KARPAGAM ACADEMY OF HIGHER EDUCATION
(Deemed University Established Under Section 3 of UGC Act, 1956)

Coimbatore - 641 021, India

FACULTY OF ARTS, SCIENCE AND HUMANITIES (FASH)

Department of CS,CA & IT

I M.Sc CS **I SEMESTER** **BATCH : 2017- 2019**

17CSP101 **WEB TECHNOLOGY** **4H – 4C**

Instruction Hours / week: L: 4 T: 0 P: 0 **Marks:** Int : **40** Ext : **60** Total: **100**

SCOPE

This course relates to the interface between web servers and their clients. The course provides information includes markup languages programming interfaces and languages and standards for document identification and display. The use of Web technology makes to enhance active student learning and improves their creativity in making web pages.

OBJECTIVES:

- To Create a new webpage
- To understand the fundamental features of web applications.
- To understand the objects and components needed for a web designing.

LEARNING OUTCOMES:

The student will be able to:

• Analyze a web page and identify its elements and attributes.
• Create web pages using java script ,ASP.Net Cascading Style Sheets.
• Build dynamic web pages using JavaScript (Client side programming).
• Create XML documents and Schemas.
• Build interactive web applications using AJAX.

UNIT-I

JavaScript: Introduction to JavaScript – Programming fundamentals – Functions and objects – Navigator object model. Form and form elements – Scripting frames and multiple windows – Event object.

UNIT-II

ASP.NET: **ASP & ASP.NET**: An Overview – Programming ASP.NET with VB.NET: ASP Data types – operators- Request Object- Response Object – Server object - Web forms and ASP.NET: Web forms

.

UNIT-III

ASP.NET: ASP.NET Configuration Scope and State: Configuration – state- application – session object- ASP.NET Objects & Components: Scripting object models- ASP components and controls- ASP.NET and SQL server-Using SQL server using database in ASP.NET applications ActiveX data objects

UNIT-IV

Creating Mark up with XML: Introduction – Parsers and well formed XML Documents – Parsing an XML Document - Characters – Mark up – CDATA Sections – XML Namespaces. Document Type Definition – Parsers Well formed and valid XML documents – Element type declarations – Attribute declarations- Attributes Types. Schemas – Schemas VS DTD's – W3C XML Schema

UNIT-V

Document Object Model: DOM implementations – DOM with JavaScript – Components- Creating nodes – Traversing the DOM. Simple API for XML: DOM vs SAX – SAX based Parsers. XLink XPointer XInclude and XBase

SUGGESTED READINGS

TEXT BOOKS

1. Professional JavaScript® for Web Developers, 2nd Edition Nicholas C. Copyright © 2009 by Wiley Publishing, Inc., Indianapolis, Indiana
2. David Flanagan.(2011). Javascript: The Definitive Guide (6th ed.). O'Reilly Media.
3. Danny Goodman. (2000). Javascript Bible(3rd ed.). IDG Books India Pvt Ltd.
4. (Page Nos.: 9-16 24-33 68-89 116-130 151-157 174-198 248-252 323-329 348-356)
5. Dave Mercer. 2010. ASP.NET – Beginner's Guide(2nd ed.). New Delhi: MCGraw Hill
6. Rohit Khurana. (2002). Javascript Professional ed.). (2nd ed.). NewDelhi: A.P.H. Publishing Company**.** (Page Nos.: 1-93 98-170)

7. Deitel & Deitel. (2001). XML How to Program(1st ed.). NewDelhi: Pearson Education.

(Page Nos: 110-127 134-159 165-186 192-227 232-258 372-391 297-314 319-347 603-608)

REFERENCES

1. David Flanagan. (2006). Javascript: The Definitive Guide. O'Reilly Media.
2. Nicholas C. Zakas. Inc Ebrary & Ebrary. 2005. Professional JavaScript for Web Developers New Delhi: John Wiley & Sons Inc.
3. Russell Jones A. (2000). Mastering ActiveServerPages 3(1st ed.). New Delhi: BPB Publishing.
4. Thau. (2007). The Book of JavaScript: A Practical Guide to Interactive WebPages.
5. Ann Novarro Chuck White & Linda Burman. (2000). Mastering XML(1st ed.). New Delhi: BPB Publications.
6. Charles Ashbacher. (2000). XML in 24 hours(1st ed.). New Delhi: Techmedia Publication.
7. Manish Jain. (2001). XML Complete (1st ed.). New Delhi: BPB Publications.
8. Steve Holzner. (2000). Inside XML (1st ed.). New Delhi: TechMedia.
9. Matthew MacDonald. (2013). ASP.NET The Complete Reference. New Delhi: McGraw Hill Education.

WEB SITES

1. www.w3schools.com/
2. www.javascriptkit.com
3. www.learn-javascript-tutorial.com
4. www.webteacher.com/javascript
5. www.asptutorial.info
6. www.aspfree.com
7. www.aspnettutorials.com

KARPAGAM ACADEMY OF HIGHER EDUCATION

COIMBATORE-641021

DEPARTMENT OF COMPUTER SCIENCE

BATCH : 2017 - 2019

LECTURE PLAN

Faculty : A.JEEVARATHINAM **Subject : Web Technology**

Subject Code : 17CSP101 **Class : I M.Sc Computer Science**

UNIT I

SI.NO	Lecture Duration (Hours)	Topics to be Covered	Support Materials
1	1	Introduction to JavaScript Programming fundamentals	T4 2 T4 23
2	1	Functions	T4 74
3	1	Objects - Array,	T4 100
4	1	Objects -String	T4 37
5	1	Objects - Number,	T4 109,146
6	1	Objects - Math, Date	T4 146
7	1	Navigator object model,	T4 201,219
8	1	Window Object	T4 219
9	1	Document Object,	W2
10	1	Form and form elements	W2
11	1	Scripting frames and multiple windows,	W2
12	1	Event Object	W2
13	1	Recapitulation and Discussion of Important Questions	
Total Periods Planned for Unit I			**13**

UNIT II

SI.NO	Lecture Duration (Hours)	Topics to be Covered	Support Materials
1	1	**ASP & ASP.Net :** An Overview	T2 22
2	1	Programming ASP.NET with Vb.NET	T2 93
3	1	ASP Data Types	T2 96
4	1	Operators	T2 99

5	1	Request Object	T2 105
6	1	Response Object	T2 120
7	1	Server Object	T2 134
8	1	Web Forms and ASP.NET: Web forms	T2 155
9	1	Web forms	T2 155
10	1	Recapitulation and Discussion of Important Questions	
Total Periods Planned for Unit II			**10**

UNIT III

SI.NO	Lecture Duration (Hours)	Topics to be Covered	Support Materials
1	1	**ASP.NET Configuration, Scope and State:** Configuration	T2 183
2	1	State	T2 192
3	1	Application	T2 206
4	1	Session Object	T2 209
5	1	**ASP.Net Objects & Components:**Scripting object model	T2 236
6	1	ASP Components & Controls	T2 278
7	1	**ASP.NET and SQL Server:** Using SQL Servers	T2 343
8	1	Using Database in ASP.NET Applications	T2 344
9	1	ActiveX Data Objects	T2 361
10	1	ActiveXData Objects	T2 361
11	1	Recapitulation and Discussion of Important Questions	
Total Periods Planned for Unit III			**11**

UNIT IV

SI.NO	Lecture Duration (Hours)	Topics to be Covered	Support Materials
1	1	Creating Mark up with XML: Introduction, Parsers and well formed XML Documents, Parsing an XML Document	T3 148
2	1	Characters	T3 152
3	1	Mark up	T3 156

4	1	CDATA Sections	T3 156
5	1	XML Namespaces.	T3 161
6	1	Document Type Definition, Parsers	T3 172
7	1	Well formed and valid XML documents	T3 174
8	1	Element type declarations, Attribute declarations	T3 175
9	1	Attributes Types	T3 185
10	1	Schemas, Schemas VS DTD's,	T3 203
11	1	W3C XML Schema	T3 206
12	1	Recapitulation and Discussion of Important Questions	
Total Periods Planned for Unit IV			**12**

UNIT V

Sl.NO	**Lecture Duration (Hours)**	**Topics to be Covered**	**Support Materials**
1	1	Document Object Model: DOM implementations, DOM with JavaScript	T3 230
2	1	Components	T3 237
3	1	Creating nodes	T3 247
4	1	Traversing the DOM	T3 251
5	1	Simple API for XML: DOM vs SAX,	T3 270
6	1	SAX based Parsers.	T3 273
7	1	XLink	T3 411
8	1	XPointer	T3 426
9	1	Xinclude,	T3 427
10	1	Xbase	T3 428
11	1	Recapitulation and Discussion of Important Questions	
12	1	Discussion of Previous ESE Question Papers	
13	1	Discussion of Previous ESE Question Papers	
14	1	Discussion of Previous ESE Question Papers	
Total Hours Planned for Unit V			**14**

Total Hours	**60**

Text Book

T1	Rohit Khurana's , 2002, "Javascript Professional edition", 2nd Edition, A.P.H. Publishing company, NewDelhi.
T2	Dave Mercer, 2010,"ASP.NET - Beginner's Guide", 2nd Edition, NewDelhi:McGraw Hill
T3	Deitel & Deitel. 2001. XML How to Program, 1st Edition, Pearson Education, New Delhi.
T4	Professional JavaScript® for Web Developers, 2nd Edition Nicholas C. Copyright © 2009 by Wiley Publishing, Inc., Indianapolis, Indiana

References

R1	Thau. 2007. The Book of JavaScript: A Practical Guide to Interactive WebPages
R2	Russell Jones A.. 2000. Mastering ActiveServerPages 3, 1st Edition, BPB Publishing, New Delhi.

Web Sites

W1	www.w3schools.com/
W2	www.learn-javascript-tutorial.com
W3	www.aspnettutorials.com

UNIT-I

JavaScript: Introduction to JavaScript – Programming fundamentals – Functions and objects – Navigator object model. Form and form elements – Scripting frames and multiple windows – Event object.

JavaScript

Introduction to JavaScript

JavaScript is *an object-based scripting language* that is lightweight and cross-platform.

JavaScript is not compiled but translated. The JavaScript Translator (embedded in browser) is responsible to translate the JavaScript code.

Where JavaScript is used

JavaScript is used to create interactive websites. It is mainly used for:

- Client-side validation
- Dynamic drop-down menus
- Displaying data and time
- Displaying popup windows and dialog boxes (like alert dialog box, confirm dialog box and prompt dialog box)
- Displaying clocks etc.

JavaScript Example

```
<h2>Welcome to JavaScript</h2>
<script>
document.write("Hello JavaScript by JavaScript");
</script>
```

Prerequisite

Before learning JavaScript, you must have the basic knowledge of HTML.

JavaScript Example

Javascript example is easy to code. JavaScript provides 3 places to put the JavaScript code: within body tag, within head tag and external JavaScript file.

Let's create the first JavaScript example.

```
<script type="text/javascript">
document.write("JavaScript is a simple language for javatpoint learners");
</script>
```

The **script** tag specifies that we are using JavaScript.

The **text/javascript** is the content type that provides information to the browser about the data.

The **document.write()** function is used to display dynamic content through JavaScript. We will learn about document object in detail later.

3 Places to put JavaScript code

1. Between the body tag of html
2. Between the head tag of html
3. In .js file (external javaScript)

1) JavaScript Example : code between the body tag

In the above example, we have displayed the dynamic content using JavaScript. Let's see the simple example of JavaScript that displays alert dialog box.

```
<script type="text/javascript">
 alert("Hello Javatpoint");
</script>
```

2) JavaScript Example : code between the head tag

Let's see the same example of displaying alert dialog box of JavaScript that is contained inside the head tag.

In this example, we are creating a function msg(). To create function in JavaScript, you need to write function with function_name as given below.

To call function, you need to work on event. Here we are using onclick event to call msg() function.

```
<html>
<head>
<script type="text/javascript">
function msg()
{
 alert("Hello Javatpoint");
```

```
}
</script>
</head>
<body>
<p>Welcome to JavaScript</p>
<form>
<input type="button" value="click" onclick="msg()"/>
</form>
</body>
</html>
```

External JavaScript file

We can create external JavaScript file and embed it in many html page.

It provides **code reusability** because single JavaScript file can be used in several html pages.

An external JavaScript file must be saved by .js extension. It is recommended to embed all JavaScript files into a single file. It increases the speed of the webpage.

Let's create an external JavaScript file that prints Hello Javatpoint in a alert dialog box.

message.js

```
function msg(){
 alert("Hello Javatpoint");
}
```

Let's include the JavaScript file into html page. It calls the JavaScript function on button click.

index.html

```
<html>
<head>
<script type="text/javascript" src="message.js"></script>
</head>
<body>
<p>Welcome to JavaScript</p>
<form>
<input type="button" value="click" onclick="msg()"/>
</form>
</body>
```

```
</html>
```

Programming fundamentals

JavaScript Variable

A **JavaScript variable** is simply a name of storage location. There are two types of variables in JavaScript : local variable and global variable.

There are some rules while declaring a JavaScript variable (also known as identifiers).

1. Name must start with a letter (a to z or A to Z), underscore(_), or dollar($) sign.
2. After first letter we can use digits (0 to 9), for example value1.
3. JavaScript variables are case sensitive, for example x and X are different variables.

Correct JavaScript variables

```
var x = 10;
var _value="sonoo";
```

Incorrect JavaScript variables

```
var  123=30;
var *aa=320;
```

Example of JavaScript variable

Let's see a simple example of JavaScript variable.

```
<script>
var x = 10;
var y = 20;
var z=x+y;
document.write(z);
</script>
```

Output of the above example

```
30
```

JavaScript local variable

A JavaScript local variable is declared inside block or function. It is accessible within the function or block only. For example:

```
<script>
function abc(){
var x=10;//local variable
}
</script>
```

Or,

```
<script>
If(10<13){
var y=20;//JavaScript local variable
}
</script>
```

JavaScript global variable

A **JavaScript global variable** is accessible from any function. A variable i.e. declared outside the function or declared with window object is known as global variable. For example:

```
<script>
var data=200;//gloabal variable
function a(){
document.writeln(data);
}
function b(){
document.writeln(data);
}
a();//calling JavaScript function
b();
</script>
```

Javascript Data Types

JavaScript provides different **data types** to hold different types of values. There are two types of data types in JavaScript.

1. Primitive data type
2. Non-primitive (reference) data type

JavaScript is a **dynamic type language**, means you don't need to specify type of the variable because it is dynamically used by JavaScript engine. You need to use **var** here to

specify the data type. It can hold any type of values such as numbers, strings etc. For example:

```
var a=40;//holding number
var b="Rahul";//holding string
```

JavaScript primitive data types

There are five types of primitive data types in JavaScript. They are as follows:

Data Type	Description
String	represents sequence of characters e.g. "hello"
Number	represents numeric values e.g. 100
Boolean	represents boolean value either false or true
Undefined	represents undefined value
Null	represents null i.e. no value at all

JavaScript non-primitive data types

The non-primitive data types are as follows:

Data Type	Description
Object	represents instance through which we can access members
Array	represents group of similar values
RegExp	represents regular expression

We will have great discussion on each data type later.

JavaScript If-else

The **JavaScript if-else statement** is used *to execute the code whether condition is true or false*. There are three forms of if statement in JavaScript.

1. If Statement

2. If else statement
3. if else if statement

JavaScript If statement

It evaluates the content only if expression is true. The signature of JavaScript if statement is given below.

```
if(expression){
//content to be evaluated
}
```

Let's see the simple example of if statement in javascript.

```
<script>
var a=20;
if(a>10){
document.write("value of a is greater than 10");
}
</script>
```

Output of the above example

```
value of a is greater than 10
```

Let's see the example of if-else statement in JavaScript to find out the even or odd number.

```
<script>
var a=20;
if(a%2==0){
document.write("a is even number");
}
else{
document.write("a is odd number");
}
</script>
```

Output of the above example

```
a is even number
```

JavaScript Switch

The **JavaScript switch statement** is used *to execute one code from multiple expressions*. It is just like else if statement that we have learned in previous page. But it is convenient than*if..else..if* because it can be used with numbers, characters etc.

The signature of JavaScript switch statement is given below.

```
switch(expression){
case value1:
 code to be executed;
 break;
case value2:
 code to be executed;
 break;
......

default:
 code to be executed if above values are not matched;
}
```

Let's see the simple example of switch statement in javascript.

```
<script>
var grade='B';
var result;
switch(grade){
case 'A':
result="A Grade";
break;
case 'B':
result="B Grade";
break;
case 'C':
result="C Grade";
break;
default:
result="No Grade";
}
document.write(result);
</script>
```

Output of the above example B Grade

JavaScript Loops

The **JavaScript loops** are used *to iterate the piece of code* using for, while, do while or for-in loops. It makes the code compact. It is mostly used in array.

There are four types of loops in JavaScript.

1. for loop
2. while loop
3. do-while loop
4. for-in loop

1) JavaScript For loop

The **JavaScript for loop** *iterates the elements for the fixed number of times*. It should be used if number of iteration is known. The syntax of for loop is given below.

```
for (initialization; condition; increment)
{
   code to be executed
}
```

Let's see the simple example of for loop in javascript.

```
<script>
for (i=1; i<=5; i++)
{
document.write(i + "<br/>")
}
</script>
```

Output:

```
1
2
3
4
5
```

2) JavaScript while loop

The **JavaScript while loop** *iterates the elements for the infinite number of times*. It should be used if number of iteration is not known. The syntax of while loop is given below.

```
while (condition)
{
   code to be executed
}
```

Let's see the simple example of while loop in javascript.

```
<script>
var i=11;
while (i<=15)
{
document.write(i + "<br/>");
i++;
}
</script>
```

Output:

```
11
12
13
14
15
```

3) JavaScript do while loop

The **JavaScript do while loop** *iterates the elements for the infinite number of times* like while loop. But, code is *executed at least* once whether condition is true or false. The syntax of do while loop is given below.

```
do{
   code to be executed
}while (condition);
```

Let's see the simple example of do while loop in javascript.

```
<script>
var i=21;
do{
document.write(i + "<br/>");
```

```
i++;
}while (i<=25);
</script>
```

Output:

```
21
22
23
24
25
```

4) JavaScript for in loop

The **JavaScript for in loop** is used *to iterate the properties of an object*. We will discuss about it later.

JavaScript Functions

JavaScript functions are used to perform operations. We can call JavaScript function many times to reuse the code.

Advantage of JavaScript function

There are mainly two advantages of JavaScript functions.

1. **Code reusability**: We can call a function several times so it save coding.
2. **Less coding**: It makes our program compact. We don't need to write many lines of code each time to perform a common task.

JavaScript Function Syntax

The syntax of declaring function is given below.

```
function functionName([arg1, arg2, ...argN]){
 //code to be executed
}
```

JavaScript Functions can have 0 or more arguments.

JavaScript Function Example

Let's see the simple example of function in JavaScript that does not has arguments.

```
<script>
function msg(){
alert("hello! this is message");
}
</script>
<input type="button" onclick="msg()" value="call function"/>
```

Output of the above example

Call Function button **hello! this is message**

JavaScript Function Arguments

We can call function by passing arguments. Let's see the example of function that has one argument.

```
<script>
function getcube(number){
alert(number*number*number);
}
</script>
<form>
<input type="button" value="click" onclick="getcube(4)"/>
</form>
```

Output of the above example

Click button 64

Function with Return Value

We can call function that returns a value and use it in our program. Let's see the example of function that returns value.

```
<script>
function getInfo(){
return "hello javatpoint! How r u?";
}
</script>
<script>
document.write(getInfo());
</script>
```

Output of the above example

```
hello javatpoint! How r u?
```

JavaScript Objects

A javaScript object is an entity having state and behavior (properties and method). For example: car, pen, bike, chair, glass, keyboard, monitor etc.

JavaScript is an object-based language. Everything is an object in JavaScript.

JavaScript is template based not class based. Here, we don't create class to get the object. But, we direct create objects.

Creating Objects in JavaScript

There are 3 ways to create objects.

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

1) JavaScript Object by object literal

The syntax of creating object using object literal is given below:

```
object={property1:value1,property2:value2.....propertyN:valueN}
```

As you can see, property and value is separated by : (colon).

Let's see the simple example of creating object in JavaScript.

```
<script>
emp={id:102,name:"Shyam Kumar",salary:40000}
document.write(emp.id+" "+emp.name+" "+emp.salary);
</script>
```

Output of the above example

```
102 Shyam Kumar 40000
```

2) By creating instance of Object

The syntax of creating object directly is given below:

```
var objectname=new Object();
```

Here, **new keyword** is used to create object.

Let's see the example of creating object directly.

```
<script>
var emp=new Object();
emp.id=101;
emp.name="Ravi Malik";
emp.salary=50000;
document.write(emp.id+" "+emp.name+" "+emp.salary);
</script>
```

Output of the above example

```
101 Ravi 50000
```

3) By using an Object constructor

Here, you need to create function with arguments. Each argument value can be assigned in the current object by using this keyword.

The **this keyword** refers to the current object.

The example of creating object by object constructor is given below.

```
<script>
function emp(id,name,salary){
this.id=id;
this.name=name;
this.salary=salary;
}
e=new emp(103,"Vimal Jaiswal",30000);

document.write(e.id+" "+e.name+" "+e.salary);
</script>
```

Output of the above example

```
103 Vimal Jaiswal 30000
```

Defining method in JavaScript Object

We can define method in JavaScript object. But before defining method, we need to add property in the function with same name as method.

