

JAVA Lab

Lab Experiment No 10

Name: Aamir Ansari

Batch: A

Roll no: 01

Aim: To write a Java program to implement AWT components.

Problem statement:

Write a program to create a window/frame with four text fields for the name, street, city and pincode with suitable labels. Also the window/frame contains a button MyInfo. When the user types the name, his street, city and pincode ,then clicks the button, the typed details must appear in Arial Font with Size 32, Italics.

Theory:

What is AWT?

AWT stands for Abstract Window Toolkit. It is a platform-dependent API to develop GUI (Graphical User Interface) or window-based applications in Java. It is heavy-weight in use because it is generated by the system's host operating system. It contains a large number of classes and methods, which are used for creating and managing GUI.

What is the class hierarchy of AWT?

All the elements like buttons, text fields, scrollbars etc are known as components. In AWT we have classes for each component .To have everything placed on a screen to a particular position, we have to add them to a container. A container is like a screen wherein we are placing components like buttons, text fields, checkbox etc. In short a container contains and controls the layout of components. A container itself is a component thus we can add a container inside container.

What is a Layout manager? What are the different types of Layout managers?

The Layout managers enable us to control the way in which visual components are arranged in the GUI forms by determining the size and position of components within the containers.

Types of Layout Manager:

Flow Layout: It arranges the components in a container like the words on a page. It fills the top line from left to right and top to bottom. The components are arranged in the order as they are added i.e. first components appears at top left, if the container is not wide enough to display all the components, it is wrapped around the line. Vertical and horizontal gap between components can be controlled. The components can be left, centre or right aligned.

Border Layout: It arranges all the components along the edges or the middle of the container i.e. top, bottom, right and left edges of the area. The components added to the top or bottom gets its preferred height, but its width will be the width of the container and also the components added to the left or right gets its preferred width, but its height will be the remaining height of the container. The components added to the centre gets neither its preferred height or width. It covers the remaining area of the container.

Grid Layout: It arranges all the components in a grid of equally sized cells, adding them from the left to right and top to bottom. Only one component can be placed in a cell and each region of the grid will have the same size. When the container is resized, all cells are automatically resized. The order of placing the components in a cell is determined as they were added.

Advantages and disadvantages of using Layout managers.

Advantage: The layout manager automatically positions all the components within the container.

Java provide us with various layout manager to position the controls.. When the size of the applet or the application window changes the size, shape and arrangement of the components also changes in response i.e. the layout managers adapt to the dimensions of appletviewer or the application window.

The layout manager is associated with every Container object.

Disadvantage: If you use a Layout Manager, you can no longer change the size and location of a Component through the setSize and setLocation methods.

Program:

```
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class DisplayInfo extends Frame implements ActionListener {
    private static final long serialVersionUID = 1L;

    TextField nameText = new TextField();
    TextField streetText = new TextField();
    TextField cityText = new TextField();
    TextField pincodeText = new TextField();

    DisplayInfo() {

        Label nameLabel = new Label("Name");
        Label streetLabel = new Label("Street");
        Label cityLabel = new Label("City");
        Label pincodeLabel = new Label("Pincode");

        nameLabel.setBounds(90, 50, 50, 20);
        streetLabel.setBounds(90, 100, 50, 20);
        cityLabel.setBounds(90, 150, 50, 20);
        pincodeLabel.setBounds(90, 200, 50, 20);

        nameText.setBounds(180, 50, 150, 20);
        streetText.setBounds(180, 100, 150, 20);
        cityText.setBounds(180, 150, 150, 20);
        pincodeText.setBounds(180, 200, 150, 20);

        Button button = new Button("MyInfo!");
        button.setBounds(160, 250, 100, 35);
        button.addActionListener(this);

        add(nameLabel);
        add(nameText);
        add(streetLabel);
        add(streetText);
        add(cityLabel);
        add(cityText);
        add(pincodeText);
        add(pincodeLabel);

        add(button);
    }
}
```

```

        setSize(500, 500);
        setLayout(null);
        setVisible(true);
    }

    public static void main(String[] args) {

        new DisplayInfo();

    }

    @Override
    public void actionPerformed(ActionEvent e) {

        String name = nameText.getText();
        String street = streetText.getText();
        String city = cityText.getText();
        String pincode = pincodeText.getText();

        // redirects to Info
        new Info(name, street, city, pincode);
        dispose();
    }
}

public class Info extends Frame {
    private static final long serialVersionUID = 1L;

    Info (String name, String street, String city, String pincode) {

        Font myFont = new Font("Arial",Font.ITALIC,32);

        Label nameLabel = new Label(name);
        nameLabel.setFont(myFont);
        nameLabel.setBounds(40, 50, 400, 40);

        Label streetLabel = new Label(street);
        streetLabel.setFont(myFont);
        streetLabel.setBounds(40, 100, 400, 40);

        Label cityLabel = new Label(city);
        cityLabel.setFont(myFont);
        cityLabel.setBounds(40, 150, 400, 40);

        Label pincodeLabel = new Label(pincode);
        pincodeLabel.setFont(myFont);
        pincodeLabel.setBounds(40, 200, 400, 40);

        add(nameLabel);
        add(streetLabel);
        add(cityLabel);
    }
}

