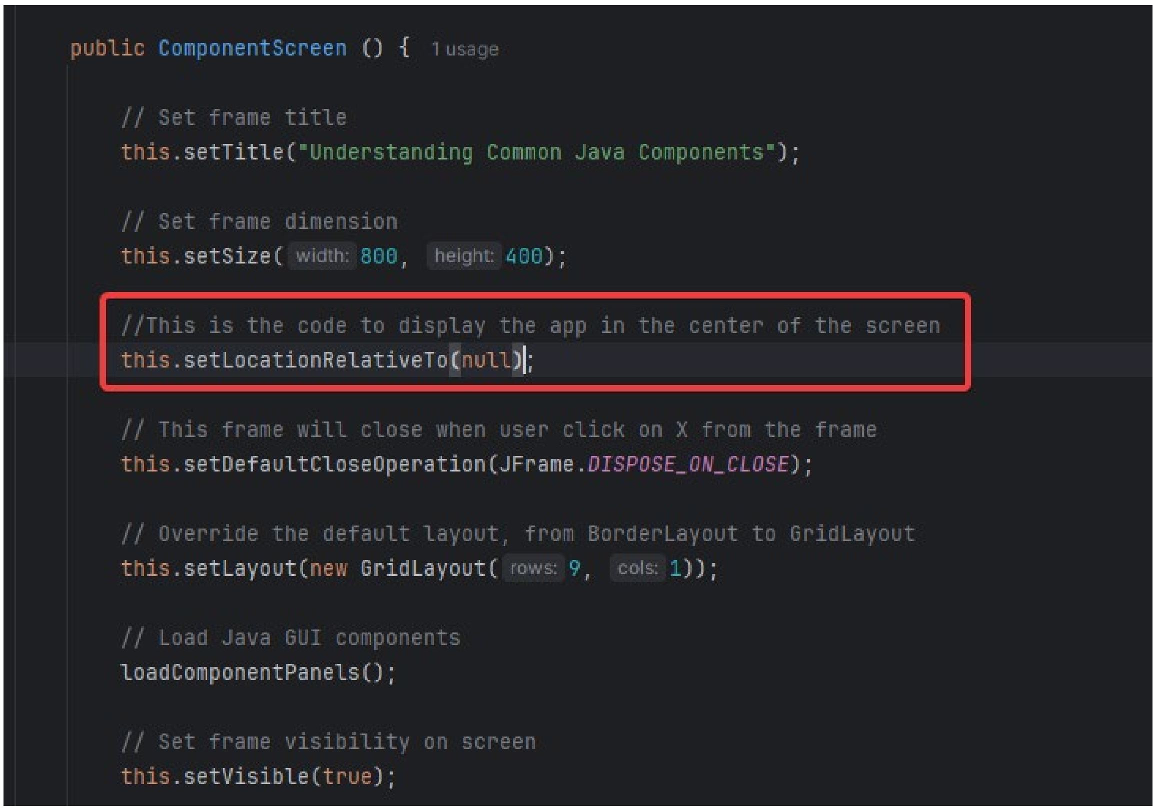
```
/**
 * This method creates a panel with a label and a button
 */
private JPanel loadButtonPanel () {  1 usage

    // Create a label object
    String text = "Button -> ";
    JLabel label = new JLabel(text);

    // Create a button object
    btnSave = new JButton( text: "Save");

    // Add to panel object - using default alignment
    JPanel panel = new JPanel(new FlowLayout(FlowLayout.LEFT));
    panel.add(label);
    panel.add(btnSave);

    //add button color
    btnSave.setBackground(new Color( r: 0, g: 150, b: 136));
    btnSave.setForeground(Color.WHITE);

    return panel;

}
```

Output



Exercise 7: Relocate Initial Screen Display

a. Amend the class to display the screen in the center of the screen.
b. Highlight the code to show your solution.



```
public ComponentScreen () {  1 usage

    // Set frame title
    this.setTitle("Understanding Common Java Components");

    // Set frame dimension
    this.setSize( width: 800,  height: 400);

    //This is the code to display the app in the center of the screen
    this.setLocationRelativeTo(null);

    // This frame will close when user click on X from the frame
    this.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);

    // Override the default layout, from BorderLayout to GridLayout
    this.setLayout(new GridLayout( rows: 9,  cols: 1));

    // Load Java GUI components
    loadComponentPanels();

    // Set frame visibility on screen
    this.setVisible(true);
```

Exercise 8: Layout Manager Identification

a. Identify the classes that contains the term Layout from the code.
b. These are known as layout manager classes. A layout manager class arrange the GUI components layout on a panel or frame. List the layout object and classes in the in Table 3.

Table 3: Layout manager class observation results

Object	Package	Constructor
flowLayout	java.awt	new FlowLayout()
gridLayout	java.awt	new GridLayour(9, 1)
borderLayout	java.awt	new BorderLayout()

Exercise G: Observing Layout Manager Behavior

The purpose of layout manager is to hold a group of GUI components and arrange it according to layout policy design for a particular manager class.

a. Observe the arrangement of GUI components in each panel.
b. Record your observation in Table 4.

Table 4: Panel arrangement observation results

Panel	Observation
Label panel	Contains a single label centered horizontally with the sentence "This is a label, to display a text" using *FlowLayout*
Text field panel	The label for the text field and the box itself is aligned to the right using *FlowLayout.RIGHT*
Password field panel	Label and password field arranged in centered format which is default
Button panel	The label "Button ->" and the Save button is aligned to the left using FlowLayout.LEFT

Check boxes panel	Label “Inline Checkboxes ->” and the button (Salad, Whipped Potato, Coleslaw) are aligned to the left with custom color for the text
Radio buttons panel	Label “Inline radio buttons ->” and the button (Rib eye, T-Bone, Sirloin, Tomahawk) are arranged horizontally and aligned to the left
Combo box panel	Label “Combo box ->” and the list (Mushroom, Black Pepper , Barbeque) inside the dropdown are placed side by side using default layout
List panel	Label “List ->” and the list (Very Rare, Rare, Medium Rare, Medium, Medium Well, Well Done) inside the dropdown are placed side by side using default layout

Exercise 10: Concluding Layout Behavior

Draw some conclusions based on the observation results in Exercise 9. Record your observation in ulearn.

- **Use of FlowLayout**
  - Most of the panels use FlowLayout as their guideline for arrangement of components. This allows the layout manager to align components in a single row or left/center/right to suit the design of each panel.

- Custom Alignment will enhance readability
  - Panels that are aligned with the ***FlowLayout.RIGHT*** for the **Text Field Panel** and ***FlowLayout.LEFT*** for the **Button Panel** shows that the different alignments of the button can help improve the logical grouping and visual clarity.

- Grouping Similar Components:
  - The use of **ButtonGroup** for radio buttons ensures that only one option can be selected at a time and placing them inline (horizontally) improves space utilization and user experience.

- Scrollable Lists Improve Usability:
  - The List Panel uses a **JScrollPane**, which allows users to scroll through options, making it suitable for data with multiple entries.

- Panel-specific Styling:
  - Some panels, like the **Check Boxes Panel**, apply custom foreground and background colors, enhancing visual appeal and distinguishing between component groups.

End of Document