Sukkur Institute of Business Administration University
Department of Computer Science
BS – II (CS/SE/AI) Spring 2024

Object Oriented Programming
Lab # 11: To become familiar with GUI (Graphical User Interface)

Instructor: Engr. Zainab Umair Kamangar

Lab Report Rubrics (Add the points in each column, then add across the bottom row to find the total score)					**Total Marks**
S.No	**Criterion**	**0.5**	**0.25**	**0.125**	
1	Accuracy	☐ Desired output	☐ Minor mistakes	☐ Critical mistakes	
2	Timing	☐ Submitted within the given time	☐ 1 day late	☐ More than 1 day late	

<u>Submission Profile</u>

Name:
Enrollment ID:
Comments:

Submission date (dd/mm/yy):
Receiving authority name and signature:

____
Instructor Signature

Note: Submit this lab hand-out in the next lab with attached solved activities and exercises

Objectives

After performing this lab, students will be able to understand,

1. What is GUI
2. Overview of AWT
3. Overview of Swing

What is GUI

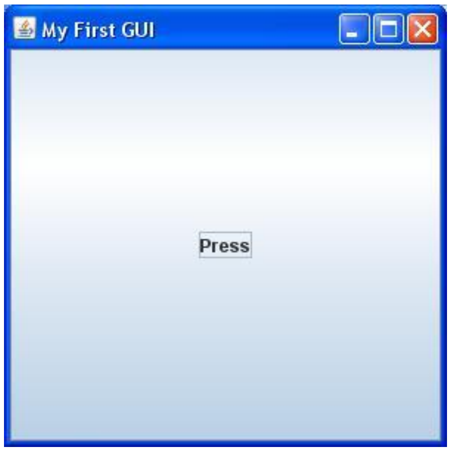
GUI (Graphical User Interface) in Java is an easy-to-use visual experience builder for Java applications. It is mainly made of graphical components like buttons, labels, windows, etc. through which the user can interact with an application. GUI plays an important role to build easy interfaces for Java applications.

Example

Now in this Swing Java Tutorial, let's understand GUI with Java Swing examples.

Example: To learn Java GUI programming in this Java GUI tutorial

```
import javax.swing.*;
class gui{
    public static void main(String args[]){
       JFrame frame = new JFrame("My First GUI");
       frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
       frame.setSize(300,300);
       JButton button = new JButton("Press");
       frame.getContentPane().add(button); // Adds Button to content pane of frame
       frame.setVisible(true);
    }
}
```



How about adding two buttons? Copy the following code into an editor.

```
import javax.swing.*;
class gui{
      public static void main(String args[]){
           JFrame frame = new JFrame("My First GUI");
           frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
           frame.setSize(300,300);
          JButton button1 = new JButton("Button 1");
          JButton button2 = new JButton("Button 2");
          frame.getContentPane().add(button1);
          frame.getContentPane().add(button2);
          frame.setVisible(true);
     }}
```

Overview of AWT

Java AWT (Abstract Window Toolkit) is an API to develop GUI or window-based applications in java.
Java AWT components are platform-dependent i.e. components are displayed according to the view of operating system. AWT is heavyweight i.e. its components are using the resources of OS.
The java.awt package provides classes for AWT api such as TextField, Label, TextArea, RadioButton, CheckBox, Choice, List etc.

Container

The Container is a component in AWT that can contain another components like buttons, textfields, labels etc. The classes that extends Container class are known as container such as Frame, Dialog and Panel.

Window

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

Panel

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

Frame

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

```
import java.awt.*;
class First extends Frame{
First(){
Button b=new Button("click me");
b.setBounds(30,100,80,30);// setting button position
add(b);//adding button into frame
setSize(300,300);//frame size 300 width and 300 height
setLayout(null);//no layout manager
setVisible(true);//now frame will be visible, by default not visible
}
```

```
public static void main(String args[]){
First f=new First();
}}
import java.awt.*;
import java.awt.event.*;
public class ButtonExample {
public static void main(String[] args) {
    Frame f=new Frame("Button Example");
    final TextField tf=new TextField();
    tf.setBounds(50,50, 150,20);
    Button b=new Button("Click Here");
    b.setBounds(50,100,60,30);
    b.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent e){
            tf.setText("Welcome to Javatpoint.");
        }
    });
    f.add(b);f.add(tf);
    f.setSize(400,400);
    f.setLayout(null);
    f.setVisible(true);
}
}
```

Overview of Swing

Swing in Java is a Graphical User Interface (GUI) toolkit that includes the GUI components. Swing provides a rich set of widgets and packages to make sophisticated GUI components for Java applications. Swing is a part of Java Foundation Classes(JFC), which is an API for Java programs that provide GUI.

