Discussion-2: Creating Graphical User Interfaces in Java

Discussion Topic:

To create a graphical user interface application program in Java, there are a number of components that can be utilized. Specifically, you can use a frame or a panel to create objects. What are the differences between these two components? Provide an example that illustrates creating a simple GUI with an appropriate frame element. In response to your peers, provide an additional example on the GUI w/frame element that was posted.

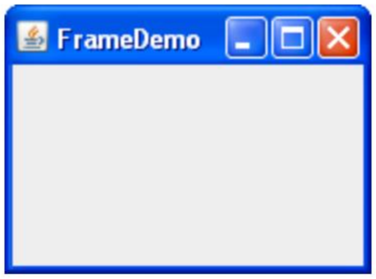
My Post:

Hello class,

Swing, a GUI widget toolkit for Java, provides a collection of components for constructing graphical user interfaces (GUIs). (Oracle Docs, n.d.a) Swing components are written entirely in the Java programming language. There are three generally useful top-level container classes: ***JFrame***, ***JDialog***, and ***JApplet*** (Oracle Docs, n.d.b).

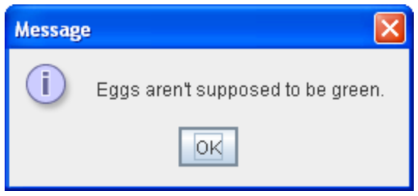
Top-level container:

- A ***JFrame*** is a top-level window with a title and a border.



(Oracle Docs, n.d.b, How to Make Frames (Main Windows)).

- A ***JDialog*** window is an independent sub-window that temporaries notice apart from the main Swing Application Window.



Eggs aren't supposed to be green.

(Oracle Docs, n.d.b, How to Make Dialogs)

- "***JApplet***, a Java applet is a special kind of Java program that a browser enabled with Java technology can download from the internet and run. An applet is typically embedded inside a web page and runs in the context of a browser. An applet must be a subclass of the

java.applet.Applet class. The Applet class provides the standard interface between the applet and the browser environment" (Oracle Docs, n.d.c).

Every program that utilizes Swing components includes at least one top-level container. This top-level container serves as the root of a containment hierarchy, which encompasses all the Swing components within the container (Oracle Docs, n.d.b)..
Typically, a standalone application with a Swing-based GUI will have at least one containment hierarchy with a ***JFrame*** as the root. For instance, if an application features a main window and two dialogs, it will have three containment hierarchies, each with its own top-level container. The main window will have a ***JFrame*** as its root, while each dialog will have a ***JDialog*** as its root.
A Swing-based applet also has at least one containment hierarchy, with one rooted by a ***JApplet*** object. For example, an applet that displays a dialog will have two containment hierarchies. The components within the browser window belong to a containment hierarchy rooted by a ***JApplet*** object, while the dialog belongs to a containment hierarchy rooted by a ***JDialog*** object.

JComponent Class:

Except for top-level containers, all Swing components that start with "***J***" are derived from the ***JComponent*** class. For instance, ***JPanel***, ***JScrollPane***, ***JButton,*** and ***JTable*** all inherit from ***JComponent***. However***, JFrame*** and ***JDialog*** do not, as they are top-level containers (Oracle Docs, n.d.b, The JComponent Class)

Differences between Frame and Panel:

- Frame:
    - A ***JFrame*** is a top-level container that represents a window with a title, borders, and buttons.
    - It is typically used as the main window of an application.
    - A ***JFrame*** can contain multiple components, including ***JPanel***, ***JScrollPane***, ***JButton, JTable***, and etc.
- Panel:
    - A ***JPanel*** is a generic container that is used to group a set of components together within a window.
    - It does not have window decorations like a title bar or close button.
    - A ***JPanel*** is often used to organize and manage layout within a ***JFrame***.

