AWT, Swing and Event Handling [Unit-04 and Unit-05]

AWT and SWING (Unit-04)

- Introduction to AWT and Swings
- Swings advantages over AWT
- Swing applications
- Swing Controls
- Jbutton
- Jlabel
- JTextFiled
- JTextArea
- JRadioButton
- JCheckBox
- Jlist
- JComboBox
- JScrollBar
- Jtable
- Graphics in swing **[AWT]**

Event Handling (Unit-05)

- Event delegation model and Classes
- Event Listener Interfaces
- Adapter classes.

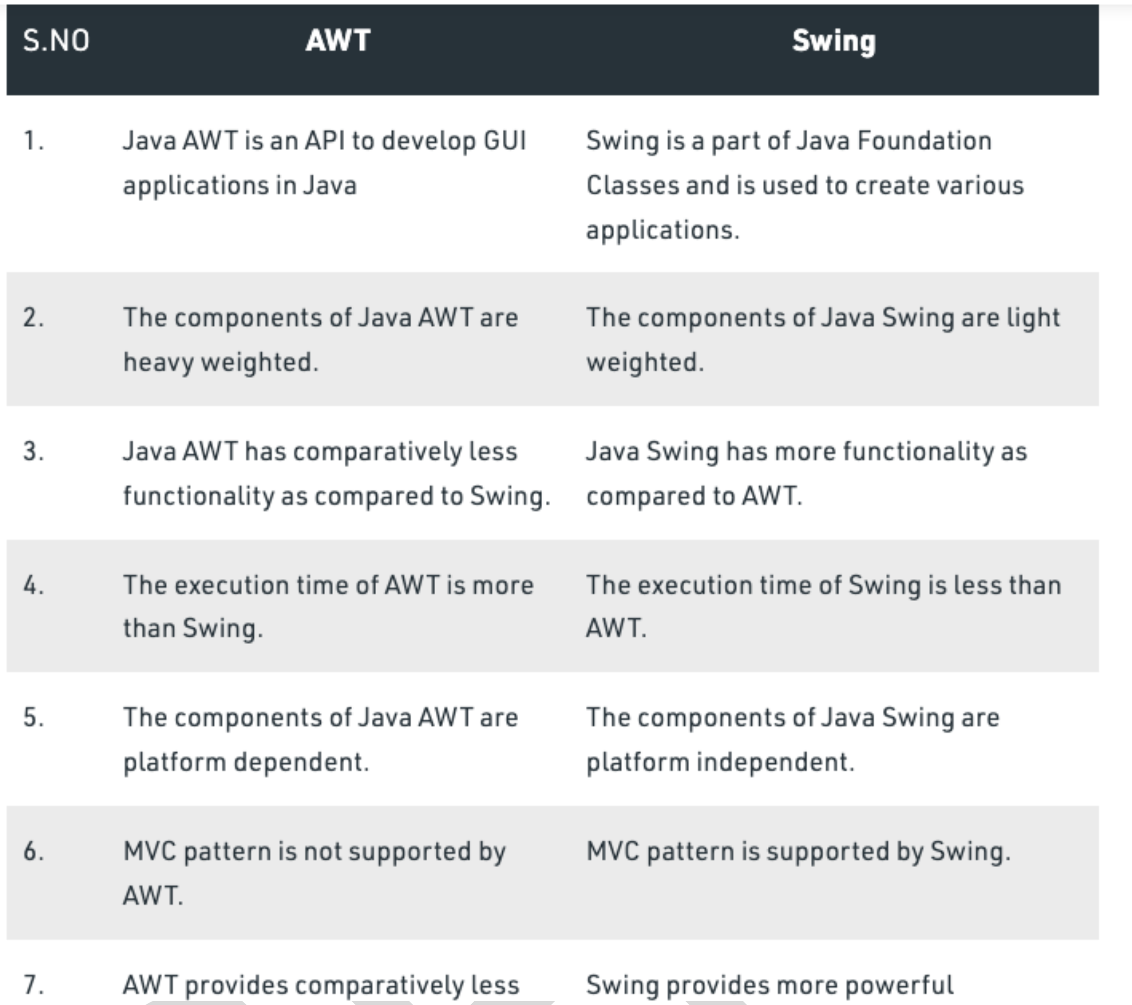
AWT and SWING

Swing Introduction

- In order to create some GUI applications, we can use Swing
- It is light weight in compare to AWT, because all the component are made of java
- Swing replaces AWT due to above mentioned reasons.
- **In order to use Swing in Java, we need to have 1.7v JRE or above. Else you would not be able to use Swing library.**

Abstract Window Toolkit (AWT)

- AWT (Abstract window toolkit): Application Program Interface (API)
- It is a set of application program interfaces (API s) used by Java programmers to create graphical user interface (GUI) objects, such as buttons, scroll bars, and windows. AWT is part of the Java Foundation Classes (JFC) from Sun Microsystems, the company that originated Java.
- It is Heavy in compare to Swing, because all the component are **made of native languages**



S.NO	AWT	Swing
1.	Java AWT is an API to develop GUI applications in Java	Swing is a part of Java Foundation Classes and is used to create various applications.
2.	The components of Java AWT are heavy weighted.	The components of Java Swing are light weighted.
3.	Java AWT has comparatively less functionality as compared to Swing.	Java Swing has more functionality as compared to AWT.
4.	The execution time of AWT is more than Swing.	The execution time of Swing is less than AWT.
5.	The components of Java AWT are platform dependent.	The components of Java Swing are platform independent.
6.	MVC pattern is not supported by AWT.	MVC pattern is supported by Swing.
7.	AWT provides comparatively less	Swing provides more powerful

Set Up your system for Swing:

- Right click on project
- Go to Properties
- Go to build path
- Go to libraries
- Add libraries
  - Choose JRE System Libraries
  - **Choose 1.7v JRE or above**
  - Eclipse should support this version also
  - Apply -> Apply and close
- Restart your Eclipse for this change
- Now you can import these classes now:
  - import javax.swing.JFrame;
  - import javax.swing.JLabel;

Note: Create a separate Java Project for Swing as we are changing the execution environment JRE to use Swing, it may affect out previous work as well

```
package Unit_04;

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

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JTextField;

public class P17_AddTwoNumbers_GUI {

      public static void main(String[] args) {

            AddNumbers obj = new AddNumbers();

      }

}

class AddNumbers extends JFrame{

      JTextField t1;
      JTextField t2;
      JLabel l;
      JButton b;

      public AddNumbers() {

            t1 = new JTextField(20);
            t2 = new JTextField(20);

            l = new JLabel("Result");
            b = new JButton("Ok");

            add(t1);
            add(t2);
            add(b);
            add(l);

            ActionListener al = new ActionListener() {

                  @Override
                  public void actionPerformed(ActionEvent e) {

                        int num1 = Integer.parseInt(t1.getText());
                        int num2 = Integer.parseInt(t2.getText());

                        Integer sum = num1 + num2;
```

```
                l.setText(sum.toString());

            }

        };


        b.addActionListener(al);


        setLayout(new FlowLayout());
        setVisible(true);
        setSize(400, 400);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }



}
```