Bachelor Of Engineering
In
Information Technology

Semester Six, Third Year(Even semester)

10th May 2022

18th Offline Lecture

Padre Conceicao College of Engineering

Verna Goa 403722 India

Web Technology

RC 2019-20

Unit 3

UNIT 4 :Refer

Web Technologies: HTML, Javascript, PHP, Java, JSP, ASP.NET, XML and AJAX, Black Book; Publisher: Dreamtech Press(2015) ; ISBN: 978-81-7722-997-4

Sr.No	Title	Chapter
1	JavaScript for AJAX,	29
2	Asynchronous data transfer with XML Http Request	30
3	Implementing AJAX Frameworks	31

Javascript for AJAX

- Introduction to Javascript
- Basics of Javascript
- What is DOM
- Creating a Javascript Application without AJAX
- Javascript and AJAX
- Creating a Javascript Application with AJAX

Introduction to Javascript

- With the advent of the web, HTML and CGI were the only 2 technologies available for developing the web pages.
- HTML defined the part of the text document and instructed the browser to display it.HTML has one disadvantage of being static. If you want to change something or want to use the data entered by the user, you need to make a round trip to the server. This procedure, indeed , took a lot of time.
- When dynamic technologies such as ASP,ASP.NET, or JSP ,arrived in the market, we got an ability to create dynamic web pages, which had the ability to interact with the user. In this case ,the user entered data into the forms, which was then sent to the server.

- The server processed the data and then provided the response to the browser in the form of HTML document. The problem with this type of approach was that each time there was a change; the whole process had to be repeated. The display of page meant the reloading of page, which was a slow process and failed frequently. This was cumbersome and not impressive.
- There arose the need for a new technology ,which would allow the web developers to develop web applications that provide immediate feedback to the users and change the HTML without reloading the page from the server every time. Suppose there is a form, which is reloaded every time there is an error in any of the fields of the form. This would be a slow process. But,you can make it a fast process by using Javascript in your application. This is done by Javascript at the client side, which flag an error message as soon as the error occurs in the form without reloading the page and without moving to the Web Server.

- Javascript is executed by the browser on the user’s computer. This type of code is called client-side code and this type of approach results in fast-running web sites. It is a scripting language - a system of programming codes- created by Netscape and can be embedded into HTML of a web page in order to add more interactivity and functionality. Javascript is an interpreted language,i.e.it doesn’t need any compiler to execute a code.
- When Javascript was first introduced, its name was LiveScript,but Netscape changed the name Javascript . Javascript is a scripting language and it is not related to Java in any way. Netscape included Javascript in their Netscape Navigator 2.0 browser via an interpreter, which read and executed the Javascript code added to .html pages.
- Script languages,such as Javascript ,are faster and easier than the structured languages,such as C++ and Java.

where to use Javascript in your Web Applications

1. Data entry validations
2. HTML interactivity
3. Serverless CGIs
4. CGI prototyping
5. Offloading a busy server
6. Adding life to otherwise –dead pages

Data entry validations

- A Javascript can be used to validate the data entered in the form, before it is sent to the server for further processing.
- This saves a lot of time as well as effort of the server from extra processing.

HTML interactivity

- You can use DHTML for positioning the contents precisely on the web page, but if you want interactivity in your web page, you need Javascript.
- By the word interactivity, we mean that the user is able to see the changes or get the feedback as soon as he enters the data in a form.

Serverless CGIs

- This term is used to describe the processes that would be programmed as CGIs (if javascript was not there) on the server-side,yielding slow performance because of the interactivity required between the program and the user.
- This includes the tasks, such as small data collection lookup, modification of images, and generation of HTML in other frames and windows-based on the user input.
- But , with the availability of Javascript, all these tasks can now be done on the client-side.

CGI prototyping

- Sometimes you may want a CGI program to be at the root of the application because it reduces potential incompatibilities among browser brands and versions.
- It may be easier to create a prototype of CGI in client-side Javascript.
- Use this opportunity to polish the user interface before implementing the application as a CGI.

Offloading a busy server

- If you have a highly trafficked web site, it may be beneficial to convert the frequently used CGI processes to the client-side Javascript.
- After a page is downloaded the server is free to serve other visitors.
- Not only will this lighten the server load, but users also experience quicker response to the application embedded in the page.

Adding life to otherwise –dead pages

- HTML by itself is quite flat.
- Adding a blinking chunk of text doesn’t help much;animated GIF images more often distract from , rather than contribute to the user experience at your site.
- But if you can dream up your ways to add some interactive zip to your page, it may engage the user and encourage recommendation to friends or repeat visits.

Merits of Javascript

1. Less server interaction
2. Immediate feedback to the visitors
3. Automated fixing of minor errors
4. Increased usability by allowing visitors to change and interact with the user interface without reloading the page
5. Increased interactivity
6. Richer interfaces
7. Lightweight environment

- Less server interaction-you can validate the user input before sending the page off to the server. This saves server traffic, which means saving money.
- Immediate feedback to the visitors-The user doesn’t have to wait for a page reload to see if they forgot to enter some data.
- Automated fixing of minor errors – If you have a database system that expects a date in the format dd-mm-yyyy and the visitor enters it in the form of dd/mm/yyyy, a clever JavaScript could change this minor mistake prior to sending the form to the server. If that was the only mistake the visitor made , you can save her an error message – thus making it less frustrating to use the site.
- Increased usability – A classic example of this would be select boxes that allow immediate filtering, such as only showing the available destinations for a certain airport,, without making you reload the page and wait for the result.

- Increased interactivity - you can create interfaces that react when the user hovers over them with a mouse or activates them via the keyboard. This is partly possible with CSS and HTML as well, but JavaScript offers you a lot wider –and more widely supported – range of options.
- Richer interfaces – you an use JavaScript to include items, such as drag-and-drop components and sliders – something that originally was only possible in thick client applications.
- Lightweight environment- Instead of downloading, a large file like a java applet or a Flash movie, scripts are small in file size and get cached once they are loaded. JavaScript also uses the browser controls for functionality rather than its own user interfaces, like Flash or Java applets do. This makes it easier for users, as they already know these controls and how to use them.

Basics of JavaScript

The following are some essential points regarding the Syntax of JavaScript

1.JavaScript is case sensitive. The variable name XYZ is different from the variable name xyz.

- To use JavaScript in your web document, you need to use the <script> tag. All the JavaScript code is written between the <script> and </script> tag.
- You can add JavaScript code anywhere in the HTML document and browsers will interpret it. It is a bad practice of inserting JavaScript code in the document. You should always try to include the JavaScript code in the body of the HTML page

2.// is used to make the current line as the comment . The interpreter does not run the content written after the //.These comments are used to insert the notes in the document. These notes help in understanding the code and are also helpful for the other person who wants to read and understand the code.

3.There is one more category of comments – the multi-line comments. The multi-line comments start with /* and end with */.Such comments are useful when you don’t want to execute a certain section of the code, but also not want to delete the code. For example, if you were having a problem with a block of code and you also do not know the cause of the problem, then in such a situation you can comment that block of code using multi-line comments so as to isolate the problem.

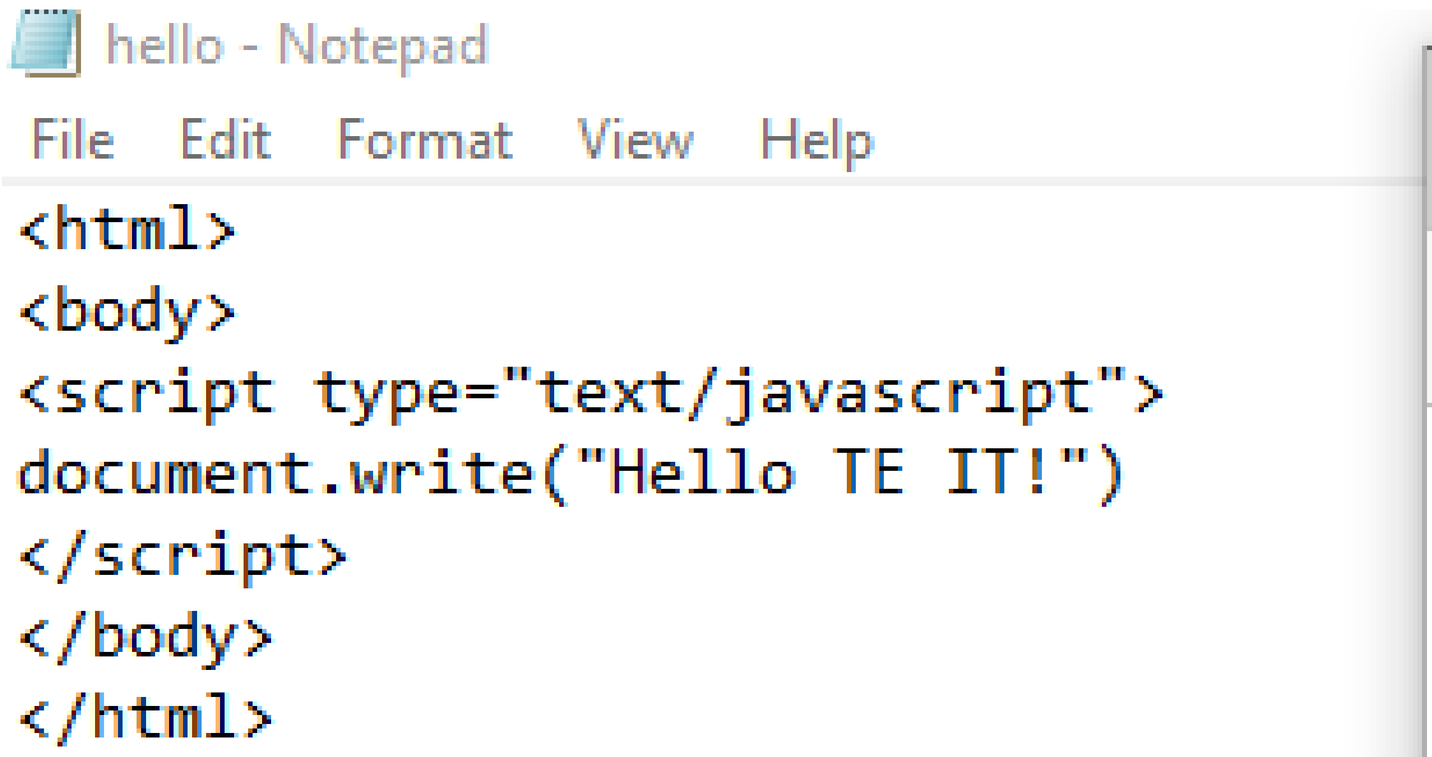
4.Curly braces({and }) are used to indicate a block of code. The code inside the braces is treated as one block.

- 5. Semicolons are optional in JavaScript , but it is a good practice to use semicolons in the document because this makes the code easier to read and debug. A semicolon is used to define the end of a statement. Although you can put many statements in a single line, it is better to put each statement on a separate line.
- The <script type="text/javascript"> tag shows the starting of the JavaScript code
- The </script> tag shows the end point of the JavaScript code
- The command document.write is the standard JavaScript command for writing the output to a page.

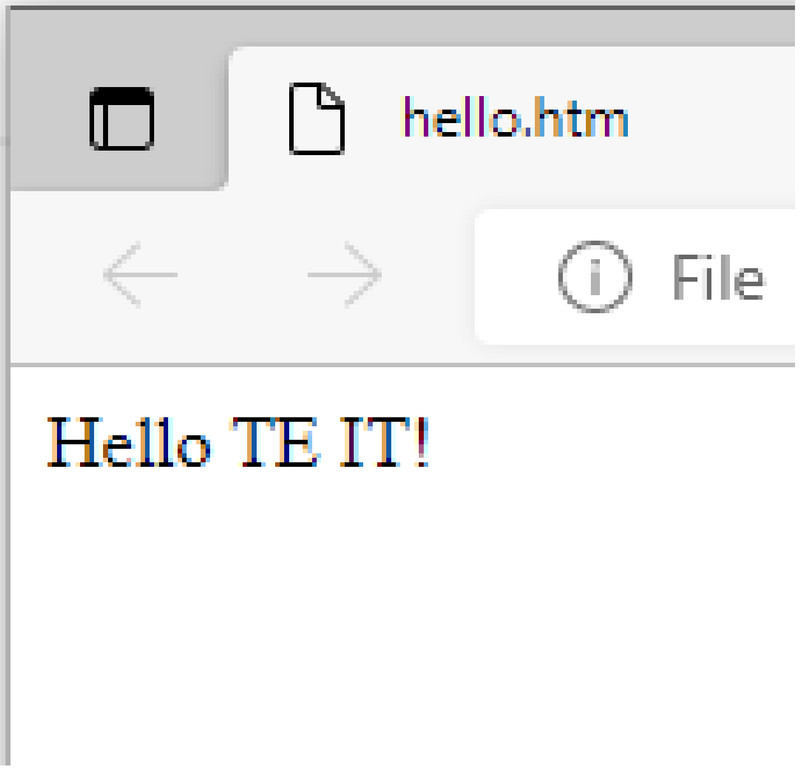
Syntax of inserting JavaScript in your web document

```
<html>
<body>
<script type="text/javascript">
…….
</script>
</body>
</html>
```

Example of JavaScript



```
<html>
<body>
<script type="text/javascript">
document.write("Hello TE IT!")
</script>
</body>
</html>
```



Hello TE IT!

Bachelor Of Engineering In Information Technology

Semester Six, Third Year(Even semester)
11th May 2022
19th Offline Lecture

Padre Conceicao College of Engineering

Verna Goa 403722 India

Data Types

- JavaScript provides the following data types for handling data.The data is stored in variables and the data type defines the kind of data which a variable can store

1. Number-it includes integers and floating –point numbers.eg 8
2. String-includes any group of one or more characters. A string type data is always shown by enclosing it in quotation marks.eg.”8”
3. Boolean-true or false
4. Undefined-The undefined value occurs only when a variable is declared ,but has not been assigned any value.
5. Null-A null value means that the object in the question doesn’t exist.

Reference data types

- Object and Function type.
- They contain the reference to the memory location that holds the particular data.
- You don’t need to declare a data type, while declaring or initializing a variable in JavaScript . JavaScript uses dynamic typing, i.e. the data type is inferred by the context of the JavaScript statement. This facilitates the use of the same variable for different data types and at different times.
- Var num = “lucy”; // num contains a string
- Num=9;   // num contains a number

Variables

- A variable can be referred as the container for storing information. This information can be of any type available in JavaScript. The value of a variable can change during the script.The lifetime of a variable depends on a number of factors. But the instant a web page clears the window, any variable it knows about are discarded.

Operators

- Operators enable you to perform an action on a variable.
- The basic JavaScript operators include assignment ,arithmetic , comparison , logical, and increment/decrement operators.

1.Assignment operators: used to assign a value to a variable

```
var mycolor=“orange”;
```

2.Arithmetic operators: +,-,*,/

```
Var x=1;
Var y=2;
Var z=x+y; //z=3
x=“TE”;
Y=“IT”;
Z=x+y //z=”TE IT”
```

- JavaScript includes a Modulus (%) operator, also known as the Remainder operator. It is used to calculate the value of the remainder after a number is divided by another number

```
Var myMod=11%2;  //myMod=1
```

Comparison operator - Comparison operators are used to evaluate an expression and return a Boolean value (true or false) indicating whether the comparison is true or false.They are <,>,<=,>=,==,!=

If a=2 and b=4,then the expression a>b is false

Logical operators- AND(&&),OR(||) and NOT (!).These operators also return a Boolean value(true or false).The AND operator returns true only if all operands are true. The OR operator returns true if any of the operand is true. The NOT operator always returns the value opposite to that of the operand

- Increment and decrement operators- they provide a shortcut for adding or subtracting 1 from a value. They can appear before a variable (prefix) or after a variable (postfix).
- If a variable with a prefix operator is used in an expression ,the value is incremented or decremented before the expression is evaluated. If a postfix operator is used, the value is incremented or decremented after the expression is evaluated.

```
var a=1;
a++;        //a=2
var b=12
b--;   //b=11
```

Conditional statements and Loops

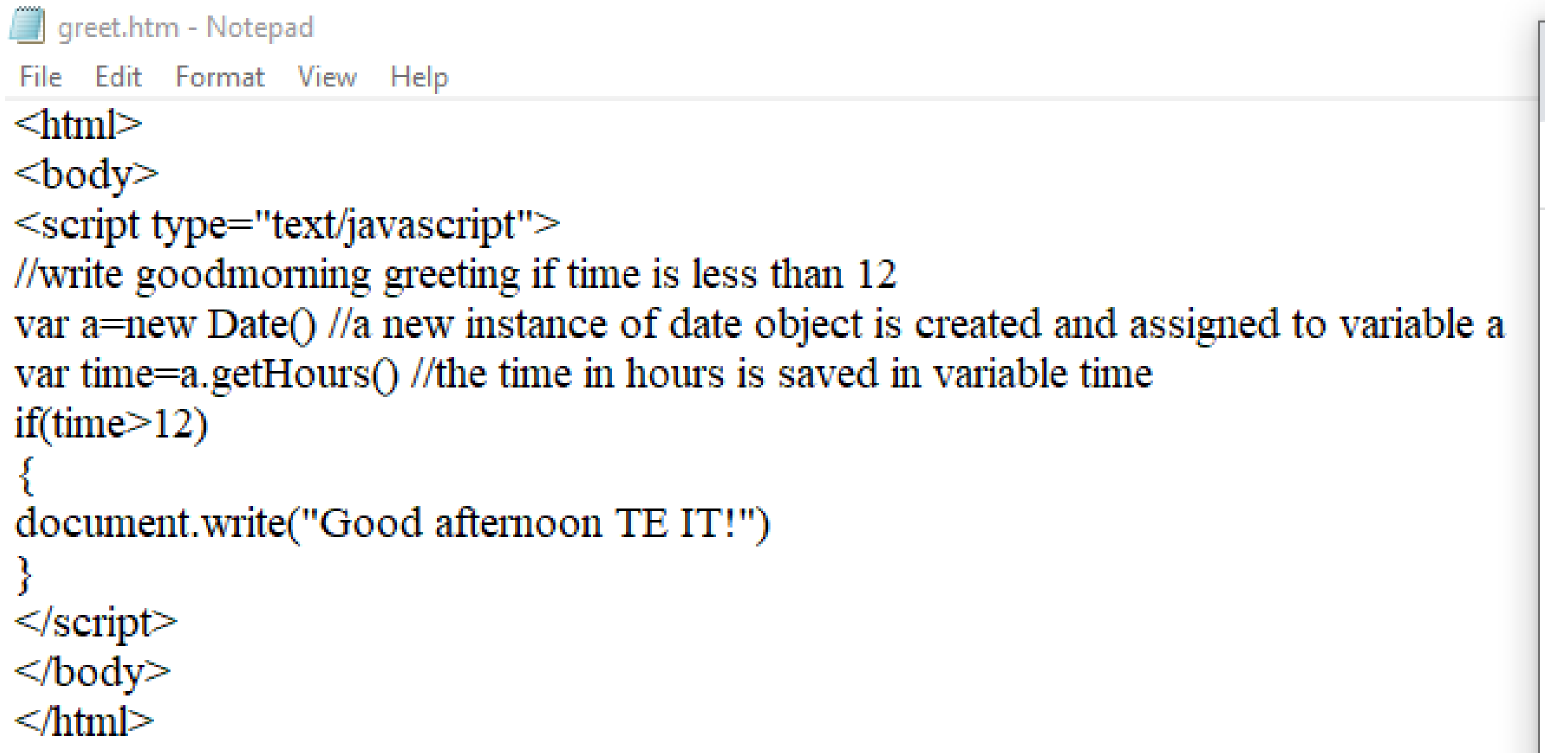
- A JavaScript program is composed of statements.
- Variable declarations, assignments and initializations are examples of JavaScript statements.
- JavaScript also includes a core set of programming statements similar to those used in other programming languages – conditionals(if/else , switch) and loops(for , while)