The example of defining method in object is given below.

```
<script>
function emp(id,name,salary){
this.id=id;
this.name=name;
this.salary=salary;

this.changeSalary=changeSalary;
function changeSalary(otherSalary){
this.salary=otherSalary;
}
}
e=new emp(103,"Sonoo Jaiswal",30000);
document.write(e.id+" "+e.name+" "+e.salary);
e.changeSalary(45000);
document.write("<br>"+e.id+" "+e.name+" "+e.salary);
</script>
```

Output of the above example

103	Sonoo	Jaiswal	30000
103	Sonoo	Jaiswal	45000

Browser Object Model

The **Browser Object Model** (BOM) is used to interact with the browser.

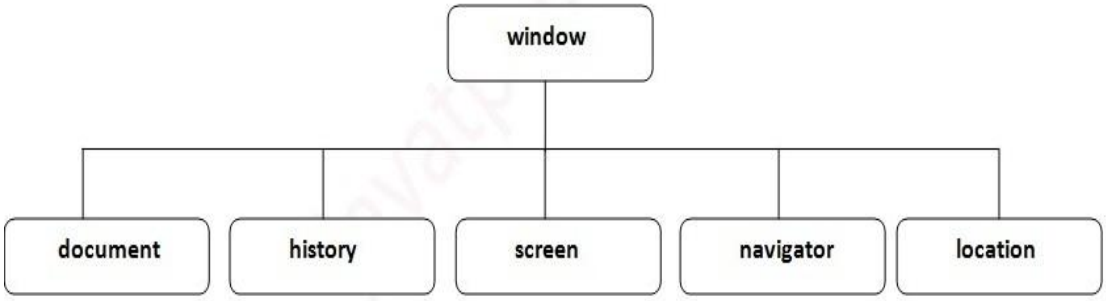
The default object of browser is window means you can call all the functions of window by specifying window or directly. For example:

1. window.alert("hello javatpoint"); is same as:

2. alert("hello javatpoint");

You can use a lot of properties (other objects) defined underneath the window object like document, history, screen, navigator, location, innerHeight, innerWidth,

Note: The document object represents an html document. It forms DOM (Document Object Model).



Window Object

The **window object** represents a window in browser. An object of window is created automatically by the browser.

Window is the object of browser, **it is not the object of javascript**. The javascript objects are string, array, date etc.

Note: if html document contains frame or iframe, browser creates additional window objects for each frame.

Methods of window object

The important methods of window object are as follows:

Method	Description
alert()	displays the alert box containing message with ok button.
confirm()	displays the confirm dialog box containing message with ok and cancel button.
prompt()	displays a dialog box to get input from the user.
open()	opens the new window.

close()	closes the current window.
setTimeout()	performs action after specified time like calling function, evaluating expressions etc.

Example of alert() in javascript

It displays alert dialog box. It has message and ok button.

```
<script type="text/javascript">
function msg(){
 alert("Hello Alert Box");
}
</script>
<input type="button" value="click" onclick="msg()"/>
```

Example of confirm() in javascript

It displays the confirm dialog box. It has message with ok and cancel buttons.

```
<script type="text/javascript">
function msg(){
var v= confirm("Are u sure?");
if(v==true){
alert("ok");
}
else{
alert("cancel");
}

}
</script>
 <input type="button" value="delete record" onclick="msg()"/>
```

JavaScript Navigator Object

The **JavaScript navigator object** is used for browser detection. It can be used to get browser information such as appName, appCodeName, userAgent etc.

The navigator object is the window property, so it can be accessed by:

window.navigator Or,

navigator

Property of JavaScript navigator object

There are many properties of navigator object that returns information of the browser.

No.	Property	Description
1	appName	returns the name
2	appVersion	returns the version
3	appCodeName	returns the code name
4	cookieEnabled	returns true if cookie is enabled otherwise false
5	userAgent	returns the user agent
6	language	returns the language. It is supported in Netscape and Firefox only.
7	userLanguage	returns the user language. It is supported in IE only.
8	plugins	returns the plugins. It is supported in Netscape and Firefox only.
9	systemLanguage	returns the system language. It is supported in IE only.
10	mimeTypes[]	returns the array of mime type. It is supported in Netscape and Firefox only.
11	platform	returns the platform e.g. Win32.
12	online	returns true if browser is online otherwise false.

Methods of JavaScript navigator object

The methods of navigator object are given below.

No.	Method	Description
1	javaEnabled()	checks if java is enabled.

2	taintEnabled()	checks if taint is enabled. It is deprecated since JavaScript 1.2.

Example of navigator object

Let's see the different usage of history object.

```
<script>
document.writeln("<br/>navigator.appCodeName: "+navigator.appCodeName);
document.writeln("<br/>navigator.appName: "+navigator.appName);
document.writeln("<br/>navigator.appVersion: "+navigator.appVersion);
document.writeln("<br/>navigator.cookieEnabled: "+navigator.cookieEnabled);
document.writeln("<br/>navigator.language: "+navigator.language);
document.writeln("<br/>navigator.userAgent: "+navigator.userAgent);
document.writeln("<br/>navigator.platform: "+navigator.platform);
document.writeln("<br/>navigator.onLine: "+navigator.onLine);
</script>
```

```
navigator.appCodeName: Mozilla
navigator.appName: Netscape
navigator.appVersion: 5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/37.0.2062.124 Safari/537.36
navigator.cookieEnabled: true
navigator.language: en-US
navigator.userAgent: Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/37.0.2062.124 Safari/537.36
navigator.platform: Win32
navigator.onLine: true
```

JavaScript Form Validation

It is important to validate the form submitted by the user because it can have inappropriate values. So validation is must.

The JavaScript provides you the facility the validate the form on the client side so processing will be fast than server-side validation. So, most of the web developers prefer JavaScript form validation.

Through JavaScript, we can validate name, password, email, date, mobile number etc fields.

JavaScript form validation example

In this example, we are going to validate the name and password. The name can't be empty and password can't be less than 6 characters long.

Here, we are validating the form on form submit. The user will not be forwarded to the next page until given values are correct.

```
<script>
function validateform(){
var name=document.myform.name.value;
var password=document.myform.password.value;

if (name==null || name==""){
  alert("Name can't be blank");
  return false;
}else if(password.length<6){
  alert("Password must be at least 6 characters long.");
  return false;
  }
}
</script>
<body>
<form name="myform" method="post" action="abc.jsp" onsubmit="return validateform
()" >
Name: <input type="text" name="name"><br/>
Password: <input type="password" name="password"><br/>
<input type="submit" value="register">
</form>
```

JavaScript Retype Password Validation

```
<script type="text/javascript">
function matchpass(){
var firstpassword=document.f1.password.value;
var secondpassword=document.f1.password2.value;

if(firstpassword==secondpassword){
return true;
}
else{
alert("password must be same!");
return false;
}
}
</script>

<form name="f1" action="register.jsp" onsubmit="return matchpass()">
Password:<input type="password" name="password" /><br/>
```

```
Re-enter Password:<input type="password" name="password2"/><br/>
<input type="submit">
</form>
```

JavaScript Number Validation

Let's validate the textfield for numeric value only. Here, we are using isNaN() function.

```
<script>
function validate(){
var num=document.myform.num.value;
if (isNaN(num)){
 document.getElementById("numloc").innerHTML="Enter Numeric value only";
 return false;
}else{
 return true;
 }
}
</script>
<form name="myform" onsubmit="return validate()" >
Number: <input type="text" name="num"><span id="numloc"></span><br/>
<input type="submit" value="submit">
</form>
```

HTML/DOM events for JavaScript

HTML or DOM events are widely used in JavaScript code. JavaScript code is executed with HTML/DOM events. So before learning JavaScript, let's have some idea about events.

Events	Description
onclick	occurs when element is clicked.
ondblclick	occurs when element is double-clicked.
onfocus	occurs when an element gets focus such as button, input, textarea etc.
onblur	occurs when form looses the focus from an element.

onsubmit	occurs when form is submitted.
onmouseover	occurs when mouse is moved over an element.
onmouseout	occurs when mouse is moved out from an element (after moved over).
onmousedown	occurs when mouse button is pressed over an element.
onmouseup	occurs when mouse is released from an element (after mouse is pressed).
onload	occurs when document, object or frameset is loaded.
onunload	occurs when body or frameset is unloaded.
onscroll	occurs when document is scrolled.
onresized	occurs when document is resized.
onreset	occurs when form is reset.
onkeydown	occurs when key is being pressed.
onkeypress	occurs when user presses the key.
onkeyup	occurs when key is released.

Window frames Property

Definition and Usage

The frames property returns an array-like object, which represents all <iframe> elements in the current window.

The <iframe> elements can be accessed by index numbers. The index starts at 0.
Tip: Use frames.length to find the number of frames.

Note: This property will also work for <frame> elements. However, the <frame> element is not supported in HTML5.
This property is read-only.

Syntax
window.frames

Example

```
<!DOCTYPE html>
<html>
<body>
<p>Click the button to loop through the frames on this page, and change the location of every frame to "w3schools.com".</p>
<button onclick="myFunction()">Try it</button>
<br><br>
<iframe src="https://www.cnn.com"></iframe>
<iframe src="https://www.bbc.com"></iframe>
<iframe src="https://www.nytimes.com"></iframe>
<script>
function myFunction() {
  var frames = window.frames;
  var i;
  for (i = 0; i < frames.length; i++) {
    frames[i].location = "https://www.w3schools.com";
  }
}
</script>
</body>
</html>
```

POSSIBLE QUESTIONS

PART A – Multiple Choice Questions

1. _____ JavaScript is also called client-side JavaScript.
   a)Microsoft b)Navigator c)LiveWire d)Native

2. _____ tag is an extension to HTML that can enclose any number of JavaScript statements.
   a)<SCRIPT> b)<BODY> c)<HEAD> d)<TITLE>

3. Inside which HTML element do we put the JavaScript?
   a)<js> b)<scripting> c)<script> d)<javascript>
4. Choose the server-side JavaScript object?
   a)FileUpLoad b)Function c)File d)Date

5. JavaScript is _____ Side Scripting Language.

a)ISP b)Browser c)Server d) none

PART B – 2 Mark Questions

1. How to Put a JavaScript into an HTML Page?
2. How to create Objects in JavaScript
3. What are the ways of making comments in JavaScript?
4. How to create arrays in JavaScript?
5. What is the use of Math Object in JavaScript?

PART C – 8 Mark Questions

1. Explain about functions and objects in Javascript.
2. Describe about scripting frames and multiple windows in detail.
3. Write a coding to change the font color on reloading a webpage by using JavaScript
4. Explain the following:
   (i) for statements (ii) if –else with example
5. Describe JavaScript navigator objects with suitable examples
6. Discuss about Web forms in detail.
7. Write a coding to generate web page that represents clock-every 60 see the page updated with server current time Using JavaScript

KARPAGAM ACADEMY OF HIGHER EDUCATION

DEPARTMENT OF CS,CA & IT

I M.Sc CS

WEB TECHNOLOGY (17CSP101)

UNIT I

S.No	Questions	Choice 1	Choice 2	Choice 3	Choice 4	Answer
1	Why so JavaScript and Java have similar name?	JavaScript is a	JavaScript's	They both originated on	None of the	JavaScript's
2	_____ JavaScript is also called client-side JavaScript.	Microsoft	Navigator	C. LiveWire	Native	Navigator
3	What are variables used for in JavaScript Programs?	Storing numbers, dates, or other values	Varying randomly	Causing high-school algebra flashbacks	None of the above	Storing numbers, dates, or other
4	_____ JavaScript is also called server-side JavaScript.	Microsoft	Navigator	LiveWire	Native	LiveWire
5	Which of the following are capabilities of functions in	Return a value	Accept	Accept parameters	None of the	Accept
6	_____ tag is an extension to HTML that can enclose	<SCRIPT>	<BODY>	<HEAD>	<TITLE>	<SCRIPT>
7	What is the correct JavaScript syntax to write "Hello World"?	System.out.println("Hello World")	println ("Hello World")	document.write("Hello World")	response.write("Hello World")	document.write("Hello World")
8	Inside which HTML element do we put the JavaScript?	<js>	<scripting>	<script>	<javascript>	<script>
9	JavaScript entities start with _____ and end with	Semicolon, colon	Semicolon,	Ampersand, colon	Ampersand,	Ampersand,
10	Choose the server-side JavaScript object?	FileUpLoad	Function	File	Date	File
11	Choose the client-side JavaScript object?	Database	Cursor	Client	FileUpLoad	FileUpLoad
12	Which of the following is not considered a JavaScript	new	this	delete	typeof	this
13	JavaScript is interpreted by _____	Client	Server	Object	None of the	Client
14	Using _____ statement is how you test for a specific	Select	If	Switch	For	If
15	How to create a Date object in JavaScript?	dateObjectName = new	dateObjectName .new	dateObjectName := new Date([parameters])	dateObjectName	dateObjectName = new
16	The _____ method of an Array object adds and/or	Reverse	Shift	Slice	Splice	Splice
17	To set up the window to capture all Click events, we use which of the following statement?	window.captureEvents(Event.CLICK);	window.handleEvents	window.routeEvents(Event.CLICK);	window.raiseEvents(Event.CLIC	window.captureEvents(Event.
18	Which tag(s) can handle mouse events in Netscape?	<IMG>	<A>	 	None of the	<A>

19	_____ is the tainted property of a window	Pathname	Protocol	Defaultstatus	Host	Defaultstatus
20	What is mean by "this" keyword in javascript?	It refers current	It referes	It is variable which	None of the	It refers
21	Choose the client-side JavaScript object:	Database	Cursor	Client	FileUpLoad	FileUpLoad
22	Which best explains getSelection()?	Returns the VALUE of a selected OPTION	Returns document.URL	Returns the value of cursor-selected text	Returns the VALUE of a	Returns the value of cursor-
23	Scripting language are	High Level	Assembly Level	Machine level	None of the	High Level
24	The syntax of close method for document object is _	Close(doC.	Close(object)	Close(val)	Close()	Close()
25	The syntax of capture events method for document object is	captureEvents()	captureEvents(args eventType)	captureEvents(eventType)	captureEvents(eventVal)	captureEvents(eventType)
26	The syntax of a blur method in a button object is	Blur()	Blur(contrast)	Blur(value)	Blur(depth)	Blur()
27	The JavaScript exception is available to the Java code as an instance of	netscape.javascript.JSObject	netscape.javascript.JSException	netscape.plugin.JSException	None of the above	netscape.javascript.JSExcepti
28	A _____ object is a reference to one of the classes in	JavaArray	JavaClass	JavaObject	JavaPackage	JavaClass
29	_____ is a wrapped Java array, accessed from	JavaArray	JavaClass	JavaObject	JavaPackage	JavaArray
30	When a JavaScript object is sent to Java, the runtime	ScriptObject	JSObject	JavaObject	Jobject	JSObject
31	_____method evaluates a string of JavaScript code in	Eval	ParseInt	ParseFloat	Efloat	Eval
32	Javascript is _____ language.	Programming	Application	None of These	Scripting	Scripting
33	JavaScript is _____ Side Scripting Language.	ISP	Browser	Server	none of the	Browser
34	JavaScript is designed for following purpose -	To Perform Server	To add	To Execute Query	To Style HTML	To add
35	JavaScript Code is written inside file having extension	.javascript	.jsc	.jvs	.js	.js
36	_____ attribute is used to specify the character	None of These	type	charset	character	charset
37	Which built-in method combines the text of two strings	append()	concat()	attach()	None of the	concat()
38	Which of the following code creates an object?	var book = Object();	var book = new Object();	var book = new OBJECT();	var book = new Book();	var book = new Object();
39	Which of the following function of Array object returns a	toSource()	sort()	splice()	toString()	toSource()
40	Which built-in method returns the characters in a string	substr()	getSubstring()	slice()	None of the	substr()
41	Which of the following function of String object returns	charAt()	charCodeAt()	concat()	indexOf()	charAt()
42	Which of the following code creates an object?	var book = Object();	var book = new Object();	var book = new OBJECT();	var book = new Book()	var book = new Object();

43	Which built-in method calls a function for each element in the array?	while()	loop()	forEach()	None of the above.	forEach()
44	Which built-in method sorts the elements of an array?	changeOrder(order)	order()	sort()	None of the above	sort()
45	The type of a variable that is volatile is	Volatile variable	Mutable variable	Immutable variable	Dynamic	Mutable
46	Which of the following is not considered as an error in	Syntax error	Missing of	Division by zero	All of the above	Division by
47	The escape sequence '\f' stands for	Floating numbers	Representation	\f is not present in	Form feed	Form feed
48	A function definition expression can be called	Function prototype	Function literal	Function definition	Function	Function
49	Which of the operator is used to test if a particular	in	exist	within	exists	in
50	Among the following, which one is a ternary operator?	+	:	?:	-	?:
51	The "var" and "function" are	Keywords	Declaration	Datatypes	Prototypes	Declaration
52	What does the subexpression /java(script)?/ result in ?	It matches "java"	It matches "java"	It matches "java"	None of the	It matches
53	Which function among the following lets to register a	setTimeout()	setTotaltime()	setInterval()	None of the	setTimeout()
54	Which property is used to obtain browser vendor and	modal	version	browser	navigator	navigator
55	The setTimeout() belongs to which object?	Element	Window	Location	None of the	Window
56	To which object does the location property belong?	Window	Position	Element	Location	Window
57	Which method receives the return value	clearTimeout()	clearInterval()	clearSchedule()	None of the	clearTimeout()
58	The Cookie manipulation is done using which property?	cookie	cookies	manipulate	None of the	cookie
59	Which of the following explains Cookies nature?	Non Volatile	Volatile	Intransient	Transient	Transient
60	Which of the following is a boolean cookie attribute?	bool	secure	lookup	domain	secure

UNIT-II

ASP.NET: **ASP & ASP.NET**: An Overview – Programming ASP.NET with VB.NET: ASP Data types – operators- Request Object- Response Object – Server object - Web forms and ASP.NET: Web forms

ASP.NET: ASP & ASP.NET

What is ASP?

- ASP stands for **A**ctive **S**erver **P**ages
- ASP is a Microsoft Technology
- ASP is a program that runs inside a web server

What is an ASP File?

- An ASP file has the file extension ".asp"
- An ASP file is just the same as an HTML file
- An ASP file can contain server scripts in addition to HTML
- Server scripts in an ASP file are executed on the server

What can ASP do for you?

- Edit, change, add content, or customize any web page
- Respond to user queries or data submitted from HTML forms
- Access databases or other server data and return results to a browser
- Provide web security since ASP code cannot be viewed in a browser
- Offer simplicity and speed

How Does it Work?

When a browser requests a normal HTML file, the server just returns the file.

When a browser requests an ASP file, the server passes the request to the ASP engine which reads the ASP file and executes the server scripts in the file.

Finally the ASP file is returned to the browser as plain HTML.

Example

```
<!DOCTYPE html>
<html>
<body>
<%
response.write("My first ASP script!")
%>
</body>
</html>
```

OUTPUT My first ASP script!

The ASP Technology

ASP and ASP.NET are server side technologies.
Both technologies enable computer code to be executed by an Internet server.
When a browser requests an ASP or ASP.NET file, the ASP engine reads the file, executes any code in the file, and returns the result to the browser.

Classic ASP - Active Server Pages

ASP (aka Classic ASP) was introduced in 1998 as Microsoft's first server side scripting language.
Classic ASP pages have the file extension **.asp** and are normally written in VBScript.

Asp.Net Overview

ASP.NET web pages are simply pure text, like HTML files. ASP.NET web pages are the main building block for application development. You can develop your applications in any language compatible with the common language runtime, including Microsoft Visual Basic and C#. These languages enable you to develop ASP.NET applications that benefit from the common language runtime, type safety, inheritance, and so on. ASP.NET incorporates all the important standards of our time, such as XML and SOAP, plus with ADO.NET and the foundation class libraries.

ASP.NET

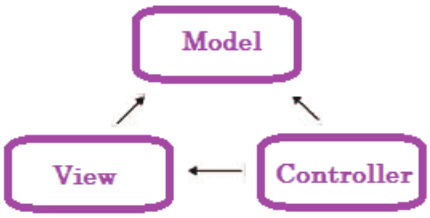
ASP.NET was released in 2002 as a successor to Classic ASP.
ASP.NET pages have the extension **.aspx** and are normally written in C# (C sharp).
ASP.NET 4.6 is the latest official version of ASP.NET.
ASP.NET 5 was expected to be an important redesign of ASP.NET.
However, the development of ASP.NET 5 was stopped in favor of ASP.NET Core.

ASP.NET Web Pages

ASP.NET Web Pages is an SPA application model (Single Page Application).

The SPA model is quite similar to PHP and Classic ASP.
ASP.NET Web Pages is being merged into the new ASP.NET Core.

ASP.NET MVC



ASP.NET MVC is an MVC application model (Model-View-Controller).
ASP.NET MVC is being merged into the new ASP.NET Core.

Asp.Net Extension

You can extend the functionality of ASP.Net web pages by adding some extensions framework that released by Microsoft. ASP.NET AJAX, ASP.NET MVC Framework, ASP.NET Razor view engine, ASP.NET Dynamic Data, ASP.NET SignalR etc. are some of the popular Asp.Net extensions

ASP.NET Web API

ASP.NET API is an API application model (Application Programming Interface).
ASP.NET API is being merged into the new ASP.NET Core.
ASP.NET API is not covered in this tutorial.

ASP.NET Web Forms

ASP.NET Web Forms is an event driven application model.
ASP.NET Web Forms is **not** a part of the new ASP.NET Core.
ASP.NET Web Forms is **not** covered in this tutorial.

ASP.NET Core

ASP.NET Core was released in 2016.
ASP.NET Core merges ASP.NET MVC, ASP.NET Web API, and ASP.NET Web Pages into one application framework.
ASP.NET Core is not covered in this tutorial.

Programming ASP.NET with VB.NET

Visual Studio

Visual Studio .NET is an excellent development tool for constructing ASP.NET web applications. It provides all of the necessary tools and support for creating ASP.NET web applications. ASP.NET web applications are hosted by **Internet Information Server (IIS)**, which accepts requests from clients and optionally authenticates them before passing the requests on to the Web application.

