**Module 23 - Introduction to Swings**

**1. Module Introduction**

**Learning Objectives**

* Introduction to Swings
* MVC Architecture
* Swing features
* Hierarchy of Java Swing classes
* Difference between AWT and Swing
* Problem sets

Introduction

Unlike AWT, Java Swing provides platform-independent and lightweight components.

The javax.swing package provides classes for java swing API such as JButton, JTextField, JTextArea, JRadioButton, JCheckbox, JMenu, JColorChooser etc.

* Swing API is set of extensible GUI Components to ease developer's life to create JAVA based Front End/ GUI Applications.
* It is build upon top of AWT API and acts as replacement of AWT API as it has almost every control corresponding to AWT controls.
* Swing component follows a Model-View-Controller architecture to fulfill the following criterias.

MVC Architecture

Swing API architecture follows loosely based MVC architecture in the following manner.

* A Model represents component's data.
* View represents visual representation of the component's data.
* Controller takes the input from the user on the view and reflects the changes in Component's data.

Swing component have Model as a separate element and View and Controller part are clubbed in User Interface elements. Using this way, Swing has pluggable look-and-feel architecture.

Swing Features

* Light Weight -

Swing component are independent of native Operating System's API as Swing API controls are rendered mostly using pure JAVA code instead of underlying operating system calls.

* Rich controls -

Swing provides a rich set of advanced controls like Tree, TabbedPane, slider, colorpicker, table controls

* Highly Customizable -

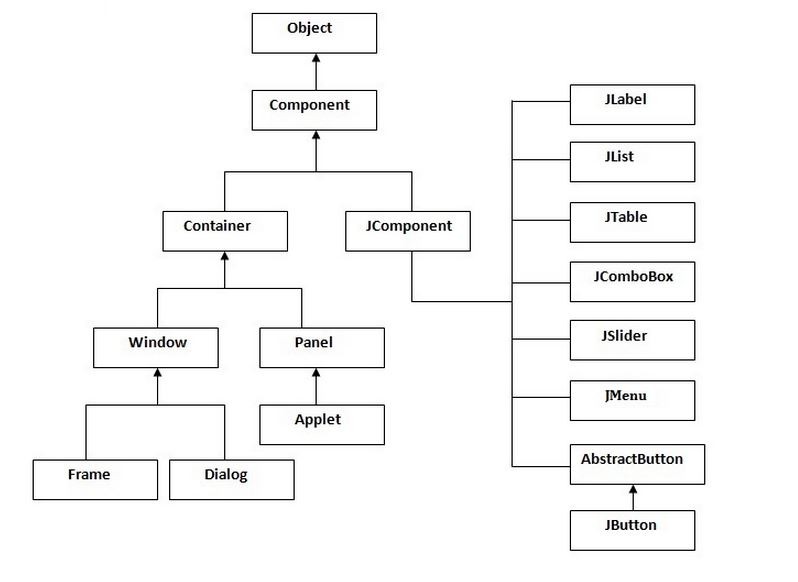
Swing controls can be customized in very easy way as visual apperance is independent of internal representation.

* Pluggable look-and-feel-

SWING based GUI Application look and feel can be changed at run time based on available values.

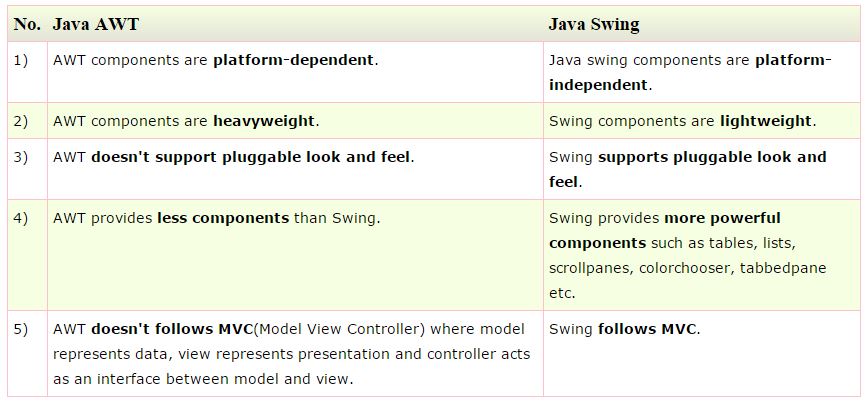
Hierarchy of Java Swing classes

The hierarchy of java swing API is given below.