The Java Swing library is built on top of the Java Abstract Widget Toolkit (**AWT**), an older, platform dependent GUI toolkit. You can use the Java GUI programming components like button, textbox, etc. from the library and do not have to create the components from scratch.

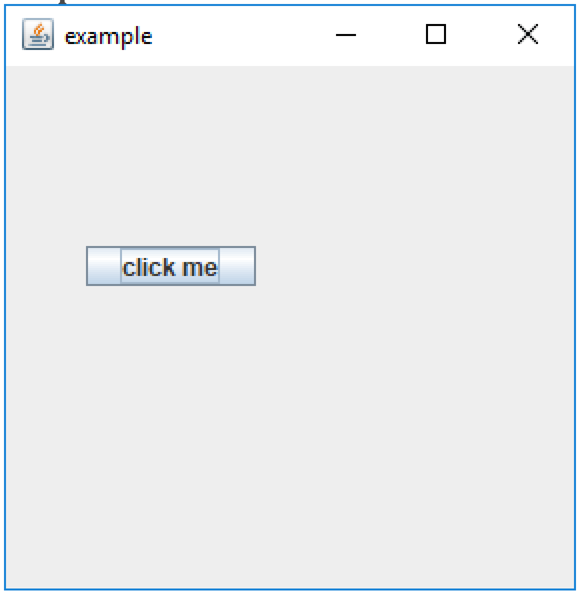
JButton Class

It is used to create a labelled button. Using the ActionListener it will result in some action when the button is pushed. It inherits the AbstractButton class and is platform independent.

Example:

```
import javax.swing.*;
public class example{
public static void main(String args[]) {
JFrame a = new JFrame("example");
JButton b = new JButton("click me");
b.setBounds(40,90,85,20);
a.add(b);
a.setSize(300,300);
a.setLayout(null);
a.setVisible(true);
}
}
```

Output:



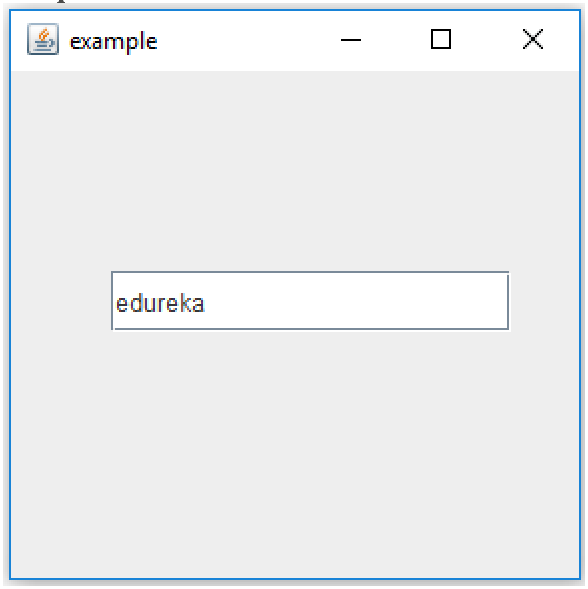
JTextField Class

It inherits the JTextComponent class and it is used to allow editing of single line text.

Example:

```
	import javax.swing.*;
	public class example{
	public static void main(String args[]) {
	JFrame a = new JFrame("example");
	JTextField b = new JTextField("edureka");
	b.setBounds(50,100,200,30);
	a.add(b);
	a.setSize(300,300);
	a.setLayout(null);
	a.setVisible(true);
	}
```

Output:



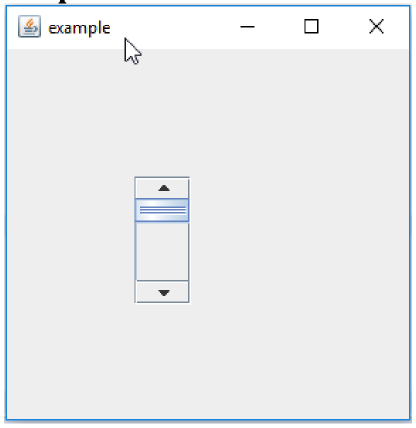
JScrollBar Class

It is used to add scroll bar, both horizontal and vertical.