What is ASP.NET ?

ASP.NET introduces entirely a new object-oriented execution model. In order to overcome the limitations of ASP (Active Server Pages) , Microsoft has developed a new technology called Microsoft ASP.NET . ASP.NET is more than just the next version of ASP technology, using its mature programming languages VB.NET and C#.

ASP.NET is built on the Common Language Runtime (CLR), allowing programmers to write ASP.NET code using any supported .NET languages like VB.NET and C#. Unlike the ASP runtime, ASP.NET uses the Common Language Runtime (CLR) provided by the .NET Framework. The Microsoft .NET Platform provides all of the tools and technologies that are needed to build distributed Web applications. ASP.NET is integrated with Visual Studio .NET, which provides a GUI designer, a rich toolbox, and a fully integrated debugger. In ASP.NET, you can write the HTML code in the .aspx file and the code for programming logic in the code-behind file (.aspx.vb or .aspx.cs). Also ASP.NET introduces two sets of controls, the HTML controls and the Web controls, which are collectively known as "server controls."

Asp.Net and .Net Framework

The .NET Framework is an essential technology for ASP.NET development. It provides the basic system services that support ASP.NET, as well as Windows Forms development, the new rich client development technology provided by .NET. The .NET Framework consists of two main parts:

1. The common language runtime : It provides a run-time environment for the execution of code written in any .NET language.

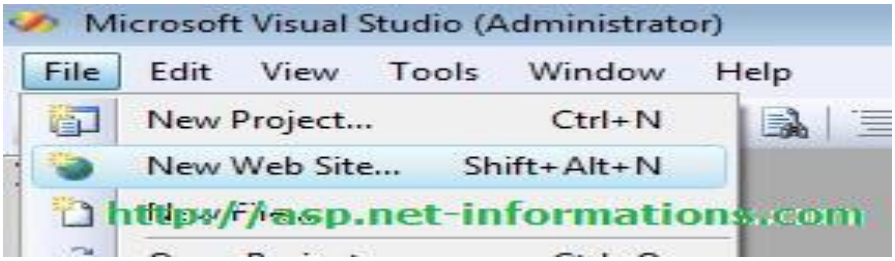
2. The .NET Framework class library : It is designed to support the efforts of developers by providing base classes from which developers can inherit.

Click the following link to learn .Net Framework in detail.Microsoft .Net Framework Step by Step

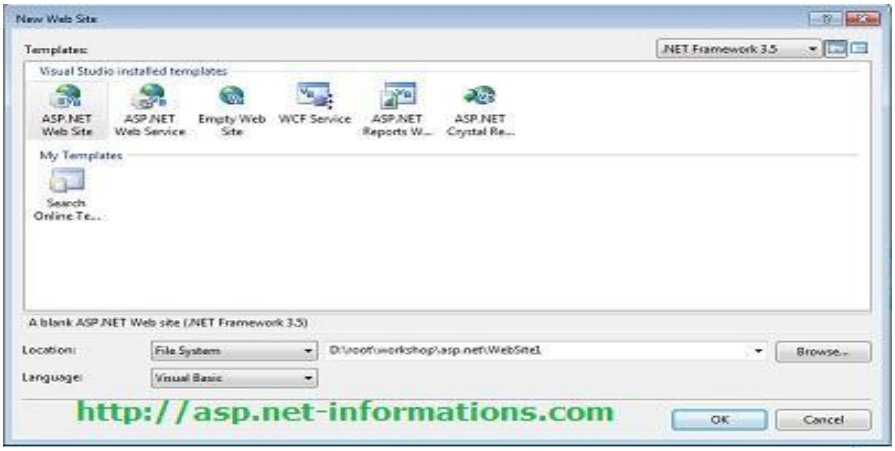
First Asp.Net Program

ASP.NET is integrated with Visual Studio .NET, which provides a GUI designer, a rich toolbox, and a fully integrated debugger. In ASP.NET, you can write the HTML code in the .aspx file and the code for programming logic in the code behind file (.aspx.vb or .aspx.cs) depends on which .Net language you are selected. Web Forms allow you to apply Rapid Application Development techniques to building web applications. Simply drag and drop controls onto your form, double-click on a control, and write the code respond to the associated event. You can program ASP.NET in any language that supports the .NET CLS.

Select Visual Studio from Start->All Programs . Select New Web Site from File menu of Visual Studio.



Then you can see a window similar to the following image.

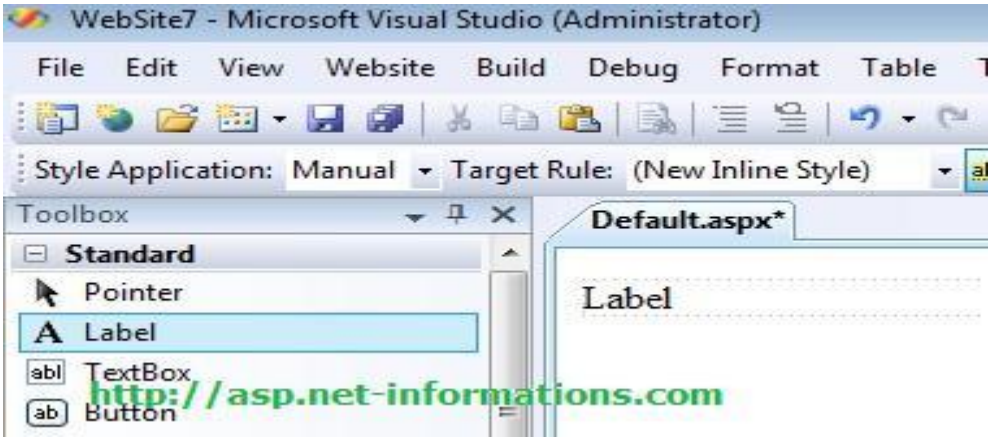


Select ASP.NET Web Site from the New Web Site dialog box. The Name and Location edit fields will contain default values. If you want to change the name , you can do it by editing the Location field. From this dialog box you can select which language (C# or VB.NET) you want to

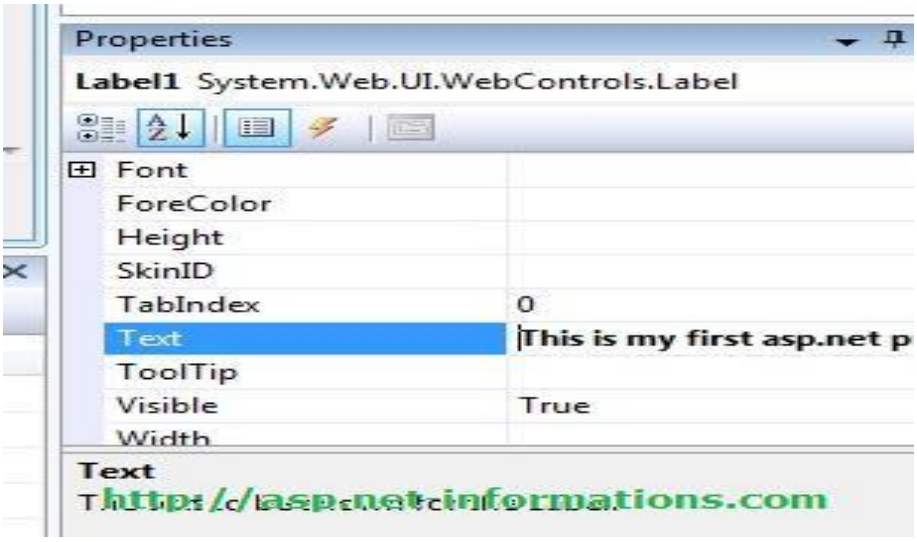
develop ASP.NET website. You can select your convenient language from the Language combobox.

After click the OK button you will get a blank form. You can select controls from the Toolbox which is located left side to the blank form.

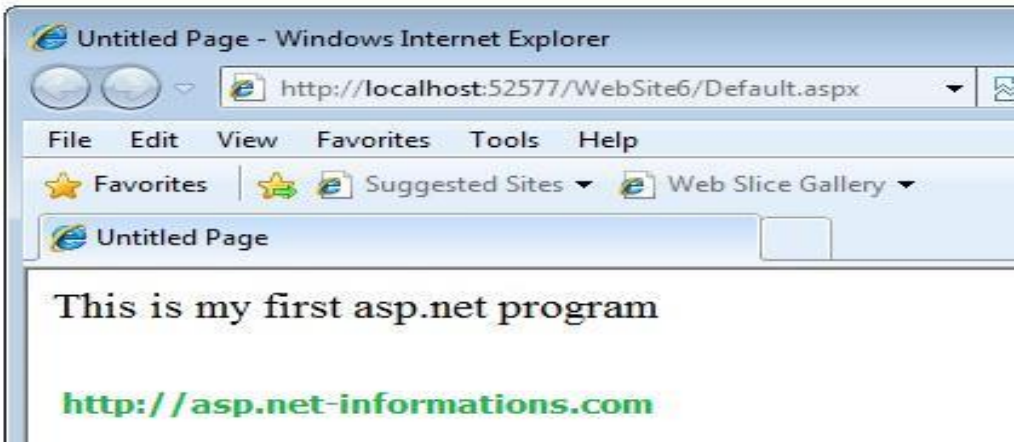
Click on the Label control and drag it to the design surface. It will automatically go to the upper-left corner of the design surface and contain the word Label.



Select the label control and change the Text property of label control to "This is my first asp.net program". The property window is placed right side down of your Visual Studio Environment.



After the coding, press F5 to run the web application. When either the C# or VB.NET version is run, it will look like the browser shown.



Deploy an ASP.NET Web Application

When you want to deploy your ASP.NET program on IIS, all you need to do is set up an IIS virtual directory on the target machine and copy the application files to it. You can simply copy the application files to the directory structure on the production server, using one of the following tools:

1. The XCOPY command-line tool can copy files from one computer to another on an intranet or internal network.

2. Windows Explorer can copy files from one computer to another on an intranet or internal network.

3. You can use the FTP tool of your choice to copy application files from one computer to another over the Internet.

4. If your application has been developed using Visual Studio .NET, you can use the Copy Project command to copy files from one computer to another on an intranet or internal network.

ASP Data types

Variables is for temporary storage of data. the keywords is DIM.

Variables

A Variable is memory allocation for data storage. It has got an **address, name, value and type**

Address: Physical space in the computer memory where data is stored.

Name: Unique identifier for which it can be referred as. Naming convention:

A) a-z, A-Z, 0-9

B) It can not start with a number

C) It can not be part of reserved keywords of the language

D) No special characters except underscore.

Value: Data stored in the variable.

Type: Classification of data stored according to its format.

- String = Variable type that stores characters
- Integer = Variable type that stores whole numbers lesser that 32000 aprox.
- Long = Variable type that stores whole numbers greater that 32000 aprox.
- Decimal = Variable type that stores number that require a decimal value
- DateTime = Variable type that stores date values

Boolean = Variable type that stores either true or false values

```
<%
    'Declare variable and assign it a value
    DIM a AS Decimal = 50.14
    'show on the screen
    Response.Write(a)

    Response.Write("<BR>")
%>
```

Data types

Data Type	Storage Allocation	Value Range
Boolean	Depends on implementing platform	**True** or **False**
Byte	1 byte	0 through 255 (unsigned)
Char	2 bytes	0 through 65535 (unsigned)
Date	8 bytes	0:00:00 (midnight) on January 1, 0001 through 11:59:59 PM on December 31, 9999

Decimal	16 bytes	0 through +/- 79,228,162,514,264,337,593,543,950,335 (+/- 7.9...E+28) with no decimal point; 0 through +/- 7.9228162514264337593543950335 with 28 places to the right of the decimal
Double	8 bytes	-1.79769313486231570E+308 through -4.94065645841246544E-324, for negative values 4.94065645841246544E-324 through 1.79769313486231570E+308, for positive values
Integer	4 bytes	-2,147,483,648 through 2,147,483,647 (signed)
Long	8 bytes	-9,223,372,036,854,775,808 through 9,223,372,036,854,775,807(signed)
Object	4 bytes on 32-bit platform 8 bytes on 64-bit platform	Any type can be stored in a variable of type Object
SByte	1 byte	-128 through 127 (signed)
Short	2 bytes	-32,768 through 32,767 (signed)
Single	4 bytes	-3.4028235E+38 through -1.401298E-45 for negative values; 1.401298E-45 through 3.4028235E+38 for positive values
String	Depends on implementing platform	0 to approximately 2 billion Unicode characters
UInteger	4 bytes	0 through 4,294,967,295 (unsigned)
ULong	8 bytes	0 through 18,446,744,073,709,551,615 (unsigned)
User-Defined	Depends on implementing platform	Each member of the structure has a range determined by its data type and independent of the ranges of the other members
UShort	2 bytes	0 through 65,535 (unsigned)

EXAMPLE

```
Private Sub btnEvaluate_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnEvaluate.Click
    Dim Quantity As Integer
    Dim UnitPrice As Double
    Dim TotalPrice As Double
    Dim strCD As String

    Quantity = CInt(txtQuantity.Text)

    UnitPrice = 5.5
    If Quantity < 50 Then UnitPrice = 6.25

    txtUnitPrice.Text = UnitPrice
    TotalPrice = Quantity * UnitPrice
    txtTotalPrice.Text = TotalPrice
End Sub
```

ASP OPERATORS

ASP is programmed in VBScript by default, thus ASP's operators are VBScript operators by default.

Operators in ASP fall into four categories Math, Comparisons, the somewhat more advanced Logic operators, and Leftovers(those that don't fit well into any category).

asp arithmetic operators

The mathematical operators in ASP are similar to many other programming languages. However, ASP does not support shortcut operators like ++, --, +=, etc.

Operator	English	Example	Result
+	Addition	myNum = 3 + 4	myNum = 7
-	Subtraction	myNum = 4 - 1	myNum = 3
*	Multiplication	myNum = 3 * 2	myNum = 6

/	Division	myNum = 9 / 3	myNum = 3
^	Exponential	myNum = 2 ^ 4	myNum = 16
Mod	Modulus	myNum = 23 Mod 10	myNum = 3
-	Negation	myNum = -10	myNum = -10
\	Integer Division	myNum = 9 \ 3	myNum = 3

comparison operators

Comparison operators are used when you want to compare two values to make a decision. Comparison operators are most commonly used in conjunction with "If...Then" and "While something is true do this..." statements, otherwise known as conditional statements. The items that are most often compared are numbers. The result of a comparison operator is either TRUE or FALSE.

Operator	English	Example	Result
=	Equal To	4 = 3	False
<	Less Than	4 < 3	False
>	Greater Than	4 > 3	True
<=	Less Than Or Equal To	4 <= 3	False
>=	Greater Than Or Equal To	4 >= 3	True
<>	Not Equal To	4 <>3	True

logical operators

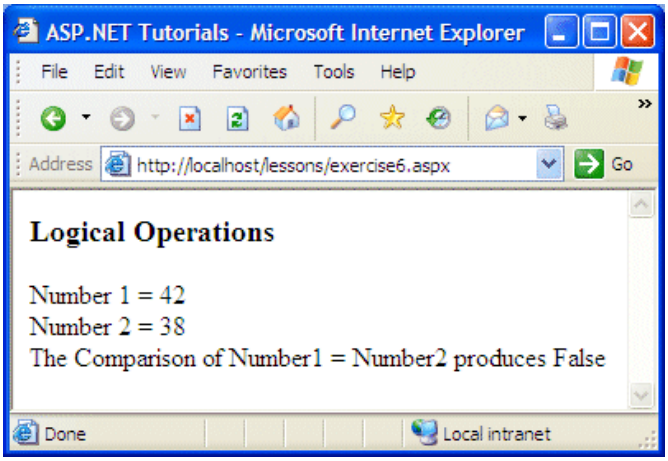
The above comparison operators result in a truth value of TRUE or FALSE. A logical operator is used for complex statements that must make decisions based on one or more of these truth values.

Operator	English	Example	Result

And	Both Must be TRUE	True and False	False
Or	One Must be TRUE	True or False	True
Not	Flips Truth Value	Not True	False

```
<%@ Page Language="VB" %>
<html>
<head>
<title>ASP.NET Tutorials</title>
</head>
<body>
<h3>Logical Operations</h3>

<%
   Dim Number1 As Integer = 42
   Dim Number2 As Integer = 38

   Response.Write("Number 1 = " & Number1)
   Response.Write("<br>")
   Response.Write("Number 2 = " & Number2)
   Response.Write("<br>")
   Response.Write("The Comparison of Number1 = Number2 produces " & _
            (Number1 = Number2))
%>

</body>
</html>
```

This would produce:



stringoperators

The only string operator is the string concatenation operator "&" that takes two strings and slams them together to form a new string. An example would be string1 = "Tim" and string2 = " is a Hero". The following code would combine these two strings into one: string3 = string1 & string2

Operator	English	Example	Result
&	String Concatenation	string4 = "Bob" & " runs"	string4 = "Bob runs"

We will be using these operators throughout the tutorial, so chances are you will get understand them more and more as this tutorial goes on.

ASP.NET - Server Side

We have studied the page life cycle and how a page contains various controls. The page itself is instantiated as a control object. All web forms are basically instances of the ASP.NET Page class. The page class has the following extremely useful properties that correspond to intrinsic objects:

- Session
- Application
- Cache
- Request
- Response
- Server
- User

- Trace

We will discuss each of these objects in due time. In this tutorial we will explore the Server object, the Request object, and the Response object.

Server Object

The Server object in Asp.NET is an instance of the System.Web.HttpServerUtility class. The HttpServerUtility class provides numerous properties and methods to perform various jobs.

Properties and Methods of the Server object

The methods and properties of the HttpServerUtility class are exposed through the intrinsic Server object provided by ASP.NET.

The following table provides a list of the properties:

Property	Description
MachineName	Name of server computer
ScriptTimeOut	Gets and sets the request time-out value in seconds.

The following table provides a list of some important methods:

Method	Description
CreateObject(String)	Creates an instance of the COM object identified by its ProgID (Programmatic ID).
CreateObject(Type)	Creates an instance of the COM object identified by its Type.
Equals(Object)	Determines whether the specified Object is equal to the current Object.
Execute(String)	Executes the handler for the specified virtual path in the context of the current request.
Execute(String, Boolean)	Executes the handler for the specified virtual path in the context of the current request and specifies whether to clear the QueryString and Form collections.
GetLastError	Returns the previous exception.

GetType	Gets the Type of the current instance.
HtmlEncode	Changes an ordinary string into a string with legal HTML characters.
HtmlDecode	Converts an Html string into an ordinary string.
ToString	Returns a String that represents the current Object.
Transfer(String)	For the current request, terminates execution of the current page and starts execution of a new page by using the specified URL path of the page.
UrlDecode	Converts an URL string into an ordinary string.
UrlEncodeToken	Works same as UrlEncode, but on a byte array that contains Base64-encoded data.
UrlDecodeToken	Works same as UrlDecode, but on a byte array that contains Base64-encoded data.
MapPath	Return the physical path that corresponds to a specified virtual file path on the server.
Transfer	Transfers execution to another web page in the current application.

Request Object

The request object is an instance of the System.Web.HttpRequest class. It represents the values and properties of the HTTP request that makes the page loading into the browser.

The information presented by this object is wrapped by the higher level abstractions (the web control model). However, this object helps in checking some information such as the client browser and cookies.

Properties and Methods of the Request Object

The following table provides some noteworthy properties of the Request object:

Property	Description
AcceptTypes	Gets a string array of client-supported MIME accept

	types.
ApplicationPath	Gets the ASP.NET application's virtual application root path on the server.
Browser	Gets or sets information about the requesting client's browser capabilities.
ContentEncoding	Gets or sets the character set of the entity-body.
ContentLength	Specifies the length, in bytes, of content sent by the client.
ContentType	Gets or sets the MIME content type of the incoming request.
Cookies	Gets a collection of cookies sent by the client.
FilePath	Gets the virtual path of the current request.
Files	Gets the collection of files uploaded by the client, in multipart MIME format.
Form	Gets a collection of form variables.
Headers	Gets a collection of HTTP headers.
HttpMethod	Gets the HTTP data transfer method (such as GET, POST, or HEAD) used by the client.
InputStream	Gets the contents of the incoming HTTP entity body.
IsSecureConnection	Gets a value indicating whether the HTTP connection uses secure sockets (that is, HTTPS).
QueryString	Gets the collection of HTTP query string variables.
RawUrl	Gets the raw URL of the current request.
RequestType	Gets or sets the HTTP data transfer method (GET or POST) used by the client.
ServerVariables	Gets a collection of Web server variables.

TotalBytes	Gets the number of bytes in the current input stream.
Url	Gets information about the URL of the current request.
UrlReferrer	Gets information about the URL of the client's previous request that is linked to the current URL.
UserAgent	Gets the raw user agent string of the client browser.
UserHostAddress	Gets the IP host address of the remote client.
UserHostName	Gets the DNS name of the remote client.
UserLanguages	Gets a sorted string array of client language preferences.

The following table provides a list of some important methods:

Method	Description
BinaryRead	Performs a binary read of a specified number of bytes from the current input stream.
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from object.)
GetType	Gets the Type of the current instance.
MapImageCoordinates	Maps an incoming image-field form parameter to appropriate x-coordinate and y-coordinate values.
MapPath(String)	Maps the specified virtual path to a physical path.
SaveAs	Saves an HTTP request to disk.
ToString	Returns a String that represents the current object.
ValidateInput	Causes validation to occur for the collections accessed through the Cookies, Form, and QueryString properties.

Response Object

The Response object represents the server's response to the client request. It is an instance of the System.Web.HttpResponse class.

