Java GUI 3: The GUI and User Input

1. TextBoxes

In the previous activities, you had an introductory look at the JButton, JLabel, JPanel, listening for and responding to events, and the placement of GUI elements through the use of the LayoutManager interface.

In the next set of activities you will explore some of the elements designed for user input, namely JTextField, JRadioButton and JCheckbox. You will also take a look at the JList object as an additional output element.

JTextField

The JTextField is a very recognizable element for getting input from the user. The coding to create a text box is:

```
txtInput = new JTextField();
```

Optionally you can pass a number to the constructor to specify the number of columns for the initial display.

```
txtInput = new JTextField(25);
```

The argument (25 in the example) does not limit the number of characters that can be entered. It is merely used to assign an initial size (width) to the JTextField for display purposes.

Remember that anything entered by the user is received as a String. Therefore if your program is looking for numerical input, it will need to parse the data. The JTextField object has a `getText()` method to retrieve the text entered and `setText(String)` method to set the default text, if desired. An example use would be as follows:

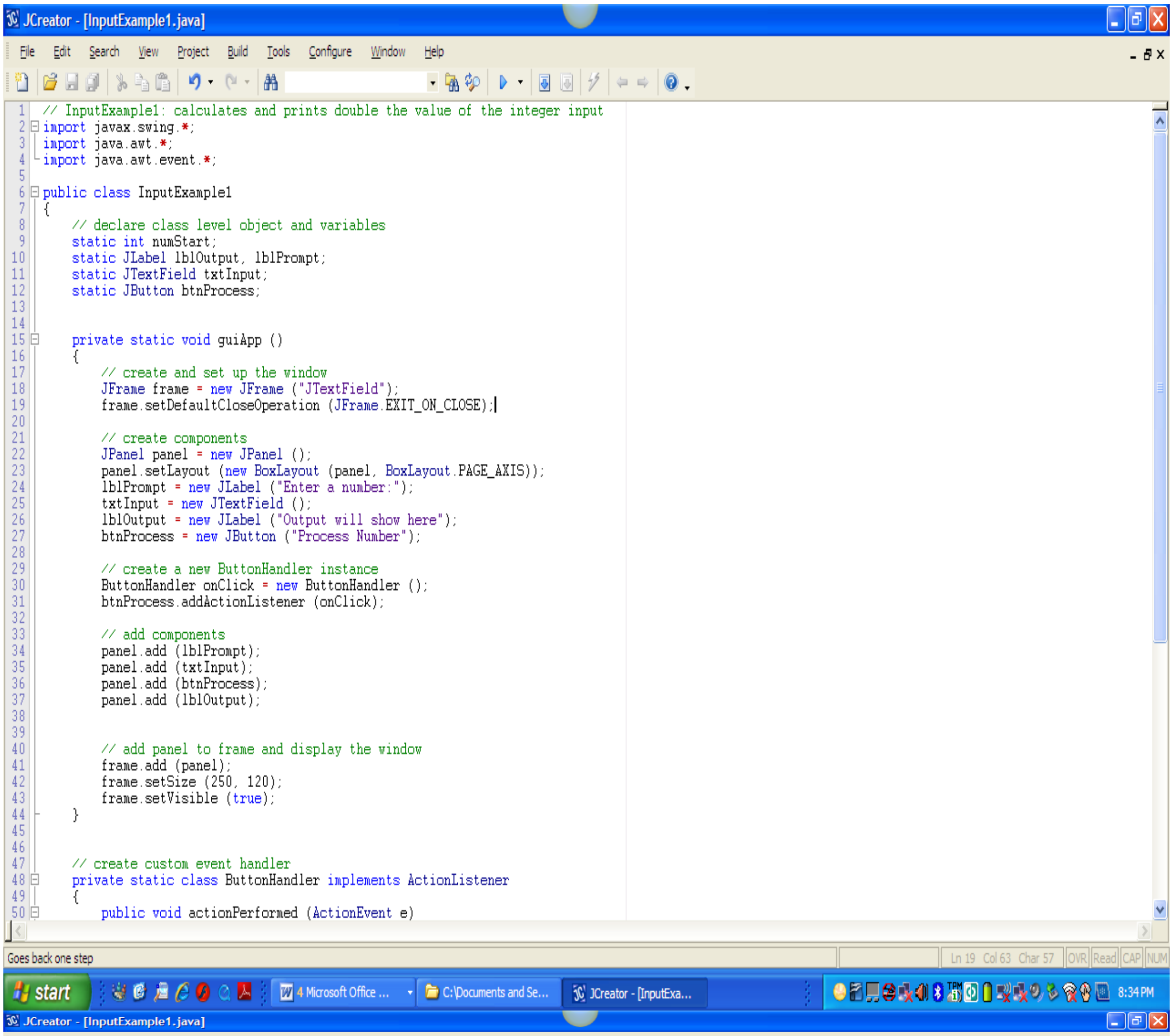
```
txtInput = new JTextField();
String data = txtInput.getText();
numStart = Integer.parseInt(data);
```

A very simple GUI program illustrating the JTextField is illustrated here.

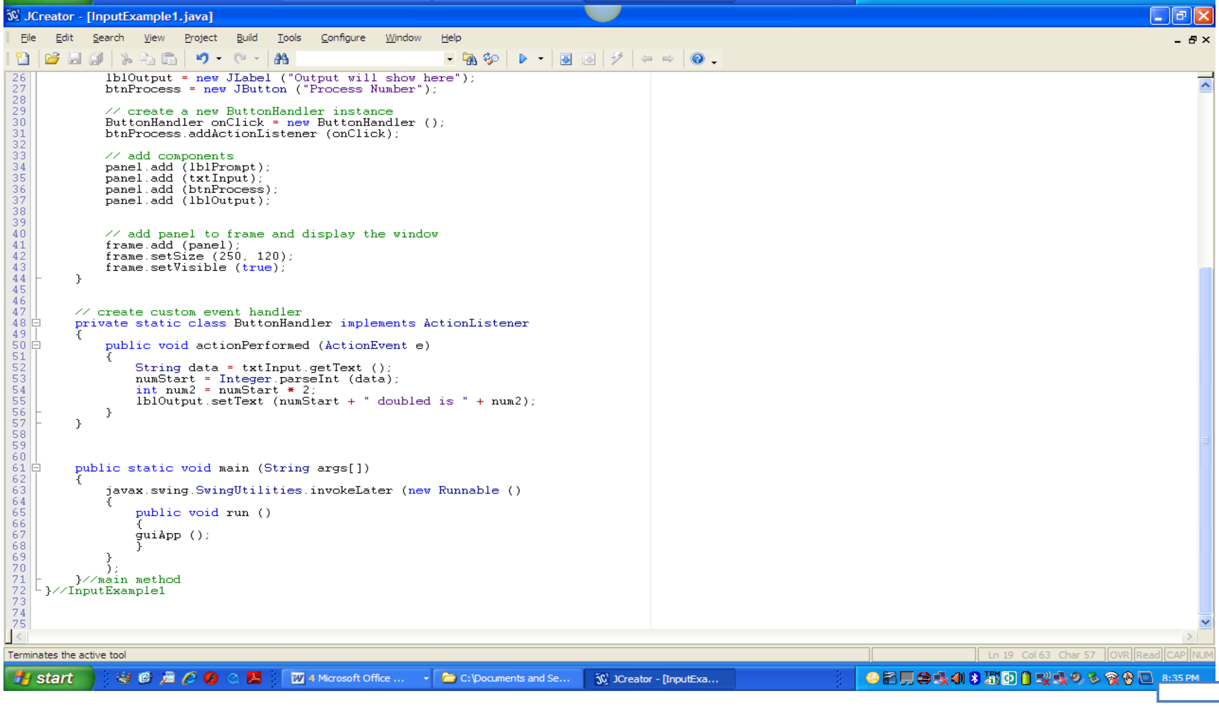
Use the GridLayout to assign 4 rows and 1 column to the panel. In the first row you will insert a label (the user prompt). The second row will contain the textbox for the user input. The third row will contain the button for the user to click. The last row displays the result of doubling the number that the user keyed in. This is programmed in the ButtonHandler method.

