GUI:

GUI stands for Graphical User Interface, a term used not only in Java but in all programming languages that support the development of GUIs. A program's graphical user interface presents an easy-to-use visual display to the user. It is made up of graphical components (e.g., buttons, labels, windows) through which the user can interact with the page or application.

To make graphical user interfaces in Java, use either Swing or JavaFX.

Typical Elements

A GUI includes a range of user interface elements — which just means all the elements that display when you are working in an application. These can include as buttons, dropdown lists, checkboxes, and textfields, labels, banners, icons, or notification dialogs, menus, etc.

Java GUI Frameworks: Swing and JavaFX

Java has included Swing, an API for creating GUIs, in its Java Standard Edition since Java 1.2, or 2007. It's designed with a modular architecture so that elements are easily plug-and-play and can be customized. It has long been the API of choice for Java developers when creating GUIs.

JavaFX has also been around a long time — Sun Microsystems, which owned Java before the current owner Oracle, released the first version in 2008.

Oracle's intention is to eventually replace Swing with JavaFX. Java 8, released in 2014, was the first release to include JavaFX in the core distribution.

If you are new to Java, you should learn JavaFX rather than Swing, although you may need to understand Swing because so many applications incorporate it, and so many developers are still actively using it.

JavaFX features an entirely different set of graphic components as well as a new terminology and has many features that interface with web programming, such as support for Cascading Style Sheets (CSS), a web component for embedding a web page inside an FX application, and the functionality to play web multimedia content.

JFC:

JFC stands for Java Foundation Classes, which include a group of features for building Graphical User Interfaces (GUIs) and adding rich graphics functionality and interactivity to Java applications. It is a collection of five APIs: AWT, Accessibility, 2D API, Drag and Drop and Swing. The JFC is a standard Java APIs for client-side Graphical User Interfaces (GUIs), and related programming tasks.

- **AWT API** is used to display GUI components which is the original toolkit supplied by Java. AWT provides the foundation upon which the rest of the JFC is built. AWT components lack many GUI features which other GUI technologies give. For example, we cannot place an image on the button or change the border of a button. All these are overcome in Swing. Swing adds many features to AWT components. To achieve this, designers added new classes and came out with a special packages **javax.swing** and **javax.swing.event**. It is preferred to know **AWT** before attempting Swing.
- **Accessibility** API is used to map components according to screen setting in terms of pixels. It consists of classes that enable Swing components to interact with assistive(related) technologies for users.
- **2D API** is used to display GUI components in two – dimensional and three – dimensional elevations. The border of a component can be drawn in different thicknesses styles.
- **Drag and Drop API** is used in JavaBeans to drag components from ToolBox and placing them on to the BeanBox.
- **Swing API** is a set of mostly **lightweight components** built on top of the AWT. Swing provides lightweight replacements for the AWT's **heavyweight components**. Swing also supplies multiple components which AWT is lacking like JScrollPane, JPasswordField, JColorChooser etc. Swing offers a very attractive look and feel appearance for its components than AWT.

Java Swing

Java provides many GUI frameworks that help us in developing a variety of GUI applications. AWT (Abstract Window Toolkit) is one of the oldest GUI frameworks in Java and is also platform dependent. The disadvantage of AWT is its heavyweight components.

Another GUI framework in Java i.e. "SWING". The Swing framework in Java is a part of Java Foundation Classes or commonly called JFCs. JFC is an API that is similar to MFCs (Microsoft Foundation Classes) in C++. JFC contains Swing, AWT, and Java2D.

The Swing framework in Java is built on top of the AWT framework and can be used to create GUI applications just like AWT. But unlike AWT, the Swing components are light-weight and are platform-independent.

The Swing framework is written entirely in Java. The Swing framework in Java is provided through the 'javax.swing' package. The classes in the javax.swing package begins with the letter 'J'. So in a javax.swing package, we will have classes like JLabel, JButton, JFrame, JTextField, JTextArea, etc.

In general, the Swing API has every control defined in javax.swing package that is present in AWT. So swing in a way acts as a replacement of AWT. Also, Swing has

various advanced component tabbed panes. Swing API in Java adapts MVC (Model View Controller) Architecture.

The features of the swing API are given below.

1. Swing components are platform-independent.
2. The API is extensible.
3. Swing components are light-weight. The swing components are written in pure Java and also components are rendered using Java code instead of underlying system calls.
4. Swing API provides a set of advanced controls like TabbedPane, Tree, Colorpicker, table controls, etc. which are rich in functionality.
5. The swing controls are highly customizable. This is because the appearance or look-and-feel of the component is independent of internal representation and hence we can customize it in the way we desire.
6. We can simply change the values and thus alter the look-and-feel at runtime.

What are lightweight and heavyweight components?

In AWT, all GUI controls are referred as ***heavyweight*** components as they are dependent on underlying OS(e.g. Windows,Solaris etc.) to provide it(paint it and redraw it).An AWT button control in MacOS is actually a MacOS button.

All Swing components are ***lightweight*** components(except for the top-level ones: JWindow, JFrame, JDialog, and JApplet) as they do not require underlying OS to provide them. JVM renders Swing components and hence they are platform independent but they have performance related issues as compared to heavyweight components which are faster to be rendered due to hardware acceleration.

The lightweight components have transparent pixels unlike heavyweights which have opaque pixels. Mouse events on a lightweight component fall through to its parent; while on a heavyweight component it does not. It is generally not recommended to mix heavyweight components with lightweight components while building the GUI.