If statement

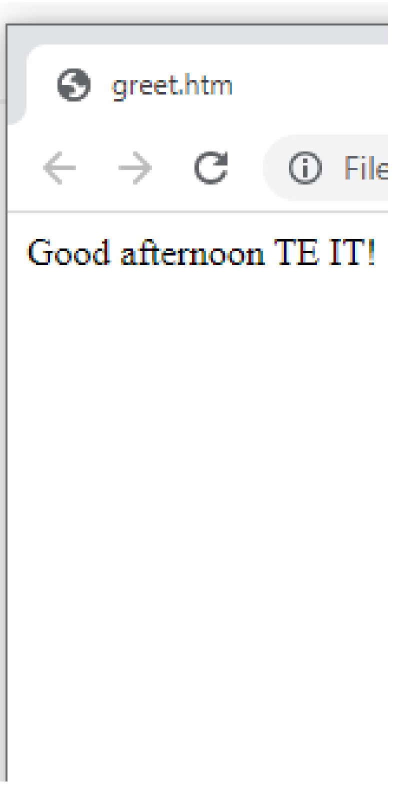
- An if statement evaluates the Boolean value of an expression (true or false), and then executes the code based on the result of that evaluation. If the expression in the parentheses evaluates to true, the block of code contained in the curly braces({ }) following the expression is executed. The syntax of if statement is given here.

```
if(condition to be checked)
{
Code to be executed if the condition is true
}
```

Example of If statement



```
<html>
<body>
<script type="text/javascript">
//write goodmorning greeting if time is less than 12
var a=new Date() //a new instance of date object is created and assigned to variable a
var time=a.getHours() //the time in hours is saved in variable time
if(time>12)
{
document.write("Good afternoon TE IT!")
}
</script>
</body>
</html>
```



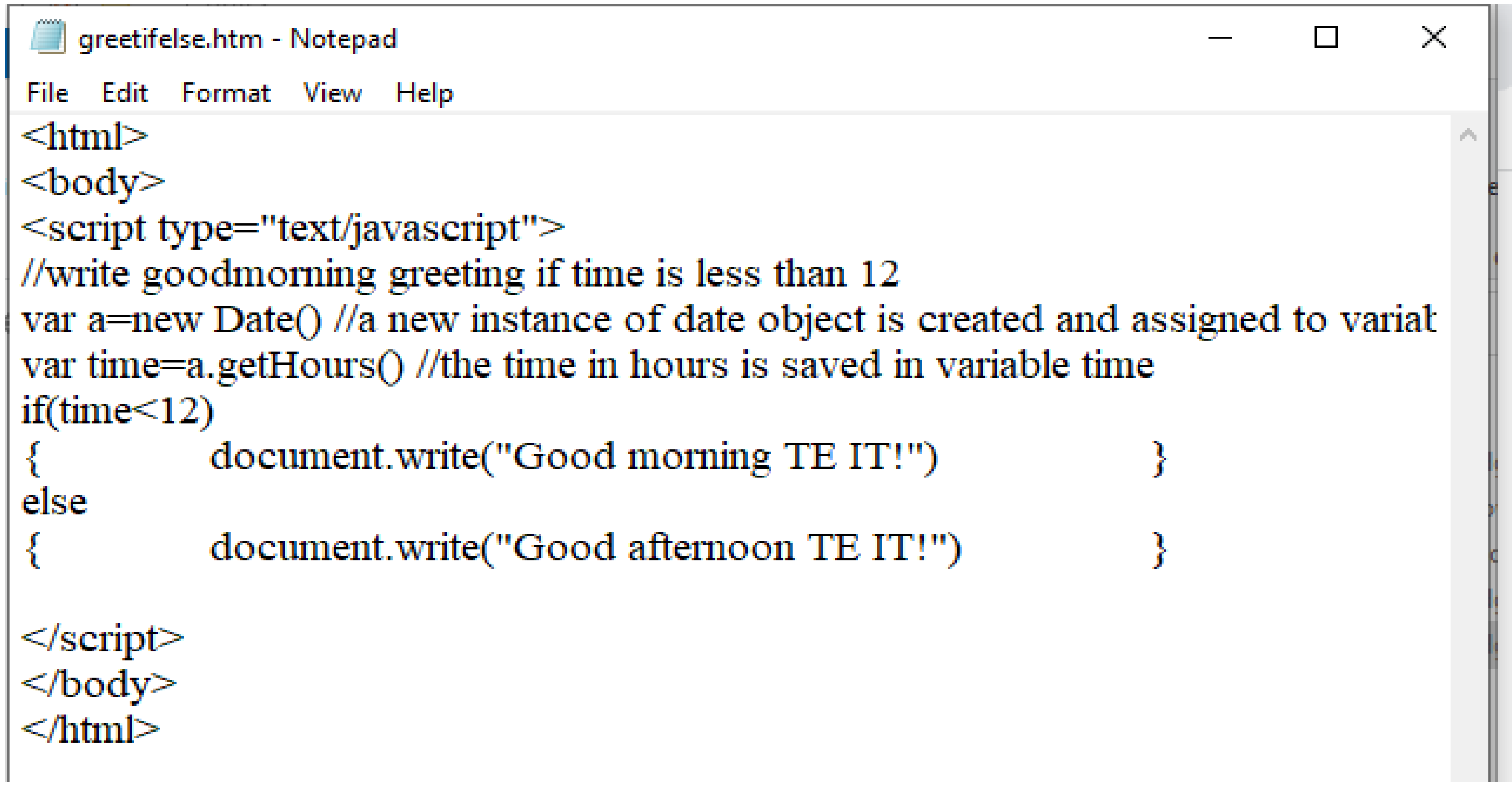
```
Good afternoon TE IT!
```

If else statement

- The if-else statement is used if you want to execute some code when the condition is true and the other code if the condition is false.
- The syntax of the if-else statement is as follows

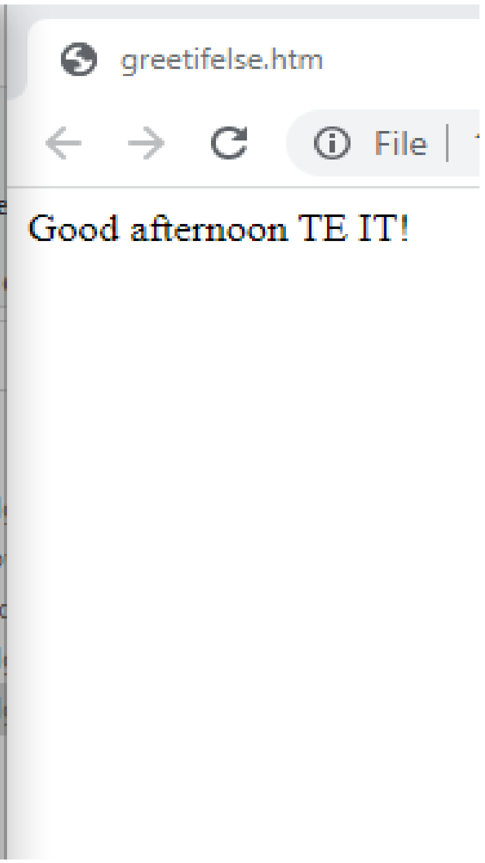
```
if(condition)
{ code to be executed if the condition is true}
else
{ code to be executed if the condition is true}
```

Example of if else statement



```
<html>
<body>
<script type="text/javascript">
//write goodmorning greeting if time is less than 12
var a=new Date() //a new instance of date object is created and assigned to variat
var time=a.getHours() //the time in hours is saved in variable time
if(time<12)
{           document.write("Good morning TE IT!")           }
else
{           document.write("Good afternoon TE IT!")         }

</script>
</body>
</html>
```



```
Good afternoon TE IT!
```

Switch statement

- A switch statement can be used instead of a series of if statements.
- Syntax is as follows

```
Switch(n){
Case 1:
        Execute code block 1
        Break
Case 2:
        Execute code block 1
        Break
Default:
        Code to be executed if n is different from case 1 and 2
}
```

Switch statement

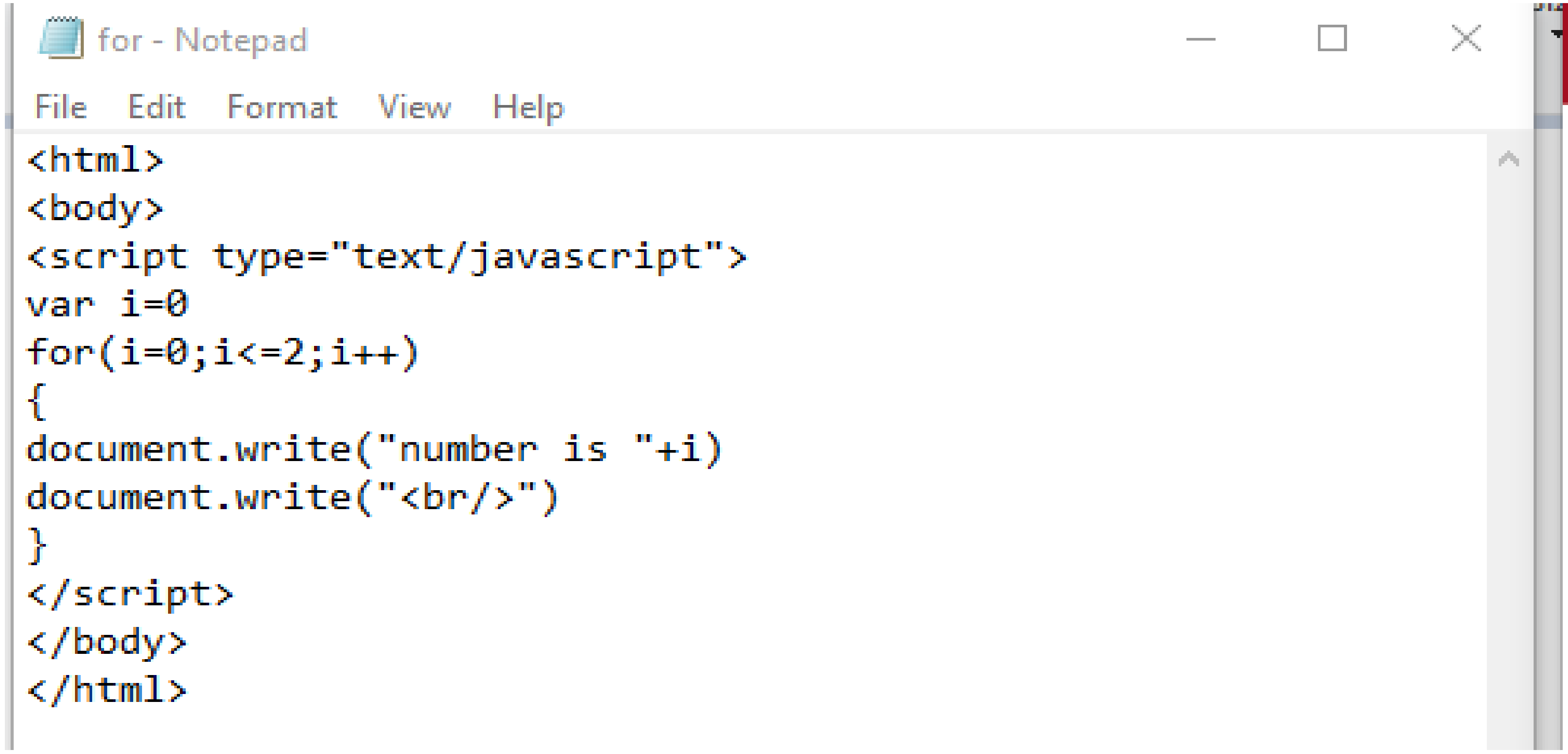
- First we have a single expression n (most often a variable), that is evaluated once. The value of the expression is then compared with the values for each case in the structure. If there is a match, the block of code associated with that case is executed. Use break to prevent the code from running into the next case automatically. The default case plays a similar role to an else statement. If there is not a specific case for a value, the default is used.

For Loop

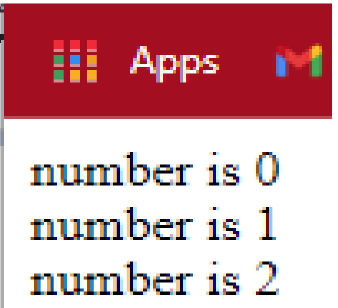
- For loop is used when you know in advance how many times the script should run. A for loop includes three components, like initialization, test condition, and iterator.
- The syntax of a for loop is

```
for (var= startvalue ; var<= endvalue ; var = var + increment)
{
Code to be executed
}
```

For Loop Example



```
<html>
<body>
<script type="text/javascript">
var i=0
for(i=0;i<=2;i++)
{
document.write("number is "+i)
document.write("<br/>")
}
</script>
</body>
</html>
```



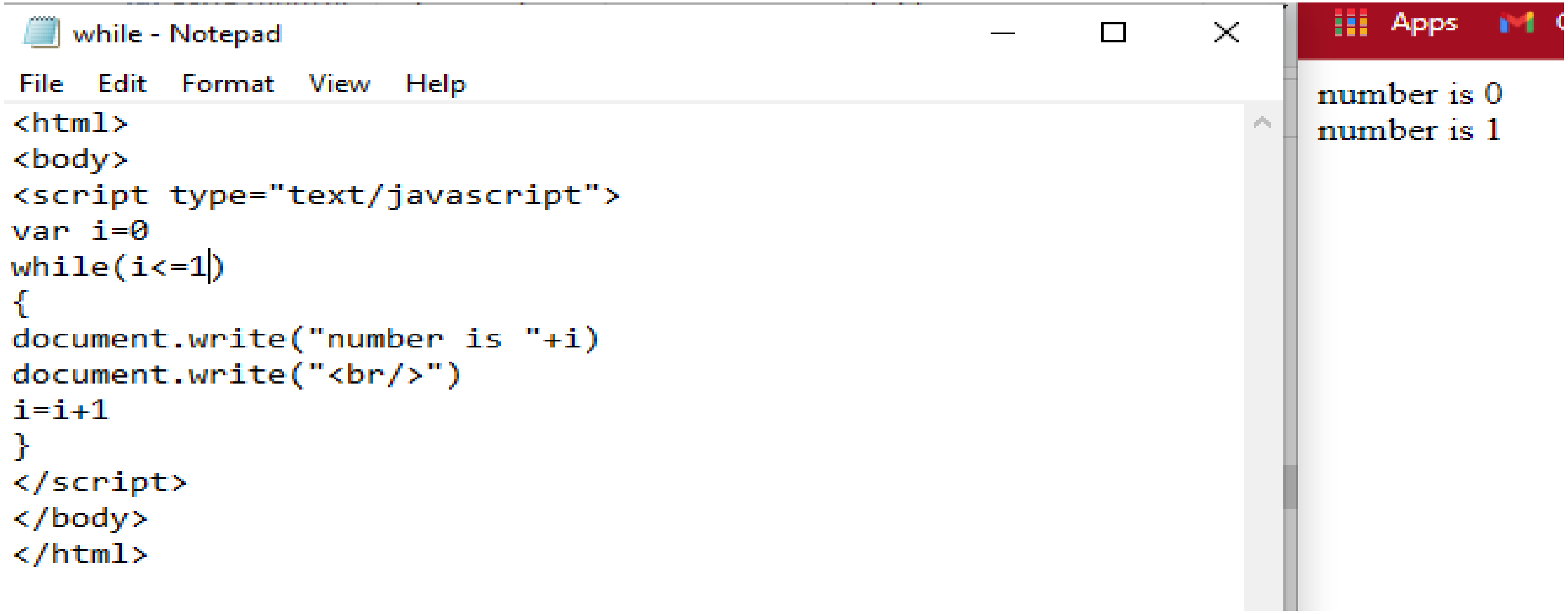
```
number is 0
number is 1
number is 2
```

While Loop

- A while loop is used when you want the loop to execute and continue executing while the specified condition is true.
- The syntax of a while loop is as follows

```
while (var <= endvalue)
{
Code to be executed
var++
}
```

while Loop example



```
<html>
<body>
<script type="text/javascript">
var i=0
while(i<=1)
{
document.write("number is "+i)
document.write("<br/>")
i=i+1
}
</script>
</body>
</html>
```

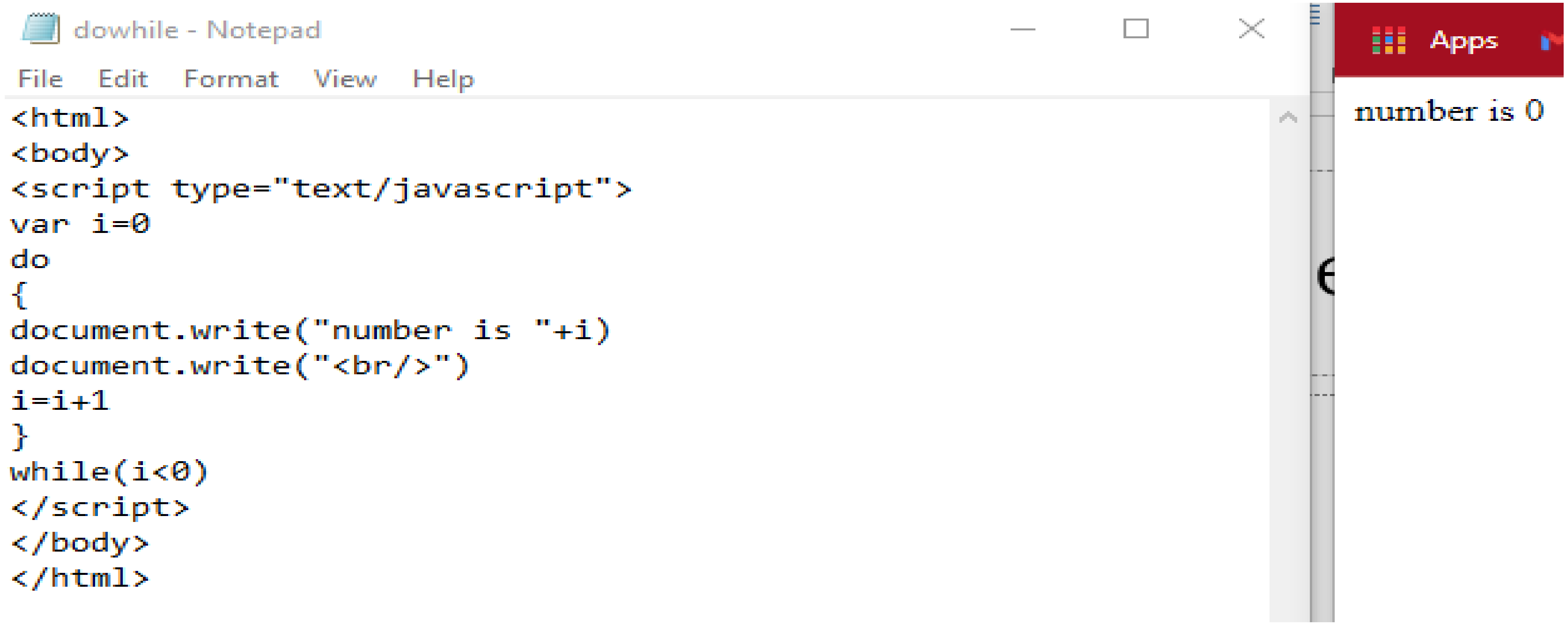
```
number is 0
number is 1
```

Do while loop

- The do while loop is a variant of the while loop.
- This loop will always execute a block of code once, and then it will repeat the loop as long as the specified condition is true.
- This loop will always be executed at least once, even if the condition is false, because the code is executed before the condition is tested.
- The syntax is

```
do
{
Code to be executed
var ++
}
While (var <= endvalue)
```

Do while loop example



```
<html>
<body>
<script type="text/javascript">
var i=0
do
{
document.write("number is "+i)
document.write("<br/>")
i=i+1
}
while(i<0)
</script>
</body>
</html>
```

```
number is 0
```

Functions

- A function is a reusable group of code statements that are treated as a unit.
- Javascript includes many built-in functions, and you can also create your own functions for code blocks that you want to reuse.
- Function definitions are usually included in script blocks in the head section of an HTML page. A function must be defined before it is used. Because the code in the head section of the page is read and interpreted before the code in the body section, functions are usually defined in the head section and called from the body section.

