**Introduction**

Java applets—Java programs that typically execute in a browser. We overview the JDK’s sample applets, then show you how to write and execute your own applets.

An *applet* - a small program embed in Web pages.

With this applet, the Web page becomes “alive”. Each applet in the Web page is given the amount of space, usually measured in pixels. This is the space that the applet will use when it displays itself at the Web page.

**Applet Containers**

The application in which an applet executes is known as the **applet container**. It’s the applet container’s responsibility to load the applet’s class(es), create an instance of the applet and manage its life cycle. The Java Development Kit (JDK) includes one called the **appletviewer** for testing applets as you develop them and before you embed them in web pages.We demonstrate applets using both the appletviewer and web browsers, which execute Java applets via the **Java Plug-In**. Some browsers don’t come with the plug-in by default. You can visit java.com to determine whether your browser is ready to execute Java applets. If not, you can click the **Free Java Download** button to install Java for your browser. Several popular browsers are supported. We tested our applets in Mozilla’s Firefox 3.6, Microsoft’s Internet Explorer 8, Google’s Chrome, Opera 11 and Apple’s Safari 5.

**Using Command Label, Button and Text Field**

Label – for labelling.

CommandButton - is used to begin, interrupt or end a process.

TextField – display information entered by the user or assigned to the Text property.

Example 1: Design three labels.

1. At the Microsoft NotePad, write the following code:

import java.awt.\*;

import java.awt.event.\*;

public class labelsname extends Applet

{

Label label1;

Label label2;

Label label3;

TextField text1;

Font fnt1= new Font("Arial",Font.BOLD,30);

Font fnt2= new Font("Arial",Font.ITALIC,30);

Font fnt3= new Font("Arial",Font.BOLD|Font.ITALIC,30);

public void init()

{

text1= new TextField(20);

text1.setBackground(Color.pink);

label1 = new Label("Computer");

label1.setBackground(Color.red);

label1.setForeground(Color.green);

label1.setFont(fnt1);

add(label1);

label2 = new Label("Science");

label2.setBackground(Color.blue);

label2.setForeground(Color.black);

label2.setFont(fnt2);

add(label2);

label3 = new Label("Department");

label3.setBackground(Color.yellow);

label3.setForeground(Color.gray);

label3.setFont(fnt3);

add(label3);

}

}

1. Save the java applet with filename: ***labelsname.java***
2. At DOS prompt, activate the java compiler batch file: **javac** ***labelsname.java***
3. If no error(s), a file ***labelsname.class*** will be created.
4. Create a file ***labelsname.html*** with the following code:

<html>

<!-Module 1 - written in Java Applet>

<head>

<title>Three labels</title>

</head>

<body>

<hr>

<applet

code=labelsname.class

width=400

height=400>

</applet>

<hr>

</body>

</html>

1. Type the following at DOS prompt:

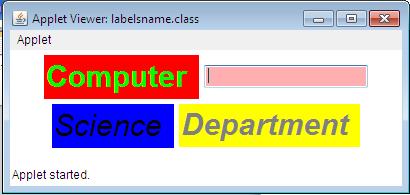
C:\javaprog\**labelsname.html**

Or

Using appletviewer:

C:\javaprog\appletviewer **labelsname.html**

**OUTPUT:**

****

Example 2. Design and develop a java applet that will display the message “Welcome to Computer Science!” at the text field on the applet.

1. At Microsoft NotePad, Write the following code:

import java.awt.\*;

public class text1 extends java.applet.Applet

{

TextField txtText1;

public void init()

{

txtText1 = new TextField (20);

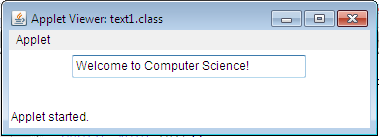
add(txtText1);

txtText1.setText(“Welcome to Computer Science!”);

}

}

**OUTPUT:**



Example 3. Design and develop a java applet when the user clicks the command button, the message “ Command button clicked” will be displayed at the text field.