In ASP.NET, the response object does not play any vital role in sending HTML text to the client, because the server-side controls have nested, object oriented methods for rendering themselves.

However, the HttpResponse object still provides some important functionalities, like the cookie feature and the Redirect() method. The Response.Redirect() method allows transferring the user to another page, inside as well as outside the application. It requires a round trip.

Properties and Methods of the Response Object

The following table provides some noteworthy properties of the Response object:

Property	Description
Buffer	Gets or sets a value indicating whether to buffer the output and send it after the complete response is finished processing.
BufferOutput	Gets or sets a value indicating whether to buffer the output and send it after the complete page is finished processing.
Charset	Gets or sets the HTTP character set of the output stream.
ContentEncoding	Gets or sets the HTTP character set of the output stream.
ContentType	Gets or sets the HTTP MIME type of the output stream.
Cookies	Gets the response cookie collection.
Expires	Gets or sets the number of minutes before a page cached on a browser expires.
ExpiresAbsolute	Gets or sets the absolute date and time at which to remove cached information from the cache.
HeaderEncoding	Gets or sets an encoding object that represents the encoding for the current header output stream.
Headers	Gets the collection of response headers.
IsClientConnected	Gets a value indicating whether the client is still connected to the server.

Output	Enables output of text to the outgoing HTTP response stream.
OutputStream	Enables binary output to the outgoing HTTP content body.
RedirectLocation	Gets or sets the value of the Http Location header.
Status	Sets the status line that is returned to the client.
StatusCode	Gets or sets the HTTP status code of the output returned to the client.
StatusDescription	Gets or sets the HTTP status string of the output returned to the client.
SubStatusCode	Gets or sets a value qualifying the status code of the response.
SuppressContent	Gets or sets a value indicating whether to send HTTP content to the client.

The following table provides a list of some important methods:

Method	Description
AddHeader	Adds an HTTP header to the output stream. AddHeader is provided for compatibility with earlier versions of ASP.
AppendCookie	Infrastructure adds an HTTP cookie to the intrinsic cookie collection.
AppendHeader	Adds an HTTP header to the output stream.
AppendToLog	Adds custom log information to the InterNET Information Services (IIS) log file.
BinaryWrite	Writes a string of binary characters to the HTTP output stream.
ClearContent	Clears all content output from the buffer stream.
Close	Closes the socket connection to a client.

End	Sends all currently buffered output to the client, stops execution of the page, and raises the EndRequest event.
Equals(Object)	Determines whether the specified object is equal to the current object.
Flush	Sends all currently buffered output to the client.
GetType	Gets the Type of the current instance.
Pics	Appends a HTTP PICS-Label header to the output stream.
Redirect(String)	Redirects a request to a new URL and specifies the new URL.
Redirect(String, Boolean)	Redirects a client to a new URL. Specifies the new URL and whether execution of the current page should terminate.
SetCookie	Updates an existing cookie in the cookie collection.
ToString	Returns a String that represents the current Object.
TransmitFile(String)	Writes the specified file directly to an HTTP response output stream, without buffering it in memory.
Write(Char)	Writes a character to an HTTP response output stream.
Write(Object)	Writes an object to an HTTP response stream.
Write(String)	Writes a string to an HTTP response output stream.
WriteFile(String)	Writes the contents of the specified file directly to an HTTP response output stream as a file block.
WriteFile(String, Boolean)	Writes the contents of the specified file directly to an HTTP response output stream as a memory block.

Example

The following simple example has a text box control where the user can enter name, a button to send the information to the server, and a label control to display the URL of the client computer.

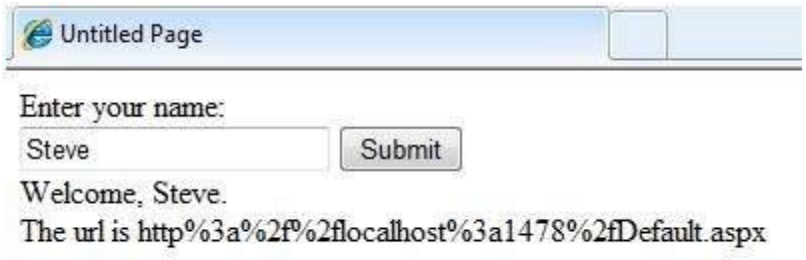
The content file:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
  Inherits="server_side._Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >

  <head runat="server">
    <title>Untitled Page</title>
  </head>

  <body>
    <form id="form1" runat="server">
      <div>

        Enter your name:
        <br />
        <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
        <asp:Button ID="Button1" runat="server" OnClick="Button1_Click" Text="Submit" />
        <br />
        <asp:Label ID="Label1" runat="server"/>

      </div>
    </form>
  </body>

</html>
```

The code behind Button1_Click:

```
protected void Button1_Click(object sender, EventArgs e) {


  if (!String.IsNullOrEmpty(TextBox1.Text)) {


    // Access the HttpServerUtility methods through

    // the intrinsic Server object.

    Label1.Text = "Welcome, " + Server.HtmlEncode(TextBox1.Text) + ". <br/> The url is " +
Server.UrlEncode(Request.Url.ToString())

  }

}
```

Run the page to see the following result:



ASP.NET Web Forms

ASP.NET is a development framework for building web pages and web sites with HTML, CSS, JavaScript and server scripting.

ASP.NET supports three different development models:
Web Pages, MVC (Model View Controller), and Web Forms.

What is Web Forms?

Web Forms is one of the 3 programming models for creating ASP.NET web sites and web applications.
The other two programming models are Web Pages and MVC (Model, View, Controller).
Web Forms is the oldest ASP.NET programming model, with event driven web pages written as a combination of HTML, server controls, and server code.
Web Forms are compiled and executed on the server, which generates the HTML that displays the web pages.

Web Forms comes with hundreds of different web controls and web components to build user-driven web sites with data access.

Visual Studio Express 2012/2010

Visual Studio Express is a free version of Microsoft Visual Studio.
Visual Studio Express is a development tool tailor made for Web Forms (and MVC).
Visual Studio Express contains:

- MVC and Web Forms
- Drag-and-drop web controls and web components
- A web server language (Razor using VB or C#)
- A web server (IIS Express)
- A database server (SQL Server Compact)
- A full web development framework (ASP.NET)

If you install Visual Studio Express, you will get more benefits from this tutorial.
If you want to install Visual Studio Express, click on one of these links:
Visual Web Developer 2012 (If you have Windows 7 or Windows 8)
Visual Web Developer 2010 (If you have Windows Vista or XP)

Hello W3Schools in ASP.NET

The simplest way to convert an HTML page into an ASP.NET page is to copy the HTML file to a new file with an **.aspx** extension.

This code displays our example as an ASP.NET page:

```
<html>
<body bgcolor="yellow">
<center>
<h2>Hello W3Schools!</h2>
</center>
</body>
</html>
```

If you want to try it yourself, save the code in a file called "**firstpage.aspx**", and create a link to the file like this: firstpage.aspx

How Does it Work?

Fundamentally an ASP.NET page is just the same as an HTML page.
An HTML page has the extension .htm. If a browser requests an HTML page from the server, the server sends the page to the browser without any modifications.
An ASP.NET page has the extension .aspx. If a browser requests an ASP.NET page, the server processes any executable code in the page, before the result is sent back to the browser.
The ASP.NET page above does not contain any executable code, so nothing is executed. In the next examples we will add some executable code to the page to demonstrate the difference between static HTML pages and dynamic ASP pages.

Classic ASP

Active Server Pages (ASP) has been around for several years. With ASP, executable code can be placed inside HTML pages.
Previous versions of ASP (before ASP .NET) are often called Classic ASP.
ASP.NET is not fully compatible with Classic ASP, but most Classic ASP pages will work fine as ASP.NET pages, with only minor changes.
If you want to learn more about Classic ASP, please visit our ASP Tutorial.

Dynamic Page in Classic ASP

To demonstrate how ASP can display pages with dynamic content, we have added some executable code (in red) to the previous example:

```
<html>
<body bgcolor="yellow">
<center>
<h2>Hello W3Schools!</h2>
<p><%Response.Write(Now())%></p>
</center>
</body>
</html>
```

The code inside the <% --%> tags is executed on the server.
Response.Write is ASP code for writing something to the HTML output stream.
Now() is a function returning the servers current date and time.
If you want to try it yourself, save the code in a file called "**dynpage.asp**", and create a link to the file like this: dynpage.asp

Dynamic Page in ASP .NET

This following code displays our example as an ASP.NET page:

```
<html>
<body bgcolor="yellow">
<center>
<h2>Hello W3Schools!</h2>
<p><%Response.Write(Now())%></p>
</center>
</body>
</html>
```

If you want to try it yourself, save the code in a file called "**dynpage.aspx**", and create a link to the file like this: dynpage.aspx

ASP.NET vs Classic ASP

The previous examples didn't demonstrate any differences between ASP.NET and Classic ASP. As you can see from the two latest examples there are no differences between the two ASP and ASP.NET pages.

In the next chapters you will see how server controls make ASP.NET more powerful than Classic ASP.

ASP.NET Web Forms - Server Controls

Server controls are tags that are understood by the server.

Limitations in Classic ASP

The listing below was copied from the previous chapter:

```
<html>
<body bgcolor="yellow">
<center>
<h2>Hello W3Schools!</h2>
<p><%Response.Write(now())%></p>
</center>
</body>
</html>
```

The code above illustrates a limitation in Classic ASP: The code block has to be placed where you want the output to appear.

With Classic ASP it is impossible to separate executable code from the HTML itself. This makes the page difficult to read, and difficult to maintain.

ASP.NET - Server Controls

ASP.NET has solved the "spaghetti-code" problem described above with server controls.

Server controls are tags that are understood by the server.

There are three kinds of server controls:

- HTML Server Controls - Traditional HTML tags
- Web Server Controls - New ASP.NET tags
- Validation Server Controls - For input validation

ASP.NET - HTML Server Controls

HTML server controls are HTML tags understood by the server.

HTML elements in ASP.NET files are, by default, treated as text. To make these elements programmable, add a runat="server" attribute to the HTML element. This attribute indicates that the element should be treated as a server control. The id attribute is added to identify the server control. The id reference can be used to manipulate the server control at run time.

Note: All HTML server controls must be within a <form> tag with the runat="server" attribute. The runat="server" attribute indicates that the form should be processed on the server. It also indicates that the enclosed controls can be accessed by server scripts.

In the following example we declare an HtmlAnchor server control in an .aspx file. Then we manipulate the HRef attribute of the HtmlAnchor control in an event handler (an event handler is a subroutine that executes code for a given event). The Page_Load event is one of many events that ASP.NET understands:

```
<script runat="server">
Sub Page_Load
link1.HRef="http://www.w3schools.com"
End Sub
</script>

<html>
<body>
```

```
<form runat="server">
<a id="link1" runat="server">Visit W3Schools!</a>
</form>

</body>
</html>
```

The executable code itself has been moved outside the HTML.

ASP.NET - Web Server Controls

Web server controls are special ASP.NET tags understood by the server.

Like HTML server controls, Web server controls are also created on the server and they require a runat="server" attribute to work. However, Web server controls do not necessarily map to any existing HTML elements and they may represent more complex elements.

The syntax for creating a Web server control is:

```
<asp:control_name id="some_id" runat="server" />
```

In the following example we declare a Button server control in an .aspx file. Then we create an event handler for the Click event which changes the text on the button:

```
<script runat="server">
Sub submit(Source As Object, e As EventArgs)
button1.Text="You clicked me!"
End Sub
</script>

<html>
<body>

<form runat="server">
<asp:Button id="button1" Text="Click me!"
runat="server" OnClick="submit"/>
</form>
```

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

ASP.NET - Validation Server Controls

Validation server controls are used to validate user-input. If the user-input does not pass validation, it will display an error message to the user.

Each validation control performs a specific type of validation (like validating against a specific value or a range of values).

By default, page validation is performed when a Button, ImageButton, or LinkButton control is clicked. You can prevent validation when a button control is clicked by setting the CausesValidation property to false.

The syntax for creating a Validation server control is:

```
<asp:control_name id="some_id" runat="server" />
```

In the following example we declare one TextBox control, one Button control, and one RangeValidator control in an .aspx file. If validation fails, the text "The value must be from 1 to 100!" will be displayed in the RangeValidator control:

Example

```
<html>
<body>

<form runat="server">
<p>Enter a number from 1 to 100:
<asp:TextBox id="tbox1" runat="server" />
<br /><br />
<asp:Button Text="Submit" runat="server" />
</p>

<p>
<asp:RangeValidator
ControlToValidate="tbox1"
MinimumValue="1"
```

```
MaximumValue="100"
Type="Integer"
Text="The value must be from 1 to 100!"
runat="server" />
</p>
</form>
</body>
</html>
```

POSSIBLE QUESTIONS

PART A – Multiple Choice Questions

1. Which of the following object is not an ASP component?

a)LinkCounter b)Counter c)AdRotator d)File Access

2. Default scripting language in ASP.

a)EcmaScript b)VBScript c)PERL d)JavaScript

3. _____ is the method which Creates an instance of an object

a)CreateObject b)Createobject c)createObject d)Create

4. The syntax of close method for document object is _

a)Close(doC. b)Close(object) c)Close(val) d)Close()

5. _____is a program that runs inside IIS

a)ASP b)HTML c)JavaScript d)DHTML

PART B – 2 Mark Questions

1. What is ASP.NET?
2. What is the basic difference between ASP and ASP.NET?
3. List out Properties and Methods of the Server object
4. What is an ASP.NET Web Form?
5. What is IIS? Why is it used?

PART C – 8 Mark Questions

1. Describe Response object with its collections, properties and methods.
2. Explain the data types in ASP.net in detail
3. Explain about ASP.Net session object.
4. Discuss ASP.NET Web forms with neat sketch
5. Illustrate ASP.NET application object with neat sketch.

KARPAGAM ACADEMY OF HIGHER EDUCATION

DEPARTMENT OF CS,CA & IT

I M.Sc CS

WEB TECHNOLOGY (17CSP101)

UNIT 2

S.No	Questions	Choice 1	Choice 2	Choice 3	Choice 4	Answer
1	Choose the form in which Postback occur	HTMLForms	Webforms	Winforms	None of the	Webforms
2	Which of the following object is not an ASP component?	LinkCounter	Counter	AdRotator	File Access	LinkCounter
3	The first event triggers in an aspx page is.	Page_Init()	Page_Load()	Page_click()	None of the	Page_Init()
4	What class does the ASP.NET Web Form class inherit from by default?	System.Web.UI.Page	System.Web.UI.Form	System.Web.GUI.Page	System.Web.Form	System.Web.UI.Page
5	We can manage states in asp.net application using	Session Objects	Application	Viewstate	All of the above	All of the above
6	Attribute must be set on a validator control for the validation to work.	ControlToValidate	ControlToBind	ValidateControl	Validate	ControlToValidate
7	Caching type supported by ASP.Net	Output Caching	DataCaching	a and b	none of the above	a and b
8	What is used to validate complex string patterns like an e-	Extended	Basic expressions	Regular	Irregular	Regular expressions
9	File extension used for ASP.NET files.	.Web	.ASP	.ASPX	None of the	.ASP
10	An alternative way of displaying text on web page using	asp:label	asp:listitem	asp:button	None of the	asp:label
11	Default Session data is stored in ASP.Net.	StateServer	Session Object	InProcess	all of the above	InProcess
12	Default scripting language in ASP.	EcmaScript	VBScript	PERL	JavaScript	VBScript
13	How do you get information from a form that is submitted	Request.QueryString	Request.Form	Response.write	Response.writeln	Request.Form
14	Which object can help you maintain data across users?	Application object	Session object	Response object	Server object	Application object
15	Which of the following ASP.NET object encapsulates the	Session object	Application object	Response object	Server object	Session object
16	Which of the following control is used to validate that two fields are equal?	RegularExpressionValidator	CompareValidator	equals() method	RequiredFieldValidator	CompareValidator
17	Mode of storing ASP.NET session	InProc	StateServer	SQL Server	All of the above	All of the above

18	In ASP.NET in form page the object which contains the user name is _____ ?	Page.User.Identity	Page.User.IsInRole	Page.User.Name	None of the Above	Page.User.Identity
19	Which of the following transfer execution directly to another page?	Server.Transfer	Response.Redirect	Both A. and B.	None of the Above	Server.Transfer
20	The actual work process of ASP.NET is taken care by _____?	inetinfo.exe	aspnet_isapi.dll	aspnet_wp.exe	None of the Above	aspnet_wp.exe
21	Which of the following allow writing formatted output?	Response.Write()	Response.Output.Write()	Both A. and B.	None of the Above	Response.Output.Write()
22	Which of the following denote the property in every	ControlToValidate	Text property	Both A. and B.	None of the	Both A. and B.
23	Which of the following languages can be used to write	C-sharp	VB	C++	A and B	A and B
24	When an .aspx page is requested from the web server, the	HTML	XML	WML	JSP	HTML
25	What's the difference between Response.Write() andResponse.Output.Write()?	Response.Output.Write() allows you to flush output	Response.Output.Write() allows you to buffer output	Response.Output.Write() allows you to write formatted	Response.Output.Write() allows you to stream output	Response.Output.Write() allows you to buffer output
26	Which property of the session object is used to set the	SessionId	LCID	Item	Key	Key
27	Select the caching type supported by ASP.Net	Output Caching	DataCaching	a and b	none of the above	DataCaching
28	Which one of the following namespaces contains the definition for IdbConnection?	System.Data.Interfaces	System.Data.Common	System.Data	System.Data.Connection	System.Data.Connection
29	Which of the following languages are used to write server	C-sharp	VB	Both C-sharp	C++	Both C-sharp and VB
30	How do you get information from a form that is submitted	Request.QueryString	Request.Form	Response.Write	Response.Writeln	Request.Form
31	Default scripting language in ASP.	EcmaScript	VBScript	PERL	JavaScript	VBScript
32	_____is a program that runs inside IIS	ASP	HTML	JavaScript	DHTML	ASP
33	"<%=" is the same as:	<%Equal	<%Write	<%Document.W	<%Response.Writ	<%Response.Write
34	_____ can be used to access databases from your web	ADO	AAO	ADA	AVO	ADO
35	_____ is a collection of html ducuments reside on the	Web Site	Page	Program	instruction	Web Site
36	_____ event is fired every time the web application is	OnEnd	OnStart	OnPause	OnStop	OnEnd
37	_____ is the property of session object that identifies the	Session ID	Session	Application ID	ObjectID	Session ID
38	_____ is the method which Gets and displays a content	ChooseContent	Choosecontent	ChooseCnt	ChooseCollection	ChooseContent

39	_____ is the method which Moves a specified file from	move()	mov	Moves()	Match()	move()
40	_____ is the method which Returns a Folder object for	GetFolder	Folder	getFolder	folder	GetFolder
41	_____ is the property of drive object which Returns the total size of a specified drive or network share	AvailableSpace	TotalSpace	TotalSize	Space	TotalSize
42	_____ is used to return information about a specified file.	The Drive	The Folder Object	The File object	The Directories Object	The File object
43	_____ is used to store and access variables from any page	Requst Object	The Application object	Session Object	The File object	The Application object
44	_____ is the method which Deletes a specified file	Delete	Del	DELETE()	D	Delete
45	_____ is the event of session object which Occurs	Session_OnEnd	Session_Onend	Session_OnStart	Session_onStart	Session_OnStart
46	_____ is the method which Creates an	CreateObject	Createobject	createObject	Create	CreateObject
47	_____ is the method which Executes an ASP	Execute	execute	Exe()	exe()	Execute
48	_____is a method used to end a session	Abandon	Aband	Quit	Terminate	Abandon
49	_____is the method which Copies one or more files	CopyFile	Copyfile	copyFile	Cpy	CopyFile
50	_____is the method which Returns HTML that	GetAdvertisement	Getadvertisement	GetAdvertiseme	GetAd	GetAdvertisement
51	_____is the method which Checks if a specified file	FileExists	Fileexists	Exists	exists	FileExists
52	_____is the method which executes the specified SQL	Execute	Start	Close	Stop	Execute
53	An ASP file has the file extension	.asp	.ap	.asa	.aspa	.asp
54	An ASP file has the file extension	.asp	.ap	.asa	.aspa	.asp
55	ASP is a	Microsoft	Sun Micro System	IBM	Infosys	Microsoft Technology
56	ASP is a _____	Server Sise Scripting	Client Side	Cilient Side	all the above	Server Sise Scripting
57	ASP server scripts are surrounded by delimiters, which?	<%...%>	<%>...</%>	<script>...</scri	<&>...</&>	<%...%>
58	How can you script your ASP code in JavaScript?	JavaScript is the	Start the	Start the	End the	Start the document
59	How do you create a FileSystemObject?	Server.CreateObject	Server.CreateObje	Create("FileSyst	Create	Server.CreateObject("
60	PWS Stands for	Personal Web Server	Personal Web	Personal Web	Procedures Web	Personal Web Server

UNIT-III

ASP.NET: ASP.NET Configuration Scope and State: Configuration – state- application – session object- ASP.NET Objects & Components: Scripting object models- ASP components and controls- ASP.NET and SQL server-Using SQL server using database in ASP.NET applications ActiveX data objects

ASP.NET Configuration Scope and State

ASP.NET Configurations

ASP.NET provides an XML-based configuration system that is easy to read and write. A configuration file is a text file that contains XML elements that affect the behavior of an application. In ASP.NET, configuration information is stored in one of two files: machine.config or web.config

How to Asp.Net configurations

After an ASP.NET application is finished, it needs to be configured on an application Web Server for launching it as a Web site on the network system. The configuration information for the ASP.NET applications is stored XML based configuration files, so they are readable and can be edited in any XML editor or in Visual Studio .NET directly.

In ASP.NET, configuration information is stored in one of two files: machine.config or web.config. The main difference between these two files is that the machine.config file is applied system level configuration while the web.config is implemented to each application based on the inheritance rules.

How to Machine.Config

In ASP.NET, configuration information is stored in one of two files: machine.config or web.config. Machine.config file is using as root configuration file and provides ASP.NET configuration settings for every application in the Web server. Machine.config configuration file comes with the Microsoft .NET Framework and contains the default settings. While an application can have as many web.config files as it has directories and subdirectories , there is only one machine.config file per machine.

The Machine.config file is located in the folder %windir%\Microsoft.NET\Framework\%version%\config, where %version% is a folder that is named for the current installed version of the .NET Framework. Machine.config file contains the default configuration information for every web application, as well as other application types, on the machine. The machine.config file is processed first and

any values specified within the machine .config file are inherited throughout every ASP.NET application on your Web server.

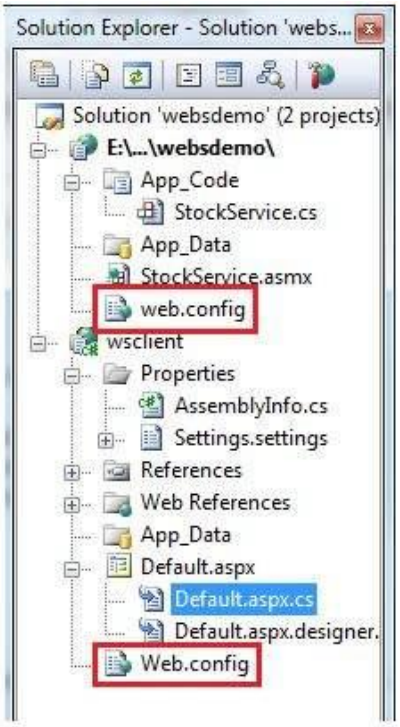
How to Web.Config

The web.config file is a text file which contains XML elements that affect the behavior of an application. It is efficient to store application-level configuration settings in XML-based files that are cached at runtime. Every ASP.NET application contains its own Web.config file , which resides in the root of the application.

The Web.config file overrides many of the settings in Machine.config, within the scope of the specific ASP.NET application where the Web.config file resides. The default Web.config file created by Visual Studio contains comments specific to the most commonly used settings. These comments provide guidance for using the available parameters for a given configuration element . ASP.NET enables you to manage the entire security configuration from the Web.config file.

Elements in a web.config File

The web.config file is a text file which contains XML elements that affect the behavior of an application. The following are some of the mportant tags in a web.config file.



<configuration>

In a Web.config file, all the configuration information for an ASP.NET application must reside between the <configuration> and </configuration> tags. This is the root node, which contains the declaration of all other sections of the Web.config file.

<appSettings>

This section of the Web.config file provides a way to define custom application settings for an application. The section can have multiple <add> subelements.

