**Ex. No. 1PROGRAMS TO DEMONSTRATE THE USE OF VARIOUS**

**COMPONENTS**

**Text Field**

Aim

To write a java program to demonstrate the Text Field Control

Algorithm

Step 1: Import all necessary packages and classes

Step 2: Define a class that extends frame and implements action listener

Step 3: Declare two text fields and a text area then declare labels for them

Step 4: Declare a button

Step 5: In the class constructor do the following:

i) Create the text fields, text area, and the labels

ii) Set the layout of the frame to border layout

iii) Add the controls to the frame

iv) Add action listener to the button

v) Add window listener to the frame which listens to a window adapter object

Step 6: In the actionPerformed() method, add the text of all the text fields to the text area

Step 7: In the main method () create a frame, set its size and the make it visible

Step 8: Define a Window Adapter class and handle the window closing event

Program:

import java.awt.\*;

import java.awt.event.\*;

class TextFrame extends Frame implements ActionListener

{

TextField txtName;

TextField txtRollNo;

TextArea areaDisplay;

Label lblName,lblRollNo,lblDisplay;

Button btnSubmit;

TextFrame()

{

txtName = new TextField(20);

txtRollNo = new TextField(6);

lblRollNo = new Label("Enter your Roll Number Here");

areaDisplay = new TextArea(5,25);

lblName = new Label("Enter Your Name Here");

btnSubmit = new Button("Submit");

lblDisplay = new Label("You Have Entered The Following Details");

setLayout(new FlowLayout());

add(lblName);

add(txtName);

add(lblRollNo);

add(txtRollNo);

add(btnSubmit);

add(lblDisplay);

add(areaDisplay);

btnSubmit.addActionListener(this);

addWindowListener( new MyWindowAdapter(this));

}

public void actionPerformed(ActionEvent e)

{

areaDisplay.setText("Name: "+txtName.getText()+"\n Roll Number:

"+txtRollNo.getText());

}

public static void main(String[] a)

{

TextFrame frame = new TextFrame();

frame.setSize(400,400);

frame.setVisible(true);

}

}

class MyWindowAdapter extends WindowAdapter

{

TextFrame frame;

MyWindowAdapter(TextFrame frame)

{

this.frame=frame;

}

public void windowClosing(WindowEvent evt)

{

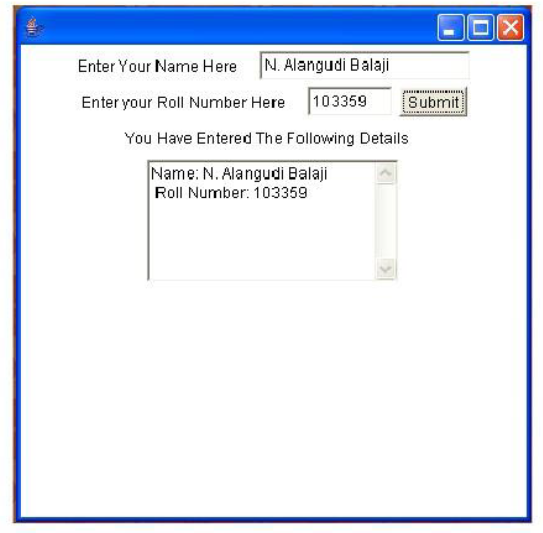
frame.dispose();

System.exit(0);

}

}

Output



**Button**

Aim

To write a java program to demonstrate AWT Button

Algorithm

Step 1: Write a html applet tag with code property set to the class name and comment it

Step 2: Import necessary packages and classes

Step 3: Define a class that extends applet and implements action listener

Step 4: Declare a string msg and three buttons, yes, no, and maybe

Step 5: In the init() method, do the following:

i) Create the buttons

ii) Add the buttons to the applet

iii) Add action listener to the buttons

Step 6: In the actionPerformed() method, do the following:

i) Get the action command of the event happened

ii) If it is equal to yes, set the msg string that the user pressed yes button

iii) Repeat i) and ii) for the other buttons with appropriate msg

iv) Repaint the applet

Step 7: In the paint() method, display the msg string

Program

/\* <applet code=ButtonApplet height=300 width=400> </applet>\*/

import java.awt.\*;

import java.awt.event.\*;

import java.applet.\*;

public class ButtonApplet extends Applet implements ActionListener

{

String msg = "";

Button yes, no, maybe;

public void init()

{

yes = new Button("Yes");

no = new Button("No");

maybe = new Button("Undecided");

add(yes);

add(no);

add(maybe);

yes.addActionListener(this);

no.addActionListener(this);

maybe.addActionListener(this);

}

public void actionPerformed(ActionEvent e)

{

String str = e.getActionCommand();

if(str.equals("Yes"))

msg = "You pressed Yes";

else if(str.equals("No"))

msg = "You pressed No";

else

msg = "You pressed Undecided";

repaint();

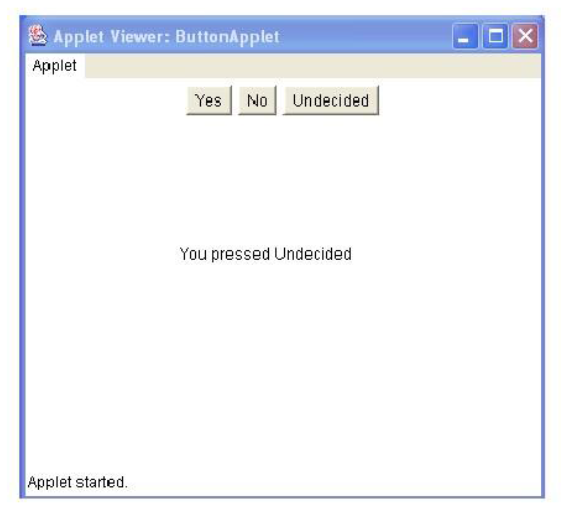
}

public void paint(Graphics g)

{

g.drawString(msg,6,100); } }

Output



Scroll Bar

Aim

To write a java program to demonstrate the AWT Scrollbar

Algorithm

Step 1: Write a html applet tag with code property set to the class name and comment it

Step 2: Import necessary packages and classes

Step 3: Define a class that extends applet and implements adjustment listener

Step 4: Declare three scroll bars red, green, and blue

Step 5: Declare a panel to hold scroll bars and three labels for them

Step 6: In the init() method, do the following:

i) Create the scroll bars, set their unit increment and add adjustment listern to them

ii) Create the panel and set its layout to grid layout of 3 rows and two columns

iii) Create the labels and add them to the panel and then add the scroll bars

iv) Add the scroll panel to the applet

Step 7: In the adjustmentValueChanged() method, repaint the applet

Step 8: In the paint() method, do the following:

i) Create a color from the values of red, green and blue scroll bars

ii) Draw a rectangle and fill it with the created color

Program

/\* <applet code=ScrollApplet height = 400 width = 600>

</applet> \*/

import java.awt.\*;

import java.awt.event.\*;

import java.applet.\*;

public class ScrollApplet extends Applet implements AdjustmentListener{

Scrollbar red;

Scrollbar green;

Scrollbar blue;

Panel scrollpanel;

Label lred,lgreen,lblue;

public void init(){

//red.setMinimumSize(100);

red = new Scrollbar(Scrollbar.HORIZONTAL,10,1,0,255);

red.setUnitIncrement(10);

red.addAdjustmentListener(this);

green = new Scrollbar(Scrollbar.HORIZONTAL,40,1,0,255);

green.setUnitIncrement(20);

green.addAdjustmentListener(this);

blue = new Scrollbar(Scrollbar.HORIZONTAL,50,1,0,255);

blue.setUnitIncrement(30);

blue.addAdjustmentListener(this);

lred = new Label("RED");

lgreen = new Label("GREEN");

lblue = new Label("BLUE");

scrollpanel = new Panel();

scrollpanel.setLayout(new GridLayout(3,2));

scrollpanel.add(lred);

scrollpanel.add(red);

scrollpanel.add(lgreen);

scrollpanel.add(green);

scrollpanel.add(lblue);

scrollpanel.add(blue);

add(scrollpanel);

}

public void adjustmentValueChanged(AdjustmentEvent ae){

repaint();

}

public void paint(Graphics g){

Color mycolor = new Color(red.getValue(),green.getValue(),blue.getValue());

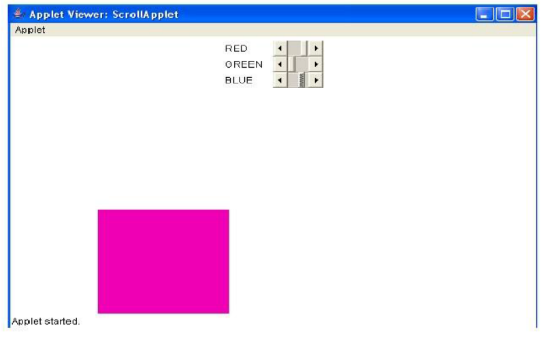
g.setColor(mycolor);

g.fillRect(100,250,150,300);

}

}

Output



**Choice**

Aim

To write a java program to demonstrate the AWT Choice Control

Algorithm

Step 1: Write a html applet tag with code property set to the class name and comment it