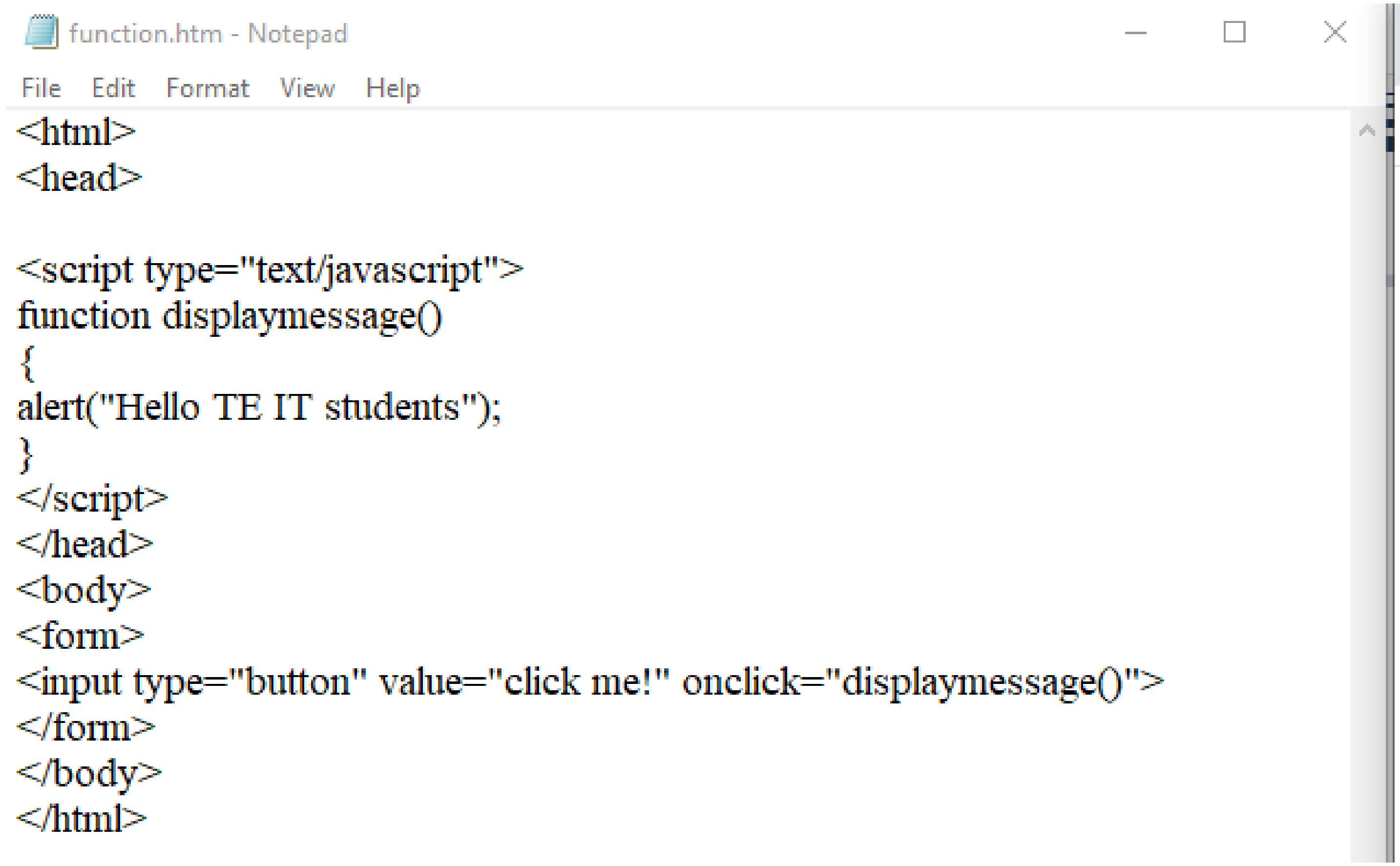
Syntax

- The syntax for creating a function is as follows

```
Function functionname(var1 , var2, ...,varx)
{
Some code
}
```

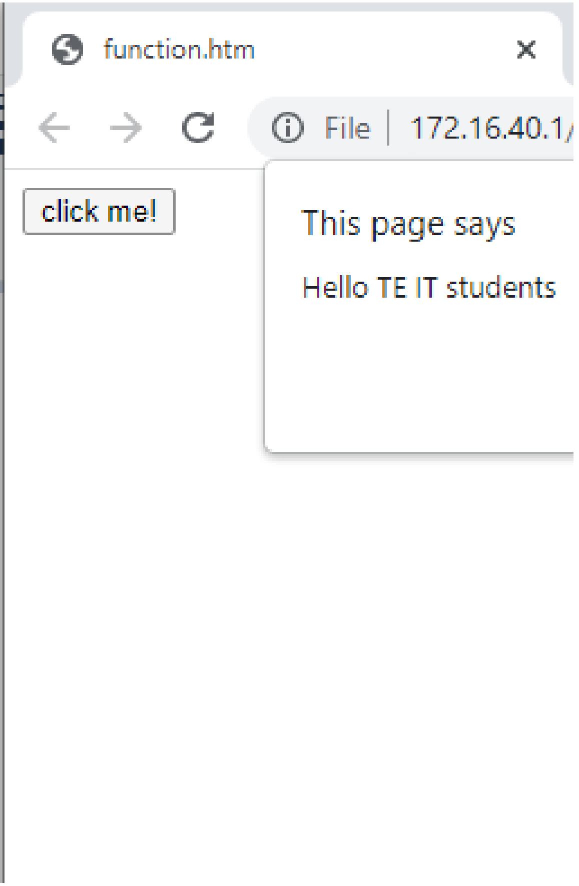
Here,var1,var2, etc. are the variables passed to the function. The curly brackets { and } defines the start and end of the function.

Example of function



```
<html>
<head>

<script type="text/javascript">
function displaymessage()
{
alert("Hello TE IT students");
}
</script>
</head>
<body>
<form>
<input type="button" value="click me!" onclick="displaymessage()">
</form>
</body>
</html>
```

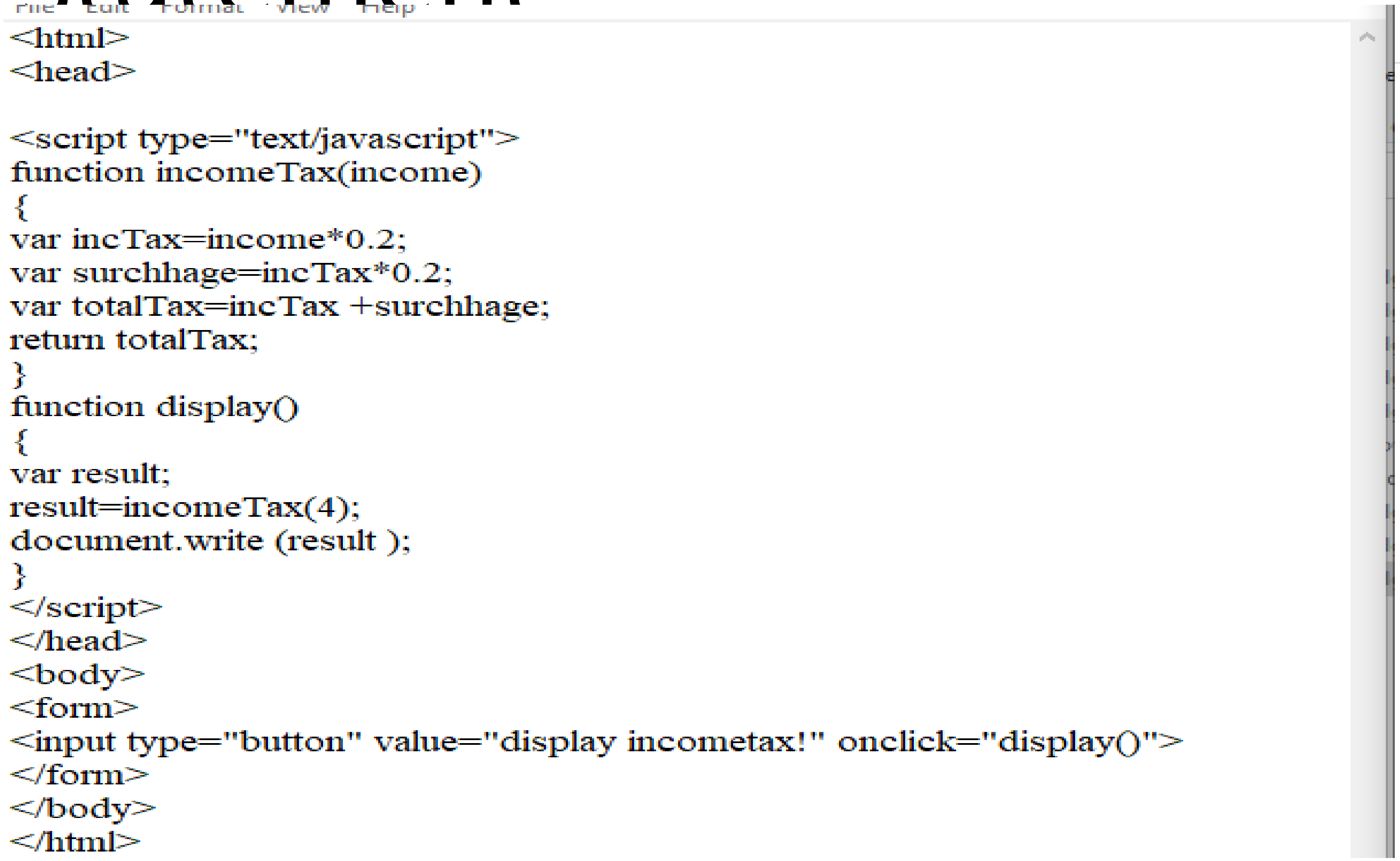


```
This page says
Hello TE IT students
```

Example of function with return statement

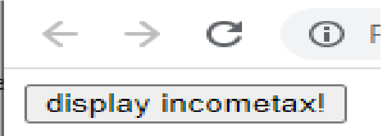
```
function incomeTax(income)
{
var incTax=income*0.2;
var surchhage=incTax*0.2;
var totalTax=incTax +surchhage;
return totalTax;
}
```

Example of function with return statement



```
<html>
<head>

<script type="text/javascript">
function incomeTax(income)
{
var incTax=income*0.2;
var surchhage=incTax*0.2;
var totalTax=incTax +surchhage;
return totalTax;
}
function display()
{
var result;
result=incomeTax(4);
document.write (result );
}
</script>
</head>
<body>
<form>
<input type="button" value="display incometax!" onclick="display()">
</form>
</body>
</html>
```



Bachelor Of Engineering
In
Information Technology

Semester Six, Third Year(Even semester)
16th May 2022
20th Offline Lecture

Padre Conceicao College of Engineering

Verna Goa 403722 India

JavaScript Objects

- An object is a collection of properties and methods that are grouped together with a single name.The objects available in the JavaScript can be divided into 3 categories.

1. Built-in objects
2. Browsers objects
3. User-defined objects

Built-in objects

- There are 9 built-in objects available in JavaScript.
- Built-in objects are those objects, which are available regardless of the content of window.
- These objects operate independently of whichever pages your browser has loaded.
- These objects are also called core language objects.

Array

- There are 3 ways to construct array in JavaScript

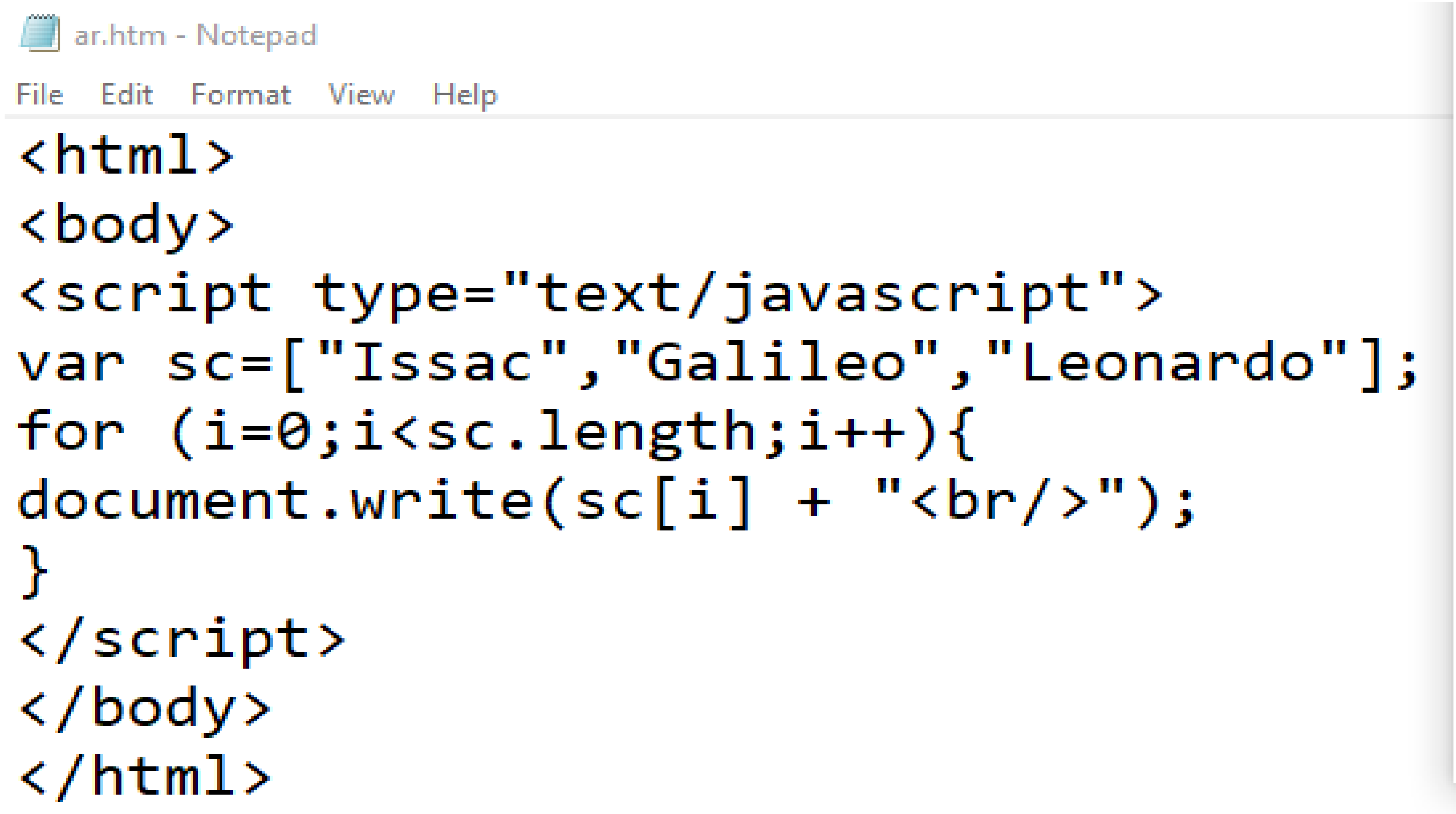
1. By array literal
2. By creating instance of Array directly (using new keyword)
3. By using an Array constructor (using new keyword)

JavaScript array literal

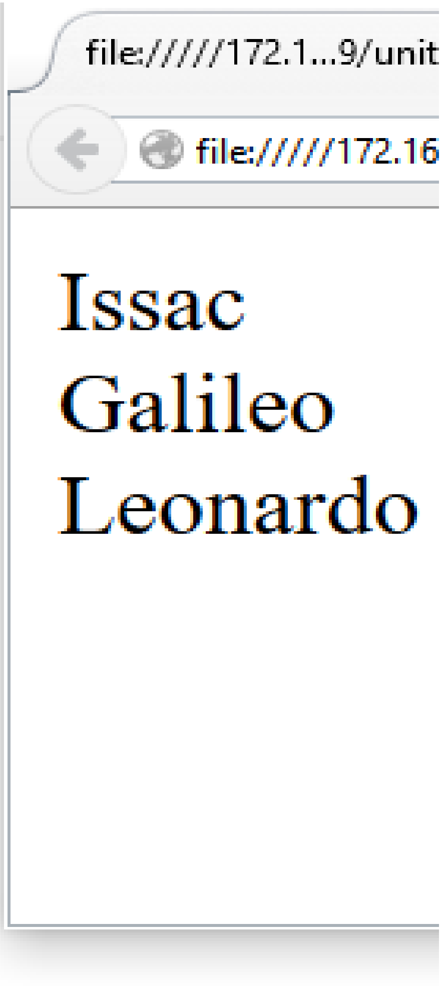
- The syntax of creating array using array literal is given below:

```
var arrayname=[value1,value2.....valueN];
```

Example of creating and using array in JavaScript



```
<html>
<body>
<script type="text/javascript">
var sc=["Issac","Galileo","Leonardo"];
for (i=0;i<sc.length;i++){
document.write(sc[i] + "<br/>");
}
</script>
</body>
</html>
```