```
<appSettings>
        <add key="connectionstring"
value="localhost;uid=readonly;pwd=user"/>
</appSettings>
```

<compilation>

This element sets several compilation settings for the application. Some of the settings involve setting a default language and debug option for the application setting. Debug pages are larger and execute more slowly, so you should use them only for testing purposes. This section also provides support for the <compilers>, <assemblies> , and <namespaces> subelements.

<namespaces>

This subelement is used to add or remove namespace references for assemblies that must be made available when compiling Web pages.

<authentication>

The <authentication> element sets the authentication policy for the application. Possible modes are "Windows," "Forms," "Passport," and "None."

<pages>

The <pages> element allows configuration of page application specific settings.

<customErrors>

The <customErrors> element provides a means for defining custom error messages for an ASP.NET application. This is generally used to point users to a friendlier message than the default error messages. The <customErrors> element section supports multiple <error> subelements that are used to define custom errors.

ASP.NET State Management

HTTP is a stateless protocol, actually statelessness is one of the many advantages of the World Wide Web. In a statelessness condition all information associated with the page and the controls on the page would be lost with each round trip . Round trip means a

request from a user to Web Server and response from Webserver to user . There is no standard way in which the server can determine whether a subsequent HTTP request is from the same user.

To overcome this limitation of traditional Web programming, ASP.NET includes several features that help you preserve data on both a per-page basis and an application-wide basis by its State Management Capabilities. ASP.NET provides various options for state management such as Client-Based State Management Options and Server-Based State Management Options.

Client Based State Management
- Control State
- Hidden Fields
- Cookies
- Query Strings

Server Based State Management
- application state
- session state
- profile properties

Each option has distinct advantages and disadvantages, depending on the scenario. From the following chapters you can see some important state management features in detail.

ASP.NET View State

The ViewState indicates the status of the page when submitted to the server. The retention of state is called the view state and is stored within a hidden control on the page. When an ASP.NET page is submitted to the server, the state of the controls is encoded and sent to the server at every form submission in a hidden field known as __VIEWSTATE. The server sends back the variable so that when the page is re-rendered, the controls render at their last state, so no extra programming is needed. If you try to view source code from browser you can see that ASP .NET has added a hidden field in the form to maintain the ViewState

```
<input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE"
value="/wEPDwUKMTkwNjc4NTIwMWRkL5PbikezkXhkI2kmrh3VUjVDGoU=" />
```

Maintaining the ViewState is the default setting for ASP.NET Web Forms. By default, most ASP.NET controls enable view state. If you want to not maintain the ViewState, include the directive <%@ Page EnableViewState="false" %> at the top of an .aspx page or add the attribute EnableViewState="false" to any control. You can also maintain ViewState for all pages in a web.config file .

```
<configuration>
        <system.web>
                <pages
                buffer="true"
                enableSessionState="true"
                enableViewState="true"
                autoEventWireup="true"
                />
        </system.web>
</configuration>
```

View state is a great idea to store information in a page, but if you wish to save more complex data, and keep them from page to page, you should look into using cookies or sessions is better.

ASP.NET Session State

A session is defined as the period of time that a unique user interacts with a Web application. When a new user begins to interact with the application, a new session ID is generated and associated with all subsequent requests from that same client and stored in a cookie on the client machine.

Session state is maintained in an instance of the HttpSessionState class and is accessible through the Session property of both the Page and HttpContext classes. It is not necessary for you to instantiate the Session Object. An instance is automatically provided for you. The Session object allows you to maintain a list of name/value pairs that can be accessed by any Web page in your application.

vb.net

```
Dim username As String
If Session("userName") IsNot Nothing Then
        username = Session("userName")
Else
        Session("userName") = "John"
```

End If

For every client Session data store separately, means session data is stored as per client basis. Along with advantages, some times session can causes performance issue for heavy traffic sites because its stored on server memory and clients read data from the server itself.

The Session object's collections, properties, methods, and events are described below:

Collections

Collection	Description
Contents	Contains all the items appended to the session through a script command
StaticObjects	Contains all the objects appended to the session with the HTML <object> tag

Properties

Property	Description
CodePage	Specifies the character set that will be used when displaying dynamic content
LCID	Sets or returns an integer that specifies a location or region. Contents like date, time, and currency will be displayed according to that location or region
SessionID	Returns a unique id for each user. The unique id is generated by the server
Timeout	Sets or returns the timeout period (in minutes) for the Session object in this application

Methods

Method	Description
Abandon	Destroys a user session

Contents.Remove	Deletes an item from the Contents collection
Contents.RemoveAll()	Deletes all items from the Contents collection

Events

Event	Description
Session_OnEnd	Occurs when a session ends
Session_OnStart	Occurs when a session starts

Application

Application state is a server side state management technique. The date stored in application state is common for all users of that particular ASP.NET application and can be accessed anywhere in the application. It is also called application level state management. Data stored in the application should be of small size.
How to get and set a value in the **application object**:

```
Application["Count"] = Convert.ToInt32(Application["Count"]) + 1; //Set Value to The Application Object
Label1.Text = Application["Count"].ToString(); //Get Value from the Application Object
```

Application events in ASP.NET

There are three types of events in ASP.NET. Application event is written in a special file called *Global.asax*. This file is not created by default, it is created explicitly by the developer in the root directory. An application can create more than one *Global.asax* file but only the root one is read by ASP.NET.

Application_start: The Application_Start event is raised when an app domain starts. When the first request is raised to an application then the Application_Start event is raised. Let's see the*Global.asax* file.

```
void Application_Start(object sender, EventArgs e)
{
   Application["Count"] = 0;
}
```

Application_Error: It is raised when an unhandled exception occurs, and we can manage the exception in this event.

Application_End: The Application_End event is raised just before an application domain ends because of any reason, may IIS server restarting or making some changes in an application cycle.

Application Object

An application on the Web may consists of several ASP files that work together to perform some purpose. The Application object is used to tie these files together.
The Application object is used to store and access variables from any page, just like the Session object. The difference is that ALL users share ONE Application object (with Sessions there is ONE Session object for EACH user).

The Application object holds information that will be used by many pages in the application (like database connection information). The information can be accessed from any page. The information can also be changed in one place, and the changes will automatically be reflected on all pages.
The Application object's collections, methods, and events are described below:

Collections

Collection	Description
Contents	Contains all the items appended to the application through a script command
StaticObjects	Contains all the objects appended to the application with the HTML <object> tag

Methods

Method	Description
Contents.Remove	Deletes an item from the Contents collection
Contents.RemoveAll()	Deletes all items from the Contents collection
Lock	Prevents other users from modifying the variables in the Application object
Unlock	Enables other users to modify the variables in the Application object (after it has been locked using the Lock method)

Events

Event	Description
Application_OnEnd	Occurs when all user sessions are over, and the application ends
Application_OnStart	Occurs before the first new session is created (when the Application object is first referenced

ASP.NET Objects & Components

ASP.NET Objects

Objects are a way of encapsulating multiple methods (they're like functions) and variables in one easy to manage Uber-Variable (an Object). Objects in ASP resemble other Object Oriented Programming languages. In this lesson we will be using the ASP CDO.Message object as our example object to be dissected.

asp object overview

Objects were created to combat the increasing complexity of programming. The rationale for understanding and using Objects in your programming is to make programming easier and your code more human readable.

asp create an object - server.createobject

An object in ASP is created by passing a *name* string to the *Server.CreateObject* function(actually referred to as a **method**). The string to create a *Message* object is "CDO.Message". We will be creating a CDO.Message object in this example.

Note: Because objects are special there is a special way that you create and destroy them using the *Set* keyword. These areas are marked in red in the example below.

ASP Code:

```
<%
Dim myObject
Set myObject = Server.CreateObject("CDO.Message")

'You must Set your objects to "nothing" to free up the
'the computer memory that was allocated to it
Set myObject = nothing
```

```
%>
```

That wasn't too painful, was it? Let's cover some more bases on the object model.

Objects are a collection of related things that are combined into this blob of programming goo that can be created and destroyed whenever we may need it. For example say that you wanted to make an object that allowed you to send an email...

Well there are certain things all emails have: To, From, CC, Subject, etc. This list of variables that are common to every email would be quite tiresome to have to create for every email we sent. Wouldn't it be nice if we could create some sort of Uber-Variable(Object) that would group all these smaller variables into one thing?

asp object properties

These smaller variables are commonly referred to as an object's properties and the format for setting these properties is nearly identical to setting a variable equal to a value.

The correct syntax for setting an object's properties is:

- objectName.propertyName = someValue

In this tiny example below we are creating a new mail object and setting its *To*and *From* properties.

ASP Code:

```
<%
Dim myObject
Set myObject = Server.CreateObject("CDO.Message")

'Then we set the To and From properties
myObject.To = "little.timmy@example.com"
myObject.From = "huge.jill@example.com"

'You must Set your objects to "nothing" to free up the
'the computer memory that was allocated to it
Set myObject = nothing
%>
```

Now I know we didn't DO anything in the above example, but we still need to learn a bit more about objects before we can get anything done! Objects, besides having a clump of associated common variables, may also have a collection of functions(which become referred to as **methods**) associated with them.

These methods are processes that you would want to commonly do to either manipulate the variables of the object or to use the variables to do something. In our *Message* object we have a collection of information that, when put together into the proper email form and sent to an email service will become an email.

All this complex code has been programmed by Microsoft employees and stored into the *Message* objects *Send* method.

asp object methods

We cannot see the code that was used to program the *Send* method, but that's one of the great things about using object programming. You know what you need to know and nothing more. In our example below we create a Message object and set the necessary properties and send it off with the *Send* method.

ASP Code:

```
<%
Dim myObject
Set myObject = Server.CreateObject("CDO.Message")

'Then we set the To and From properties
myObject.To = "little.timmy@example.com"
myObject.From = "huge.jill@example.com"
myObject.Subject = "Can you see me?"
myObject.TextBody = "I'm really really big!"

myObject.Send()

'You must Set your objects to "nothing" to free up the
'the computer memory that was allocated to it
Set myObject = nothing
%>
```

Note: If you are running this on your home computer there are a slough of issues that may arise with sending an email. Microsoft has a large FAQ about using the

CDO.Message object that may help you. Knowing Microsoft, that link may go dead soon, so Contact Us if it expires!

ASP COMPONENTS

An ASP Server Component is a collection of code that has been made by Microsoft (advanced users can also create their own components), and included in IIS. With the use of ASP you can unlock the power of this pre-made code.

An ASP Server Component is a collection of code that has been made by Microsoft (advanced users can also create their own components), and included in IIS. With the use of ASP you can unlock the power of this pre-made code.

ASP AdRotator Component

The ASP AdRotator component creates an AdRotator object that displays a different image each time a user enters or refreshes a page. A text file includes information about the images.

Note: The AdRotator does not work with Internet Information Server 7 (IIS7).

Syntax

```
<%
set adrotator=server.createobject("MSWC.AdRotator")
adrotator.GetAdvertisement("textfile.txt")
%>
```

ASP AdRotator Example

Assume that we have the following text file, named "ads.txt":

```
REDIRECT banners.asp
*
w3s.gif
https://www.w3schools.com
Free Tutorials from W3Schools
50
xmlspy.gif
https://www.altova.com
XML Editor from Altova
50
```

The lines below the asterisk in the text file above specifies the name of the images (ads) to be displayed, the hyperlink addresses, the alternate text (for the images), and the display rates (in percent).

The first line in the text file above specifies what to happen when a visitor clicks on one of the images. The redirection page (banners.asp) will receive a querystring with the URL to redirect to.

Tip: To specify the height, width, and border of the image, you can insert the following lines under REDIRECT:

```
REDIRECT banners.asp
WIDTH 468
HEIGHT 60
BORDER 0
*
w3s.gif
...
```

The "banners.asp" file looks like this:

Example

```
<%
url=Request.QueryString("url")
If url<>"" then Response.Redirect(url)
%>

<!DOCTYPE html>
<html>
<body>
<%
set adrotator=Server.CreateObject("MSWC.AdRotator")
response.write(adrotator.GetAdvertisement("textfile.txt"))
%>
</body>
</html>
```

using asp components

Making use of Microsoft's ASP Components in your ASP programming will allow you to do so much with ASP that you'll be kicking yourself for not using components earlier.

You can access these built in components by creating objects of them. See our previous lesson if you need a refresher on what ASP Objects are.

In this lesson we will be utilizing Microsoft's FileSystem Component to display all the files in our current directory. The first thing we need to do is to create a FileSystem object so we can use all the properties and methods that are in this component. Note: FSO stands for File System Object in this example.

tizagComponent.asp ASP Code:

```
<%
Dim myFSO
Set myFSO = _
Server.CreateObject("Scripting.FileSystemObject")
Set myFSO = nothing
%>
```

accessing an asp component's features

Once you have created an object of your desired component you can access all the methods and variables of that component. We have created an instance of the File System Component and stored it into *myFSO*. We need the folder that we're going to list all the files of and the *GetFolder* method of our FSO will do the job.

Using the same directory we decided to use back in the Running ASP lesson we are going to set the path of this FSO to "C:\Inetpub\wwwroot\tizagASP\", the same directory that "tizagComponent.asp" was saved in.

Updated tizagComponent.asp ASP Code:

```
<%
Dim myFSO, myFolder
Set myFSO = _
Server.CreateObject("Scripting.FileSystemObject")
Set myFolder = _myFSO.GetFolder("C:\Inetpub\wwwroot\tizagASP")
Response.Write("Current folder is: " & myFolder.Name)
Set myFolder = nothing
Set myFSO = nothing
%>
```

Display:

Current folder is: tizagASP

finishing up

The last thing on our to-do list is to get access to the names of the files in our working directory. The *Folder* object contains a method that returns a collection of all the files in the current directory. The code for accessing this collection and displaying the filenames is in the example below.

Updated tizagComponent.asp ASP Code:

```
<%
Dim myFSO, myFolder
Set myFSO = _
Server.CreateObject("Scripting.FileSystemObject")
Set myFolder = _myFSO.GetFolder("C:\Inetpub\wwwroot\tizagASP")
Response.Write("Current folder is: " & myFolder.Name)

For Each fileItem In myFolder.Files
        Response.Write("<br />" & fileItem.Name)
Next
Set myFolder = nothing
Set myFSO = nothing
%>
```

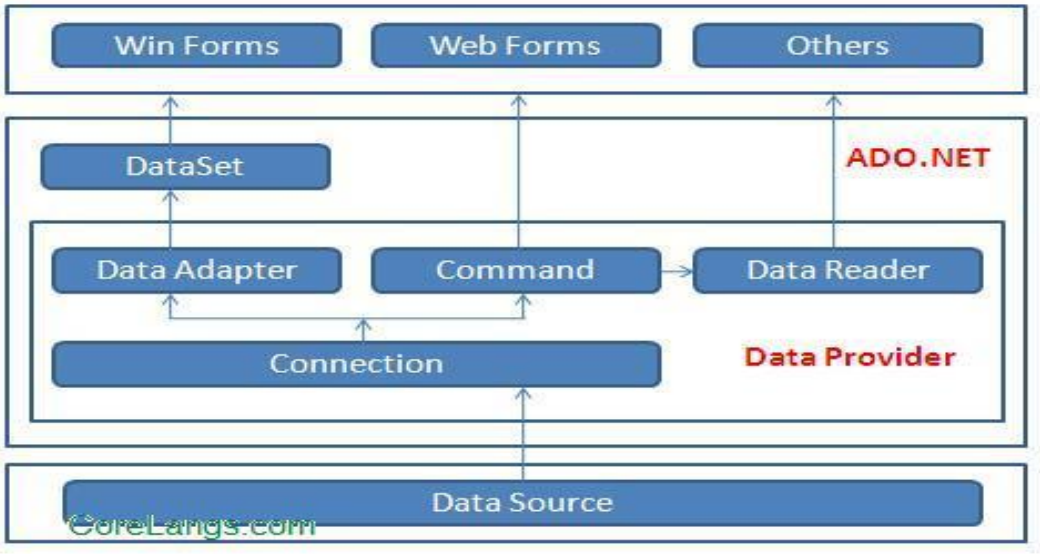
If you have done every example in this ASP Tutorial your web page produced would look like this. Notice that the files are automatically sorted alphabetically!

Display:

firtscript.asp
tizagComponent.asp
tizagEmail.asp
tizagForm.html
tizagGet.asp
tizagPost.asp

ADO.NET Architecture

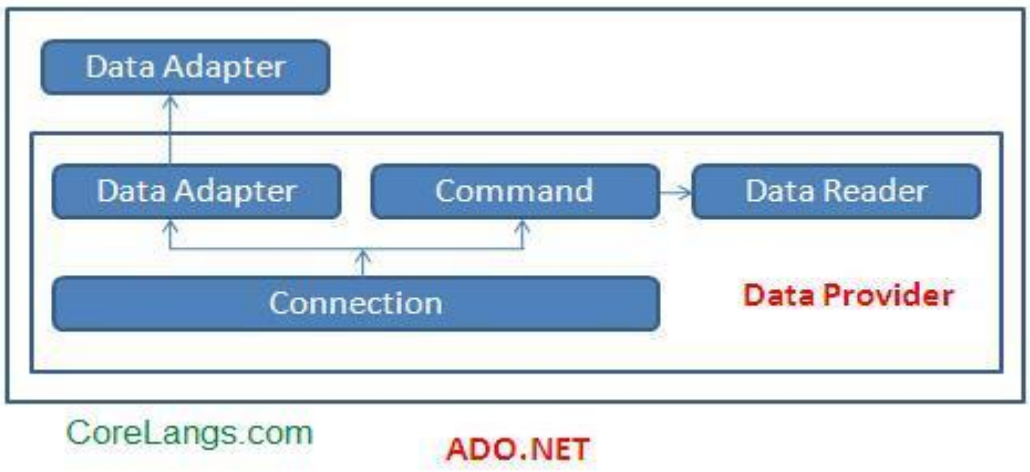
ADO.NET



ADO.NET consist of a set of Objects that expose data access services to the .NET environment. It is a data access technology from Microsoft .Net Framework , which provides communication between relational and non relational systems through a common set of components .

System.Data namespace is the core of ADO.NET and it contains classes used by all data providers. ADO.NET is designed to be easy to use, and Visual Studio provides several wizards and other features that you can use to generate ADO.NET data access code.

Data Providers and DataSet



The two key components of ADO.NET are Data Providers and DataSet . The Data Provider classes are meant to work with different kinds of data sources. They are used to perform all data-management operations on specific databases. DataSet class provides mechanisms for managing data when it is disconnected from the data source.

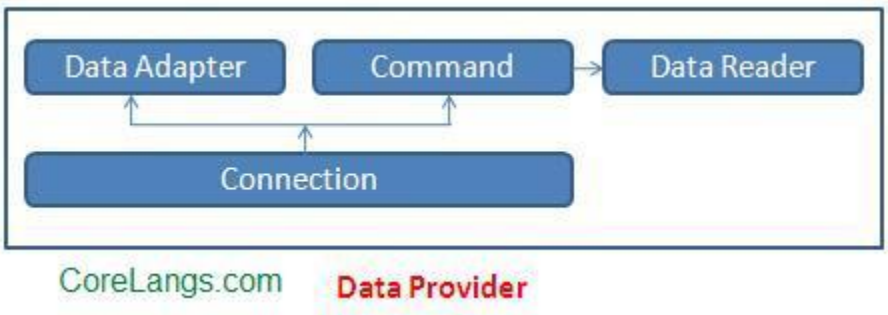
Data Providers

The .Net Framework includes mainly three Data Providers for ADO.NET. They are the Microsoft SQL Server Data Provider , OLEDB Data Provider and ODBC Data Provider . SQL Server uses the SqlConnection object , OLEDB uses the OleDbConnection Object and ODBC uses OdbcConnection Object respectively.

ASP.NET SQL Server Connection

ASP.NET OLEDB Connection

ASP.NET ODBC Connection



A data provider contains Connection, Command, DataAdapter, and DataReader objects. These four objects provides the functionality of Data Providers in the ADO.NET.

Connection

The Connection Object provides physical connection to the Data Source. Connection object needs the necessary information to recognize the data source and to log on to it properly, this information is provided through a connection string.

ASP.NET Connection

Command

The Command Object uses to perform SQL statement or stored procedure to be executed at the Data Source. The command object provides a number of Execute methods that can be used to perform the SQL queries in a variety of fashions.

ASP.NET Command

DataReader

The DataReader Object is a stream-based , forward-only, read-only retrieval of query results from the Data Source, which do not update the data. DataReader requires a live connection with the databse and provides a very intelligent way of consuming all or part of the result set.

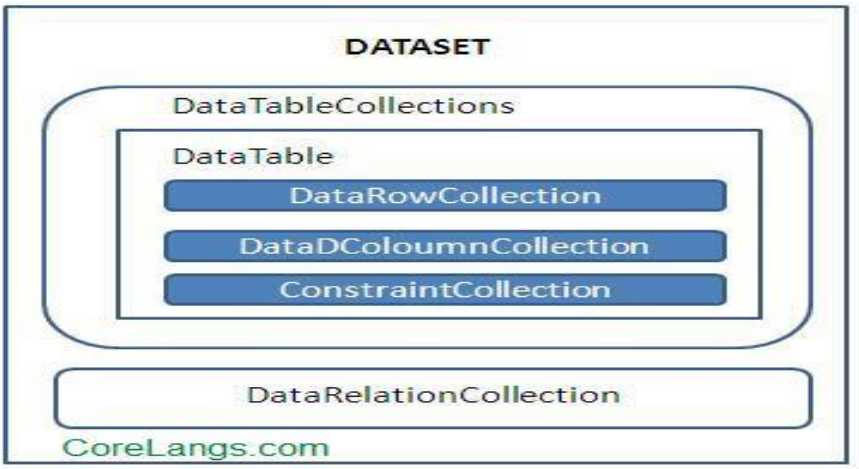
ASP.NET DataReader

DataAdapter

DataAdapter Object populate a Dataset Object with results from a Data Source . It is a special class whose purpose is to bridge the gap between the disconnected Dataset objects and the physical data source.

ASP.NET DataAdapter

DataSet



DataSet provides a disconnected representation of result sets from the Data Source, and it is completely independent from the Data Source. DataSet provides much greater flexibility when dealing with related Result Sets.

DataSet contains rows, columns,primary keys, constraints, and relations with other DataTable objects. It consists of a collection of DataTable objects that you can relate to each other with DataRelation objects. The DataAdapter Object provides a bridge between the DataSet and the Data Source.

ASP.NET Sql Server Connection

The SqlConnection Object is Handling the part of physical communication between the ASP.NET application and the SQL Server Database . An instance of the SqlConnection class in ASP.NET is supported the Data Provider for SQL Server Database.

vb.net

