

Q1 A) Explain the Domain name System with example.

5

Ans:

Marks distribution: 3 marks explanation & 2 marks example.

The **Domain Name System (DNS)** is a hierarchical naming system built on a distributed database for computers, services, or any resource connected to the Internet or a private network. It associates various information with domain names assigned to each of the participating entities. Most importantly, it translates domain names meaningful to humans into the numerical identifiers associated with networking equipment for the purpose of locating and addressing these devices worldwide. The Domain Name System makes it possible to assign domain names to groups of Internet resources and users in a meaningful way, independent of each entity's physical location. Because of this, World Wide Web (WWW) hyperlinks and Internet contact information can remain consistent and constant even if the current Internet routing arrangements change or the participant uses a mobile device. Internet domain names are easier to remember than IP addresses such as 208.77.188.166 (IPv4) or 2001:db8:1f70::999:de8:7648:6e8 (IPv6). Users take advantage of this when they recite meaningful Uniform Resource Locators (URLs) and e-mail addresses without having to know how the computer actually locates them. The Domain Name System also specifies the technical functionality of this database service. It defines the DNS protocol, a detailed definition of the data structures and communication exchanges used in DNS, as part of the Internet Protocol Suite

A DNS Example

Let's say that you type the URL `www.howstuffworks.com` into your browser. The browser contacts a DNS server to get the IP address. A DNS server would start its search for an IP address by contacting one of the root DNS servers. The root servers know the IP addresses for all of the DNS servers that handle the top-level domains (.COM, .NET, .ORG, etc.). Your DNS server would ask the root for `www.howstuffworks.com`, and the root would say, "I don't know the IP address for `www.howstuffworks.com`, but here's the IP address for the .COM DNS server."

Your name server then sends a query to the .COM DNS server asking it if it knows the IP address for `www.howstuffworks.com`. The DNS server for the COM domain knows the IP addresses for the name servers handling the `www.howstuffworks.com` domain, so it returns those.

Internet Programming

B) Write a style sheet such that 5

- 1) The web page will have the background image "img1.jpg".
- 2) The table headings will have red background colour.
- 3) All the headings on the page will be aligned to left.
- 4) The hyperlink on the page will not have underline.
- 5) Paragraphs in the web page will have left and right margins of 100px

Ans: Marks distribution: Each carries 1 mark

- 1) body {background-image:url('paper.gif');}
- 2) #customers th { font-size:1.1em;text-align:left;padding-top:5px;background-color:red;}
- 3) .left {margin-left:auto;margin-right:auto;width:70%;background-color:#b0e0e6;}
- 4) a:link {text-decoration:none;}
- 5) p.ex1 { margin-top:100px;margin-bottom:100px;margin-right:100px;margin-left:100px;}

C) Write a code in javaScript to set a Cookie. Assume suitable data for it. 5

Ans: Marks distribution: any correct program 5 marks

```
<html>
<head>
<script type="text/javascript">
function getCookie(c_name)
{
if (document.cookie.length>0)
{
c_start=document.cookie.indexOf(c_name + "=");
if (c_start!=-1)
{
c_start=c_start + c_name.length+1;
c_end=document.cookie.indexOf(";",c_start);
if (c_end==-1) c_end=document.cookie.length;
return unescape(document.cookie.substring(c_start,c_end));
}
}
}
return "";
}
```

Internet Programming

```
function setCookie(c_name,value,expiredays)
{
var exdate=new Date();
exdate.setDate(exdate.getDate()+expiredays);
document.cookie=c_name+ "=" +escape(value)+
((expiredays==null) ? "" : ";expires="+exdate.toUTCString());
}

function checkCookie()
{
username=getCookie('username');
if (username!=null && username!="")
{
alert('Welcome again '+username+!');
}
else
{
username=prompt('Please enter your name:', "");
if (username!=null && username!="")
{
setCookie('username',username,365);
}
}
}
</script>
</head>

<body onload="checkCookie()">
</body>
</html>
```

D) **Explain the difference between ASP and JSP.** 5

Ans: Marks distribution: any five differences each carry 1 mark.

1. JSP and ASP are both server side scripting languages
2. JSP is from Sun Microsystems while ASP is from Microsoft
3. ASP costs money while JSP is free.
4. ASP code is interpreted while JSP code is compiled at run time
5. JSP code can run faster than ASP if there are fewer changes
6. Majority of Windows users use ASP while users of open source operating systems like Linux use JSP among others.

Internet Programming

Q2 A) *State the properties and methods available on ASP Session Object. Write a code in ASP so that the server terminates the session automatically after 30 minutes if it remains idle .How much is the default timeout.*
10

Ans: Marks distribution: properties and methods carries 5 marks, any correct program carries 4 marks & default timeout 1 mark.

Session Object

When you are working with an application on your computer, you open it, do some changes and then you close it. This is much like a Session. The computer knows who you are. It knows when you open the application and when you close it. However, on the internet there is one problem: the web server does not know who you are and what you do, because the HTTP address doesn't maintain state.

ASP solves this problem by creating a unique cookie for each user. The cookie is sent to the user's computer and it contains information that identifies the user. This interface is called the Session object.

The Session object stores information about, or change settings for a user session.

Variables stored in a Session object hold information about one single user, and are available to all pages in one application. Common information stored in session variables are name, id, and preferences. The server creates a new Session object for each new user, and destroys the Session object when the session expires.

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

Internet Programming

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

Method	Description
Abandon	Destroys a user session
Contents.Remove	Deletes an item from the Contents collection

Code:

```
<html>
<body>

<%
response.write("<p>")
response.write("Default Timeout is: " & Session.Timeout & " minutes.")
response.write("</p>")

Session.Timeout=30

response.write("<p>")
response.write("Timeout is now: " & Session.Timeout & " minutes.")
response.write("</p>")
%>

</body>
</html>
```

The default timeout is 20 minutes.

