University Of Babylon
College of Computer Technology
Department Of Information Networks

Web Page Design

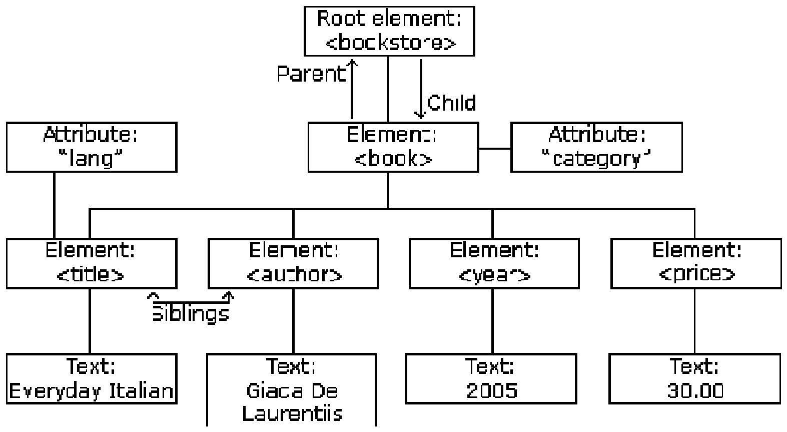
XML DOM

HAIDER M. HABEEB
MSC. INFORMATION TECHNOLOGY

XML DOM

- The XML DOM defines a standard way for accessing and manipulating XML documents.
- The DOM presents an XML document as a tree-structure.
- Knowing the XML DOM is a must for anyone working with XML.

XML DOM Tree



XML DOM

- **What You Should Already Know?**
  - HTML
  - XML
  - JavaScript

Ready? ☺ OR ☹

XML DOM

- **What is the XML DOM?**
- The XML DOM defines the **objects and properties** of all XML elements, and the **methods** (interface) to access them.

- **The XML DOM is a standard for how to get, change, add, or delete XML elements.**

XML DOM

- **DOM Nodes**
- According to the DOM, everything in an XML document is a **node**.
  - The entire document is a document node
  - Every XML element is an element node
  - The text in the XML elements are text nodes
  - Every attribute is an attribute node
  - Comments are comment nodes

XML DOM

- **Text is Always Stored in Text Nodes**
  - A common error in DOM processing is to expect an element node to contain text.
  - However, the text of an element node is stored in a text node.
  - In this example: **<year>2005</year>**, the element node <year>, holds a text node with the value "2005".
  - "2005" is **not** the value of the <year> element!

XML DOM Parser

- **XML Parser**
  - The XML DOM contains methods (functions) to traverse XML trees, access, insert, and delete nodes.
  - However, before an XML document can be accessed and manipulated, it must be loaded into an XML DOM object.
  - An XML parser reads XML, and converts it into an XML DOM object that can be accessed with JavaScript.
  - Most browsers have a built-in XML parser.

XML DOM Parser

- **Load an XML Document**

```
if (window.XMLHttpRequest)
  {
  xhttp=new XMLHttpRequest();
  }
else // IE 5/6
  {
  xhttp=new ActiveXObject("Microsoft.XMLHTTP");
  }
xhttp.open("GET","books.xml",false);
xhttp.send();
xmlDoc=xhttp.responseXML;
```

XML DOM Parser

- **The loadXMLDoc() Function**

```
function loadXMLDoc(dname)
{
if (window.XMLHttpRequest)
  {
  xhttp=new XMLHttpRequest();
  }
else
  {
  xhttp=new ActiveXObject("Microsoft.XMLHTTP");
  }
xhttp.open("GET",dname,false);
xhttp.send();
return xhttp.responseXML;
}
```

XML DOM Parser

- **An External JavaScript for loadXMLDoc()**

```
<html>
<head>
<script type="text/javascript" src="loadxmldoc.js">
</script>
</head>
<body>

<script type="text/javascript">
xmlDoc=loadXMLDoc("books.xml");

code goes here.....

</script>

</body>
</html>
```