1. Source code:

import java.applet.Applet;

import java.awt.\*;

import java.awt.event.\*;

public class button1 extends Applet implements ActionListener

{

TextField txtText1;

Button cmdButton1;

public void init()

{

txtText1 = new TextField(20);

add(txtText1);

cmdButton1 = new Button(“Click”);

add(cmdButton1);

cmdButton1.addActionListener(this);

}

public void actionPerformed(ActionEvent objEvent)

{

String strMessage = new String(“Command button clicked”);

if (objEvent.getSource() ==cmdButton1)

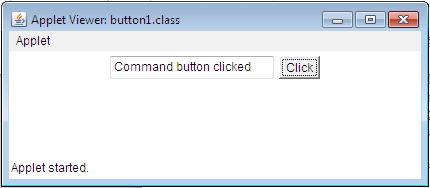
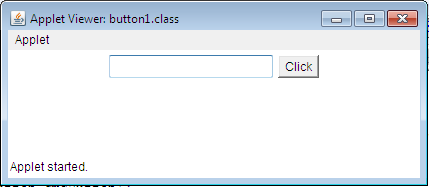
{

txtText1.setText(strMessage);

}

}

}



**Using Check Boxes and Option Buttons.**

Check Boxes – Users check as many check boxes as they want.

Option buttons – users can only select one option at a time.

1. Check Boxes.

Design and develop a java applet that when the user clicks one of the three boxes, it will indicate in the text field on which check box the user had clicked. Also, it will display a message if the check box unchecked.

import java.applet.\*;

import java.awt.\*;

import java.awt.event.\*;

public class checksample extends Applet implements ItemListener {

Checkbox one= new Checkbox("1");

Checkbox two= new Checkbox("2");

Checkbox three= new Checkbox("3");

Label lab1 = new Label("What do you like?");

TextField txt1 = new TextField(" ",20);

public void init() {

one.setState(true);

txt1.setText("1 is checked for you!");

add(lab1);

add(one);

add(two);

add(three);

add(txt1);

one.addItemListener(this);

two.addItemListener(this);

three.addItemListener(this);

}

public void itemStateChanged(ItemEvent change) {

boolean state;

if (change.getItemSelectable()==one) {

state=one.getState();

if (state==true)

txt1.setText("1 is checked!");

else

txt1.setText(" 2 is unchecked!");

}

if (change.getItemSelectable()==two) {

state=two.getState();

if (state==true)

txt1.setText("2 is checked!");

else

txt1.setText("2 is unchecked!");

}

if (change.getItemSelectable()==three) {

state=three.getState();

if (state==true)

txt1.setText("3 is checked!");

else

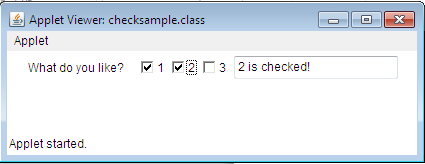
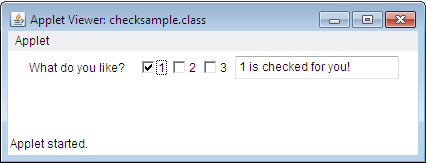
txt1.setText("3 is unchecked!");

}

}

}

**OUTPUT:**

****

1. Option Buttons.

Design and develop a Java applet that when the user clicks one of the three option buttons, it will indicate in the text field on which the option button the user had clicked.

import java.applet.Applet;

import java.awt.\*;

import java.awt.event.\*;

public class optionbutton extends Applet implements ItemListener

{

CheckboxGroup optButtonGroup1;

Checkbox optButton1, optButton2, optButton3;

TextField txtText1;

public void init()

{

optButtonGroup1 = new CheckboxGroup();

optButton1 = new Checkbox("option 1", false, optButtonGroup1);

add(optButton1);

optButton1.addItemListener(this);

optButton2 = new Checkbox("option 2", false, optButtonGroup1);

add(optButton2);

optButton2.addItemListener(this);

optButton3 = new Checkbox("option 3", false, optButtonGroup1);

add(optButton3);

optButton3.addItemListener(this);

txtText1 = new TextField(20);

add(txtText1);

}

public void itemStateChanged(ItemEvent objEvent)

{

if (objEvent.getItemSelectable()==optButton1)

{

txtText1.setText("Option button 1 selected!");