B) Explain the differences between HTML and XML.

5

Ans: Marks distribution: any 5 differences carries 5 marks.

Internet Programming

- 1 .HTML is presentation language where as XML is not either a programming language or a presentation language. It is used to transfer data between applications and databases.
2. HTML is not case-sensitive where as XML is case-sensitive.
3. In XML we can define our own tags as it is not possible in HTML.
4. In XML it is mandatory to close each and every tag where as in HTML it is not required.
5. XML describes the data where as HTML only defines the data.

C) **What do you understand by a web service? Explain with example.** 5

Ans: Marks distribution: explanation 4 marks & example 1 mark

A web service is typically an application programming interface (API) or Web API that is accessed via Hypertext Transfer Protocol (HTTP) and executed on a remote system, hosting the requested service. Web services tend to fall into one of two camps: big web services and RESTful web services.

The W3C defines a "web service" as "a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically Web Services Description Language WSDL). Other systems interact with the web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards."

The W3C also states, "We can identify two major classes of Web services, REST-compliant Web services, in which the primary purpose of the service is to manipulate XML representations of Web resources using a uniform set of "stateless" operations; and arbitrary Web services, in which the service may expose an arbitrary set of operations."

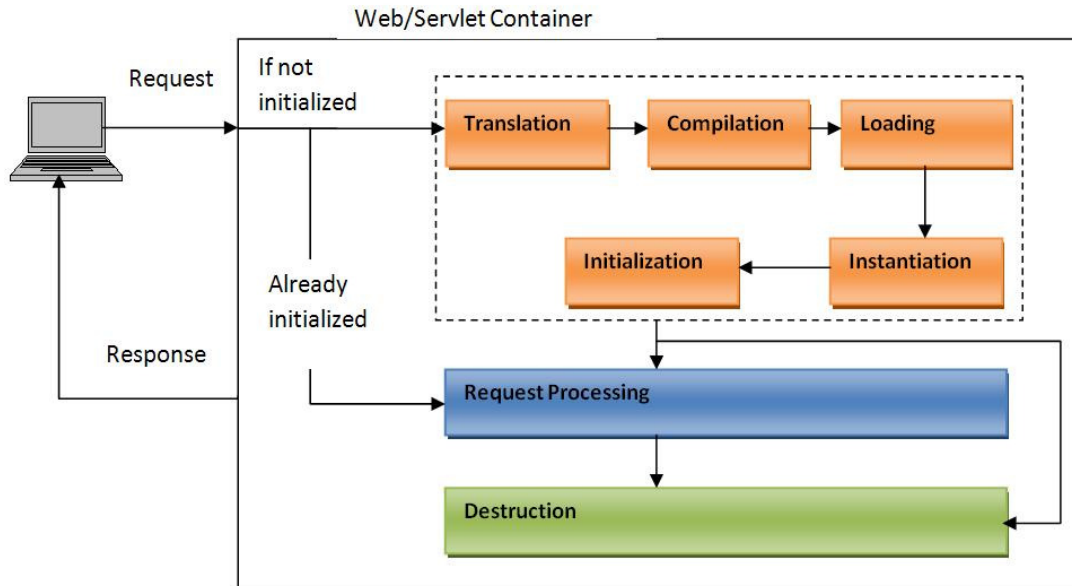
Q 3 A) **Explain how JSP works with neat diagram.** 10

Ans: Marks distribution: explanation 7 marks & 3marks neat diagram

Following diagram shows the different life cycle stages of jsp. Broadly, these stages can be classified into three.

- Instantiation
- Request Processing
- Destruction

Internet Programming

**Instantiation:**

When a web container receives a jsp request (may be first or subsequent), it checks for the jsp's servlet instance. If no servlet instance is available or if it is older than the jsp, then, the web container creates the servlet instance using following stages.

- Translation
- Compilation
- Loading
- Instantiation
- Initialization

Translation:

Web container translates (converts) the jsp code into a servlet code. This means that jsp is actually a servlet. After this stage, there is no jsp, everything is a servlet. This task will create a complete jsp page, by considering all included components. Here on, the static content and dynamic contents are treated differently. The resultant is a java class instead of an html page (which we wrote).

Compilation:

The generated servlet is compiled to validate the syntax. As it is a java class, the compilation is done using javac command. This will generate the byte code to be run on JVM.

Loading:

The compiled byte code is loaded by the class loader used by web container. This is a standard process of using any java class.

Internet Programming

Instantiation:

In this step, instance of the servlet class is created so that it can serve the request.

Initialization:

Initialization is done by calling the `jspInit()` method. This is one time activity at the start of the initialization process. Initialization will make the `ServletContext` and `ServletConfig` objects available. One can access many attributes related to the web container and the servlet itself. After initialization the servlet is ready to process requests.

Request Processing:

Entire initialization process is done to make the servlet available in order to process the incoming request. `jspService()` is the method that actually processes the request. It prints the response in html (any other) format, using 'out' object.

Destroy:

Whenever the server is shutting down or when the server needs memory, the server removes the instance of the servlet. The destroy method `jspDestroy()` can be called by the server after initialization and before or after request processing. Once destroyed the jsp needs to be initialized again.

Just to summarize, web container handles incoming requests to a jsp by converting it into a servlet and then by using this servlet to generate the response. Also when the server shuts down, the container needs to clear the instances.

B) What do you understand by a mark-up language ? Write HTML statements for the following 10

- 1. Create a checkbox**
- 2. Refresh the web page automatically after every 2 minutes**
- 3. Insert an image in the background**
- 4. Enumerate list of five items .Numbering should be in capital roman letters which starts from VI**

Ans: Marks distribution: explanation 2 marks each example carries 2 marks

A markup language is a modern system for annotating a text in a way that is syntactically distinguishable from that text. The idea and terminology evolved from the "marking up" of manuscripts, i.e. the revision instructions by editors, traditionally written with a blue pencil on authors' manuscripts. Examples are typesetting instructions such as those found in troff and LaTeX, and structural markers such as XML tags. Markup is typically omitted from the version of the text which is displayed for end-user consumption. Some markup languages, like HTML have presentation semantics, meaning their specification prescribes how the structured data is to be presented, but other markup languages, like XML, have no predefined semantics.