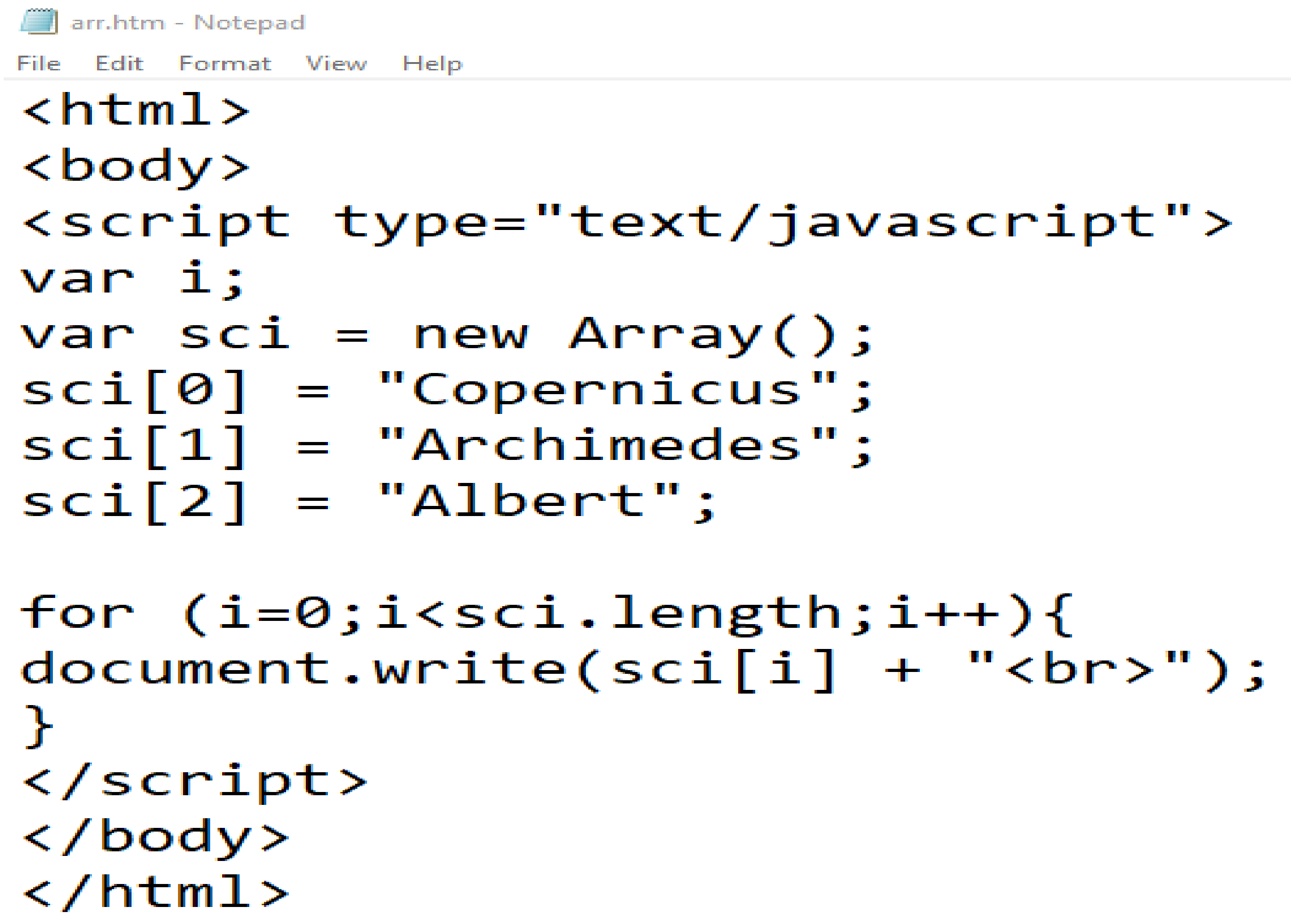


```
Issac
Galileo
Leonardo
```

JavaScript Array directly (new keyword)

- The syntax of creating array directly is given below:

var arrayname=new Array();

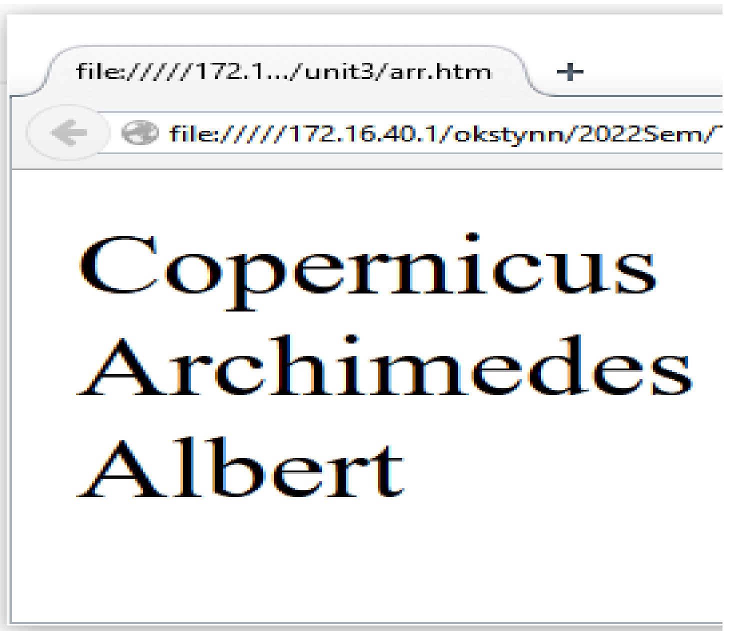


arr.htm - Notepad

File Edit Format View Help

```
<html>
<body>
<script type="text/javascript">
var i;
var sci = new Array();
sci[0] = "Copernicus";
sci[1] = "Archimedes";
sci[2] = "Albert";

for (i=0;i<sci.length;i++){
document.write(sci[i] + "<br>");
}
</script>
</body>
</html>
```



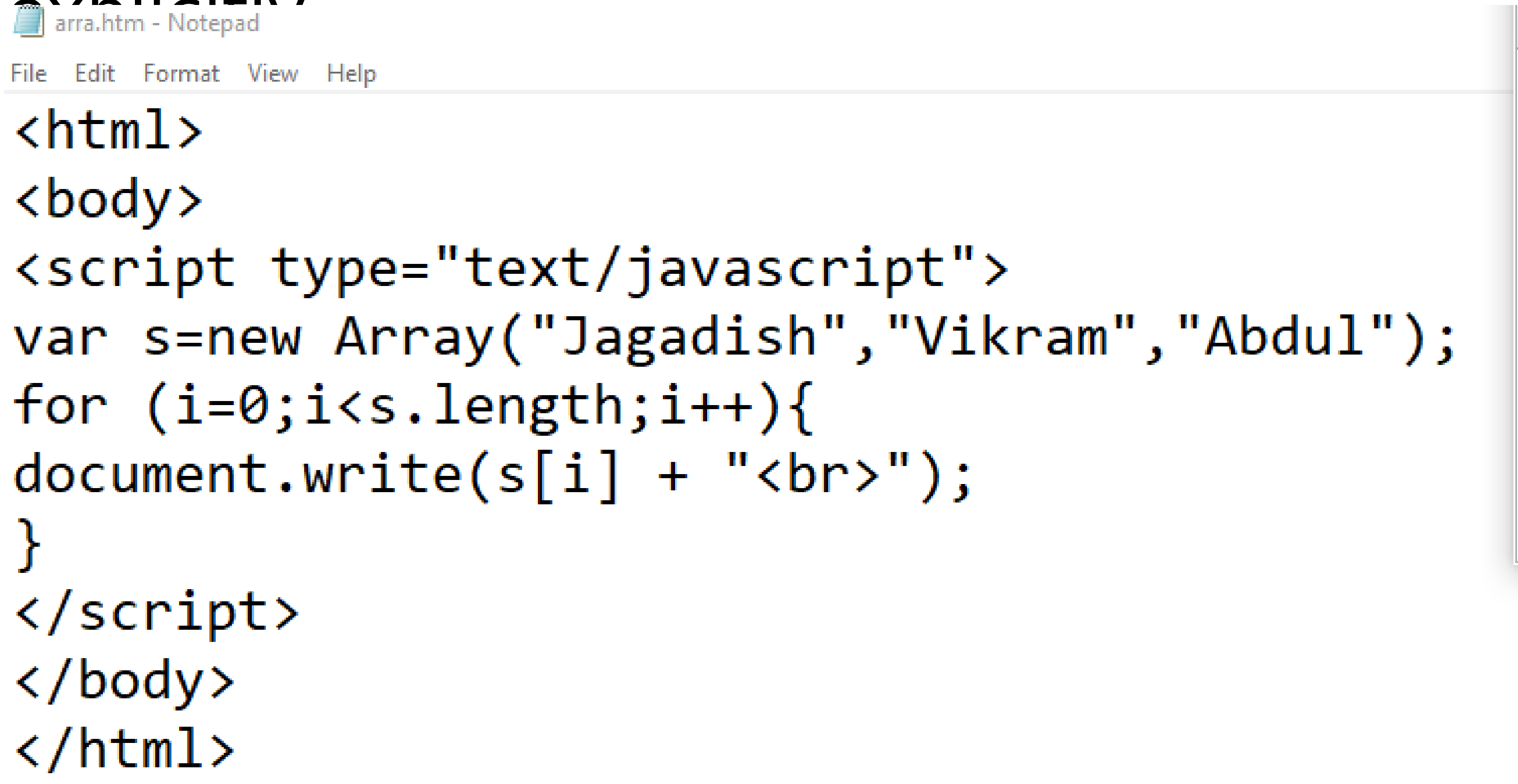
file://///172.1.../unit3/arr.htm

file://///172.16.40.1/okstynn/2022Sem/

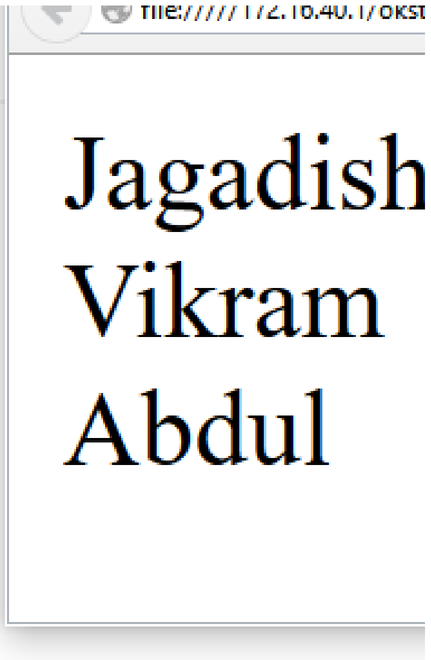
```
Copernicus
Archimedes
Albert
```

JavaScript array constructor (new keyword)

- you must create instance of array by passing arguments in constructor so that we don't have to provide value explicitly



```
<html>
<body>
<script type="text/javascript">
var s=new Array("Jagadish","Vikram","Abdul");
for (i=0;i<s.length;i++){
document.write(s[i] + "<br>");
}
</script>
</body>
</html>
```



```
Jagadish
Vikram
Abdul
```

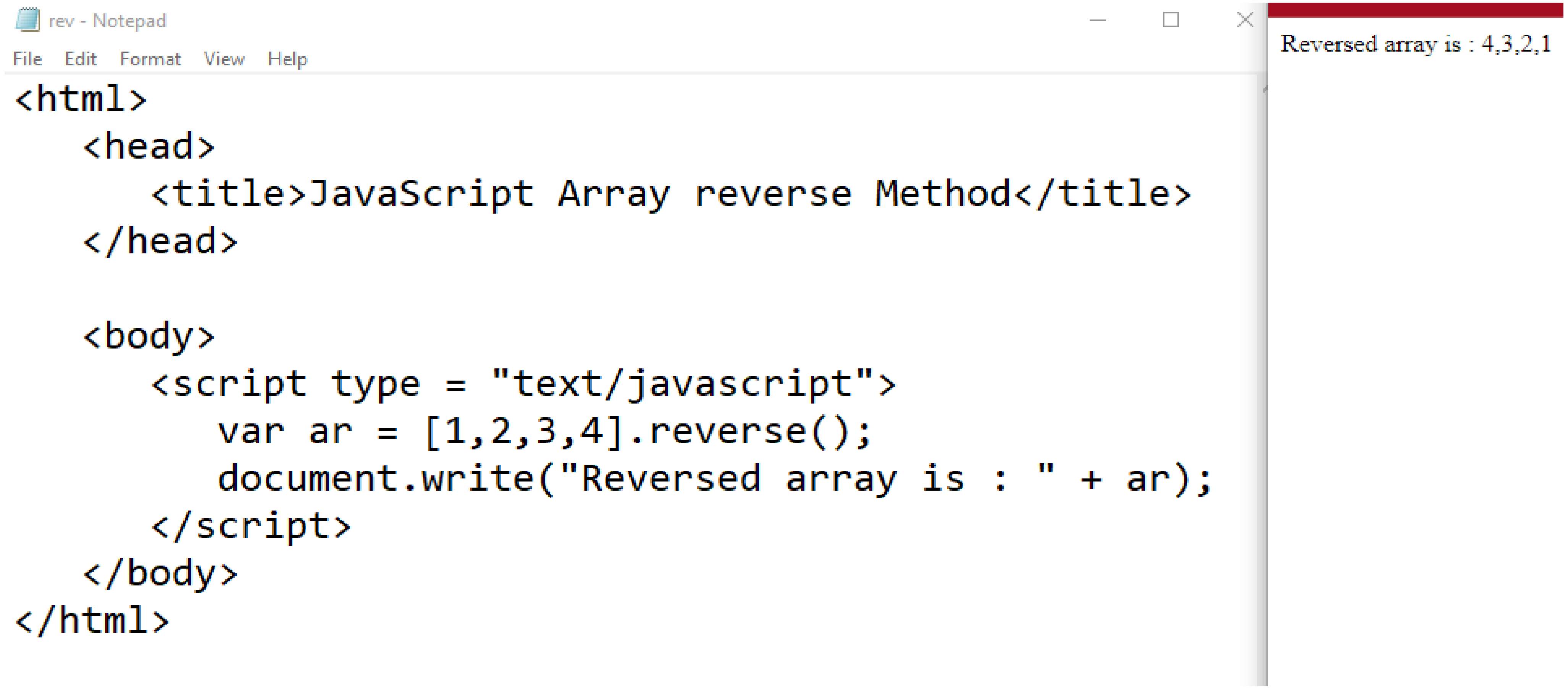
reverse() function

Methods are the specific functions that belong to a particular type of object.

The Array object, for example, includes a reverse() function that reverses the order of members of the array:

```
var backwards = myArray.reverse();
```

JavaScript Array reverse Method



```
<html>
   <head>
      <title>JavaScript Array reverse Method</title>
   </head>

   <body>
      <script type = "text/javascript">
         var ar = [1,2,3,4].reverse();
         document.write("Reversed array is : " + ar);
      </script>
   </body>
</html>
```

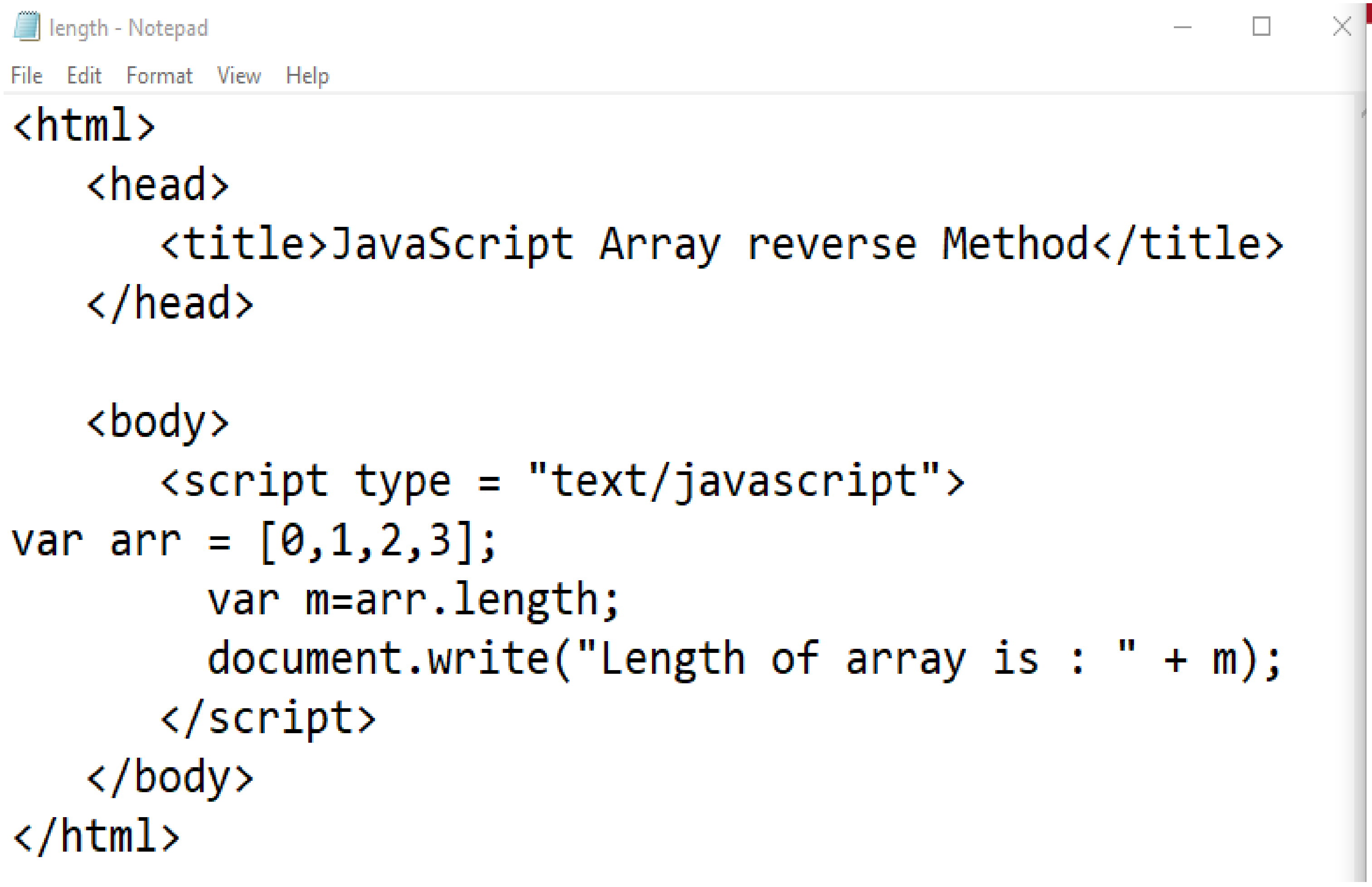
```
Reversed array is : 4,3,2,1
```

- Objects also include properties.
- Properties are attributes of an object, e.g. the Array object includes a length property:

```
 var howlong = myArray.length;
```