```
Dim connectionString As String

connectionString =
ConfigurationManager.ConnectionStrings("SQLDbConnection").ToString
```

When the connection is established , SQL Commands will execute with the help of the Command Object and retrieve or manipulate the data in the database. Once the Database activities is over , Connection should be closed and release the Data Source resources .

The Close() method in SqlConnection Class is used to close the Database Connection. The Close method rolls back any pending transactions and releases the Connection from the SQL Server Database.

The following ASP.NET program connect to a database server and display the message in the Label control.

web.config

```
 <?xml version="1.0"?>
<configuration>
 <connectionStrings>
        <add name="SQLDbConnection"
                connectionString="Server=servername;  Database=pubs;  User
Id=username; password=password"
                providerName="System.Data.SqlClient" />
 </connectionStrings>
</configuration>
```

You have to fill appropriate web.config database parameters

default.aspx.vb

```
Imports System.Data
Imports System.Data.SqlClient
Imports System.Configuration

Partial Class _Default
        Inherits System.Web.UI.Page

        Protected Sub Button1_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Button1.Click
                Dim connectionString As String
                Dim connection As SqlConnection
                connectionString =
ConfigurationManager.ConnectionStrings("SQLDbConnection").ToString
                connection = New SqlConnection(connectionString)
                connection.Open()
                Label1.Text = "Connected to Database Server !!"
                connection.Close()
        End Sub
End Class
```

POSSIBLE QUESTIONS

PART A – Multiple Choice Questions

1. Which of the following server control shows data in a tabular format?

a)ListBox b)Repeater c)Data Source d)GridView

2. _____ Web Controls are used to achieve graphics in the display.

a)AdRotator b)LinkButton c)TextBox d)Calendar

3.What's the difference between Response.Write() andResponse.Output.Write()?

a)Response.Output.Write() allows you to flush output

b)Response.Output.Write() allows you to buffer output

c)Response.Output.Write() allows you to write formatted output

d)Response.Output.Write() allows you to stream output

4. _____is a program that runs inside IIS

a)ASP b)HTML c)JavaScript d)DHTML

5. ASP.NET applications are configured with special _____ file

a)XML b)HTML c)VB d)NONE

PART B – 2 Mark Questions

1. What is the lifespan for items stored in ViewState?
2. Which is the parent class of the Web server control?
3. What is Query String? What are its advantages and limitations?
4. What is ViewState?
5. Which method is used to force all the validation controls to run?

PART C – 8 Mark Questions

1. Explain about components in Asp.Net with suitable example..
2. Describe about using databases in ASP.NET applications.
3. Explain Application object with example.
4. Describe scripting object model in ASP.NET
5. Explain about ASP server object.

KARPAGAM ACADEMY OF HIGHER EDUCATION

DEPARTMENT OF CS,CA & IT

I M.Sc CS

WEB TECHNOLOGY (17CSP101)

UNIT 3

S.No	Questions	Choice 1	Choice 2	Choice 3	Choice 4	Answer
1	What is the extension of a web user control file ?	.Asmx	.Ascx	.Aspx	.Aspz	. Ascx
2	Select the validation control used for “PatternMatching”	FieldValidator	RegularExpressi	RangeValidator	PatternValidator	RegularExpressionV
3	Which of the following server control shows data in a tabular format?	ListBox	Repeater	Data Source	GridView	GridView
4	Skins with SkindID’s are known as _____	Application Skins	Named Skins	Default Skins	Reference Skins	Named Skins
5	Which of the following webserver control used as container for other server controls in a ASP.Net Page?	PlaceHolder	Table	Panel	IamgeMap	Panel
6	Attribute must be set on a validator control for the validation to work.	ControlToValidat e	ControlToBind	ValidateControl	Validate	ControlToValidate
7	If a developer of ASP.NET defines style information in a common location.Then that location is called as	Master Page	Theme	Customization	Skin	Theme
8	_____ checks the required fields have a value entered by the user submitting the form	RangeValidator	CustomValidator	CompareValidator	RequiredFieldVali dator	RequiredFieldValid ator
9	_____ forms a group of radio buttons that can be selected in a mutually exclusive manner	RadioButton	CheckBoxList	RadioButtonList	DropDownList	RadioButtonList
10	Which of the following does not have any visible interface?	Datagrid	Repeater	DropdownLi	Datalist	Repeater
11	In ASP.NET in form page the object which contains the user name is _____ ?	Page.User.Identit y	Page.User.IsInRo le	Page.User.Name	Page.User.Role	Page.User.Identity
12	Which of the following method must be overridden in a custom control?	The Paint() method	The Control_Build()	The default constructor	The Render() method	The Render() method
13	Which is the file extension used for an ASP.NET file?	.asn	.asp	.aspn	.aspx	.aspx
14	Which property is used to name a web control?	ControlName	Designation	ID	Name	ID

15	Which user action will not generate a server-side event?	Mouse Move	Text Change	Button Click	Load	Mouse Move
16	What class does the ASP.Net Web Form class inherit from by default?	System.Web.Form	System.Web.UI.Form	System.Web.GUI.Page	System.Web.UI.Page	System.Web.UI.Page
17	Which of the following denote the web control associated with Table	DataList	ListBox	TableRow	TableColumn	TableRow
18	ASP.Net separates the HTML output from program logic using a feature	Exception	Code-behind	Code-front	Code-back	Code-behind
19	_____ Web Controls are used to achieve graphics in the display.	AdRotator	LinkButton	TextBox	Calendar	AdRotator
20	The Asp.net server control provides an alternative way of displaying text	<asp:listitem>	<asp:button>	<asp:label>	<asp:image>	<asp:label>
21	The _____attribute specifies where to store session state	Cookieless	Mode	Timeout	ConnectionString	Mode
22	Which attribute specifies the number of minutes before a session is	Cookieless	Mode	Timeout	ConnectionString	Timeout
23	Which attribute specifies the server and port for storing session state	Cookieless	Mode	Timeout	ConnectionString	ConnectionString
24	The current status of a Web Forms page and its controls is called the	viewstate	webstatus	round trips	request/response	**viewstate**
25	ASP.NET applications uses _____ protocol.	WSDL	SOAP	HTTP	TCP/IP	**HTTP**
26	_____ dynamically buids Web-based client server applications.	XML	VB	ASP.NET	None	**ASP.NET**
27	The extension of all ASP.NET files are_____	asp	aspx	asa	asax	**aspx**
28	All controls with the attribute RUNAT="SERVER" are called _____	web controls	server controls	html controls	.net controls	server controls
29	ASP.NET applications are configured with special _____ file	XML	HTML	VB	NONE	XML
30	The configuration setting of the ASP.NET applications are in a file named	ASP.config	Web.config	Session.config	Config.Net	Web.config
31	Which of the following is used to store the state information of the server	Sesson State	Cookies	Hidden fields	Query Strings	Sesson State object
32	Which of the following is used to store the state information of the client	Sesson State	Cookies	Application state	Databases	Cookies
33	DOM stands for	Design Object	Document	Document Object	Database Object	Document Object
34	The term _____ refers to the current condition of an application	section	attribute	state	cookie	state
35	Which object has the state information	Request Object	Response Object	Application Object	Server Object	Application Object
36	The root element of the web.config file is _____	<configuration>	<system.web>	<compilation>	<httpHandlers>	<configuration>
37	Whenever an individual user first requests a page of wep application is created.	Session Object	Response Object	Application Object	Server Object	Session Object
38	Whenever the first page of a web applicatiom is requests for the first time_____ is created.	Session Object	Response Object	Application Object	Server Object	Application Object

39	Cookies are read and written using the _____ object	Request Object	Response Object	Application Object	Server Object	Response Object
40	Codes that execute in response to starting an Application are placed in subroutine.	Applicaion_onStart	Applicaion_Start	Application_BeginRequest	Application-Error	Applicaion_Start
41	This method allows only the current page access the Contents collection	Set()	Contents()	Current()	Lock()	Lock()
42	_____ maintains the state informations across page requests.	session	scope	acess specifiers	form	**session**
43	_____ is created for every user, only when he requests the page for first time	Application Object	Request Object	Response Object	Session Object	**Session Object**
44	The_____ control is used to enforce a value-required rule	Compare	Range Validator	Required Field	Regular	**Required Field**
45	The _____ control is used to make comparisons between two form	Compare	Range Validator	Required Field	Regular	**Compare Validator**
46	A valid comment block in ASP.NET is	<!- - - Comment - - - >	<!- - Comment - - >	<% - - Comment - - %>	<% ! - - Comment - - >	**<% - - Comment - - %>**
47	_____ refers to the current Object	Me	this	super	Current	**Me**
48	Which session method ends the current usr session and destroys the	remove()	removeAll()	Abandon()	AbandonAll()	Abandon()
49	_____ are small strings of text with a name/value pairs that are attached to the end of the URL and sent with page requests.	Hidden fields	Query String	View State property	Cookies	**Query String**

UNIT IV

Creating Mark up with XML: Introduction – Parsers and well formed XML Documents – Parsing an XML Document - Characters – Mark up – CDATA Sections – XML Namespaces. Document Type Definition – Parsers Well formed and valid XML documents – Element type declarations – Attribute declarations- Attributes Types. Schemas – Schemas VS DTD's – W3C XML Schema

Creating Mark up with XML:

Introduction

XML stands for **E**xtensible **M**arkup **L**anguage. It is a text-based markup language derived from Standard Generalized Markup Language (SGML).

XML tags identify the data and are used to store and organize the data, rather than specifying how to display it like HTML tags, which are used to display the data. XML is not going to replace HTML in the near future, but it introduces new possibilities by adopting many successful features of HTML.

There are three important characteristics of XML that make it useful in a variety of systems and solutions −

- **XML is extensible** − XML allows you to create your own self-descriptive tags, or language, that suits your application.
- **XML carries the data, does not present it** − XML allows you to store the data irrespective of how it will be presented.
- **XML is a public standard** − XML was developed by an organization called the World Wide Web Consortium (W3C) and is available as an open standard.

XML Usage

A short list of XML usage says it all −

- XML can work behind the scene to simplify the creation of HTML documents for large web sites.
- XML can be used to exchange the information between organizations and systems.
- XML can be used for offloading and reloading of databases.
- XML can be used to store and arrange the data, which can customize your data handling needs.

- XML can easily be merged with style sheets to create almost any desired output.
- Virtually, any type of data can be expressed as an XML document.

What is Markup?

XML is a markup language that defines set of rules for encoding documents in a format that is both human-readable and machine-readable. So *what exactly is a markup language?* Markup is information added to a document that enhances its meaning in certain ways, in that it identifies the parts and how they relate to each other. More specifically, a markup language is a set of symbols that can be placed in the text of a document to demarcate and label the parts of that document.

Following example shows how XML markup looks, when embedded in a piece of text −

```
<message>
  <text>Hello, world!</text>
</message>
```

This snippet includes the markup symbols, or the tags such as <message>...</message> and <text>... </text>. The tags <message> and </message> mark the start and the end of the XML code fragment. The tags <text> and </text> surround the text Hello, world!.

Is XML a Programming Language?

A programming language consists of grammar rules and its own vocabulary which is used to create computer programs. These programs instruct the computer to perform specific tasks. XML does not qualify to be a programming language as it does not perform any computation or algorithms. It is usually stored in a simple text file and is processed by special software that is capable of interpreting XML.

XML - Syntax

n this chapter, we will discuss the simple syntax rules to write an XML document. Following is a complete XML document −

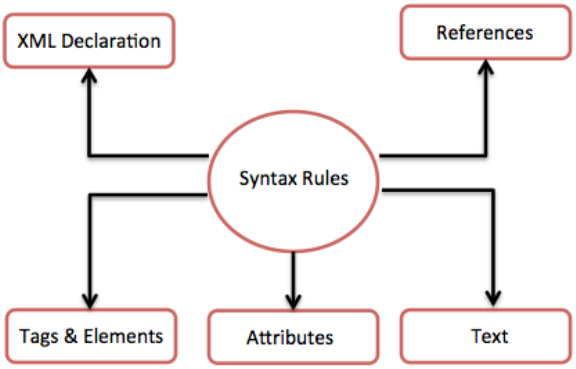
```
<?xml version = "1.0"?>
<contact-info>
  <name>Tanmay Patil</name>
  <company>TutorialsPoint</company>
  <phone>(011) 123-4567</phone>
```

```
</contact-info>
```

You can notice there are two kinds of information in the above example −

- Markup, like <contact-info>
- The text, or the character data, *Tutorials Point* and *(040) 123-4567*.

The following diagram depicts the syntax rules to write different types of markup and text in an XML document.



Let us see each component of the above diagram in detail.

XML Declaration

This chapter covers XML declaration in detail. **XML declaration** contains details that prepare an XML processor to parse the XML document. It is optional, but when used, it must appear in the first line of the XML document.

Syntax

Following syntax shows XML declaration −

```
<?xml
  version = "version_number"
  encoding = "encoding_declaration"
  standalone = "standalone_status"
?>
```

Each parameter consists of a parameter name, an equals sign (=), and parameter value inside a quote. Following table shows the above syntax in detail −

Parameter	Parameter_value	Parameter_description
Version	1.0	Specifies the version of the XML standard used.
Encoding	UTF-8, UTF-16, ISO-10646-UCS-2, ISO-10646-UCS-4, ISO-8859-1 to ISO-8859-9, ISO-2022-JP, Shift_JIS, EUC-JP	It defines the character encoding used in the document. UTF-8 is the default encoding used.
Standalone	yes or no	It informs the parser whether the document relies on the information from an external source, such as external document type definition (DTD), for its content. The default value is set to *no*. Setting it to *yes* tells the processor there are no external declarations required for parsing the document.

Rules

An XML declaration should abide with the following rules −

- If the XML declaration is present in the XML, it must be placed as the first line in the XML document.
- If the XML declaration is included, it must contain version number attribute.
- The Parameter names and values are case-sensitive.
- The names are always in lower case.
- The order of placing the parameters is important. The correct order is: *version, encoding and standalone.*
- Either single or double quotes may be used.
- The XML declaration has no closing tag i.e. **</?xml>**

XML Declaration Examples

Following are few examples of XML declarations −

XML declaration with no parameters −

```
<?xml >
```

XML declaration with version definition −

```
<?xml version = "1.0">
```

XML declaration with all parameters defined −

```
<?xml version = "1.0" encoding = "UTF-8" standalone = "no" ?>
```

XML declaration with all parameters defined in single quotes −

```
<?xml version = '1.0' encoding = 'iso-8859-1' standalone = 'no' ?>
```

The XML document can optionally have an XML declaration. It is written as follows −

```
<?xml version = "1.0" encoding = "UTF-8"?>
```

Where *version* is the XML version and *encoding* specifies the character encoding used in the document.

Syntax Rules for XML Declaration

- The XML declaration is case sensitive and must begin with "**<?xml>**" where "**xml**" is written in lower-case.
- If document contains XML declaration, then it strictly needs to be the first statement of the XML document.
- The XML declaration strictly needs be the first statement in the XML document.
- An HTTP protocol can override the value of *encoding* that you put in the XML declaration.

Parsers and well formed XML Documents

Use our XML validator to syntax-check your XML.

Well Formed XML Documents

An XML document with correct syntax is called "Well Formed".

The syntax rules were described in the previous chapters:

- XML documents must have a root element

- XML elements must have a closing tag
- XML tags are case sensitive
- XML elements must be properly nested
- XML attribute values must be quoted

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

XML Errors Will Stop You

Errors in XML documents will stop your XML applications.

The W3C XML specification states that a program should stop processing an XML document if it finds an error. The reason is that XML software should be small, fast, and compatible.

HTML browsers are allowed to display HTML documents with errors (like missing end tags).

With XML, errors are not allowed.

Valid XML Documents

A "well formed" XML document is not the same as a "valid" XML document.

A "valid" XML document must be well formed. In addition, it must conform to a document type definition.

There are two different document type definitions that can be used with XML:

- DTD - The original Document Type Definition
- XML Schema - An XML-based alternative to DTD

A document type definition defines the rules and the legal elements and attributes for an XML document.

Parsing an XML Document

All major browsers have a built-in XML parser to access and manipulate XML.

XML Parser

The XML DOM (Document Object Model) defines the properties and methods for accessing and editing XML.

However, before an XML document can be accessed, it must be loaded into an XML DOM object.

All modern browsers have a built-in XML parser that can convert text into an XML DOM object.

Parsing a Text String

This example parses a text string into an XML DOM object, and extracts the info from it with JavaScript:

Example

```
<html>
<body>

<p id="demo"></p>

<script>
var text, parser, xmlDoc;

text = "<bookstore><book>" +
"<title>Everyday Italian</title>" +
"<author>Giada De Laurentiis</author>" +
"<year>2005</year>" +
"</book></bookstore>";

parser = new DOMParser();
xmlDoc = parser.parseFromString(text,"text/xml");

document.getElementById("demo").innerHTML =
    xmlDoc.getElementsByTagName("title")[0].childNodes[0].nodeValue;
</script>

</body>

</html>
```

CDATA vs PCDATA

CDATA

CDATA: (Unparsed Character data): CDATA contains the text which is not parsed further in an XML document. Tags inside the CDATA text are not treated as markup and entities will not be expanded.

Let's take an example for CDATA:

```
1. <?xml version="1.0"?>
2. <!DOCTYPE employee SYSTEM "employee.dtd">
3. <employee>
4. <![CDATA[
5.   <firstname>vimal</firstname>
6.   <lastname>jaiswal</lastname>
7.   <email>vimal@javatpoint.com</email>
8. ]]>
9. </employee>
```

In the above CDATA example, CDATA is used just after the element employee to make the data/text unparsed, so it will give the value of employee:

```
<firstname>vimal</firstname><lastname>jaiswal</lastname><email>vimal@javatpoint.com</email>
```

PCDATA

PCDATA: (Parsed Character Data): XML parsers are used to parse all the text in an XML document. PCDATA stands for Parsed Character data. PCDATA is the text that will be parsed by a parser. Tags inside the PCDATA will be treated as markup and entities will be expanded.

In other words you can say that a parsed character data means the XML parser examine the data and ensure that it doesn't content entity if it contains that will be replaced.

Let's take an example:

```
1. <?xml version="1.0"?>
2. <!DOCTYPE employee SYSTEM "employee.dtd">
3. <employee>
4.   <firstname>vimal</firstname>
5.   <lastname>jaiswal</lastname>
6.   <email>vimal@javatpoint.com</email>
```

7. **</employee>**

In the above example, the employee element contains 3 more elements 'firstname', 'lastname', and 'email', so it parses further to get the data/text of firstname, lastname and email to give the value of employee as:

```
vimal jaiswal vimal@javatpoint.com
```

XML Namespaces

XML **Namespace** is used *to avoid element name conflict* in XML document.

XML Namespace Declaration

An XML namespace is declared using the reserved XML attribute. This attribute name must be started with "xmlns".

Let's see the XML namespace syntax:

1. **<element** xmlns:name = "URL"**>**

Here, namespace starts with keyword **"xmlns"**. The word **name** is a namespace prefix. The **URL** is a namespace identifier.

Let's see the example of XML file.

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <cont:contact xmlns:cont="http://sssit.org/contact-us">
3.   <cont:name>Vimal Jaiswal</cont:name>
4.   <cont:company>SSSIT.org</cont:company>
5.   <cont:phone>(0120) 425-6464</cont:phone>
6. </cont:contact>
```

Namespace Prefix: cont

Namespace Identifier: http://sssit.org/contact-us

It specifies that the element name and attribute names with cont prefix belongs to http://sssit.org/contact-us name space.