A) Checkbox:

```
<html>
```

Internet Programming

```
<body>

<form action="">

<input type="checkbox" name="vehicle" value="Bike" /> I have a bike<br />

<input type="checkbox" name="vehicle" value="Car" /> I have a car

</form>

</body>

</html>
```

B) Refresh:

```
<html>
<title>My Refresh Page</title>
<head>

<meta http-equiv='refresh' content='120'; >
</head>
<body>
<h1>Refreshing page automatically after 2 minutes</h1>
</body>
</html>
```

C) Image background

```
<html>

<body style="background-image: url('/images/pulpit.jpg'); border="0" src="/images/pulpit.jpg" alt="Pulpit rock"
width="304" height="228" />

<h2>Norwegian Mountain Trip</h2>

</body>

</html>
```

Internet Programming

D) Enumerate List

```
<html>
<body>
<h4>Roman numbers list:</h4>
<ol type="I" start="VI">
<li>Apples</li>
<li>Bananas</li>
<li>Lemons</li>
<li>Oranges</li>
</ol>
</body>
</html>
```

Q 4 A) **Explain the sequence of steps required to access a database from an ASP page with example. 10**

Ans:

Marks distribution: basic explanation 3 marks & 7 step carries 7 marks.

Accessing database from an ASP page

Databases are a way of organizing and keeping your data. The data stored in databases can be anything from user email addresses to binary files. Databases have become so popular in the past decade that it is almost unimaginable to not to use them on the web. In this tutorial I will guide through the creation of a simple Microsoft Access database to incorporating it in to your ASP web pages. Creating and making use of a database on the web is so very much easy that it will be only after reading this article you will realize the same and will then hopefully start creating databases according to your own needs and then playing with them from the web pages.

Requirements

You are required to have Microsoft Access database (any version will do the trick, 97, 98 2000), MDAC 2.0 or above (latest is MDAC 2.5), either PWS 4 or IIS 4.0, Windows platform and a web browser. Don't worry if you don't know about MDAC (Microsoft Data Access Components), you can check if you have already got them by going to Start -> Settings -> Control Panel. There you will find a small icon 'ODBC 32'. If you can see the icon then you have got MDAC

Internet Programming

but if you cannot find the icon then you will most probably have to download them from Microsoft's site. If you are running Windows2000 Professional then you can find the "Data Sources ODBC" icon in the "Administrative Tools" section of the control panel. For displaying our database contents on the web browser we will be using Microsoft's Active Server Pages technology. For that either PWS 4 or IIS 4 (or above) will be required. Both of them are free and can be downloaded from www.microsoft.com.

Step 1 : Start Microsoft Access by clicking 'Microsoft Access' icon in the Program Files menu. Start -> Program Files -> Microsoft Access.

Step 2 : Microsoft Access will start with default windows opening up at start up. Click 'cancel' to exit any windows that appear.

Step 3 : Click the File -> New button at the top left main windows of Access. This will bring up a 'New' dialog box window. Of the two tabs click the 'General' tab and then the 'Database' icon. This will select 'Database', then hit the 'OK' button.

Step 4 : This will bring up 'File New Database' dialog box. It will ask for the database name and location to store that database to. Type 'odbc_exmp' in the 'File Name' input box and give it any location to store that database to. For this tutorial we will assume that our database 'odbc_exmp.mdb' was saved at c:/stardeveloper/db/odbc_exmp.mdb . Then hit the 'Create' button.

Step 5 : Our database 'odbc_exmp.mdb' is now created. But it is empty and we will need to populate it a bit so that we can later use it. In the Microsoft Access, you will now be seeing a 'odbc_exmp : Database' dialog box showing quite a lot of options on the left column and three options in the right column. Double click the 'Create table in Design view' option in the right column.

Step 6 : This will bring up 'Table : Table' dialog box. Just in case if you don't know, data is stored in tables in a database. There can be many tables within one database. Tables in turn consist of Fields (columns) and rows (records). Fields (columns) do not accept data of all type. We have to specify the data type that a Field (column) will hold and then we can add records for that data type in the rows. It is this 'Design View' in Microsoft Access that is used to specify the number of columns our table will have and what data type that Fields (columns) will hold. Ok now type the 'Field Names' and 'Data Types' exactly as shown below. Note that you can select 'AutoNumber' and 'Text' from the drop down options in the 'Data Type' column as required. There is no need to edit any values in the 'General' and 'Lookup' tabs in the 'Field Properties' section of the 'Table Design View'. Now click the File -> Save button. A 'Save As' dialog box will prompt you to enter the name for this table, type 'names' in that dialog box and hit 'OK'.