}

if (objEvent.getItemSelectable()==optButton2)

{

txtText1.setText("Option button 2 selected!");

}

if (objEvent.getItemSelectable()==optButton3)

{

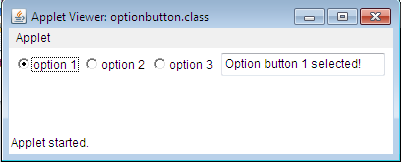
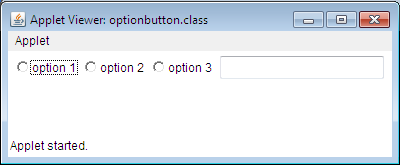
txtText1.setText("Option button 3 selected!");

}

}

}

**OUTPUT:**

****

**Using Controls with Input/Output Functions.**

The ParseInt() and valueOf() Methods

In calculating and displaying data in applet, ParseInt() and valuOf() are used. These two methods simply convert our data from numeric to string data or vice versa. The ParseInt() methods converts the displayed data into numeric, while the valuOf() method converts the displayed data into string data. The ParseInt() method is under the java integer class that takes text string and returns a numeric integer value. For example, we can get the text string in the fiels txtText1 this way:

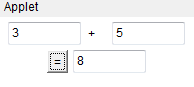
txtText1.getText()

And we convert that string to a numeric integer value this way:

Integer.parseInt(txtText1())

Example 1: Design and develop a Java applet that calculates the sum of two input numbers.

Sample Output:



import java.applet.Applet;

import java.awt.\*;

import java.awt.event.\*;

public class sum1 extends Applet implements ActionListener

{

TextField txtNum1, txtNum2, txtSum;

Label lblPlus;

Button cmdEqual;

public void init(){

txtNum1= new TextField(7);

add(txtNum1);

lblPlus= new Label("+");

add(lblPlus);

txtNum2=new TextField(7);

add(txtNum2);

cmdEqual= new Button("=");

add(cmdEqual);

cmdEqual.addActionListener(this);

txtSum= new TextField(7);

add(txtSum);

}

public void actionPerformed(ActionEvent objEvent)

{

int varSum=0;

if(objEvent.getSource()==cmdEqual){

varSum=Integer.parseInt(txtNum1.getText()) + Integer.parseInt(txtNum2.getText());

txtSum.setText(String.valueOf(varSum));

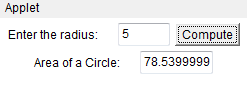
}

}

}

Example No. 2 Design a java applet that computes the area of a circle. Use the formula: A=3.1416 \* r \* r.

Sample output:



import java.applet.Applet;

import java.awt.\*;

import java.awt.event.\*;

public class area1 extends Applet implements ActionListener

{

TextField txtRadius, txtArea;

Label lblRadius, lblArea;

Button cmdCompute;

public void init()

{

lblRadius=new Label("Enter the radius: ");

add(lblRadius);

txtRadius= new TextField(4);

add(txtRadius);

cmdCompute= new Button("Compute");

add(cmdCompute);

cmdCompute.addActionListener(this);

lblArea= new Label("Area of a Circle: ");

add(lblArea);

txtArea= new TextField(7);

add(txtArea);

}

public void actionPerformed(ActionEvent objEvent) {

int varRadius=0;

double varArea=0;

if(objEvent.getSource()==cmdCompute){

varRadius=Integer.parseInt(txtRadius.getText());

varArea= 3.1416 \* (varRadius\*varRadius);

txtArea.setText(String.valueOf(varArea));

}

}

}

Panel – Arranging of controls. A panel is a rectangular region that contains a group of control such as check boxes or option button.

Example. Design and develop a Java applet that when the user chooses the Hamburger option, the ingredients to be marked with the check are: Beef and Spices; its corresponding price is 49.25(to be displayed at the text field). If the user chooses the Cheeseburger option, the ingredients to be checked are: Beef, Spices, and Cheese; its corresponding price is 69.50. Now if the user chooses the Baconburger option, all the ingredients are marked with check and its corresponding price is 89.75.