In XML, elements name are defined by the developer so there is a chance to conflict in name of the elements. To avoid these types of confliction we use XML Namespaces. We can say that XML Namespaces provide a method to avoid element name conflict.

Generally these conflict occurs when we try to mix XML documents from different XML application.

Let's take an example with two tables:

Table1:

```
<table>
  <tr>
    <td>Aries</td>
    <td>Bingo</td>
  </tr>
</table>
```

Table2: This table carries information about a computer table.

```
<table>
  <name>Computer table</name>
  <width>80</width>
  <length>120</length>
</table>
```

If you add these both XML fragments together, there would be a name conflict because both have <table< element. Although they have different name and meaning.

XML Reserved Characters

In XML, some characters are reserved for internal use and you must replace them by entity references when they are used in data.

Entity References

The following table shows the characters that must be replaced by their entity references in all application definition files (ADFs) and instance configuration files (ICFs).

Character	Meaning	Entity reference
>	Greater than	&gt;
<	Less than	&lt;
&	Ampersand	&amp;

%	Percent	&#37;

The Notification Services XML vocabulary reserves the percent sign (%) for denoting parameters.

XML DTD

What is DTD

DTD stands for **Document Type Definition**. It defines the legal building blocks of an XML document. It is used to define document structure with a list of legal elements and attributes.

Purpose of DTD

Its main purpose is to define the structure of an XML document. It contains a list of legal elements and define the structure with the help of them.

Checking Validation

Before proceeding with XML DTD, you must check the validation. An XML document is called "well-formed" if it contains the correct syntax.

A well-formed and valid XML document is one which have been validated against DTD.

Visit **http://www.xmlvalidation.com** to validate the XML file.

Valid and well-formed XML document with DTD

Let's take an example of well-formed and valid XML document. It follows all the rules of DTD.

employee.xml

```
1. <?xml version="1.0"?>
2. <!DOCTYPE employee SYSTEM "employee.dtd">
3. <employee>
4.   <firstname>vimal</firstname>
5.   <lastname>jaiswal</lastname>
6.   <email>vimal@javatpoint.com</email>
7. </employee>
```

In the above example, the DOCTYPE declaration refers to an external DTD file. The content of the file is shown in below paragraph.

employee.dtd

```
1. <!ELEMENT employee (firstname,lastname,email)>
2. <!ELEMENT firstname (#PCDATA)>
3. <!ELEMENT lastname (#PCDATA)>
4. <!ELEMENT email (#PCDATA)>
```

Description of DTD

<!DOCTYPE employee : It defines that the root element of the document is employee.

<!ELEMENT employee: It defines that the employee element contains 3 elements "firstname, lastname and email".

<!ELEMENT firstname: It defines that the firstname element is #PCDATA typed. (parse-able data type).

<!ELEMENT lastname: It defines that the lastname element is #PCDATA typed. (parse-able data type).

<!ELEMENT email: It defines that the email element is #PCDATA typed. (parse-able data type).

XML DTD with entity declaration

A doctype declaration can also define special strings that can be used in the XML file.

An entity has three parts:

1. An ampersand (&)
2. An entity name
3. A semicolon (;)

Syntax to declare entity:

```
1. <!ENTITY entity-name "entity-value">
```

Let's see a code to define the ENTITY in doctype declaration.

author.xml

```
1. <?xml version="1.0" standalone="yes" ?>
2. <!DOCTYPE author [
3.   <!ELEMENT author (#PCDATA)>
4.   <!ENTITY sj "Sonoo Jaiswal">
5. ]>
```

6. **<author>**&sj;**</author>**

In the above example, sj is an entity that is used inside the author element. In such case, it will print the value of sj entity that is "Sonoo Jaiswal".

XML Attributes

XML elements can have attributes. By the use of attributes we can add the information about the element.

XML attributes enhance the properties of the elements.

Note: XML attributes must always be quoted. We can use single or double quote.

Let us take an example of a book publisher. Here, book is the element and publisher is the attribute.

<book publisher="Tata McGraw Hill"**></book>**

Or

<book publisher='Tata McGraw Hill'**></book>**

Metadata should be stored as attribute and data should be stored as element.

1. **<book>**
2. **<book** category="computer"**>**
3. **<author>** A & B **</author>**
4. **</book>**

Data can be stored in attributes or in child elements. But there are some limitations in using attributes, over child elements.

Why should we avoid XML attributes

- Attributes cannot contain multiple values but child elements can have multiple values.
- Attributes cannot contain tree structure but child element can.
- Attributes are not easily expandable. If you want to change in attribute's vales in future, it may be complicated.
- Attributes cannot describe structure but child elements can.
- Attributes are more difficult to be manipulated by program code.

- Attributes values are not easy to test against a DTD, which is used to define the legal elements of an XML document.

Difference between attribute and sub-element

In the context of documents, attributes are part of markup, while sub elements are part of the basic document contents.

In the context of data representation, the difference is unclear and may be confusing.

Same information can be represented in two ways:

1st way:

<book publisher="Tata McGraw Hill"**> </book>**

2nd way:

1. **<book>**
2. **<publisher>** Tata McGraw Hill **</publisher>**
3. **</book>**

In the first example publisher is used as an attribute and in the second example publisher is an element.

Both examples provide the same information but it is good practice to avoid attribute in XML and use elements instead of attributes.

XML Schema

What is XML schema

XML schema is a language which is used for expressing constraint about XML documents. There are so many schema languages which are used now a days for example Relax- NG and XSD (XML schema definition).

An XML schema is used to define the structure of an XML document. It is like DTD but provides more control on XML structure.

Checking Validation

An XML document is called "well-formed" if it contains the correct syntax. A well-formed and valid XML document is one which have been validated against Schema.

Visit **http://www.xmlvalidation.com** to validate the XML file against schema or DTD.

XML Schema Example

Let's create a schema file.

employee.xsd

```
1. <?xml version="1.0"?>
2. <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
3. targetNamespace="http://www.javatpoint.com"
4. xmlns="http://www.javatpoint.com"
5. elementFormDefault="qualified">
6.
7. <xs:element name="employee">
8.   <xs:complexType>
9.     <xs:sequence>
10.      <xs:element name="firstname" type="xs:string"/>
11.      <xs:element name="lastname" type="xs:string"/>
12.      <xs:element name="email" type="xs:string"/>
13.    </xs:sequence>
14.  </xs:complexType>
15. </xs:element>
16.
17. </xs:schema>
```

Let's see the xml file using XML schema or XSD file.

employee.xml

```
1. <?xml version="1.0"?>
2. <employee
3. xmlns="http://www.javatpoint.com"
4. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5. xsi:schemaLocation="http://www.javatpoint.com employee.xsd">
6.
7.   <firstname>vimal</firstname>
8.   <lastname>jaiswal</lastname>
9.   <email>vimal@javatpoint.com</email>
10. </employee>
```

Description of XML Schema

<xs:element name="employee"> : It defines the element name employee.

<xs:complexType> : It defines that the element 'employee' is complex type.

<xs:sequence> : It defines that the complex type is a sequence of elements.

<xs:element name="firstname" type="xs:string"/> : It defines that the element 'firstname' is of string/text type.

<xs:element name="lastname" type="xs:string"/> : It defines that the element 'lastname' is of string/text type.

<xs:element name="email" type="xs:string"/> : It defines that the element 'email' is of string/text type.

XML Schema Data types

There are two types of data types in XML schema.

1. simpleType
2. complexType

simpleType

The simpleType allows you to have text-based elements. It contains less attributes, child elements, and cannot be left empty.

complexType

The complexType allows you to hold multiple attributes and elements. It can contain additional sub elements and can be left empty.

DTD - Elements

XML elements can be defined as building blocks of an XML document. Elements can behave as a container to hold text, elements, attributes, media objects or mix of all.

A DTD element is declared with an ELEMENT declaration. When an XML file is validated by DTD, parser initially checks for the root element and then the child elements are validated.

Syntax

All DTD element declarations have this general form:

```
<!ELEMENT elementname (content)>
```

- *ELEMENT* declaration is used to indicate the parser that you are about to define an element.
- *elementname* is the element name (also called the *generic identifier*) that you are defining.
- *content* defines what content (if any) can go within the element.

Element Content Types

Content of elements declaration in a DTD can be categorized as below:

- Empty content
- Element content
- Mixed content
- Any content

Empty Content

This is a special case of element declaration. This element declaration does not contain any content. These are declared with the keyword **EMPTY**.

Syntax

Following is the syntax for empty element declaration:

```
<!ELEMENT elementname EMPTY >
```

In the above syntax:

- **ELEMENT** is the element declaration of category *EMPTY*
- **elementname** is the name of empty element.

Example

Following is a simple example demonstrating empty element declaration:

```
<?xml version="1.0"?>
<!DOCTYPE hr[
```

```
  <!ELEMENT address EMPTY>
]>
<address />
```

In this example *address* is declared as an empty element. The markup for*address* element would appear as <address />.

Element Content

In element declaration with element content, the content would be allowable elements within parentheses. We can also include more than one element.

Syntax

Following is a syntax of element declaration with element content:

```
<!ELEMENT elementname (child1, child2...)>
```

- **ELEMENT** is the element declaration tag
- **elementname** is the name of the element.
- *child1, child2..* are the elements and each element must have its own definition within the DTD.

Example

Below example demonstrates a simple example for element declaration with element content:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<!DOCTYPE address [
  <!ELEMENT address (name,company,phone)>
  <!ELEMENT name (#PCDATA)>
  <!ELEMENT company (#PCDATA)>
  <!ELEMENT phone (#PCDATA)>
]>
<address>
```

```
  <name>Tanmay Patil</name>
  <company>TutorialsPoint</company>
  <phone>(011) 123-4567</phone>
</address>
```

In the above example, *address* is the parent element and *name*, *company* and*phone_no* are its child elements.

POSSIBLE QUESTIONS

PART A – Multiple Choice Questions

1. XML is a subset of -----
   a)HTML b)SGML c)DHTML d)GGML

2. The XML1.0 specification is only around---- pages.
   a)10 b)20 c)30 d)40

3. The character entities such as---- are used to insert special characters.
   a)&special b)&character c)&quot d)&notation

4. To avoid the rendeering of unknown markup onscreen, you could use the ---- attribute for the<xml> element.
   a)<HREF> b)<IMGSRC> c)<SRC> d)<Source>

5. _____ is used for pointing to a documents contents
   a)Xlink b)Xpointer c)Xinclude d)Xbase

PART B – 2 Mark Questions

1. What is PCDATA in XML?
2. Mention any 3 XML Parsers
3. What is the purpose of the XML DTD?
4. What is the use of xml?
5. Compare SGML-based versus XML-based HTML

PART C – 8 Mark Questions

1. Explain the types of Elements used in xml in detail.
2. Write about XML schemas and XML Namespaces with example.
3. Explain: (i) XML Namespace (ii) XML schema
4. Explain Attribute description in DTD.
5. Explain briefly xml Transformation?

KARPAGAM ACADEMY OF HIGHER EDUCATION

DEPARTMENT OF CS,CA & IT

I M.Sc CS

WEB TECHNOLOGY (17CSP101)

UNIT 4

S.No	Questions	Choice 1	Choice 2	Choice 3	Choice 4	Answer
1	XML is a subset of -----	HTML	SGML	DHTML	GGML	SGML
2	To support efficient web usage, xml doesn't allow the use of many----- constructs that are used to define documents.	XML	Vbscript	SGML	DHTML	SGML
3	Writing----- sounds like a documenting task, requtiring an esotesic knowledge of SGML beypnd the capabilities of most	XML	HTML	Javascript	Vbscript	XML
4	Expand MML	Mean Maskup Lg	Mask Meal Lg	Meal Markup Lg	Mandatory mackup Lg	Meal Markup Lg
5	In, all attribute values must be qouted in--------	Single quotes	double quotes	exclamation	semicolon	double quotes
6	In, XML, all elements with empty contents must be self identifying by ending in---------	>	</	/>	<	/>
7	In XML if you start a new element such as <Burger>, then you must close it as----------	</Burger>	</BURGER>	</burger>	</bURGER>	</Burger>
8	A valid XML file may not contain certain characters that have------ meanings.	constant	reserved	different	same	reserved
9	A document constructed according to the previous simple rules is known as-----	Document type definition	Well formed formula	Well formed documents	Document type definition	Well formed documents
10	Conventional SGML uses the notation of-------- documents	Definition	Valid	XML	Nonvalidate	Valid
11	The ----- supports both well formed and valid documents	SGML	DTD	HTML	XML	XML
12	A document that conforms to a DTD is said to be-----	SGML	Well formed document	Valid	Nonvalidate	Valid
13	One interesting aspect of using DTD with an---- file is that the correctness of the documents can be checked.	XML	SMGL	DHTML	HTML	XML

14	The full SGML standards is about----- pages.	1000	500	1500	2000	500
15	The XML1.0 specification is only around---- pages.	10	20	30	40	30
16	XML removes about----- constructs that SGML uses todefine DTDS	10	20	40	30	30
17	------- following the BODY identifies indicates that the start & end tags are optional & can be omitted.	A	O	B	Vbscript	O
18	BODY is a type of identifies known as an-----	inclusion	exclusion	inclusive	exclusive	exclusion
19	------ doesn't support exclusions.	HTML	DTD	SGML	XML	XML
20	+(INS| DELS) specifies an-----	insertion	deletion	inclusion	exclusive	inclusion
21	(INS|DELS) is also known as a -----, a construct not supported by XML, that is used to indicate that a declaration applies to	name space	namegroup	spacegroup	Entities	namegroup
22	------- supports comments but not inside declarations.	DTD	URL	DOCTYPE	XML	XML
23	XML---- which essentially are macros that associate an identifies with replacement text defined either inside the declaration or in	general entities	comments	documents	name group	general entities
24	The character entities such as---- are used to insert special characters.	&special	&character	&quot	&notation	&quot
25	Microsoft Internet Explorer relies upon an---------- bared lg to support push functionality	HTML	XML	SGML	DHTML	XML
26	-------- lg is used to support the automatic downloading of software	channel definition	XML	Open software	none	Open software description
27	The slogan that micro soft uses to describe the HTML/XML relationship is ----- is for presentation, while----- is for data.	SGML & HTML	HTML & DHTML	XML &Vbdcript	HTML & XML	HTML & XML
28	XML based language documents are identified by a special file extention such as---- that is analogous to the special extension	.cdf	.gdv	.avh	.efh	.cdf
29	----- files are retrived like files for html media inserts, but are processed in a special way.	SGML	DHTML	XML	none	XML
30	The ----- language illustrates the characteristics of the "XML as data" model.	cgf	cdf	gcd	gcf	cdf
31	The ----- element can be used anywhere within an HTML document to enclose XML content	<script>	<html xml>	<xml html>	<xml>	<xml>

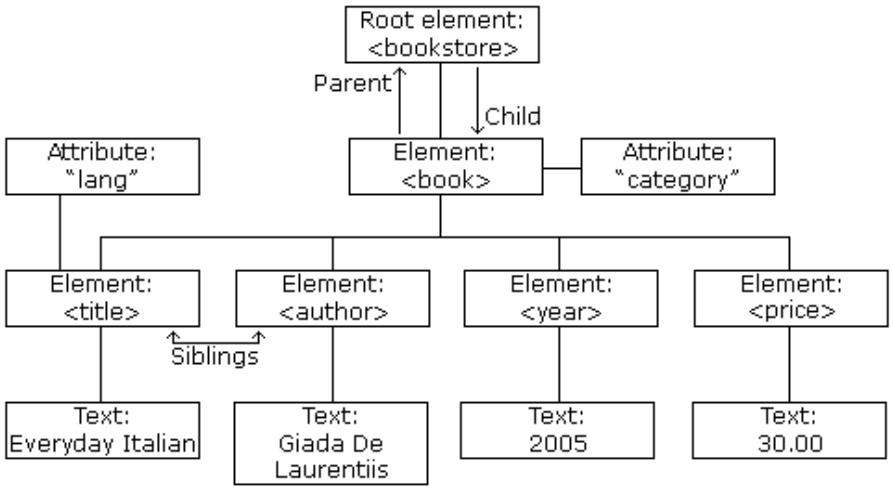
32	To avoid the rendeering of unknown markup onscreen, you could use the ---- atttribute for the<xml> element.	<HREF>	<IMGSRC>	<SRC>	<Source>	<SRC>
33	Placing XML elements inside an HTML documents------ element is consistent with the xml as data model.	<body>	<head>	<title>	<xml>	<head>
34	The most basic way to render xml is to translate it into-----	<DHTML	<SGML>	<Vbscrip>	HTML	HTML
35	The main benefit of translating XML to HTML approach is------ provided	css	Attributes	fragments	abstraction	abstraction
36	By using a new technology called----- you should be able to deliver XML documents right to the screen.	DTD	External sheet	Extenible style sheet	Inline style sheet	Extenible style sheet language
37	You can use ------ programs to translate your XML documents to HTML dicuments at delivery time	Client side program	server script program	server side program	client script program	server side program
38	-------- doesn't provide the rich structuring that XML does	HTML	DHTML	SGML	GGML	HTML
39	Expand DSSSL	Document specification	Document style semantic &	Document semantic	Document Script style sheet Lg	Document style semantic &
40	The lack of flow objects in----- makes properly displaying the XML document very difficult.	DSSSL	XML	CSS	SSS	CSS
41	----- a complex lg than css is based on a style sheet technology called document style semantics and specification Lg	CSS.P	XML	XSL	XSLT	XSL
42	----- is being used with HTML and files to define push channels for interner explorer.	cfd	cdf	xsl	xdl	cdf
43	------- is being used by real platform to deefine presentations	cdf	siml	sgml	smil	smil
44	In late----- the w3c released its first draft of hopw HTML could be rewritten in light of XML.	1998	1999	2000	2001	1998
45	------ form of HTML broke the language into modules and applied all xml rules to HTML elements.	FFML	SMIL	SDF	Code named voyager	Code named voyager
46	In XML the empty elements must include a --------- slash	ending	beginning	trailing	Commence	trailing
47	_____ is a new idea added to XML that enables you to indicate where elements are defined	workspace	namespace	namegroup	attribute	namespace
48	The use of namespace inXML allows more than one author to define an element such as	<head>	<body>	<base>	<title>	<title>
49	The last important rule in XML is that first element within the <head> of the document must be _____ element.	<base>	<title>	<script>	</script>	<title>

50	The standard generalised markup language is _____ language	sub	meta	alias	hierarchical	meta

UNIT-V

Document Object Model: DOM implementations – DOM with JavaScript – Components- Creating nodes – Traversing the DOM. Simple API for XML: DOM vs SAX – SAX based Parsers. XLink XPointer XInclude and XBase

Document Object Model



What is the DOM?

The DOM defines a standard for accessing and manipulating documents:

"The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document."

The HTML DOM defines a standard way for accessing and manipulating HTML documents. It presents an HTML document as a tree-structure.

The XML DOM defines a standard way for accessing and manipulating XML documents. It presents an XML document as a tree-structure.

Understanding the DOM is a must for anyone working with HTML or XML.

The HTML DOM

All HTML elements can be accessed through the HTML DOM.

This example changes the value of an HTML element with id="demo":

Example

```
<h1 id="demo">This is a Heading</h1>

<script>
document.getElementById("demo").innerHTML = "Hello World!";
</script>
```

This example changes the value of the first <h1> element in an HTML document:

Example

```
<h1>This is a Heading</h1>

<h1>This is a Heading</h1>

<script>
document.getElementsByTagName("h1")[0].innerHTML = "Hello World!";
</script>
```

Note: Even if the HTML document contains only ONE <h1> element you still have to specify the array index [0], because the getElementsByTagName() method always returns an array.

You can learn a lot more about the HTML DOM in our JavaScript tutorial.

The XML DOM

All XML elements can be accessed through the XML DOM.

The XML DOM is:

- A standard object model for XML
- A standard programming interface for XML
- Platform- and language-independent
- A W3C standard

In other words: **The XML DOM is a standard for how to get, change, add, or delete XML elements.**

Get the Value of an XML Element

This code retrieves the text value of the first <title> element in an XML document:

Example

```
txt = xmlDoc.getElementsByTagName("title")[0].childNodes[0].nodeValue;
```

Loading an XML File

The XML file used in the examples below is books.xml.

This example reads "books.xml" into xmlDoc and retrieves the text value of the first <title> element in books.xml:

Example