Example:

```
	import javax.swing.*;
	class example{
	example(){
	JFrame a = new JFrame("example");
	JScrollBar b = new JScrollBar();
	b.setBounds(90,90,40,90);
	a.add(b);
	a.setSize(300,300);
	a.setLayout(null);
	a.setVisible(true);
	}
	public static void main(String args[]){
	new example();
	}
	}
```

Output:

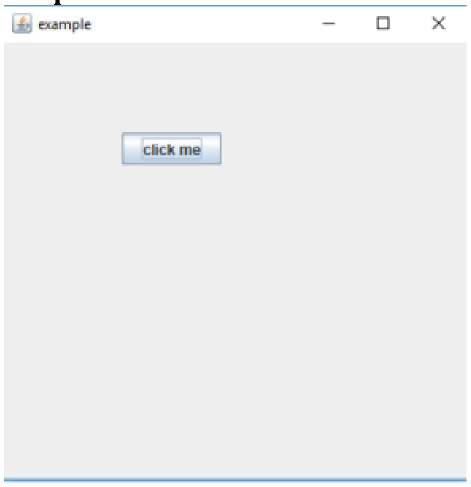


JPanel Class

It inherits the JComponent class and provides space for an application which can attach any other component.

```
import java.awt.*;
import javax.swing.*;
public class Example{
Example(){
JFrame a = new JFrame("example");
JPanel p = new JPanel();
p.setBounds(40,70,200,200);
JButton b = new JButton("click me");
b.setBounds(60,50,80,40);
p.add(b);
a.add(p);
a.setSize(400,400);
a.setLayout(null);
a.setVisible(true);
}
public static void main(String args[])
{
new Example();
}
}
```

Output:

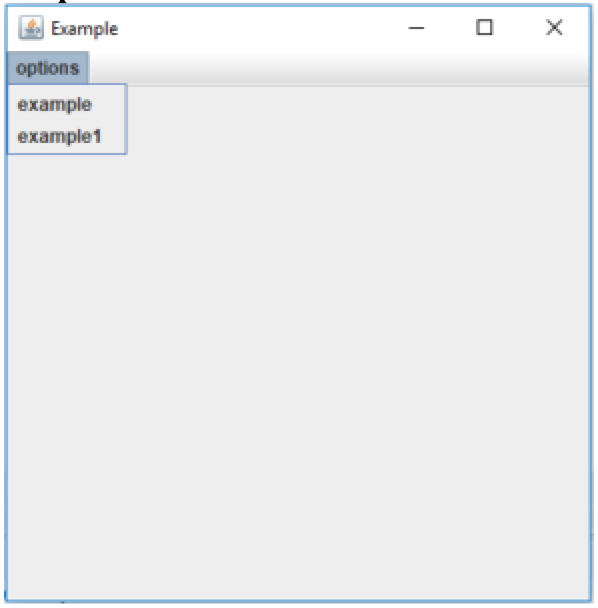


JMenu Class

It inherits the JMenuItem class, and is a pull down menu component which is displayed from the menu bar.

```
import javax.swing.*;
class Example{
JMenu menu;
JMenuItem a1,a2;
Example()
{
JFrame a = new JFrame("Example");
menu = new JMenu("options");
JMenuBar m1 = new JMenuBar();
a1 = new JMenuItem("example");
a2 = new JMenuItem("example1");
menu.add(a1);
menu.add(a2);
m1.add(menu);
a.setJMenuBar(m1);
a.setSize(400,400);
a.setLayout(null);
a.setVisible(true);
}
public static void main(String args[])
{
new Example();
}
}
```

Output:

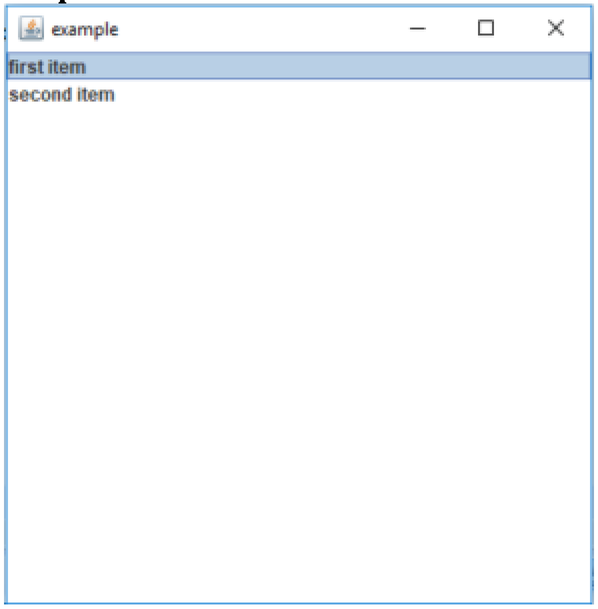


JList Class

It inherits JComponent class, the object of JList class represents a list of text items.
Programming & Frameworks Training

```
import javax.swing.*;
public class Example
{
Example(){
JFrame a  = new JFrame("example");
DefaultListModel<String> l = new DefaultListModel< >();
l.addElement("first item");
l.addElement("second item");
JList<String> b = new JList< >(l);
b.setBounds(100,100,75,75);
a.add(b);
a.setSize(400,400);
a.setVisible(true);
a.setLayout(null);
}
public static void main(String args[])
{
new Example();
}
}
```