Internet Programming

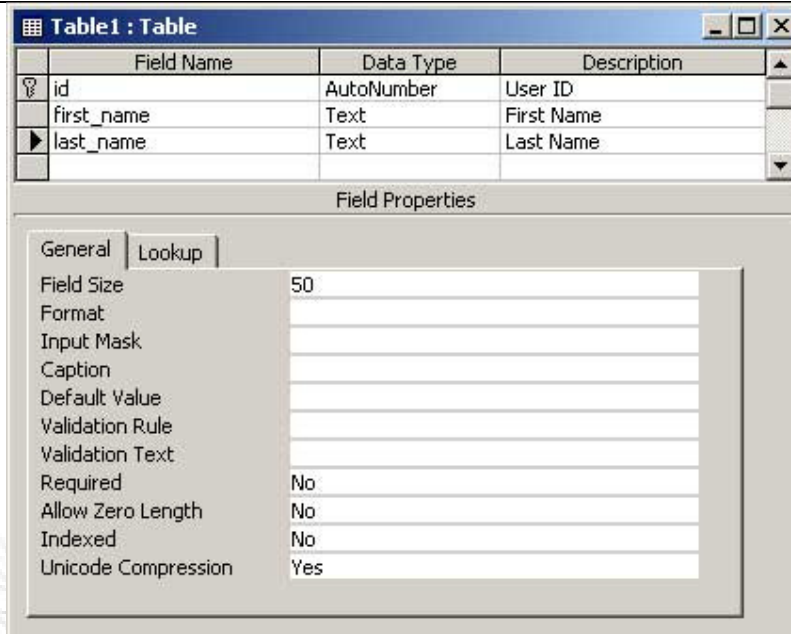


Table1 Table

Step 7 : Close the 'names' table design view window. Now you will see 'names' table being added to the right column of the 'odbc_exmp : Database' window. Double click the 'names' table. This will bring up the 'names : Table' window showing an empty row and three columns with 'Field Names' which we specified earlier. It is used to add data to the table. We will add five names to our 'names' table. There is no need to add anything to the 'id' Field as it will autoincrement one number upon the addition of records to the rows one by one. If you don't understand what I mean by autoincrementing then just leave this field for a moment and you will come to know what it does later when we add records. Ok now add five names (first, last) in the empty row under their respective Field Names as shown below.



'names' table

See the numbers in the 'id' Field. That's what autoincrement does. It adds the numbers in a sequential way. Now hit the 'save' button to save the records which we have added in our 'odbc_exmp.mdb' database. This completes our task of creating a simple Access database.

You have just seen that how easy it is to create a database. You have also learned what are tables, rows and columns. You have also learned what 'Data Types' are and how to specify a 'Data Type' in the table column. You have also added records to the database. Now we will

Internet Programming

move forward and will register our database in the System registry by assigning it a Data Source Name (DSN). Well done, now continue to the next page.

DSN stands for Data Source Name. Data source can be a database, spreadsheet, text file etc. We assign DSN to a data source so that irrespective of the data source details and location, we can use that data source; add, modify or delete records, just by knowing it's DSN.

B) **Explain the Document Object Model with an example.** **10**

Ans: Marks distribution: diagram 3 marks & explanation 7 marks.

The DOM is a programming API for documents. It is based on an object structure that closely resembles the structure of the documents it models. For instance, consider this table, taken from an XHTML document:

The name "Document Object Model" was chosen because it is an "object model" in the traditional object oriented design sense: documents are modeled using objects, and the model encompasses not only the structure of a document, but also the behavior of a document and the objects of which it is composed. In other words, the nodes in the above diagram do not represent a data structure, they represent objects, which have functions and identity. As an object model, the DOM identifies:

- the interfaces and objects used to represent and manipulate a document
- the semantics of these interfaces and objects - including both behavior and attributes
- the relationships and collaborations among these interfaces and objects

The structure of SGML documents has traditionally been represented by an abstract data model, not by an object model. In an abstract data model, the model is centered around the data. In object oriented programming languages, the data itself is encapsulated in objects that hide the data, protecting it from direct external manipulation. The functions associated with these objects determine how the objects may be manipulated, and they are part of the object model.

Where the Document Object Model came from

The DOM originated as a specification to allow JavaScript scripts and Java programs to be portable among Web browsers. "Dynamic HTML" was the immediate ancestor of the Document Object Model, and it was originally thought of largely in terms of browsers. However, when the DOM Working Group was formed at W3C, it was also joined by vendors in other domains, including HTML or XML editors and document repositories. Several of these vendors had worked with SGML before XML was developed; as a result, the DOM has been influenced by SGML Groves and the HyTime standard. Some of these vendors had also developed their own object models for documents in order to provide an API for SGML/XML editors or document repositories, and these object models have also influenced the DOM.

Entities and the DOM Core

In the fundamental DOM interfaces, there are no objects representing entities. Numeric character references, and references to the pre-defined entities in HTML and XML, are replaced by the single character that makes up the entity's replacement. For example, in:

Internet Programming

<p>This is a dog & a cat</p>

the "&" will be replaced by the character "&", and the text in the P element will form a single continuous sequence of characters. Since numeric character references and pre-defined entities are not recognized as such in CDATA sections, or in the SCRIPT and STYLE elements in HTML, they are not replaced by the single character they appear to refer to. If the example above were enclosed in a CDATA section, the "&" would not be replaced by "&"; neither would the <p> be recognized as a start tag. The representation of general entities, both internal and external, are defined within the extended (XML) interfaces of Document Object Model Core.

Note: When a DOM representation of a document is serialized as XML or HTML text, applications will need to check each character in text data to see if it needs to be escaped using a numeric or pre-defined entity. Failing to do so could result in invalid HTML or XML. Also, implementations should be aware of the fact that serialization into a character encoding ("charset") that does not fully cover ISO 10646 may fail if there are characters in markup or CDATA sections that are not present in the encoding.

DOM Architecture

The DOM specifications provide a set of APIs that forms the DOM API. Each DOM specification defines one or more modules and each module is associated with one feature name. For example, the DOM Core specification (this specification) defines two modules:

- The Core module, which contains the fundamental interfaces that must be implemented by all DOM conformant implementations, is associated with the feature name "Core";
- The XML module, which contains the interfaces that must be implemented by all conformant XML 1.0 [XML 1.0] (and higher) DOM implementations, is associated with the feature name "XML".