Try to complete as much coding as you can, then turn over the page to check your answer.

Sample Code: JTextField Example



```java
// InputExample1: calculates and prints double the value of the integer input
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class InputExample1
{
    // declare class level object and variables
    static int numStart;
    static JLabel lblOutput, lblPrompt;
    static JTextField txtInput;
    static JButton btnProcess;


    private static void guiApp ()
    {
        // create and set up the window
        JFrame frame = new JFrame ("JTextField");
        frame.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);

        // create components
        JPanel panel = new JPanel ();
        panel.setLayout (new BoxLayout (panel, BoxLayout.PAGE_AXIS));
        lblPrompt = new JLabel ("Enter a number:");
        txtInput = new JTextField ();
        lblOutput = new JLabel ("Output will show here");
        btnProcess = new JButton ("Process Number");

        // create a new ButtonHandler instance
        ButtonHandler onClick = new ButtonHandler ();
        btnProcess.addActionListener (onClick);

        // add components
        panel.add (lblPrompt);
        panel.add (txtInput);
        panel.add (btnProcess);
        panel.add (lblOutput);


        // add panel to frame and display the window
        frame.add (panel);
        frame.setSize (250, 120);
        frame.setVisible (true);
    }


    // create custom event handler
    private static class ButtonHandler implements ActionListener
    {
        public void actionPerformed (ActionEvent e)
```



JCreator - [InputExample1.java]
File Edit Search View Project Build Tools Configure Window Help
Terminates the active tool
Ln 19 Col 63 Char 57
OVR Read CAP NUM
start
4 Microsoft Office ...
C:\Documents and Se...
JCreator - [InputExa...
8:35 PM

```java
        lblOutput = new JLabel ("Output will show here");
        btnProcess = new JButton ("Process Number");

        // create a new ButtonHandler instance
        ButtonHandler onClick = new ButtonHandler ();
        btnProcess.addActionListener (onClick);

        // add components
        panel.add (lblPrompt);
        panel.add (txtInput);
        panel.add (btnProcess);
        panel.add (lblOutput);


        // add panel to frame and display the window
        frame.add (panel);
        frame.setSize (250, 120);
        frame.setVisible (true);
    }


    // create custom event handler
    private static class ButtonHandler implements ActionListener
    {
        public void actionPerformed (ActionEvent e)
```

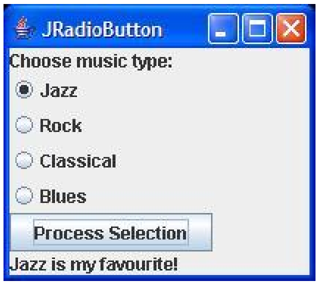
```java
        {
            String data = txtInput.getText ();
            numStart = Integer.parseInt (data);
            int num2 = numStart * 2;
            lblOutput.setText (numStart + " doubled is " + num2);
        }
    }



    public static void main (String args[])
    {
        javax.swing.SwingUtilities.invokeLater (new Runnable ()
        {
            public void run ()
            {
            guiApp ();
            }
        }
        );
    }//main method
}//InputExample1
```

Java GUI 4: The GUI and User Input

2. JRadioButton

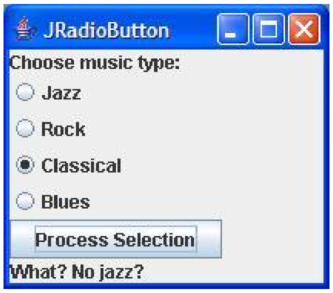
The JRadioButton is used when you want to set up a series of options for the user to select from. You would use a JRadioButton group when you need those choices to be mutually exclusive, i.e. only one option is to be selected at any one time. In order to ensure this, the buttons need to be "grouped" together.

Here is a simple GUI that incorporates JRadioButtons:



Jazz is my favourite!

JRadioButton for user input

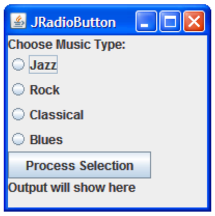


What? No jazz?

Alternate choice being made

As you can see, you would use a JRadioButton group when only one choice is to be made. In order to ensure that only one choice can be made from the group, each JRadioButton is added to a logical ButtonGroup. The code to accomplish this might appear as follows:

```
/* set up JRadioButtons
* jazz, rock, classical, blues would be declared earlier
* as being type JRadioButton
*/
jazz = new JRadioButton ("Jazz", false);
rock = new JRadioButton ("Rock", false);
classical = new JRadioButton ("Classical", false);
blues = new JRadioButton ("Blues", false);

//set up logical relationship for group
music = new ButtonGroup ();
music.add (jazz);
music.add (rock);
music.add (classical);
music.add (blues);
```