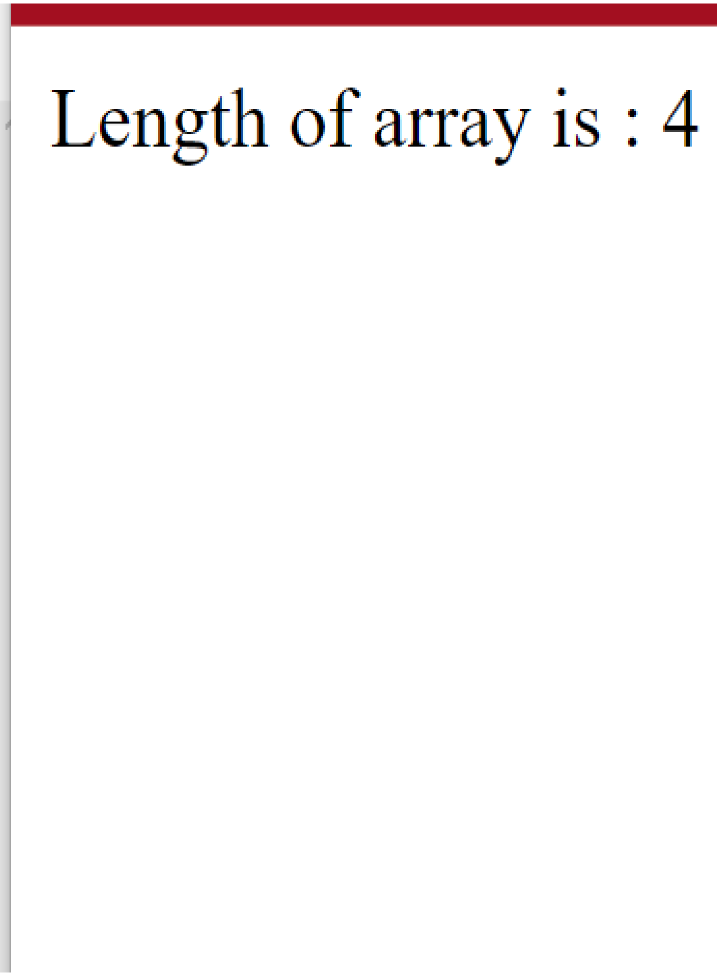
You can use the dot notation to either access the methods or properties of an object.

```
myArray.reverse();
myArray.length;
```



```
<html>
   <head>
      <title>JavaScript Array reverse Method</title>
   </head>

   <body>
      <script type = "text/javascript">
var arr = [0,1,2,3];
        var m=arr.length;
        document.write("Length of array is : " + m);
      </script>
   </body>
</html>
```



```
Length of array is : 4
```

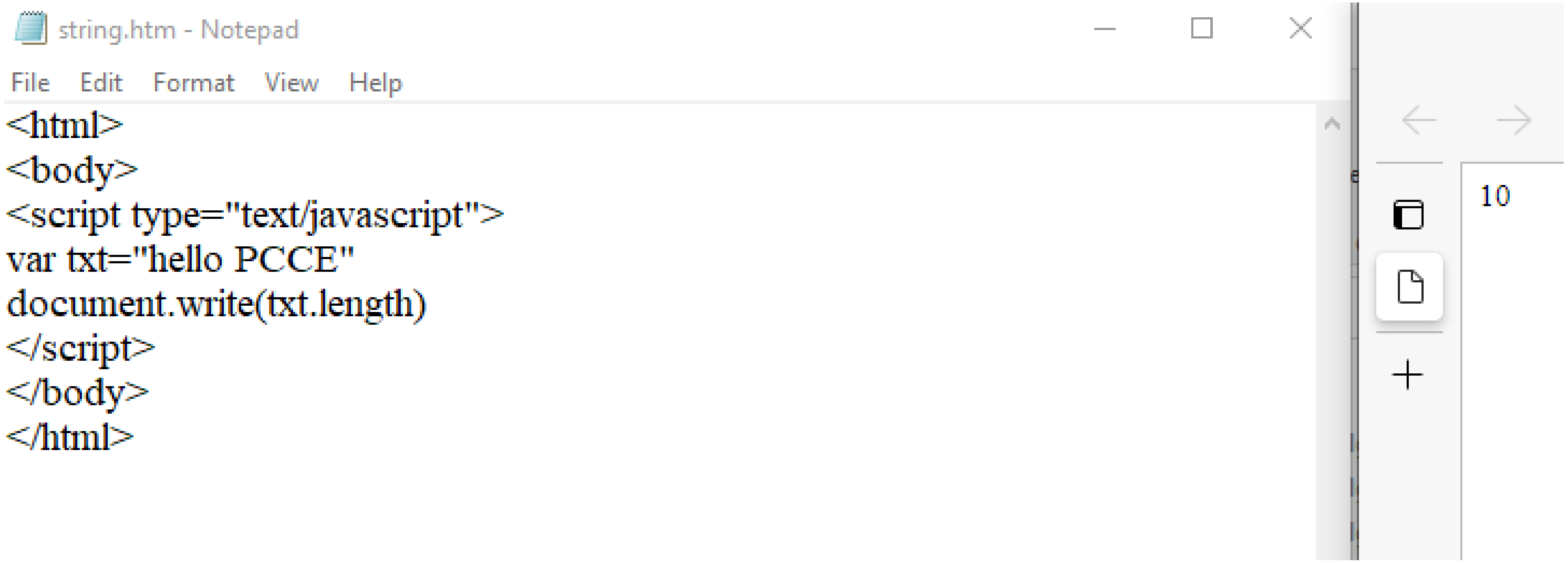
String

- The string object is used to manipulate a stored piece of text.The following example uses the length property of the string object to find the length of a string.

```
var txt=“hello PCCE”
document.write(txt.length)
```

The output is 10

length property of the string object



```
<html>
<body>
<script type="text/javascript">
var txt="hello PCCE"
document.write(txt.length)
</script>
</body>
</html>
```

```
10
```

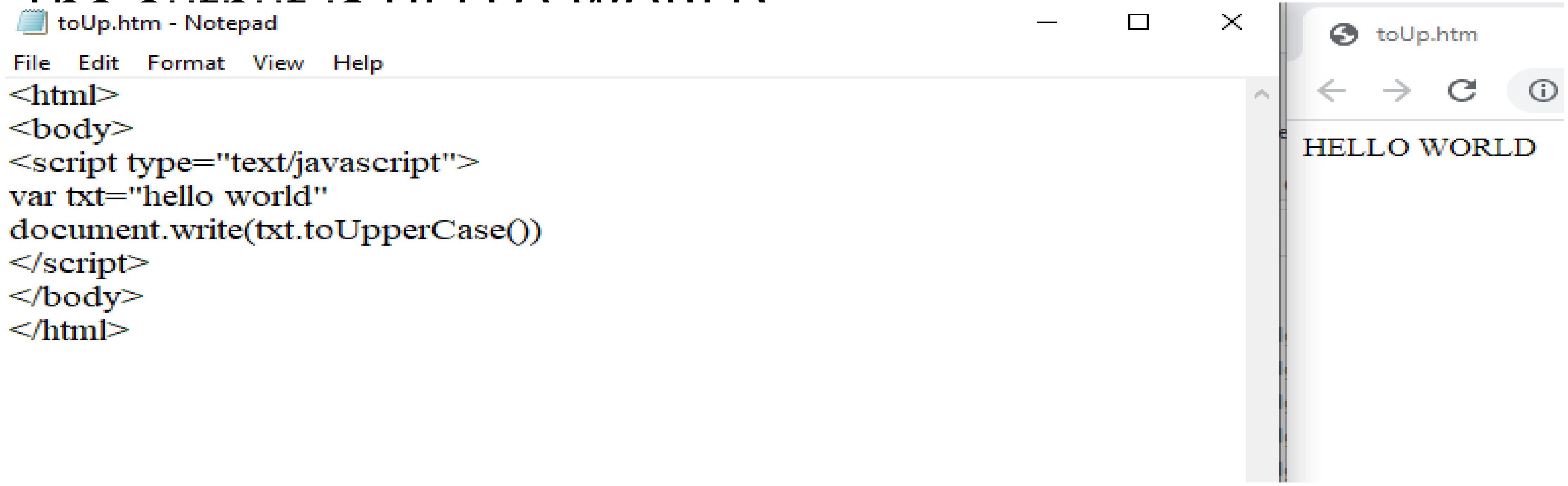
toUpperCase() method

The following example uses the toUpperCase() method of the string object to convert a string into uppercase letters:

var txt=“hello world”

Document.write(txt. toUpperCase())

The output is HELLO WORLD



```
<html>
<body>
<script type="text/javascript">
var txt="hello world"
document.write(txt.toUpperCase())
</script>
</body>
</html>
```

```
HELLO WORLD
```

Math

- The Math object includes many mathematical values and functions. This object allows you to perform common mathematical tasks. You do not need to define the math object before using it. JavaScript provides eight mathematical values that can be accessed from the Math object. You may reference these values from your JavaScript like this

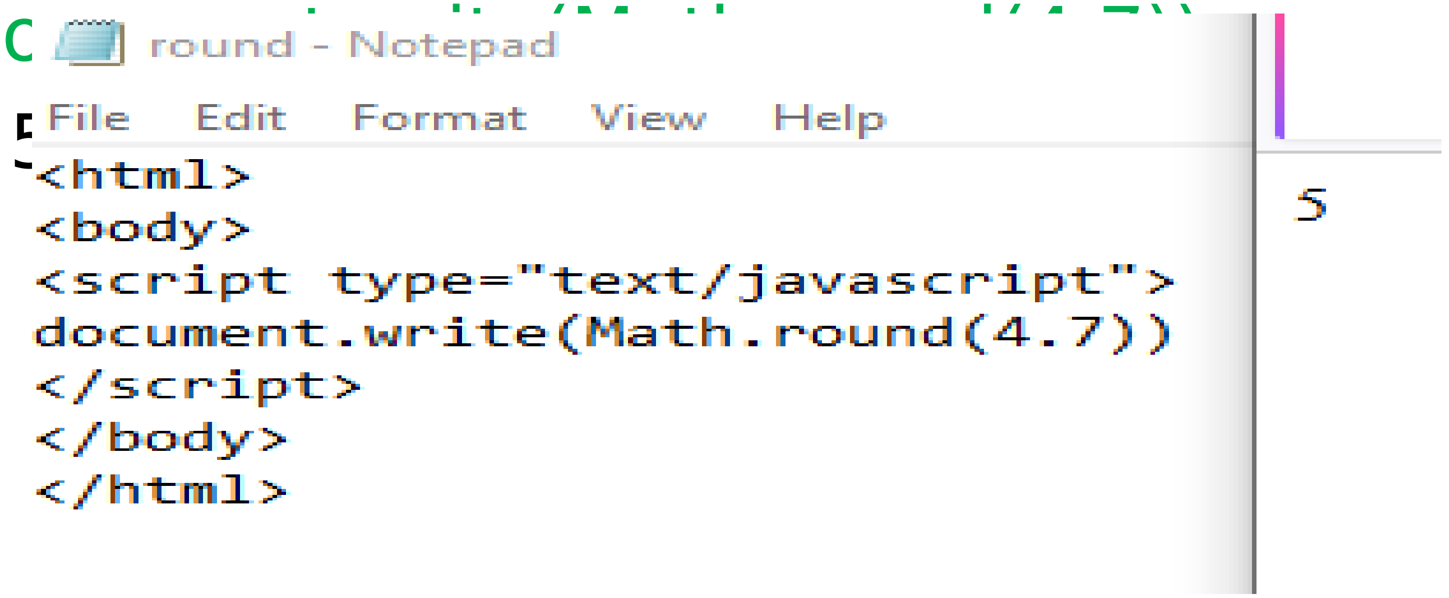
1. Math.E
2. Math.PI
3. Math.SQRT2
4. Math.SQRT1_2
5. Math.LN2
6. Math.LN10
7. Math.LOG2E
8. Math.LOG10E

round() method of the Math object

- Besides the mathematical values that can be accessed from the Math object, several functions are also available. The following example uses the round() method of the Math object to round a number to the nearest integer.

dc

The output is 5



round - Notepad

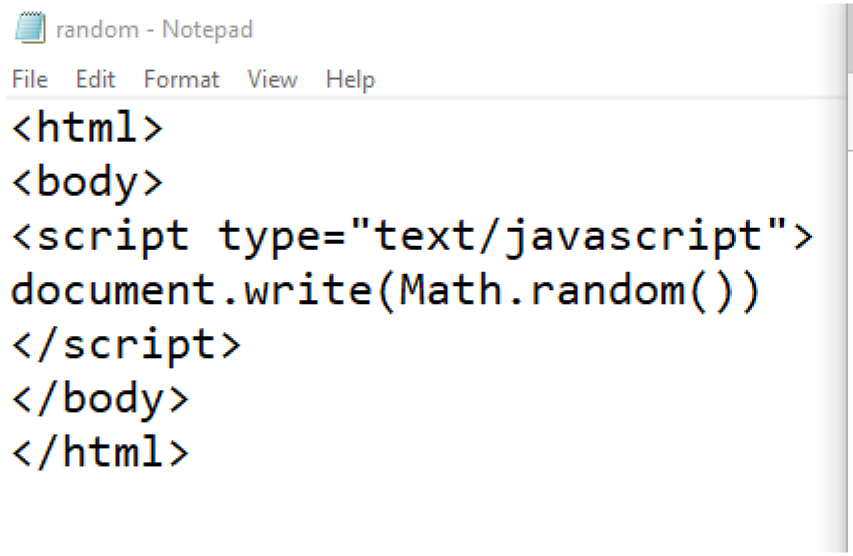
```
<html>
<body>
<script type="text/javascript">
document.write(Math.round(4.7))
</script>
</body>
</html>
```

```
5
```

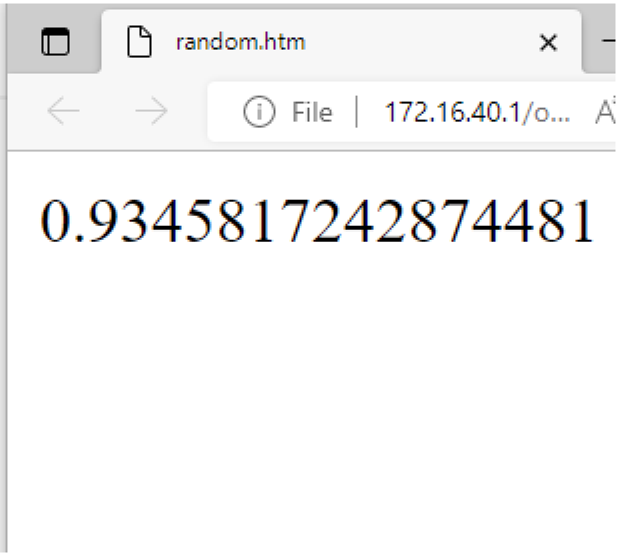
random() method of the Math object

The following example uses the random() method of the Math object to return a random number between 0 and 1

document.write(Math.random())



```
<html>
<body>
<script type="text/javascript">
document.write(Math.random())
</script>
</body>
</html>
```



```
0.9345817242874481
```

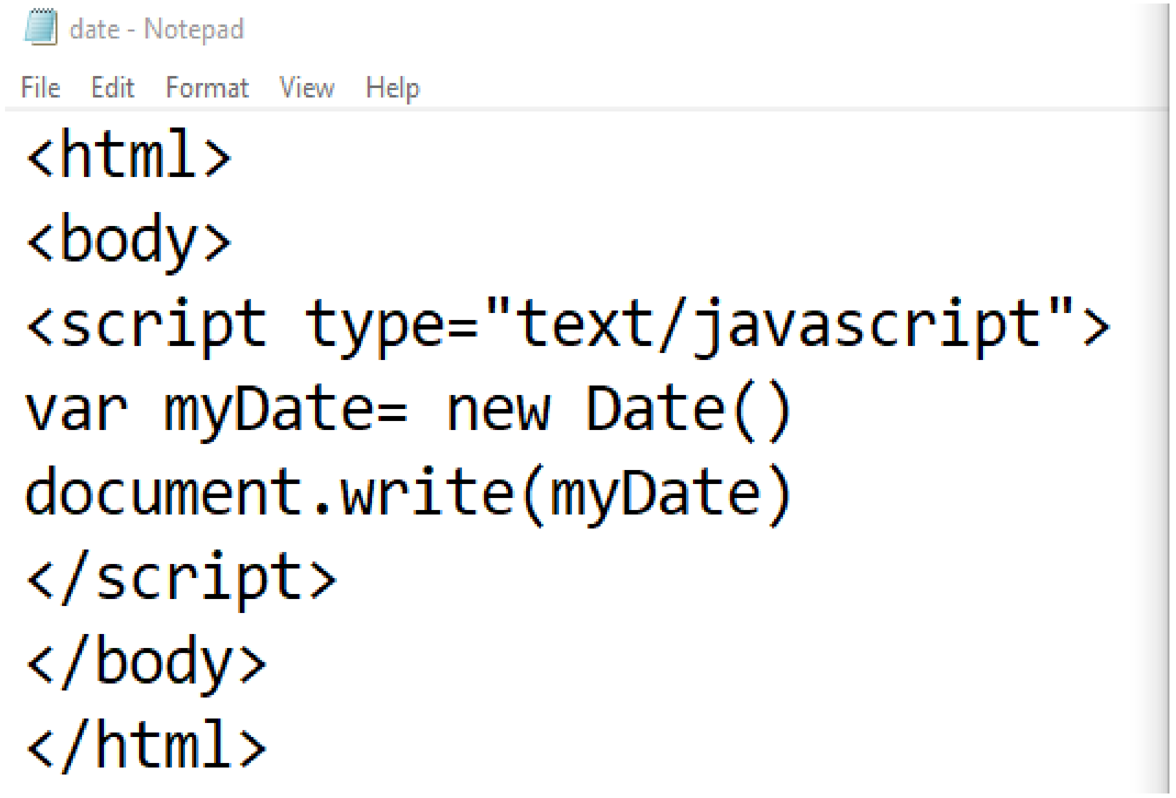
Date

- The Date object is used to work with dates and times. We define a Date object with the new keyword.
- The following code line defines a Date object called myDate

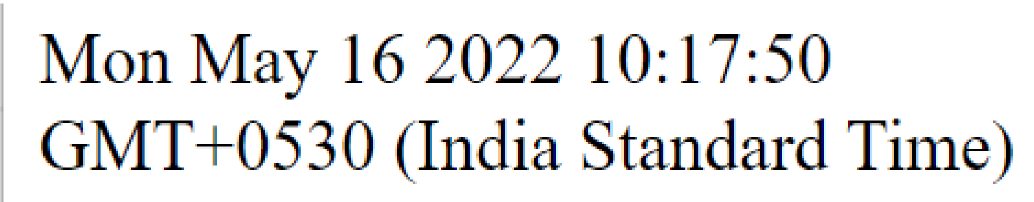
We can manipulate the date by using the methods available for the Date object.In the example here,we set a Date object to a specific date (29th January 2022)

```
var myDate= new Date()
myDate.setFullYear(2022,0,29)
```

The Date object



```
<html>
<body>
<script type="text/javascript">
var myDate= new Date()
document.write(myDate)
</script>
</body>
</html>
```