The example below includes a ***JFrame*** and a ***JPanel***, as well as additional components like buttons, text fields, and labels using a ***GridBagLayout***. Moreover, it also displays a message using ***JDialog***, the ***JOptionPane*** component, and a Dialog window component. It is a simple graphical user interface (GUI) contact form using Swing components.

```
//--- Abstract Window Toolkit (AWT)

// Provides layout manager for arranging components in five regions:
// north, south, east, west, and center.
import java.awt.BorderLayout;
// Grid layout - Specifies constraints for components that are laid out using the GridBagLayout.
```

```java
import java.awt.GridBagConstraints;
// Grid - layout manager that aligns components vertically and horizontally,
// without requiring the components to be of the same size.
import java.awt.GridBagLayout;
// Gird padding - Specifies the space (padding) between components and their borders.
import java.awt.Insets;
// Button - Provides the capability to handle action events like button clicks.
import java.awt.event.ActionEvent;
// Button event - Allows handling of action events, such as button clicks.
import java.awt.event.ActionListener;

//--- swing GUI

// Button - Provides a button component that can trigger actions when clicked.
import javax.swing.JButton;
// Frame - Provides a window with decorations
// such as a title, border, and buttons for closing and minimizing.
import javax.swing.JFrame;
// Labels - Provides a display area for a short text string or an image, or both.
import javax.swing.JLabel;
// Submition Message - Provides standard dialog boxes such as message, input, and confirmation
dialogs.
import javax.swing.JOptionPane;
// Panel - Provides a generic container for grouping components together.
import javax.swing.JPanel;
// Scroll user message - Provides to the a scrollable view of a lightweight component.
import javax.swing.JScrollPane;
// User message - Provides a multi-line area to display/edit plain text.
import javax.swing.JTextArea;
// Name & Email - Provides a single-line text field for user input.
import javax.swing.JTextField;

/**
 * This class generates a simple contact form. The form includes fields for the
 * user's name, email, and message, and a submit button to submit the form.
 *
 * @author Alejandro Ricciardi
 * @version 1.0
 * @date 06/16/2024
 */
public class SimpleContactForm {

        /**
         * The main method to create and display the contact form.
         *
         * @param args Command line arguments
         */
        public static void main(String[] args) {

                /*------------
                 |   Frame   |
                 ------------*/

                // ---- Initializes frame
                // Creates the main application frame
                JFrame frame = new JFrame("Contact Form");
                frame.setSize(400, 300); // Set the size of the frame
                // Close the application when the frame is closed
                frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                frame.setLayout(new BorderLayout()); // Use BorderLayout for the frame

                /*------------
                 |   Panel   |
                 ------------*/

                // ---- Initializes panel
                // Create a panel with GridBagLayout
                JPanel panel = new JPanel(new GridBagLayout());
                GridBagConstraints gridForm = new GridBagConstraints();
                gridForm.insets = new Insets(5, 5, 5, 5); // Add padding around components

                // ---- Creates and adds grid components to the panel

                // -- Name
                // Adds "Name" label
                JLabel nameLabel = new JLabel("Name:");
                gridForm.gridx = 0; // Position at column 0
                gridForm.gridy = 0; // Position at row 0
                panel.add(nameLabel, gridForm);
                // Add text field for name input
                JTextField nameField = new JTextField(20);
                gridForm.gridx = 1; // Position at column 1
                gridForm.gridy = 0; // Position at row 0
                panel.add(nameField, gridForm);

                // -- Email
                // Add "Email" label
                JLabel emailLabel = new JLabel("Email:");
                gridForm.gridx = 0; // Position at column 0
                gridForm.gridy = 1; // Position at row 1
                panel.add(emailLabel, gridForm);
                // Adds text field for email input
                JTextField emailField = new JTextField(20);
                gridForm.gridx = 1; // Position at column 1
                gridForm.gridy = 1; // Position at row 1
                panel.add(emailField, gridForm);

                // Adds "Message" label
                JLabel messageLabel = new JLabel("Message:");
                gridForm.gridx = 0; // Position at column 0
                gridForm.gridy = 2; // Position at row 2
                panel.add(messageLabel, gridForm);

                // -- Message
                // Adds text area for message input with a scroll pane
                JTextArea messageArea = new JTextArea(5, 20);
                JScrollPane scrollPane = new JScrollPane(messageArea);
                gridForm.gridx = 1; // Position at column 1
                gridForm.gridy = 2; // Position at row 2
                panel.add(scrollPane, gridForm);
                // Adds "Submit" button
                JButton submitButton = new JButton("Submit");
```

```
                gridForm.gridx = 1; // Position at column 1
                gridForm.gridy = 3; // Position at row 3
                panel.add(submitButton, gridForm);

                // Adds the panel to the frame's center
                frame.add(panel, BorderLayout.CENTER);

                // Make the frame visible
                frame.setVisible(true);

                /*------------
                 |  JDialog  |
                 ------------*/
                // Add action listener to the submit button
                ActionListener submitBtnClicked = new ActionListener() {
                        @Override
                        public void actionPerformed(ActionEvent e) {
                                // Display a message dialog when the submit button is clicked
                                JOptionPane.showMessageDialog(frame, "Message was sent!");
                        }
                };

                submitButton.addActionListener(submitBtnClicked);
        }
}
```

The following video describes how to implement a JDialog JOptionPane message: Java JOptionPane

-Alex

References:

Oracle Docs. (n.d.a). *Swing*. Oracle. https://docs.oracle.com/javase/8/docs/technotes/guides/swing/

Oracle Docs. (n.d.b). *Using Top-Level Containers*. The Java™ Tutorials. Oracle. https://docs.oracle.com/javase/tutorial/uiswing/components/toplevel.html

Oracle Docs. (n.d.c). *Java Applets*. The Java™ Tutorials. Oracle. https://docs.oracle.com/javase/tutorial/deployment/applet/index.html