Output will show here

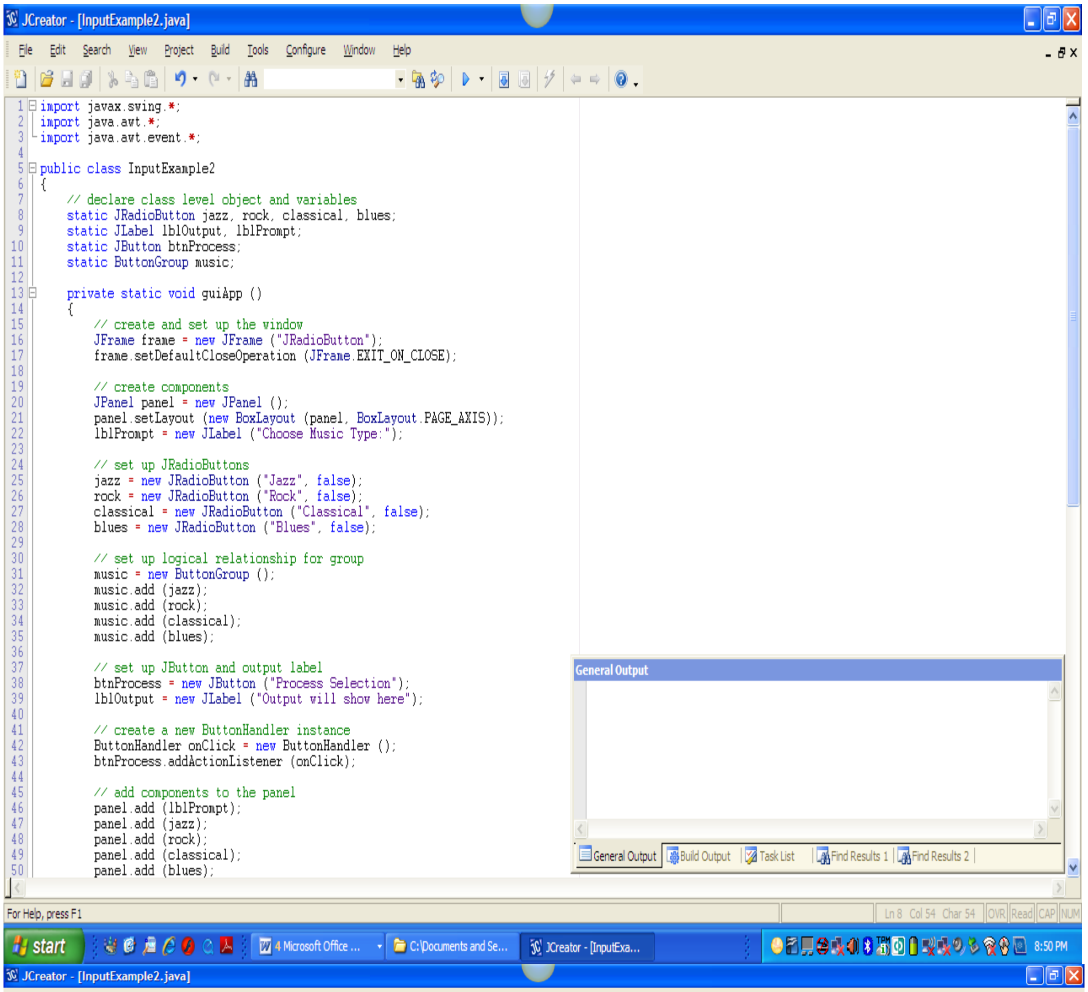
The 'false' argument `jazz = new JRadioButton("Jazz", false)` sets the JRadioButton to not being selected. Using 'true' instead, `jazz = new JRadioButton("Jazz", true)` would mean that the button would appear selected when you run the program. Obviously, you can only set one button in the group to 'true'. You can, of course, create more than one ButtonGroup if your program requires multiple sets of JRadioButton objects.

Testing if a JRadioButton is selected is easily accomplished by checking the isSelected() method:

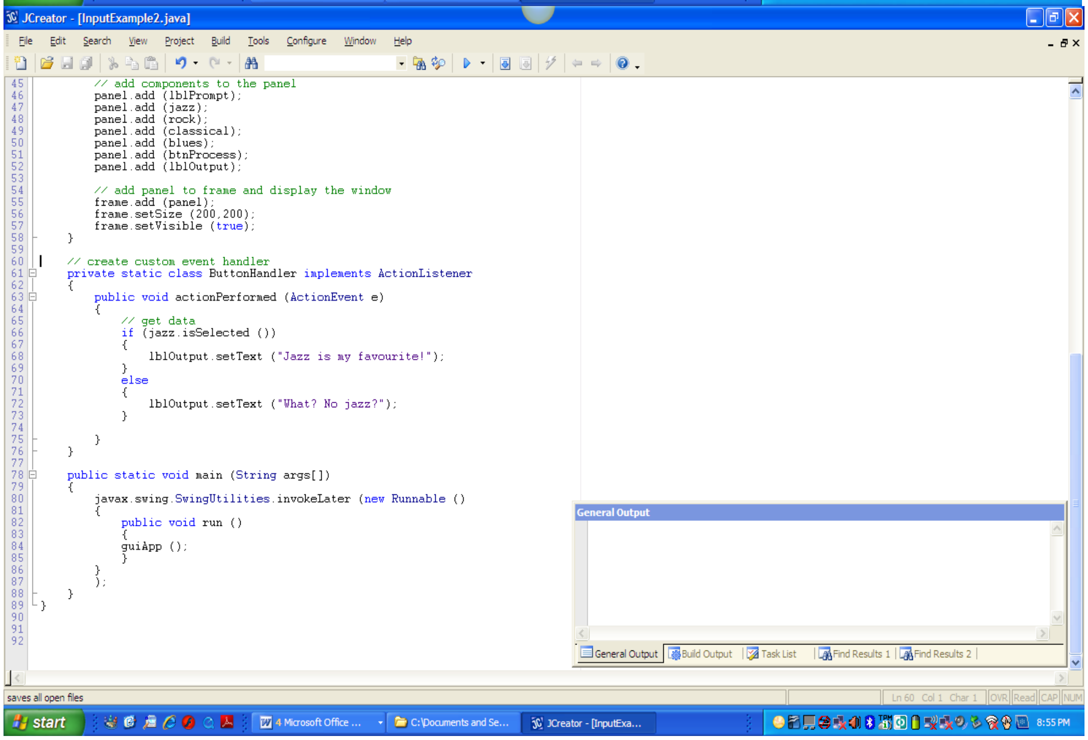
```
if (jazz.isSelected())
{
     //code here...
}
```

Try to complete the RadioButton assignment on your own using a gridlayout of 7 rows and 1 column. If you get stuck turn the page over for help.

Code: JRadioButton Example


JCreator - [InputExample2.java]
File Edit Search View Project Build Tools Configure Window Help
General Output
General Output | Build Output | Task List | Find Results 1 | Find Results 2
For Help, press F1
Ln 8 Col 54 Char 54 OVR Read CAP NUM
start
4 Microsoft Office ...
C:\Documents and Se...
JCreator - [InputExa...
8:50 PM

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

public class InputExample2
{
    // declare class level object and variables
    static JRadioButton jazz, rock, classical, blues;
    static JLabel lblOutput, lblPrompt;
    static JButton btnProcess;
    static ButtonGroup music;

    private static void guiApp ()
    {
        // create and set up the window
        JFrame frame = new JFrame ("JRadioButton");
        frame.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);

        // create components
        JPanel panel = new JPanel ();
        panel.setLayout (new BoxLayout (panel, BoxLayout.PAGE_AXIS));
        lblPrompt = new JLabel ("Choose Music Type:");

        // set up JRadioButtons
        jazz = new JRadioButton ("Jazz", false);
        rock = new JRadioButton ("Rock", false);
        classical = new JRadioButton ("Classical", false);
        blues = new JRadioButton ("Blues", false);

        // set up logical relationship for group
        music = new ButtonGroup ();
        music.add (jazz);
        music.add (rock);
        music.add (classical);
        music.add (blues);

        // set up JButton and output label
        btnProcess = new JButton ("Process Selection");
        lblOutput = new JLabel ("Output will show here");

        // create a new ButtonHandler instance
        ButtonHandler onClick = new ButtonHandler ();
        btnProcess.addActionListener (onClick);

        // add components to the panel
        panel.add (lblPrompt);
        panel.add (jazz);
        panel.add (rock);
        panel.add (classical);
        panel.add (blues);
```


JCreator - [InputExample2.java]
File Edit Search View Project Build Tools Configure Window Help
General Output
General Output | Build Output | Task List | Find Results 1 | Find Results 2
saves all open files
Ln 60 Col 1 Char 1 OVR Read CAP NUM
start
4 Microsoft Office ...
C:\Documents and Se...
JCreator - [InputExa...
8:55 PM

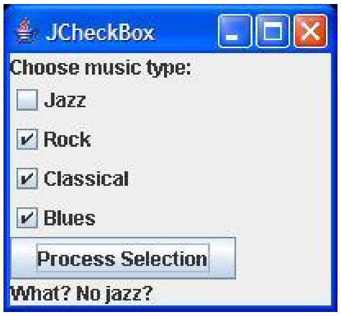
```java
        // add components to the panel
        panel.add (lblPrompt);
        panel.add (jazz);
        panel.add (rock);
        panel.add (classical);
        panel.add (blues);
        panel.add (btnProcess);
        panel.add (lblOutput);

        // add panel to frame and display the window
        frame.add (panel);
        frame.setSize (200,200);
        frame.setVisible (true);
    }

    // create custom event handler
    private static class ButtonHandler implements ActionListener
    {
        public void actionPerformed (ActionEvent e)
        {
            // get data
            if (jazz.isSelected ())
            {
                lblOutput.setText ("Jazz is my favourite!");
            }
            else
            {
                lblOutput.setText ("What? No jazz?");
            }

        }
    }

    public static void main (String args[])
    {
        javax.swing.SwingUtilities.invokeLater (new Runnable ()
        {
            public void run ()
            {
            guiApp ();
            }
        }
        );
    }
}
```

(be sure to copy and paste the main method to the end of this program)

Java GUI 4: The GUI and User Input

3. JCheckBox

Similar to the JRadioButton, a JCheckBox can be used to present a series of items for the user to choose from. The difference is that the choice is non-restrictive; i.e. the user can select as many as required. Here is a sample GUI with multiple JCheckBox objects:



Choose music type:

□ Jazz
☑ Rock
☑ Classical
☑ Blues

What? No jazz?

Multiple choices with JCheckBox

The constructor for the JCheckBox is similar to the JRadioButton.

```
jazz = new JCheckBox("Jazz",false);
rock = new JCheckBox("Rock",false);
classical = new JCheckBox("Classical",false);
blues = new JCheckBox("Blues",false);
```

Determining if a JCheckBox is selected also uses the same `isSelected()` method that the JRadioButton uses.

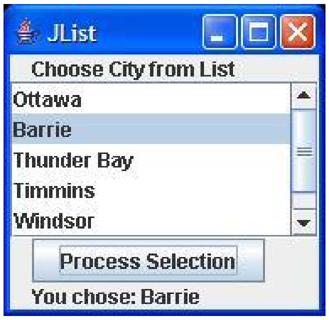
Assignment:

Convert **InputExample2** to use checkboxes instead of radio buttons. Since you want the user to be able to select more than one option you will not need a button group to restrict their choices. Save this checkbox assignment as **InputExample2b**.

Java GUI 4: The GUI and User Input

4. JList

A JList contains a list of items. In a GUI, the JList can be used for both an input and an output element. Here is a sample GUI that asks the user to pick a city from a list of city names:



You chose: Barrie

Choosing items from a JList

A JList is based on a ListModel object. The ListModel contains the item that will be shown in the list. The JList is also not capable of scrolling by itself. So, in order to give the list the scroll capability, it is placed inside a ScrollPane object. Here are the steps that were used to set up the JList in the GUI example after the elements were declared:

[illegible]

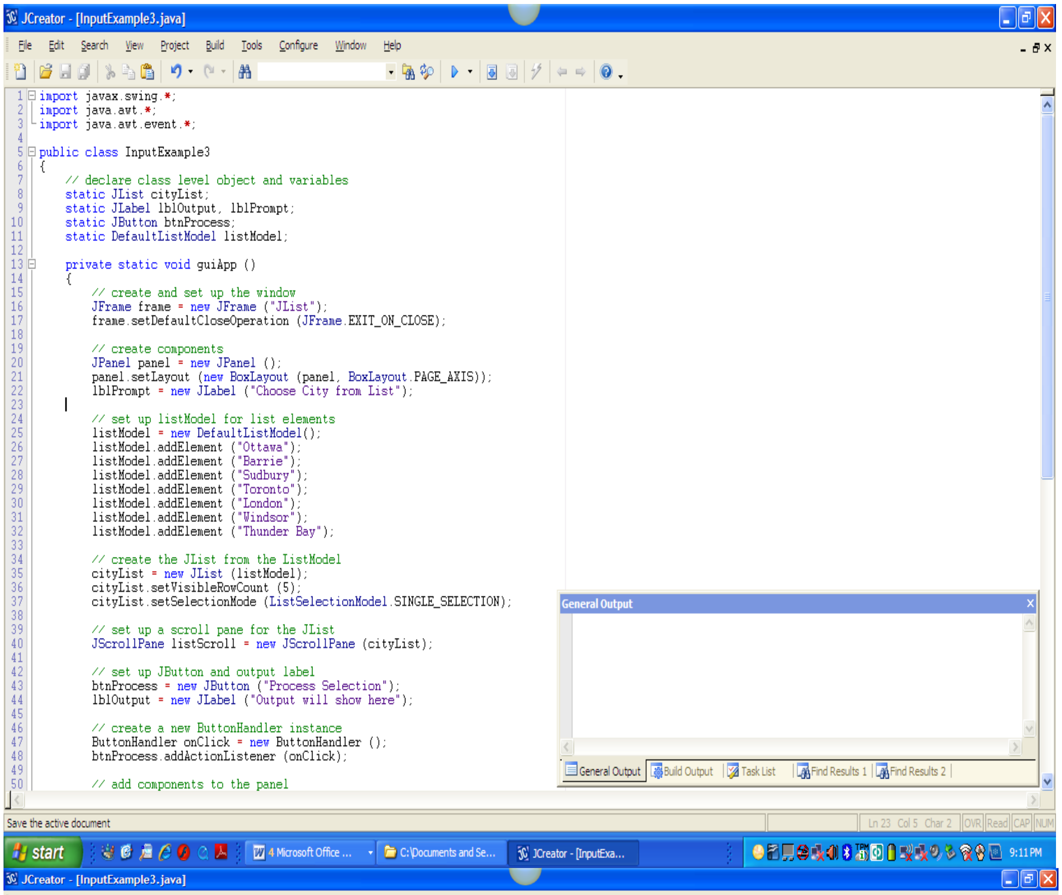
And here is what the code might look like:

```
//set up listModel for list elements
listModel = new DefaultlistModel();
listModel.addElement("Ottawa");
listModel.addElement("Barrie");
listModel.addElement("Thunder Bay");
listModel.addElement("Timmins");
listModel.addElement("Windsor");
listModel.addElement("Sioux Falls");

//create the JList from the ListModel
cityList = new JList(listModel);
cityList.setVisibleRowCount(5);
cityList.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);

//set up a scroll pane for the JList
JScrollPane listScroll = new JScrollPane(cityList);
```

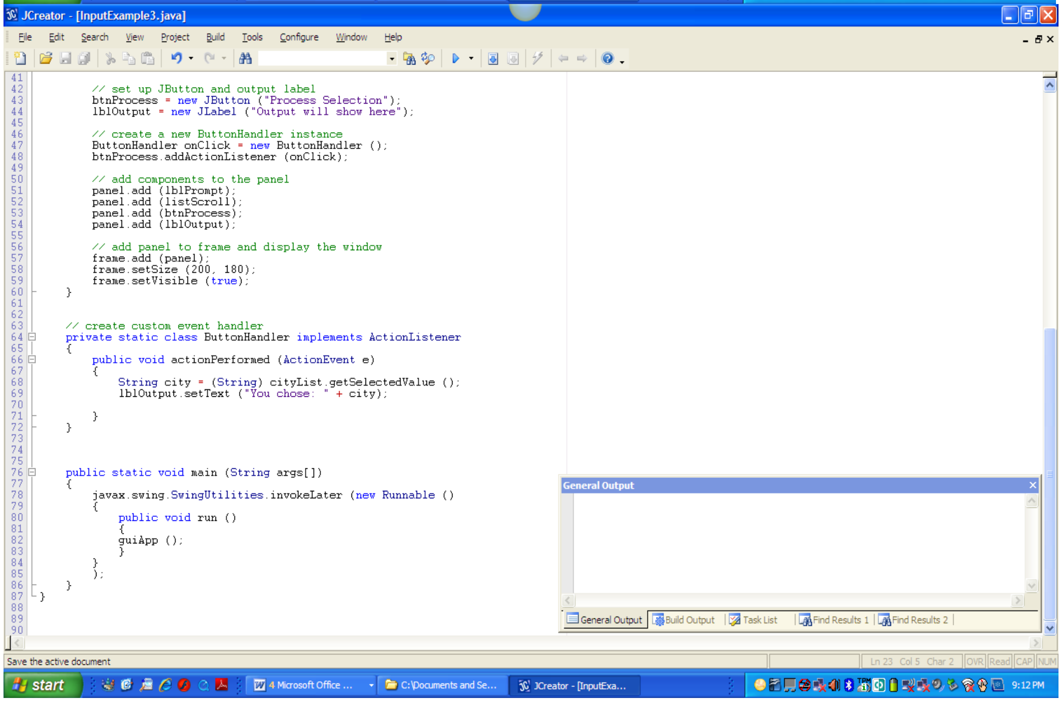
The item that is selected can be read using the `getSelectedValue()` method. If you want to read the text into a String variable, you will need to cast as in:

```
String city = (String) cityList.getSelectedValue();
```

Sample Code: JList Example


JCreator - [InputExample3.java]
File Edit Search View Project Build Tools Configure Window Help
General Output
General Output | Build Output | Task List | Find Results 1 | Find Results 2
Save the active document
Ln 23 Col 5 Char 2 OVR Read CAP NUM
start
4 Microsoft Office ...
C:\Documents and Se...
JCreator - [InputExa...
9:11 PM

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

public class InputExample3
{
    // declare class level object and variables
    static JList cityList;
    static JLabel lblOutput, lblPrompt;
    static JButton btnProcess;
    static DefaultListModel listModel;

    private static void guiApp ()
    {
        // create and set up the window
        JFrame frame = new JFrame ("JList");
        frame.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);

        // create components
        JPanel panel = new JPanel ();
        panel.setLayout (new BoxLayout (panel, BoxLayout.PAGE_AXIS));
        lblPrompt = new JLabel ("Choose City from List");

        // set up listModel for list elements
        listModel = new DefaultListModel();
        listModel.addElement ("Ottawa");
        listModel.addElement ("Barrie");
        listModel.addElement ("Sudbury");
        listModel.addElement ("Toronto");
        listModel.addElement ("London");
        listModel.addElement ("Windsor");
        listModel.addElement ("Thunder Bay");

        // create the JList from the ListModel
        cityList = new JList (listModel);
        cityList.setVisibleRowCount (5);
        cityList.setSelectionMode (ListSelectionModel.SINGLE_SELECTION);

        // set up a scroll pane for the JList
        JScrollPane listScroll = new JScrollPane (cityList);

        // set up JButton and output label
        btnProcess = new JButton ("Process Selection");
        lblOutput = new JLabel ("Output will show here");

        // create a new ButtonHandler instance
        ButtonHandler onClick = new ButtonHandler ();
        btnProcess.addActionListener (onClick);

        // add components to the panel
```


JCreator - [InputExample3.java]
File Edit Search View Project Build Tools Configure Window Help
General Output
General Output | Build Output | Task List | Find Results 1 | Find Results 2
Save the active document
Ln 23 Col 5 Char 2 OVR Read CAP NUM
start
4 Microsoft Office ...
C:\Documents and Se...
JCreator - [InputExa...
9:12 PM

```java

        // set up JButton and output label
        btnProcess = new JButton ("Process Selection");
        lblOutput = new JLabel ("Output will show here");

        // create a new ButtonHandler instance
        ButtonHandler onClick = new ButtonHandler ();
        btnProcess.addActionListener (onClick);

        // add components to the panel
        panel.add (lblPrompt);
        panel.add (listScroll);
        panel.add (btnProcess);
        panel.add (lblOutput);

        // add panel to frame and display the window
        frame.add (panel);
        frame.setSize (200, 180);
        frame.setVisible (true);
    }


    // create custom event handler
    private static class ButtonHandler implements ActionListener
    {
        public void actionPerformed (ActionEvent e)
        {
            String city = (String) cityList.getSelectedValue ();
            lblOutput.setText ("You chose: " + city);

        }
    }



    public static void main (String args[])
    {
        javax.swing.SwingUtilities.invokeLater (new Runnable ()
        {
            public void run ()
            {
            guiApp ();
            }
        }
        );
    }
}
```

(there are no changes to the main method)

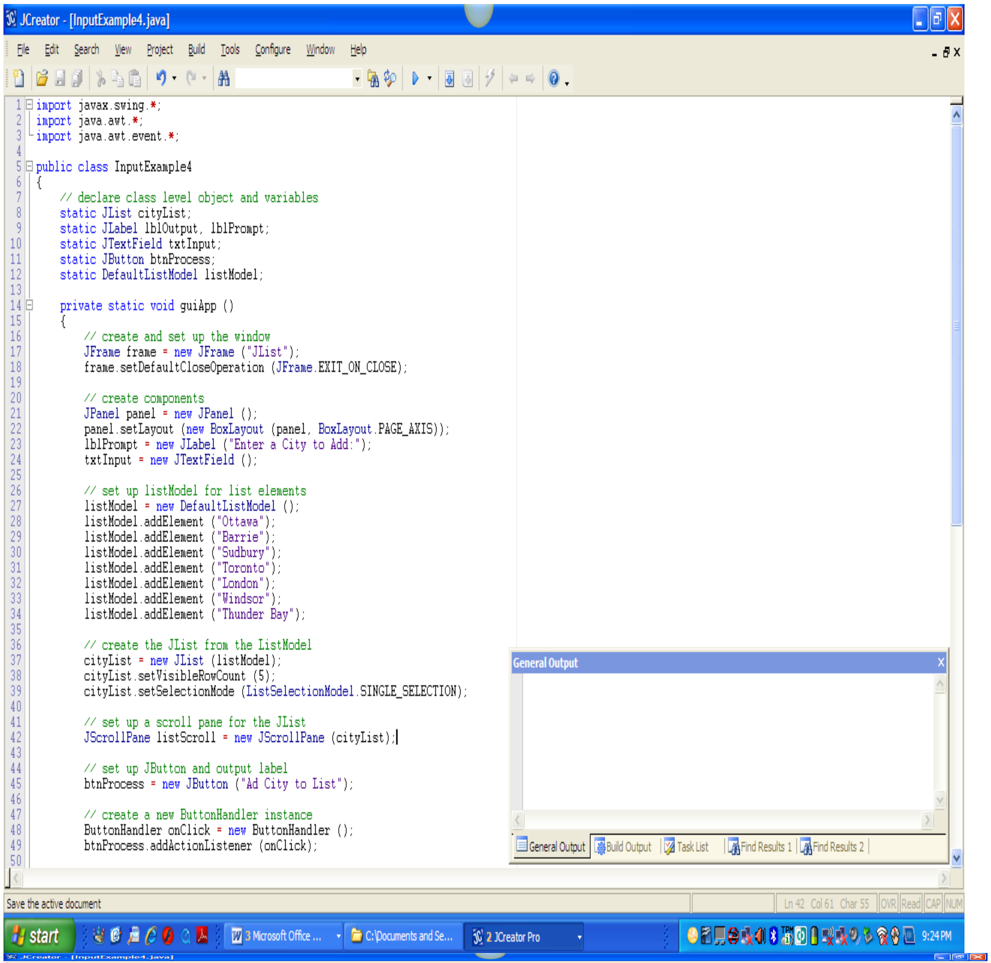
Part 2: Dynamically Adding Items to JList

You can also dynamically add or remove elements from a JList by performing operations on the ListModel that the JList is built from. For example, you can read text from a JTextField and add it to the list with code like the following:

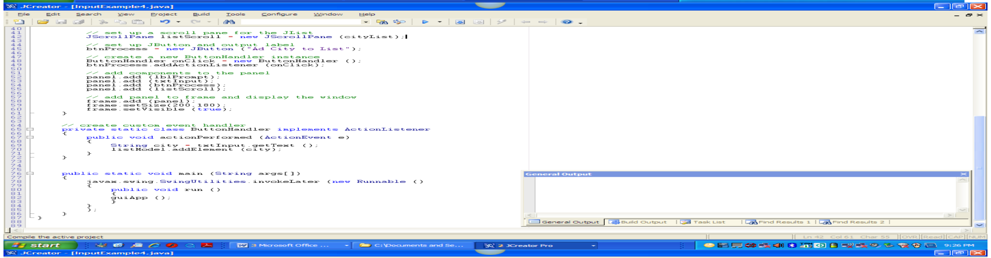
```
String city = txtInput.getText();
listModel.addElement(city);
```

The `addElement()` method will add the item to the end of the list of items as shown. This will be the basis for InputExample4.

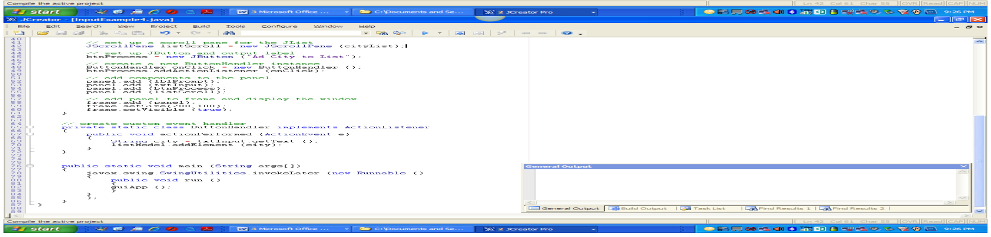
Sample Code: Dynamically Adding Items to JList


JCreator - [InputExample4.java]
File Edit Search View Project Build Tools Configure Window Help
General Output
General Output | Build Output | Task List | Find Results 1 | Find Results 2
Save the active document
Ln 42 Col 61 Char 55 OVR Read CAP NUM
start
3 Microsoft Office ...
C:\Documents and Se...
2 JCreator Pro
9:24 PM

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

public class InputExample4
{
    // declare class level object and variables
    static JList cityList;
    static JLabel lblOutput, lblPrompt;
    static JTextField txtInput;
    static JButton btnProcess;
    static DefaultListModel listModel;

    private static void guiApp ()
    {
        // create and set up the window
        JFrame frame = new JFrame ("JList");
        frame.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);

        // create components
        JPanel panel = new JPanel ();
        panel.setLayout (new BoxLayout (panel, BoxLayout.PAGE_AXIS));
        lblPrompt = new JLabel ("Enter a City to Add:");
        txtInput = new JTextField ();

        // set up listModel for list elements
        listModel = new DefaultListModel ();
        listModel.addElement ("Ottawa");
        listModel.addElement ("Barrie");
        listModel.addElement ("Sudbury");
        listModel.addElement ("Toronto");
        listModel.addElement ("London");
        listModel.addElement ("Windsor");
        listModel.addElement ("Thunder Bay");

        // create the JList from the ListModel
        cityList = new JList (listModel);
        cityList.setVisibleRowCount (5);
        cityList.setSelectionMode (ListSelectionModel.SINGLE_SELECTION);

        // set up a scroll pane for the JList
        JScrollPane listScroll = new JScrollPane (cityList);

        // set up JButton and output label
        btnProcess = new JButton ("Ad City to List");

        // create a new ButtonHandler instance
        ButtonHandler onClick = new ButtonHandler ();
        btnProcess.addActionListener (onClick);
```


JCreator - [InputExample4.java]
File Edit Search View Project Build Tools Configure Window Help
General Output
Compile the active project
start
3 Microsoft Office ...
C:\Documents and Se...
2 JCreator Pro

```
        // set up a scroll pane for the JList
        JScrollPane listScroll = new JScrollPane (cityList);

        // set up JButton and output label
        btnProcess = new JButton ("Ad City to List");

        // create a new ButtonHandler instance
        ButtonHandler onClick = new ButtonHandler ();
        btnProcess.addActionListener (onClick);

        // add components to the panel
        panel.add (lblPrompt);
        panel.add (txtInput);
        panel.add (btnProcess);
        panel.add (listScroll);

        // add panel to frame and display the window
        frame.add (panel);
        frame.setSize(200,180);
        frame.setVisible (true);
    }

    // create custom event handler
    private static class ButtonHandler implements ActionListener
    {
        public void actionPerformed (ActionEvent e)
        {
            String city = txtInput.getText ();
            listModel.addElement (city);
        }
    }


    public static void main (String args[])
    {
        javax.swing.SwingUtilities.invokeLater (new Runnable ()
        {
            public void run ()
            {
                guiApp ();
            }
        });
    }
}
```

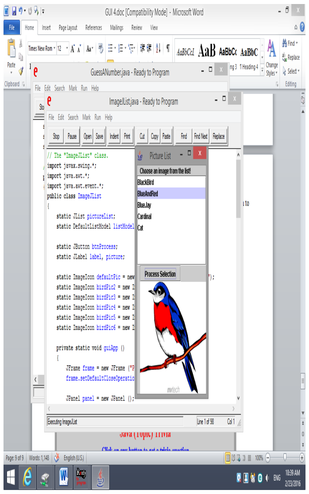


```
        // set up a scroll pane for the JList
        JScrollPane listScroll = new JScrollPane (cityList);

        // set up JButton and output label
        btnProcess = new JButton ("Ad City to List");

        // create a new ButtonHandler instance
        ButtonHandler onClick = new ButtonHandler ();
        btnProcess.addActionListener (onClick);

        // add components to the panel
        panel.add (lblPrompt);
        panel.add (txtInput);
        panel.add (btnProcess);
        panel.add (listScroll);

        // add panel to frame and display the window
        frame.add (panel);
        frame.setSize(200,180);
        frame.setVisible (true);
    }

    // create custom event handler
    private static class ButtonHandler implements ActionListener
    {
        public void actionPerformed (ActionEvent e)
        {
            String city = txtInput.getText ();
            listModel.addElement (city);
        }
    }


    public static void main (String args[])
    {
        javax.swing.SwingUtilities.invokeLater (new Runnable ()
        {
            public void run ()
            {
                guiApp ();
            }
        });
    }
}
```

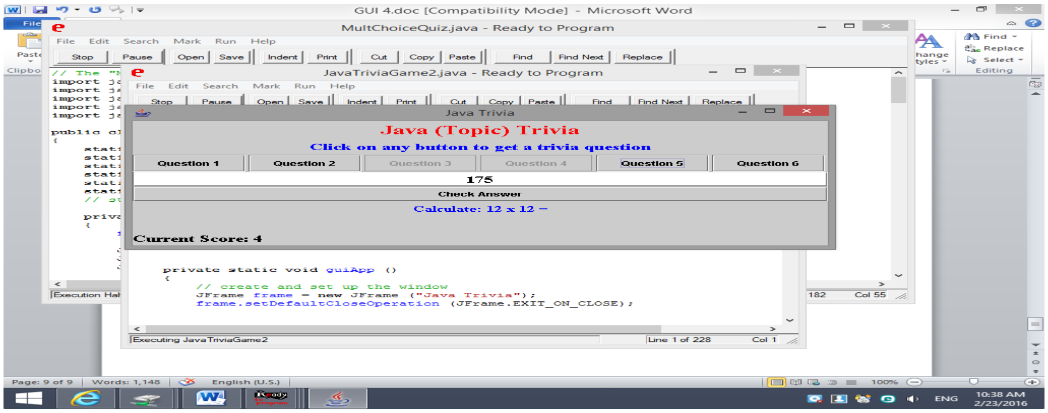
These activities have presented some of the basic properties and methods of typical widgets that are commonly used in GUI programs. The best way to get more comfortable with them is to apply them to some programming assignments.

Assignments

Create a GUI Picture-Changer application similar to the one shown below. After the user selects an item from the list, they will click the JButton which will change the picture shown in the area below. You may change the layout if you wish. You must have 1 default image and at least 5 additional options in the JList.


GuessANumber.java - Ready to Program
ImageJList.java - Ready to Program
File Edit Search Mark Run Help
Stop Pause Open Save Indent Print Cut Copy Paste Find Find Next Replace
Picture List
Choose an image from the list!
BlackBird
BlueAndRed
BlueJay
Cardinal
Cat
Process Selection
Executing ImageJList
Line 1 of 98 Col 1
Java (Topic) Trivia
Click on any button to get a trivia question
Page: 9 of 9 Words: 1,148 English (U.S.) 100%

```
// The "ImageJList" class.
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class ImageJList
{
    static JList pictureList;
    static DefaultListModel listModel

    static JButton btnProcess;
    static JLabel label, picture;

    static ImageIcon defaultPic = new                         ");
    static ImageIcon birdPic2 = new I
    static ImageIcon birdPic3 = new I
    static ImageIcon birdPic4 = new I
    static ImageIcon birdPic5 = new I
    static ImageIcon birdPic6 = new I

    private static void guiApp ()
    {
        JFrame frame = new JFrame ("P
        frame.setDefaultCloseOperatio

        JPanel panel = new JPanel ();
```



GUI 4.doc [Compatibility Mode] - Microsoft Word

MultChoiceQuiz.java - Ready to Program

JavaTriviaGame2.java - Ready to Program

Java Trivia

Java (Topic) Trivia

Click on any button to get a trivia question

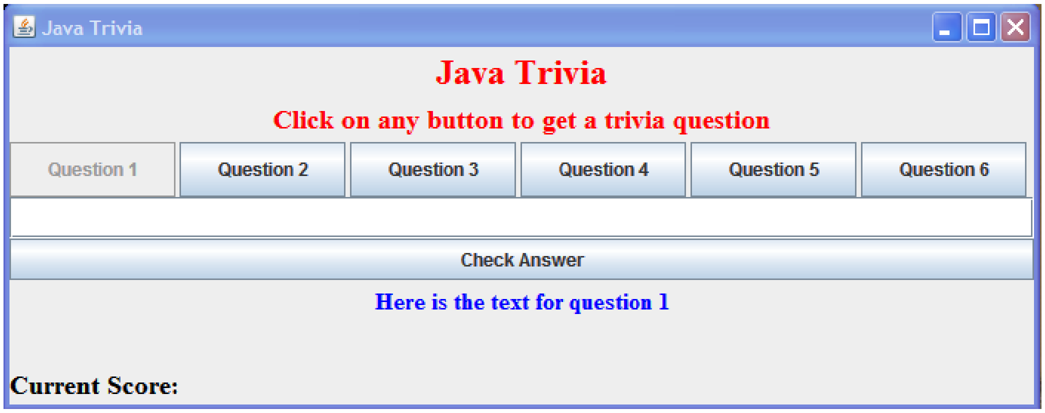
175

Calculate: 12 x 12 =

Current Score: 4