The following representation contains all DOM modules, represented using their feature names, defined along the DOM specifications:

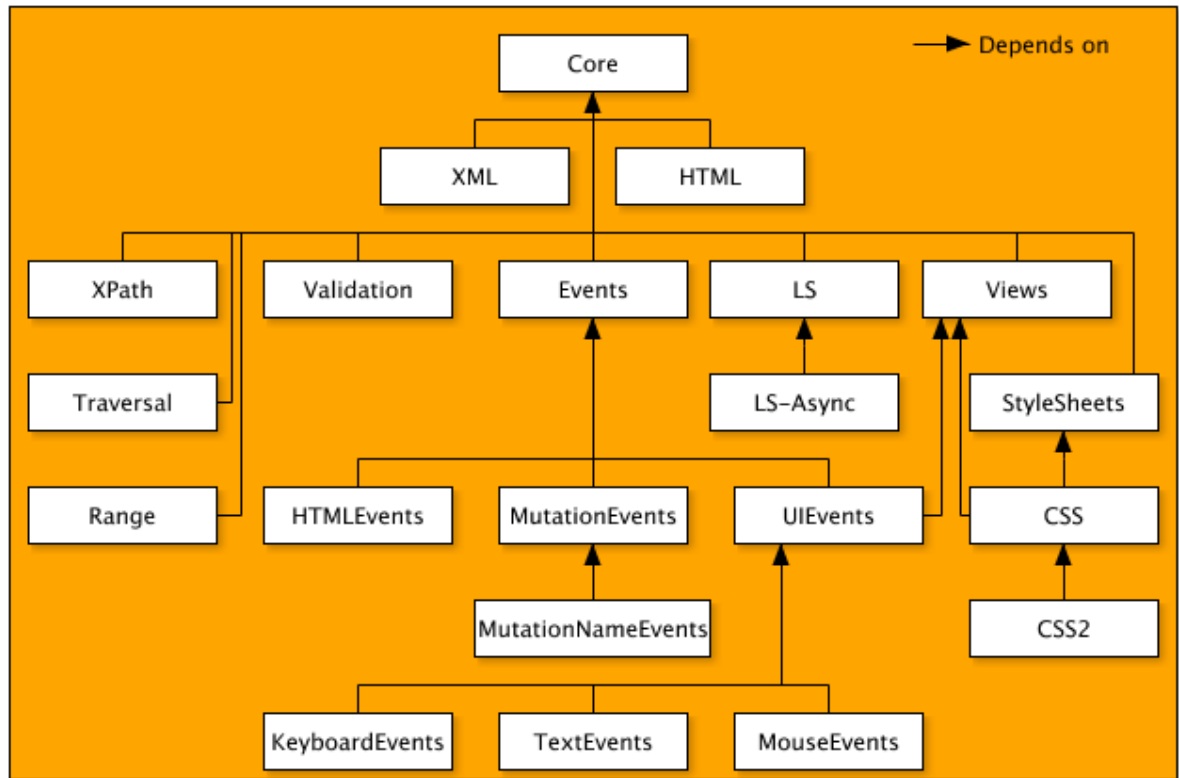


Figure: A view of the DOM Architecture

A DOM implementation can then implement one (i.e. only the Core module) or more modules depending on the host application. A Web user agent is very likely to implement the "MouseEvents" module, while a server-side application will have no use of this module and will probably not implement it.

Conformance

This section explains the different levels of conformance to DOM Level 3. DOM Level 3 consists of 16 modules. It is possible to conform to DOM Level 3, or to a DOM Level 3 module.

An implementation is DOM Level 3 conformant if it supports the Core module defined in this document (see Fundamental Interfaces: Core Module). An implementation conforms to a DOM Level 3 module if it supports all the interfaces for that module and the associated semantics.

Q5 A) *design a web page to maintain a CD catalogue.I should maintain the name of the music album, song in that album ,composer, singer and year of release.Format it in the tabular manner using XSL.*
10

Ans: Marks distribution: any correct program carry 10 marks.

Internet Programming

<CATALOG>

-

<CD>

<TITLE>Empire Burlesque</TITLE>

<ARTIST>Bob Dylan</ARTIST>

<COUNTRY>USA</COUNTRY>

<COMPANY>Columbia</COMPANY>

<PRICE>10.90</PRICE>

<YEAR>1985</YEAR>

</CD>

-

<CD>

<TITLE>Hide your heart</TITLE>

<ARTIST>Bonnie Tyler</ARTIST>

<COUNTRY>UK</COUNTRY>

<COMPANY>CBS Records</COMPANY>

<PRICE>9.90</PRICE>

<YEAR>1988</YEAR>

</CD>

-

<CD>

<TITLE>Red</TITLE>

<ARTIST>The Communards</ARTIST>

Internet Programming

<COUNTRY>UK</COUNTRY>

<COMPANY>London</COMPANY>

<PRICE>7.80</PRICE>

<YEAR>1987</YEAR>

</CD>

-

<CD>

<TITLE>Unchain my heart</TITLE>

<ARTIST>Joe Cocker</ARTIST>

<COUNTRY>USA</COUNTRY>

<COMPANY>EMI</COMPANY>

<PRICE>8.20</PRICE>

<YEAR>1987</YEAR>

</CD>

</CATALOG>

<html>

<body>

<script type="text/javascript">

if (window.XMLHttpRequest)

{// code for IE7+, Firefox, Chrome, Opera, Safari

xmlhttp=new XMLHttpRequest();

}

else

Internet Programming

```
{// code for IE6, IE5

xmlhttp=new XMLHttpRequest("Microsoft.XMLHTTP");

}

xmlhttp.open("GET","cd_catalog.xml",false);

xmlhttp.send();

xmlDoc=xmlhttp.responseXML;

document.write("<table border='1'>");
var x=xmlDoc.getElementsByTagName("CD");
for (i=0;i<x.length;i++)
{
document.write("<tr><td>");
document.write(x[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue);
document.write("</td><td>");
document.write(x[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue);
document.write("</td></tr>");
}
document.write("</table>");

</script>

</body>

</html>
```