XML DOM - Accessing Nodes

```
<html>
<head>
<script type="text/javascript"
src="loadxmldoc.js"></script>
</head>
<body>

<script type="text/javascript">

xmlDoc=loadXMLDoc("books.xml");
x=xmlDoc.getElementsByTagName("title");
document.write(x[2].childNodes[0].nodeValue);

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

XML DOM - Properties and Methods

- The nodes can be accessed with 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 and Methods

XML DOM Properties	**XML DOM Methods**
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.	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.

XML DOM - Accessing Nodes

- You can access a node in three ways:
  1. By using the getElementsByTagName() method
  2. By looping through (traversing) the nodes tree.
  3. By navigating the node tree, using the node relationships.

XML DOM - Accessing Nodes

- **The getElementsByTagName() Method**
- **Syntax**
  - *node*.getElementsByTagName(*"tagname"*);
- **Example**
  - x.getElementsByTagName("title");
  - xmlDoc.getElementsByTagName("title");

XML DOM - Accessing Nodes

- **DOM Node List**
  - The getElementsByTagName() method returns a node list. A node list is an **array of nodes**.

xmlDoc=loadXMLDoc("books.xml");
x=xmlDoc.getElementsByTagName("title");

  - The <title> elements in x can be accessed by index number. To access the third <title> you can write:

y=x[2];

- **Note:** The index starts at 0.

XML DOM - Accessing Nodes

- **DOM Node List Length**
  - The length property defines the length of a node list (the number of nodes).
  - You can loop through a node list by using the length property:

```
xmlDoc=loadXMLDoc("books.xml");

x=xmlDoc.getElementsByTagName("title");

for (i=0;i<x.length;i++)
  {
  document.write(x[i].childNodes[0].nodeValue);
  document.write("<br />");
  }
```

XML DOM - Accessing Nodes

- **Traversing Nodes**
  - The following code loops through the child nodes, that are also element nodes, of the root node:

```
xmlDoc=loadXMLDoc("books.xml");

x=xmlDoc.documentElement.childNodes;

for (i=0;i<x.length;i++)
{
  if (x[i].nodeType==1)
  {//Process only element nodes (type 1)
  document.write(x[i].nodeName);
  document.write("<br />");
  }
}
```

XML DOM - Accessing Nodes

- **Navigating Node Relationships**
  - The following code navigates the node tree using the node relationships:

```
xmlDoc=loadXMLDoc("books.xml");
x=xmlDoc.getElementsByTagName("book")[0].childNodes;
y=xmlDoc.getElementsByTagName("book")[0].firstChild;

for (i=0;i<x.length;i++)
{
if (y.nodeType==1)
  {//Process only element nodes (type 1)
  document.write(y.nodeName + "<br />");
  }
y=y.nextSibling;
}
```

XML DOM Node Information

- **Node Properties**
  - In the XML DOM, each node is an **object**.
  - Objects have methods and properties, that can be accessed and manipulated by JavaScript.

- Three important node properties are:
  - nodeName
  - nodeValue
  - nodeType

XML DOM Node Information

- **The nodeName Property**
  - The nodeName property specifies the name of a node.
  - nodeName is read-only
  - nodeName of an element node is the same as the tag name
  - nodeName of an attribute node is the attribute name
  - nodeName of a text node is always #text
  - nodeName of the document node is always #document

```
xmlDoc=loadXMLDoc("books.xml");
document.write(xmlDoc.documentElement.nodeName);
```

XML DOM Node Information

- **The nodeValue Property**
- The nodeValue property specifies the value of a node.
    - nodeValue for element nodes is undefined
    - nodeValue for text nodes is the text itself
    - nodeValue for attribute nodes is the attribute value

```
xmlDoc=loadXMLDoc("books.xml");

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

XML DOM Node Information

- **Change the Value of an Element**
- The following code changes the text node value of the first <title> element:

```
xmlDoc=loadXMLDoc("books.xml");

x=xmlDoc.getElementsByTagName("title")[0].childNodes[0];
x.nodeValue="Easy Cooking";
```

XML DOM Node Information

- **The nodeType Property**
- The nodeType property specifies the type of node.
- nodeType is read only.
- The most important node types are:

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