PERIYAR INSTITUTE OF DISTANCE EDUCATION
(PRIDE)

PERIYAR UNIVERSITY

SALEM - 636 011.

B.Sc. COMPUTER SCIENCE
THIRD YEAR
PRACTICAL – IV : (PROGRAMMING IN JAVA)

Prepared by:
Prof N. RAJENDRAN, M.C.A.,M.Phil.,
HOD of Computer Science,
Vivekanandha College of Arts and Sciences for Women,
Elayampalayam, Tiruchengode,
Namakkal (District) – 637 205.

FOREWORD

"In human affairs we have reached a point where the problems that we must solve are no longer solvable without the aid of computers. I fear not computers but the lack of them"

Issac Asimov

Dear Students,

The importance of computers is felt in every field and it has become an integral part of our society. This booklet has been planned in a way that an in depth learning about practical programs in **Programming in JAVA and JAVA SCRIPT.**

The list of practical in JAVA SCRIPT has 6 programs and JAVA has 7 programs. Mainly this booklet focused the simple concepts of JAVA and JAVA SCRIPT and how to implement it using JAVA and JAVA SCRIPT programs with output. It simply explained the programs with procedure and definitions, concepts and examples of JAVA and JAVA SCRIPT concept, which will implement in program.

In JAVA SCRIPT, the first program Dynamic Web Page creation. This program leads to change the background and foreground color of document at run time. The second program HTML Page with JAVA SCRIPT. It is a simple program to fine factorial of range of numbers.

The third program says the concept of Event handling in JAVA SCRIPT. It displays the odd and even number list in rage of number based on events. The fourth program deals the Document and Window object. It performs the biggest number among three numbers. The fifth program gives simple introduction about creation of JAVA SCRIPT clock. It produces the date time such as hour, minute and seconds of movement.

Finally the sixth program says the introduction and program using forms in JAVA SCRIPT. This program gets the user name and password and verifies it whether it is correct.

In JAVA, the first program simple applet creation. It displays a human face-using applet. The second program says about the concept of class and objects in java. Simply it produces the area of circle using object and class.

The third program explained about the concept of Inheritance and Interface. This program displays the student details using multilevel inheritance and interface. The fourth program deals the concept of array in java. It displays the given number in ascending order.

The fifth program says about the concept Threads and Multithreads. It executes the threads prime number, factorial and fibionacci series simultaneously.

The sixth program AWT package. It displays multiple objects using Graphics class methods. The seventh program used to implement the concept of I/O package. Simply it gets the string from console device and writes it on disk. Again read it from disk and display it on console device.

All the above said **Programming in JAVA and JAVA SCRIPT** practical programs in SIM pattern of this booklet have been prepared **Professor N.RAJENDRAN, M.C.A., M.Phil.,** to make your practical learning's much easier while going through it.

PRIDE would be happy in you could make use of this learning material to enrich your knowledge and skills to serve the society.

B.Sc. COMPUTER SCIENCE

THIRD YEAR

PRACTICAL - IV : (PROGRAMMING IN JAVA AND JAVA SCRIPT)

JAVA SCRIPT Programming List

1. To develop a Dynamic Web Page using Java Script.
2. Program to create an HTML page with Java Script.
3. Program to create an HTML page using Event Handling.
4. Develop a Java Script using document and window object.
5. Program to create a Java Script clock.
6. Program to work with Forms Using Java Script.

JAVA Programming List

7. Program to create a simple applet and application
8. Using java class and objects
9. Using java Inheritance and Interface
10. Using Arrays in java
11. Using Threads and Multithreads
12. Using AWT package
13. Using I/O package

JAVA Script

Introduction

JAVASCRIPT is used to developing a dynamic web page on the Internet. JavaScript came about as a joint effort between Netscape Communications Corporation and Sun Microsystems, Inc. JavaScript is an object-based, client-side scripting language that we can use to make Web Pages more dynamic.

Object Based

Object Based means that JavaScript can use items called objects. However the objects are not class based (meaning no distinction is made between a class and an instance); instead, they are just general objects.

Client Side

Client side means that JavaScript runs in the client (software) that the viewer is using, rather than on the Web server of the site serving the page. In this case, the client would be a Web browser.

Program 1 : To develop a Dynamic Web Page using Java Script.

Procedure

Step 1 : Start html program.

Step 2 : In body section include java script coding.

Step 3 : Script has four function.

i. GotFocus() – When we focus the document text it change the background color of document text

ii. LostFocus() – When we move the cursor from focused text it return back to previous state.

iii. dblClick() – When we click on page in first time, it change the background color of document area.

iv. Click() – It change the foreground color of document text.

Step 4 : Html body tag call all the functions using event handlers

Step 5 : It focus the dynamic change on the web page.

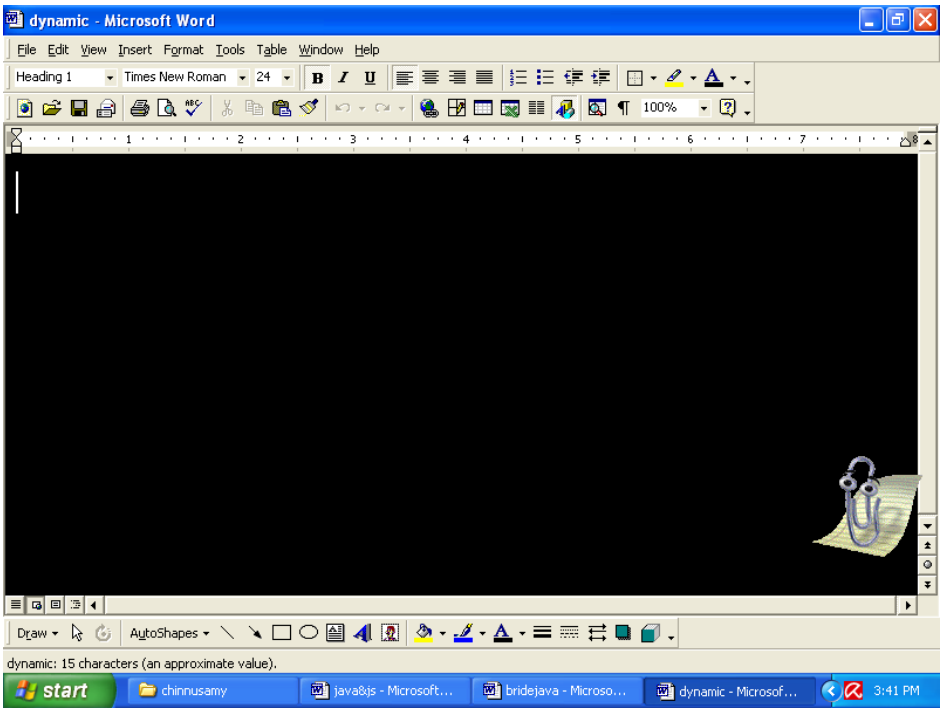
Dynamic Web Page Creation using Java Script

```
<html>
<head>
<script language="JavaScript">
 function gotFocus()
 {
  document.bgColor="#FF0600"
 }
 function lostFocus()
 {
  document.bgColor="#000000"
```

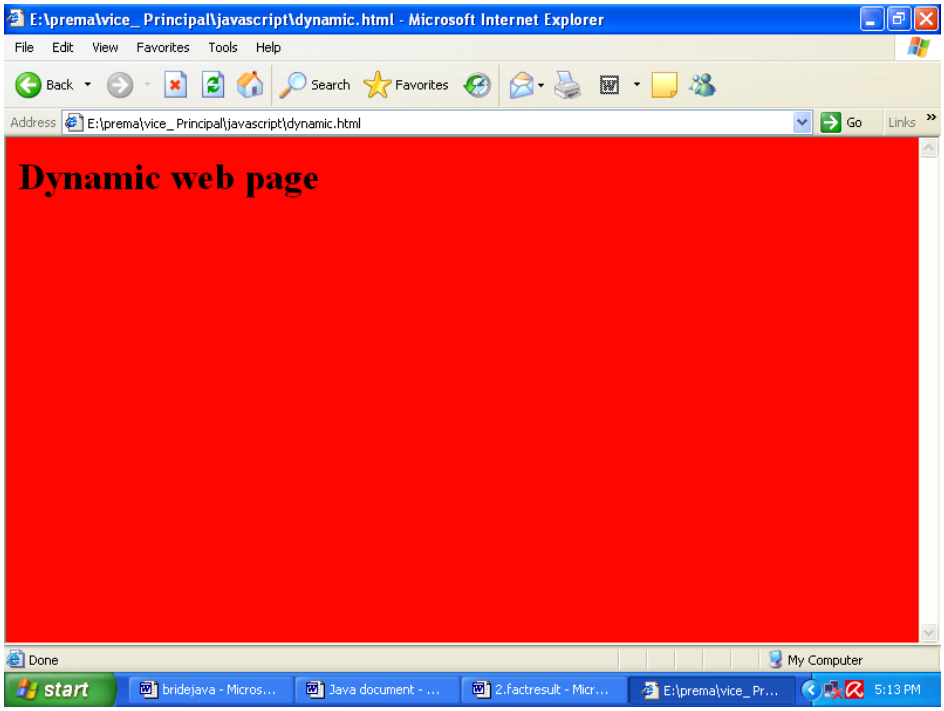
```
}
function click()
{
  document.foreColor="#FAB000"
}
function dblClick()
{
  document.bgColor="#FFFFFF"
}

</script>
</head>
<body onClick="click()" onDblClick()="dblClick()" onFocus="gotFocus()"
onBlur="lostFocus()" BGCOLOR="#000000">
<h1>Dynamic web page </h1>
</body>
</html>
```

Output

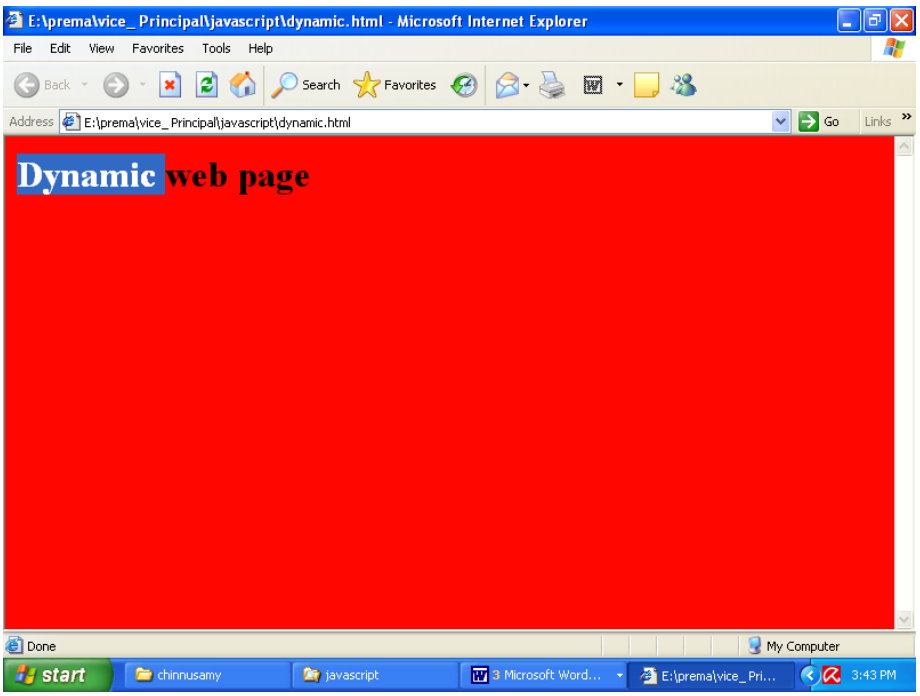
Before call the event dblClick



After call the event dblClick



After call the event onFocus



Program 2 : Program to create an HTML page with Java Script.