Solution:

html>

<applet code = "burger.class"

width=220

height=220>

</applet>

</html>

import java.applet.Applet;

import java.awt.\*;

import java.awt.event.\*;

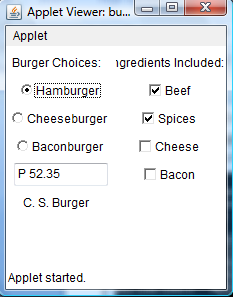
public class burger extends Applet implements ItemListener

{

Menu Panel1;

Ingredients Panel2;

public void init()

Output: 

{

setLayout(new GridLayout(1,2));

Panel1= new Menu();

Panel2= new Ingredients();

add(Panel1);

Panel1.Hamburger.addItemListener(this);

Panel1.Cheeseburger.addItemListener(this);

Panel1.Baconburger.addItemListener(this);

add(Panel2);

}

public void itemStateChanged(ItemEvent objEvent) {

if(objEvent.getItemSelectable()==Panel1.Hamburger) {

Panel2.Ingredient1.setState(true);

Panel2.Ingredient2.setState(true);

Panel2.Ingredient3.setState(false);

Panel2.Ingredient4.setState(false);

Panel1.txtPrice.setText("P 52.35");

}

if(objEvent.getItemSelectable()==Panel1.Cheeseburger) {

Panel2.Ingredient1.setState(true);

Panel2.Ingredient2.setState(true);

Panel2.Ingredient3.setState(true);

Panel2.Ingredient4.setState(false);

Panel1.txtPrice.setText("P 69.00");

}

if(objEvent.getItemSelectable()==Panel1.Baconburger) {

Panel2.Ingredient1.setState(true);

Panel2.Ingredient2.setState(true);

Panel2.Ingredient3.setState(true);

Panel2.Ingredient4.setState(true);

Panel1.txtPrice.setText("P 93.50");

}

}

}

class Menu extends Panel {

CheckboxGroup CGroup;

Checkbox Hamburger,Cheeseburger,Baconburger;

TextField txtPrice;

Label lblMenu;

Label lblTitle;

Menu() {

CGroup=new CheckboxGroup();

lblMenu=new Label("Burger Choices:");

add(lblMenu);

add(Hamburger=new Checkbox("Hamburger", CGroup,false));

add(Cheeseburger=new Checkbox("Cheeseburger", CGroup,false));

add(Baconburger=new Checkbox("Baconburger", CGroup,false));

txtPrice= new TextField(10);

add(txtPrice);

Label lblTitle= new Label ("C. S. Burger");

add(lblTitle);

}

}

class Ingredients extends Panel {

Checkbox Ingredient1, Ingredient2, Ingredient3, Ingredient4;

Label lblIngredient;

Ingredients() {

lblIngredient= new Label("Ingredients Included:");

add(lblIngredient);

add(Ingredient1= new Checkbox("Beef"));

add(Ingredient2= new Checkbox("Spices"));

add(Ingredient3= new Checkbox("Cheese"));

add(Ingredient4= new Checkbox("Bacon"));

}

}

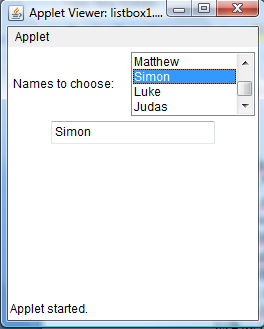
**Scrolling List (List Box) Controls**

The Scrolling List(List box) control is an ideal way of presenting a list of data to user, It is an effective way to present a large number of data to the user in a limited amount of space.

Example1. Design and develop a Java applet that demonstrates how to preload a collection of items to scrolling list. When the user chooses an item at the scrolling list by double-clicking it, that particular item will be displayed at the text field.

Solution:

import java.applet.Applet;



import java.awt.\*;

import java.awt.event.\*;

public class listbox1 extends Applet implements ActionListener{

Label lblMessage = new Label("Names to choose:");

TextField txtText1 = new TextField(20);

List lstBox1;

public void init(){

add(lblMessage);

lstBox1 = new List(4,false);

lstBox1.add("Peter");

lstBox1.add("Paul");

lstBox1.add("John");

lstBox1.add("James");

lstBox1.add("Matthew");

lstBox1.add("Simon");

lstBox1.add("Luke");

lstBox1.add("Judas");

add(lstBox1);

lstBox1.addActionListener(this);

add(txtText1);

}

public void actionPerformed(ActionEvent objEvent) {

if (objEvent.getSource()==lstBox1)

txtText1.setText(((List)objEvent.getSource()).getSelectedItem());

}

}

Example 2. Java applet that demonstrates how to preload a collection of items to a Scrolling list and with capability adding more items into it.

Solution:

import java.applet.Applet;

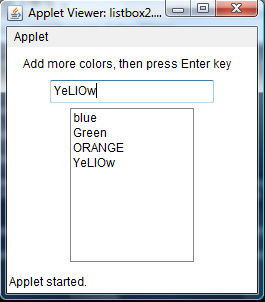
import java.awt.\*;

import java.awt.event.\*;

public class listbox2 extends Applet implements ActionListener{

Label lblMessage = new Label("Add more colors, then press Enter key");

TextField txtText1 = new TextField(20);



List lstBox1;

String strColor=" ";

public void init(){

add(lblMessage);

add(txtText1);

txtText1.addActionListener(this);

lstBox1 = new List(10,false);

lstBox1.add("blue");

lstBox1.add("Green");

add(lstBox1);

}

public void actionPerformed(ActionEvent objEvent) {

strColor=txtText1.getText();

lstBox1.add(strColor);

repaint();

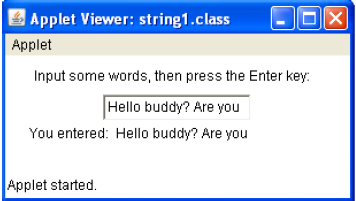
}

}

**Self-Review Exercises**

**Laboratory Exercise no. 1**

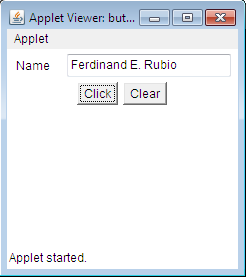
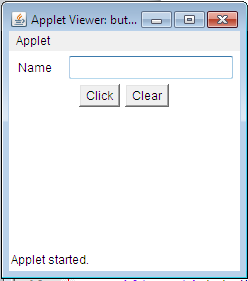
Design and develop a Java applet that allows you to both enter text string into a text box and display the text string on the applet. The text string will be displayed only after the Enter key is pressed. Follow the design specification below:



**Laboratory Exercise no. 2**

Create a java applet that will display your name on a textbox once a command button (click )clicked with Label “Name” and a clear button to clear the textbox.

Sample Output:



**Laboratory Exercise no. 3**

Design and develop a java applet using the following:

1(one) label

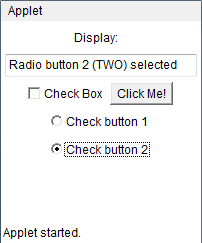
1( one) command button

1( one) check box

2( two) option buttons and

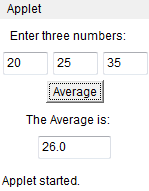
1(one) Text box.

Follow the given output format:



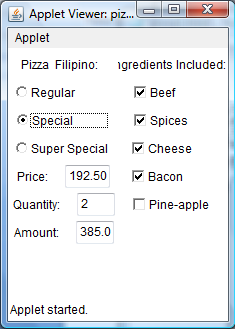
**Laboratory Exercise no. 4**

Design and develop a java applet that computes the average of three input numbers. Then display the result. Sample output:



**Laboratory Exercise no. 5**

Design and develop a Java applet that when the user chooses the Regular pizza option, the ingredients to be marked with a check are: Beef, Spices, and Cheese; its corresponding price is 175.25 (to be displayed at the text field). If the user inputs 2 in the Quantity text field, the amount to be displayed should be 350.50 (2 times 175.25). If the quantity is 3 then the amount to be displayed is 525.75, and so on. When the user chooses the Special pizza option, the ingredients to be checked are: Beef, Spices, Cheese and Bacon; its corresponding price is 192.50. When the user chooses the Super Special pizza option, all ingredients should be marked with a check; its corresponding price is 327.50.



Follow the design: