CHAPTER 3

■ ■ ■

Advanced Swing

QUESTIONS AND EXERCISES

1. Create a `JButton` with a `Close` label. The label should be formatted in italics and boldface font using HTML.

   Solution:

```
JButton closeBtn = new JButton();
closeBtn.setText("<html><body><i><b>Close</b></i></body></html>");
```

2. Create a `JButton` with a `<html>Use the <b> tag for bold</html>` label. Note that the label contains HTML tags and the tags should be displayed as shown in the label.

   Solution:

```
JButton closeBtn = new JButton();
closeBtn.putClientProperty("html.disable", Boolean.TRUE);
closeBtn.setText("<html>Use the <b> tag for bold</html>");
```

3. How do you check if your code is running in the event dispatch thread?

   Answer:

   It is very simple to know whether your code is executing in the event dispatch thread or not, by using the static method `isEventDispatchThread()` of the `SwingUtilities` class. The method returns `true` if your code is executing in the event dispatch thread. Otherwise, it returns `false`. For debugging purposes, you can write the following statement anywhere in your Java code. If it prints `true`, it means your code was executed in the event dispatch thread.

```
System.out.println(SwingUtilities.isEventDispatchThread());
```

4. Compare and contrast the use of the `invokeLater()` and `invokeAndWait()` methods of the `SwingUtilities` class.

Answer:

The `SwingUtilities` class is in the javax.swing package. It contains two static method: `invokeLater(Runnable r)` and `invokeAndWait(Runnable r)` methods.

The `invokeLater()` method returns immediately and the `run()` method of its `Runnable` argument is executed asynchronously. That is, its `run()` method's execution is queued to the event dispatch thread for later execution.

The `invokeAndWait()` is executed synchronously and it does not return until the `run()` method of its `Runnable` argument has finished executing on the event dispatch thread. This method may throw an `InterruptedException` or `InvocationTargetException`.

The `invokeAndWait()` method should not be called from the event dispatch thread because the thread that executes this method call waits until the `run()` method has finished. If you execute this method from the event dispatch thread, it will be queued to the event dispatch thread and the same thread (the event dispatch thread) will be waiting. Executing this method call in the event dispatch thread generates a runtime error.

5. What Swing utility class do you need to use when you want to run a long-running task in non-event dispatch thread and publish the result of the task in a Swing application?

Answer:

```
javax.swing.SwingWorker<T,V>
```

6. How do you set the system (or native) look and feel for a Swing application?

Answer:

```
UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
```

7. What is drag and drop (DnD)? Create an application with a `JTextArea`. The user should be able to drag and drop text from any other application such as an Internet browser into the `JTextArea` and the dragged text should be inserted into the `JTextArea`.

Answer:

Drag and drop (DnD) is a way to transfer data in an application. DnD lets you transfer data by dragging a component and dropping it onto another component. The component that is dragged is called the *drag source*; it supplies the data to be transferred. The component onto which the drag source is dropped is called the *drop target*; it is the receiver of the data. It is the responsibility of the drop target to accept the drop action and import the data supplied by the drag source. The data transfer is accomplished using a `Transferable`. `Transferable` is an interface in the `java.awt.datatransfer` package, which is in the `java.datatransfer` module.

```
// DnDTextArea.java
package com.jdojo.swing.advanced.exercises;

import java.awt.BorderLayout;
import java.awt.Container;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JTextArea;
import static javax.swing.WindowConstants.EXIT_ON_CLOSE;

public class DnDTextArea extends JFrame {
    private final JTextArea contents = new JTextArea(30, 80);

    public DnDTextArea() {
        super("Notepad");
        this.initFrame();
    }

    private void initFrame() {
        this.setDefaultCloseOperation(EXIT_ON_CLOSE);
        Container contentPane = this.getContentPane();

        contentPane.add(contents);

        // Enable drag and drop for the text area
        contents.setDragEnabled(true);

        String msg = "Drag and drop selected text from other app into the text area";
        contentPane.add(new JLabel(msg), BorderLayout.NORTH);
    }

    public static void main(String[] args) {
        DnDTextArea frame = new DnDTextArea();
        frame.pack();
        frame.setVisible(true);
    }
}
```

8. What are SDI, MDI, and TDI applications? Give an example of each kind.

Answer:

In an SDI application, only one window is opened at any time. In an MDI application, one main window (also the called parent window) is opened, and multiple child windows are opened within the main window. In a TDI application, one window is opened, which has multiple windows open as tabs. Microsoft Notepad is an example of an SDI application, Microsoft Word 97 is an example of an MDI application (newer versions of Microsoft Word are SDI), and Google Chrome browser is an example of a TDI application.

9. When do you use instances of the `JInternalFrame` and `JDesktopPane` classes?

Answer:

In an MDI application, you can open multiple frames that are instances of the `JInternalFrame` class. An instance of the `JDesktopPane` class is used as a container (not as a top-level container) for all child windows that are instances of the `JInternalFrame` class. It uses a `null` layout manager. You add it to a `JFrame`.

10. What is the role of a `DesktopManager` in a Swing application?

Answer:

A `JDesktopPane` uses an instance of the `DesktopManager` interface to manage all internal frames. The `DefaultDesktopManager` class is an implementation of the `DesktopManager` interface.

11. What kind of functionalities are provided by instances of the `java.awt.Toolkit` class?

Answer:

An instance of the `java.awt.Toolkit` class provides access to the native operating system features. Java needs to communicate with the native system to provide most of the basic GUI functionalities. It uses a specific class on each platform to achieve that. The `Toolkit` is an abstract class. Java uses a subclass of the `Toolkit` class on each platform to communicate with the native toolkit system. The `Toolkit` class provides a static `getDefaultToolkit()` factory method to get the toolkit object used on a particular platform. The `Toolkit` class contains useful methods to let you work with screen size and resolution, get access to the system clipboard, and to make a beeping sound, etc.

12. Write a program using the `Toolkit` class that prints the screen size to the standard output.

Solution:

```
// ScreenSizePrinter.java
package com.jdojo.swing.advanced.exercises;

import java.awt.Dimension;
import java.awt.Toolkit;

public class ScreenSizePrinter {
    public static void main(String[] args) {
        Toolkit tk = Toolkit.getDefaultToolkit();
        Dimension screenSize = tk.getScreenSize();
        System.out.printf("width=%d, height=%d%n", screenSize.width, screenSize.height);
    }
}
```

13. When do you use a `JLayer` in your Swing application?

Answer:

The `JLayer` class represents a Swing component. It is used to decorate another component, which is called the *target* component. It lets you perform custom painting over the component it decorates. It can also receive notifications of all events that are generated within its border. In other words, a `JLayer` lets you perform custom processing based on events occurring in the component it decorates.

When you work with the `JLayer` class, you also need to work with the `LayerUI<V extends Component>` class. A `JLayer` delegates its work to a `LayerUI` for custom painting and event handling. To do anything meaningful with a `JLayer`, you need to create a subclass of the `LayerUI` class and override its appropriate methods to write your code.