

XML Strengths & Weakness' with DOM, ASP & XSL

By Nakul Goyal (nakul@cwsteam.com)

Visit the Author's Homepage at <http://www.nakul.net>

XML Strengths & Weakness' with DOM, ASP & XSL

Introduction

This article explains in a very simple manner how you can use XML in ASP and VB. Since the inception of XML people many developers have wondered why we need XML... How is it better than HTML? For starters, XML is far more powerful than HTML, and that power resides in the "X" in XML (which stands for "eXtensible"). Rather than providing a set of pre-defined tags, as in the case of HTML, XML specifies the standards with which you can define your own markup languages with their own sets of tags. XML is therefore a meta-markup language, allowing you to define an infinite number of markup languages based upon the standards defined by XML.

XML was created so that richly structured documents could be used over the web. The only viable alternatives, HTML and SGML, are not practical for this purpose. XML allows you to define all sorts of tags with all sorts of rules, such as tags representing business rules or tags representing data description or data relationships

XML Definitions

As with any technology, XML has its own acronym-riddled lingo. Some of the important acronyms to know include:

- DTD - In XML, the definition of a valid markup is handled by a Document Type Definition (DTD), which communicates the structure of the markup language. The DTD specifies the validity of each tag.
- XSL - The Extensible Style Language (XSL) is the style language for XML.
- XML Pointer Language (XPointer) and XML Linking Language (XLink) - these two technologies define a standard way to represent links between resources. In addition to simple links, like HTML's <A> tag, XML has mechanisms for links between multiple resources and links between read-only resources. XPointer describes how to address a resource; XLink describes how to associate two or more resources.
- XML Flow Architecture - XML offers a three-tier architecture. It can be generated from existing databases using 3-tier model. We can maintain business rules separately

Why XML Should Be Used

XML contains a bevy of benefits. Some of the most profound benefits include:

- Authors and providers can design their document using XML, instead of being stuck with HTML. They can be explicitly tailored for an audience, so the

Currently undergoing his Master of Sciences in Information Technology at Panjab University, Chandigarh, Nakul Goyal is a BCA from Punjab Technical University, Jalandhar, India. He's an MCP, CIW Associate & Brainbench Certified 'Most Valuable Professional', Co-Founder of CWSTEAM.COM

XML Strengths & Weakness' with DOM, ASP & XSL

By Nakul Goyal (nakul@cwsteam.com)

Visit the Author's Homepage at <http://www.nakul.net>

cumbersome problems with HTML can evaporate: therefore, authors and designers will be free to invent their own markup elements.

- Information can be richer and easier to use, because the hypertext linking abilities of XML are much greater and simpler than those of HTML.
- XML can provide more (and better) facilities for browser presentation and performance.
- XML certainly compresses exceedingly well. Since data compression algorithms operate on the concept of maximizing the entropy of a given input stream, it stands to reason that a highly ordered input stream consisting of regular, repeating tag sequences will compress exceedingly well- much better than standard text which contains generally far less order thus increase in performance.

Weakness of XML

XML is, obviously, not a cure-all, free of disadvantages... else we would be using XML and nothing else! There are some drawbacks and weaknesses of XML:

- XML markup can be incredibly verbose, depending on the vocabulary in question.
- XML is platform-neutral
- All the pieces aren't yet in place to do whatever you want with XML - certainly not in a fully standards-compliant form, anyhow. We've got XSL/XSLT but they are not fully developed yet.
- There are still some problems with Microsoft XML Parser July release.
- XML Hypertext Transfer Protocol (XML-HTTP) problems still exist.

Performance of XML

When you are designing your XML-based Web application and you need to know what kind of performance to expect from your XML server. It is hard to generalize, because there are so many variables -- such as the size of the XML documents, the amount of script code required to process the documents, the amount of output generated, etc. For example, major variables that can affect the performance of MSXML include:

- The Kind of XML Data
- The ratio of tags to text
- The ratio of attributes to elements
- The amount of discarded white space

XML and DOM

Microsoft has provided us DOM (Data Object Model for XML). With the XML Document Object Model (DOM), you can load and parse XML files, gather information about those files, and navigate and manipulate those files. To know

Currently undergoing his Master of Sciences in Information Technology at Panjab University, Chandigarh, Nakul Goyal is a BCA from Punjab Technical University, Jalandhar, India. He's an MCP, CIW Associate & Brainbench Certified 'Most Valuable Professional', Co-Founder of CWSTEAM.COM

XML Strengths & Weakness' with DOM, ASP & XSL

By Nakul Goyal (nakul@cwsteam.com)

Visit the Author's Homepage at <http://www.nakul.net>

about the details of XML DOM, please refer to the XML DOM site at Microsoft's site: <http://msdn.microsoft.com/xml/>.

We are going to create an XML file using static data and Data from Database using ADO. The DOM methods createNode and appendChild, and the text property are used to construct an XML tree.

