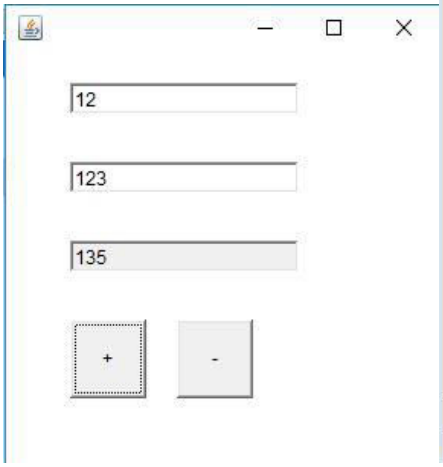
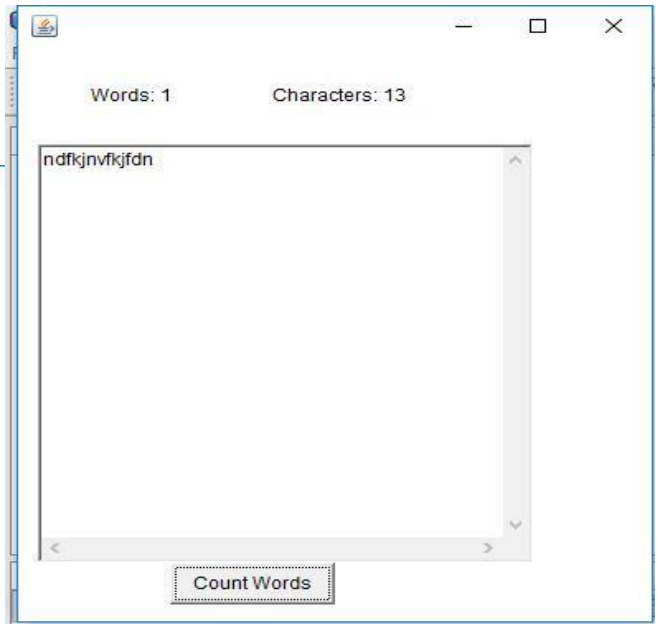
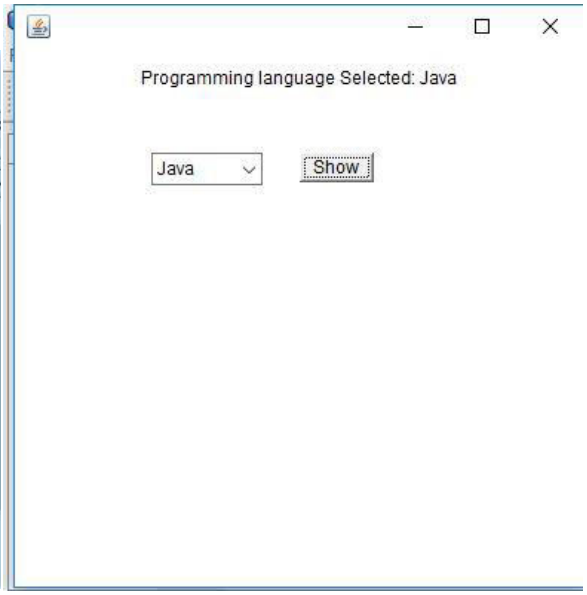
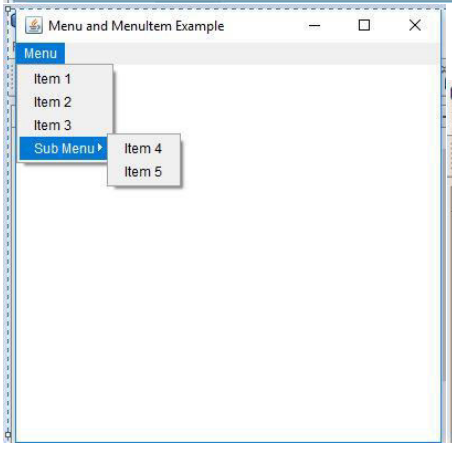
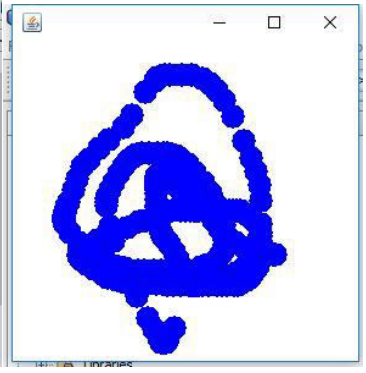