Introduction

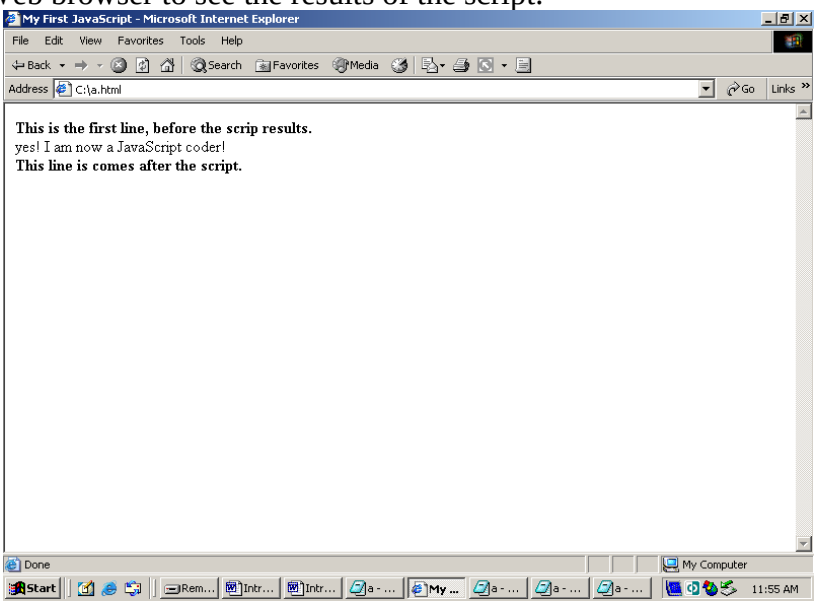
Inserting the Script into the HTML Document

To insert the script in the document, place a script between the <HEAD> and </HEAD> tags, or between the <BODY> and </BODY> tags. If the script is going to write the something directly to the page, it is normally placed in the BODY section of the document, where we want to results to appear.

Example:

```
<HTML>
<HEAD>
<TITLE>My First JavaScript</TITLE>
</HEAD>
<BODY>
<B>This is the first line, before the scrip results.</B>
<BR>
<SCRIPT   language=”JavaScript”>
<! –
document .write(“yes! I am now a JavaScript coder!”);
//- - >
</SCRIPT>
<BR>
<B> This line is comes after the script.</B>
<BODY>
</HTML>
```

Save this html document again. We should now be able to open the document in our Web browser to see the results of the script.



Procedure

Step 1 : Start html program

Step 2 : In head section insert the scripting function factorial() that it used to calculate the factorial value of n numbers.

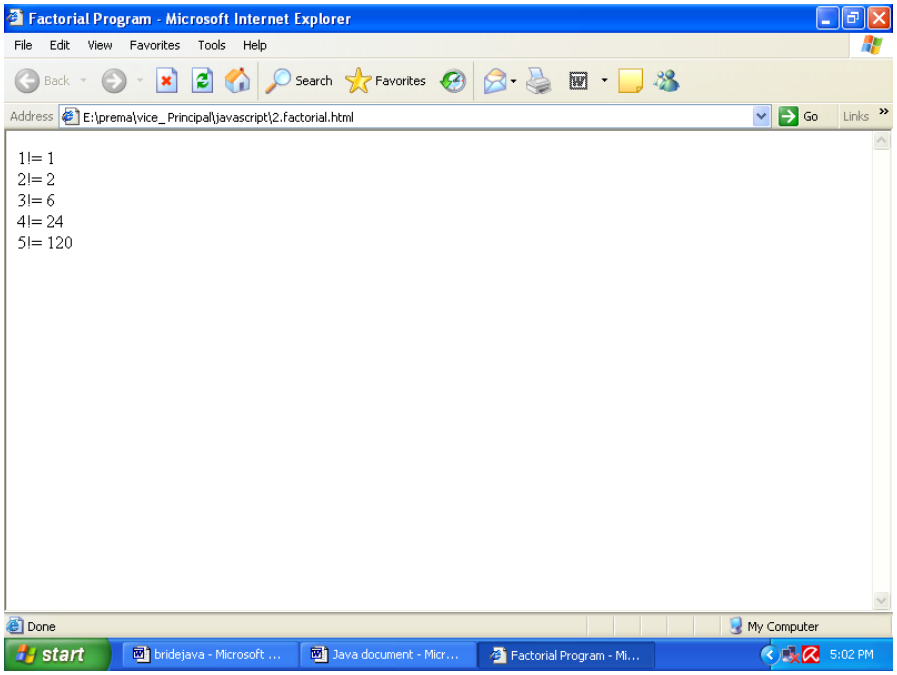
Step 3 : In body section insert the script that it used to call the factorial() function 'n' times and display the result.

Program

HTML page using Java Script

```
<html>
<head><title>Factorial Program </title>
<script language="JavaScript">
function factorial(n)
{
   var fact = 1
   for(i=1;i<=n;i++)
   {
     fact = fact * i
   }
   return fact
}
</script>
</head>
<body>
<script language="JavaScript">
  for(x=1;x<=5;x++)
  {
       document.write(x+"!"+"= "+factorial(x))
       document.write("<br>")
  }
</script>
</body>
</html>
```

Output


Factorial Program - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address E:\prema\vice_Principal\javascript\2.factorial.html
Go Links
1!= 1
2!= 2
3!= 6
4!= 24
5!= 120
Done
My Computer
start
bridejava - Microsoft ...
Java document - Micr...
Factorial Program - Mi...
5:02 PM

Program 3 : Program to create an HTML page using Event Handling.

Introduction

Event Handler

An event handler is a predefined JavaScript keyword that is used to handle an event on a Web page. Often, an event is something that happens when the viewer of the page performs some sort of action. This action may be a mouse click, the clicking of a button on the page, changing the contents of a form element, or moving the mouse over a link on the page.

Using Event Handlers

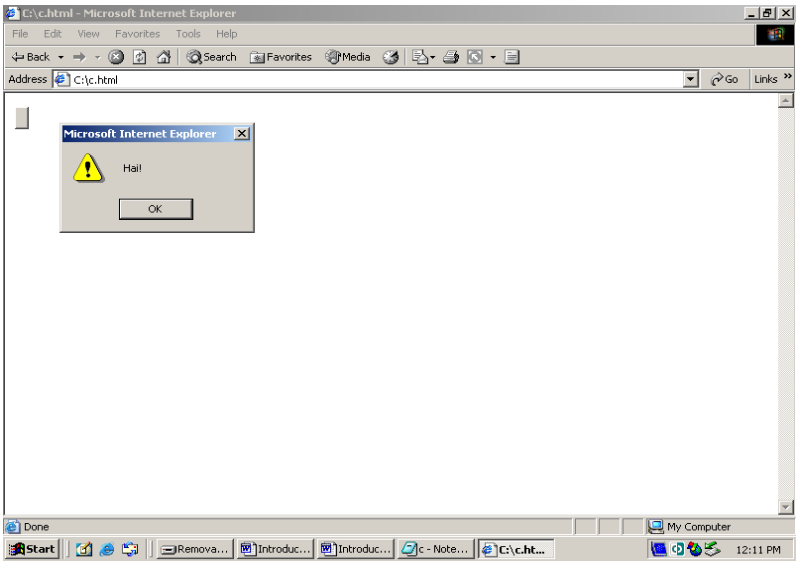
To use an event handler, we need to know the keyword for the event handler and where to place the event handler within the HTML code. To get an example, we will introduce the onClick event handler. The onClick event handler is used to make something happen when the viewer clicks a specific area of the document.

One of the valid places to be clicked is a form button. So, let's say we want to alert something to the viewer when he/she clicks a forms button.

Example:

```
<HTML>
<BODY>
<FORM>
<INPUT  type="button" onClick=window.alert("Hai!");">
</FORM>
</BODY>
</HTML>
```

When the viewer clicks this button, an alert will show the Hai message in the window object.



Event	Event Handler	Event Trigger
Click	onClick	Viewer click an area (such as button or form input area)
Mouseover	onMouseover	Viewer moves the mouse over a click
Mouseout	onMouseout	Viewer moves the mouse away from a click
Load	onLoad	Web page finishes loading
Unload	onUnload	Viewer leaves the current page
Focus	onFocus	Viewer gives the focus to something
Blur	onBlur	Viewer removes the focus from something
Change	onChange	Viewer changes the contents of a form Element
Submit	onSubmit	Viewer submits a form on a page
Abort	onAbort	Viewer stops the loading of an image
Dragdrop	onDragdrop	Viewer uses the drag-and-drop feature of an Operating system and drops somethlng in to the window
select	onSelect	User makes a selection in a form field.

Procedure

Step 1 : Start html program.

Step 2 : In head section insert scripting function odd_even().

Step 3 : The function display the odd number with in the range 15 and its sum and even numbers and its sum.

Step 4 : <Body> element call the function odd_even() at time loading.

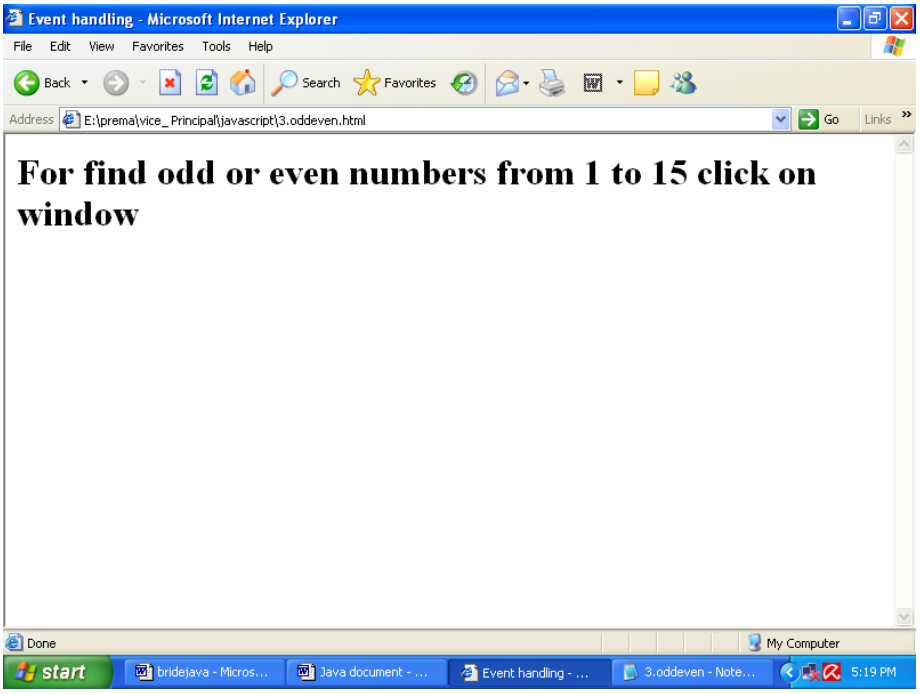
Program

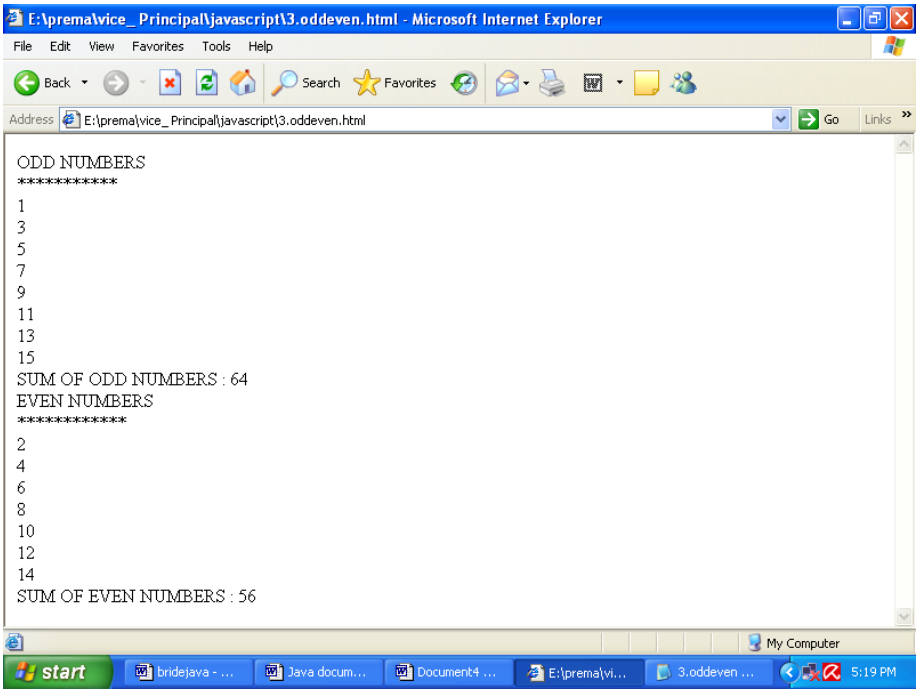
Event Handling

```
<html>
<head>
<title>Event handling</title>
<script language="JavaScript">
function odd_even(n)
{
  var odd_sum=0
  document.write("ODD NUMBERS"+"<br>")
  document.write("**********"+"<br>")
  for(i=1;i<=n;i+=2)
  {
```

```
    document.write(i+"<br>")
    odd_sum=odd_sum+i
  }
  document.write("SUM OF ODD NUMBERS  : "+odd_sum+"<br>")
  var even_sum=0
  document.write("EVEN NUMBERS"+"<br>")
  document.write("***********"+"<br>")
  for(i=2;i<=n;i+=2)
  {
    document.write(i+"<br>")
    even_sum=even_sum+i
  }
   document.write("SUM OF EVEN NUMBERS  : "+even_sum+"<br>")
}
</script>
</head>
<body onClick="odd_even(15)">
<h1>For find odd or even numbers from 1 to 15 click on window </h1>
</body>
</html>
```

