JFC is short for Java Foundation Classes, which encompass a group of features for building graphical user interfaces (GUIs) and adding rich graphics functionality and interactivity to Java applications. It is defined as containing the Swing features shown as below.

Feature	Description
Swing GUI Components	Includes everything from buttons to split panes to tables. Many components are capable of sorting, printing, and drag and drop, to name a few of the supported features.

The goal of this lesson is to introduce the Swing API by designing a simple application. Its GUI will be basic, focusing on only a subset of the available Swing components. We will use the NetBeans IDE GUI builder, which makes user interface creation a simple matter of drag and drop. Its automatic code generation feature simplifies the GUI development process, letting you focus on the application logic instead of the underlying infrastructure.

AWT to Swing: earlier, we used AWT to develop GUI, but now we have Swing which is lighter and powerful.

AWT: Abstract Windowing Toolkit
import java.awt.*

Swing: new with Java 2
import javax.swing.*
Extends AWT
New improved components
Standard dialog boxes, tooltips, …
Look-and-feel, skins
Event listeners

GUI Component API : Some are listed below and you have many more components…

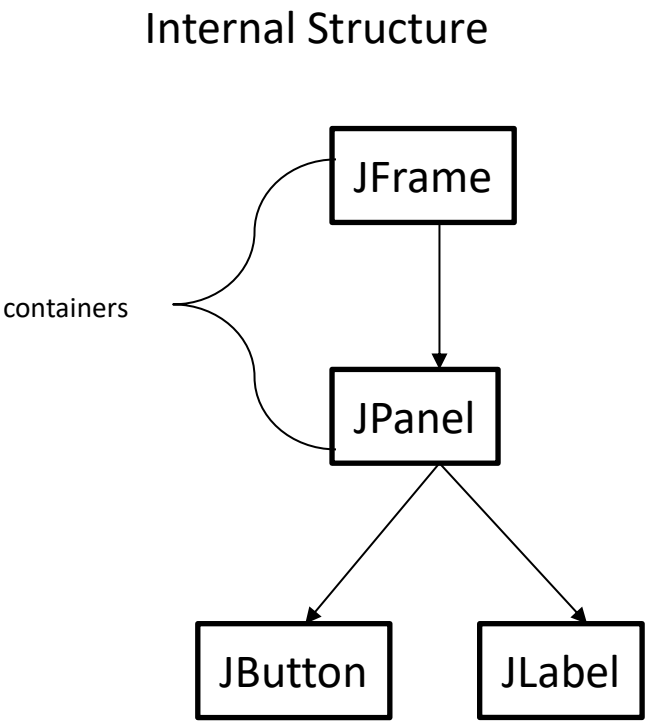
Answers

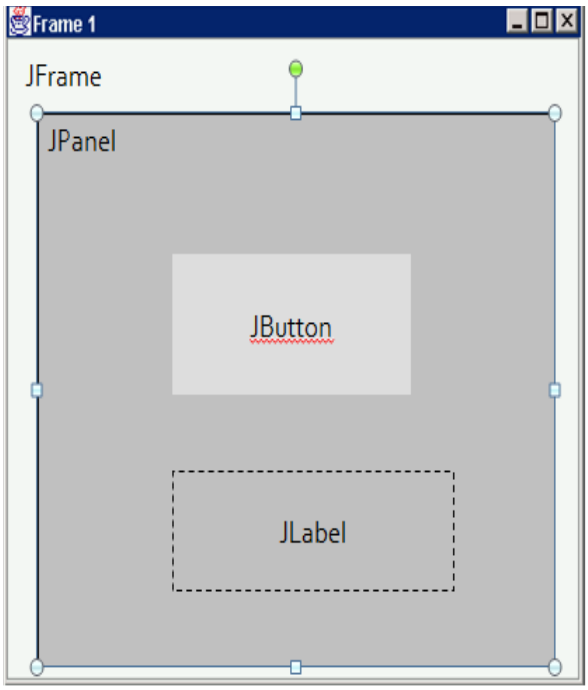
```
import java.awt.Color;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class ColorChanger {

   private JFrame frame;
   private JButton button;
   private Color background;

   public ColorChanger() {
      frame = new JFrame("Color Changer");
      button = new JButton("Click to Change Color");
      background = Color.WHITE;

      button.addActionListener(new ActionListener() {
         @Override
         public void actionPerformed(ActionEvent e) {
            background = background == Color.WHITE ? Color.RED : Color.WHITE;
            frame.getContentPane().setBackground(background);
         }
      });

      frame.getContentPane().add(button);
      frame.setSize(300, 200);
      frame.setVisible(true);
   }

   public static void main(String[] args) {
      new ColorChanger();
   }
}
```

Using a GUI Component: Step by step proces

1. **Create it**
   - Instantiate object: b = new JButton(“press me”);
2. **Configure it**
   - Properties: b.text = “Click Me”;
   - Methods: b.setText(“press me”);
3. **Add it**
   - panel.add(b);
4. **Listen to it – when you want to perform an event based on user action we use Listeners.**
   - Events: Listeners – are listening to the component and when there’s an action (state change e.g. Button click , mouse over) it will trigger an event.

Internal structure of a UI:





Explanation or answer

- To create a JButton, you would first need to create a new JFrame. A JFrame is a Swing component that can be used to create a window in a Java application.

- Once you have created a new JFrame, you would need to add a JButton to the JFrame. You can add a JButton to the JFrame by calling the getContentPane() method on the JFrame and then adding the JButton to the getContentPane() method's return value.

