-

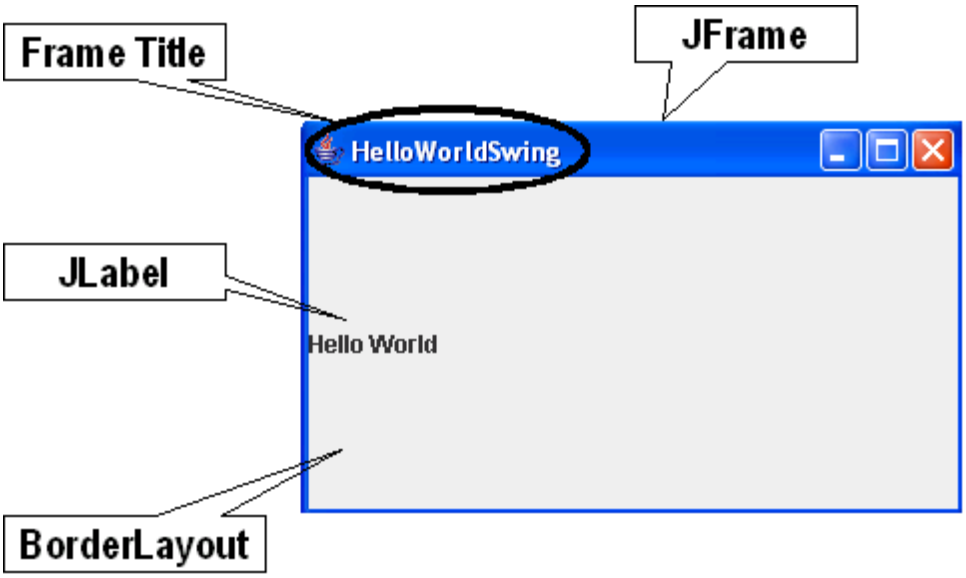
1. **GUI Construction**
   A Java technology GUI can be created using either of the following techniques.
   - Programmatic construction
     This technique use code to create the GUI. This technique is used for learning GUI construction. However, it is very laborious to use in production environment.
   - Construction using a GUI builder tool
     This technique uses a GUI builder tool to create the GUI. The GUI developer uses a visual approach to drag-and-drop containers and components to a work area.
2. **Example of programmatic construction**
   This sections a simple GUI that prints a Hello World. The code shown in below create a container JFrame with a title HelloWorldSwing. It later adds a JLabel with the Accessible Name property set to Hello World.

```
import javax.swing.*;
public class HeloWorldSwing {
 private static void createAndShowGUI() {
 JFrame frame = new JFrame("HelloWorldSwing");
//Set up the window.


frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
JLabel label = new JLabel("Hello World");
// Add Label
frame.add(label);
 frame.setSize(300,200);
// Display Window
 frame.setVisible(true);
 }

 public static void main(String[] args) {
        javax.swing.SwingUtilities.invokeLater(new Runnable() {
        //Schedule for the event-dispatching thread:
        //creating,showing this app's GUI.
        public void run() {createAndShowGUI();}
        });
        }
        }
```

The output generated from the program



3. Key Methods

Methods for setting up the JFrame and adding JLabel:

- setDefaultCloseOperationJFrame.EXIT_ON_CLOSE)
  - –Creates the program to exit when the close button is clicked. There are four possible ways of handling this:
    a. DO_NOTHING_ON_CLOSE: does nothing when the close operation is initiated. This constant is defined in WindowsConstants.
    b. Hide_ON_ClOSE: invokes any WindowListener objects and hides the frame. This constant is defined in WindowsConstants.
    c. DISPOSE_ON_CLOSE: invokes any WindowListener objects and hides and disposes the frame. This constant is defined in WindowsConstants.

- setVisible(true)– Makes the JFrame visible.
- add(Component c)– this method adds the components to the container.

To handle these tasks efficiently the Swing framework uses threads that are ***light-weight process*** . The tasks described can be handled by these threads *separately and concurrently*.

The programmer should utilities these threads. The Swing framework provides a collection of utility methods in the SwingUtilities class.

```
SwingUtilites.invokeLater(new Runnable())
```

In the java programming language, threads are created using the Runnable interface. This interface defines a method run that should be implemented by all the classes using this interface. The `invokeLater` method schedule the GUI creations taks to execute the run method a synchronously by the event-handling thread after all the pending events are completed.

4. GUI-Based Applications

You now know how to set up a Java GUI for both graphic output and interactive user input. However, only a few of the components from which GUIs can be built have been described.

The question here is of How can WE create a menu for your GUI frame?

5. How to Create a Menu

1. Create a `JMenuBar` object, and set it into a menu container, such as a `JFrame`.
2. Create one or more `JMenu` objects, and add them to the menu bar object.
3. Create one or more `JMenuItem` objects, and add them to the menu object.

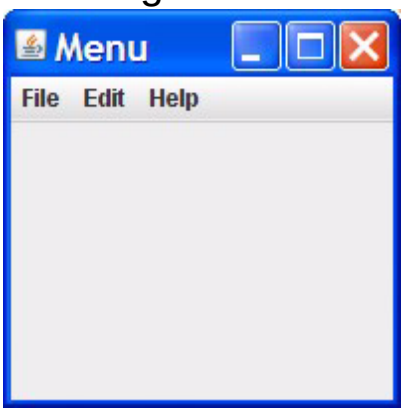
Creating a `JMenuBar`

```
f = new JFrame("MenuBar");
mb = new JMenuBar(); f.setJMenuBar(mb);
```


```
f = new JFrame("Menu");
mb = new JMenuBar();
```

```
m1 = new JMenu("File");
m2 = new JMenu("Edit");
m3 = new JMenu("Help");
mb.add(m1);// adds objects to JMenuBar
mb.add(m2);
mb.add(m3);
f.setJMenuBar(mb);
```

Creating a `JMenu`



6. Creating a JMenus Items

```
import javax.swing.*;

public class menubar extends JFrame{

public menubar(){
     JMenuBar menubar = new JMenuBar();
     setJMenuBar(menubar);

     JMenu shape = new JMenu("File");
     menubar.add(shape);

     JMenuItem rect = new JMenuItem("Rectangle");
     shape.add(rect);

     JMenuItem star = new JMenuItem("Star");
```

```
        shape.add(star);

        JMenu color = new JMenu("Color");
        menubar.add(color);

        JMenuItem black = new JMenuItem("Black");
        color.add(black);

        JMenuItem orange = new JMenuItem("Orange");
        color.add(orange);
    }

    public static void main(String[] args) {
        menubar gui = new menubar();
        gui.setTitle("Menu Bar");
        gui.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        gui.setSize(500,300);
        gui.setVisible(true);
        gui.setLocationRelativeTo(null);
    }
    }
```