- B) **Write a javaScript code for the following** **10**
- A) **To display a digital clock on the screen.**
- B) **To change the advertisement stored in the form of an image after every 5 minutes automatically.**

Internet Programming

Ans : Marks distribution: any correct program for A carries 5 marks & B carries 5 marks

A) Digital clock

```
<html>

<head>

<script type="text/javascript">

function startTime()
{
var today=new Date();
var h=today.getHours();
var m=today.getMinutes();
var s=today.getSeconds();
// add a zero in front of numbers<10
m=checkTime(m);
s=checkTime(s);
document.getElementById('txt').innerHTML=h+":"+m+": "+s;
t=setTimeout('startTime()',500);
}

function checkTime(i)
{
if (i<10)
{
i="0" + i;
}
}
```

```
return i;
}
</script>
</head>
```

```
<body onload="startTime()">
<div id="txt"></div>
</body>
</html>
```

B) To change Advertisement

```
<html>
</body>
<div id="slide_show">
<br />
Default Text
</div>
<div id="user_controls">
<form action="#">
<input type="button" name="goback" id="goback" class="controls" value="&lt;- " />
<input type="button" name="pause_slide" id="pause_slide" class="controls" value="|| " />
<input type="button" name="goforward" id="goforward" class="controls" value="- &gt;" />
<input type="button" name="restart" id="restart" class="controls" value="Restart" />
</form>
</div>
```

Internet Programming

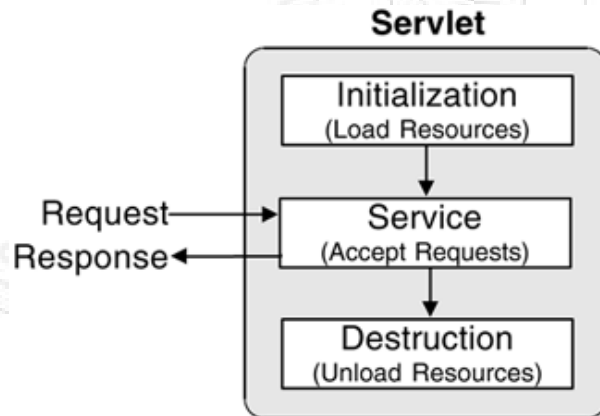
```
<script type="text/javascript" src="simage4.js"></script>
```

```
</body>
```

```
</html>
```

Q 6 A) Explain the Servlet life cycle in detail. 10

Ans: Marks distribution: 3 marks neat diagram + 7 marks explanation



Servlet Life Cycle:

The interface `javax.servlet.Servlet` defines the following three methods known as servlet life cycle methods.

```

public void init(ServletConfig config) throws ServletException
public void service(ServletRequest req, ServletResponse res) throws ServletException,
java.io.IOException
public void destroy()
  
```

- Creation and initialization**
The container first creates the servlet instance and then executes the `init()` method. `init()` can be called only once in its life cycle by the following ways:
 a) Through the 'load-on-startup' tag using the `web.xml`. This makes the servlet to be loaded and initialized when the server starts.
 b) For the first time only in its life cycle, just before the `service()` is invoked.
 c) Server administrator can request for the initialization of a servlet directly.
- Execution of service**
Whenever a client requests for the servlet, everytime the `service()` method is invoked during its life cycle. From `service()` then it is branched to the `doGet()` or `doXx..()` methods for a `HttpServlet`. The `service()` method should contain the code that serves the Servlet purpose.
- Destroy the servlet**
`destroy()` method is invoked first, then Servlet is removed from the container and then eventually garbage collected. `destroy()` method generally contains code to free any resources like `jdbc` connection that will not be garbage collected.

B) **Explain Response Object of ASP with its properties and methods.** 5

Ans: Marks distribution: explanation 1 mark + properties 2 marks + methods 2 marks

Response Object

The ASP Response object is used to send output to the user from the server. Its properties and methods are described below:

Properties

Buffer

Specifies whether to buffer the page output or not

CacheControl

Sets whether a proxy server can cache the output generated by ASP or not

Charset

Appends the name of a character-set to the content-type header in the Response object

ContentType

Sets the HTTP content type for the Response object

Expires

Sets how long (in minutes) a page will be cached on a browser before it expires

ExpiresAbsolute

Sets a date and time when a page cached on a browser will expire

IsClientConnected

Indicates if the client has disconnected from the server

Pics

Appends a value to the PICS label response header

Methods

AddHeader

Adds a new HTTP header and a value to the HTTP response

AppendToLog

Adds a string to the end of the server log entry

BinaryWrite

Writes data directly to the output without any character conversion

Clear

Clears any buffered HTML output

End

Stops processing a script, and returns the current result

Flush

Sends buffered HTML output immediately

Redirect

Redirects the user to a different URL

Write

Writes a specified string to the output

C) ***Explain the in-built objects provided in javascript with their properties and methods.*** 10

Ans: Marks distribution: each object with properties & methods carries 2 marks

String Object

The String object is used to manipulate a stored piece of text. String objects are created with new String().

String Object Properties

constructor

Returns the function that created the String object's prototype

length

Returns the length of a string

prototype

Allows you to add properties and methods to an object

String Object Methods

charAt()

Returns the character at the specified index

charCodeAt()

Returns the Unicode of the character at the specified index

concat()

Joins two or more strings, and returns a copy of the joined strings

fromCharCode()

Converts Unicode values to characters

Date Object

The Date object is used to work with dates and times.

Date Object Properties

constructor

Returns the function that created the Date object's prototype

prototype

Allows you to add properties and methods to an object

Date Object Methods

getDate()

Returns the day of the month (from 1-31)

getDay()

Returns the day of the week (from 0-6)

getFullYear()

Returns the year (four digits)

getHours()

Returns the hour (from 0-23)

Math Object

The Math object allows you to perform mathematical tasks.

