JAVA ADVANCED
AWT

INTRODUCTION: - AWT stands for Abstract window toolkit is an Application programming interface (API) for creating Graphical User Interface (GUI) in Java. It allows Java programmers to develop window-based applications. It was developed by heavily Sun Microsystems In 1995. It is heavy-weight in use because it is generated by the system's host operating system. It contains a large number of classes and methods, which are used for creating and managing GUI.
AWT provides various components like button, label, checkbox, etc. used as objects inside a Java Program. AWT components use the resources of the operating system, i.e., they are platform-dependent, which means, component's view can be changed according to the view of the operating system. The classes for AWT are provided by the Java.awt package for various AWT components.

CREATE PROJECT: - create a project in netbean software to use awt

- Open netbean application, open file menu in tool bar.
- Click on new project to create new project, it will pop up new project window.
- Select java with Ant in categories and java application in project then click on next.
- Set project name and location then click on finish.
- It will open new project to work with some premade code.

Java Ant: - Apache Ant (Another Neat Tool) is an open source project started by Apache Software Foundation. Ant is a Java library and a software tool used for automate software build processes such as compile, run, test and assemble Java application.

Run project: -

- When project created there will be class file with public class named as project name
- Then write code for see output in main function of class.
  Ex: - System.out.println("testing");
- Then click on 'run' in menu bar then 'run project' to see output.
- It will show output in output window at bottom.
- If output did not show then click on 'clean and build project' in run menu.
- Then again run project.

CREATING GUI: - for create GUI application, it will need to import awt package
LABEL: - it is used for show text in frame.
TEXTFIELD: - it is class that used for take input from user.
BUTTON: - it is class to display button and perform some action on button click.

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

```
class myframe extends Frame implements ActionListener{

   Label l1,l2;
   TextField t1;
   Button b1;

   myframe()
   {
      setLayout(null);

      l1= new Label("Name:- ");
      l1.setBounds(10,30,50,20);
      l1.setBackground(Color.red);
      add(l1);

      t1= new TextField();
      t1.setBounds(60,30,100,20);
      add(t1);

      b1= new Button("show");
      b1.setBounds(30, 60, 50, 20);
      b1.addActionListener(this);
      add(b1);

      l2=new Label("Hello ");
      l2.setBounds(10,90,150,20);
      add(l2);
   }

   @Override
   public void actionPerformed(ActionEvent e) {

      String s=t1.getText();
      l2.setText("Hello "+s);
   }
}

public class Main {

   public static void main(String[] args) {

      myframe f= new myframe();
      f.setSize(200,200);
      f.setVisible(true);
   }
}
```

Handle multiple buttons action: -

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

class frame extends Frame implements ActionListener
{
   Label l1,l2,l3;
   TextField t1,t2,t3;
   Button b1,b2,b3,b4;

   frame()
   {
      setLayout(null);

      l1=new Label("No1 ");
      l1.setBounds(10,30,50,20);
      add(l1);

      t1=new TextField();
      t1.setBounds(70,30,100,20);
      add(t1);

      l2=new Label("No2 ");
      l2.setBounds(10,60,50,20);
      add(l2);

      t2=new TextField();
      t2.setBounds(70,60,100,20);
      add(t2);

      b1=new Button("ADD");
      b1.setBounds(10,90,50,20);
      b1.addActionListener(this);
      add(b1);

      b2=new Button("SUB");
      b2.setBounds(70,90,50,20);
      b2.addActionListener(this);
      add(b2);

      b3=new Button("MUL");
      b3.setBounds(130,90,50,20);
      b3.addActionListener(this);
      add(b3);
```

```
        b4=new Button("DIV");
        b4.setBounds(190,90,50,20);
        b4.addActionListener(this);
        add(b4);

        l3= new Label("ANS:- ");
        l3.setBounds(10,120,50,20);
        add(l3);

        t3=new TextField();
        t3.setBounds(70,120,100,20);
        add(t3);
    }

    @Override
    public void actionPerformed(ActionEvent e)
    {
        int n1=Integer.parseInt(t1.getText());
        int n2=Integer.parseInt(t2.getText());
        int n3=0;

        if(e.getSource()==b1)
        {
            n3=n1+n2;
        }
        if(e.getSource()==b2)
        {
            n3=n1-n2;
        }
        if(e.getSource()==b3)
        {
            n3=n1*n2;
        }
        if(e.getSource()==b4)
        {
            n3=n1/n2;
        }

        t3.setText(String.valueOf(n3));
    }
}

public class Main {

    public static void main(String[] args) {
        frame f= new frame();
        f.setSize(260,180);
```