```
    private static void guiApp ()
    {
        // create and set up the window
        JFrame frame = new JFrame ("Java Trivia");
        frame.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);
```

Executing JavaTriviaGame2   Line 1 of 228   Col 1



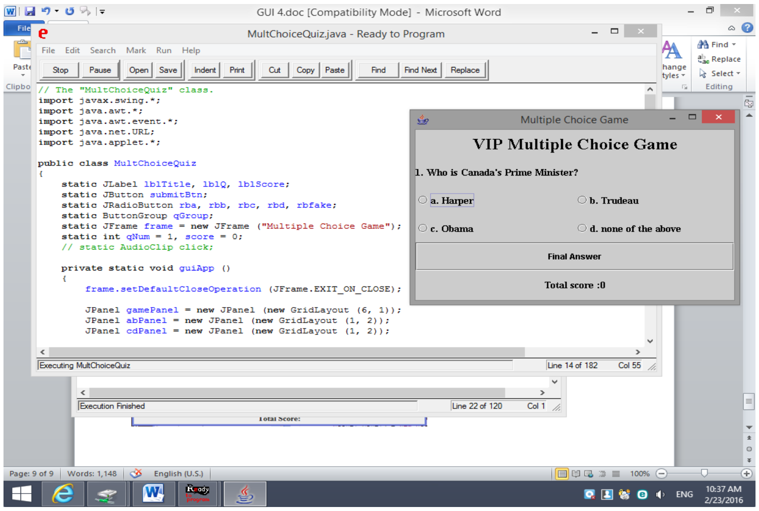
Java Trivia

Java Trivia

Click on any button to get a trivia question

Here is the text for question 1

Current Score:



```
// The "MultChoiceQuiz" class.
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.net.URL;
import java.applet.*;

public class MultChoiceQuiz
{
    static JLabel lblTitle, lblQ, lblScore;
    static JButton submitBtn;
    static JRadioButton rba, rbb, rbc, rbd, rbfake;
    static ButtonGroup qGroup;
    static JFrame frame = new JFrame ("Multiple Choice Game");
    static int qNum = 1, score = 0;
    // static AudioClip click;

    private static void guiApp ()
    {
        frame.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);

        JPanel gamePanel = new JPanel (new GridLayout (6, 1));
        JPanel abPanel = new JPanel (new GridLayout (1, 2));
        JPanel cdPanel = new JPanel (new GridLayout (1, 2));
```

Executing MultChoiceQuiz | Line 14 of 182 | Col 55

Execution Finished | Line 22 of 120 | Col 1

Multiple Choice Game

VIP Multiple Choice Game

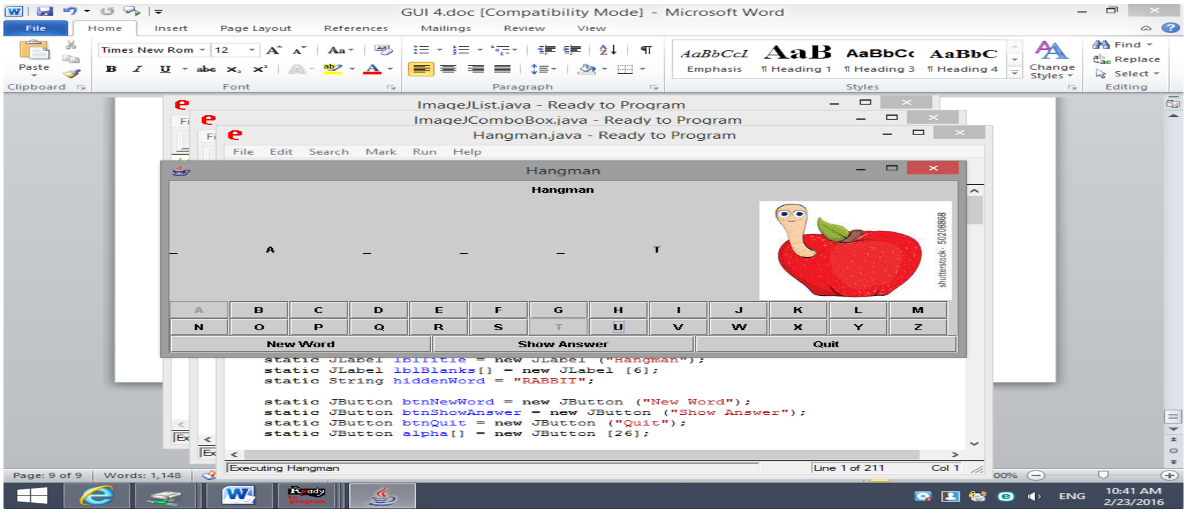
1. Who is Canada's Prime Minister?

○ a. Harper ○ b. Trudeau

○ c. Obama ○ d. none of the above

Final Answer

Total score :0

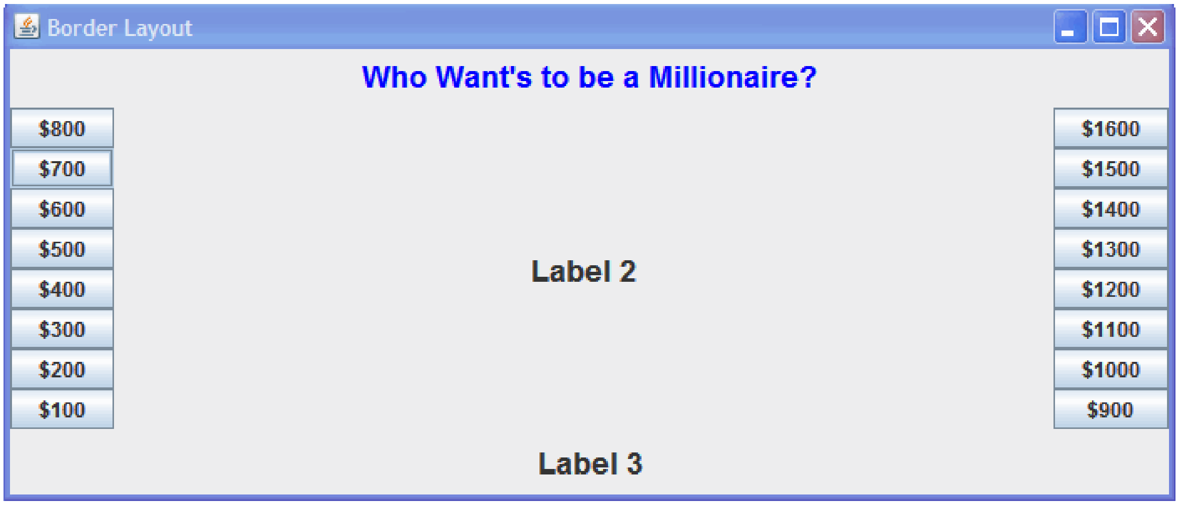


Hangman

_ A _ _ _ T

```
    static JLabel lblTitle = new JLabel ("Hangman");
    static JLabel lblBlanks[] = new JLabel [6];
    static String hiddenWord = "RABBIT";

    static JButton btnNewWord = new JButton ("New Word");
    static JButton btnShowAnswer = new JButton ("Show Answer");
    static JButton btnQuit = new JButton ("Quit");
    static JButton alpha[] = new JButton [26];
```

Executing Hangman | Line 1 of 211 | Col 1



Who Want's to be a Millionaire?

Label 2

Label 3