Math Object Properties

E

Returns Euler's number (approx. 2.718)

LN2

Returns the natural logarithm of 2 (approx. 0.693)

LN10

Returns the natural logarithm of 10 (approx. 2.302)

Math Object Methods

abs(x)

Returns the absolute value of x

acos(x)

Returns the arccosine of x, in radians

asin(x)

Returns the arcsine of x, in radians

atan(x)

Returns the arctangent of x as a numeric value between $-\pi/2$ and $\pi/2$ radians

atan2(y,x)

Returns the arctangent of the quotient of its arguments

Array Object

The Array object is used to store multiple values in a single variable.

Array Object Properties

constructor

Returns the function that created the Array object's prototype

length

Sets or returns the number of elements in an array

prototype

Allows you to add properties and methods to an object

Array Object Methods

concat()

Joins two or more arrays, and returns a copy of the joined arrays

join()

Joins all elements of an array into a string

pop()

Removes the last element of an array, and returns that element

push()

Adds new elements to the end of an array, and returns the new length

Boolean Object

The Boolean object is used to convert a non-Boolean value to a Boolean value (true or false).

Boolean Object Properties

constructor

Returns the function that created the Boolean object's prototype

prototype

Allows you to add properties and methods to an object

Boolean Object Methods

toString()

Converts a Boolean value to a string, and returns the result

valueOf()

Returns the primitive value of a Boolean object

Global object

The JavaScript global properties and functions can be used with all the built-in JavaScript objects.

JavaScript Global Properties

Infinity

A numeric value that represents positive/negative infinity

NaN

"Not-a-Number" value

undefined

Indicates that a variable has not been assigned a value

JavaScript Global Functions

decodeURI()

Decodes a URI

decodeURIComponent()

Decodes a URI component

encodeURI()

Encodes a URI

encodeURIComponent()

Encodes a URI component

Q 7 Write short notes on:

20

A) XHTML

Marks distribution: explanation 4 marks + 1 marks benefit

Before I describe XHTML, it is probably best to understand where it has come from. All web Markup languages are based on SGML, a horrendously complicated language that is not designed for humans to write. SGML is what is called a *metalanguage*; that is, a language that is used to define other languages. To make its power available to web developers, SGML was used to create XML (eXtensible Markup Language), a simplified version, and also a metalanguage.

XML is a powerful format — you create your own tags and attributes to suit the type of document you're writing. By using a set group of tags and attributes and following the rules of XML, you've created a new Markup language.

This is what has been done to create XHTML (eXtensible HyperText Markup Language) — which is why you'll see XHTML being called a *subset* or *application* of XML. The pre-existing HTML 4.01 tags and attributes were used as the vocabulary of this new Markup language, with XML providing the rules of how they are put together.

So, using XHTML, you are really writing XML code, but restricting yourself to a predetermined set of elements. This gives you all the benefits of XML (see below), while avoiding the complications of true XML; bridging the gap for developers who might not fancy taking on something as tricky as full-on XML. As you're coding under the guise of XHTML, all of the tags available to you should be

Internet Programming

familiar. Writing XHTML requires that you follow the rules of conformant XML, such as correct syntax and structure. As XHTML looks so much like classic HTML, it faces no compatibility problems as long as some simple coding guidelines are followed.

If all of this sounds a bit heavy, don't worry. Transitioning to XHTML is quite a simple process, with only a few rules to remember.

Benefits of XHTML

The benefits of adopting XHTML now or migrating your existing site to the new standards are many. First of all, they ensure excellent *forward-compatibility* for your creations. XHTML is the new set of standards that the web will be built on in the years to come, so future-proofing your work early will save you much trouble later on. Future browser versions might stop supporting *deprecated elements* from old HTML drafts, and so many old basic-HTML sites may start displaying incorrectly and unpredictably.

Once you have used XHTML for a short time, it is no more difficult to use than HTML ever was, and in ways is easier since it is built on a more simplified set of standards. Writing code is a more streamlined experience, as gone are the days of browser hacks and display tricks. Editing your existing code is also a nicer experience as it is infinitely cleaner and more self-explanatory. Browsers can also interpret and display a clean XHTML page quicker than one with errors that the browser may have to handle.

A well-written XHTML page is more accessible than an old style HTML page, and is guaranteed to work in any standards-compliant browser (which the latest round have finally become) due to the insistence on rules and sticking to accepted W3C specifications. As mentioned above, XHTML allows greater access to configurations other than a computer and browser. This interoperability is another aspect of XHTML's greater accessibility.

b) **E-commerce:**

Marks distribution: explanation with example 5 marks

Electronic commerce, commonly known as **e-commerce** or **eCommerce**, consists of the buying and selling of products or services over electronic systems such as the Internet and other computer networks. The amount of trade conducted electronically has grown extraordinarily with widespread Internet usage. The use of commerce is conducted in this way, spurring and drawing on innovations in electronic funds transfer, supply chain management, Internet marketing, online transaction processing, electronic data interchange (EDI), inventory management systems, and automated data collection systems. Modern electronic commerce typically uses the World Wide Web at least at some point in the transaction's lifecycle, although it can encompass a wider range of technologies such as e-mail as well.

A large percentage of electronic commerce is conducted entirely electronically for virtual items such as access to premium content on a website, but most electronic commerce involves the transportation of physical items in some way. Online retailers are sometimes known as e-tailers and

Internet Programming

online retail is sometimes known as **e-tail**. Almost all big retailers have electronic commerce presence on the World Wide Web.