Step 2: Import necessary packages and classes

Step 3: Define a class that extends applet and implements item listener

Step 4: Declare a choice fonts

Step 5: In the init() method, do the following:

i) Declare the choice control

ii) Populate the control with various font names

iii) Add item listener to the choice

iv) add the fonts choice to the applet

Step 6: In the itemStateChanged method repaint the applet

Step 7: In the paint() method do the following:

i) Get the selected choice item

ii) Create a font of the selected item value

iii) Set the font of the graphics of the applet to the created font

iv) Display any text on the applet

Program

/\* <applet code = ChoiceApplet height = 400 width = 600>

</applet> \*/

import java.awt.\*;

import java.awt.event.\*;

import java.applet.\*;

public class ChoiceApplet extends Applet implements ItemListener{

Choice fonts;

public void init(){

fonts = new Choice();

fonts.add("Arial");

fonts.add("Arial Black");

fonts.add("Book Antiqua");

fonts.add("Bookman Old Style");

fonts.add("Garamond");

fonts.add("Symbol");

fonts.addItemListener(this);

add(fonts);

}

public void itemStateChanged(ItemEvent ie){

repaint();

}

public void paint(Graphics g){

String name = fonts.getSelectedItem();

Font myFont = new Font(name,Font.PLAIN,30);

g.setFont(myFont);

g.drawString("The New Font Selected Is "+name, 100,300);

}

}

Output



**List**

Aim

To write a java program to demonstrate the AWT List Control

Algorithm

Step 1: Write a html applet tag with code set to class name and comment the tag

Step 2: Import all necessary packages and classes

Step 3: Define a class that extends applet and implements action listener

Step 4: Declare two lists fruits and drinks

Step 5: Declare panel, label and string for each list

Step 6: In the init() method, do the following:

i) Create fruits list with multiple select option set to false and add action listener to it

ii) populate the list with various fruit names

iii) Create label for the fruits prompt to select a fruit

iv) Create a panel and set the layout of the panel to flow layout then add the list to the

panel

v) Repeat the steps i) to iv) for the drinks list except that list is set with multiple

select set to true and populated with various beverage names

vi) Set the layout of the applet to grid layout with 3 rows and 1 column

vii) Add the fruit panel and drink panel to the applet

Step 7: In the actionPerformed() method, do the following:

i) Get the names of the selected drinks and fruits and store them in two Strings

ii) repaint the applet

Step 8: In the paint() method, do the following:

i) Display the strings set in actionPerformed() method

Program

/\*\* <applet code=ListApplet width = 600 height = 400>

</applet>\*/

import java.awt.\*;

import java.awt.event.\*;

import java.applet.\*;

public class ListApplet extends Applet implements ActionListener{

List fruits;

List drinks;

Panel fruitpanel;

Panel drinkpanel;

Label fruitlabel;

Label drinklabel;

String msg,fruit;

public void init(){

fruits = new List(4,false);

fruits.addActionListener(this);

fruits.add("Apple");

fruits.add("Graphes");

fruits.add("Mango");

fruits.add("Banana");

fruits.add("Orange");

fruits.select(3);

fruitlabel = new Label("Select Your Favourite Fruit");

fruitpanel = new Panel();

fruitpanel.setLayout(new FlowLayout(FlowLayout.CENTER));

fruitpanel.add(fruitlabel);

fruitpanel.add(fruits);

drinks = new List(6,true);

drinks.addActionListener(this);

drinks.add("Pepsi");

drinks.add("Coke");

drinks.add("Sprite");

drinks.add("Maza");

drinks.add("Fruity");

drinks.add("7up");

drinks.add("Mirinda");

drinks.add("Fanta");

drinklabel = new Label("Select as Many drinks as you want");

drinkpanel = new Panel();

drinkpanel.setLayout(new FlowLayout(FlowLayout.CENTER));

drinkpanel.add(drinklabel);

drinkpanel.add(drinks);

setLayout(new GridLayout(3,1));

add(fruitpanel);

add(drinkpanel);

}

public void actionPerformed(ActionEvent ae){

msg = "";

int[] drinkarr = drinks.getSelectedIndexes();

for(int i = 0;i<drinkarr.length;i++)

msg+=drinks.getItem(drinkarr[i])+",";

msg+=".";

repaint();

}

public void paint(Graphics g){

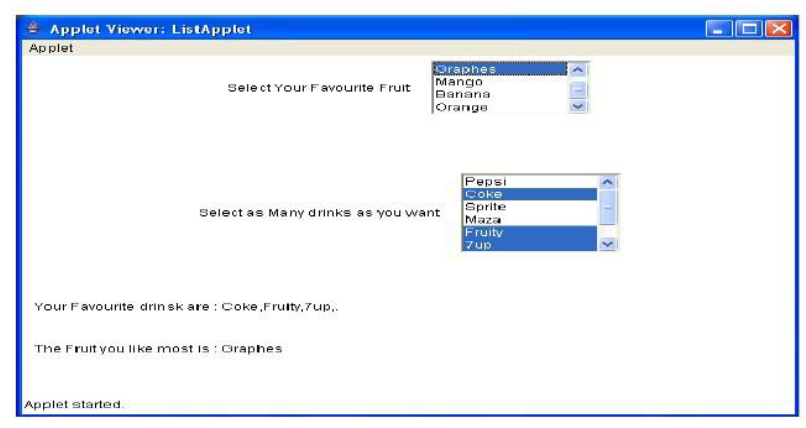
String fruit = fruits.getSelectedItem();

g.drawString("The Fruit you like most is : " + fruit,10,350);

g.drawString("Your Favourite drinsk are : " + msg,10,300);

} }

Output



**Checkbox**

Aim

To write a java program to demonstrate the use of AWT Checkbox Control

Algorithm

Step 1: Write a html applet tag with code set class name and comment the tag

Step 2: Import all necessary packages and classes

Step 3: Define a class that extends applet and implements item listener

Step 4: Declare checkboxes be,me,mtech,bsc,msc,music, sport,read,stamp

step 5: Declare a checkbox group and two labels degree and hobby

Step 6: Declare a string msg

Step 7: In the init() method, do the following:

i) Create a checkboxgroup and create checkboxes be,me,mtech,bsc and msc belongs

to the checkbox group and add item listener to checkboxes

ii) add the checkboxes to the applet

iii) Create checkboxes music, sport, read and stamp then item listener to them and add

them to the applet

Step 8: In the itemStateChanged() method repaint the applet

Step 9: In the paint() method, do the following:

i) Get the selected checkbox label from the checkbox group and display it

ii) check all other checkboxes whether they are set selected or not. If selected display

them

Program

/\* <applet code=CheckBoxApplet width=300 height=400></applet>\*/

import java.awt.\*;

import java.awt.event.\*;

import java.applet.\*;

public class CheckBoxApplet extends Applet implements ItemListener

{

Checkbox be,me,mtech,bsc,msc;

Checkbox music,sport,read,stamp;

CheckboxGroup cbg;

Label lbldegree,lblhobby;

String msg;

public void init()

{

lbldegree = new Label("Degree : ");

lblhobby = new Label("Hobby :");

cbg = new CheckboxGroup();

be = new Checkbox("B.E",cbg,true);

me = new Checkbox("M.E",cbg,false);

mtech = new Checkbox("M.Tech",cbg,false);

bsc = new Checkbox("B.Sc.",cbg,false);

msc = new Checkbox("M.Sc.",cbg,false);

be.addItemListener(this);

me.addItemListener(this);

mtech.addItemListener(this);

bsc.addItemListener(this);

msc.addItemListener(this);

add(lbldegree);

add(be);

add(me);

add(mtech);

add(bsc);

add(msc);

music = new Checkbox("Listening Music");

sport = new Checkbox("Playing Sports");

read = new Checkbox("Reading Novels");

stamp = new Checkbox("Collecting Stamps");

music.addItemListener(this);

sport.addItemListener(this);

read.addItemListener(this);

stamp.addItemListener(this);

add(lblhobby);

add(music);

add(sport);

add(read);

add(stamp);

}

public void itemStateChanged(ItemEvent ie)

{

repaint();

}

public void paint(Graphics g)

{

msg = " Summary : You are pursuing your ";

msg+= ((Checkbox)cbg.getSelectedCheckbox()).getLabel();

msg+= " degree";

g.drawString(msg,200,200);

msg="Your hobbies are: ";

if(music.getState()==true)

msg+=music.getLabel()+" " ;

if(sport.getState()==true)

msg+=sport.getLabel()+" ";

if(read.getState()==true)

msg+=read.getLabel()+" ";

if(stamp.getState()==true)

msg+=stamp.getLabel()+" ";

g.drawString(msg,200,300);

}

}

Output



**Ex. No. 2**

**PROGRAMS TO DEMONSTRATE THE USE OF VARIOUS LAYOUTS**

To write a java program to demonstrate the Flow Layout

Algorithm

Step 1: Write a html applet tag with code set to class name and comment it

Step 2: Import all necessary packages and classes

Step 3: Define a class that extends Applet

Step 4: Declare an array of buttons and three panels

Step 5: In the init() method of the applet do the following:

i) Set the layout of the applet to Grid layout that can hold three columns

ii) Create the array of buttons

iii) Add a panel, set the layout to Flow layout to add buttons from the center

iv) Add the buttons to the panel

v) Similarly create two more panels and set layout to flow layout to add the

buttons from the left and right of the flow layout

vi) Add the buttons in to these panels

Program

/\* <applet code=FlowLayoutApplet height=400 width=600> </applet>\*/

import java.awt.\*;

import java.awt.event.\*;

import java.applet.\*;

public class FlowLayoutApplet extends Applet

{

Button[] b;