- Once you have added a JButton to the JFrame, you would need to configure the JButton. You can configure the JButton by setting the JButton's text, the JButton's size, and the JButton's location.

Here are the steps on how to create a JButton in Java:

I. Create a new JFrame.
II. Add a JButton to the JFrame.
III. Configure the JButton.
IV. Show the JFrame.

```
public class JButtonExample {

   public static void main(String[] args) {
      JFrame frame = new JFrame("JButton Example");
      JButton button = new JButton("Click Me!");
      button.setSize(100, 50);
      button.setLocation(100, 100);
      frame.getContentPane().add(button);
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      frame.setVisible(true);
   }
}
```

Introduction to GUI Building Using Netbeans:

Please follow the link below. This gives you a detailed description on how to develop a UI using Netbeans. Complete the steps and build your first UI ☺

https://netbeans.org/kb/docs/java/gui-functionality.html

Next, try to develop the following user interfaces.. For last interface, you don't need to make any actions..

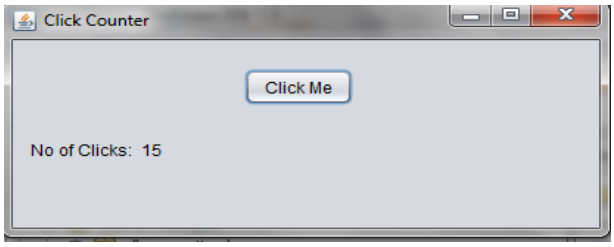
The following UI is used to change the background color of the UI. When user press on the button color will change…

Answer

```
import java.awt.Color;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;


public class ColorChanger {


   private JFrame frame;
   private JButton button;
   private Color background;


   public ColorChanger() {
      frame = new JFrame("Color Changer");
      button = new JButton("Click to Change Color");
      background = Color.WHITE;
```

```
        button.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                background = background == Color.WHITE ? Color.RED : Color.WHITE;
                frame.getContentPane().setBackground(background);
            }
        });

        frame.getContentPane().add(button);
        frame.setSize(300, 200);
        frame.setVisible(true);
    }

    public static void main(String[] args) {
        new ColorChanger();
    }
}
```

This is a click counter UI. When you press the button it will increment the number of clicks. Start from 0…

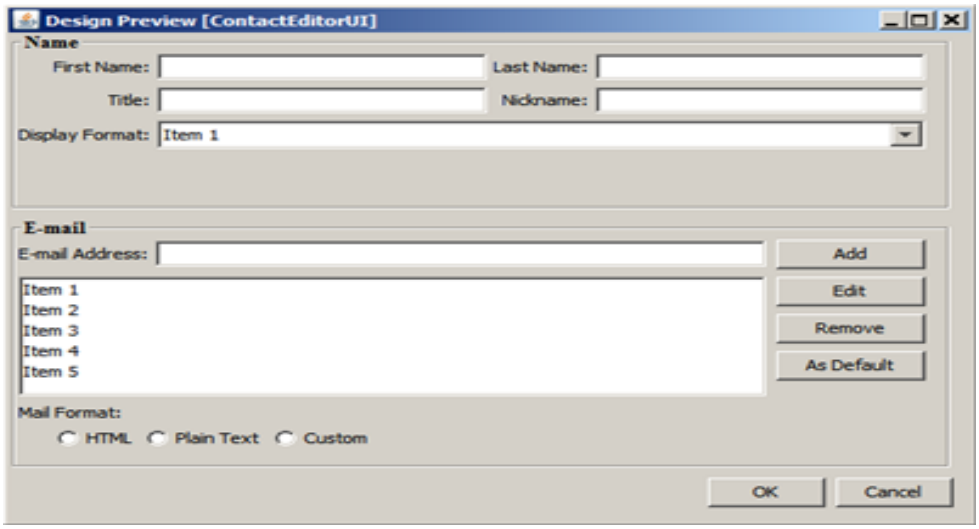


Answers

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;


public class ClickCounter {


    private int count = 0;
    private JButton button;


    public ClickCounter() {
        button = new JButton("Click me!");
        button.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                count++;
                button.setText("Clicks: " + count);
            }
        });
    }
```

```
    public static void main(String[] args) {
        new ClickCounter();
    }
}
```

Bit complex UI with different components.



Explanation Or Answer

- Once you have added the libraries to your project, you would need to open the Design Preview dialog box. The Design Preview dialog box can be opened by clicking on the Window menu and selecting Design Preview.

- Once the Design Preview dialog box is open, you would need to add the different components to your UI. You can add components to your UI by dragging and dropping them from the Palette onto the Design Preview window.

- Once you have added the different components to your UI, you would need to configure the properties of the components. You can configure the properties of the components by double-clicking on the components in the Design Preview window.

- Once you have configured the properties of the components, you would need to save your project. Once you have saved your project, you can run your UI by clicking on the Run menu and selecting Run.

Here are the steps on how to use the Design Preview dialog box to create a bit complex UI with different components.

I. Create a new project in NetBeans.
II. Add the java.swing and java.awt libraries to your project.
III. Open the Design Preview dialog box.
IV. Add the different components to your UI by dragging and dropping them from the Palette onto the Design Preview window.
V. Configure the properties of the components by double-clicking on the components in the Design Preview window.
VI. Save your project.
VII. Run your UI by clicking on the Run menu and selecting Run.