```
        f.setVisible(true);
    }
}
```

CHECKBOX: - it is used for display checkbox and performs operations with checkbox.

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

class frame extends Frame implements ActionListener
{
    Checkbox c1,c2;
    Button b1;
    Label l1;
    frame()
    {
        setLayout(null);

        c1=new Checkbox("ch1");
        c1.setBounds(10,30,50,20);
        add(c1);

        c2=new Checkbox("ch2");
        c2.setBounds(10, 60, 50, 20);
        add(c2);

        b1= new Button("show");
        b1.setBounds(10,90,50,20);
        b1.addActionListener(this);
        add(b1);

        l1=new Label();
        l1.setBounds(10,120, 60, 20);
        add(l1);
    }

    @Override
    public void actionPerformed(ActionEvent e)
    {
        int c=0;
        if(c1.getState()==true)
        {
            c=c+1;
        }

        if(c2.getState()==true)
        {
```

```
        c=c+1;
      }

      l1.setText("check = "+c);
   }
}

public class Main {

   public static void main(String[] args) {
      frame f = new frame();
      f.setSize(300,200);
      f.setVisible(true);
   }
}
```

SWING

INTRODUCTION: - Java Swing is a very powerful GUI toolkit for Java applications, introduced as an extension of AWT. Unlike AWT, Swing provides a rich set of components and features that are all implemented in Java. While AWT components are based on the native platform, Swing components are simply entirely written in Java which provides a consistent look and feel across different platforms. And with this feature Swing simply becomes a very popular choice for cross-platform applications.

Java Swing is simply optimized for performance and provides efficient rendering of complex UIs that makes it suitable for applications requiring a high level of responsiveness. It offers very good performance for applications with moderate to high complexity.

SWING COMPONENT: - these are same as AWT component class but declare with J before every component class. Ex. Frame is in swing as JFrame. Button as JButton, Label as JLabel, TextField as JTextField, other methods in Swing as same as AWT, ex, add(), setLayout(), setVisible(), setSize() etc. use isSelected() method for CheckBox in swing instead of getState() to check checkbox is selected or not.

RADIOBUTTON: - there are used for select only one option from multiple choices.

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

class frame extends JFrame implements ActionListener{

   JRadioButton r1,r2;
   ButtonGroup bg1;
   JButton b1;
```

```
JLabel l1,l2;
JCheckBox c1;

frame()
{
   setLayout(null);

   r1=new JRadioButton("Male");
   r1.setBounds(10,10,80,20);
   add(r1);

   r2=new JRadioButton("Female");
   r2.setBounds(10,40,80,20);
   add(r2);

   bg1=new ButtonGroup();
   bg1.add(r1);
   bg1.add(r2);

   b1=new JButton("show");
   b1.setBounds(25,70,70,20);
   b1.addActionListener(this);
   add(b1);

   l1=new JLabel();
   l1.setFont(new Font("Times New Roman",Font.BOLD,10))
   l1.setBounds(10,100,120,20);
   add(l1);

   l2=new JLabel();
   l2.setFont(new Font("Times New Roman",Font.BOLD,10))
   l2.setBounds(10,130,120,20);
   add(l2);

   c1= new JCheckBox("Married");
   c1.setBounds(140,10,80,20);
   add(c1);
}

@Override
public void actionPerformed(ActionEvent e)
{
   l1.setText("Select Gender");
   if(r1.isSelected())
   {
      l1.setText("You are Male");
   }
```

```
        if(r2.isSelected())
        {
           l1.setText("You are Female");
        }
        if(c1.isSelected())
        {
           l2.setText("You are Married");
        }
        else
        {
           l2.setText("You are Unmarried");
        }
    }
}

public class Main {
    public static void main(String[] args) {
        frame f=new frame();
        f.setSize(300,200);
        f.setVisible(true);
        f.getContentPane().setBackground(Color.yellow);
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

WEB APPLICATION

CREATE PROJECT: -

- Open netbean application, open file menu in tool bar.
- Click on new project to create new project, it will pop up new project window.
- Select java web in java with Ant in categories and web application in project then click on next.
- Set project name and location then click on next.
- Then click on add in server to add server if server is not added.
- Then click on finish it will create new web project.

Adding server: -

- After click on add in server it will pop up window for Add Server Instance.
- Choose server as 'Apache Tomcat' and set name then click on next
- Select server location from browse.
- Select server location of apache tomcat, download apache server and give server path or choose path from xampp folder if xampp is install.
  Path: - C:\xampp\tomcat
- Set username and password then click on finish.

Run project: -

- First start server from services window, if services window is display, then in menu bar click on window tab, and select services window.
- In services window expand servers folder then select server.
- Then right click on server and start server.
- It will pop up Authentication window enter username and password then click ok.
- Then select project then Web Pages and double click on index.html file to open in edit.
- Then click on run project it will open website in browser.

SERVLET: - Java servlets are the Java programs that run on the Java-enabled web server or application server. They are used to handle the request obtained from the web server, process the request, produce the response, and then send a response back to the web server.

Properties: -

- Servlets work on the server side.
- Servlets are capable of handling complex requests obtained from the web server.

Need: - The Server-Side Extensions are nothing but the technologies that are used to create dynamic Web pages. Actually, to provide the facility of dynamic Web pages, Web pages need a container or Web server. To meet this requirement, independent Web server providers offer some proprietary solutions in the form of APIs (Application Programming Interface).
These APIs allow us to build programs that can run with a Web server. In this case, Java Servlet is also one of the component APIs of Java Platform Enterprise Edition (nowdays known as – 'Jakarta EE') which sets standards for creating dynamic Web applications in Java.