Panel p1,p2,p3;

public void init()

{

setLayout(new GridLayout(3,1));

p1 = new Panel();

p1.setLayout(new FlowLayout(FlowLayout.RIGHT));

add(p1);

p2 = new Panel();

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

add(p2);

p3 = new Panel();

p3.setLayout(new FlowLayout(FlowLayout.CENTER));

add(p3);

b = new Button[15];

for(int i=0;i<15;i++)

{

b[i] = new Button();

b[i].setLabel("Button"+(i+1));

p1.add(b[i]);

}

for(int i=0;i<15;i++)

{

b[i] = new Button();

b[i].setLabel("Button"+(i+1));

p2.add(b[i]);

}

for(int i=0;i<15;i++)

{

b[i] = new Button();

b[i].setLabel("Button"+(i+1));

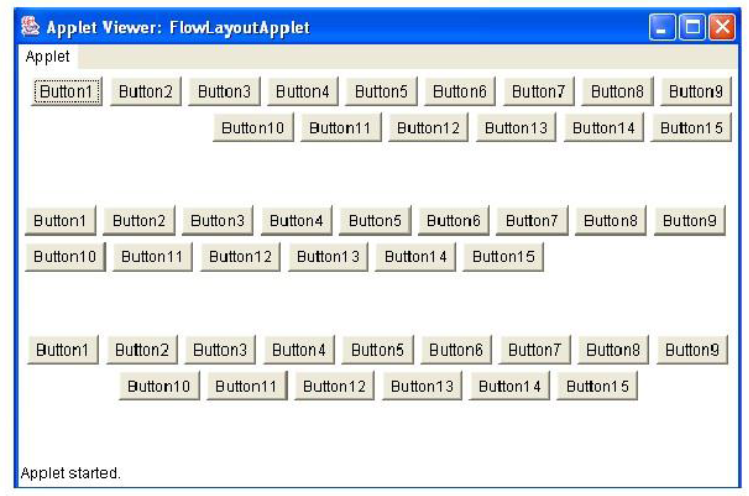
p3.add(b[i]);

}

}

}

Output



**Border Layout**

Aim

To write a java program demonstrating the Border Layout

Algorithm

Step 1. Include the applet tag with code set to the class name of the applet as comment

Step 2. Import all the required packages and classes

Step 3. Define class which extends Applet and implements ActionListener

Step 4. Declare buttons green, red, blue, red and font

Step 5. Declare msg of type String and initialize to "sethu Institute"

Step 6. Declare a Textarea area

Step 7. In the init method of the applet, do the following

i) Set the layout to border layout

ii) Create area object and set its text to msg and add it to the center of the borderlayout

iii) Create green button and add it to the north of borderlayout. Similarly, create red,

blue,font buttons and add them to south,east, west of border layout.

iv) Add action listener to all the buttons

Step 8: In the actionperformed method do the following

i) Get the action command of a button pressed in a string str

ii) If str is equal to any one of the color button's text then set the foreground to that color.

iii) If str is equal to font then create a new font of type Garmond and set the area's font to

the newly created font

iv) Repaint the applet

Step 9: write the paint method with no body (empty paint method)

Program

import java.applet.\*;

import java.awt.\*;

import java.awt.event.\*;

/\*<applet code=BorderLayoutApplet width=300 height=100>

</applet>\*/

public class BorderLayoutApplet extends Applet implements ActionListener{

Button green=new Button("green");

Button red=new Button("red");

Button blue=new Button("blue");

Button font=new Button("font");

String msg="Sethu";

TextArea area;

public void init()

{

setLayout(new BorderLayout());

area = new TextArea();

area.setText(msg);

add(green,BorderLayout.NORTH);

add(red,BorderLayout.SOUTH);

add(blue,BorderLayout.EAST);

add(font,BorderLayout.WEST);

add(area,BorderLayout.CENTER);

green.addActionListener(this);

red.addActionListener(this);

blue.addActionListener(this);

font.addActionListener(this);

}

public void actionPerformed (ActionEvent a)

{

String str= a.getActionCommand();

if(str.equals("green"))

{

area.setForeground(Color.green);

}

else if(str.equals("red"))

{

area.setForeground(Color.red);

}

else if(str.equals("blue"))

{

area.setForeground(Color.blue);

}

else

{

Font myfont = new Font("Garamond",Font.BOLD,25);

area.setFont(myfont);

area.setForeground(Color.cyan);

}

repaint();

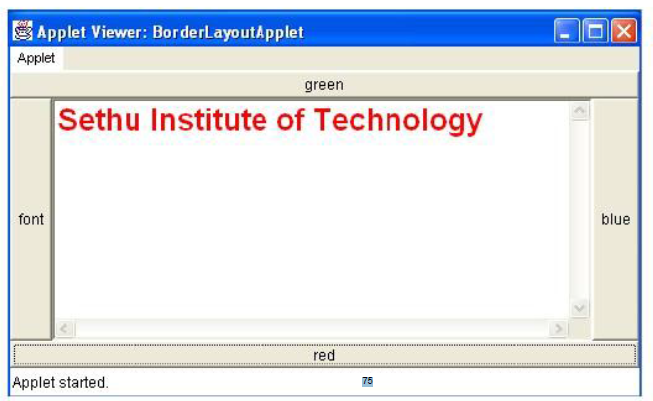
}

public void paint(Graphics g)

{}

}

Output



**Grid Layout**

Aim

To write a java program to demonstrate the Grid Layout

Algorithm

Step 1: Write an applet tag with code property set to class name and comment it

Step 2: import all the necessary packages

Step 3: Define a class which extends Applet

Step 4: Declare any number of buttons

Step 5: In the init() method do the following:

i) Create a panel

ii) create the buttons

iii) set the layout of the panel to Gridlayout

iv) add the buttons to the panel

v) set the layout of the applet to borderlayout

vi) add the panel to the center of the border layout

Step 6: Write empty paint() method

Program

/\*<applet code=GridLayoutApplet width=100 height=100>

</applet>\*/

import java.applet.Applet;

import java.awt.\*;

public class GridLayoutApplet extends Applet

{

private Button one,two,three,four,five,six,seven,eight,nine,ten;

public void init()

{

Panel panel=new Panel();

one=new Button("1");

two=new Button("2");

three=new Button("3");

four=new Button("4");

five=new Button("5");

six=new Button("6");

seven=new Button("7");

eight=new Button("8");

nine=new Button("9");

ten=new Button("10");

panel.setLayout(new GridLayout(3,0,10,10));

panel.add(one);

panel.add(two);

panel.add(three);

panel.add(four);

panel.add(five);

panel.add(six);

panel.add(seven);

panel.add(eight);

panel.add(nine);

panel.add(ten);

setLayout(new BorderLayout());

add(panel,BorderLayout.CENTER);

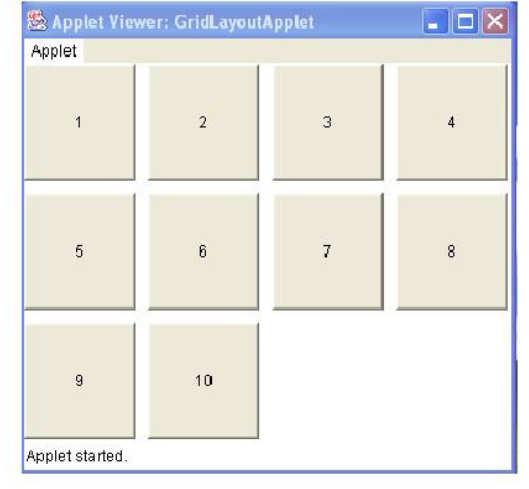
}

public void paint(Graphics g)

{}

}

Output



**GridBag Layout**

Aim

To write a java program to demonstrate the Gridbag Layout

Algorithm

Step 1: Write a html applet tag with code property set to class name and comment the tag

Step 2: Import all necessary packages

Step 3: Define a class which extends applet class

Step 4: Declare all the necessary awt controls to be included in the purchase form

Step 5: In the init() method, do the following:

i) Create a gridbag layout and set the applet's layout to gridbag layout

ii) Create a gridbag constraint

iii) Create a awt control

iv) Set the gridbag constraints for this control

v) Add the control to the applet

vi) Similarly do the steps from iii) to v) for all other control

Program

/\*\* <applet code = PurchaseApplet width = 600 height = 400>

</applet> \*/

import java.awt.\*;

import java.applet.Applet;

public class PurchaseApplet extends Applet

{

Panel ButtonPanel;

Label title,name,address,payment,phone,city,state;

TextField namefield,addressfield,cityfield,statefield;

Choice paymentChoice;

Button paymentButton,cancelButton;

public void init()

{

ButtonPanel = new Panel();

title=new Label("Purchase Form");

name=new Label("Name");

address=new Label("Address");

payment=new Label("payment");

phone=new Label("phone");

city=new Label("city");

state=new Label("state");

namefield=new TextField(25);

addressfield=new TextField(30);

cityfield=new TextField(20);

statefield=new TextField(20);

paymentChoice=new Choice();

paymentButton=new Button("purchase");

cancelButton=new Button("cancel");

GridBagLayout gbl=new GridBagLayout();