[](http://wiki2016.msitprogram.net/index.php/File:Swing_hier.JPG)

Difference between AWT and Swing

There are many differences between java awt and swing that are given below.

[](http://wiki2016.msitprogram.net/index.php/File:Awt_swing.JPG)

**2. Resources**

<https://youtu.be/ZDw18yUwz6I>

## Example Problems

There are two ways to create a frame:

* By creating the object of Frame class (association)
* By extending Frame class (inheritance)

We can write the code of swing inside the main(), constructor or any other method.

Let's see a simple swing example where we are creating one button and adding it on the JFrame object inside the main() method.

import javax.swing.\*;

public class FirstSwingExample {

public static void main(String[] args) {

JFrame f=new JFrame();//creating instance of JFrame

JButton b=new JButton("click");//creating instance of JButton

b.setBounds(130,100,100, 40);//x axis, y axis, width, height

f.add(b);//adding button in JFrame

f.setSize(400,500);//400 width and 500 height

f.setLayout(null);//using no layout managers

f.setVisible(true);//making the frame visible

}

}

We can also write all the codes of creating JFrame, JButton and method call inside the java constructor.

import javax.swing.\*;

public class Simple {

JFrame f;

Simple(){

f=new JFrame();//creating instance of JFrame

JButton b=new JButton("click");//creating instance of JButton

b.setBounds(130,100,100, 40);

f.add(b);//adding button in JFrame

f.setSize(400,500);//400 width and 500 height

f.setLayout(null);//using no layout managers

f.setVisible(true);//making the frame visible

}

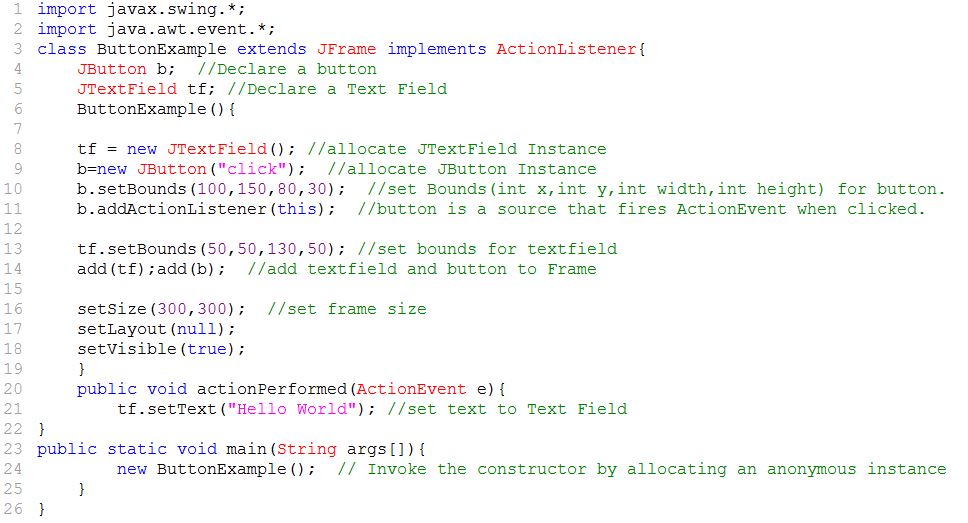
public static void main(String[] args) {

new Simple();

}

}

#### Hello World Example

[](http://wiki2016.msitprogram.net/index.php/File:Swing_Hello.JPG)