Graphical User Interface (GUI)

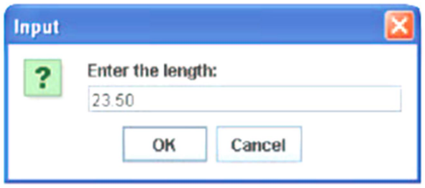
class JOptionPane

Java provides the class JOptionPane, which allows the programmer to use GUI components for I/O. The class JOptionPane is contained in the package javax.swing.

The two methods of this class that we use are: showInputDialog and showMessageDialog. The method showInputDialog allows the user to input a string from the keyboard; the method showMessageDialog allows the programmer to display the results.

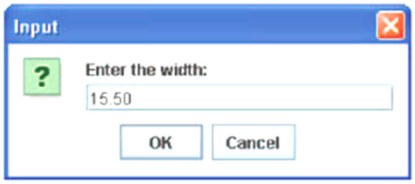
Practice the following examples by hand

```
import javax.swing.JOptionPane;

public class Rectangle
{
    public static void main(String[] args)
    {
        double width, length, area, perimeter;

        String lengthStr, widthStr, outputStr;

        lengthStr =
              JOptionPane.showInputDialog("Enter the length: ");
        length = Double.parseDouble(lengthStr);

        widthStr =
               JOptionPane.showInputDialog("Enter the width: ");
        width = Double.parseDouble(widthStr);

        area = length * width;
        perimeter = 2 * (length + width);

        outputStr = "Length: " + length + "\n" +
                    "Width: " + width + "\n" +
                    "Area: " + area + " square units\n" +
                    "Perimeter: " + perimeter + " units\n";

        JOptionPane.showMessageDialog(null, outputStr,
                              "Rectangle",
                              JOptionPane.INFORMATION_MESSAGE);

        System.exit(0);
    }
}
```



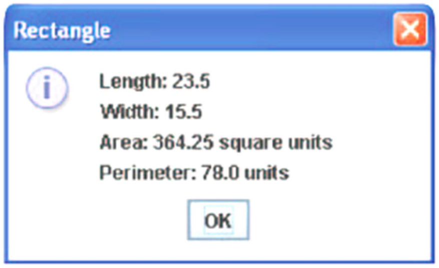
Enter the length:

23.50



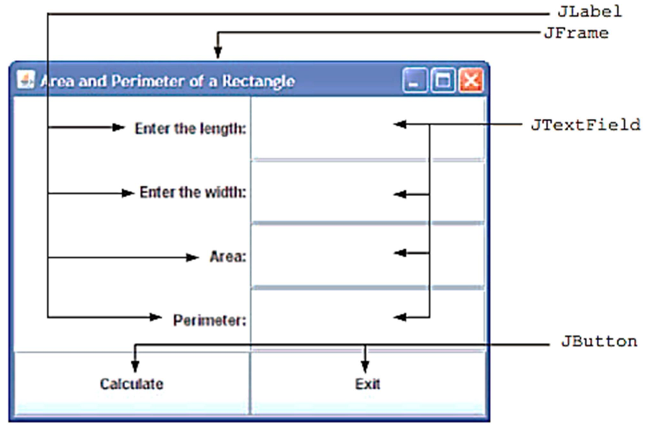
Enter the width:

15.50



Length: 23.5
Width: 15.5
Area: 364.25 square units
Perimeter: 78.0 units

There is better GUI for the above mentioned example. Notice the classes used for developing components.



Java GUI components

class JFrame

The class JFrame provides various methods to control the attributes of a window.

class JLabel

To create labels, you instantiate objects of type JLabel.

class JTextField

Text fields are objects belonging to the class JTextField.

class JTextArea

Objects of JTextArea class either collect multiple lines of input from the user or to display multiple lines of output.

class JCheckBox & JRadioButton

The JCheckBox and JRadioButton classes allow a user to select a value from a set of given values.

class JButton

To create a button, Java provides the class JButton. Clicking a JButton creates an event, known as an action event, which sends a message to another object, known as an action listener.

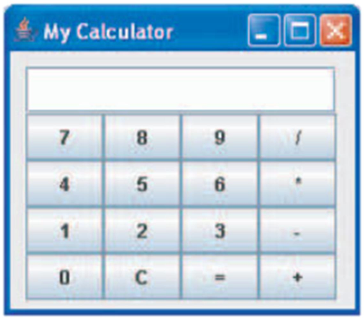
1. For each JButton, you must specify the corresponding listener object. In Java, this is known as registering the listener.
2. You must define the methods that will be invoked when the event is sent to the listener. Normally, you will write these methods and you will never write the code for invocation.

The class ActionListener that handles the action event is an interface so Java does not allow you to instantiate an object of type ActionListener.

Practice the following examples by hand

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class RectangleProgram extends JFrame
{
   private JLabel lengthL, widthL, areaL, perimeterL;

   private JTextField lengthTF, widthTF, areaTF, perimeterTF;

   private JButton calculateB, exitB;

   private CalculateButtonHandler cbHandler;
   private ExitButtonHandler ebHandler;

   private static final int WIDTH = 400;
   private static final int HEIGHT = 300;

   public RectangleProgram()
   {
             //Create the four labels
      lengthL = new JLabel("Enter the length: ",
                                  SwingConstants.RIGHT);

      widthL = new JLabel("Enter the width: ",
                                  SwingConstants.RIGHT);
      areaL = new JLabel("Area: ", SwingConstants.RIGHT);
      perimeterL = new JLabel("Perimeter: ",
                                  SwingConstants.RIGHT);

             //Create the four text fields
      lengthTF = new JTextField(10);
      widthTF = new JTextField(10);
      areaTF = new JTextField(10);
      perimeterTF = new JTextField(10);

             //Create Calculate Button
      calculateB = new JButton("Calculate");
      cbHandler = new CalculateButtonHandler();
      calculateB.addActionListener(cbHandler);

             //Create Exit Button
      exitB = new JButton("Exit");
      ebHandler = new ExitButtonHandler();
      exitB.addActionListener(ebHandler);

             //Set the title of the window
      setTitle("Area and Perimeter of a Rectangle");

             //Get the container
      Container pane = getContentPane();

             //Set the layout
      pane.setLayout(new GridLayout(5, 2));

             //Place the components in the pane
      pane.add(lengthL);
      pane.add(lengthTF);
      pane.add(widthL);
      pane.add(widthTF);
      pane.add(areaL);
      pane.add(areaTF);
      pane.add(perimeterL);
      pane.add(perimeterTF);
      pane.add(calculateB);
      pane.add(exitB);

             //Set the size of the window and display it
      setSize(WIDTH, HEIGHT);
      setVisible(true);
      setDefaultCloseOperation(EXIT_ON_CLOSE);
    }
    private class CalculateButtonHandler implements ActionListener
    {
       public void actionPerformed(ActionEvent e)
       {
          double width, length, area, perimeter;

          length = Double.parseDouble(lengthTF.getText());
          width = Double.parseDouble(widthTF.getText());
          area = length * width;
          perimeter = 2 * (length + width);

          areaTF.setText("" + area);
          perimeterTF.setText("" + perimeter);
       }
    }

   private class ExitButtonHandler implements ActionListener
   {
       public void actionPerformed(ActionEvent e)
       {
           System.exit(0);
       }
   }

   public static void main(String[] args)
   {
       RectangleProgram rectObject = new RectangleProgram();
   }
}
```

TASK: Design a program that simulates a calculator.



Practice Material:

https://www.javatpoint.com/java-swing