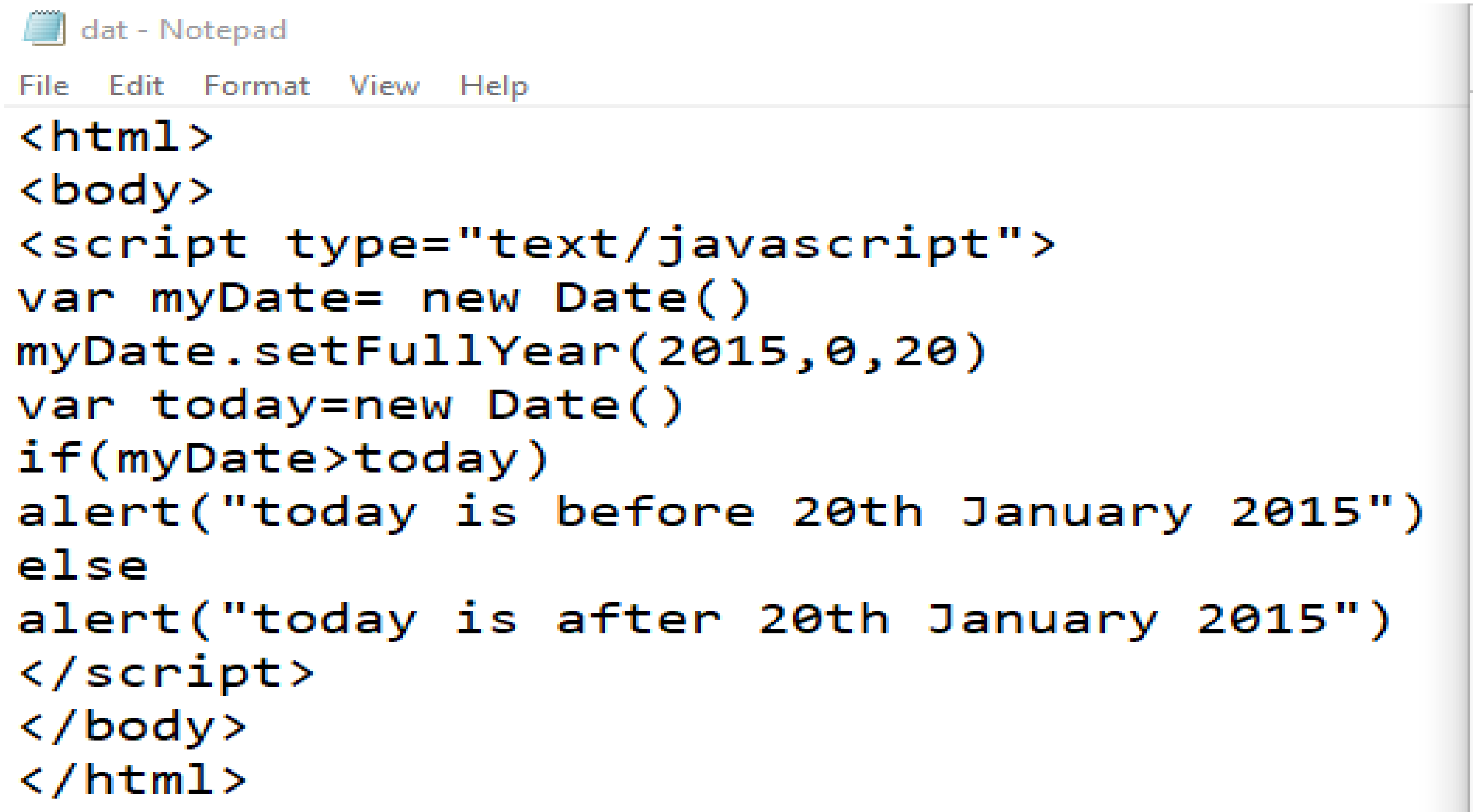


```
Mon May 16 2022 10:17:50
GMT+0530 (India Standard Time)
```

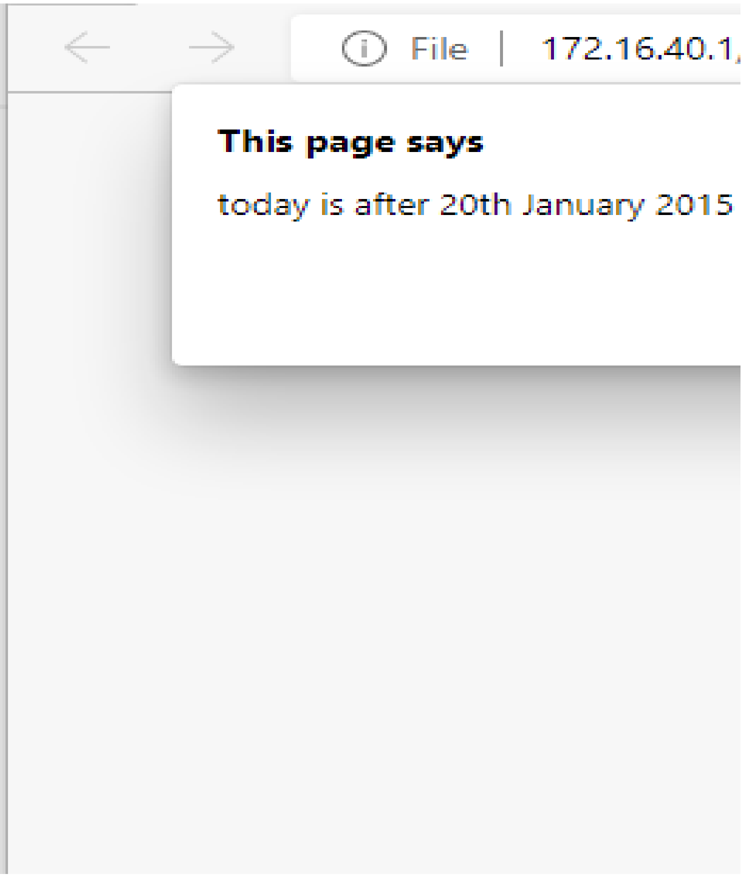
- The Date object is also used to compare two dates.

```
var myDate= new Date()
myDate.setFullYear(2015,0,20)
Var today=new Date()
If(myDate>today)
Alert(“today is before 20th January 2015”)
Else
Alert(“today is after 20th January 2015”)
```

To compare two dates



```
<html>
<body>
<script type="text/javascript">
var myDate= new Date()
myDate.setFullYear(2015,0,20)
var today=new Date()
if(myDate>today)
alert("today is before 20th January 2015")
else
alert("today is after 20th January 2015")
</script>
</body>
</html>
```



This page says

today is after 20th January 2015

Browser Objects

- The Browser Object Model (BOM) is a collection of objects that interact with the browser window.
- These objects include the following:

1. window object
2. history object
3. location object
4. navigator object
5. screen object
6. document object

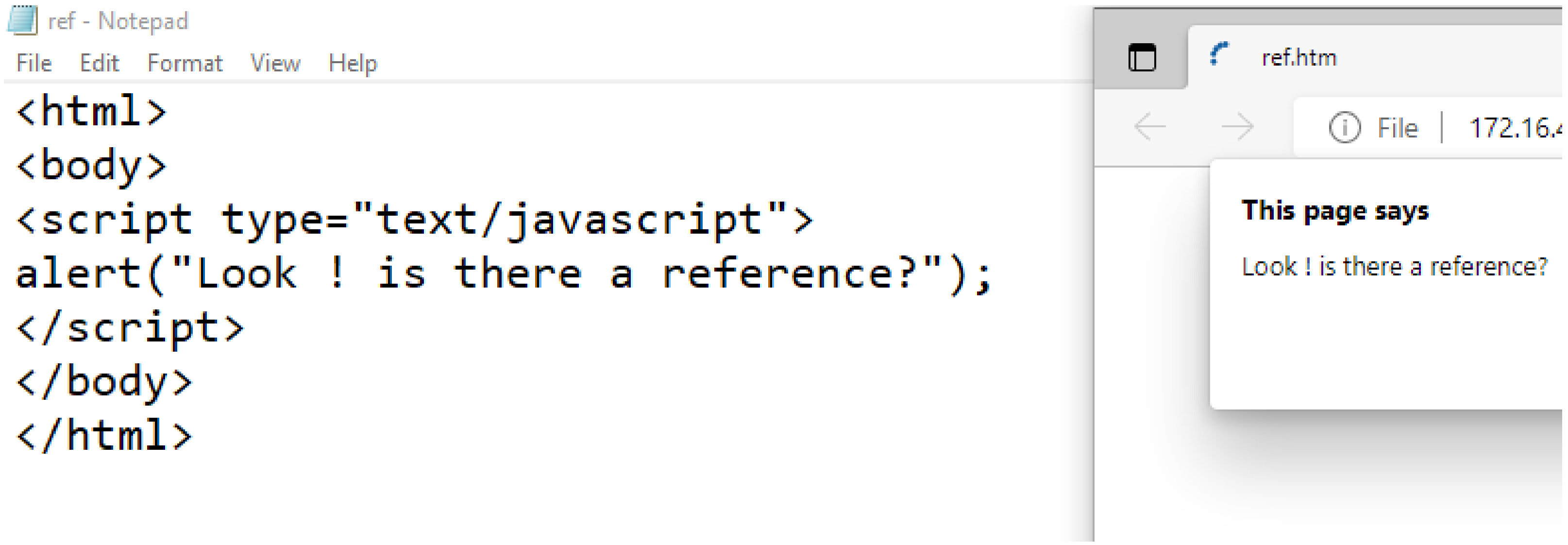
The window object

- The window object is the top object in the BOM hierarchy. The window object can be used to move and resize windows as well as create new windows. The window object also includes methods to create dialog boxes, such as the alert dialogs. The window object is a unique object – you don't have to explicitly include a reference to it when you use its methods or properties.
- Consider the following example:

```
alert(“Look ! is there a reference?”);
```

This function is exactly the same as the following

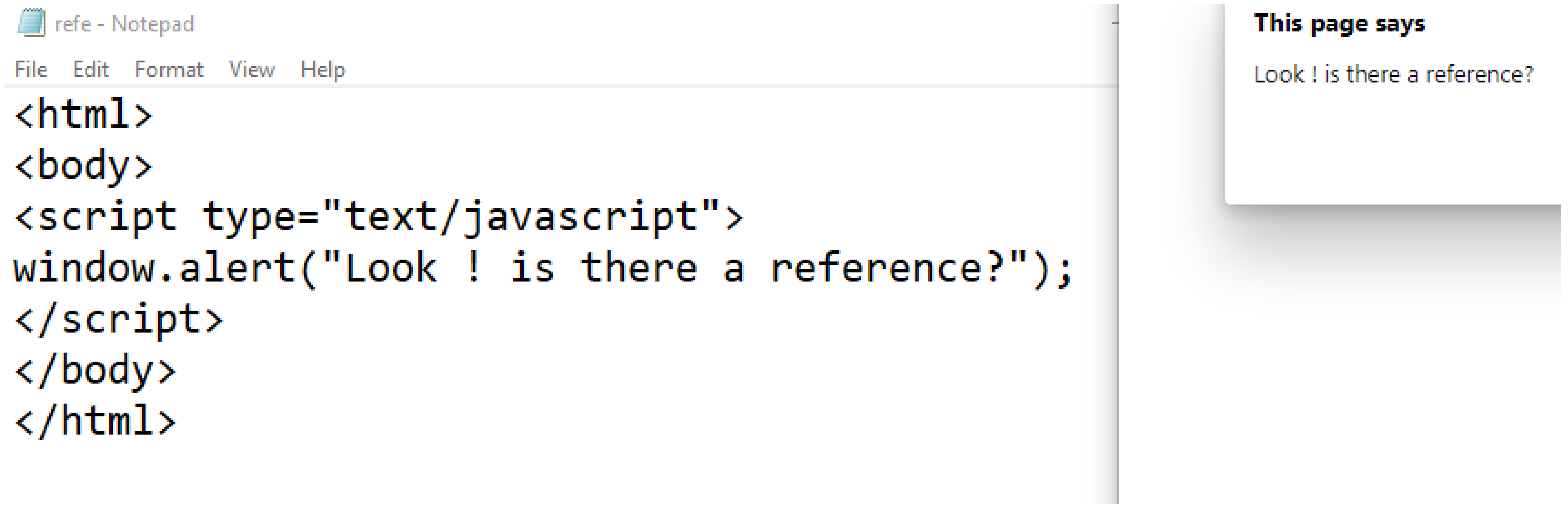
```
window.alert(“Look ! is there a reference?”);
```



```
<html>
<body>
<script type="text/javascript">
alert("Look ! is there a reference?");
</script>
</body>
</html>
```

This page says

Look ! is there a reference?



```
<html>
<body>
<script type="text/javascript">
window.alert("Look ! is there a reference?");
</script>
</body>
</html>
```

This page says

Look ! is there a reference?

The history object

- The history object keeps track of every page the user visits.
- Methods of this object include

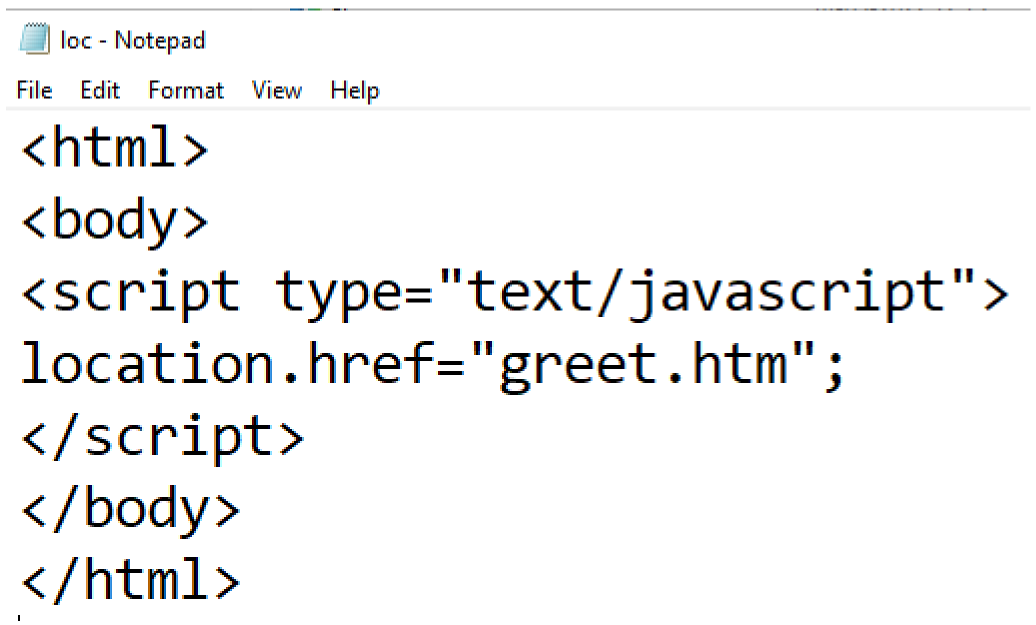
1. forward
2. back
3. go

```
history.back(2) //go back by 2 pages
```

The location object

- The location object contains the URL of the page.
- You can use its href property to go to a new page.

location.href=“greet.htm”



```
<html>
<body>
<script type="text/javascript">
location.href="greet.htm";
</script>
</body>
</html>
```

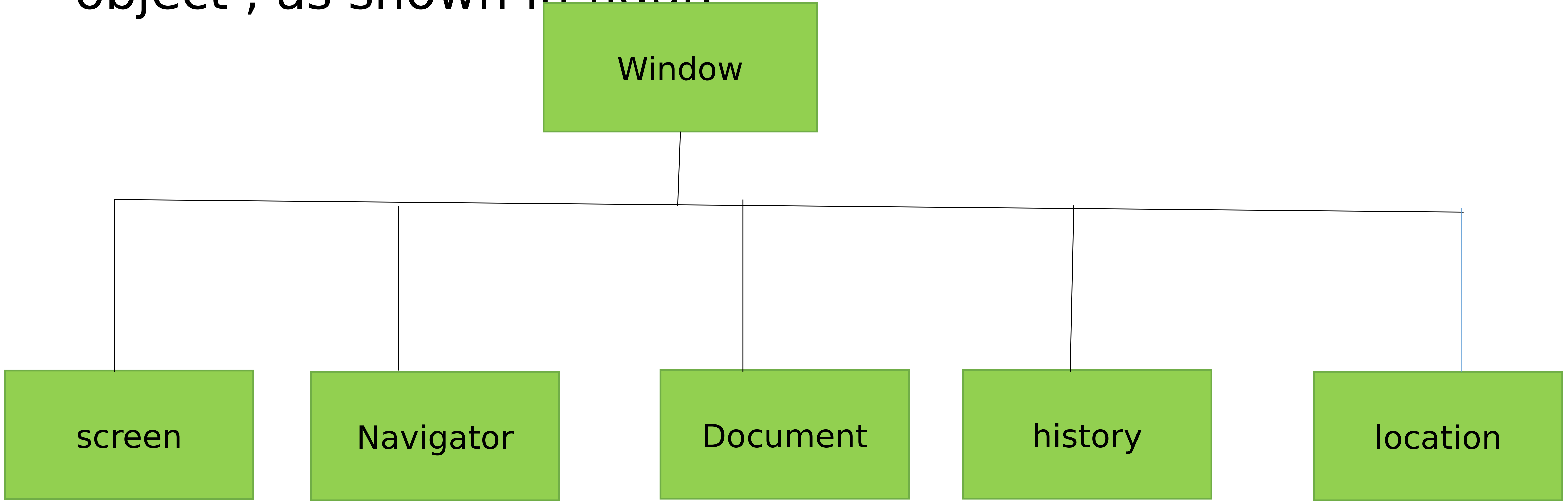
The navigator object

- The navigator object contains information about the browser name and version.
- Properties include

1. appName
2. appVersion
3. userAgent

The screen object

- The screen object provides information about the display characteristics, such as screen height and width in pixels. The document object is included in the window object , as shown in figure



The window object

- At the top of the object hierarchy is the window object. This object is very important as it plays the role of master container for all the contents you view in the Web browser. In addition to the content part of the window, where the documents will go, a window’s sphere of influence includes the dimension of the window and all the stuff that surrounds the content area. The area where the scrollbars, tool bars, status bar, and menu bar are present is known as the window’s chrome.
- The window object is a convenient place for the DOM to attach the methods that will display model dialog boxes and adjust the text that displays at the bottom of the browser window. A window object method enables you to create a separate window that appears on screen.

- You can reference the properties and methods of the window object in many ways - depending more on the whim and style as compared to the syntactical requirements. The most common way to compose such references include the window object in the reference as shown here:

window.propertyName

window.methodName(parameters)

A window object also has a synonym when the script doing the referencing points to the window that houses the document. The synonym is self. In this case, you replace the window object by its synonym self and the syntax changes to what is shown here.

self.propertyName

self.methodName([parameters])

- A user doesn’t create the main browser window. The user does so by opening the browser or by opening a URL or file from the browser’s menus. The script can generate any number of subwindows when the main window is already open. The method that generates the new window is window.open().
- This method can contain up to three parameters, which will define the characteristics, such as the URL of the document to load, is name for the target attribute reference purpose in HTML tags, and the physical appearance.The following code shows the creation of a new sub window:

```
var subwindow=window.open(“greet.html”,”def”,”height=225,width=400”);
```

You can close this window by using the method close() as follows:

- Some of the important methods available with the window object are alert(), confirm() and prompt().
- The alert() method generates a dialog box that displays the text that is passes as a parameter. A single OK button enables the user to dismiss the alert.
- The next method is confirm().This dialog box represents two buttons – ‘Cancel’ and ‘OK’ and is called a confirm dialog box. This is one of the methods that return a value :true if the user clicks on ‘OK’ and false when the user clicks on the ‘Cancel’ button.
- The prompt() dialog box displays the message that you set and provides a text field for entering the user’s response. The two buttons ‘OK’ and ‘Cancel’ enable the user to dismiss the dialog box. The user either cancels the entire operation by clicking on “Cancel” button or accepts the input typed in the dialog box by clicking on the “OK” button.

Some Properties available with the window object

1. Closed-it returns a Boolean value that indicates whether or not a window is closed
2. defaultStatus-It sets or returns the default text in the status bar of the window.
3. Document-it is used to access document object
4. History- it is used to access history object
5. Location-it is used to access Location object
6. Name – it sets or returns the name of the window
7. Opener – it returns a reference to the window that created the window
8. Outerheight-it sets or returns the outer height of a window
9. Outerwidth-it sets or returns the outer width of a window
10. Self – it returns a reference to the current window.
11. Status-it sets the text in the status bar of a window

Some methods of the window object

- alert() – the alert() method is used to display an alert box with a specified message and an OK button. The syntax of the alert() method is alert(message)
- Close() – the close() method is used to close the current window. The syntax of the close() method is window.close()
- Confirm()-this method displays a dialog box with a message and an ‘OK’ and a ‘Cancel’ button. Syntax is confirm(message)
- Open()-this method opens a new browser window. The syntax of the open() method is as follows

window.open(URL,name,specs,replace)

All the four parameters are optional

- Print()- the Print() method is used to print the contents of the current window. The syntax of the Print() method is as follows

window.print()

Prompt() – this method displays a dialog box that prompts the user for input. Syntax is prompt(text,defaultText)

Both the parameters passed are optional

The location object – The location object represents the URL loaded into the window . A URL consists of many components that define the address and method of the data transfer for a file. A URL can be divided into two parts – protocol and hostname. You can access all these items as the properties of the location object.

In most cases, your script would be interested in only 1 property, the href property , which defines the complete URL.

Setting the location.href property is the most common way for your Script which navigates to other pages.

location.href=http://www.pccegoa.edu.in;

Bachelor Of Engineering
In
Information Technology

Semester Six, Third Year(Even semester)
17th May 2022
21st Offline Lecture

Padre Conceicao College of Engineering

Verna Goa 403722 India

- You can navigate to other pages in your own web site by specifying the relative URL rather than the complete URL.For the pages outside the domain, you need to specify the complete URL .If the page is loaded in another window or frame, the window reference should be part of the statement. For example, if your script opens a new window and assigns its reference to a variable named new Window, the statement that loads a page into the sub window is as follows:

```
newWindow.location.href=http://www.pccegoa.edu.in
```

Important properties of location object

- Host-this property sets or returns the hostname and port number of the current URL
- Hostname-this property sets or returns the hostname of the current URL
- Href- it sets or returns the entire URL
- Pathname-it sets or returns the path of the current URL
- Protocol-it sets or returns the protocol of the current URL

Important methods of location object

- assign() method loads a new document.Syntax is location.assign(URL)
- The parameter URL represents the URL for the document to be loaded.
- reload() method is used to reload the current document. Syntax is location.reload()
- replace() method replaces the current document with the new one. Syntax is location.replace(newURL)

The Navigator object

- The Navigator object is implemented in every scriptable browser. All browsers also implement some properties which reveal the same kind of properties that the browser sends to the servers, with each page request. Thus ,the navigator.userAgent property returns a string with a number of details about the browser and the operating system.

The document object

- The document object holds the real contents of the page. Properties and methods of the document generally affect the look and content of the document that occupies the window. All W3C DOM-compatible browsers allow script access to the text contents of a page when the document is loaded. The document.write() method lets a script create content dynamically as the page loads on the browser. Many document object properties are the arrays of other objects in the document. These properties provide additional ways to reference these objects. You can access the document object’s properties and methods as shown in the syntax below

Window.document.propertyName

Window.document.methodName(parameters)

4 important collections with the document object

- Anchors[ ] – the anchors collection returns a reference to all Anchor objects in the document. Syntax is document.anchors[ ]
- Forms[ ] –The forms collection returns a reference to all Form objects in the document. Syntax is document.forms[ ]
- Images[ ]-The images collection returns a reference to all Image objects in the document. Syntax is document.images[ ]
- Links[ ]-The links collection returns a reference to all Area and Link objects in the document. Syntax is document.links[ ]

Properties of the document object

- Cookie-the cookie property sets or returns all cookies associated with the current document. Syntax is document.cookie
- Domain-the Domain property returns the domain name for the current document. Syntax is document.domain
- lastModified-the lastModified property returns the date and time the document was last modified. Syntax is document.lastModified
- Referrer-the referrer property returns the URL of the document that loaded the current document. Syntax is document.referrer
- Title-the title property returns the title of the current document. Syntax is document.title
- URL- the URL property returns the URL of the current document. Syntax is document.URL

Important methods of the document object

- Close()-the close() method closes an output stream opened with the document.open method and displays the collected data. Syntax is document.close()
- Open()- the open() method opens a stream to collect the output from any document.write or document.writeln methods. Syntax is document.open(mimetype,replace)
- Write()-the write() method writes HTML expression or Javascript code to a document.Syntax is document.write(exp1,exp2,exp3,……)
- writeLn()-the writeLn() method is identical to the write() method,with the addition of writing a new line character after each expression.Syntax is document.writeln(exp1,exp2,exp3,……)

The history object

- As the user surfs the web, the browser maintains a list of URLs for the most recent stops. This list is represented in the scriptable object model by the history object. A script cannot secretively extract actual URLs maintained in that list unless you use signed scripts and the user grants permission. Under unsigned conditions, a script can methodically navigate to each URL in the history, in which case the user sees the browser navigating on its own, as though possessed by a script.
- You should use the history object and its methods with extreme care. The design should be smart enough to watch what the user is doing with your pages. Otherwise, you run the risk of confusing your user by navigating to unexpected places. Your script can also get into trouble because it cannot detect where the current document is in the Back-forward sequence in history.
- The history object is part of the window object and is accessed through the window.history property.

Important methods of the history object

Back()-the back() method loads the previous URL in the history list. Syntax is history.back()

Forward() –the Forward() method loads the next URL in the history list.Syntax is history.forward()

Go()-the go() method loads a specific page in the history list.Syntax is history.go(number|URL)

The screen object

- The screen object is a read-only object that lets the script learn about the physical environment in which the browser is running. For example, this object reveals the number of pixels high and wide available in the monitor.

Important Properties of the screen object

- availHeight – The availHeight property returns the height of the client’s display screen excluding the windows Taskbar. Syntax is screen.availHeight
- availWidth-The availWidth property returns the width of the client’s display screen excluding the Windows Taskbar. Syntax is screen.availWidth
- bufferDepth-The bufferDepth property sets or returns the bit depth of the color palette in the off-screen bitmap buffer. Syntax is screen.bufferDepth=number
- colorDepth-The colorDepth property returns the bit depth of the color palette on the destination device or buffer. Syntax is screen.colorDepth
- Height-the height property returns the height of the client’s display screen. Syntax is screen.height
- Width-the Width property returns the Width of the client’s display screen. Syntax is screen.width

What is Document Object Model

- JavaScript uses the DOM for accessing objects associated with any HTML page and changes their properties. DOM is a platform and language independent standard object model for representing HTML or XML in tree formats. Since the DOM supports navigation in any direction(eg parent and previous sibling) and allows the arbitrary modifications, an implementation must at least buffer the document that is read so far .Hence, the DOM is likely to be best suited for applications where the document must be accessed repeatedly or out of sequence order. If the application is strictly sequential and one-pass, the SAX(Simple API for XML)model is likely to be faster and use less memory.

- The DOM is a tree-based representation of a document. The DOM was created by the WWW Consortium for XML and HTML/XHTML.The DOM provides a set of objects for representing the structure of the document,as well as for accessing those objects.The DOM is divided into the following 3 parts.

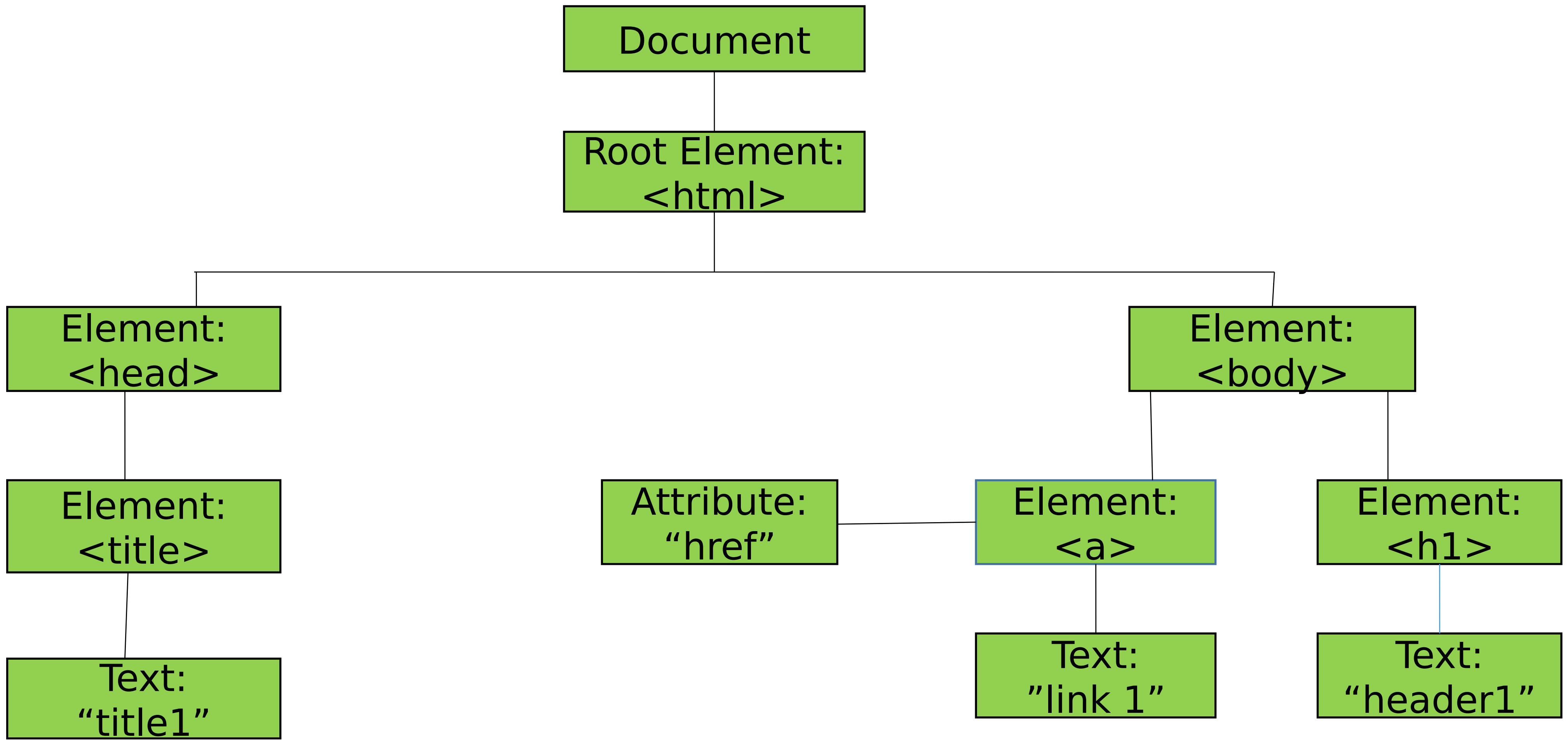
1. The Core DOM,which includes objects that XML and HTML have in common.
2. The XML DOM includes the XML objects
3. The HTML DOM includes the HTML objects

- All the elements in a page are related to the topmost object. You can access any object in an HTML page and change its properties by using JavaScript in this model.

- For example , you can
- Change its position
- Change its source file
- Change its style properties
- Change its content
- Add new content
- A document can be viewed as a node tree. In the node tree view, a document is a collection of nodes. The nodes symbolize the branches and leaves on the document tree. There are many types of nodes, but the three main types –

1. Element nodes
2. Text nodes
3. Attribute nodes

Figure 29.2 A node tree for an HTML Document



Key components of the node structure

- Element nodes -Elements are the basic building blocks of documents and give them their structure. Elements can contain other elements e.g.’<html>’,’<head>’,’<body>’,etc. are the Elements nodes
- Text node -In HTML/XHTML Text nodes are always contained in Element nodes,e.g’title1’,’link1’, and ‘header1’ are all Text nodes
- Attribute nodes - Attributes provide more information about elements. Attribute nodes are always contained in Element nodes. For example, the ‘href’ is an Attribute nodes.
- Every node has some properties that contain some information about the node. The properties are as follows:

1. nodeName
2. nodeValue
3. nodeType

Bachelor Of Engineering
In
Information Technology

Semester Six, Third Year(Even semester)
23th May 2022
22nd Offline Lecture

Padre Conceicao College of Engineering

Verna Goa 403722 India

1.nodeName – The nodeName property contains the name of a node

- The nodeName of an Element node is the tag name
- The nodeName of an Attribute node is the Attribute name
- The name of a Text node is always #text
- The name of the Document node is always #document

2.nodeValue – on text nodes, the nodeValue property contains the text. On Attribute nodes, the nodeValue property contains the Attribute value. The nodeValue property is not available on Document and Element nodes.

3.nodeType – The nodeType property returns the type of node. The most important node types are described in the Table

Nodes of HTML document

Element type	Node type
Element	1
Attribute	2
Text	3
Comment	8
Document	9

Methods Available in DOM for accessing Objects

- There are 2 methods available in DOM for accessing various elements of the document.
- These 2 methods are

1. getElementById
2. getElementsByTagName

getElementById method

- The getElementById method returns the element with the specified ID.The syntax is
  document.getElementById(“someID”);
- If an element has an id, the simplest way to access it is by using the getElementById() method

```
<p id=“someID”>Greetings, Hello!</p>

….

Var a=document. getElementById(‘someID’);
```

a is a shortcut for accessing the unique element with the id value of someID.To change a property of this element, for example the fontWeight,use the following code snippet.

```
x.style.fontweight = “bold”;     //changes the font weight to bold
```

- You can also use getElementById to access elements by using node properties. For example, if the head and body elements in Figure 29.2 include id values, such as the following,you can use these id values to access all the other elements in the page.

```
<head id="e1">

....

<body id="e2">
```

To access the parent node of the head and body elements, you can use either of the following:

```
document. getElementById('e1').parentNode;

document. getElementById('e2').parentNode;
```

Both of these access the HTML element.

- To access all the children of the body element, use the following

```
document. getElementById(‘e2’).childNodes;
```

The child nodes are contained in an array. You can access individual child nodes by using the array index value.

```
document. getElementById(‘e2’).childNodes[0];
```

This accesses the first child node of the body element <a>, since array indexes start with 0.If you want to access a sibling node, use the following.

```
document. getElementById(‘e2’).previousSibling;  //accesses head element
document. getElementById(‘e1’).nextSibling;         //accesses body element
```

getElementsByTagName() method

- The getElementsByTagName() method returns all elements (as a nodeList) with the specifdied tag name that are descendants of the element when you are using this method.
- The getElementsByTagName() can be used on any HTML element, and also on the document.
- The syntax of getElementsByTagName() method is as

```
document. getElementsByTagName(“tagname”) ;
```

Or

```
document. getElementById(‘someID’).
getElementsByTagName(“tagname”) ;
```

- To use getElementsByTagName method , you use a tagname rather than an id
- For example

```
Var x=document. getElementsByTagName(‘h1’) ;
```

Here, x contains a reference to h1 element in the document. To change the text color of h1,use the following

```
x.style.color=“green”;
```

DOM Levels

- Level 0 – the application supports an intermediate DOM, which existed before the creation of DOM Level 1.Examples include the DHTML Object Model or the Netscape intermediate DOM . Level 0 is not a formal specification published by the W3C , but rather a shorthand that refers to what existed before the standardization process.
- Level 1- includes the navigation of DOM document and content manipulation.HTML-specific elements are included as well
- Level 2-XML namespace support, filtered views and events

DOM Level 3

1. DOM Level 3 core
2. DOM Level 3 Load and Save
3. DOM Level 3 XPath
4. DOM Level 3 Views and Formatting
5. DOM Level 3 Requirements
6. DOM Level 3 Validation, which further enhances the DOM

DOM Level 1

- The DOM Level 1 is an API (application programming interface) that allows programs and scripts to dynamically access and update the content, structure and style of HTML and XML 1.0 documents. The Document Object Model provides a standard set of objects for representing HTML and XML documents, a standard model of how these objects can be combined, and a standard interface for accessing and manipulating them. Vendors can support DOM as an interface to their proprietary data structures and APIs and content authors can write to the standard DOM interfaces rather than products –specific APIs, thus increasing interoperability on the web.

- The goal of the DOM specification is to define a programmatic interface for XML and HTML . The DOM Level 1 specification is separated into two parts-Core and HTML. The Core DOM Level 1 section provides a low-level set of fundamental interfaces that can represent any structured document, as well as define extended interfaces for representing an XML document. These extended XML interfaces need not be implemented by a DOM implementation that only provides access to HTML documents; all of the fundamental interfaces in the Core section must be implemented. A compliant DOM implementation that implements the extended XML interfaces is required to also implement the fundamental Core interfaces, but not the HTML interfaces. The HTML Level 1 section provides additional, higher –level interfaces that are used with the fundamental interfaces defined in the Core Level 1 section to provide a more convenient view of an HTML document. A compliant implementation of the HTML DOM implements all of the fundamental Core interfaces as well as the HTML interfaces.

DOM Level 2

- The DOM Level 2 extends Level 1 with support for XML 1.0 with namespaces and adds support for Cascading style sheets (CSS),events (user interface events and tree manipulation events), and enhances tree manipulations(tree ranges and traversal mechanisms).The DOM Level 2 defines the following specifications, which have reached in their final form:
- DOM Level 2 Core
- DOM Level 2 Views
- DOM Level 2 Events
- DOM Level 2 Style
- DOM Level 2 Traversal and Range
- DOM Level 2 HTML

DOM Level 2 Core

- This specification defines the Document Object Model Level 2 Core, a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content and structure of documents. The DOM Level 2 Core extends the DOM Level 1 Core.
- The DOM Level 2 Core is made of a set of core interfaces to create and manipulate the structure and contents of a document. The Core also contains specialized interfaces dedicated to XML.

DOM Level 2 views

- This specification defines the DOM Level 2 views. This is a platform and language –neutral interface that allows programs and scripts to dynamically access and update the content of a HTML and XML document. The DOM Level 2 Views extends the DOM Level 2 Core.
- A document may have one or more views associated with it.e.g.a computed view on a document after applying a CSS stylesheet , or multiple presentations (e.g. HTML Frame) of the same document in a client. That is, a view is some alternate representation of a source document.
- A view may be static, reflecting the state of the document when the view was created, or dynamic, reflecting changes in the target document as they occur, subsequent to the view being created. This level of the DOM specification makes no statement about these behaviors.