```

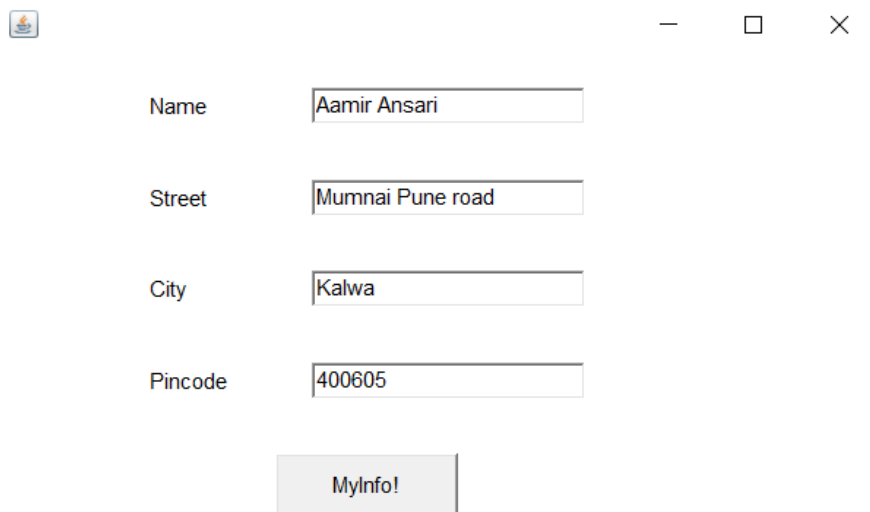
```
        add(pincodelabel);

        setSize(500, 500);
        setLayout(null);
        setVisible(true);

    }

}
```

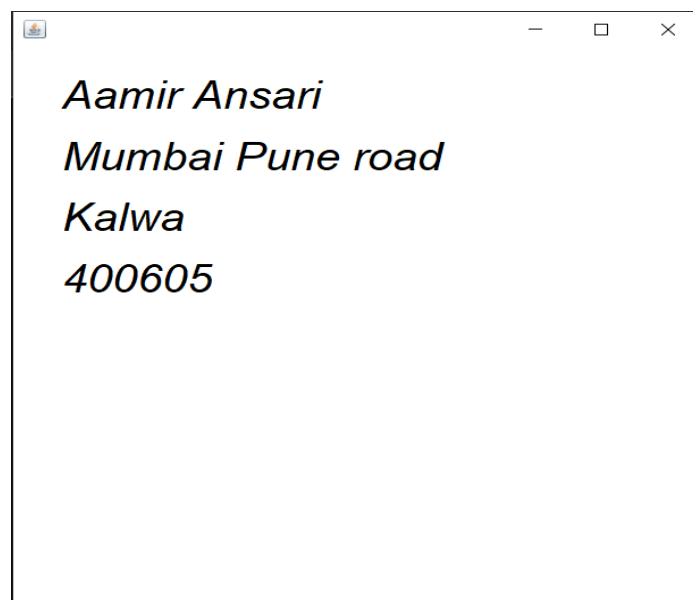
// output



A screenshot of a Java Swing window titled "output". The window contains a form with four text input fields and a button. The fields are labeled "Name", "Street", "City", and "Pincode". The button is labeled "MyInfo!".

Name	Aamir Ansari
Street	Mumnai Pune road
City	Kalwa
Pincode	400605

MyInfo!



A screenshot of a Java Swing window showing the output of the "MyInfo!" button click. The window displays the user's information in a bold, italicized font.

Aamir Ansari
Mumbai Pune road
Kalwa
400605