Output:

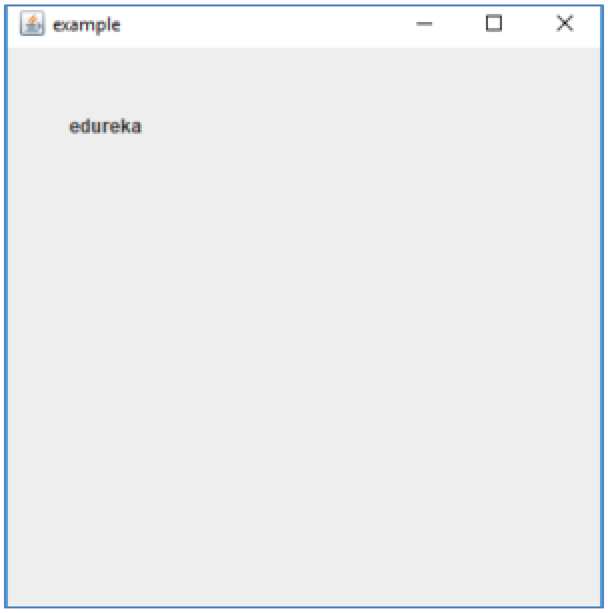


```
first item
second item
```

JLabel Class
It is used for placing text in a container. It also inherits JComponent class.

```
      import javax.swing.*;
      public class Example{
      public static void main(String args[])
      {
      JFrame a = new JFrame("example");
      JLabel b1;
      b1 = new JLabel("edureka");
      b1.setBounds(40,40,90,20);
      a.add(b1);
      a.setSize(400,400);
      a.setLayout(null);
      a.setVisible(true);
      }
      }
```

Output:

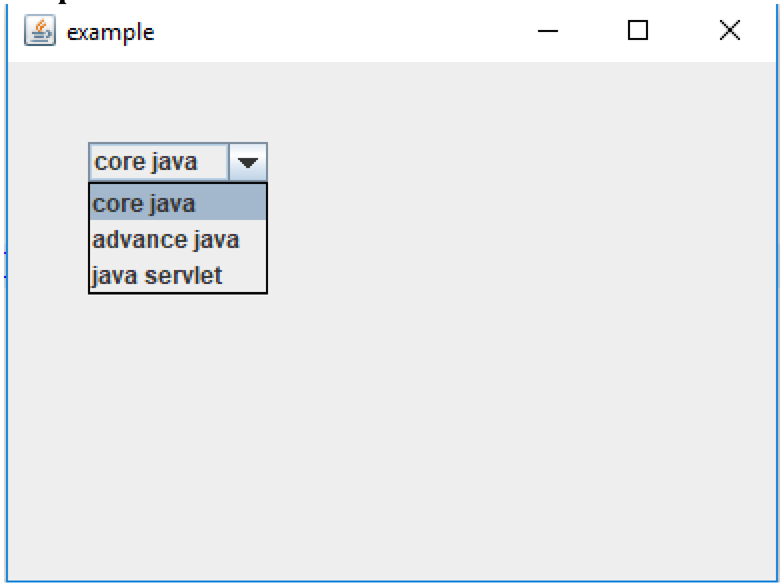


JComboBox Class

It inherits the JComponent class and is used to show pop up menu of choices.

```
import javax.swing.*;
public class Example{
JFrame a;
Example(){
a = new JFrame("example");
string courses[] = { "core java","advance java", "java servlet"};
JComboBox c = new JComboBox(courses);
c.setBounds(40,40,90,20);
a.add(c);
a.setSize(400,400);
a.setLayout(null);
a.setVisible(true);
}
public static void main(String args[])
{
new Example();
}
}
```