GridBagConstraints gbc=new GridBagConstraints();

setLayout(gbl);

paymentChoice.add("visa");

paymentChoice.add("master card");

paymentChoice.add("cod");

title.setFont(new Font("Times-Roman",Font.BOLD+Font.ITALIC,16));

gbc.anchor=GridBagConstraints.NORTH;

gbc.gridwidth=GridBagConstraints.REMAINDER;

add(title,gbc);

gbc.fill=GridBagConstraints.HORIZONTAL;

gbc.insets=new Insets(10,0,10,0);

gbc.gridwidth = 1;

add(name,gbc);

gbc.gridwidth=GridBagConstraints.REMAINDER;

add(namefield,gbc);

gbc.insets=new Insets(0,0,10,0);

gbc.gridwidth=1;

add(address,gbc);

gbc.gridwidth=GridBagConstraints.REMAINDER;

add(addressfield,gbc);

gbc.gridwidth=1;

add(city,gbc);

add(cityfield,gbc);

add(state,gbc);

gbc.gridwidth=GridBagConstraints.REMAINDER;

add(statefield,gbc);

gbc.fill = GridBagConstraints.NONE;

gbc.gridwidth=1;

add(payment,gbc);

gbc.gridwidth = GridBagConstraints.REMAINDER;

add(paymentChoice,gbc);

ButtonPanel.add(paymentButton);

ButtonPanel.add(cancelButton);

gbc.anchor=GridBagConstraints.SOUTH;

gbc.gridwidth = 4;

add(ButtonPanel,gbc);

}

}

Output



**Card Layout**

Aim

To write a java program to demonstrate the Card Layout

Algorithm

Step 1: Write a html applet tag with code set class name and comment the tag

Step 2: Import all necessary packages and classes

Step 3: Define a class that extends applet and implements action listener

Step 4: Declare buttons tiny, large, medium, and small

Step 5: Create panels for card layout and for cards of buttons

Step 6: In the init() method, do the following:

i) Create the card layout

ii) Create the cardpanel and set its layout to card layout

iii) Create other panels and set their layout to border layout

iv) Create buttons and change their fonts to vriuos font face and sizes according to the

button name

v) Add action listener to the buttons and each button to one panel of appropriate name

vi) Add the panels to the card panel

vii) Set the layout of the applet to border layout and add card panel to the applet

Step 7: In the actionPerformed() method, move to the next card in the card panel

Program

\*<applet code=cardLayoutApplet width=600 height=400>

</applet>\*/

import java.awt.event.\*;

import java.awt.\*;

import java.applet.Applet;

import java.awt.event.ActionListener;

public class cardLayoutApplet extends Applet implements ActionListener

{

private Button tiny,large,medium,small;

private Panel cardpanel=new Panel();

private Panel tinypanel=new Panel();

private Panel smallpanel=new Panel();

private Panel medpanel=new Panel();

private Panel largepanel=new Panel();

private CardLayout card=new CardLayout(10,5);

public void init()

{

cardpanel.setLayout(card);

tinypanel.setLayout(new BorderLayout());

smallpanel.setLayout(new BorderLayout());

medpanel.setLayout(new BorderLayout());

largepanel.setLayout(new BorderLayout());

tiny=new Button("Tiny CardLayout");

small=new Button("Small CardLayout");

medium=new Button("Medium CardLayout");

large=new Button("Large CardLayout");

tiny.setFont(new Font("Helvetica",Font.BOLD,20));

small.setFont(new Font("Times-Roman",Font.BOLD,12));

medium.setFont(new Font("Arial",Font.BOLD,23));

large.setFont(new Font("Courier",Font.BOLD,32));

tiny.addActionListener(this);

small.addActionListener(this);

medium.addActionListener(this);

large.addActionListener(this);

tinypanel.add(tiny,BorderLayout.CENTER);

smallpanel.add(small,BorderLayout.CENTER);

medpanel.add(medium,BorderLayout.CENTER);

largepanel.add(large,BorderLayout.CENTER);

cardpanel.add("tiny",tinypanel);

cardpanel.add("small",smallpanel);

cardpanel.add("medium",medpanel);

cardpanel.add("large",largepanel);

setLayout(new BorderLayout());

add(cardpanel,BorderLayout.CENTER);

}

public void actionPerformed(ActionEvent e)

{

card.next(cardpanel);

}

}

Output:



**Ex. No. 3**

**PROGRAM TO CREATE APPLETS**

Aim

To write a java program to create applets with the following features

a. Create a color palette with matrix of buttons.

b. Set background and foreground of the control text area by selecting a color from color palette.

c. In order to select fore ground or background use checkbox controls as radio buttons.

d. To set background images.

Algorithm

Step 1: Write a html applet tag with code set class name and comment the tag

Step 2: Import all necessary packages and classes

Step 3: Define a class that extends applet and implements action listener and item listener

Step 4: Declare an array of buttons to set colors, two checkboxes for foreground and background colors

Step 5: Declare a text area to hold the text, a checkbox group for checkboxes

Step 6: Declare three panels, buttonpanel, palettepanel and checkpanel

Step 7: Declare a string, color.

Step 8: In the init() method, do the following:

i) Create the buttonpanel and set the layout to gridlayout of 3 X 3

ii) Create an array of 9 buttons for various colors

iii) Add action listener to each button and add all of them to button panel

iv) Create the checkbpanel and set its layout to flow layout

v) Create a checkbox group. Then create forground and background checkboxes

vi) Add item listener to each checkbox and add them to the checkpanel

vii) Create a text area and change its font to desired one.

viii) Create the palettepanel and set is layout to border layout

ix) Add the text area, buttonpanel, and checkpanel to various zones of the border layout

x) Add the palettepanel to the applet

Step 9: Write an empty itemStateChanged() method

Step 10: In the actionPerformed() method, do the following:

i) Get the action command in the string, color

ii) If foreground is checked then set the foreground color to the selected color

iii) If background is checked then set the background color to the selected color

Program

/\* <applet code=palette height=600 width=600>

</applet> \*/

import java.awt.\*;

import java.awt.event.\*;

import java.applet.\*;

public class palette extends Applet implements ActionListener,ItemListener

{

Button[] colors;

Checkbox foreground,background;

TextArea workarea;

CheckboxGroup cbg;

Panel buttonpanel,checkpanel,palettepanel;

String colour;

public void init()

{

buttonpanel=new Panel();

buttonpanel.setLayout(new GridLayout(3,3));

colors=new Button[9];

colors[0]=new Button("RED");

colors[1]=new Button("GREEN");

colors[2]=new Button("BLUE");

colors[3]=new Button("CYAN");

colors[4]=new Button("ORANGE");

colors[5]=new Button("WHITE");

colors[6]=new Button("BLACK");

colors[7]=new Button("YELLOW");

colors[8]=new Button("PINK");

for(int i=0;i<9;i++)

{

colors[i].addActionListener(this);

buttonpanel.add(colors[i]);

}

checkpanel=new Panel();

checkpanel.setLayout(new FlowLayout());

cbg=new CheckboxGroup();

foreground=new Checkbox("ForeGround",cbg,true);

background=new Checkbox("BackGround",cbg,false);

foreground.addItemListener(this);

background.addItemListener(this);

checkpanel.add(foreground);

checkpanel.add(background);

workarea=new TextArea(8,40);

workarea.setFont(new Font("Garamond",Font.BOLD,20));

palettepanel=new Panel();

palettepanel.setLayout(new BorderLayout());

palettepanel.add(workarea,BorderLayout.CENTER);

palettepanel.add(checkpanel,BorderLayout.EAST);

palettepanel.add(buttonpanel,BorderLayout.SOUTH);

add(palettepanel);

}

public void itemStateChanged(ItemEvent ie)

{

}

public void actionPerformed(ActionEvent ae)

{

colour=ae.getActionCommand();

if(foreground.getState()==true)

workarea.setForeground(getColour());

if(background.getState()==true)

workarea.setBackground(getColour());

}

public Color getColour()

{

Color mycolor=null;

if(colour.equals("RED"))

mycolor=Color.red;

if(colour.equals("GREEN"))

mycolor=Color.green;

if(colour.equals("BLUE"))

mycolor=Color.blue;

if(colour.equals("CYAN"))

mycolor=Color.cyan;

if(colour.equals("ORANGE"))

mycolor=Color.orange;

if(colour.equals("WHITE"))

mycolor=Color.white;

if(colour.equals("BLACK"))

mycolor=Color.black;

if(colour.equals("YELLOW"))

mycolor=Color.yellow;

if(colour.equals("PINK"))

mycolor=Color.pink;

return mycolor;

}

}

Output



**Ex. No. 4**

**PROGRAM TO SET URL, DOWNLOAD THE HOME PAGE & DISPLAY**

**CONTENTS OF THE PAGE**

Aim

To write a java program to download a page from a web site and also to display properties of the page

Algorithm

Step 1: Import all necessary packages

Step 2: Define a class

Step 3: In the main() method, which throws Exception, do the following:

i) Create a URL to any web site and open a URL Connection with it

ii) Get the date, content type, last modified and length of the page and display them

iii) Open an input stream with the URL Connection

iv) Till the end of the content read the content character by character then display it

v) If no content is available then display " no content is available"

Program

import java.net.\*;

import java.io.\*;

import java.util.Date;

class URLDemo

{

public static void main(String[] arg) throws Exception

{

int c;

URL yahoo = new URL("http://www.msn.com");

URLConnection yahoocon = yahoo.openConnection();

System.out.println("Date: " + new Date(yahoocon.getDate()));

System.out.println("Content-Type: " + yahoocon.getContentType());

System.out.println("Expires: " + yahoocon.getExpiration());

System.out.println("Last-Modified: " + new Date(yahoocon.getLastModified()));

int len = yahoocon.getContentLength();

System.out.println("Content-Length: " + len);

if(len>0)

{

System.out.println("+++++++++ CONTENT ++++++++");

InputStream input= yahoocon.getInputStream();

int i = len;

while(((c = input.read()) != -1) && (--i > 0))

{

System.out.print((char) c);

}

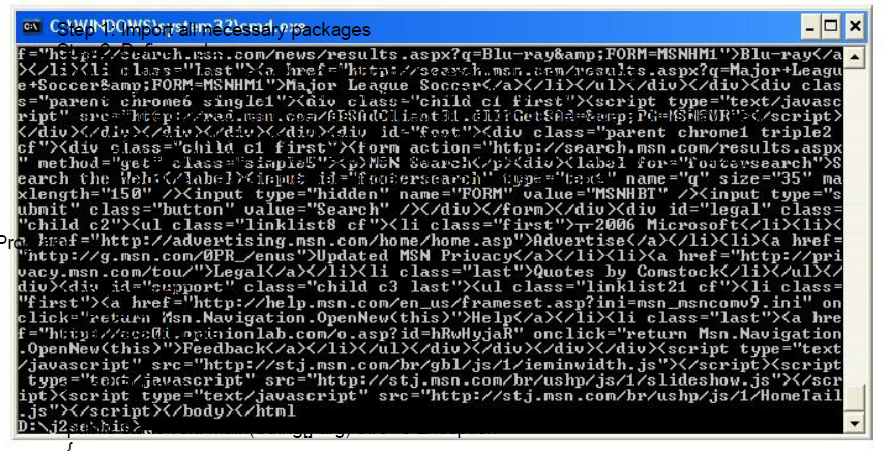
input.close();

}

else

System.out.println("No Content Available"); } }

Output:



**Ex. No. 5**

**IMPLEMENT HTTP, FTP, SMTP & POP3 USING SOCKETS**

**HTTP Request**

Aim

To write a java socket program to demonstrate HTTP Request

Algorithm

Step 1: Import all the necessary packages

Step 2: Create an URL to the server specifying the html page

Step 3: Get the host and port details from the URL

Step 4: Request the file from the server using GET method of HTTP Request

Step 5: Receive the response from the server

Step 6: Display the response on the console

Program

import java.io.\*;

import java.net.\*;

public class HTTP

{

public static void main(String[] args)

{

try

{

OutputStream to\_file = new FileOutputStream("f:\\temp.txt");

URL url = new

URL("http://www.wickedlysmart.com/HeadFirst/HeadFirstJava/HeadFirstJavaIndex.html");

String protocol = url.getProtocol();

String host = url.getHost();

int port = url.getPort();

if(port == -1) port = 80;

String filename =url.getFile();

System.out.println(filename);

Socket socket = new Socket(host, port);

InputStream from\_server = socket.getInputStream();

PrintWriter to\_server = new PrintWriter(socket.getOutputStream());

to\_server.print("Get" + filename +"\n\n");

to\_server.flush();

byte[] buffer = new byte[4096];

int byte\_read;

while((byte\_read = from\_server.read(buffer)) != -1)

{

to\_file.write(buffer,0,byte\_read);

System.out.print((char)byte\_read);

}

socket.close();

to\_file.close();

}

catch(Exception e)

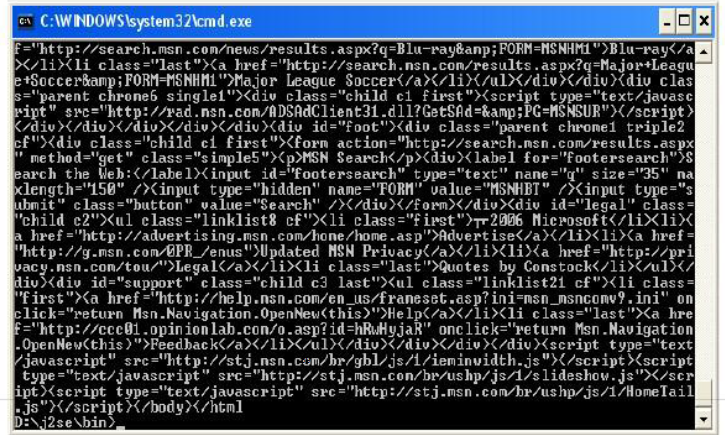
{

e.printStackTrace();

}

}}

Output



**FTP**

Aim

To write a java program to demonstrate a simple FTP operation

Algorithm

FTP Client:

Step 1: Establish a connection with the server at a particular port

Step 2: Specify the name of the file to be read

Step 3: Receive the contents of the file from the server

FTP Server:

Step 1: Accept the connection with the client

Step 2: Listen to the port for the name of the file to be sent

Step 3: Send the file character by character

Step 4: Terminate the connection

Program

FTP Client:

import java.io.\*;

import java.net.\*;

public class FTPClient

{

public static void main(String[] args)

{

try

{

Socket client = new Socket("127.0.0.1",10000);

PrintWriter writer = new PrintWriter(client.getOutputStream());

writer.println("f:/demo/HTTP.java");

writer.flush();

InputStreamReader stream = new InputStreamReader(client.getInputStream());

BufferedReader reader = new BufferedReader(stream);

String str = null;

while((str = reader.readLine()) != null)

{

System.out.println(str);

}

reader.close();

}

catch(Exception e)

{

System.out.println("Connection is terminated by the Server");

} } }

FTP Server:

import java.io.\*;

import java.net.\*;

public class FTPServer

{

public static void main(String[] arg)

{

try

{

ServerSocket server = new ServerSocket(10000);

Socket client;

client= server.accept();

InputStreamReader stream = new InputStreamReader(client.getInputStream());

BufferedReader reader = new BufferedReader(stream);

String filename = reader.readLine();

PrintWriter writer = new PrintWriter(client.getOutputStream());

FileInputStream fileStream = new FileInputStream(new File(filename));

int ch;

while((ch = fileStream.read()) != -1)

{

writer.write(ch);

writer.flush();

}

writer.close();

}

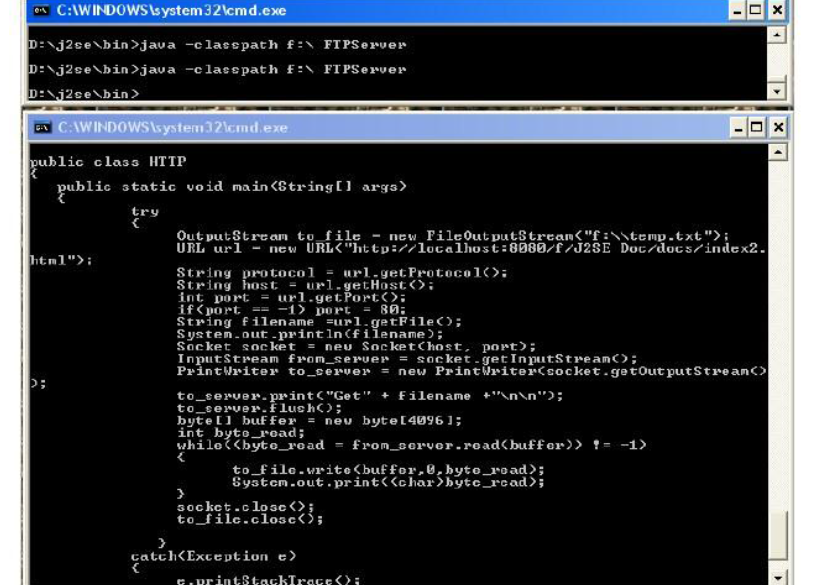
catch(Exception e)

{

e.printStackTrace();

} } }

Output



**SMTP**

Aim

To write a java socket program to implement a simple SMTP client

Algorithm

Step 1: Import all necessary packages

Step 2: Establish a connection with the server

Step 3: Read the acceptance from the server

Step 4: Say HELO to the server

Step 5: Read the greeting from the server

Step 6: Send sender address to server

Step 7: Read the verification of sender from server

Step 8: Send recipient address to server

Step 9: Read the verification of recipient from server

Step 10: Send DATA command to the server

Step 11: Read the start indication from server

Step 12: Send the message to the server

Step 13: Read the acceptance of message from server

Step 14: Close the connection

Program

import java.io.\*;

import java.net.Socket;

public class SMTPClient {

public static void main(String[] args) throws Exception {

String results = send("localhost", 25, "sender@somewhere.com", "localhost/localdomain", "Test Email", "<b>You got mail!</b>");

System.out.println(results);

}

public static String send(String host,int port,String from,String to,String subject,

String message) throws Exception

{

StringBuffer buffer = new StringBuffer();

try {

Socket smtpSocket = new Socket(host, port);

DataOutputStream output = new

DataOutputStream(smtpSocket.getOutputStream());

BufferedReader input =new BufferedReader(new InputStreamReader(

new

DataInputStream(smtpSocket.getInputStream())));

try {

read(input, buffer);

send(output, "HELO localhost.localdomain\r\n", buffer);

read(input, buffer);

send(output, "MAIL FROM: " + from + "\r\n", buffer);

read(input, buffer);

send(output, "RCPT to: " + to + "\r\n", buffer);

read(input, buffer);

send(output, "DATA\r\n", buffer);

read(input, buffer);

send(output, "Subject: " + subject + "\r\n", buffer);

send(output, message, buffer);

send(output, "\r\n.\r\n", buffer);

93

read(input, buffer);

smtpSocket.close();

}

catch (IOException e) {

System.out.println("Cannot send email as an error occurred.");

}

}

catch (Exception e) {

System.out.println("Host unknown");

}

return buffer.toString();

}

private static void send(DataOutputStream output,String data,StringBuffer buffer)

throws IOException

{

output.writeBytes(data);

buffer.append(data);

}

private static void read(BufferedReader br, StringBuffer buffer) throws IOException

{

int c;

while ((c = br.read()) != -1)

{

buffer.append((char) c);

if (c == '\n')

{

break;

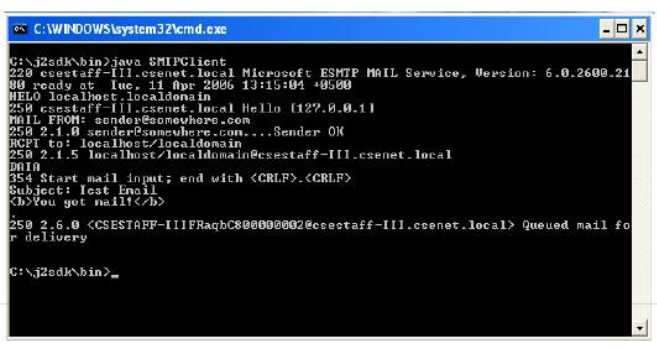
}

}

}

}

Output



**POP3**

Aim

To write a java socket program to implement a POP3 Client Algorithm

Step 1: Get the host name, mailbox user name and password

Step 2: Establish the connection with the server

Step 3: Get the number of messages

Step 4: Retrieve a message whose number is specified by the user

Step 5: Repeat steps 3 and 4 until the user enters Q to quit

Program

import java.io.\*;

import java.net.\*;

import java.util.\*;

public class Pop3ClientDemo

{

protected int port = 110;

protected String hostname = "localhost";

protected String username = "";

protected String password = "";

protected Socket socket;

protected BufferedReader br;

protected PrintWriter pw;

// Constructs a new instance of the POP3 client

public Pop3ClientDemo() throws Exception

{

try

{

// Get user input

getInput();

// Get mail messages

displayEmails();

}

catch(Exception e)

{

System.err.println ("Error occured - details follow");

e.printStackTrace();

System.out.println(e.getMessage());

}

}

// Returns TRUE if POP response indicates success, FALSE if failure

protected boolean responseIsOk() throws Exception

{

String line = br.readLine();

System.out.println("< "+line);

return line.toUpperCase().startsWith("+OK");

}

// Reads a line from the POP server, and displays it to screen

protected String readLine(boolean debug) throws Exception

{

String line = br.readLine();

// Append a < character to indicate this is a server protocol response

if (debug)

System.out.println("< "+line);

else

System.out.println(line);

return line;

}

// Writes a line to the POP server, and displays it to the screen

protected void writeMsg(String msg) throws Exception

{

pw.println(msg);

pw.flush();

System.out.println("> "+msg);

}

// Close all writers, streams and sockets

protected void closeConnection() throws Exception

{

pw.flush();

pw.close();

br.close();

socket.close();

}

// Send the QUIT command, and close connection

protected void sendQuit() throws Exception

{

System.out.println("Sending QUIT");

writeMsg("QUIT");

readLine(true);

System.out.println("Closing Connection");

closeConnection();

}

// Display emails in a message

protected void displayEmails() throws Exception

{

BufferedReader userinput = new BufferedReader( new InputStreamReader

(System.in) );

System.out.println("Displaying mailbox with protocol commands and responses below");

System.out.println("--------------------------------------------------------------");

// Open a connection to POP3 server

System.out.println("Opening Socket");

socket = new Socket(this.hostname, this.port);

br = new BufferedReader(new InputStreamReader(socket.getInputStream()));

pw = new PrintWriter(new OutputStreamWriter(socket.getOutputStream()));

// If response from server is not okay

if(! responseIsOk())

{

socket.close();

throw new Exception("Invalid POP3 Server");

}

// Login by sending USER and PASS commands

System.out.println("Sending username");

writeMsg("USER "+this.username);

if(!responseIsOk())

{

sendQuit();

throw new Exception("Invalid username");

}

System.out.println("Sending password");

writeMsg("PASS "+this.password);

if(!responseIsOk())

{

sendQuit();

throw new Exception("Invalid password");

}

// Get mail count from server ....

System.out.println("Checking mail");

writeMsg("STAT");

// ... and parse for number of messages

String line = readLine(true);

StringTokenizer tokens = new StringTokenizer(line," ");

tokens.nextToken();

int messages = Integer.parseInt(tokens.nextToken());

int maxsize = Integer.parseInt(tokens.nextToken());

if (messages == 0)

{

System.out.println ("There are no messages.");

sendQuit();

return;

}

System.out.println ("There are " + messages + " messages.");

System.out.println("Press enter to continue.");

userinput.readLine();

for(int i = 1; i <= messages ; i++)

{

System.out.println("Retrieving message number "+i);

writeMsg("RETR "+i);

System.out.println("--------------------");

line = readLine(false);

while(line != null && !line.equals("."))

{

line = readLine(false);

}

System.out.println("--------------------");

System.out.println("Press enter to continue. To stop, type Q then enter");

String response = userinput.readLine();

if (response.toUpperCase().startsWith("Q"))

break;

}

sendQuit();

}

public static void main(String[] args) throws Exception

{

Pop3ClientDemo client = new Pop3ClientDemo();

}

// Read user input

protected void getInput() throws Exception

{

String data=null;

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

System.out.print("Please enter POP3 server hostname:");

data = br.readLine();

if(data == null || data.equals("")) hostname="localhost";

else

hostname=data;

System.out.print("Please enter mailbox username:");

data = br.readLine();

if(!(data == null || data.equals("")))

username=data;

System.out.print("Please enter mailbox password:");

data = br.readLine();

if(!(data == null || data.equals("")))

password=data;

}

}

**Ex. No. 6**

**PROGRAMS TO CREATE SIMPLE CHAT APPLICATION WITH**

**DATAGRAMSOCKETS & PACKETS**

Aim

To write a java program to create a simple chat application with datagram sockets and packets

Algorithm

Server Side

Step 1: Import net and io packages and class

step 2: Create a datagram socket and datagram packet

step 3: While client send datagram packet to server listen to client port

step 4: Get the datagram packet into a string

step 5: Display the string

Client Side

step 1: Import net and io packages

step 2: Create a datagram socket and datagram packet

step 3: Get input from the user and convert the string into a datagram packet

step 4: send the datagram packet to the server through serve port

Program

Server Side

import java.io.\*;

import java.net.\*;

class Server

{

public static DatagramSocket serversocket;

public static DatagramPacket dp;

public static BufferedReader dis;

public static InetAddress ia;

public static byte buf[]=new byte[1024];

public static int cport=789,sport=790;

public static void main(String a[])throws IOException

{

serversocket=new DatagramSocket(sport);

dp=new DatagramPacket(buf,buf.length);

dis=new BufferedReader(new InputStreamReader(System.in));

ia=InetAddress.getLocalHost();

System.out.println("Server is waiting for data from client");

while(true)

{

serversocket.receive(dp);

String s=new String(dp.getData(),0,dp.getLength());

System.out.println(s);

}

}

}

Client Side

import java.io.\*;

import java.net.\*;

class Client

{

public static DatagramSocket clientsocket;

public static BufferedReader dis;

public static InetAddress ia;

public static byte buf[]=new byte[1024];

public static int cport=789,sport=790;

public static void main(String a[])throws IOException

{

clientsocket = new DatagramSocket(cport);

dis=new BufferedReader(new InputStreamReader(System.in));

ia=InetAddress.getLocalHost();

System.out.println("Client is sending data to Server ");

while(true)

{

String str=dis.readLine();

buf=str.getBytes();

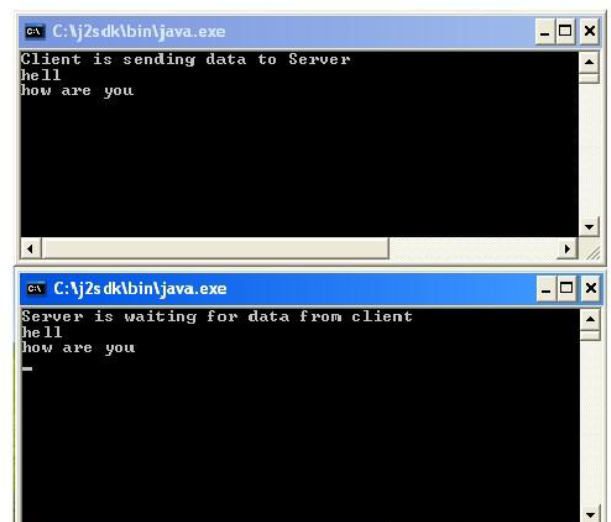
clientsocket.send(new DatagramPacket(buf,str.length(),ia,sport));

}

}

}

Output



**Ex. No. 7**

**PROGRAMS TO INVOKE SERVLETS FROM HTML FORMS USING**

**SERVLETS**

Aim

To write html and servlet to demonstrate invoking a servlet from a html

Algorithm

Step 1: Write the html file with form whose action attribute is set to the location of servlet and method is set to post

Step 2: Create an input in the html to trigger calling servlet

Step 3: Write a servlet program with doPost method which will give a response to the user as an html file

Program

Html

<html>

<head>

<title> Invoking Servlet From Html </title>

</head>

<form action = "http://localhost:8080/servlet/SampleServlet" method="GET">

<p>Press the button to invoke servlet<p>

<input type="submit" value="Go for it!">

</form>

Servlet

import javax.servlet.\*;

import javax.servlet.http.\*;

import java.io.\*;

public class SampleServlet extends HttpServlet

{

public void doGet(HttpServletRequest req, HttpServletResponse res) throws

ServletException

{

PrintWriter out;

try

{

out= res.getWriter();

out.print("<html><head><title> Response From Servlet </title></head>");

out.print("<body> <p> <h3> This Text Is Generated By Invoked Servlet </h3>

</body></html>");

}

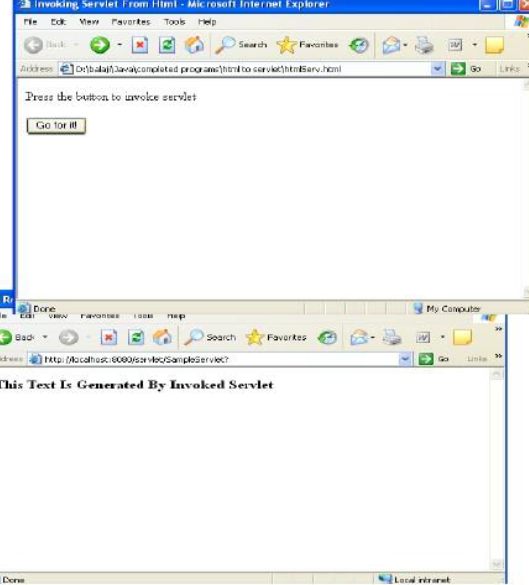
catch(IOException ie)

{

ie.printStackTrace();

} } }

Output



**Ex. No. 8**

**PROGRAMS TO CREATE 3 TIER APPLICATIONS USING SERVLETS**

**ONLINE EXAMINATION & STUDENT MARK LIST**

Aim

To write java servlet programs to conduct online examination and to display student mark list available in a database

Online Examination

Algorithm

HTML

Step 1: Write a html file which contains a form that is directed to a servlet

Step 2: Write various questions and choices

Step 3: Include to buttons reset and click

Servlet

Step 1: Import all necessary packages

Step 2: Define a class that extends servlet

Step 3: In the doPost() method, do the following:

i) Set the content type of the response to "text/html"

ii) Create a writer to the response

iii) Get a paratmeter from the request

iv) If its value is equal to right answer then add 5 to mark variable

v) Similarly repeat step iv) for all parameters

vi) Display the result in an html format using the writer

Program

HTML

<html>

<head><title>Exam Form</title></head>

<body>

<form action="http://localhost:8080/servlet/serv" method=POST>

<strong> <p> QUESTIONS: </p></strong>

1.JavaUses:

<p> <input type=radio name=ans1 value="inter">Interpreter

<input type=radio name=ans1 value="comp">Compiler

<input type=radio name=ans1 value="intercomp">Interpreter&Compiler

</p>

2.Net is an OS:

<p> <input type=radio name=ans2 value="yes">yes

<input type=radio name=ans2 value="no">no

</p>

3.ASP Uses:

<p> <input type=radio name=ans3 value="vbs">VBScript

<input type=radio name=ans3 value="js">JavaScript

<input type=radio name=ans3 value="vbjs">VBScript&JavaScript

</p>

<input type=submit value="click">

<input type=reset>

</form>

</body>

</html>

Servlet

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class serv extends HttpServlet

{

public void doPost(HttpServletRequest req,HttpServletResponse resp) throws

ServletException,IOException

{

int s=0;

resp.setContentType("text/html");

PrintWriter out=resp.getWriter();

String res=req.getParameter("ans1");

if(res.equals("intercomp"))

s=s+5;

res=req.getParameter("ans2");

if(res.equals("no"))

s=s+5;

res=req.getParameter("ans3");

if(res.equals("vbjs"))

s=s+5;

out.println("<html><body><p><center><h1><b>RESULT</b></h1><br><br><hr>"+s+"

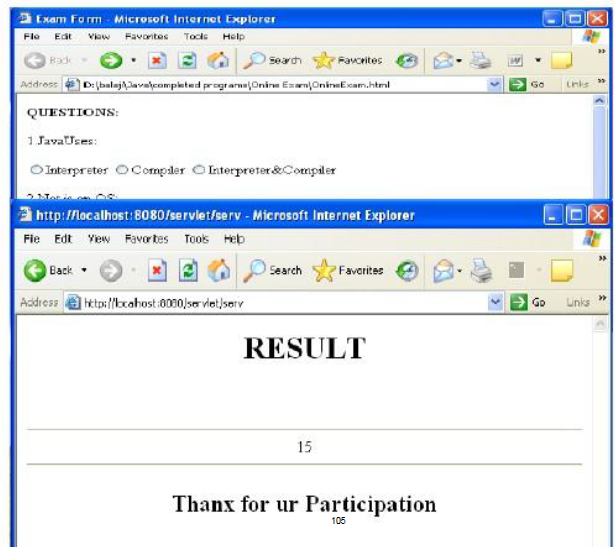
<hr><h2>Thanx for ur Participation</h2></center></p></body></html>");

out.close();

}

}

Output



Student Mark List Database

Algorithm

Step 1: Create a database in MS Access which contains students mark list information

Step 2: Create an ODBC interface to the database

Step 3: Write a servlet program to display the information in a html format

To create a Database and Table

Step 1: Open MS Access and create a database

Step 2: Design the table and populate the table

To create ODBC interface

Step 1: Click Start->Control Panel-> Administrative Tools

Step 2: Create a Microsoft Access Driver

Step 3: Add the database created using MS Access to it

To Write Servlet

Step 1: Import necessary to java packages and javax packages and classes

Step 2: Create a class that extends HttpServlet and implements ServletException and

IOException

Step 3: In the doGet() method, do the following:

i) Create a PrintWriter object

ii) Open a connection with the data source name

iii) Write a sql query and execute to get the resultset

iv) Display the resultset information in html form

Program

import java.io.\*;

import java.util.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

import java.sql.\*;

public class demoser extends HttpServlet

{

public void doGet(HttpServletRequest req,HttpServletResponse res) throws

ServletException,IOException

{

PrintWriter out=res.getWriter();

try

{

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

Connection conn = DriverManager.getConnection ("jdbc:odbc:tour");

Statement stmt = conn.createStatement ();

ResultSet rs1=stmt.executeQuery("select \* from Table1");

out.println("<html><body bgcolor=\"aqua\"

color=\"red\"><br><br><center><h1>Student Details</h1><br><br><table

border=1><tr><th>Name</th><th>RollNo</th><th>

Mark1</th><th>Mark2</th><th>Average</th><th>total</th></tr>");

while(rs1.next())

{

out.println("<tr> ");

out.print("<td>"+rs1.getString(1)+"</td>");

out.print("<td> "+rs1.getInt(2)+"</td>");

out.print("<td> "+rs1.getInt(3)+"</td>");

out.print("<td> "+rs1.getInt(4)+"</td>");

out.print("<td> "+rs1.getInt(5)+"</td>");

out.print("<td> "+rs1.getInt(6)+"</td>");

out.println("</tr> "); }

out.println("</table></center></body></html> ");

stmt.close();

conn.close();

}

catch (Exception e)

{

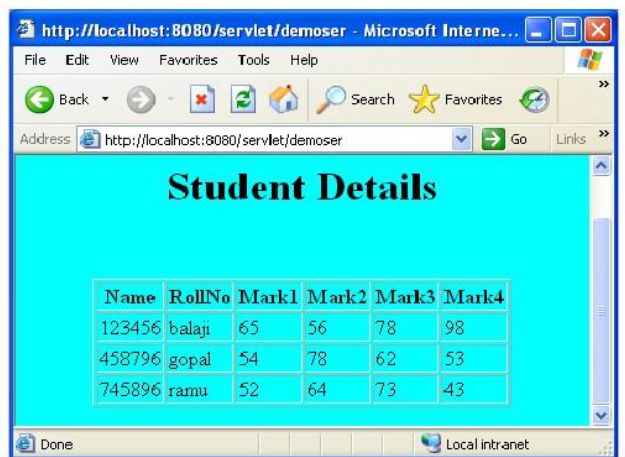
System.out.println("Error : "+e);

}

}

}

Output



**Ex. No. 9**

**PROGRAM TO CREATE A WEBPAGE USING HTML**

Aim

To create a web page which includes a map and display the related information when a hot spot is clicked in the map

Algorithm

Step 1: Create a html file with map tag

Step 2: Set the source attribute of the img tag to the location of the image and also set the usemap attribute

Step 3: Specify an area with name, shape and href set to the appropriate values

Step 4: Repeat step 3 as many hot spots you want to put in the map

Step 5: Create html files for each and every hot spots the user will select

Program

World.html

<html>

<head>

<title> World Information </title>

</head>

<body>

<map name="world">

<p> <img height=300 width=800 src = "..\..\images\pastel.gif" ismap

usemap=#world> <p>

<area name="asia" shape="rect" coords="520,20,730,185" href="asia.html">

<area name="measia" shape="circle" coords="480,125,30" href="measia.html">

<area name="samerica" shape = "circle" coords="235,200,55"

href="samerica.html">

<area name="africa" shape="circle" coords="420,180,55" href="africa.html">

<area name="namerica" shape = "rect" coords="55,15,325,180"

href="namerica.html">

<area name="europe" shape = "rect" coords="335,40,495,85" href="europe.html">

<area name="australia" shape="rect" coords="615,185,750,275"

href="oceania.html">

</map>

<br>

<p> <h3> General information about our world <h3>

<small><i>

<ul>

<li> Most populated country : China Population 1,307,000,000</li>

<li>world's tallest mountain, Mt Everest in Nepal 29,025 ft. (8.850 m)</li>

<li>world's lowest point, found in the Dead Sea, Israel/Jordan, at 1,286 ft (392m)

below sea level</li>

<li>world's deepest lake Lake Baykal, Siberia, Russia, 5369 ft (1637 m)</li>

<li>worlds longest river nile river 4,160 miles (6,693 km) in length</li>

<li>worlds largest desert sahara desert approximately 3,500,000 sq. miles (9,065,000

sq. km)</li>

<li>World's longest coast line: coast line in Canada longest at 243792 km (151485

miles)</li>

<li>Worlds's largest city: Tokyo, Japan (35327000 )</li>

</ul>

</i></small>

</body>

</html>

Africa.html

<head><title> AFRICA </title></head>

<body>

<h3 align=center>Africa includes 53 individual countries</h3>

<br>

<p>

<img align="left" src="..\..\images\africa.gif" height=320 width=320></img>

<br>

<table border=0>

<tr>

<th align=left>Continent Size:</th>

<td align=left>30,065,000 sq km (11,608,000 sqm)</td>

</tr>

<tr>

<th align=left>Percent of Earth's Land:</th>

<td align=left>20.2%</td>

</tr>

<tr>

<th align=left>Total Population:</th>

<td>877,500,000</td>

</tr>

<tr>

<th align=left>Most populated countries:</th>

</tr>

<tr>

<td></td><td><table>

<tr>

<th><small>Country</small></th>

<th><small>Population</small></th>

</tr>

<tr>

<td><small><i>Nigeria</i></small></td>

<td align=right><small><i>125,750,356</i></small></td>

</tr>

<tr>

<td><small><i>Egypt</i></small></td>

<td align=right><small><i>76,117,421</i></small></td>

</tr>

<tr>

<td><small><i>Ethiopia</i></small></td>

<td align=right><small><i>71,336,571</i></small></td>

</tr>

<tr>

<td><small><i>Congo</i></small></td>

<td align=right><small><i>58,317,930</i></small></td>

</tr>

<tr>

<td><small><i>South Africa</i></small></td>

<td align=right><small><i>44,448,470</i></small></td>

</tr>

<tr>

<td><small><i>Sudan</i></small></td>

<td align=right><small><i>39,148,162</i></small></td>

</tr>

</table></td>

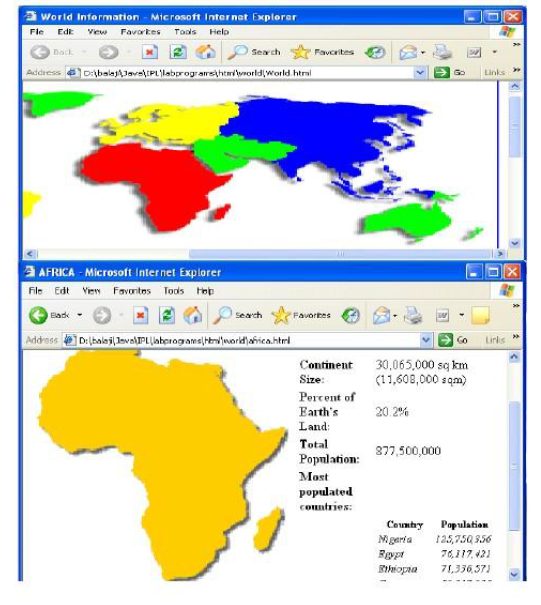
</tr>

</table>

</p>

</body>

Output



**Ex. No. 10**

**PROGRAM TO CREATE A WEBPAGE USING CASCADING STYLE**

**SHEETS, EMBEDDED SHEETS AND INLINE STYLE SHEETS**

To create web pages using college information to demonstrate the style sheets

**Algorithm**

Step 1: Create a web page with frame sets consisting two frames

Step 2: In the first frame include the links

Step 3: In the second frame set display the web page of the link

home.html

Step 1: Include html and head tags

Step 2: In the head insert title and link tags. The link should refer to the external style sheet

file

Step 3: Insert a frame set of two columns

Step 4: In the frame set insert two frames, first frame refers to link.html and second frame

refers to sit.html

link.html

Step 1: Include html, head, and body tags

Step 2: In the head tag, insert a link tag that refers to external style sheet page

Step 3: In the body tag, insert three links for home.html, department.html and courses.html

sit.html

Step 1: Include html, head, and body tags

Step 2: In the head tag, insert a link tag that refers to external style sheet page and a

embedded style tag

Step 3: In the body tag, insert all the necessary college information using various tags

step 4: Insert a table to disply the students strength of various years. In table tag, include a

style property

department.html

Step 1: Include html, head, and body tags

Step 2: In the body include all the necessary information

Step 3: If needed, insert external, inline or embedded styles

course.html

Step 1: Include html, head, and body tags

Step 2: In the body include all the necessary information

Step 3: If needed, insert external, inline or embedded styles

Program

home.html

<html>

<head>

<title>Sethu Institute Of Technology</title>

<link rel="stylesheet" href="..\styles\main.css" style="text/css">

</head>

<frameset cols="30%,\*" noresize="noresize" name="home">

<frame name="link" src="link.html">

<frame name="display" src="sit.html">

</frameset>

</html>

sit.html

<html>

<head>

<title> Sethu Institute of Technology </title>

<link rel="stylesheet" href="..\styles\main.css" type="text/css">

<style>

p.important{font-family:"Tahoma"; border:solid; border-width:thin;

width:100%;color:blue}

</style>

</head>

<body>

<marquee> <h3>Welcome to Sethu Institute Of Technology </h3> </marquee>

<p>

Our college, SETHU INSTITUTE OF TRECHNOLOGY was established in the year

1995 at Pulloor, Kariapatti, Virdhunagar District, by SETHU EDUCATIONAL TRUST

with the motto of KNOWLEDGE, SERVICE, PROGRESS”.

</p>

<p>

Sethu Institute of Technology, the brainchild of our Chairman, <span class=highlight>

Mr. S. Mohamed Jaleel,</span> renders its noble service since 1995 to the sustained growth

of the educationally, industrially, economically backward rural area.

</p>

<p class="important">

The college started with 180 students in the year 1995 with three disciplines, have

flourished to the level of 2100 students in 9 disciplines with 2 Post Graduate Courses.

</p>

<p>

<span class=underline>Location:</span>

<br>

The college is situated on Madurai-Tuticorin National Highway (NH 45B), Pulloor,

Kariapatti, Virudhunagar District over 45 acres. The nearest Landmark is Madurai Airport,

which is about 12 Km from the college.

</p>

<p>

<span class=underline>Growth of the Institution:</span>

<br>

<table Style ="width:100%;border-color:red;text-align:center" >

<tr><th id=1>Year</th><th id=1>Students</th></tr>

<tr><td>1995-96</td><td>195</td></tr>

<tr><td>1996-97</td><td>435</td></tr>

<tr><td>1997-98</td><td>691</td></tr>

<tr><td>1998-99</td><td>989</td></tr>

<tr><td>1999-00</td><td>1099</td></tr>

<tr><td>2000-01</td><td>1337</td></tr>

<tr><td>2001-02</td><td>1488</td></tr>

<tr><td>2002-03</td><td>1770</td></tr>

<tr><td>2003-04</td><td>2100</td></tr>

</tbody>

</table>

</p>

</body>

</html>

link.html

<html>

<link rel="stylesheet" href="..\styles\main.css">

<body>

<ul>

<li><bullet><a href="sit.html" target="display">Home Page</a></bullet></li>

<li><a href="department.html" target="display">Departments</a></li>

<li><a href="course.html" target="display">Courses</a></li>

</ul>

</body>

</html>

main.css

BODY

{

FONT-SIZE: 1em;

MARGIN-LEFT: 5%;

COLOR: #099669;

TEXT-INDENT: 3em;

MARGIN-RIGHT: 5%;

FONT-STYLE: italic;

FONT-FAMILY: verdana, sans serif

}

H1

{

COLOR: #000099;

FONT-FAMILY: tahoma

}

H2

{

COLOR: #000099;

FONT-FAMILY: tahoma

}

H3

{

COLOR: #000099;

FONT-FAMILY: tahoma

}

H4

{

COLOR: #000099;

FONT-FAMILY: tahoma

}

H5

{

COLOR: #000099;

FONT-FAMILY: tahoma

}

H6

{

COLOR: #000099;

FONT-FAMILY: tahoma

}

TH#1

{

FONT-SIZE: 2em;

TEXT-TRANSFORM: uppercase;

COLOR: maroon;

FONT-FAMILY: Arial, 'Arial Black', 'Arial Greek';

TEXT-ALIGN: center

}

SPAN.underline

{

TEXT-DECORATION: underline

}

SPAN.highlight

{

COLOR: white;

BACKGROUND-COLOR: transparent

}