ADD java servlet: -

- First open java web application existing project or create new project.
- Then right click on that project in project window, in 'New' option choose 'Servlet', it will pop up 'New Servlet' window.
- Enter class name and package name for servlet and click on next.
- Then checked checkbox for add information to deployment descriptor in web.xml file.
- Enter new Servlet name and url pattern if want or not. Then click on finish, it will create new servlet.

Web.xml file will create in Project -> Web Pages -> WEB-INF folder.

Servlet code: -

```
package serv;

import java.io.IOException;
import java.io.PrintWriter;
import jakarta.servlet.ServletException;
import jakarta.servlet.http.HttpServlet;
```

```
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;

public class serv1 extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and <code>POST</code>
     * methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        try (PrintWriter out = response.getWriter()) {
            /* TODO output your page here. You may use following sample code. */
            out.println("<!DOCTYPE html>");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet serv1</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Servlet serv1 at " + request.getContextPath() + "</h1>");
            out.println("</body>");
            out.println("</html>");
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the
left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
```

```
    /**
     * Handles the HTTP <code>POST</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     *
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

Run servlet code: -

Index.html: -

```
<!DOCTYPE html>

<html>
    <head>
        <title>Hello</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
    </head>
    <body>
        <form action="serv1" method="get">
            <input type="text" name="id">
            <input type="submit">
        </form>
    </body>
</html>
```

- Servlet take request from both get and post methods.

- Assign servlet name to action in form tag.
- When user clicks on submit button then it will redirect to servlet webpage.

Take User input: -

Html code: -

```
<form action="serv1" method="get">
        <input type="text" name="id">
        <input type="submit">
</form>
```

Java servlet code: -

```
out.println("ID= "+request.getParameter("id"));
```

- getParameter method is used for take input from html to servlet.
- It is method from HttpServletRequest class.
- It returns string value that enters in input tag and take name of input tag as string parameter.

Perform operation with user input: -

Html code: -

```
<form action="serv2" method="post">
    NO1 <input type="text" name="n1"><br><br>
    NO2 <input type="text" name="n2"><br><br>
    <input type="submit" >
</form>
```

Java servlet code: -

```
out.println("<body>");
int n1=Integer.parseInt(request.getParameter("n1"));
int n2=Integer.parseInt(request.getParameter("n2"));
int n3=n1+n2;
out.println("<h1>Addition = "+n3+"</h1>");
out.println("</body>");
```

- It takes two inputs from user and converts them into integer.
- Then perform addition and display it in html with out.println() method.

Handle multiple buttons: -

Html code: -

```
<form action="serv4" method="post">
   NO1 <input type="text" name="n1"><br><br>
   NO2 <input type="text" name="n2"><br><br>
   <input type="submit" name="op" value="add">
```

```
   <input type="submit" name="op" value="sub">
   <input type="submit" name="op" value="mul">
   <input type="submit" name="op" value="div">
</form>
```

Java servlet code: -

```
out.println("<body>");
int n1=Integer.parseInt(request.getParameter("n1"));
int n2=Integer.parseInt(request.getParameter("n2"));
String op=request.getParameter("op");
int n3 = 0;
if(op.equals("add"))
{
   n3=n1+n2;
}
if(op.equals("sub"))
{
   n3=n1-n2;
}
if(op.equals("mul"))
{
   n3=n1*n2;
}
if(op.equals("div"))
{
   n3=n1/n2;
}
out.println("Result = "+n3);
out.println("</body>");
```

- Take multiple buttons in html with same name and different value.
- In servlet check value of name with request.getParameter() and store it in string object.
- Compare that object with other string with equal method of string and if it is equal then set corresponding operation.

  Ex: -

  ```
  if(op.equals("add"))
  {
     n3=n1+n2;
  }
  ```

CHECKBOX: - it is used for take multiple choices from user.

Html code: -

```
<form action="serv3" method="post">
      <input type="checkbox" name="sub" value="c">C Language<br>
      <input type="checkbox" name="sub" value="c++">C++ Language<br>
      <input type="checkbox" name="sub" value="java">JAVA Language<br>
```

```
      <input type="checkbox" name="sub" value="python">PYTHON Language<br>
      <input type="submit" value="fees"><br>
</form>
```

Java servlet code: -

```
out.println("<body>");
String sub[]=request.getParameterValues("sub");
int t=0;
if(sub!=null)
{

   for(int i=0;i<sub.length;i++)
   {
      if(sub[i].equals("c"))
      {
        t+=2000;
      }
      if(sub[i].equals("c++"))
     {
         t+=4000;
     }
     if(sub[i].equals("java"))
     {
        t+=20000;
     }
     if(sub[i].equals("python"))
     {
        t+=10000;
     }
  }
}
out.println("Total Fees = "+t);
out.println("</body>");
```

- Take multiple checkbox in html with same name and different value.
- In servlet take all values of name with request.getParameterValue() and store it in string array.
- Compare that array element with other string with ‘equals’ method of string and if it is equal then set corresponding operation.