Difference between AWT and Swing

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

Java AWT	Java Swing
AWT components are **platform-dependent**.	Java swing components are **platform-independent**.
AWT components are **heavyweight**.	Swing components are **lightweight**.
AWT **doesn't support pluggable look and feel**.	Swing **supports pluggable look and feel**.
AWT provides **less components** than Swing.	Swing provides **more powerful components** such as tables, lists, scrollpanes, colorchooser, tabbedpane etc.

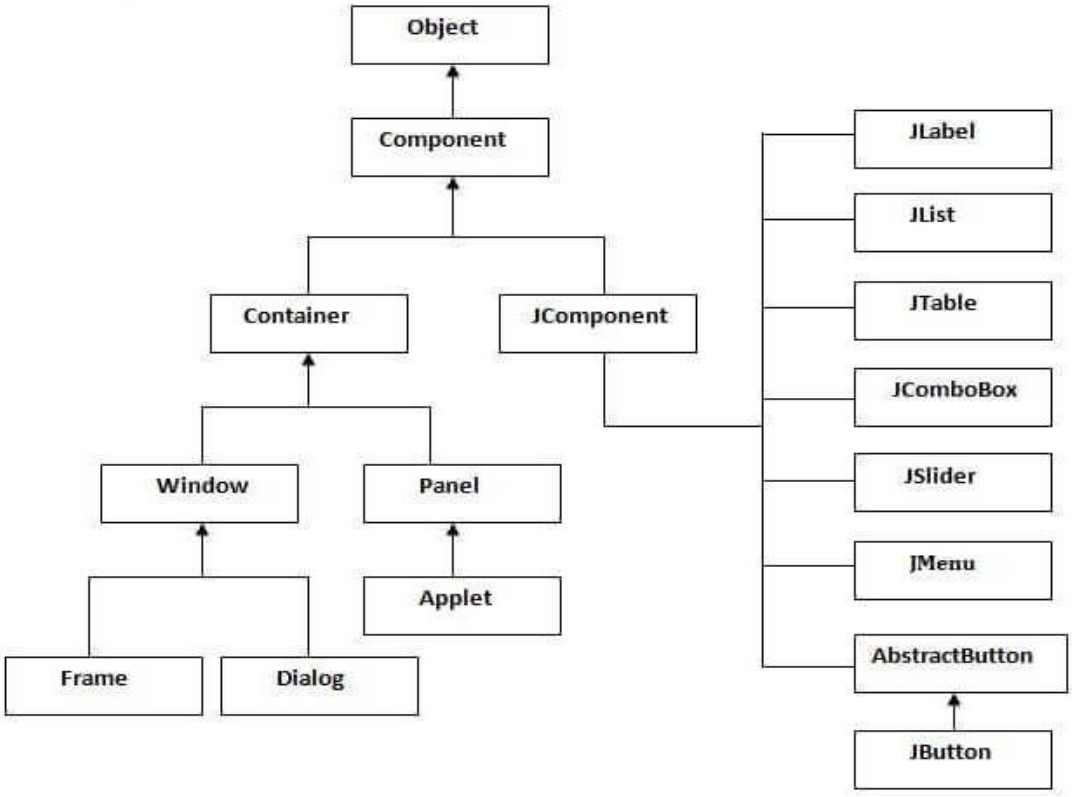
AWT **doesn't follows MVC** (Model View Controller) where model represents data, view represents presentation and controller acts as an interface between model and view.	Swing **follows MVC**.

Pluggable look and feel

Swing is **GUI Widget Toolkit** for Java. It is an API for providing Graphical User Interface to Java Programs. Unlike AWT, Swing components are written in Java and therefore are platform-independent. Swing provides platform specific Look and Feel and also an option for pluggable Look and Feel, allowing application to have Look and Feel independent of underlying platform.

Hierarchy of Java Swing classes

The hierarchy of java swing API is given below



As seen from the above hierarchy we have Container classes – frame, dialog, Panel, Applet, etc. There are also Component classes derived from the JComponent class of Swing API. Some of the classes that inherit from JComponent are JLabel, JList, JTextBox, etc.

Swing Components

A component can be defined as a control that can be represented visually and is usually independent. It has got a specific functionality and is represented as an individual class in Swing API.

Swing has a big set of components that we can include in our programs and avail the rich functionalities using which we can develop highly customized and efficient GUI applications.

For example, class JButton in swing API is a button component and provides the functionality of a button.

One or more components form a group and this group can be placed in a "Container". A container provides a space in which we can display components and also manage their spacing, layout, etc.

Container

Containers are an integral part of SWING GUI components. A container provides a space where a component can be located. The Container is a component in AWT that can contain another component like Buttons, textFields, Labels etc.

Following are certain points to be considered.

- The classes that extend Container class are known as container such as JFrame, JPanel and JWindow.
- Container can add only a Component to itself.
- A default layout is present in each container which can be overridden using **setLayout** method.

JWindow

The window is the container that has no borders and menu bars. You must use frame, dialog or another window for creating a window.

JPanel

The Panel is the container that doesn't contain title bar and menu bars. It can have other components like button, textfield etc.

JFrame

The Frame is the container that contain title bar and can have menu bars. It can have other components like button, textfield etc.

Popular Java Editors

To write your Java programs, you will need a text editor. There are even more sophisticated IDE available in the market. But for now, you can consider one of the following –

- **Notepad** – On Windows machine, you can use any simple text editor like Notepad (Recommended for this tutorial), TextPad.
- **Netbeans** – Netbeans is a Java IDE that is open source and free, which can be downloaded from https://www.netbeans.org/index.html.
- **Eclipse** – Eclipse is also a Java IDE developed by the Eclipse open source community and can be downloaded from https://www.eclipse.org/