12
123
135
+
-


Words: 1
Characters: 13
ndfkjnvfkjfdn
Count Words


CheckBox Example
C++ checkbox: Checked
C++
Java


Programming language Selected: Java
Java
Show


Menu and MenuItem Example
Menu
Item 1
Item 2
Item 3
Sub Menu
Item 4
Item 5



Sound beep event.

GUIs generate events when the user interacts with GUI. For example,

— Clicking a button
— Moving the mouse
— Closing Window etc.

Both AWT and swing components (not all) generate even
— java.awt.event.*;
— javax.swing.event.*;

These objects tells us about event and its source. Examples are:
— ActionEvent (Clicking a button)
— WindowEvent (Doing something with window e.g. closing , minimizing)

Event Handling Steps

For a programmer the event Handling is a three step process in terms of code

– **Step 1:** Create components which can generate events (Event Generators)

– **Step 2:** Build component (objects) that can handle events (Event Handlers)

– **Step 3:** Register handlers with generators

Event Handling Process

Step 1: Event Generators

The first step is that you create an event generator. You have already seen a lot of event generators like:

– Buttons
– Mouse
– Key
– Window etc

Most of GUI components can be created by calling their constructors. For example

JButton b1 = new JButton("Hello");

Now b1 can generate events

Note: We do not create Mouse/Keys etc as they are system components

Step 2: Event Handlers/ Event Listener

The second step is that you build components that can handle events

First Technique - *By Implementing Listener Interfaces*

– Java defines interfaces for every event type

– If a class needs to handle an event. It needs to implement the corresponding listener interface

– To handle "ActionEvent" a class needs to implement "ActionListener"

– To handle "KeyEvent" a class needs to implement "KeyListener"

– To handle "MouseEvent" a class needs to implement "MouseListener" and so on

Step 3: Registering Handler with Generator

The event generator is told about the object which can handle its events

Event Generators have a method
— addXXXListener(_reference to the object of Handler class_)

For example, if b1 is JButton then
— b1.addActionListener(this); // if listener and generator are same class

Event Handling Example

```
1. import java.awt.*;
2. import javax.swing.*;
3. import java.awt.event.*;
/* Implementing the interface according to the type of the event, i.e. creating event handlpart of step 2 of our process) */
4. public class ActionEventTest
implements ActionListner{
5.    JFrame frame;
6.    JButton hello;

   // setting layout components
7.    public void initGUI ( ) {
8.       frame = new JFrame();
9.       Container cont =  frame.getContentPane();
10.       cont.setLayout(new FlowLayout());
      //Creating event generator step-1 of our process
11.hello = new JButton("Hello");
      /* Registering event handler with event generator.
      Since event handler is in same object that contains
      button, we have used this to pass the reference.(step
      3 of the process) */
12.hello.addActionListener(this);
13.      cont.add(hello);
14.      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
15.      frame.setSize(150, 150);
16.      frame.setVisible(true);
17.   }
  //constructor
18.   public ActionEventTest ( ) {
19.       initGUI();
20.   }
   /* Override actionPerformed method of ActionListener's
      interfacemethod of which will be called when event
      takes place (second part of step 2 of our process) */

21.  public void actionPerformed(ActionEvent event) {
22.    JOptionPane.showMessageDialog(null,"Hello is pressed");
23.  }
24.  public static void main(String args[]) {
25.     ActionEventTest aeTest = new ActionEventTest();
26.  }
                                                                    27.} // end class
```