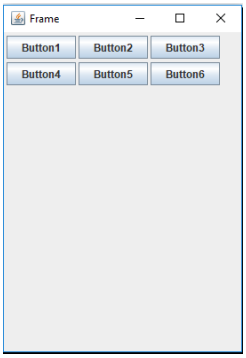
ASSIGNMENT 8: LAB 7

FlowLayout Manager :-

```
import javax.swing.*;
import java.awt.*;
public class Frame
{
	public static void main(String[] args)
	{
		JFrame frame = new JFrame("Frame");
		frame.setSize(300, 400);
		frame.setVisible(true);
		frame.setLayout(new FlowLayout(FlowLayout.LEFT,3,4));
		Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
		int x = (screenSize.width - frame.getWidth()) / 2;
		int y = (screenSize.height - frame.getHeight()) / 2;
		for (int i=1;i<=6;i++)
		frame.getContentPane().add(new JButton("Button"+i));
		frame.setLocation(x,y);
		frame.setVisible(true);
		frame.setDefaultCloseOperation(
		JFrame.EXIT_ON_CLOSE);
	}
}
```



FlowLayout

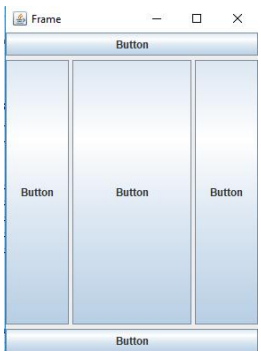
GridLayout Manager :-

```
import javax.swing.*;
import java.awt.*;
public class Frame
{
	public static void main(String[] args)
	{
		JFrame frame = new JFrame("Frame");
		frame.setSize(300, 400);
		frame.setVisible(true);
		frame.setLayout(new GridLayout(4,3,5,6));
		Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
		int x = (screenSize.width - frame.getWidth()) / 2;
		int y = (screenSize.height - frame.getHeight()) / 2;
		for (int i=1;i<=6;i++)
		frame.getContentPane().add(new JButton("Button"+i));
		frame.setLocation(x,y);
		frame.setVisible(true);
		frame.setDefaultCloseOperation(
		JFrame.EXIT_ON_CLOSE);
	}
}
```

GridLayout

BorderLayout Manager :-

```
import javax.swing.*;
import java.awt.*;
public class Frame
{
	public static void main(String[] args)
	{
		JFrame frame = new JFrame("Frame");
		frame.setSize(300, 400);
		frame.setVisible(true);
		frame.setLayout(new GridLayout(4,3,5,6));
		Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
		int x = (screenSize.width - frame.getWidth()) / 2;
		int y = (screenSize.height - frame.getHeight()) / 2;
		frame.getContentPane().add(new JButton("Button"),BorderLayout.CENTER);
		frame.getContentPane().add(new JButton("Button"),BorderLayout.PAGE_START);
		frame.getContentPane().add(new JButton("Button"),BorderLayout.PAGE_END);
		frame.getContentPane().add(new JButton("Button"),BorderLayout.EAST);
		frame.getContentPane().add(new JButton("Button"),BorderLayout.WEST);
		frame.setLocation(x,y);
		frame.setVisible(true);
		frame.setDefaultCloseOperation(
		JFrame.EXIT_ON_CLOSE);
	}
}
```



BorderLayout

Question 2 - Event Handling

Modify your code from Question #1 so that your application will print a message directly on the window indicating which button was clicked. Please use your favourite layout manager.