Now that we've discussed the reasons for using XML, it's time to look at some source code! Now we will examine some ASP scripts that create and display XML data!

Above, we looked at the advantages and disadvantages of XML. Now we'll roll up our sleeves and get down to creating (and displaying) some XML documents using ASP code!

XML with ASP

The following example illustrates how to create an XML tree (in memory) and then persist it to disk (using the save method).

```
<%  
Dim xmldoc  
Set xmldoc = Server.CreateObject("Microsoft.XMLDOM")  
  
' Check to see if a document has data. If it does, don't build it  
If (xmldoc.childNodes.length = 0) Then  
' Build the XML document  
Set root = xmldoc.createNode("element", "Hi-Tech", "")  
xmldoc.appendChild (root)  
  
Set onode = xmldoc.createNode("element", "Employee", "")  
onode.Text = "Gurpreet Singh"  
  
xmldoc.documentElement.appendChild (onode)  
Set inode = xmldoc.createNode("element", "Address", "")  
  
onode.appendChild (inode)  
  
Set child = xmldoc.createNode("element", "Address1", "")  
child.Text = "Nepean Ont"  
  
inode.appendChild (child)  
  
Set child = xmldoc.createNode("element", "Address2", "")
```

Currently undergoing his Master of Sciences in Information Technology at Panjab University, Chandigarh, Nakul Goyal is a BCA from Punjab Technical University, Jalandhar, India. He's an MCP, CIW Associate & Brainbench Certified 'Most Valuable Professional', Co-Founder of CWSTEAM.COM

XML Strengths & Weakness' with DOM, ASP & XSL

By Nakul Goyal (nakul@cwsteam.com)

Visit the Author's Homepage at <http://www.nakul.net>

```
child.Text = "Canada"
inode.appendChild (child)
End If

xmldoc.save (Server.MapPath("savedI2.xml"))
%>
```

Here we have created an XMLDOM Object. We then create a Root node and its child node using the createNode function. Finally we append the nodes after assigning the text property to nodes. In the end we save the in-memory XML tree to a file.

We can also build an XML file from the results of a database query. The following example illustrates how to accomplish this. (For this example I used the pubs database, which comes along with SQL Server.)

```
<%
'Open database connection
Set conn = Server.CreateObject("ADODB.Connection")
dsn = "DSN=pubs;UID=sa;PWD="
conn.Open dsn

'Create XMLDOM Object
Dim xmldoc
Set xmldoc = Server.CreateObject("Microsoft.XMLDOM")

If (xmldoc.childNodes.length = 0) Then
' Build the XML document
Set root = xmldoc.createNode("element", "Hi-Tech", "")

xmldoc.appendChild (root)

' Queries the database for customer data
Sql = "select au_lname,au_fname,au_id from authors"
Set rs = conn.Execute(Sql)

rs.MoveFirst

'Loop through the recordset
Do While Not rs.EOF
Set onode = xmldoc.createNode("element", "Employee", "")
xmldoc.documentElement.appendChild (onode)

Set inode = xmldoc.createNode("element", "Name", "")
```

Currently undergoing his Master of Sciences in Information Technology at Panjab University, Chandigarh, Nakul Goyal is a BCA from Punjab Technical University, Jalandhar, India. He's an MCP, CIW Associate & Brainbench Certified 'Most Valuable Professional', Co-Founder of CWSTEAM.COM

XML Strengths & Weakness' with DOM, ASP & XSL

By Nakul Goyal (nakul@cwsteam.com)

Visit the Author's Homepage at <http://www.nakul.net>

```
inode.Text = rs.fields(0) & " " & rs.fields(1)
onode.appendChild (inode)

'Grab another recordset based on the authorID
Sql = "select title_id,royaltyper from titleauthor " & _
      "where au_id = '" & rs.fields(2) & "'"

Set rs2 = conn.Execute(Sql)

If Not (rs2.EOF = True And rs2.bof = True) Then
    Set inode = xmldoc.createElement("element", "Titles", "")
    onode.appendChild (inode)
    Set child = xmldoc.createElement("element", "TitleId", "")
    child.Text = rs2.fields(0)
    inode.appendChild (child)
    Set child = xmldoc.createElement("element", "royalty", "")
    child.Text = rs2.fields(1)
    inode.appendChild (child)

    rs2.Close
    Set rs2 = Nothing
End If

rs.movenext
Loop

Set rs = Nothing
End If

'Save the XML doc
xmldoc.save server.mappath("saved.xml")

'----- DISPLAY THE XML DATA -----
' Linking XML and XSL together
sourceFile = Server.MapPath("saved.xml")
styleFile = Server.MapPath("saved.xsl")

set source = Server.CreateObject("Microsoft.XMLDOM")
source.async = false
source.load(sourceFile)
set style = Server.CreateObject("Microsoft.XMLDOM")
style.async = false
```

Currently undergoing his Master of Sciences in Information Technology at Panjab University, Chandigarh, Nakul Goyal is a BCA from Punjab Technical University, Jalandhar, India. He's an MCP, CIW Associate & Brainbench Certified 'Most Valuable Professional', Co-Founder of CWSTEAM.COM

XML Strengths & Weakness' with DOM, ASP & XSL

By Nakul Goyal (nakul@cwsteam.com)

Visit the Author's Homepage at <http://www.nakul.net>

```
style.load(styleFile)
Response.Write source.transformNode(style)
%>
```

In the above example we first make the connection to our SQL Server database using Connection Object of ADO. Next, we create the recordset, populating it with the names of our authors in the authors table. We populate a second recordset based on the current authors ID. Eventually, all of this data is incorporated into the XML Tree using the createNode function and finally appending the nodes. Finally, the XML data is displayed using an XSL stylesheet. Remember that XML is designed to **not** store information on how the data should be displayed. Rather, XML is used only as a holding place for data. To turn an XML document into a nice-looking HTML document, you need to use XSL.

Converting XML to HTML

We can maintain XML data on the server and format it into HTML using XSL and then send it to the client. We can do so using any server-side techniques, such as an ASP page. To use XSL, you need to first create an XSL document. This document is a glorified stylesheet, explaining how to display the various XML tags. For example, we could have the following XSL stylesheet:

```
<?xml version='1.0'?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">
  <xsl:template match="/">
    <HTML>
    <STYLE>
      .fr1 { width: 30em; }
      BODY { margin:0px; width: 30em;
            font-family: Arial, Helvetica, sans-serif; font-size: smaller;}
      P { margin-top: .5em; margin-bottom: .25em; }
      HR { color: #888888; }
      .H1 { color: #660033; font-weight: bold; vertical-align: top; }
      .Param { font-size: smaller; vertical-align: top; }
      .tagline { font-style: italic; font-size: smaller; text-align: right; }
      .body { text-align: justify; background-color: #FFFFDD; }
      .dingbat { font-family: WingDings; font-style: normal; font-size: xx-small;
      }
      .person { font-weight: bold; }
      .label { font-weight: bold; }
      .self { font-style: italic; font-size: smaller;}
      #menu { border: 2px solid black; padding: 1em; background-color:
#888833; }
      .menutext { color: #FFFFDD; font-family: Times, serif; font-style: italic;
```

Currently undergoing his Master of Sciences in Information Technology at Panjab University, Chandigarh, Nakul Goyal is a BCA from Punjab Technical University, Jalandhar, India. He's an MCP, CIW Associate & Brainbench Certified 'Most Valuable Professional', Co-Founder of CWSTEAM.COM

XML Strengths & Weakness' with DOM, ASP & XSL

By Nakul Goyal (nakul@cwsteam.com)

Visit the Author's Homepage at <http://www.nakul.net>

```
        vertical-align: top; text-align:center; }
    .menuhead { color: #FFFFDD; font-family: Times, serif; font-weight: bold;
        vertical-align: top; text-align:center; margin-bottom: .5em; }
</STYLE>
<xsl:for-each select="Hi-Tech/Employee">
    <TABLE>
<TR><TD Class="H1"><xsl:value-of select="Name" /></TD> </TR>
<TR>
<xsl:for-each select="Titles">
<TR><TD><xsl:entity-ref name="nbsp"/></TD>
    <TD Class="H1"><xsl:value-of select="TitleId" /></TD>
    <TD Class="H1"><xsl:value-of select="royalty" /></TD>
</TR>
</xsl:for-each></TR>
</TABLE>
</xsl:for-each>
</HTML>
</xsl:template>
</xsl:stylesheet>
```

This XSL stylesheet can then be loaded and applied using the transformNode method. (For an example of this, see the previous code example, where, at the end, we displayed the XML data from an ADO recordset using an XSL stylesheet.

Well, I hope this article has answered some of your questions on XML. Hopefully you've learned some of the advantages and disadvantages of XML, when it should be used, and how it can be used...

--

About the Author

Nakul Goyal, currently doing Master of Sciences in Information Technology from Panjab University, Chandigarh. A Bachelor of Computer Applications from Panjab Technical University, he is passionate towards the Cyber World & he likes to write about Technology. He's also a Microsoft Certified Professional and a Brainbench Certified 'MVP'(Most Valuable Professional). Also the Co-Founder of CWSTeam (<http://www.cwsteam.com>). Contact Nakul Goyal by Email: nakul@cwsteam.com

Currently undergoing his Master of Sciences in Information Technology at Panjab University, Chandigarh, Nakul Goyal is a BCA from Panjab Technical University, Jalandhar, India. He's an MCP, CIW Associate & Brainbench Certified 'Most Valuable Professional', Co-Founder of CWSTEAM.COM