- This section defines an AbstractView interface, which provides a base interface from which all such views shall derive. It defines an attribute, which references the target document of the AbstractView.
- The only semantics of the AbstractView defined here create an association between a view and its target document. There is no sub interfaces of AbstractView defined in the DOM Level 2.
- AbstractView is defined and used in this Level in two places:

1.A document may implement a DocumentView that has a default view attribute associated with it.This default view is typically dependent on the implementation, eg the browser frame rendering the document. The default view can be used in order to identify and/or associate a view with its target document(by testing object equality

2. A UIEvent typically occurs upon a view of a Document, e.g. a mouse click on a browser frame rendering a particular Document instance. A UIEvent has an AbstractView associated with it which identifies both the particular view in which the event occurs, and the target document the UIEvent is related to.

In order to fully support this module, an implementation must also support the Core feature defined in the DOM Level 2 Core specification.

DOM Level 2 Events

- This specification defines the DOM Level 2 Events, a platform- and language-neutral interface that gives to programs and scripts a generic event system. The DOM Level 2 Events extends the DOM Level 2 views. The DOM Level 2 Event Model is designed with two main goals.

1.The first goal is the design of a generic event system, which allows registration of event handlers, describes event flow through a tree structure, and provides basic contextual information for each event. The specification will provide standard modules of events for user interface control and document mutation notifications, including defined contextual information for each of these event modules.

2.The second goal of the event model is to provide a common subset of the current event systems used in DOM Level 0 browsers. This is intended to foster interoperability of existing scripts and content. It is not expected that this goal will be met with full backwards compatibility. The specification attempts to achieve this when possible.

DOM Level 2 style

- This specification defines the DOM Level 2 style sheets and CSS , a platform and language neutral Interface that allows programs and scripts to dynamically access and update the content of style sheet documents. The DOM Level 2 style extends the DOM Level 2 Core and on the DOM Level 2 Views. The DOM Level 2 style sheet interfaces are base interfaces used to represent any type of style sheet. The expectation is that DOM modules that represent a specific style sheet language may contain interfaces that derive from these interfaces.
- The DOM Level 2 CSS interfaces are designed with the goal of exposing CSS constructs to object model consumers.CSS is a declarative syntax for defining presentation rules, properties and ancillary constructs used to format and render Web documents. This documents specifies a mechanism to programmatically access and modifies the rich style and presentation control provided by CSS.This augments CSS by providing a mechanism to dynamically control the illusion and exclusion of individual style sheets, as well as manipulate CSS rules and properties.
- The CSS interfaces are organized in a logical , rather than physical structure. A collection of all style sheets referenced by or embedded in the document is accessible on the document interface.

DOM Level 2 Traversal and Range

- This specification defines the DOM Level 2 Traversal and Range, platform and language neutral interfaces that allow programs and scripts to dynamically traverse and identify a range of content in a document. The DOM Level 2 Traversal and Range specification extends the DOM Level 2 Core specification.
- The DOM Level 2 Traversal and Range specification is composed of two modules. The two modules contain specialized interfaces dedicated to traversing the document structure and identifying and manipulating a range in a document.

DOM Level 2 HTML

- This specification defines the DOM Level 2 HTML, a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content and structure of (HTML and XHTML) documents. The DOM Level 2 HTML extends the DOM Level 2 core and is not backward compatible with DOM Level 1 HTML.
- This section extends the DOM Level 2 core API to describe the objects and the methods specific to HTML documents and XHTML documents. In general, the functionality needs to manipulate hierarchical document structures, elements, and attributes will be found in the core section; functionality that depends on the specific elements defined in HTML will be found in this section.

- The goals of the HTML specific DOM API are as follows:
- To specialize and add functionality that relates specially to HTML documents and elements.
- To address issues of backward compatibility with the DOM Level 0
- To provide convenience mechanisms, wherever appropriate, for common and frequent operations on HTML documents.
- The key differences between the Core DOM and the HTML application of DOM is that the HTML DOM exposes a number of convenience methods and properties that are consistent with the existing models and are more appropriate to script writers.
- In many cases, these enhancements are not applicable to general DOM because they rely on the presence of a predefined DTD.The transitional or frameset DTD for HTML 4.01, or the XHTML 1.0 DTDs are assumed.

- Interoperability between implementations is only guaranteed for elements and attributes that are specified in the HTML 4.01 and XHTML 1.0 DTDs.
- This document includes the following specialization for HTML.
- An HTMLDocument interface derived from the core Document interface, HTMLDocument specifies the operations and queries that can be made on a HTML document.
- An HTMLElement interface derived from the core Element interface, HTMLElement specifies the operations and queries that can be made on any HTML element. Methods on HTMLElement include those that allow for the retrieval and modification of attributes that apply to all HTML elements.
- Specializations for all HTML elements, which have attributes that extend beyond those specified in the HTMLElement interface. For all such attributes, the derived interface for the element contains explicit methods for setting and getting the values.
- The DOM Level 2 includes mechanisms to access and modify style specified through CSS and define an event model that can be used with HTML documents.

DOM Level 3

- The DOM Level 3 will extend Level 2 by finishing support for XML 1.0 with namespaces and will extend the user interface events (keyboard).
- It will also add abstract schemas support ,the ability to load and save a document or an abstract schema, explore further mixzed markup vocabularies and the implications on the DOM API , and support path.

DOM Level 3 Core

- This specification defines the DOM Core Level 3, a platform-and-language –neutral interface that allows programs and scripts to dynamically access and update the content, structure and style of documents. The DOM Core Level 3 extends the DOM Core Level 2.This version enhances DOM Level 2 Core by completing the mapping between DOM and the XML information Set, including the support for XML Base, adding the ability to attach user information to DOM nodes or to bootstrap a DOM implementation , providing mechanisms to resolve namespace prefixes or to manipulate ID attributes, giving to type information, etc.

- This specification defines a set of objects and interfaces for accessing and manipulating document objects. The functionality specified is sufficient to allow software developers and Web script authors to access and manipulate parsed HTML and XML content inside conforming products.The DOM Core API also allows creation and population of a Document object using only DOM API calls.

DOM Level 3 Load and Save

- This specification defines the DOM Load and Save Level 3 ,a platform-and –language –neutral interface that allows programs and scripts to dynamically load the content of an XML document into a DOM document and serialize a DOM document into an XML document; DOM documents being defined in DOM Level 2 Core or newer, and XML documents being defined in XML 1.0 or newer. It also allows filtering of content at load time and serialization time.

DOM Level 3 Validation

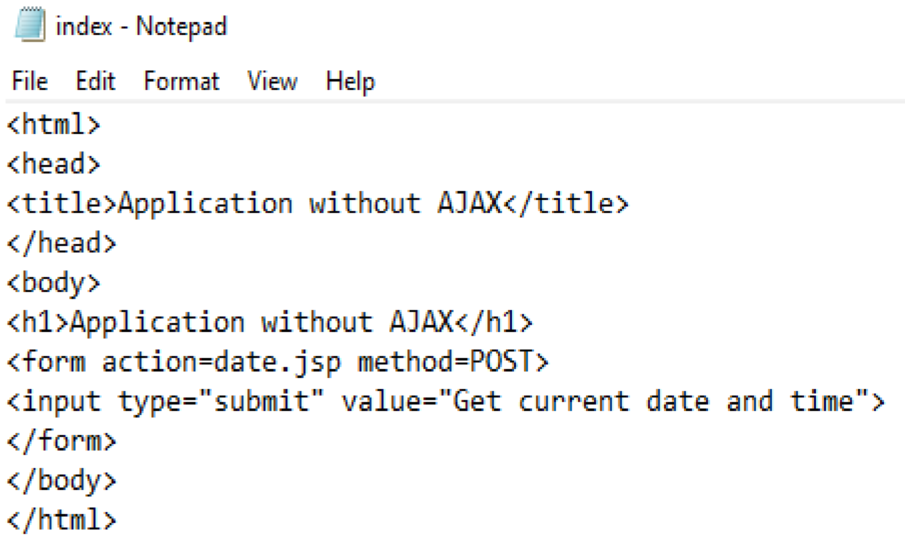
- This specification defines the DOM Validation Level 3 , a platform and language neutral interface. This module provides guidance to programs and scripts to dynamically update the content and the structure of documents while ensuring that the document remains valid, or to ensure that the document becomes valid.
- This module provides Application Programming Interfaces(APIs) to guide construction and editing of XML documents. Examples of such guided editing are queries that combine questions like “what does the schema allow me to insert/delete here?” and “if I insert/delete here, will the document still be valid?”

- To aid users in the editing and creation of XML documents, other queries may expose different levels of details,e.g. all the possible children, lists of defined symbols of a given kind. Some of these queries would prompt check and warn users if they are about to conflict with or overwrite such data.
- Finally, users would like to validate an edited or newly constructed document before serializing it or passing it to other users. They may edit, come up with an invalid document, and then edit again to result in a valid document. During this process, these APIs can allow the user to check the validity of the document or sub tree on demand. If necessary, these APIs can also require that the document or sub tree remain valid during this editing process via the DocumentEditVal.continuousValidityChecking flag.
- A DOM application can use the hasFeature(feature,version) method of the DOMImplementation interface to determine with parameter values Validation and 3.0, respectively, whether or not these interfaces are supported by the Implementation. This Implementation is dependent on [DOM Level 2 core] and the [DOM Level 3 core] DOMConfiguration interface.

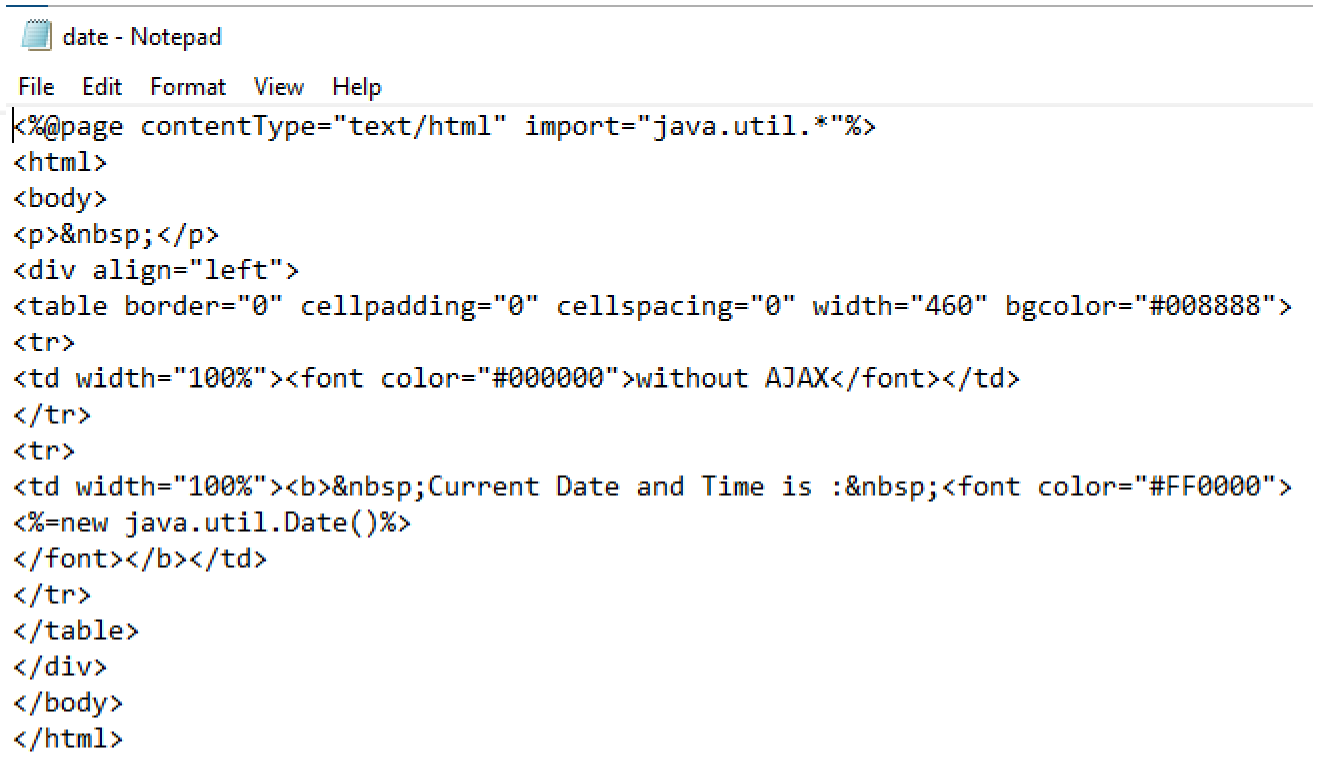
Creating a Javascript Application without AJAX

- In this application ,we will send a POST request to the server without using AJAX technique. This application demonstrates that from sending the request to the server, the server performing the request processing and returning the HTML page to the client, the client has to keep waiting. The application sends a request for the system’s current date and time to the server. The application starts with a HTML page, index.html.The HTML page displays a button labelled “Get Current Date and Time” and when the client presses this button it sends a POST request to the server for accessing the current date and time from the file date.jsp. When you open the HTML file, index.html , it is displayed as shown

Index.html and date.jsp



```
<html>
<head>
<title>Application without AJAX</title>
</head>
<body>
<h1>Application without AJAX</h1>
<form action=date.jsp method=POST>
<input type="submit" value="Get current date and time">
</form>
</body>
</html>
```



```
<%@page contentType="text/html" import="java.util.*"%>
<html>
<body>
<p>&nbsp;</p>
<div align="left">
<table border="0" cellpadding="0" cellspacing="0" width="460" bgcolor="#008888">
<tr>
<td width="100%"><font color="#000000">without AJAX</font></td>
</tr>
<tr>
<td width="100%"><b>&nbsp;Current Date and Time is :&nbsp;<font color="#FF0000">
<%=new java.util.Date()%>
</font></b></td>
</tr>
</table>
</div>
</body>
</html>
```

Figure 29.3 Accessing current Date and Time without using AJAX

- When you click on the button “Get Current Date and Time", the HTML form sends a POST request to the server for accessing current date and time from the file date.jsp.The JSP page displays the current date and time when you click the” Get Current Date and Time” button of the index.html page, as shown in figure 29.4

Javascript and AJAX

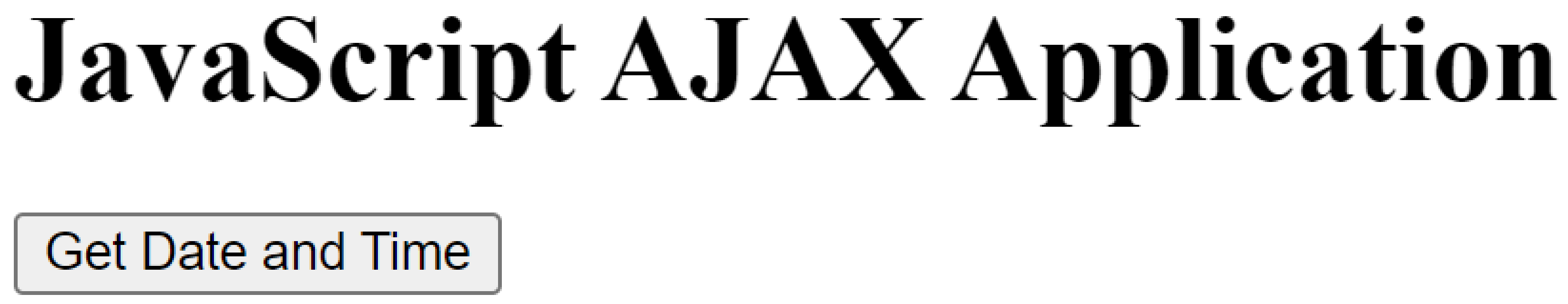
- With AJAX your JavaScript can communicate directly with the server, using the JavaScript object. With this object, your JavaScript can trade data with a web server, without reloading the page. AJAX uses asynchronous data transfer (HTTP requests) between the browser and the Web server, allowing web pages to request small bits of information from the server, instead of whole pages. The AJAX technique makes Internet application smaller, faster, and more user-friendly. AJAX is a browser technology independent of Web server software.
- JavaScript plays an important role in the AJAX technique. Using JavaScript technology, an HTML page can asynchronously make calls to the server from which it loads and fetches contents that may be formatted as XML documents , HTML content , plain text, or JavaScript Object Notation (JSON).The JavaScript technology may then use the content to update or modify the DOM of the HTML page. The term Asynchronous JavaScript technology and XML (AJAX) has emerged recently to describe this interaction model.

- AJAX is not a new technique as JavaScript and XML are being already in use for a long time. What has changed is the inclusion of support for the XMLHttpRequest object in the JavaScript runtimes of he mainstream browsers. Although this object is not specified in the formal JavaScript technology specification, all of today’s mainstream browsers support it.
- What makes AJAX-based clients unique is that the client contains page-specific control logic embedded as JavaScript technology .The page interacts with the JavaScript technology based on events, such as the loading of a document, a mouse click, focus changes, or even a timer. AJAX interactions allow for a clear separation of presentation logic from the data. An HTML page can pull in bite-size pieces to be displayed.
- With an HTTP request, a web page can make a request to, and get a response from a Web server-without reloading the page. The user will stay on the same page, and he will not notice that scripts might request pages, or send data to a server in the background. This is the most important functionality that JavaScript provides to AJAX.

Creating a JavaScript Application with AJAX

- In this application we will implement how JavaScript uses the XMLHttpRequest object for sending the request to the server and receiving response from the server. The application starts with a HTML page,datetime.html,that displays a button labelled “Get Current Date and Time”

- When you open the datetime.htm page,it is displayed as shown in Figure 29.5



- When you click on the Get Current Date and Time button, the XMLHttpRequest sends a POST request to the server for accessing current date and time from the file date.jsp.
- After clicking the Get Current Date and Time button on the datetime.htm page, the HTML page displays the current date and time, as shown in figure 29.6.
- In this application we first create an XMLHttpRequest object. Next we define a JavaScript function getDateTime() which uses the XMLHttpRequest for sending asynchronous request to the server. In this function, we call the XMLHttpRequest object’s open() function and passed the date.jsp page as a parameter to this function. The XMLHttpRequest object sends a asynchronous POST request for date.jsp file to the server. After opening the XMLHttpRequest object, the XMLHttpRequest object has the property named onreadystatechange,which allows handling the asynchronous loading operations. If this property is assigned to the name of the JavaScript function then, this function will be called each time the XMLHttpRequest object’s state changes.

- Finally, when the XMLHttpRequest object is in its ready state and the status is equal to 200, then the data is fetched. The status 200 refers to the ‘OK’ state of the XMLHttpRequest object. So, to make sure that the data is completely downloaded ,we check the value of the status property with 200.Finally when the data is downloaded, the data is retrieved in either the standard HTML or the XML format. If responseText property is used then the data is retrieved in the standard HTML format. If your data is formatted as XML,then responseXML property is used. We are using responseText since our data is in standard HTML format that is returned by the date.jsp page.
- After retrieving the data, in order to display the data on the Web page, you can assign that text to the <div> element, whose ID is targetDiv in the web page and whose name was passed to the getDateTime() function.

Assignment 3

- Q1) Demonstrate how to reverse an array in JavaScript with a suitable program. Show the output(5 marks)

- *Assignment Announced to students : AA :01st June 2022*
- *Assignment to be Submitted by students : AS: 06th June 2022*