Output




E:\prema\vice_ Principal\javascript\3.oddeven.html - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address E:\prema\vice_ Principal\javascript\3.oddeven.html
Go Links
ODD NUMBERS
****
1
3
5
7
9
11
13
15
SUM OF ODD NUMBERS : 64
EVEN NUMBERS
****
2
4
6
8
10
12
14
SUM OF EVEN NUMBERS : 56
My Computer
start
bridejava - ...
Java docum...
Document4 ...
E:\prema\vi...
3.oddeven ...
5:19 PM

Program 4 : Develop a Java Script using document and window object.

Introduction

Objects

An object is a way of modeling something real, even though the object is an abstract entity.

An introduction to the Document Object

The documents object is an object that is created by the browser for each new HTML page that is viewed. By doing this, JavaScript gives us access to a number of properties and methods that can affect the document in a number of ways.

As we know, we have been using the write() method of the document object. This method allows us to write a string of text into an HTML document.

Properties of Document Object

Property	Description
alinkColor	Returns the hexadecimal value of the active linkcolor of the document
anchors	An array of all the named anchors in the document
applets	An array of all of the java applets in a document
bgcolor	Returns the hexadecimal value of the background color of the document
cookie	Used to set JavaScript cookies in a document
domain	Returns the domain name of the server for the document

Methods of the Document Object

Open()

The open() method which allows us to open a new document and create its contents entirely with document.write() or document.writeln() statements.

Close()

The close() method which allows us to closes a new document that has been opened with open() method

Write()

The write() method which allows us to write a string of text into an HTML document.

Writeln()

The writeln() method which allows us to write a string of text into an HTML document but ends the line with a JavaScript new line character.

An Introduction to Window Object

Properties of window object

Closed property

The closed property which holds the value based on whether or not a window has been closed.

If (windowname.closed)

Defaultstatus property

It defines the default message displayed in the status bar.

<body onLoad="window.defaultStatus="welcome";>

"welcome" message is shown in the status bar.

Length property

The length property tells us how many frames are in a window.

window.frames.length

Location property

It holds the current URL of the window.

Name property

This property holds the name of the current window and also enables you to give a window a name.

```
<script language="JavaScript">
 document.write("this window is name"+window.name);
</script>
```

parent property

It is only used when there are frames on a page. It enables us to access the parent frame set of the current frame.

Self property

It is used to access the properties of the current window.

Status property

This contains the value of the text set in the status bar of the window.

```
<form>
    <input type="button" value="click for status message"
      onClick="window.status="hello";">
</form>
```

This changes the contents of the status bar on the click of a button.

Opener property

This is used to reference the window that opened the current window.

Top property

It is used to access the top window out of all the frame sets that contains

the current frame.

Methods of window object

The alert() method

The alert method is a method we have use extensively in earlier modules in our example scripts. This pops up a message to the viewer, and the viewer has to click an OK button to continue.

window.alert("Hi there!");

This would just give the viewer the "Hi there!" message we provided as an alert on the screen.

The confirm() Method

The confirm() method can be used to give the viewer a change to confirm or cancel an action. This method returns a Boolean value of true or false, so its result is often assigned to a variable when it is used.

The following is the syntax for assigning the value to a variable:

var varname=window.confirm("Your Message");

The find() Method

The find method can be used to let the viewer find the certain bit of text on your page. It tells the browser to use its built-in Find utility and enables the viewer to type in what to look for on the page.

For example, if we wanted to create a button for viewers to click when they want to find something on our page, we could use the following code:

```
<FORM>
<INPUT type="button" value="Click to find Text"
 onClick="window.find();">
</FORM>
```

This pops up the Find dialog box in the browser and enables the viewer to search for text within the page.

The print() Method

The print() method enable the viewer to print the current window. When this method is called, it should bring up the viewer's print dialog box so the printer settings can be set by the viewer to print the document.

To use it, we could create a button that enables the viewer to print the page being viewed. The following code creates the button:

```
<FORM>
<INPUT type="button" value="Click to print page" on
Click="window.print();">
</FORM>
```

The open() method

The open() method is the method that enables us to open a new window with JavaScript. This method takes in three parameters, and the third parameter sets a number or options that the window may need.

window.open("URL",name","attribute1=value,attribute2=value");

The first parameter shown as URL is replaced with URL of the HTML document to be opened in the new window. The name of the parameter is replaced with the name we wish to give to the window. The last parameters enable us to add attributes for the new window.

window.open("newpage.htm",my_window");

This would open a window with the contents of the HTML document newpage.htm and name the window my window;

The close() method

The close() method is used to close a window

Example:

<INPUT type="button" value="Close Window"
Onclick="window.close();">

When the button is clicked now, the window.close() method is invoked and closes the window just like the standard close button at the top right of a window.

The blur() method

The blur() method enables us to put the window in the background behind the main window

<Input type="button" value="Hide the window" onClick="window.blur( );">

The focus() Method

The focus method enables us to bring a window into focus.

<BODY onBlur="window.focus();">

I am a new window! I am new than that old window that opened me, so I am special. Ha, ha!

Procedure

Step 1 : Start html program.

Step 2 : In head section include the scripting function callme().

Step 3 : In function the "document" object "writeln" property is used to display the heading message of the document.

Step 4 : The function callme() perform the process of biggest number among three numbers.

Step 5 : It display the result using "window" object "alert" property.

Step 6 : In body section create form to accept the three numbers.

Step 7 : Call the function callme() using onclick() event handler on submit button.

Program

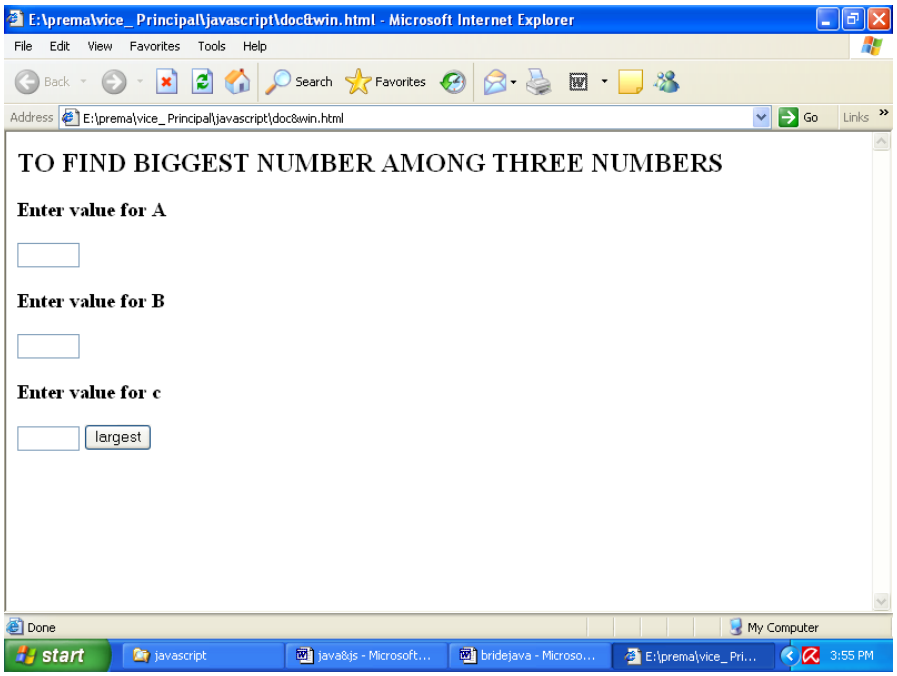
Using document and window object

```
<html>
<head>
<script language="JavaScript">
function callme(m,n,p)
{
 window.alert("welcome to JavaScript");
 document.writeln("Biggest Number among three Numbers");
 var a=m;
 var b=n;
 var c=p;

 if ((a>b)&&(a>c))
 {
    window.alert("a is largest no.");
 }
 else
 {
   if(b>c)
     window.alert("b is largest no.");
  else
    window.alert("c is largest no.");
 }
}
</script>
</head>
<body>
<form name="f1">
<H2>TO FIND BIGGEST NUMBER AMONG THREE NUMBERS</H2>
<h3>Enter value for A</h1>
<input type=text name="t1" size=5 value="">
<h3>Enter value for B</h1>
<input type=text name="t2" size=5 value="">
<h3>Enter value for c</h1>
<input type=text name="t3" size=5 value="">
<input type=button value=largest
onClick="callme(t1.value,t2.value,t3.value)";>
```

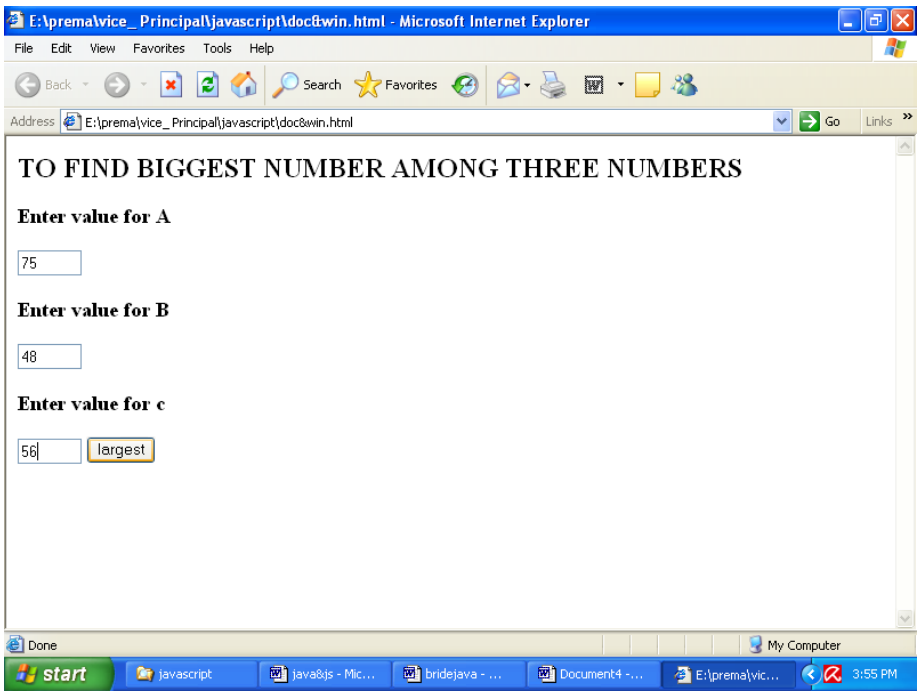
</form>
</body>
</html>

Output

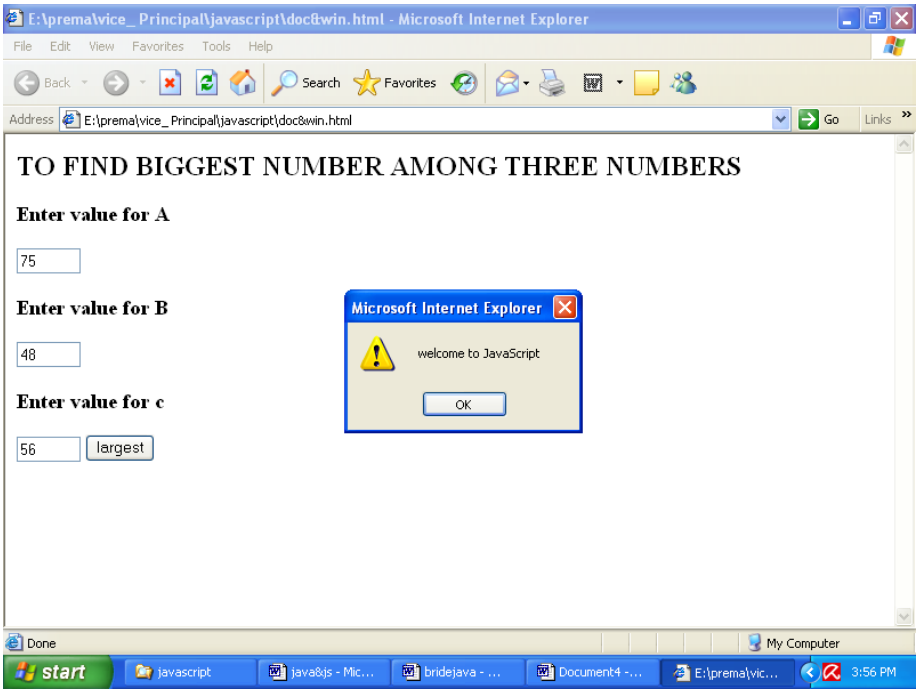
Form Displayed at run time



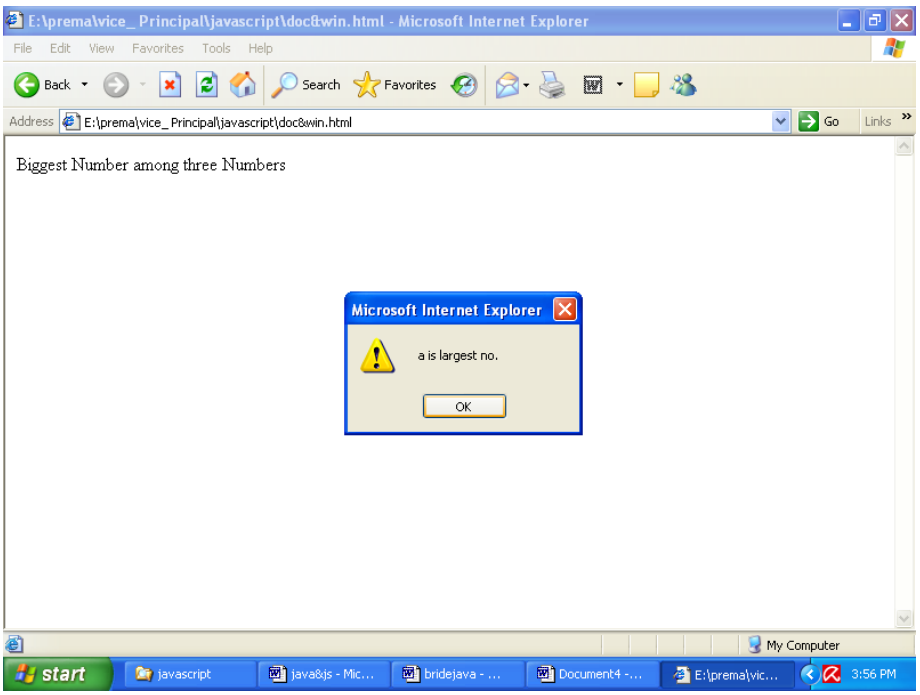
Values given to text box



After click on Largest button



After click on ok button in window



Program 5 : Program to create a Java Script clock.

Introduction

The Date object

The Date object is another predefined JavaScript object. It enables us to deal with time and to get certain time values that we can use in our scripts. In order to use this object, we need to create an instance of the object to which we can refer.To create an instance of the Date object, we use the new keyword as we have with a number of other objects, as shown in the following example:

```
var instance_name= new Date();
```

Methods of Date object

getDate()

It returns the day of the month.

```
var now=new Date()
var day=now.getDate();
```

If date is 15/4/2008 ,it returns 15.

getDay()

It enables us to get the day of the week.It returns number represents the number of days into week(0-6). If it is Sunday,the method returns 0.

getMonth()

It returns month of date.

Example

If date is "15/3/2008" , it returns 3.

getYear()

It returns year of Date.

Example

If date is "15/3/2008" , it returns 2008.

getTime(),getHours(),getMinutes() and getSeconds()

These methods returns current time ,hour,minute and second.

Example

```
var now=new Date()
var t=now.getTime()
var h=now.getHours()
var m=now.getMinutes()
var s=now.getSeconds()
```

setDate() method

This is used to change the day of the month.

Example

```
var now=new date()
now.setDate(23);
```

setMonth()

This is used to change the month of date.

now.setMonth(5);

setYear()

This is used to change the year of date.

now.setYear(2009);

setTime()

This is used to change the time of current time.

now.setTime(3:34:20Am)

setHours()

This is used to change the hour of current time.

now.setHours(4);

setMinutes()

This is used to change minutes of current time.

now.setMinutes(35);

setSeconds()

This is used to change seconds of current time.

now.setSeconds(24);

Procedure

Step 1 : Start html program

Step 2 : In head section include colckOpen() scripting function.

Step 3 : This function displays the todays date, hour, minute and seconds at movement of calling.

Step 4 : In body section create a submit button using form.

Step 5 : Call the function clockOpen() using click event handler to display the result.

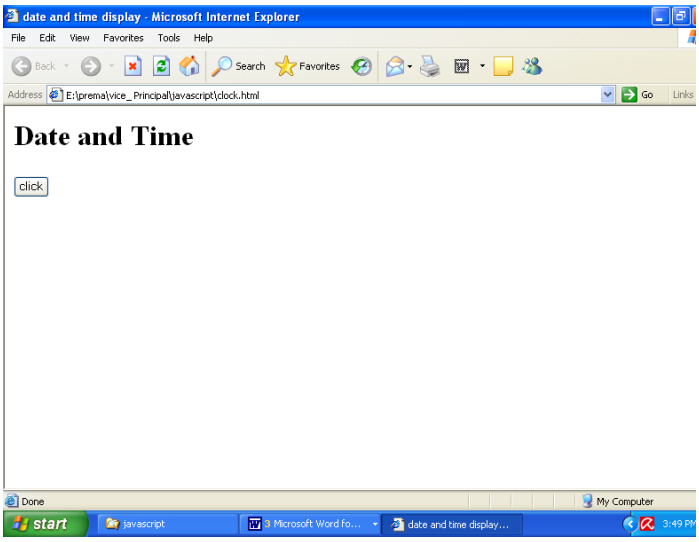
Program

Clock in Java Script

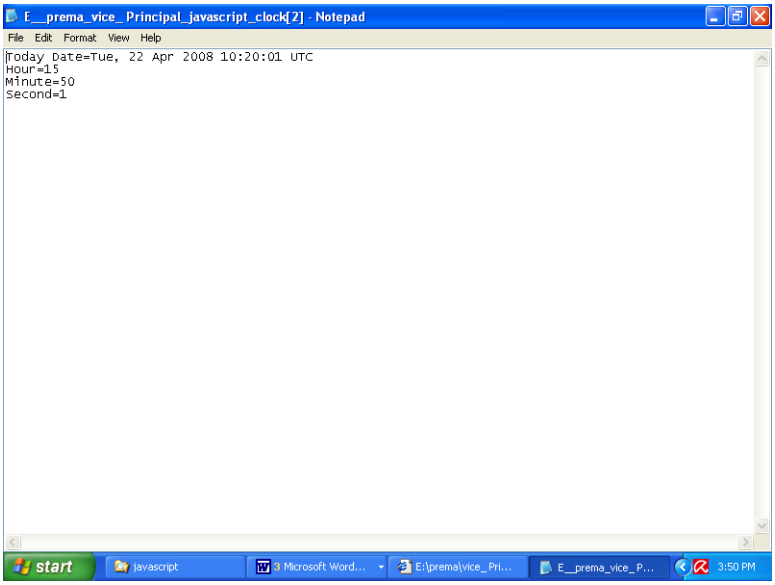
```
<html>
<head>
<title>date and time display</title>
<script language="JavaScript">
function clockOpen()
{
 var d=new Date();
 document.writeln("Today Date="+d.toGMTString());
 document.writeln("Hour="+d.getHours());
 document.writeln("Minute="+d.getMinutes());
```

```
  document.writeln("Second="+d.getSeconds());
}
</script>
</head>
<body>
<h1>Date and Time<h1>
<form name="f">
<input type="button" value=click onClick="clockOpen();">
</form>
</body>
</html>
```

Output



After the click on click button



Program 6 : Program to work with Forms Using Java Script.

Introduction to Form object in Java Script

The JavaScript form object will help us when we need to access certain elements or attributes of the form in a script. The form object has only a few properties and methods. Let's begin with the properties.

Properties of Form object

The form object's properties provide information we might need when working with forms in our scripts. Most of these properties just hold values corresponding to the various attributes in an HTML <FORM> tag. A few of them have different types of value, though.

The action property

This property allows you to access the value of the action="*url"* attribute in the opening <FORM> tag. This attribute is used to send the form to a

Property	**value**
Action	the value of the action attribute in the HTML <FORM> tag
Elements	An array that includes an array element for each form element in an HTML form
Encoding	the value of the enctype attribute, which varies with different browsers
Length	the value of the total number of element in an HTML form
Method	the value of the method attribute in an HTML <FORM> tag
Name	the value of the name attribute in an HTML <FORM> tag
Target	the value of the target attribute in an HTML <FORM> tag

Procedure

Step 1 : Start html program

Step 2 : In head section include valid() scripting function.

Step 3 : The function check whether the user name and password is correct.

Step 4 : In body section create form to accept the user name and password.

Step 5 : It call the function using onclick() event handler on submit button.

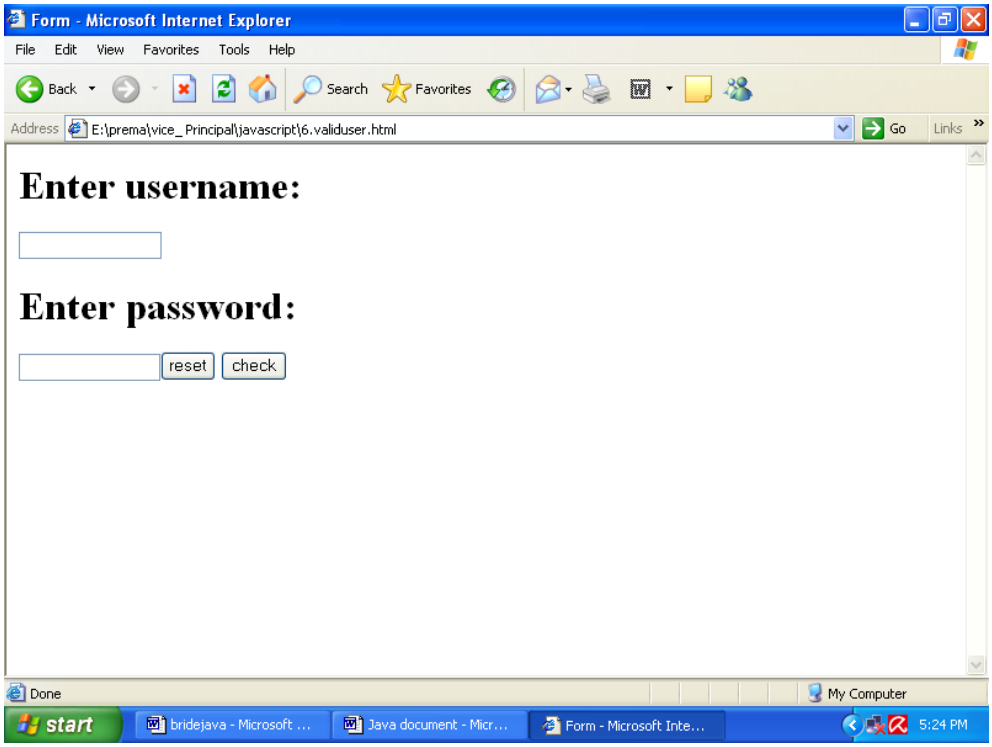
Step 6 : It produce the result.

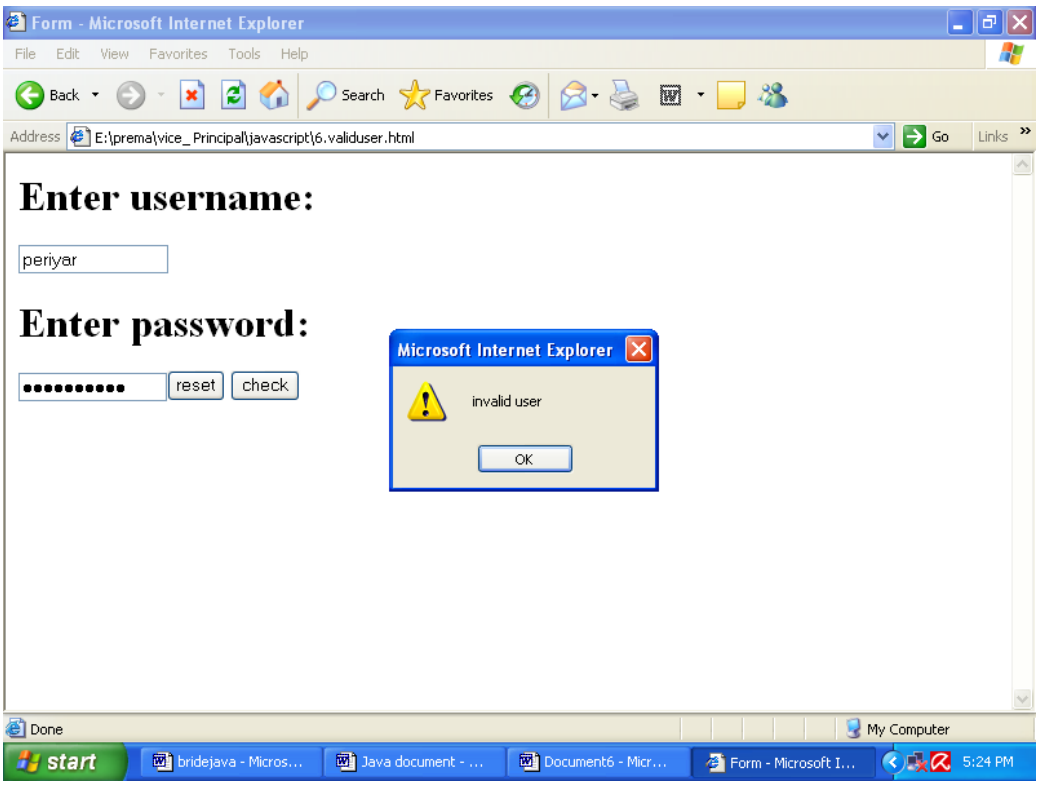
Program

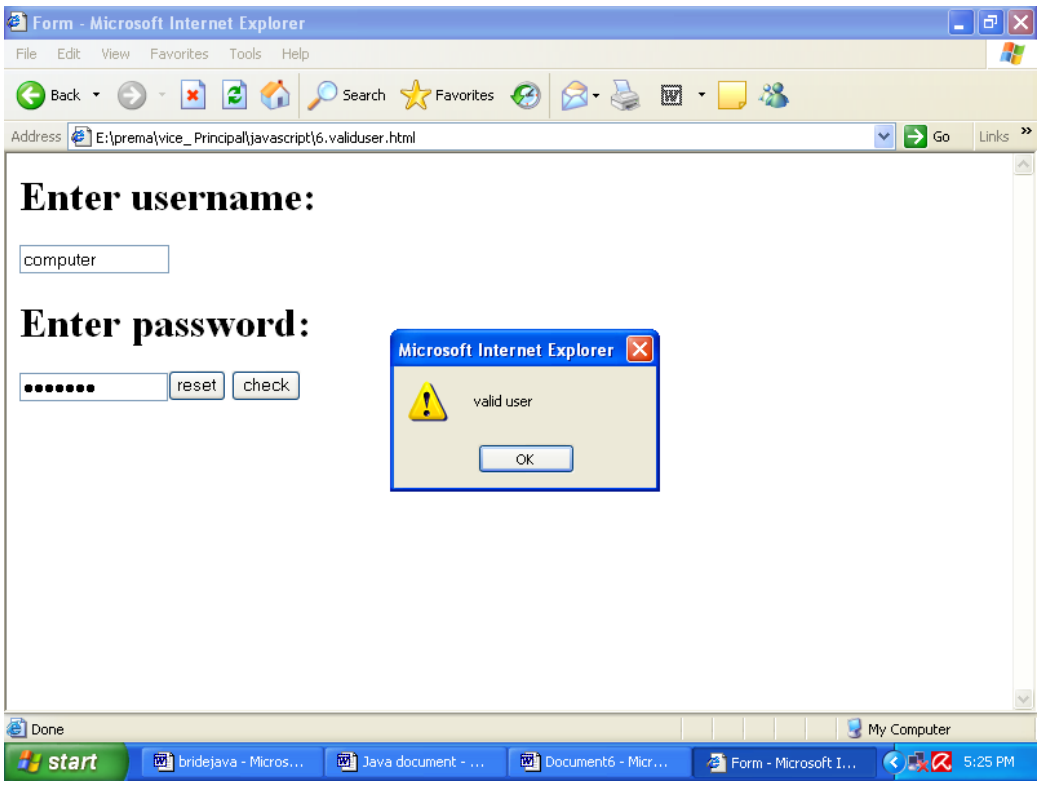
Forms using Java Script

```
<html>
<head><title> Form </title>
<script language="JavaScript">
function valid()
{
  if((f.user.value=="computer")&&(f.pass.value=="science"))
  {
  alert("valid user")
  }
  else
  {
  alert("invalid user")
  }
}
</script>
</head>
<body lang=EN-US style='tab-interval:.5in'>
<div class=Section1>
<form name=f>
<h1>Enter username:</h1>
<p><INPUT TYPE="text" SIZE="15" NAME="user"></p>
<h1>Enter password:</h1>
<p><INPUT TYPE="password" SIZE="15" NAME="pass"><INPUT
TYPE="reset" VALUE="reset">
<input type=button value=check onClick="valid()">
</p>
</form>
</body>
</html>
```

Output


Form - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address E:\prema\vice_ Principal\javascript\6.validuser.html
Go Links
Enter username:
Enter password:
reset check
Done
My Computer
start
bridejava - Microsoft ...
Java document - Micr...
Form - Microsoft Inte...
5:24 PM


Form - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address E:\prema\vice_ Principal\javascript\6.validuser.html
Go Links
Enter username:
periyar
Enter password:
reset check
Microsoft Internet Explorer
invalid user
OK
Done
My Computer
start
bridejava - Micros...
Java document - ...
Document6 - Micr...
Form - Microsoft I...
5:24 PM


Form - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address E:\prema\vice_Principal\javascript\6.validuser.html
Go Links
Enter username:
computer
Enter password:
reset
check
Microsoft Internet Explorer
valid user
OK
Done
My Computer
start
bridejava - Micros...
Java document - ...
Document6 - Micr...
Form - Microsoft I...
5:25 PM

JAVA Programming Language

Introduction

JAVA is a general-purpose object oriented programming language. It is really simple, reliable and powerful language. It is also used to developed application programs as well as Internet programs.

Java is a general-purpose, object-oriented programming language developed by **Sun Microsystems of USA in 1991**. Originally called **Oak** by **James Gosling**. Java was designed for the development of software for consumer electronic devices like TVs, VCR and Other electronic machines. The goal had a strong impact on the development team to make the language *simple, portable and reliable.*

Java Features

Sun Microsystems describes Java with the following attributes:

- Complied and Interpreted
- Platform-Independent and Portable
- Object-Oriented
- Robust and Secure
- Distributed
- Familiar, Simple and Small
- Multithreaded and Interactive
- High Performance
- Dynamic and Extensible

These features have made Java the first application language of the World Wide Web.

Compiled and Interpreted

A computer language is either **compiled** or **interpreted**. Java combines both these approaches thus making Java a two-stage system. **First**, Java compiler translates source code into byte code instruction. Byte codes are not machine instruction and in the **next** stage Java interpreter generates machine code that can be directly executed by the machine that is running the Java program.

Platform-Independent and Portable

Java programs can be easily moved from one computer to another, **anywhere and anytime.** Changes and upgrades in operating system, processors and system resources *will not force any changes in Java programs*.

Java ensures portability in *two ways*.

- Java compiler generates byte code that can be implemented on any machine.
- The sizes of the primitive data types are machine-independent.

Object-Oriented

Java is a **pure** object-oriented language. All program code and data reside within objects and classes and they are arranged in packages, that we use in our programs by inheritance. The object model in Java is simple and easy to extend.

Robust and Secure

Java provides many safeguards to ensure reliable code. It has strict compile and run time checking for data types. Java also has the concepts of exception handling, which captures errors and eliminates any risk of crashing the system.

Security is an important issue for a language that is used for programming on Internet. The absence of pointer in Java ensures that programs cannot access to memory locations without proper authorization.

Distributed

Java is designed as a distributed language for creating applications on networks. Java applications can open and access objects on Internet as easily as they can do in a local system. So multiple programmers at multiple remote locations to collaborate and work together on single project.

Familiar, Simple and Small

Java does not use pointers, preprocessor header files etc. Java eliminates operator overloading, multiple inheritance. So it is considered as a simple and small language. To make language look familiar to the exiting programmers, it modeled on C and C++ language.

Multithreaded and Interactive

Java handling multiple tasks simultaneously is called **multithreaded**. We need not wait for the application to finish one task before beginning the other. **For example** we can listen music while scrolling a page and at the same time download an applet from a distinct computer.

High Performance

Java performance is impressive for an interpreted language, mainly due to the use of intermediate byte code. Java architecture designed to reduce overheads during runtime and incorporating multithreading enhances the overall execution speed of Java programs.

Dynamic and Extensible

Java programs carry with them substantial amounts of run-time type information that is used to verify and resolve accesses to objects at run time. This makes it possible to dynamically link code in a safe and extensible manner.

Java Support Systems

Systems to support Java used to delivering information on the Internet as follows...

Support systems	Description
➢ Internet connection	- Local computer should be connected to the Internet.
➢ Web Server	- A program that accepts requests and sends the required document.
➢ Web Browser	- A program that provide access to WWW and runs Java applets.
➢ HTML	- A language for creating hypertext for the Web.
➢ APPLET Tag	- For placing Java applets in HTML document.
➢ Java Code	- Java code is used for defining Java applets.
➢ Byte Code	- Compiled code and transferred to the user Computer

Java Environment

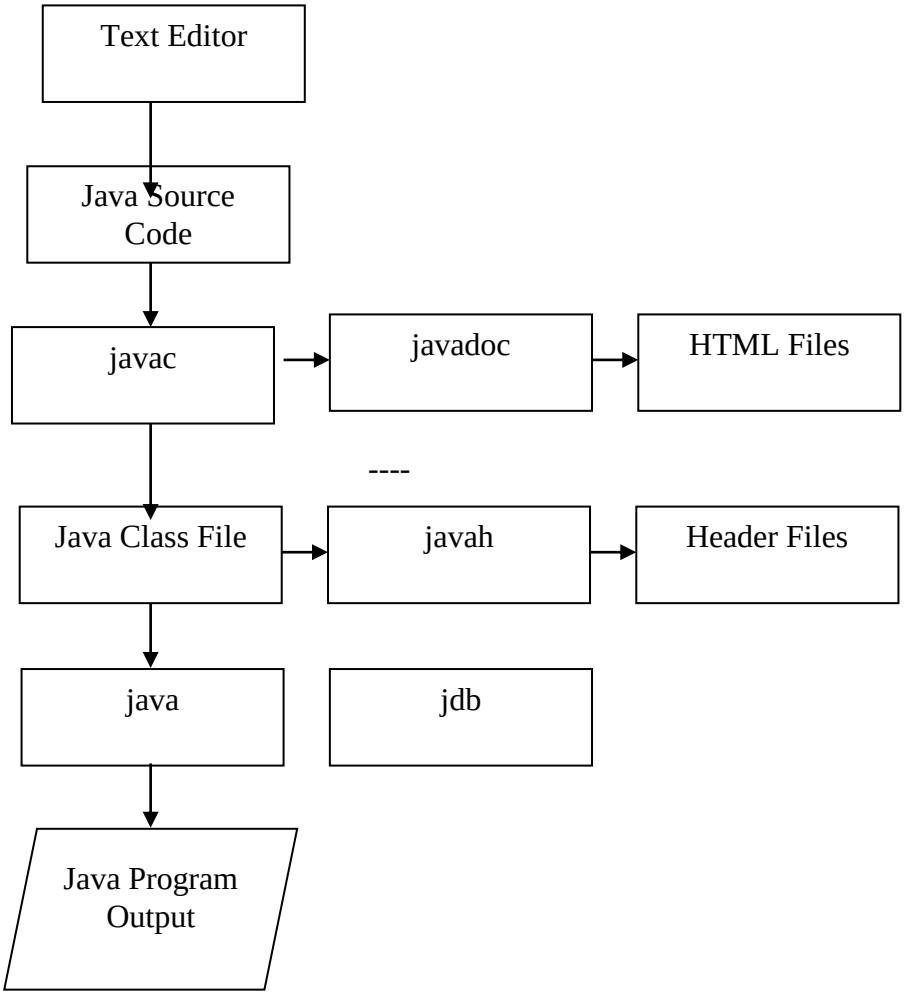
Java Environment includes a large number of development tools is known as **Java Development Kit (JDK)** and classes and methods is known as **Java Standard Library (JSL)** or **Application Programming Interface (API)**.

Java Development Kit (JDK)

The Java Development Kit with a collection of tools that are used for developing and running Java programs. Java development tools are:

- Applet viewer – Enable us to run Java applets.
- javac – Java compiler translates Java source code to byte code files.
- Java – Java interpreter, which runs applets and applications by reading & interpreting byte code files into machine code files
- Javap – Java disassembler, which enables us to convert byte code files into a program description.
- Javah – Produce header files for us with native methods.
- Javadoc – Create HTML document from Java source code files.
- Jdb – Java debugger, which helps us to find errors in our programs.

To create a Java program, we need to create a source code file using a *text editor*. The source code compiled using the Java compiler **javac** and executed using the Java interpreter **java.** The tools are applied to build and run application programs are shown below in diagram.



Process of building and running Java application programs

JAVA Practical Programs

Program 7 : Program to create a simple applet and application

Introduction about Applet

Definition

Applets are small java programs that are primarily used in Internet computing.

Types of Applets

We can embed applets into web pages in two ways

i. Local Applet

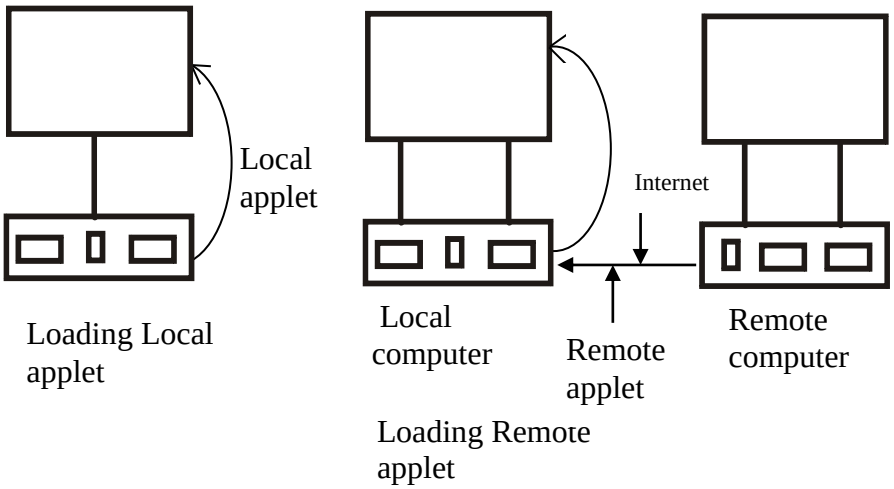
ii. Remote Applet

Local applet

An applet developed locally and stored in a local system is know as local applet.

Remote applet

A remote applet is which developed by someone else and stored on a remote computer connected to the Internet.



How applets differ from Applications

i. Applets do not use main() method

ii. Applet cannot be run independently

iii. Applets cannot read from or write to the files

iv. Applets cannot communicate with other servers on network

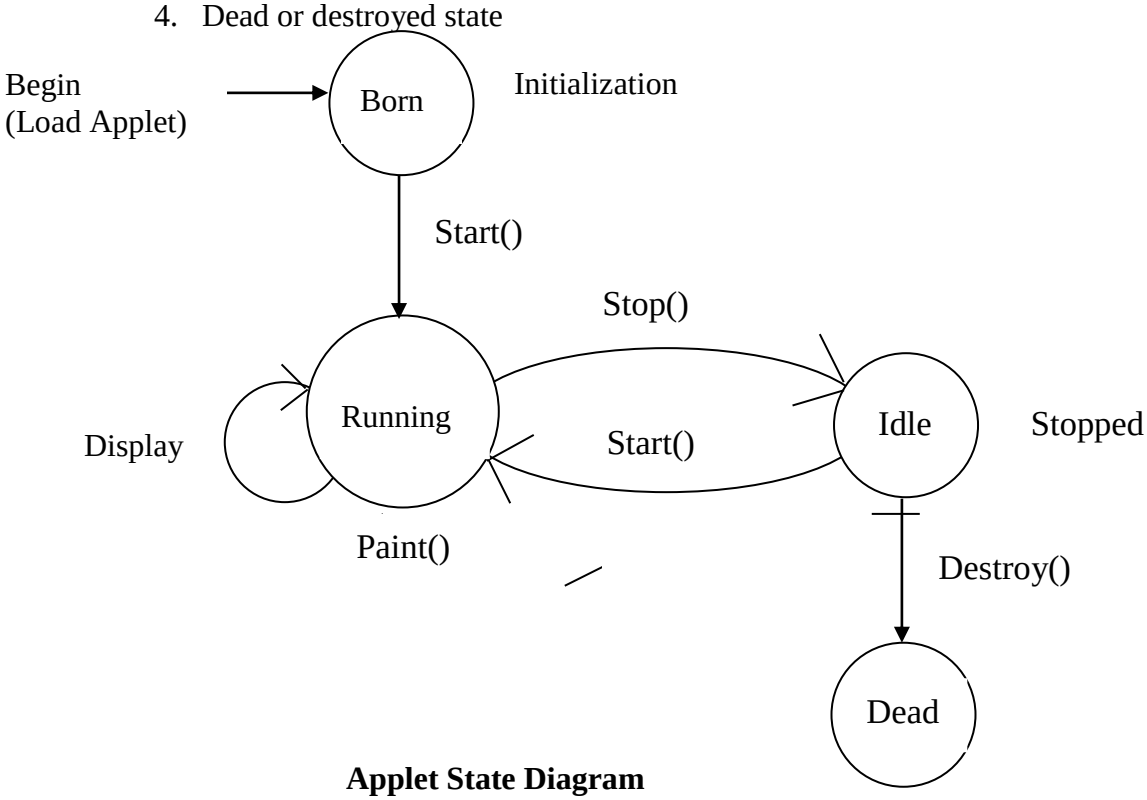
v. Applets cannot run any program form the local computer

vi. Applets are restricted from using libraries from other languages as C or C++

Applet life cycle

The applet states includes

1. Born or Initialization state
2. Running state
3. Idle state
4. Dead or destroyed state



Applet State Diagram

Building Applet code

The Applet class which is contained in the java.applet package provides life and behaviour to the applet through its methods such as init(), start(), paint().

Creating an executable Applet

Creating an executable applet through the command

```
javac filename.java
```

Procedure

Step 1 : Create a class named as draw

Step 2 : Using paint Graphics draw the oval shape for head, left-eye, right-eye, nose, left-ear and right-ear.

Step 3 : Fill ovals and arc of pupil mouth.

Step 4 : Save the java program as draw.java

Step 5 : Create a html program

Step 6 : Save the html program as applet.html

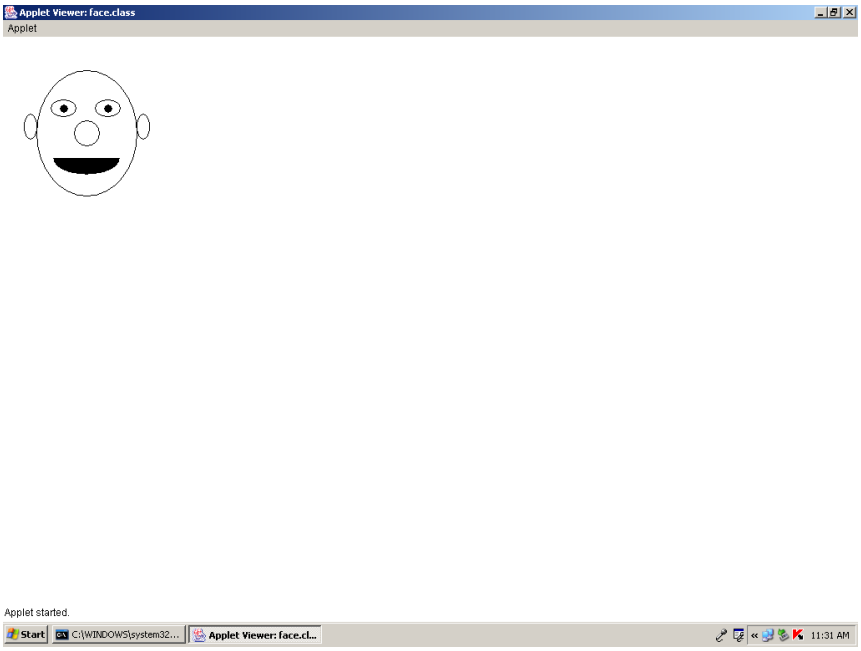
Step 7 : Compile and run the program.

Program

Simple Applet Application

```
/*  Program for Human Face Drawing Using Applet  */
import java.awt.*;
import java.applet.*;
public class face extends Applet
{
   public void paint(Graphics g)
   {
      g.drawOval(40,40,120,150);
      g.drawOval(57,75,30,20);
      g.drawOval(110,75,30,20);
      g.fillOval(68,81,10,10);
      g.fillOval(121,81,10,10);
      g.drawOval(85,100,30,30);
      g.fillArc(60,125,80,40,180,180);
      g.drawOval(25,92,15,30);
      g.drawOval(160,92,15,30);
   }
}
<! HTML Code for Human Face Drawing>
<html>
<applet
      code=face.class
      width =300
      height  =300
      >
</applet>
</html>
```

Output


Applet Viewer: face.class
Applet
Applet started.
Start
C:\WINDOWS\system32...
Applet Viewer: face.cl...
11:31 AM

Program 8: Using java class and objects

Defining a class

A class is a user-defined data type with a template that server to define its properties.

Syntax:

```
Class Class_name [Extends super_class_name]
{
        [Variable declaration;]
        [Method declaration;]
}
```

Adding Methods

Methods are declared the body of the class but immediately after the declaration of instance variable.

Syntax:

```
Type method_name(Parameter_list)
{
        Method-body;
}
```

Four basic parts of method declaration

1. The name of the method
2. The type of the value the method returns (type)
3. A list of parameters (Parameters-list)
4. The body of a method

Creating object

An object in java is essentially a block of memory that contains space to store all the instance variables.

Ex:

```
Rectangle rect1  //Declared
Rect1 = new Rectangle()   //Instantiate
```

Procedure

Step 1 : Create a class circlearea.

Step 2 : Create a main class circle

Step 3 : Create an object to class circlearea to initialize radius value of circle.

Step 4 : Call the value using object and calculate the area of circle.

Step 5 : Display the result.

Program

Program for Classes and Objects

```
class circlearea
{
        double r;
        void getdata(double r1)
        {
                r=r1;
        }
}

class circle
{
    public static void main(String args[])
    {
        double area;
        circlearea c=new circlearea();
        c.getdata(5.0);
        area=3.14*c.r*c.r;
        System.out.println("THE AREA OF CIRCLE  :  "
+area);
     }
}
```

Output

THE AREA OF CIRCLE : 78.5

Program 9 : Using Java Inheritance and Interface

Defining Interface

Interface contains methods and variables but with a major difference. The difference is that interface defines only abstract methods and final fields.

Syntax :

```
interface Interface_Name
{
        Variables declaration;
        Method decalaration;
}
```

Variables are declared as follows

```
Static final type variable_name = value;
```

Ex:

```
Interface Item
{
        Static final int code = 1001;
        Static final string name = "Fan";
        Void display();
}
```

Extending Interface

Like classes interfaces can also be extended. That is, an interface can be subinterfaced form other interfaces. This is achieved using the keyword extends.

Syntax:

```
interface name2 extends name1
{
        Body of name2;
}
```

Ex:

```
interface itemconstants
{
        int code = 1001;
        string name = "Fan";
}
interface item extends itemconstants
{
        void display();
}
```

Implementing Interfaces

Interfaces are used as "Superclasses" whose properties are inherited by classes. It is therefore necessary to create a class that inherits the given Interface.

Syntax

```
Class Class_name Implements Interfacename
{
        Body of classname;
}
```

Here the class classname "implements" the interface interfacename. General form of this

```
Class classname extends superclass
        Implements Interface1, Interface2, …
{
        Body of classname;
}
```

Ex:

```
Interface area              //Interface defined
{
        final Static float pi = 3.14;
        float compute (float x, float y);
}
class Rectangle implements Area
{
        public float compute (float x , float y)
        {
                return (x * y);
        }
}

class Circle implements Area
{
        public float compute (float x , float y)
        {
                return (pi * x * x);
        }
}

class Interface Test
```

```
{
    public Static void main (string args[])
    {
        Rectangle rect = new Rectangle();
        Circle cir = new Circle();
        Area area;
        Area = rect;
        System.out.println ("Area of Rectangle = "+area.compute(10,20);
        Area = cur;
        System.out.println ("Area of circle = " + area.compute(10,0);
    }
}
```

Procedure

Step 1 : Create class student used to initialize and display register-number and name of the student.

Step 2 : Create interface mark to initialize the subject mark

Step 3 : Create a class multiple and it acquire the features of class student and interface mark.

Step 4 : From main function call the member functions of class and display the result.

Program

Inheritance and Interface

```
class student
{
        int regno;
        String name;
        public void get(int r,String n)
        {
                regno=r;
                name=n;
        }
        public void display1()
        {
                System.out.println("Register no.:"+regno);
                System.out.println("Name        .:"+name);
        }
}
 interface mark
```

```
{
        static final int cobol=70;
        static final int sport=60;
        public void display2();
 }
  class multiple extends student implements mark
  {
         int total;
         public void display2()
        {
                total=cobol+sport;
                System.out.println("cobol mark is:"+cobol);
                System.out.println("sport mark is:"+sport);
                System.out.println("Total is:"+total);
        }
        public static void main(String args[])
        {
                multiple m=new multiple();
                m.get(100,"kumar");
                m.display1();
                m.display2();
        }
  }
```

Output

```
Register no.:100
Name        .:kumar
cobol mark is:70
sport mark is:60
Total is:130
```

Program 10 : Using Arrays in java

Array:

An array is a group of contiguous or related data items that share a common name.

Ex:

Salary [10];

Represent the salary of 10 employees, while the complete set of values is referred to as an array. The individual values are called elements.

One-Dimensional Array

Definition

A list of items can be given one variable name using only one subscript and such as variable is called a single-subscripted variable (or) a One-dimensional array.

Ex:

x[1], x[2], x[3], … x[n].

Two-Dimensional Array

Java allows us to define such table of items by using two-dimensional array.

Ex:

V[4][3]

Creating an Array

Creating of an array involves three steps.

i. Declare the array

ii. Create memory locations

iii. Put values into the memory location

Declaration of array

Arrays in java may be declared in two forms.

Form 1 :

Type arrayname [];

Form 2 :

Type [] arrayname;

Ex:

int number [];

int [] number;

Creating an arrays

Java allows us to create array using new operator only as shown below

arayname = new type [size];

Ex:

Name = new string [20];

Initialization of array

The initial step is to put values into the array created.

Arrayname [subscript] = value;

Ex:

Number [0] = 35;

Procedure

Step 1 : Create a class ascending

Step 2 : Initialize the value to array "a"

Step 3 : Check the values of array using loops.

Step 4 : It the first value in array is greater than the second value of array then swap it otherwise continue this process until complete all the values in array.

Step 5 : Display the sorted value in ascending order.

Program

Program using Array

```
class ascending
{
 public static void main(String args[])
 {
    int  a[]={4,3,2,7,6,1,5};
    int i,j,t;
 for (i=0;i<a.length;i++)
 {
 for (j=i+1;j<a.length;j++)
 {
  if (a[i]>=a[j] )
  {
    t=a[i];
    a[i]=a[j];
    a[j]=t;
 }
 }
 }
for (i=0;i<a.length;i++)
{
 System.out.println(a[i]);
System.out.print("\t");
}
}
}
```

Output

1 2 3 4 5 6 7

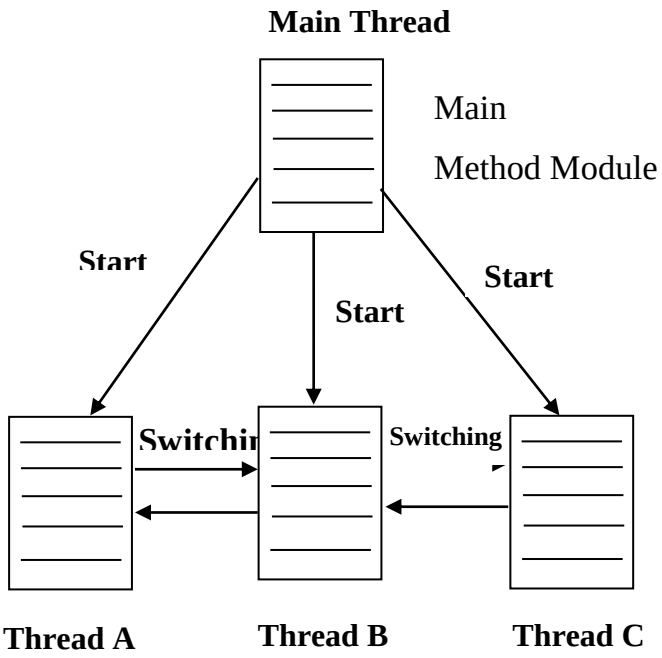
Program 11: Using Threads and Multithreads

Threads Definition :

A thread is similar to a program that has a single flow of control. It has a beginning, a body and an end, and executes commands sequentially. This is otherwise is called as single threaded.

Multithread Definition :

Multithread can execute several programs simultaneously. This ability is known as multitasking.



Creating threads

Creating threads in java is simple threads are implemented in the form of object that contain a method called run(). The run() method is the heart and soul of any thread.

Syntax :

```
public void run()
{
        …..
        …..
        …..
}
```

A new thread can be created in two ways.

1. By creating a thread class
2. By converting a class into thread

Extending the thread class

i) Declare the class as extending the thread class

ii) Implement the run() method

iii) Create a thread object and call the start() method

Declaring the class

The thread class can be extended as follows

```
class mythread extends thread
{
    …..
    …..
}
```

Implementing the run() method

The run() method has been inherited by the class mythread.

```
public void run()
{
    …..
    …..
    …..
}
```

Starting new thread

To actually create and run an instance of our thread class

```
mythread athread = new mythread();
athread start();
```

Procedure

Step 1 : Create a class “fact” with thread to find a factorial number.

Step 2 : Create a class “prime” with thread to display the given numbers are prime or not.

Step 3 : Create a class “fib” with thread to find the Fibionacci numbers.

Step 4 : From main class “multi” execute the above three threads simultaneously.

Step 5 : Display a result.

Program

Program using Threads and Multithreads

```
/*MULTITHREAD PROGRAM*/
import java.lang.*;
import java.io.*;
class fact extends Thread
{
```

```
 public void run()
 {
  for(int i=1;i<=10;i++)
  {
    int f=1;
    for(int j=1;j<=i;j++)
    {
     f=f*j;
    }
    System.out.println(i+" Factorial is "+f);
    if(i==7)
    {


      try
      {
       sleep(5000);
      }
      catch(Exception e)
      {}
     }
   }
 }
}
class prime extends Thread
{
  public void run()
  {
   int flag=0;
   for(int i=3;i<=20;i++)
   {
    for(int j=2;j<i;j++)
    {
      if(i%j==0)
      {
        System.out.println(i+" Not prime number ");
        flag=0;
        break;
```

```
          }
          else
          {
            flag=1;
          }
        }

        if(flag==1)
          System.out.println(i+" Prime number");
          if(i==10)
          {
            stop();
          }
        }
      }
 }

class fib extends Thread
{
  public void run()
  {
    int a=-1,b=1,c,k;
    for(k=1;k<=10;k++)
    {
      c=a+b;
      a=b;
      b=c;
      System.out.println("Fibionacci no is "+c);
      if(k==4)
      {
        yield();
      }
    }
  }
}
class multi
{
  public static void main(String args[])
```

```
  {
    fact f=new fact();
    prime p=new prime();
    fib fi=new fib();
    f.start();
    p.start();
    fi.start();
  }
}
```

Output

```
1 Factorial is 1
2 Factorial is 2
3 Factorial is 6
4 Factorial is 24
5 Factorial is 120
6 Factorial is 720
7 Factorial is 5040
3 Prime number
4 Not prime number
5 Prime number
6 Not prime number
7 Prime number
8 Not prime number
9 Not prime number
10 Not prime number
Fibionacci no is 0
Fibionacci no is 1
Fibionacci no is 1
Fibionacci no is 2
Fibionacci no is 3
Fibionacci no is 5
Fibionacci no is 8
Fibionacci no is 13
Fibionacci no is 21
Fibionacci no is 34
8 Factorial is 40320
9 Factorial is 362880
10 Factorial is 3628800
```

Program 12 : Using AWT package

Introduction

Definition :

An AWT stands for **Abstract Window Toolkit** in which consist of Classes and Interfaces such as TextField, Button, Label, Graphics etc., Each and every Classes and Interfaces having the different methods, which are doing different tasks. Let us take Graphics class has the following methods.

Method	Description
clearRect ( )	Erases a rectangular area of the canvas.
copyArea ( )	Copies a rectangular area of the canvas to another area.
drawArc ( )	Draws a hollow arc.
drawLine ( )	Draws a straight line.
drawOval ( )	Draws a hollow oval.
drawPolygon ( )	Draws a hollow polygon
drawRect ( )	Draws a hollow rectangle.
drawRoundRect ( )	Draws a hollow rectangle with rounded corner.
drawstring ( )	Displays a text string.
fillArc ( )	Draws a filled arc.
fillOval ( )	Draws a filled oval.
fillPolygon ( )	Draws a filled polygon
fillRect ( )	Draws a filled rectangle.
fillRoundRect ( )	Draws a filled rectangle with rounded corners.
getColor ( )	Retrieves the current drawing color.
getFont ( )	Retrieves the currently used font.
getFontMetrics ( )	Retrieves information about the current font
setColor ( )	Set the drawing color.
setFont ( )	Set the font.