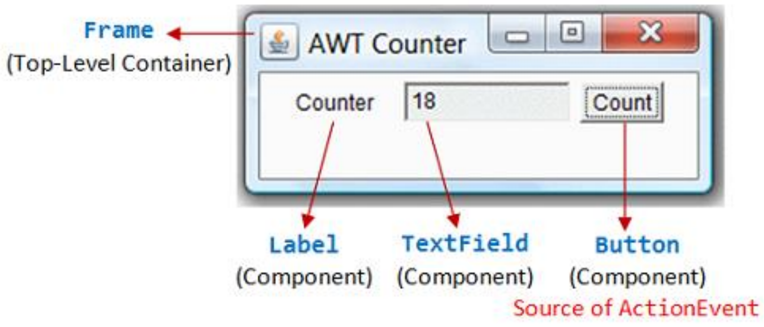
Output:



Exercises

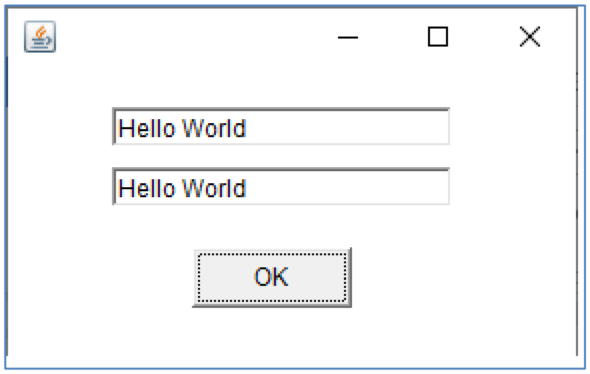
Question: 1

Write GUI application as shown in the Figure. Each time the "Count" button is clicked, the counter value shall increase by 1.



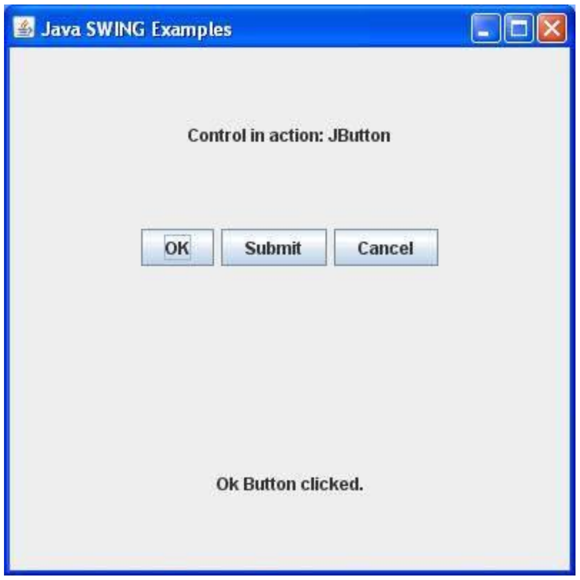
Question: 2

Examine the following picture and create a given GUI. Add 2 textboxes and one button. When a user write a text in textbox1 and click on OK button, text should be copied from textbox1 and added to textbox2.



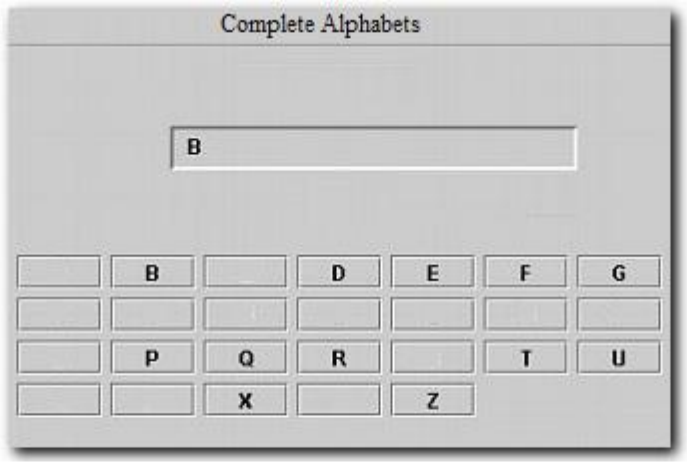
Question: 3

Examine the following picture and wrote a program for given picture.



Question: 4

Examine the following picture and write a code to create a given GUI. For this GUI you need to create and add 27 buttons and one textbox/textfield. Once a user run this application, all alphabets of buttons should not be visible until unless a user click on a particular button then it should display its label. Also if a user write a character in textbox/textfield and hit a enter key then button label should be visible. As given in picture when a user entered B and hit enter button then button B's text is visible now.


Complete Alphabets
B
B
D
E
F
G
P
Q
R
T
U
X
Z