```
<!DOCTYPE html>
<html>
<body>

<p id="demo"></p>

<script>
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
  if (this.readyState == 4 && this.status == 200) {
  myFunction(this);
  }
};
xhttp.open("GET", "books.xml", true);
xhttp.send();

function myFunction(xml) {
  var xmlDoc = xml.responseXML;
  document.getElementById("demo").innerHTML =
  xmlDoc.getElementsByTagName("title")[0].childNodes[0].nodeValue;
}
</script>

</body>
</html>
```

Example Explained

- **xmlDoc** - the XML DOM object created by the parser.
- **getElementsByTagName("title")[0]** - get the first <title> element
- **childNodes[0]** - the first child of the <title> element (the text node)
- **nodeValue** - the value of the node (the text itself)

Loading an XML String

This example loads a text string into an XML DOM object, and extracts the info from it with JavaScript:

Example

```
<html>
<body>

<p id="demo"></p>

<script>
var text, parser, xmlDoc;

text = "<bookstore><book>" +
"<title>Everyday Italian</title>" +
"<author>Giada De Laurentiis</author>" +
"<year>2005</year>" +
"</book></bookstore>";

parser = new DOMParser();
xmlDoc = parser.parseFromString(text,"text/xml");

document.getElementById("demo").innerHTML =
xmlDoc.getElementsByTagName("title")[0].childNodes[0].nodeValue;
</script>

</body>
</html>
```

Programming Interface

The DOM models XML as a set of node objects. The nodes can be accessed with JavaScript or other programming languages. In this tutorial we use JavaScript.

The programming interface to the DOM is defined by a set standard properties and methods.

Properties are often referred to as something that is (i.e. nodename is "book").

Methods are often referred to as something that is done (i.e. delete "book").

XML DOM Properties

These are some typical DOM properties:

- x.nodeName - the name of x
- x.nodeValue - the value of x
- x.parentNode - the parent node of x
- x.childNodes - the child nodes of x
- x.attributes - the attributes nodes of x

Note: In the list above, x is a node object.

XML DOM Methods

- x.getElementsByTagName(*name*) - get all elements with a specified tag name
- x.appendChild(*node*) - insert a child node to x
- x.removeChild(*node*) - remove a child node from x

Note: In the list above, x is a node object.

Traversing the Node Tree

Often you want to loop an XML document, for example: when you want to extract the value of each element.

This is called "Traversing the node tree"

The example below loops through all child nodes of <book>, and displays their names and values:

Example

```
<!DOCTYPE html>
<html>
<body>
```

```
<p id="demo"></p>

<script>
var x, i ,xmlDoc;
var txt = "";
var text = "<book>" +
"<title>Everyday Italian</title>" +
"<author>Giada De Laurentiis</author>" +
"<year>2005</year>" +
"</book>";

parser = new DOMParser();
xmlDoc = parser.parseFromString(text,"text/xml");

// documentElement always represents the root node
x = xmlDoc.documentElement.childNodes;
for (i = 0; i < x.length ;i++) {
   txt += x[i].nodeName + ": " + x[i].childNodes[0].nodeValue+ "<br>";
}
document.getElementById("demo").innerHTML = txt;
</script>

</body>
</html>
```

Output:

title: Everyday Italian
author: Giada De Laurentiis
year: 2005

Example explained:

1. Load the XML string into xmlDoc
2. Get the child nodes of the root element
3. For each child node, output the node name and the node value of the text node

Java SAX Parser - Overview

SAX (Simple API for XML) is an event-based parser for XML documents. Unlike a DOM parser, a SAX parser creates no parse tree. SAX is a streaming interface for XML, which means that applications using SAX receive event notifications about the XML document being processed an element, and attribute, at a time in sequential order starting at the top of the document, and ending with the closing of the ROOT element.

- Reads an XML document from top to bottom, recognizing the tokens that make up a well-formed XML document.
- Tokens are processed in the same order that they appear in the document.
- Reports the application program the nature of tokens that the parser has encountered as they occur.
- The application program provides an "event" handler that must be registered with the parser.
- As the tokens are identified, callback methods in the handler are invoked with the relevant information.

When to Use?

You should use a SAX parser when −

- You can process the XML document in a linear fashion from top to down.
- The document is not deeply nested.
- You are processing a very large XML document whose DOM tree would consume too much memory. Typical DOM implementations use ten bytes of memory to represent one byte of XML.
- The problem to be solved involves only a part of the XML document.
- Data is available as soon as it is seen by the parser, so SAX works well for an XML document that arrives over a stream.

Disadvantages of SAX

- We have no random access to an XML document since it is processed in a forward-only manner.
- If you need to keep track of data that the parser has seen or change the order of items, you must write the code and store the data on your own

SAX XML Parser in Java

SAX Stands for Simple API for XML Parsing. This is an event based XML Parsing and it parse XML file step by step so much suitable for large XML Files. SAX XML Parser fires an event when it encountered opening tag, element or attribute, and the parsing works accordingly. It's recommended to use SAX XML parser for parsing large XML files in Java because it doesn't require to load whole XML file in Java and it can read a big XML file in small parts. Java provides support for SAX parser and you can parse any XML file in Java using SAX Parser, I have covered an example of reading XML file using SAX Parser here. One disadvantage of using SAX Parser in java is that reading XML file in Java using SAX Parser requires more code in comparison of DOM Parser.

Difference between DOM and SAX XML Parser

Here are few high-level **differences between DOM parser and SAX Parser** in Java:

1) DOM parser loads whole XML document in memory while SAX only loads a small part of the XML file in memory.

2) DOM parser is faster than SAX because it access whole XML document in memory.

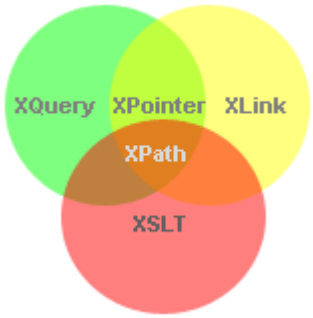
3) SAX parser in Java is better suitable for large XML file than DOM Parser because it doesn't require much memory.

4) DOM parser works on Document Object Model while SAX is an event based XML parser.

XML, XLink and XPointer

XLink

XLink is used to create hyperlinks in XML documents.



- XLink is used to create hyperlinks within XML documents
- Any element in an XML document can behave as a link
- With XLink, the links can be defined outside the linked files
- XLink is a W3C Recommendation

XLink Browser Support

There is no browser support for XLink in XML documents.

However, all major browsers support XLinks in SVG.

XLink Syntax

In HTML, the <a> element defines a hyperlink. However, this is not how it works in XML. In XML documents, you can use whatever element names you want - therefore it is impossible for browsers to predict what link elements will be called in XML documents.

Below is a simple example of how to use XLink to create links in an XML document:

```
<?xml version="1.0" encoding="UTF-8"?>

<homepages xmlns:xlink="http://www.w3.org/1999/xlink">
 <homepage xlink:type="simple"xlink:href="https://www.w3schools.com">Visit
W3Schools</homepage>
 <homepage xlink:type="simple"xlink:href="http://www.w3.org">Visit
W3C</homepage>
</homepages>
```

To get access to the XLink features we must declare the XLink namespace. The XLink namespace is: "http://www.w3.org/1999/xlink".

The xlink:type and the xlink:href attributes in the <homepage> elements come from the XLink namespace.

The xlink:type="simple" creates a simple "HTML-like" link (means "click here to go there").

The xlink:href attribute specifies the URL to link to.

XLink Example

The following XML document contains XLink features:

```
<?xml version="1.0" encoding="UTF-8"?>

<bookstore xmlns:xlink="http://www.w3.org/1999/xlink">

<book title="Harry Potter">
 <description
 xlink:type="simple"
 xlink:href="/images/HPotter.gif"
 xlink:show="new">
 As his fifth year at Hogwarts School of Witchcraft and
 Wizardry approaches, 15-year-old Harry Potter is.....
 </description>
</book>

<book title="XQuery Kick Start">
 <description
 xlink:type="simple"
 xlink:href="/images/XQuery.gif"
 xlink:show="new">
 XQuery Kick Start delivers a concise introduction
 to the XQuery standard.....
 </description>
</book>

</bookstore>
```

Example explained:

- The XLink namespace is declared at the top of the document (xmlns:xlink="http://www.w3.org/1999/xlink")

- The xlink:type="simple" creates a simple "HTML-like" link
- The xlink:href attribute specifies the URL to link to (in this case - an image)
- The xlink:show="new" specifies that the link should open in a new window

XLink - Going Further

In the example above we have demonstrated simple XLinks. XLink is getting more interesting when accessing remote locations as resources, instead of standalone pages.

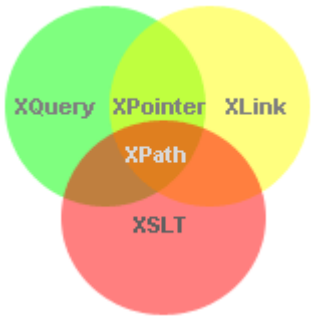
If we set the value of the xlink:show attribute to "embed", the linked resource should be processed inline within the page. When you consider that this could be another XML document you could, for example, build a hierarchy of XML documents.

You can also specify WHEN the resource should appear, with the xlink:actuate attribute.

XLink Attribute Reference

Attribute	Value	Description
xlink:actuate	onLoad onRequest other none	Defines when the linked resource is read and shown: • onLoad - the resource should be loaded and shown when the document loads • onRequest - the resource is not read or shown before the link is clicked
xlink:href	*URL*	Specifies the URL to link to
xlink:show	embed new replace other none	Specifies where to open the link. Default is "replace"
xlink:type	simple extended locator arc resource title none	Specifies the type of link

XPointer



- XPointer allows links to point to specific parts of an XML document
- XPointer uses XPath expressions to navigate in the XML document
- XPointer is a W3C Recommendation

XPointer Browser Support

There is no browser support for XPointer. But XPointer is used in other XML languages.

XPointer Example

In this example, we will use XPointer in conjunction with XLink to point to a specific part of another document.

We will start by looking at the target XML document (the document we are linking to):

```
<?xml version="1.0" encoding="UTF-8"?>

<dogbreeds>

<dog breed="Rottweiler" id="Rottweiler">
 <picture url="https://dog.com/rottweiler.gif" />
 <history>The Rottweiler's ancestors were probably Roman
 drover dogs.....</history>
 <temperament>Confident, bold, alert and imposing, the Rottweiler
 is a popular choice for its ability to protect....</temperament>
</dog>

<dog breed="FCRetriever" id="FCRetriever">
 <picture url="https://dog.com/fcretriever.gif" />
 <history>One of the earliest uses of retrieving dogs was to
 help fishermen retrieve fish from the water....</history>
 <temperament>The flat-coated retriever is a sweet, exuberant,
 lively dog that loves to play and retrieve....</temperament>
</dog>
```

```
</dogbreeds>
```

Note that the XML document above uses id attributes on each element!

So, instead of linking to the entire document (as with XLink), XPointer allows you to link to specific parts of the document. To link to a specific part of a page, add a number sign (#) and an XPointer expression after the URL in the xlink:href attribute, like this: xlink:href="https://dog.com/dogbreeds.xml#xpointer(id('Rottweiler'))". The expression refers to the element in the target document, with the id value of "Rottweiler".

XPointer also allows a shorthand method for linking to an element with an id. You can use the value of the id directly, like this: xlink:href="https://dog.com/dogbreeds.xml#Rottweiler".

The following XML document contains links to more information of the dog breed for each of my dogs:

```
<?xml version="1.0" encoding="UTF-8"?>

<mydogs xmlns:xlink="http://www.w3.org/1999/xlink">

<mydog>
 <description>
 Anton is my favorite dog. He has won a lot of.....
 </description>
 <fact xlink:type="simple"xlink:href="https://dog.com/dogbreeds.xml#Rottweiler">
 Fact about Rottweiler
 </fact>
</mydog>

<mydog>
 <description>
 Pluto is the sweetest dog on earth.....
 </description>
 <fact xlink:type="simple"xlink:href="https://dog.com/dogbreeds.xml#FCRetriever">
 Fact about flat-coated Retriever
 </fact>
</mydog>

</mydogs>
```

Xinclude

Combining XML documents

To enhance reuse and modularity, a technique for constructing new XML documents from existing ones is desirable.

XInclude provides a simple inclusion mechanism.

Why yet another specification?
● many XML documents and languages can benefit from modularity
● as for the namespace solution, a generic approach can be implemented in generic tools

Application conformance: Think of XML as if Namespaces, XInclude, and XML Base were parts of the basic XML specification. (Caveat: these extensions are quite new and not widely implemented yet.)

Example

A document containing:

```
<foo xmlns:xi="http://www.w3.org/2001/XInclude">
  <xi:include href="somewhere.xml"/>
</foo>
```

where **somewhere.xml** contains:

```
<bar>...</bar>
```

is equivalent to:

```
<foo xmlns:xi="http://www.w3.org/2001/XInclude">
  <bar>...</bar>
</foo>
```

- **http://www.w3.org/2001/XInclude** is the official **XInclude namespace**
- the **include** element name in that namespace is an **inclusion directive**
- right after parsing and before other processing, an **XInclude processor** performs the inclusion (tree substitution)
- the original and the resulting document should be considered **equivalent**
- it is an error to have cyclic includes

XML Base

A URI identifies a resource:

- **http://somewhere/somefile.xml** is an **absolute** URI
- **somefile.xml** is a **relative** URI

Inspired by the **<base href="...">** mechanism in HTML, **XML Base** provides a **uniform way of resolving relative URIs**.

In the following example:

```
<... xml:base="http://www.daimi.au.dk/">
  <... href="~mis/mn/index.html" .../>
</...>
```

the value of **href** attribute can be interpreted as the absolute URI **http://www.daimi.au.dk/~mis/mn/index.html**.

- the **xml** namespace prefix is hardwired by the Namespace specification
- **xml:base** has **lexical scope** (as namespace declarations)
- the URI used to access the document is used as **default** URI base

Examples of applications:

- XLink (requires XML Base support)
- XHTML (will use XML Base)
- Namespaces (does not conform to XML Base, but it ought to...)
- your future XML language

Future XML parsers will support Namespaces, XInclude, and XML Base.

POSSIBLE QUESTIONS

PART A – Multiple Choice Questions

1. XML is a subset of -----
   a)HTML b)SGML c)DHTML d)GGML

2. The XML1.0 specification is only around---- pages.
   a)10 b)20 c)30 d)40

3. The character entities such as---- are used to insert special characters.
   a)&special b)&character c)&quot d)&notation

4. To avoid the rendeering of unknown markup onscreen, you could use the ---- attribute for the<xml> element.
   a)<HREF> b)<IMGSRC> c)<SRC> d)<Source>

5. _____ is used for pointing to a documents contents
   a)Xlink b)Xpointer c)Xinclude d)Xbase

PART B – 2 Mark Questions

1. What is XPath?
2. What are the Element Naming Rules used in XML?
3. What is DOM? What are the different levels of DOM?
4. What are important elements in DOM Tree
5. Define xLink

PART C – 8 Mark Questions

1. Describe the functions and features of XBase.
2. Explain DOM and SAX in detail.
3. Discuss SAX Parser.
4. Explain the functions and features of XPointer
5. Differentiate DOM and SAX and mention the components of DOM.
6. Discuss XLink with example

KARPAGAM ACADEMY OF HIGHER EDUCATION

DEPARTMENT OF CS,CA & IT

I M.Sc CS

WEB TECHNOLOGY (17CSP101)

UNIT V

S.No	Questions	Choice 1	Choice 2	Choice 3	Choice 4	Answer
1	___ is used for describing links between resources	Xlink	Xpointer	Xinclude	Xbase	Xlink
2	_____ is used for pointing to a documents contents	Xlink	Xpointer	Xinclude	Xbase	Xpointer
3	_____ is used for including existing XML documents or portions of XML documents into another XML document	Xinclude	Xbase	Xlink	Xpointer	Xinclude
4	_____ is used for specifying a base URL for relative URL's	Xlink	Xpointer	Xinclude	Xbase	Xbase
5	_____ attribute for linking to documents on the web	a	href	src	Xpointer	href
6	Xlink elements that specifying linking information are called _____	src	linking elements	outbound	remote resource	linking elements
7	The linking element that references document is called a ____	remote resource	local resource	outbound	Xpointer	local resource
8	The remote referenced is called the _____	linking elements	outbound	local resource	remote resource	remote resource
9	In Xlink terminology the markup that specifies how to traverse between resources is called an _____	node	leaf	arc	href	arc
10	_____ is used for replacing the current element with the linked resource	show	new	replace	embed	embed
11	_____ is used for replacing the current resource with the linking resource	replace	embed	new	show	replace

12	Xlink also provides _____ links for linking multiple combinations of local and remote resources	undirectional link	extended links	ultidirectional li	book links	extended links
13	_____ is used to reference fragments of an XML document via URI	Xlink	Xpointer	Xinclude	Xbase	Xpointer
14	By using _____ with Xlink we can link to specific part of a resource instead of linking to the entire resource	Xlink	Xpointer	Xinclude	Xbase	Xpointer
15	In object-oriented languages such as Java and C++ methods and data are encapsulated in reusable components called	include element	classes	pointer	replace	classes
16	Xinclude provides a framework for reusing existing _____ documents	DOM	XML	C	HTML	XML
17	To add one XML document inside another as _____ is used	classes	include element	pointer	replace	include element
18	The optional Xinclude attribute _____ can have value XML or text	parse	text	href	a	parse
19	_____ allows a document author to change the base URI for relative links in a document	Xinclude	Xbase	Xlink	Xpointer	Xbase
20	Links between remote resources and local resources are known as ____ links	outbound	inbound	text	href	inbound
21	Xlink attribute _____ contains a human readable descriptrion of a link	title	head	link	panel	title
22	An XML element that has Xinclude attributes is called an _____ element	include	locations	pointer	extended	include
23	A _____ Xpointer address simplifies expresions that use id's	pointer	Base name	locator	panel	Base name
24	the XML ____ language references a resource or a fragment of a resource	Base name	panel	pointer	locator	pointer
25	Xlink attribute types ="_____" defines an element for addressing a remote resource	panel	locator	nclude element	Xinclude	locator
26	An _____ link can contain any number of links between local and/or remote resources	locations	pointer	extended	included	extended

27	_____ are available for a variety of programming languages	SAX based parsers	DOM	locator	panel	SAX based parsers
28	_____ is a tree based model that stores the documents data in a hierarchy of nodes	DOM	XML	C	HTML	DOM
29	_____ was developed by the members of the XML Dev mailing list and was released in May of 1998	DOM	C	HTML	SAX	SAX
30	SAX is an caronym for ____	Simple API for XML	Standard API for XML	Single API for XML	Static API for XML	Simple API for XML
31	The JAXP package that provides all the interfaces for the SAX parser is	org.xml.sax	sax.org.xml	org.xml	sax.xml	org.xml.sax
32	Method _____ is called when a elemnt is encountered	include	locations	pointer	method	method
33	___ is a static node constant that represents an elemnt	node.element_node	getElementsByTagname	Document.write	getLength	node.element_node
34	_____ represents the root of an XML document	Document	DOM	Xlink	Xpointer	Document
35	Method _____ is called when the end of the document is encountered	Document	end document	Xpointer	Xlink	end document
36	Method _____ replaces one child node with another	replacechild	elementnode	getLength	staticnode	replacechild
37	Method _____ creates a text node	elementnode	replacechild	create-textnode	staticnode	create-textnode
38	DOM is an acronym for _____	Document Object Model	Document Object Method	Data Object Method	Distributed Object Method	Document Object Model
39	Class _____ is a sun microsystems internal API class	Document	DOM	XML document	SAX based parsers	XML document
40	A node list contains list of _____	child	nodes	element	leaf	nodes
41	Method _____ loads and parses an XML document	parse	text	href	a	parse
42	An element node value is _____	text	null	number	Attr	null

43	Interface _____ defines methods fatalerror, error and warning	Errorhandler	Document Object Model	node.element _node	method	Errorhandler
44	_____ represents an attribute node, derives from node	Attr interface	a	href	src	Attr interface
45	The key words such as normal, bold, bolder and lighter are used to define the-----	size	font weight	color	style	font weight
46	A page can be divided into two divisions using a _____ tag	DIV	HALF	VISIT	UNVISIT	DIV
47	The mechanism that applies a style across one or more web pages is termed	Custom Style sheets	Cascading Style Sheets	Class Style Sheets	Calculating Style Sheets	Cascading Style Sheets
48	An html document can be viewed as a container having several objects like paragraph, images and this method of designing is called-----	DHTML	Style sheet	Document object model	Document type difinete	Document object model
49	The----- is a list of statements which declares the styles of various components such as heading, paragraphs etc.	Scripting sheets	multimedia	cascading sheet	style sheet	style sheet
50	---- is used to define font name, size, style, text indent, margins, background, text color and alignment.	page	font	bgcolor	styles	styles
51	The point size of the current fonts measurement is abbrevated as	pt	em	px	mt	em
52	To indicate the end of the script the _____ tag is used.	<end>	<end script>	</script>	<?script>	</script>
53	The navigator object is _____ object	invisible	visible	current	hidden	invisible
54	The history object is _____ object	invisible	visible	current	hidden	invisible
55	Plug-ins is developed by _____	Microsoft	IBM	Netscape	Sun Microsystems	Netscape
56	which object is important than all	history	location	document	navigation	document
57	When we use a particular type of style in the <h1> tag we mention the type using the----attribute.	<href>	<sql>	<type>	<class>	<class>
58	----- has a default style to present the web page as per the mark-ups given in the html document.	style sheet	scripts	browsers	Script	browsers

59	When a negative value is assigned ti font size it will----- the normal font size.	increase	add	reduce	commence	reduce