We take some methods as example

Lines And Rectangles

The **drawLine ( )** method is used to draw a line, it takes two pair of coordinates, (x1, y1) and (x2, y2) as arguments and draws a line between them. The *general form* is

g.drawLine(x1, y1, x2, y2);

Example:

```
g.drawLine( 10, 10, 50, 50);
```

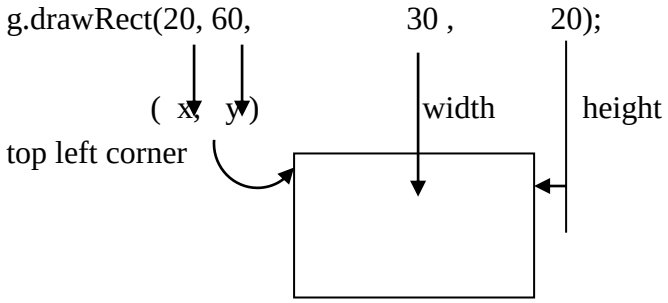
The **g** is the **Graphics** Object passed to **paint ( )** method.

The **drawRect ( )** method is used to draw a rectangle, it takes four arguments, the first two represent the x and y coordinates of the top left corner

of the rectangle, and remaining two represent width and height of the rectangle. The *general form* is

g.drawRect(x, y, width, height);

Example:

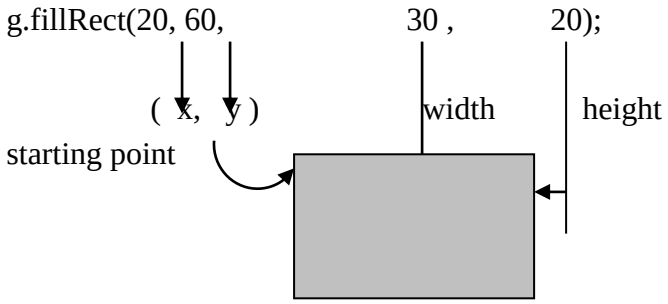


Rectangle

The **drawRect ( )** method draws only the outline of a box. We can draw a solid box using the method **fillRect ( )** method. This also takes four parameters, the first two represent the x and y coordinates of the top left corner of the rectangle, and remaining two represent width and height of the rectangle. The *general form* is

g.fillRect(x, y, width, height);

Example:



Filled Rectangle

We can also draw rounded edges rectangles, using the methods **drawRoundRect ( )** and **fillRoundRect ( )**. These two methods are same as drawRect ( ) and fillRect ( ) methods except that they take *extra two arguments* representing the width and height of the angle of corner.The *general forms* are

g.drawRoundRect(x, y, width, height , width of angle of corner, height of angle of corner);

g.fillRoundRect(x, y, width, height, width of angle of corner, height of angle of corner);

Example:

g.drawRoundRect(20, 60, 30 , 20, 10, 10);

Rounded Rectangle

g.fillRoundRect(20, 60, 30 , 20, 10, 10);

Procedure

Step 1 : Create class “graph” extends applet.

Step 2 : Using Graphics class methods in AWT create the following objects

i. Using drawLine() method to draw a line
ii. Using drawRect() method to draw a rectangle
iii. Using fillRect() method to draws a filled rectangle
iv. Using drawRoundRect () method to Draws a hollow rectangle with rounded corner
v. Using fillRoundRect () method to Draws a filled rectangle with rounded corner
vi. Using drawOval() method to Draws a hollow oval
vii. Using setColor() method to Set the drawing color
viii. Using fillOval() method to Draws a filled oval
ix. Using fillArc() method to Draws a filled arc

Step 3 : Display an integrated objects.

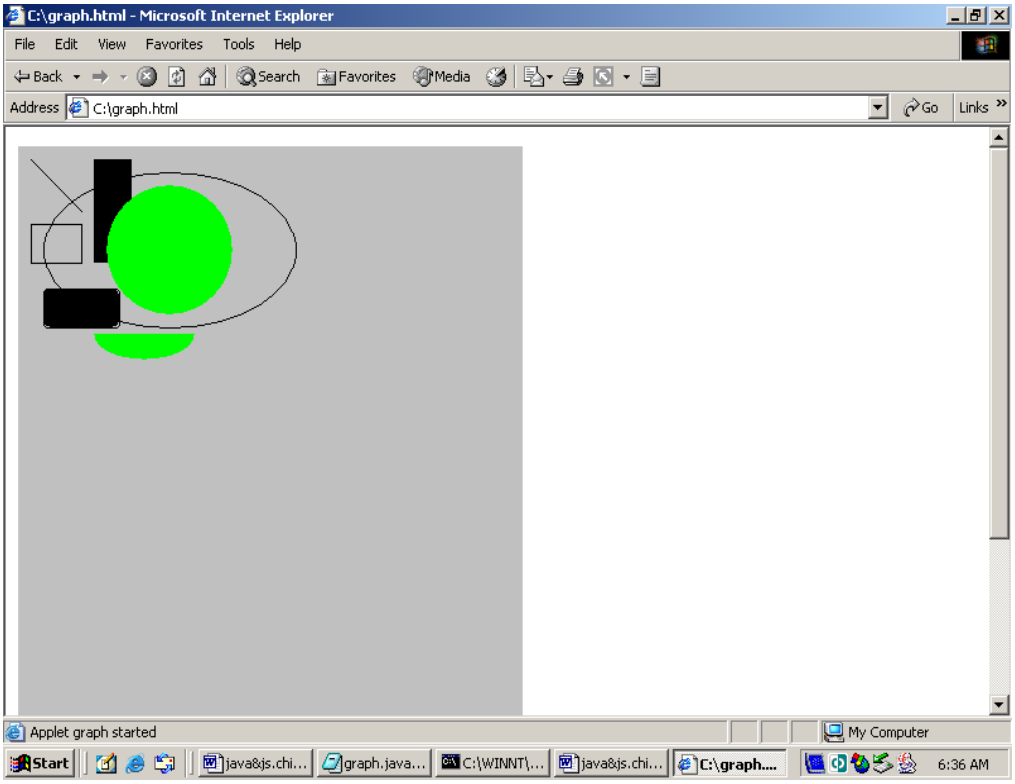
Program

AWT Pacakage

```
Import  java.awt.*;
Import java.applet.*;
Public class  graph extends Applet
{
public void paint(Graphics g)
{
g.drawLine(10,10,50,50);
g.drawRect(10,60,40,30);
g.fillRect(60,10,30,80);
g.drawRoundRect(20,110,60,30,5,5);
```

```
g.fillRoundRect(20,110,60,30,10,10);
g.drawOval(20,20,200,120);
g.setColor(Color.green);
g.fillOval(70,30,100,100);
g.fillArc(60,125,80,40,180,180);
}
<html>
<body>
<applet code=graph.class width=400 height=600>
</applet>
</body>
</html>
```

Output



Program 13 : Using I/O package

Introduction

Definition :

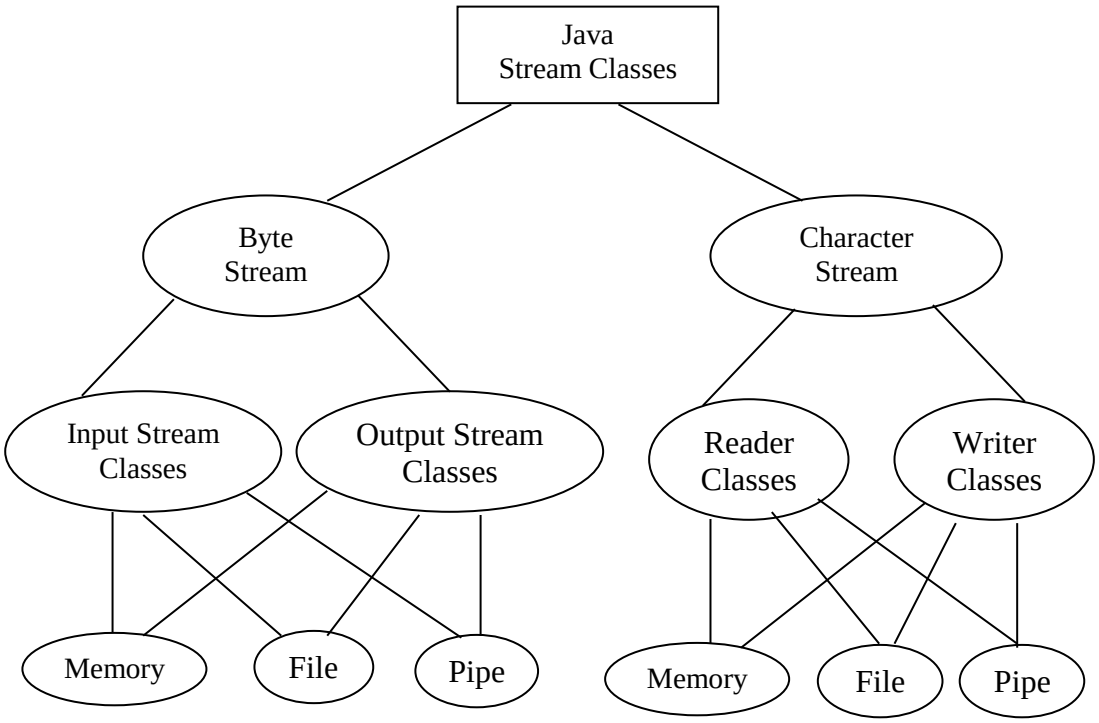
The Java I/O package java.io is an import file in which having large number of stream classes that provide capabilities for processing all types of data. Hence, The **java.io** package includes a class known as the **File class** that provides support for creating files and directories.

Classification of java stream class

The java stream classes may be categorized into two groups based on the data type on which type operate.

- Byte stream classes that provide support for handling I/O operations on bytes.
- Character stream classes that provide support for managing I/O operations on characters.

Diagram shows how stream classes are grouped based on their functions.



Classification of Java Stream Classes

Byte Stream Classes

Java provides two kinds of byte stream classes: I*nput stream classes* and O*utput stream classes*.

Input Stream Classes

Input stream classes that are used to read 8-bit bytes include super class known as **InputStream** and a number of subclasses for supporting various input-related functions such as

- Reading bytes
- Closing streams
- Marking positions in streams
- Skipping ahead in a stream
- Finding the number of bytes in a stream

Output Stream Classes

Output stream classes are derived from the base class **OutputStream**. The **OutputStream** is an abstract class and therefore we cannot instantiate it. The several subclasses of the **Outputstream** can be used for performing the output operations. The **OutputStream** includes methods that are designed to perform the following tasks:

- Writing bytes
- Closing streams
- Flushing streams

Character Stream Classes

Character streams can be used to read and write 16-bit Unicode characters. There are two kinds of character stream classes: *Reader stream classes* and W*riter stream classes.*

Reader Stream Classes

Reader stream classes are designed to read character from the files. The **Reader** class is the base class for all other classes in this group.

Writer Stream Classes

The writer stream classes are designed to perform all output operations on files. The Writer stream classes are designed to write characters. The Write class is an abstract class, which acts as a base class for all the other writer stream classes.

Using The File Classes

The **java.io** package includes a class known as the **File class** that provides support for creating files and directories. The class includes several constructors for instantiating the **File** objects. This class also contains *several methods* for supporting the operations such as

- Creating a file
- Opening a file
- Closing a file
- Deleting a file
- Getting the name of a file
- Getting the size of a file
- Checking the existence of a file
- Renaming a file
- Checking whether the file is write able

- Checking whether the file is readable

Procedure

Step 1 : Create a class wrcharacter

Step 2 : Open a city.dat file in Write mode

Step 3 : Read the character from console device and write it in city.dat file

Step 4 : Close the opened file.

Step 5 : Open a city.dat file in Read mode

Step 6 : Read the data from city.dat file and display on console device.

Step 7 : Close the opened file.

Program

I/O Package

```
import java.io.*;
class wrcharacter
{
 public static void main(String args[])
 {
  //write character
  try
  {
    FileWriter fw=new FileWriter("city.dat");
    int  ch;
    while((ch=System.in.read())!=-1)
    {
      fw.write(ch);
    }
     fw.close();
  }
  catch(IOException e)
  {
  System.out.println(e);
  System.exit(-1);
  }
  // Read character
  int b;
  try
  {
    FileReader fr=new FileReader("city.dat");
    while((b=fr.read())!=-1)
```

```
    {
      System.out.print((char)b);
    }
    fr.close();
  }
  catch(IOException e)
  {
  System.out.println(e);
  System.exit(-1);
  }

 }
}
```

Output

```
C:\>java wrcharacter
s
a
l
e
m
^Z
s
a
l
e
m
```

NOTES

NOTES