Electronic commerce that is conducted between businesses is referred to as business-to-business or B2B. B2B can be open to all interested parties (e.g. commodity exchange) or limited to specific, pre-qualified participants (private electronic market). Electronic commerce that is conducted between businesses and consumers, on the other hand, is referred to as business-to-consumer or B2C. This is the type of electronic commerce conducted by companies such as Amazon.com. Online shopping is a form of electronic commerce where the buyer is directly online to the seller's computer usually via the internet. There is no intermediary service. The sale and purchase transaction is completed electronically and interactively in real-time such as Amazon.com for new books. If an intermediary is present, then the sale and purchase transaction is called electronic commerce such as eBay.com.

Electronic commerce is generally considered to be the sales aspect of e-business. It also consists of the exchange of data to facilitate the financing and payment aspects of the business transactions.

C) **Web System Architecture:**

Marks distribution: explanation with example 5 marks

For the purpose of this Working Group and this architecture, and without prejudice toward other definitions, we will use the following definition:

A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.]

Agents and Services

A Web service is an abstract notion that must be implemented by a concrete agent. (See Figure 1-1) The agent is the concrete piece of software or hardware that sends and receives messages, while the service is the resource characterized by the abstract set of functionality that is provided. To illustrate this distinction, you might implement a particular Web service using one agent one day (perhaps written in one programming language), and a different agent the next day (perhaps written in a different programming language) with the same functionality. Although the agent may have changed, the Web service remains the same.

Requesters and Providers

Internet Programming

The purpose of a Web service is to provide some functionality on behalf of its owner -- a person or organization, such as a business or an individual. The *provider entity* is the person or organization that provides an appropriate agent to implement a particular service. (See Figure 1-1: Basic Architectural Roles.) A *requester entity* is a person or organization that wishes to make use of a provider entity's Web service. It will use a *requester agent* to exchange messages with the provider entity's *provider agent*. (In most cases, the requester agent is the one to initiate this message exchange, though not always. Nonetheless, for consistency we still use the term "requester agent" for the agent that interacts with the provider agent, even in cases when the provider agent actually initiates the exchange.)

A word on terminology: Many documents use the term service provider to refer to the provider entity and/or provider agent. Similarly, they may use the term service requester to refer to the requester entity and/or requester agent. However, since these terms are ambiguous -- sometimes referring to the agent and sometimes to the person or organization that owns the agent -- this document prefers the terms *requester entity*, *provider entity*, *requester agent* and *provider agent*. In order for this message exchange to be successful, the requester entity and the provider entity must first agree on both the semantics and the mechanics of the message exchange.

The mechanics of the message exchange are documented in a Web service description (WSD). (See Figure 1-1) The WSD is a machine-processable specification of the Web service's interface, written in WSDL. It defines the message formats, datatypes, transport protocols, and transport serialization formats that should be used between the requester agent and the provider agent. It also specifies one or more network locations at which a provider agent can be invoked, and may provide some information about the message exchange pattern that is expected. In essence, the service description represents an agreement governing the mechanics of interacting with that service. (Again this is a slight simplification that will be explained further in **3.3 Using Web Services**.)

Semantics:

The semantics of a Web service is the shared expectation about the behavior of the service, in particular in response to messages that are sent to it. In effect, this is the "contract" between the requester entity and the provider entity regarding the purpose and consequences of the interaction. Although this contract represents the overall agreement between the requester entity and the provider entity on how and why their respective agents will interact, it is not necessarily written or explicitly negotiated. It may be explicit or implicit, oral or written, machine processable or human oriented, and it may be a legal agreement or an informal (non-legal) agreement. (Once again this is a slight simplification that will be explained further in **3.3 Using Web Services**.)

While the service description represents a contract governing the mechanics of interacting with a particular service, the semantics represents a contract governing the meaning and purpose of that interaction. The dividing line between these two is not necessarily rigid. As more semantically rich languages are used to describe the mechanics of the interaction, more of the essential information may migrate from the informal semantics to the service description. As this migration occurs, more of the work required to achieve successful interaction can be automated.

Overview of Engaging a Web Service:

There are many ways that a requester entity might engage and use a Web service. In general, the following broad steps are required, as illustrated in Figure 1-1: (1) the requester and provider


Internet Programming

entities become known to each other (or at least one becomes known to the other); (2) the requester and provider entities somehow agree on the service description and semantics that will govern the interaction between the requester and provider agents; (3) the service description and semantics are realized by the requester and provider agents; and (4) the requester and provider agents exchange messages, thus performing some task on behalf of the requester and provider entities. (I.e., the exchange of messages with the provider agent represents the concrete manifestation of interacting with the provider entity's Web service.) These steps are explained in more detail in **3.4 Web Service Discovery**. Some of these steps may be automated, others may be performed manually.

D) **RSS:**

Marks distribution: explanation with example carries 5 marks

RSS (most commonly expanded as **Really Simple Syndication**) is a family of web feed formats used to publish frequently updated works—such as blog entries, news headlines, audio, and video—in a standardized format. An RSS document (which is called a "feed", "web feed", or "channel") includes full or summarized text, plus metadata such as publishing dates and authorship. Web feeds benefit publishers by letting them syndicate content automatically. They benefit readers who want to subscribe to timely updates from favored websites or to aggregate feeds from many sites into one place. RSS feeds can be read using software called an "RSS reader", "feed reader", or "aggregator", which can be web-based, desktop-based, or mobile-device-based. A standardized XML file format allows the information to be published once and viewed by many different programs. The user subscribes to a feed by entering into the reader the feed's URI or by clicking an RSS icon in a web browser that initiates the subscription process. The RSS reader checks the user's subscribed feeds regularly for new work, downloads any updates that it finds, and provides a user interface to monitor and read the feeds. RSS allows users to avoid manually inspecting all of the websites they are interested in, and instead subscribe to websites such that all new content is pushed onto their browsers when it becomes available.

RSS formats are specified using XML, a generic specification for the creation of data formats. Although RSS formats have evolved from as early as March 1999, it was between 2005 and 2006 when RSS gained widespread use, and the (")" icon was decided upon by several major Web browsers.