SOURCE CODE :-

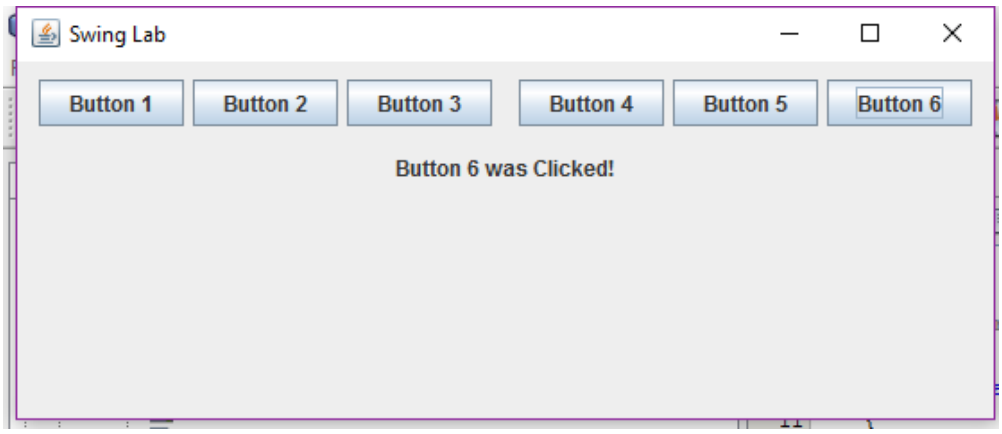
```
import java.util.Scanner;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;
/**
 * @author HARSH
 */
public class ButtonClick extends JFrame
{
   private JButton[] button=new JButton[6];
   public static void main(String[] args)
   {
      ButtonClick b=new ButtonClick();
      b.displayclicked();
   }
   public void displayclicked()
   {
      for(int i=0;i<6;i++)
      button[i]=new JButton("Button "+(i+1));
      this.setSize(500,150);//Size of Frame
      this.setTitle("Swing Lab");
      this.setDefaultCloseOperation(EXIT_ON_CLOSE);
      this.setVisible(true);//Sets Visibility
      Container container=getContentPane();
      container.setLayout(new FlowLayout());
      JPanel p1=new JPanel();
      JPanel p2=new JPanel();
```

```
p1.setLayout(new FlowLayout());  p2.setLayout(new FlowLayout());
for(int i=0;i<3;i++)
   p1.add(button[i]);
for(int i=3;i<6;i++)
   p2.add(button[i]);
container.add(p1);//Adds in first row
container.add(p2);//Adds in second row
JLabel[] label=new JLabel[6];
for(int i=0;i<6;i++)
{
   label[i]=new JLabel("Button "+(i+1)+" was Clicked!");
   label[i].setVisible(false);
}
JPanel panel=new JPanel();
this.add(panel);
for(int i=0;i<6;i++)
   panel.add(label[i]);
button[0].addActionListener(new ActionListener()
{
   public void actionPerformed(ActionEvent arg0)
   {
      label[0].setVisible(true);//Only showing the label corresponding to the button clicked
      for(int i=0;i<6;i++)
      {
         if(i!=0)
            label[i].setVisible(false);//Making other labels invisible
      }
   }   });
button[1].addActionListener(new ActionListener()
{
   public void actionPerformed(ActionEvent arg0)
```

```
    {
        label[1].setVisible(true);
        for(int i=0;i<6;i++)
        {
            if(i!=1)
                label[i].setVisible(false);
        }}          });
button[2].addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent arg0)
    {
        label[2].setVisible(true);
        for(int i=0;i<6;i++)
        {
            if(i!=2)
                label[i].setVisible(false);
        }       }           });
button[3].addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent arg0)
    {
        label[3].setVisible(true);
        for(int i=0;i<6;i++)
        {
            if(i!=3)
                label[i].setVisible(false);
        }           }    });
button[4].addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent arg0)
    {
```

```
        label[4].setVisible(true);
        for(int i=0;i<6;i++)
        {
           if(i!=4)
              label[i].setVisible(false);
        }                    }                });
    button[5].addActionListener(new ActionListener()
    {
       public void actionPerformed(ActionEvent arg0)
       {
          label[5].setVisible(true);
          for(int i=0;i<6;i++)
          {
             if(i!=5)
                label[i].setVisible(false);
          }            }    });
}}
```

OUTPUT :



QUESTION 3

Modify your code from Question #2 so that you can scribble on the window by dragging the mouse.

SOURCE CODE:

```
import java.awt.Container;
import java.awt.FlowLayout;
import java.awt.Graphics;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import java.awt.event.MouseMotionListener;
import java.util.Calendar;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;

/**
 * @author HARSH
 */

public class ButtonsAndScribble extends JFrame implements ActionListener, MouseMotionListener  {

        public static int PANEL_WIDTH = 750;
        public static int PANEL_HEIGHT = 600;
        public static String PANEL_TITLE = "Swing Lab ";
        public static String WAS_CLICKED_AT = " was clicked at ";
        private JButton[] buttons;
```

```
	private JLabel label;
	// former mouse position and current mouse position
	int x0, y0, x, y;
	public static void main(String args[]) {

		ButtonsAndScribble frame = new ButtonsAndScribble();
	}
public ButtonsAndScribble() {

		// initialise buttons, add ActionListener
		buttons = new JButton[6];
		for (int i=0; i<6; i++) {
			buttons[i] = new JButton("Button " + String.valueOf(i));
			buttons[i].addActionListener(this);
		}

		// initialise label
		label = new JLabel();

		this.setDefaultCloseOperation(EXIT_ON_CLOSE);
		this.setSize(PANEL_WIDTH, PANEL_HEIGHT);
		this.setTitle(PANEL_TITLE);
		this.setVisible(true);

		Container myContainer = getContentPane();
		myContainer.setLayout(new FlowLayout());

		// use 2 panels, 3 buttons each
		JPanel p1 = new JPanel();
		p1.setLayout(new FlowLayout());
		JPanel p2 = new JPanel();
```

```
        p2.setLayout(new FlowLayout());

        for (int i=0; i<3; i++) {
            p1.add(buttons[i]);
        }
        for (int j=3; j<buttons.length ;j++) {
            p2.add(buttons[j]);
        }
        // add panels to container
        myContainer.add(p1);
        myContainer.add(p2);
        // add label to container
        myContainer.add(label);
        // add MouseMotionListener
        this.addMouseMotionListener(this);
    }
    @Override
    public void actionPerformed(ActionEvent e) {

        // prepare info on button clicked, with time stamp
        StringBuffer myBuffer = new StringBuffer();
        myBuffer.append(e.getActionCommand()).append(WAS_CLICKED_AT);
        Calendar myCalendar = Calendar.getInstance();
        myCalendar.setTimeInMillis(e.getWhen());
        myBuffer.append(myCalendar.getTime().toString());

        label.setText(myBuffer.toString());
    }
  public void paint(Graphics g) {

        g.drawLine(x0, y0, x, y);
```

```
            //super.paint(g);
        }
        /*
         * From MouseMotionListener
         */
        @Override
        public void mouseDragged(MouseEvent arg0) {

            x0 = arg0.getX();
            y0 = arg0.getY();
            x = x0;
            y = y0;
            repaint();
        }

        /*
         * From MouseMotionListener
         */
        @Override
        public void mouseMoved(MouseEvent arg0) {
            // do nothing
        }}
```

Output: