JAVA APPLICATION DEVELOPER

PROJECT REPORT

Submitted in partial fulfillment of the
requirement for the award of the degree of the

P-1120

Bachelor of Engineering in Information Technology

Of

Bharathiar University, Coimbatore.

Submitted by

S. HARISH
0027S0076

D. SRINIVSAN
0027S0113

Under the guidance of

Ms. S. SUSANNE ROSELINE MARY. BE.
Lecturer.

DEPARTMENT OF INFORMATION TECHNOLOGY
KUMARAGURU COLLEGE OF TECHNOLOGY,
COIMBATORE – 641006.

MARCH 2004.

DEPARTMENT OF INFORMATION TECHNOLOGY

KUMARAGURU COLLEGE OF TECHNOLOGY

(Affiliated to Bharathiar University, Coimbatore)

CERTIFICATE

This is to certify that the project entitled

JAVA APPLICATION DEVELOPER

is done by

S. HARISH
0027S0076

D. SRINIVASAN
0027S0113

and submitted in partial fulfillment of the requirement for the award of the degree of the

Bachelor of Information Technology

Of

Bharathiar University, Coimbatore.

S. Thangasamy

Professor & Head of the department
(Dr.S.Thangasamy)

S. Susanne 23/03/04

Guide
(Ms. S.Susanne Roseline Mary)

Certified that the candidates were examined by us in the project work Viva voice examination held on 26-03-2004.

S. Veerapathi 26/3

Internal Examiner

26/3

External Examiner

DECLARATION

DECLARATION

We,

S. Harish	**0027S0076**
D. Srinivasan	**0027S0113**

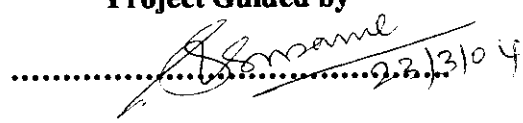
declare that the project entitled "JAD (Java Application Developer)", is done by us and to the best of our knowledge, a similar work has not been submitted earlier to the Bharathiar University or any other institution, for fulfillment of the requirement of the course study.

This project report is submitted on the partial fulfillment for the requirement for the awards of the degree of Bachelor of Engineering in Information Technology from Bharathiar University.

Place: Coimbatore. **[S. Harish]**

Date: 23/3/04 **[D. Srinivasan]**

Project Guided by



Ms. S. Susanne Roseline Mary B.E.,

ACKNOWLEDGEMENT

ACKNOWLEDGEMENT

We take this opportunity to express our gratitude to all the hands and hearts without whom this project would be incomplete.

We are grateful to Dr.K.K.Padmanaban, Principal, Kumaraguru College of Technology for giving us a golden opportunity to embark on this project.

We would like to express our heartful thanks to Dr.S.Thangasamy, Head of the Department of Computer Science and Engineering and Mr.K.R.Baskaran, Assistant Professor, Department of Information Technology, Kumaraguru College of Technology for providing us guidance and inspiration and enabling us to complete the project work.

We express a special thanks to our guide Miss.S.Susanne Roseline Mary B.E, Department of Information Technology for her excellent guidance and moral support enabling us to complete the project work.

Last but not the least, my heartiest thanks to our beloved parents and friends for their kind cooperation and encouragement enabling us to complete the project work successfully.

CONTENT

SYNOPSIS

SYNOPSIS

This project "*JAD (JAVA APPLICATION DEVELOPER)*" is to develop a Wizard and a Text Pad for java programmers using the Java Language. As Java applet applications are mostly used in the web pages, the coding for applets takes considerable amount of users time and as Sun Micro system does not provide any kind of tool for creating the application with out writing source code, This project provides an intelligent environment containing a number of ready-made tools(any of the GUI components of Java AWT package) to create the application without writing the source code.

Thus in JAD, the Wizard allows the users to compile and run the generated source code with going to the DOS prompt. The applet applications generated using the Wizard can be accessed in the web pages also. The Textpad can be used to develop standalone applications in Java. The IBMJS Speech Engine embedded in JAD helps the users to include the available GUI components into their application in their own voice.

INTRODUCTION

INTRODUCTION

CURRENT STATUS OF THE PROBLEM TAKEN UP:

In the present situation users have to go for the DOS Prompt for their compilation and running of java applications. Sun Microsystems does not provide any kind of tool for creating the application with out writing source code and doing compilation and running in the same environment.

RELEVANCE AND IMPORTANCE

Our project aims at reducing the efforts of the users while they develop their applet applications. It provides help about Java's built in classes more interactively to the programmer. It also uses IBM Via Voice (a software with an exciting feature of allowing parameters that makes speech synthesis and recognition easier) to develop the applications in an user friendly manner.

PROPOSED SYSTEM:

We have proposed a System namely "Java Application Developer " with an immense user friendliness. The proposed system is for the following:

- ✳ Providing an intelligent environment (Automation of source code) which contains a number of readymade tools to create applications with out writing the source code.
- ✳ Using the wizard and Text Pad compilation and execution of the generated source code can be done with out going to the DOS prompt.
- ✳ Using the wizard it is possible to generate screen design for any back end table.
- ✳ Designing of applets and standalone applications can be made through the Text pad.
- ✳ Providing easier debugging environment.
- ✳ Generation of error free applications in a shorter period of time.
- ✳ The generated source code can be viewed by the users at any time.

LITERATURE REVIEW

LITERATURE REVIEW

AN OVERVIEW:

"JAD (Java Application Developer)" is concerned with providing the Java programmers an easiest environment to develop and compile their applet applications. It supports automatic code generation and there by eliminates the need of writing source code for GUI related Java applications.

WIZARD:

Wizard is a kind of case tool ([CASE] Computer Aided Software Engineering). It contains lot of Graphical User Interface (GUI) components to create any application. It generates the source code for the users according to their requirements.

By reducing the overhead of writing, compiling and debugging the source code Wizard offers an easiest environment for creating the Graphical User Interface (GUI) oriented applications. IBMJS supports speech recognition and synthesis.

So the end users can also create their applications without the intervention of the programmers.

BENEFITS:

- This software allows even the end user to design and use Java applets in his web page.
- This project saves a considerable amount of users time while writing the source code.
- The applications that are built using this tool are free from errors since the source code is automatically generated.
- All the processes related with the creation of a Java application can be done on a single Window.
- The generated source code can be viewed by the users at any time.
- Using the wizard it is possible to generate screen design for any back end table.

SOFTWARE DESCRIPTION

SOFTWARE DESCRIPTION

PROBLEM DEFINITION:

Now a days Java applet applications are mostly used in the web pages. Sun micro system does not provide any kind of application developing environment to the user.

So in order to create the applet application we explicitly write codes by using Dos editor or notepad. After that we will be going to the Dos prompt for executing these applications. A programmer has to spend their time for designing the application, writing the source code and debugging the source code. so here in this project we have strived to create an programming environment where the user can be at ease while developing his application..

METHODS OF PROBLEM SOLVING:

The following steps solves our basic problem of automation of the source code development

- Creating the AWT components whenever the user is in need of it.
- Keep track of the change in their properties whenever the user modifies it.

- Writing the components with their properties into the file name specified by the user.
- To create a data file which is used for the further updating of the application.

SYSTEM SPECIFICATION:

This software has been developed on Pentium III-1.13 GHz computer. It is very powerful and high performance multi-user computing system. It is applicable in the area where programs need more memory and fast processing.

The minimum requirements for our project:

CPU Type	:	Pentium III
Cache Memory	:	512 K
CPU Clock	:	1.13 GHz
Hard Disk	:	4.3 GB
RAM Memory	:	256 MB
Operating System	:	Windows 98
Monitor	:	14" Samtron color monitor.
Input	:	Microphone

IMPLEMENTATION

IMPLEMENTATION

Developing & Operating Environment

JDK 1.3

It allows the user to do the following

- Writing robust and reliable programs.
- Building an application on any platform and run the application on any platform without recompiling the code.
- Distribute your application over the network in a secured fashion.
- The features like security, core API, distributed and dynamic, multithreading, object oriented makes java as a powerful tool for internet applications.

Support Provided By Jdk1.3:

Java mainly supports two types of programming, such as

1. Applet programming

The Java applet is a Java program that is launched from a web document that is the class file is embedded in an html file. The Java applet programs are window-based application that is displayed in the windows or frames or dialogs.

Java application wizard is an applet-oriented application. So the Java application contains the following support to the wizard.

- ❖ JFC support.
- ❖ Input output streams.
- ❖ JDBC support.
- ❖ Network support.

2. Application programming

A Java application runs in the simplest possible environment. It only gets input from the command line and displays output in the console.

AWT (Abstract Windows Toolkit)

It provides the capability to create platform independent, graphical user interface based programs and is very important contributor to Java's popularity.

The Java provides four classes to create and manage the graphical user interface component in the abstract window tool package. The classes are Basic class, Container class, Layout class, Menu class, Stream class etc.

SYSTEM DESCRIPTION

SYSTEM DESCRIPTION

PROJECT DESCRIPTION:

Our project mainly consists of two workspaces, which the Java programmers can use for developing their applications. These workspaces are,

- Java Wizard.
- Java Text Pad.

Types of Wizard:

The wizard is classified in to three types, such as

- Application Wizard.
- Active-X Wizard.
- Database Wizard.

Application Wizard:

Application wizard allows the user to create the following type of applications.

- Single document interface (SDI) applications.
- Multiple Document Interface (MDI) applications.
- Dialog Based.

It displays simple dialog box for the user input. These can be used for the following purposes.

- Opening a file.
- Save the file.
- Setting print options.
- Selecting the color and font.
- SDI applications.

Active-X Wizard:

An active-x control wizard allows the users to create the following controls.

- Animation Controls.
- Toolbar Controls.
- Tree View Controls.
- Status Bar Controls.
- Slider Controls.

Database wizard:

Data base wizard allows the user to process the database and create database applications such as the form and reports.

Java Wizard:

The Java wizard is mainly concerned with the automatic code generation. This workspace saves as a wizard for the programmers, which generates the code for the users' applications. The Java wizard development is divided in to three main modules.

- ❖ Wizard Layout.
- ❖ Automatic Code Generation.
- ❖ Run time environment

Wizard Layout:

The wizard layout is one of the important modules, which contains a number of GUI components and menus. This includes work sheets, font, color dialog and properties. Some of these components are

- **Menu**: It contains collection of choice to interact with the wizard. Each menu item represents some thing that can be selected by the user.
- **Toolbox**: Toolbox contains collection of tools such as buttons, text field, text area, radio buttons etc., each tool is identified by standard icon. You can select the desired tool from this toolbox in order to create your application.

- **Text field:** The text field is a single-line text-entry area, usually called an edit control. Text fields allow the user to enter string and to edit the text.
- **Label:** The label is a string that can be displayed on the screen. Usually this is used to name the objects.
- **Text area:** The text area is used to handle multiple lines of text.
- **List:** The list provides a compact multiple-choice scrolling selection lists. A list object can be constructed to show any number of choices in the visible window.
- **Scroll bar:** Scroll bars used to select continuous value between a specified minimum and maximum. Scroll bar may be oriented horizontally and vertically.
- **Check box:** A check box is a control that is used to turn an option on or off. It consists of a small box that can either contain a check mark or not.
- **Radio Button:** Radio button is a control that is used to turn an option on or off. It also consists of a small circle that can either contain a dot mark or not.
- **Font Dialog:** It supports all the Java fonts and we can assign the font to a desired GUI component in the work sheet. Also different font styles are also applicable to these components.

- **Color Dialog:** the Java color chooser class in the JSL allows you to select the needed color.
- **Properties:** By using the property dialog we can change the values, width, height and other attributes of the component.

Automatic Code Generation:

Automatic code generation is the most important part of our project. Whenever a user designs their layout in worksheet the source code is automatically generated in the text format by using the object output stream and file stream.

After the external designing in worksheet the user can write a control to desired component by using the vector class in Java.Whenever the user double clicks the component that time the vector is opened. So the user can write their desired event in the desired position. The code generation part of the system generates three types of files with the same name that are with the extension

.dat file

.java file

.applet file

The .dat file and the .java files is produced for both the applet application and frame application, while the .applet file is created only for

the applet application. The source code generated by the wizard can be directly opened from the wizard using the Java Text pad.

Run Time Environment:

The Run Time Environment enables the application to be compiled for the byte code and executes this byte code.

In the run time environment the compilation is the process of generating the class files of the application. The java compiler does the compilation as follows,

javac <filename>

The file that is taken for the compilation must be a Java file and with the extension as .java, the output file produced is of the type .class which is called as the byte code.

The java applications can be executed in two ways, when the application is a applet application the we use the appletviewer or any java compatible browsers for the execution. If the same application is a stand-alone application then we use the java interpreter to execute the application.

JAVA TEXT PAD:

The Java Text pad is the next part of our project, which is used for text editing and also debugging the java applications. The source code

generated by the java wizard can be viewed by using the java text pad. This text pad can be used for,

- Open an existing application.
- Edit a new application.
- Debug a java application.
- Compile a java application.
- Executed the compiled application.
- Complete help about java classes.

In text pad some editing options like cut, copy, paste, find and replace are also provided. The foreground and background of the workspace can also be changed.

TESTING

TESTING:

SYSTEM TESTING:

System Testing is the process of checking if the developed system is working according to the original objectives and requirements. Initially the system should be tested experimentally with the sample input so as to ensure that the software works according to specifications and in the way expect it to work. When it is found to be working in the right manner then the actual input is fed and the whole system is tested to check its performances.

Sample input should be compiled and executed in the software and is compared with the results obtained under Dos prompt. It is also important that the system correctly identifies errors and listing of these makes a way of easy correction.

DIFFERENT LEVELS OF TESTING:

Unit Testing:

Unit Test focuses on verification effort on the smallest unit of the design module. The Unit test considerations are

- Interface errors
- Error handling paths
- Integrity of local data structures

INTEGRATION TESTING:

Integration testing is a systematic technique for constructing the program structure. Each testing is done in separately. The bottom up approach was applied.

VALIDATION TESTING:

Validation Testing is carried out to verify whether the software function is a manner that is expected by the users, Alpha validity is done.

Thus system testing is done before the system is implemented.

BIBLIOGRAPHY

BIBLIOGRAPHY:

- "Development Intranet Application with Java "by Jerry Alban.
- "Hacking Java Professional Resource Kit" by Mark Wutka.
- "Java AWT Reference" by John Zukowski, April 1997.
- "Java By Examples" by Clayton Walnum.
- "Java Fundamental Classes Reference" by Mark Grand & Jonathan Knudsen, May 1997.
- "Java Language Reference", by Mark Grand, July 1997.
- "Java script Guide Index" E-Book.
- "Java Script Manual of Style", by Marc Johnson.
- "Special Edition Using Jscript", by Mark Reynolds and Jerry Honeycutt.
- www.Ibm.com
- www.vivisimo.com

CONCLUSION

CONCLUSION

We have developed software, which runs on the top of the java software, which inherently helps in semi automation of developing source code for various events met towards implementing GUI components.

To enhance the application and for easy accessing the following features are used. The screen of data form is automatically designed and corresponding codes are generated automatically. It is possible to compile the designed java program directly through windows under Dos.

FUTURE ENHANCEMENTS:

In future this software would be enhanced to automate the source code development for the Java Swing component. Also by migrating towards Java BEANS this software would prove to be a good tool to make Java a user-friendly language.

GLOSSARY OF TERMS

GLOSSARY OF TERMS:

- JVM-Java Virtual Machine
- JDK-Java Development Kit
- AWT-Abstract Windows Toolkit
- CASE-Case Aided Software Engineering
- GUI-Graphics User Interface
- IBMJS-International Business Machines Java Speech
- OS-Operating System
- JFC-Java Foundation Class
- JDBC-Java Data Base Connectivity
- JIT-Just In Time

APPENDIX

```
import java.io.*;
import java.sql.*;
import javax.swing.*;
import java.awt.*;
import java.lang.reflect.*;
import java.awt.event.*;
import java.util.Properties;
import java.util.Enumeration;
import java.util.StringTokenizer;
import javax.swing.*;


class DesMain1 extends Frame implements ActionListener
{

        Button b2,b3;

        DesMain1()
        {

                b2 = new Button("Java TextPad");
                    b3 = new Button("Exit");

                b2.setSize(250,25);
                b3.setSize(250,25);

                b2.setLocation(50,250);
                b3.setLocation(50,300);

                b2.addActionListener(this);
                b3.addActionListener(this);

                add(b2);
                add(b3);
                show();


                }

        public void actionPerformed(ActionEvent e)
        {


                if (e.getSource() == a2)
                {
                  try
                  {
                                OpenNewFile ope2=new OpenNewFile();
                                ope2.show();
                  }catch(Exception dde){}
                }

                if (e.getSource() == a3)
                        System.exit(0);
         }
```

```
        public static void main(String argv[])
        {
            DesMain1 dm = new DesMain1();
        }
}



class OpenNewFile extends Frame implements ActionListener
{
        public TextArea ta;
        MenuBar mb=new MenuBar();

        Menu menuarray[]=new Menu[3];
        MenuItem itemarray[][]=new MenuItem[3][8];

        String menuname[]={"File","Edit","Build"};
        String itemname[][]={
            {"Clear TextPad","New Wizard View","Open File...","Save
File...","Save File As...","-","Exit TextPad"},
            {"Cut ctrl+x","Copy ctrl+c","Paste ctrl+v"},
            {"Compile","-","Run","Run Applet"}};

        String FileName="",ClassName="";

        JHelp jhelp;
        JXHelp jxhelp;
        String Fname;
        boolean flg=false;

        Choice fc,sc;
        Button fb,fi,fp,fbi;

        Panel pa = new Panel();
        Panel p1 = new Panel();

       //  Comp_Det[] CD = new Comp_Det[50];
         ObjectOutputStream Out;
         ObjectInputStream In;
        // MouseHandler mh;

         BuildApplication bapp;

         Runtime rt;
         Process prc =null;
         Component BackComp;
         int fontstyle=0;
         String fontname="arial";
         int fontsize=12;
         boolean flag=false;

         public void paint(Graphics g)
         {
                Toolkit tk2 = getToolkit();
                setIconImage(tk2.getImage("icon.gif"));
```

```
}

OpenNewFile()throws Exception
{
        super("Java TextPad");
        setSize(750,525);
        setLocation(1,1);
        setLayout(new BorderLayout());

        pa.setLayout(null);
        pa.setBackground(Color.gray);
        add(pa,"Center");

        p1.setLayout(new FlowLayout(FlowLayout.LEFT));
        p1.setBackground(Color.white);

        add(p1,"North");

       // Add_FontList();

   Fname = FileName;


        for(int i=0;i<menuname.length;i++)
               menuarray[i]=new Menu(menuname[i]);

        for (int i=0;i<menuname.length;i++)
        for (int j=0;j<itemname[i].length;j++)
        {
                itemarray[i][j] = new MenuItem(itemname[i][j]);
                itemarray[i][j].addActionListener(this);
                menuarray[i].add(itemarray[i][j]);
                }

        for (int i=0;i<menuname.length;i++)
                mb.add(menuarray[i]);


        setMenuBar(mb);


        ta = new TextArea(32,98);
        add(ta);

        addWindowListener(new WindowAdapter()
        {
                public void windowClosing(WindowEvent e)
                {
                        dispose();
                        }
                });

        show();
        }

  OpenNewFile(Frame fa,String FileName)throws Exception
  {
```

```
    super("View Source Code");
    setSize(750,525);
    setLocation(1,1);
    setLayout(new BorderLayout());

    pa.setLayout(null);
    pa.setBackground(Color.gray);
    add(pa,"Center");

    p1.setLayout(new FlowLayout(FlowLayout.LEFT));
    p1.setBackground(Color.white);

    add(p1,"North");

//    Add_FontList();
    Fname = FileName;


    for(int i=0;i<menuname.length;i++)
            menuarray[i]=new Menu(menuname[i]);

    for (int i=0;i<menuname.length;i++)
    for (int j=0;j<itemname[i].length;j++)
    {
            itemarray[i][j] = new MenuItem(itemname[i][j]);
            itemarray[i][j].addActionListener(this);
            menuarray[i].add(itemarray[i][j]);
            }

    for (int i=0;i<menuname.length;i++)
            mb.add(menuarray[i]);


    setMenuBar(mb);


    ta = new TextArea(32,98);
    add(ta);

    FileInputStream in = new FileInputStream(FileName);
    byte[] buff = new byte[in.available()];
    in.read(buff,0,in.available());
    in.close();

    String src = new String(buff,0,buff.length);
    ta.setText(src);

    addWindowListener(new WindowAdapter()
    {
            public void windowClosing(WindowEvent e)
            {
                    dispose();
                    }
            });

    show();
    }
```

```
public void itemStateChanged(ItemEvent e)
   {
                    if (e.getSource() == fc)
                    {
                         fontname=fc.getSelectedItem();
                         ta.setFont(new Font(fontname,fontstyle,fontsize));
                    }

                    if (e.getSource() == sc)
                    {
                         fontsize=Integer.parseInt(sc.getSelectedItem());
                         ta.setFont(new Font(fontname,fontstyle,fontsize));
                    }

           }

 public void actionPerformed(ActionEvent e)
 {
    String s = e.getActionCommand();
    System.out.println(s);
     try
     {

                  if(s.equals("Clear TextPad"))

      {
            ta.setText("");
            flg=true;
      }
      /*if(s.equals("New Wizard View"))
      {
            JavaDes jdes=new JavaDes();
            jdes.show();
      }*/
      if(s.equals("Open File..."))
      {
          FileDialog fd = new FileDialog(this,"Open File..",0);
          fd.show();
          if ((fd.getDirectory() != null) && (fd.getFile() != null))
          {
                   ClassName = fd.getFile();
                   FileName = fd.getDirectory() + fd.getFile();
                   }
          else
          {
                   ClassName = "";
                   FileName = "";
                   }
          FileInputStream in = new FileInputStream(FileName);
          byte[] buff = new byte[in.available()];
          in.read(buff,0,in.available());
          in.close();

          String src = new String(buff,0,buff.length);
```

```
    ta.setText(src);
    flg=false;
    System.out.println(FileName);
}
if(s.equals("Save File..."))
{
 if(!flg)
 {

      FileOutputStream out = new FileOutputStream(Fname);
      String src = ta.getText();
      byte[] buff = src.getBytes();
      out.write(buff,0,buff.length);
      out.close();
 }
 else
 {
     FileDialog fd = new FileDialog(this,"Save File As..",1);
     fd.show();
     if ((fd.getDirectory() != null) && (fd.getFile() != null))
     {
             ClassName = fd.getFile();
             FileName = fd.getDirectory() + fd.getFile();
             }
     else
     {
             ClassName = "";
             FileName = "";
             }

      FileOutputStream out = new FileOutputStream(FileName);
      String src = ta.getText();
      byte[] buff = src.getBytes();
      out.write(buff,0,buff.length);
      out.close();
 }
}
if(s.equals("Save File As..."))
{
     FileDialog fd = new FileDialog(this,"Save File As..",1);
     fd.show();
     if ((fd.getDirectory() != null) && (fd.getFile() != null))
     {
             ClassName = fd.getFile();
             FileName = fd.getDirectory() + fd.getFile();
             }
     else
     {
             ClassName = "";
             FileName = "";
             }

      FileOutputStream out = new FileOutputStream(FileName);
      String src = ta.getText();
      byte[] buff = src.getBytes();
      out.write(buff,0,buff.length);
      out.close();
```

```
        }
        if(s.equals("Exit TextPad"))
        {
              dispose();
        }
                    if(s.equals("Compile"))
        {
              try{

                          bapp=new BuildApplication("Compile File");
                         bapp.show();
              }catch(Exception ffff){ffff.printStackTrace();}
        }

        if(s.equals("Run"))
        {
              try{
                              bapp=new BuildApplication("Run File");
                              bapp.show();
                        }catch(Exception ffff){}
        }

        if(s.equals("Run Applet"))
        {
              try{
                              bapp=new BuildApplication("Run Applet");
                              bapp.show();
              }catch(Exception ffff){}
        }


      }catch(IOException ioee){}

    }
}

class ForeChooser extends JFrame
{
      JColorChooser jcc=new JColorChooser();
      public Color col;
      public ForeChooser(String fb)
      {
                getContentPane().setLayout(null);
                Toolkit tk2 = getToolkit();
                setIconImage(tk2.getImage("icon.gif"));

                col=jcc.showDialog(getContentPane(),fb,Color.red);
           dispose();
      }
}


class BuildApplication extends Frame implements ActionListener
{
      Label lab=new Label("");
      Label lab2=new Label("");
```

```
Button b1,b2;
int fl=0;
String errormessage;

Runtime rt;
Process prc =null;
Component BackComp;

BuildApplication(String title)
{
  super(title);
  setSize(550,200);
  setLocation(100,100);
  Toolkit tk2 = getToolkit();
  setIconImage(tk2.getImage("icon.gif"));

  setLayout(null);
  lab.setBounds(20,40,400,30);
  lab.setForeground(Color.blue);
  lab.setFont(new Font("TimesRoman",Font.BOLD,16));
  add(lab);

  lab2.setBounds(20,70,500,30);
  lab2.setForeground(Color.blue);
  lab2.setFont(new Font("TimesRoman",Font.BOLD,16));
  add(lab2);

  b1=new Button("OK");
  b1.setBounds(150,120,50,30);
  b1.setForeground(Color.blue);
  b1.setFont(new Font("TimesRoman",Font.BOLD,16));
  add(b1);
  b1.addActionListener(this);
  b1.setEnabled(false);

  b2=new Button("Cancel");
  b2.setBounds(220,120,80,30);
  b2.setForeground(Color.blue);
  b2.setFont(new Font("Comic Sans MS",Font.BOLD,16));
  add(b2);
  b2.addActionListener(this);
  b2.setEnabled(false);

          addWindowListener(new WindowAdapter()
          {
                  public void windowClosing(WindowEvent e)
                  {
                          dispose();
                          }
                  });
  show();

  try
  {
   if(title.equals("Compile File"))
   {
          Compile_File();
```

```
        }
        if(title.equals("Run File"))
        {
                Run_File();
        }
        if(title.equals("Run Applet"))
        {
                Run_Applet();
        }
       }catch(Exception ddfd){}
     }
       public void Compile_File() throws Exception
       {
                rt = Runtime.getRuntime();
                prc = null;
                String[] cmd = new String[2];
                cmd[0] = "javac";
                FileDialog fd = new FileDialog(this,"Compile File ...",0);
                fd.show();

                if ((fd.getDirectory() != null) && (fd.getFile() != null))
                {
                 String name=fd.getFile();
                 int len=name.length();
                 String subs=name.substring((len-4),len);
                 System.out.println(subs);
                 if(!subs.equals("java"))
                 {
                       lab.setText("Please Select File with extension
'.java'");
                       b1.setEnabled(true);
                       fl=2;
                 }
                 else
                 {
                        cmd[1]=fd.getDirectory() + fd.getFile();

                        lab.setText("Compiling Application Named : "+cmd[1]);
                        System.out.println("Compiling application named : " +
cmd[1]);
                        prc = rt.exec(cmd);
                        prc.waitFor();
                        Display_Error();
                        }
                }
            }
        public void Run_File() throws Exception
        {
                rt = Runtime.getRuntime();
                prc = null;
                FileDialog fd = new FileDialog(this,"Run File ...",0);
                fd.show();
               String fname=fd.getFile();
               int len=fname.length();
               fname=fname.substring((len-5),len);

                if ((fd.getDirectory() != null) && (fd.getFile() != null))
```

```
                {

                 if(!fname.equals("class"))
                 {
                       lab.setText("Select File with extension '.class'");
                       b1.setEnabled(true);
                       fl=4;
                 }
                 else
                 {
                        String[] cmd = new String[2];
                        cmd[0] = "java";
                        StringTokenizer st = new
StringTokenizer(fd.getFile(),".");
                        cmd[1]=st.nextToken();
                       lab.setText("Running Application Named : "+cmd[1]);
                        System.out.println("Running Application named : " +
cmd[1]);
                        prc = rt.exec(cmd);
                        prc.waitFor();
                        Display_Output();
                  }
                }
        }

        public void Run_Applet() throws Exception
        {
                rt = Runtime.getRuntime();
                prc = null;
                FileDialog fd = new FileDialog(this,"Run Applet File ...",0);
                fd.show();
               String fname=fd.getFile();
               int len=fname.length();
               fname=fname.substring((len-4),len);

                if ((fd.getDirectory() != null) && (fd.getFile() != null))
                {

      if(!((fname.equals("java"))||(fname.equalsIgnoreCase("html"))))
                  {
                       lab.setText("Please Select File with extension
 '.java'");
                       b1.setEnabled(true);
                       fl=5;
                  }
                  else
                  {
                        String[] cmd = new String[2];
                        cmd[0] = "AppletViewer";
                        cmd[1]=fd.getFile();
                       lab.setText("Running Application Named : "+cmd[1]);
                        System.out.println("Running Applet named : " + cmd[1]);
                        prc = rt.exec(cmd);
                        prc.waitFor();
                        Display_Output();
                   }
                }
```

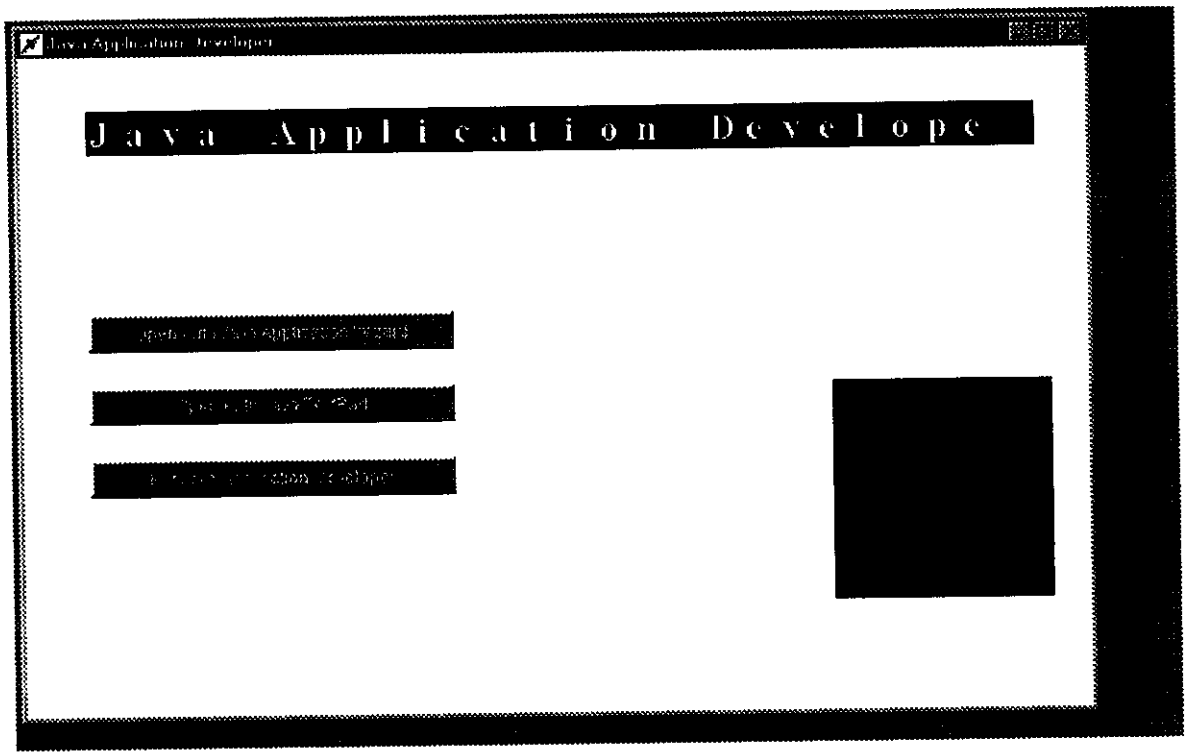
```
        }
       public void Display_Error() throws Exception
       {
               InputStream fi = prc.getErrorStream();
               byte[] buff = new byte[fi.available()];
               fi.read(buff,0,buff.length);

               if (buff.length != 0)
                {
                      fl=3;
                      b1.setEnabled(true);
                      b2.setEnabled(true);
                      lab.setText("Application is Compiled With Errors or
Warnings");
                      lab2.setText("For Details Click 'OK' else 'CANCEL'");
                      errormessage=new String(buff,0,buff.length);
                      System.out.println(new String(buff,0,buff.length));
                 }
               else
                {
                      fl=1;
                      lab.setText("Application Successfully Compiled...");
                      b1.setEnabled(true);
                      System.out.println("Successfully Compiled ...");
                 }
               }

        public void Display_Output() throws Exception
        {
               InputStream fi = prc.getInputStream();
               byte[] buff = new byte[fi.available()];
               fi.read(buff,0,buff.length);
               dispose();
               System.out.println("Inside Output ...");
               if (buff.length != 0)
                       System.out.println(new String(buff,0,buff.length));
               else
                       System.out.println("Successfully executed ...");
               }

       public void actionPerformed(ActionEvent e)
       {
            if(e.getSource()==b1)
            {
                 if(fl==1)
                      dispose();
                 if(fl==2)
                 {
                   try
                   {
                       Compile_File();
                   }catch(Exception ddd){}
                 }
                 if(fl==3)
                 {
                       b1.setEnabled(false);
```

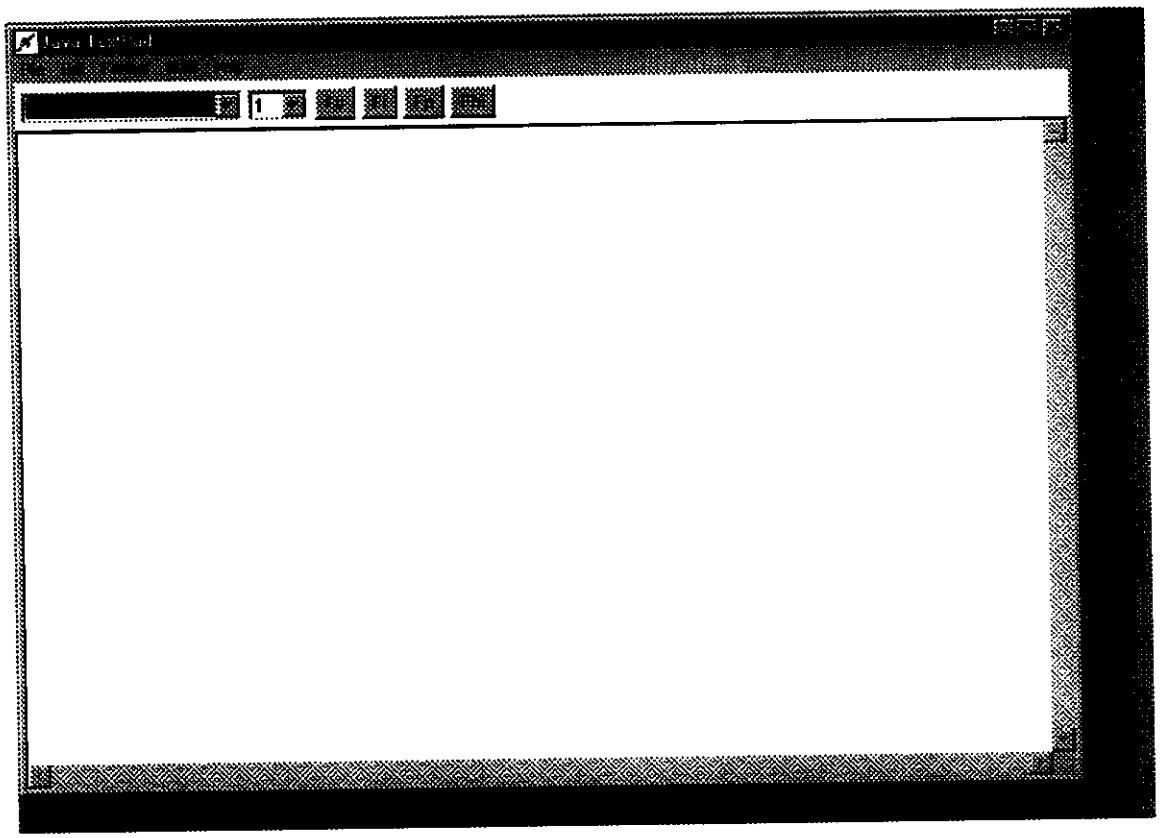
```
                    dispose();
                    ShowError serr=new ShowError(errormessage);
                    serr.show();
                }
                if(fl==4)
                {
                  try
                  {
                      Run_File();
                  }catch(Exception ddd){}
                }
                if(fl==5)
                {
                  try
                  {
                      Run_Applet();
                  }catch(Exception ddd){}
                 }

            }
            if(e.getSource()==b2)
                  dispose();
            }
}

class ShowError extends Frame
{
      TextArea ta=new TextArea(20,90);
      ShowError(String err)
      {
            setTitle("Error Details");
            setLayout(null);
            setLocation(1,1);
            setSize(700,500);

                Toolkit tk2 = getToolkit();
                setIconImage(tk2.getImage("icon.gif"));

            ta.setForeground(Color.blue);
                ta.setFont(new Font("TimesRoman",Font.BOLD,16));
            ta.setBounds(2,20,690,475);
            ta.setEditable(false);
            add(ta);
            ta.setText(err);
addWindowListener(new WindowAdapter()
{
      public void windowClosing(WindowEvent e)
      {
            dispose();
      }
});

      }
}


      public void itemStateChanged(ItemEvent e)
```

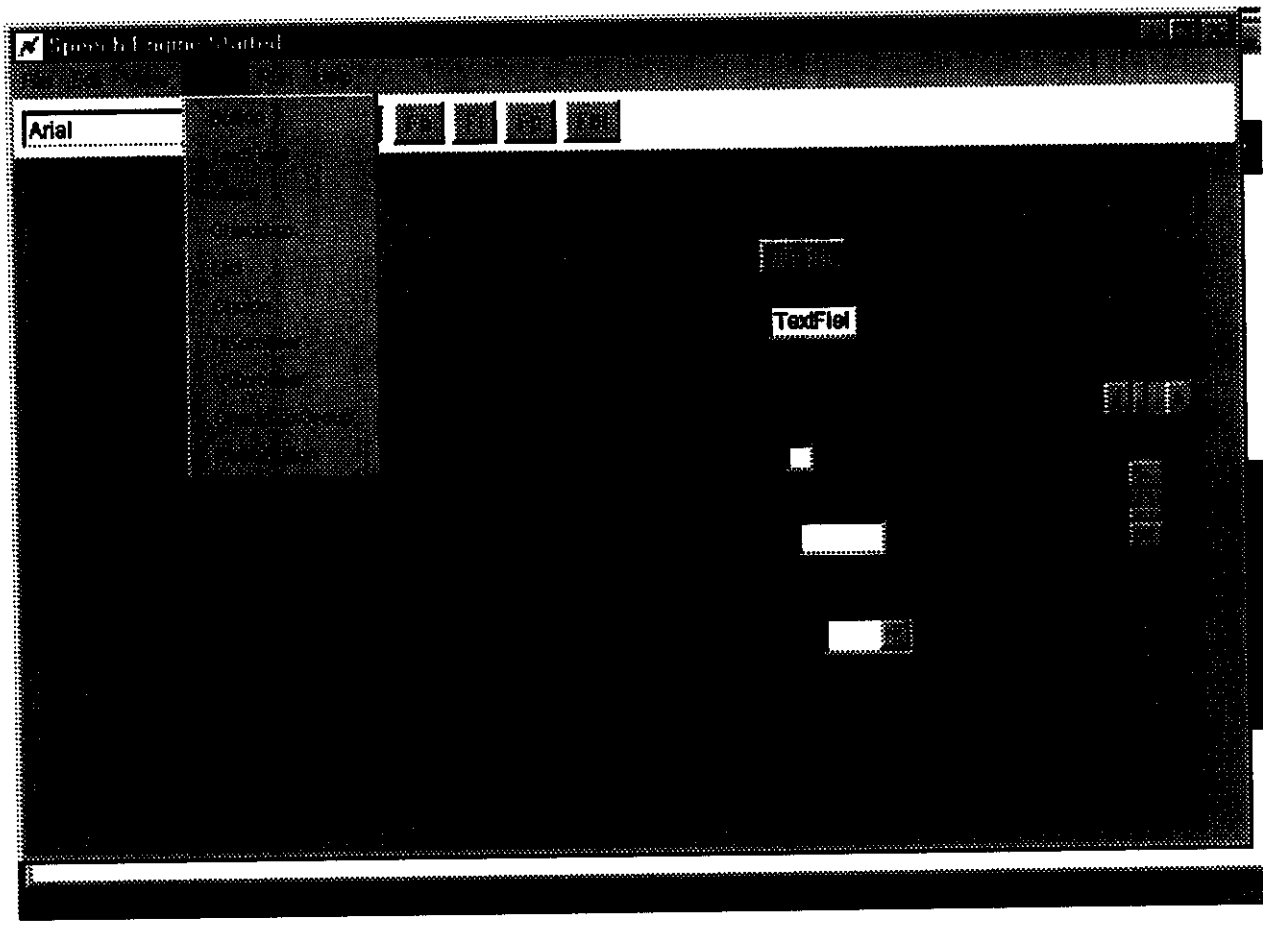
```
{
      classname=(String)e.getItem();
      }


public void actionPerformed(ActionEvent e)
{
      if(e.getSource()==But)
      {
         ta.setText("");

         try
         {
            classname="java."+classname;
            System.out.println(classname);
            String dupli = classname;
            Class c = Class.forName(classname);
            Constructor constructors[] = c.getConstructors();

            ta.append("\t\tCLASS PATH : "+dupli);
            String fulidu="CLASS PATH : "+dupli;
            ta.append("\n\t\t");
            for(int tyh=0;tyh<fulidu.length();tyh++)
            ta.append("~");
            ta.append("\n\nConstructors : \n~~~~~~~~~~~~~~~" + "\n\n");
            for(int i=0;i<constructors.length;i++)
             {
                  classname = constructors[i].toString();
                  ta.append(classname + "\n");
             }
             ta.append("\n\n");

             Field  fields[] = c.getFields();
             ta.append("Fields : \n~~~~~~~~~" + "\n\n");
             for(int i=0;i<fields.length;i++)
             {
                   classname = fields[i].toString();
                     ta.append(classname + "\n");
             }
             ta.append("\n\n");

             Method methods[] = c.getMethods();
             ta.append("Methods : \n~~~~~~~~~~" + "\n\n");
             for(int i=0;i<methods.length;i++)
             {
                   classname = methods[i].toString();
                     ta.append(classname + "\n");
             }

          }catch(Exception ee){}

       }
  }

}

   public void itemStateChanged(ItemEvent e)
```

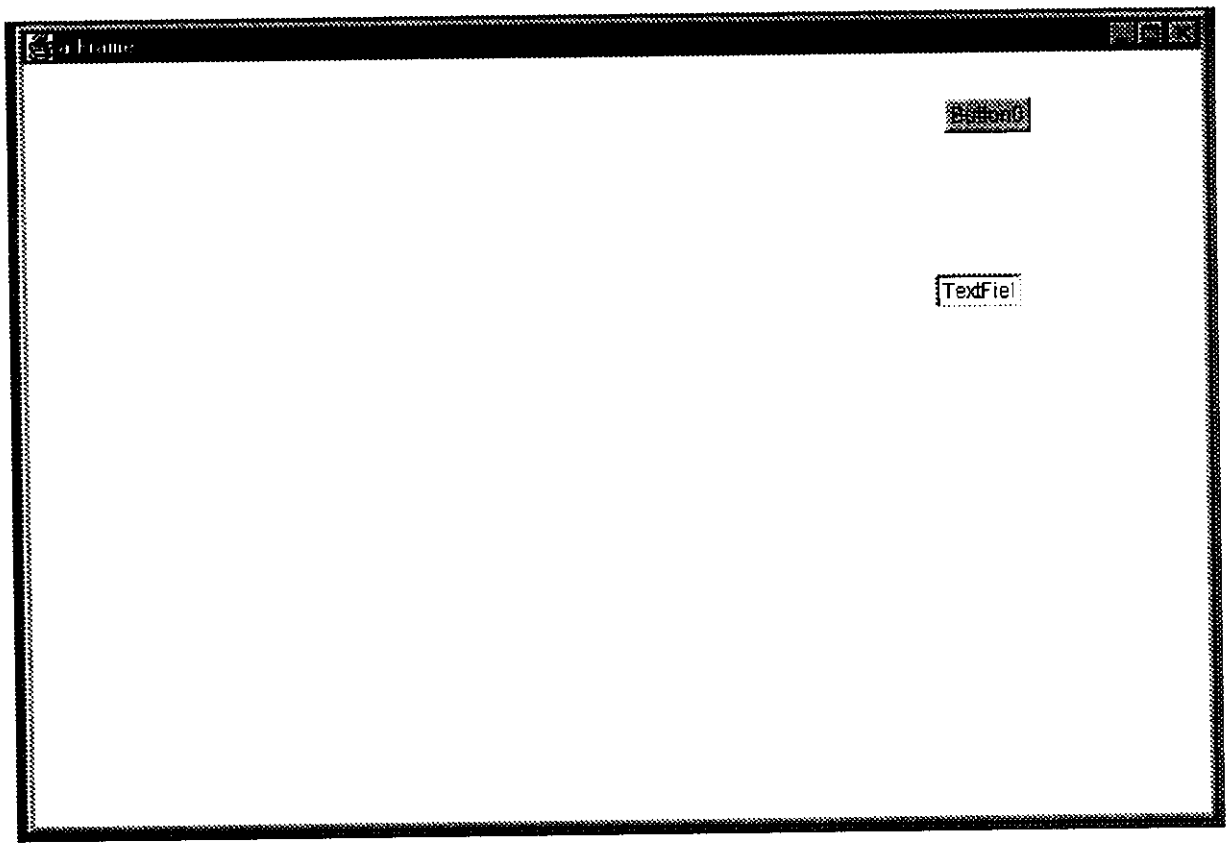
```
{
        classname=(String)e.getItem();
        }


    public void actionPerformed(ActionEvent e)
    {
        if(e.getSource()==But)
        {
           ta.setText("");

           try
           {
              classname="javax."+classname;
              System.out.println(classname);
              String dupli = classname;
              Class c = Class.forName(classname);
              Constructor constructors[] = c.getConstructors();

              ta.append("\t\tCLASS PATH : "+dupli);
              String fulidu="CLASS PATH : "+dupli;
              ta.append("\n\t\t");
              for(int tyh=0;tyh<fulidu.length();tyh++)
              ta.append("~");
              ta.append("\n\nConstructors : \n~~~~~~~~~~~~~~~" + "\n\n");
              for(int i=0;i<constructors.length;i++)
               {
                    classname = constructors[i].toString();
                    ta.append(classname + "\n");
               }
               ta.append("\n\n");

               Field  fields[] = c.getFields();
               ta.append("Fields : \n~~~~~~~~~" + "\n\n");
               for(int i=0;i<fields.length;i++)
               {
                     classname = fields[i].toString();
                       ta.append(classname + "\n");
               }
               ta.append("\n\n");

               Method methods[] = c.getMethods();
               ta.append("Methods : \n~~~~~~~~~~" + "\n\n");
               for(int i=0;i<methods.length;i++)
               {
                     classname = methods[i].toString();
                       ta.append(classname + "\n");
               }

           }catch(Exception ee){}

        }
    }
}
```

SAMPLE OUTPUTS


Java Application Develope




Arial
TextFiel


Button0
TextFiel