

### Well-Behaved XML Document

1. Every element has a **start** and an **end tag**
2. **Single, unique** root element
3. Element and attributes names are **case sensitive**
4. **Proper nesting**, no overlaps.
5. **Attribute values** must be in quotes

### XML Syntax

```
<?xml version="1.0" encoding="EBCDIC" standalone="no"?>
```

An object has properties, i.e. element name.  
**Tag** defines start and end of an element:

```
<alias>Neptune</alias>
```

**Name:** Begin with letter or "\_"

**Special characters:**

- &lt; < less than
- &gt; > greater than
- &amp; & ampersand
- &apos; ' apostrophe or single quote
- &quot; " double quotes
- &#... decimal character code
- &#x... hexadecimal character code

```
<P> The dollar is &gt; than the Ramd</P>
```

**Attributes:**

```
<name attribute="value">... <WPSchaub alias="Neptune">
```

**Comments:**

```
<!-- This is a simple comment -->
```

### XDR Syntax (XML Data Reduced)

```
<Schema name="Rocket"
  <ElementType name="Fin" content="eltOnly" order="one">
    <AttributeType name="Trian" type="string" required="yes"/>
  >
  <attribute type="Fin"/>
  <element type="weight"/>
</ElementType>
</Schema>
```

- eltOnly** Contains elements only
- empty** Can not have content
- textOnly** Can have text content only
- mixed** Can contain text and other elements
- one** Contain only one of the subelements listed
- ElementType** Declares an element and indicates content
- AttributeType** Declares an attribute
- type** Indicates type of data (boolean, string, float, int, number, date, char)
- required** Indicates if attribute is required

### DOM (Document Object Model)

```
var objDoc = new ActiveXObject("MSXML2.DOMDocument");
objDoc.async = false;

objDoc.loadXML("<news nuo='1'>Movies are great!</news>");
if ( objDoc.parseError.errorCode != 0 )
{
  result += objDoc.parseError.errorCode + "\n";
  result += objDoc.documentElement.nodeName + "\n";
  result += objDoc.documentElement.text + "\n";
}

result = rootElem.attributes.getNamedItem('no').nodeValue;

var colScenes = rootElem.selectNodes("scene")
for ( i=0; i < colScenes.length; i++ )
{
  result = colScenes.item(i).text;
}
```



### TLA Hell (Three Lettered Acronyms)

B2B	Business to Business
B2C	Business to Consumer
DOM	Document object model
DTD	Document type definition
EDI	Electronic Data Interchange
HTML	Hypertext Markup Language
SAX	Simple Api for XML
SDL	Service Description Language
SGML	Structured Generalised Markup Language
SOAP	Simple Object Access Protocol
W3C	World Wide Web Consortium
XML	Extended Markup Language
XQL	XML Query Language
XSL	Extensible Stylesheet language
XLL	XML linking language
XSD	W3C Schema syntax

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

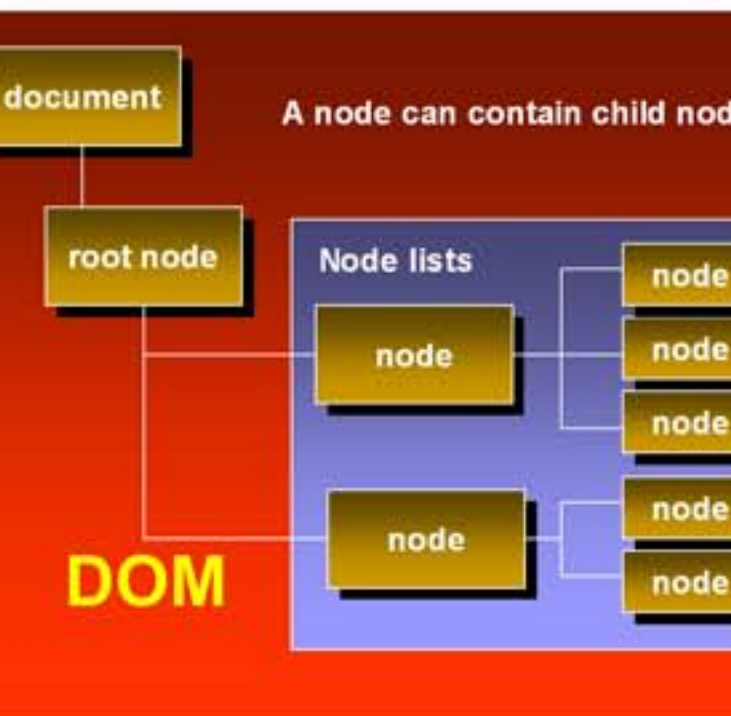
XML Declaration

```
<!DOCTYPE name [
  ...
]>
```

Document Definition

```
<doc_root_element>
  <data_element>
  ...
</data_element>
</doc_root_element>
```

Data



### XSL (Extensible Stylesheet Language)

```
<!-- Process any XML document creating a hierarchical -->
<!-- view -->
1 <?xml version="1.0" ?>
2 <xsl:stylesheet
3   version="1.0"
4   xml:space="default"
5   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
6
7   <xsl:template match="node()">
8     <DIV STYLE="margin-left:12pt;">
9       [start <xsl:value-of select="name()" />]
10      <xsl:apply-templates select="@*"/>
11      <xsl:apply-templates/>
12      [end <xsl:value-of select="name()" />]
13    </DIV>
14  </xsl:template>
15
16  <xsl:template match="@*">
17    [attr <xsl:value-of select="name()" />]:
18    <xsl:value-of select="."/>]
19  </xsl:template>
20
21  <xsl:template match="text()">
22    <xsl:value-of select="."/>]
23  </xsl:template>
24
25 </xsl:stylesheet>
```

### XML versus HTML

XML	Subset of SGML	Extensible
HTML	SGML application	Fixed Presentation

Standard Generalised Markup Language

### SOAP Data Type

XML	C++	VB	Description
byte	signed char	Not supported	1-byte signed integer, VT_I1
short integer	short int	Integer	2-byte signed integer, VT_I2
integer	int	Long	4-byte signed integer, VT_I4
float	float	Single	4-byte floating point number, VT_R4
double	double	Double	8-byte floating point number, VT_R8
string	BSTR	String	Automation string, VT_BSTR
boolean	BOOL	Boolean	Boolean, VT_BOOL

XML is a simple but flexible data format.  
 XML has a standardized programming API.

XML parsers and other tools are available for all major operating systems and programming languages

World Wide Web Consortium (W3C) Create Schemas Describing Data



HTTP is the data request and response protocol

Invoke Procedures on Remote Systems

### Simple Object Access Protocol

```

Select a single child node with the specified ID
node.nodeFromId( "3" )
Select the first child node with the specified name
node.selectSingleNode( "weather" )
Select all child nodes with the specified name
node.selectNodes( "weather" )
Select all child nodes
node.childNodes
  
```

### DTD Syntax

```

Elements <IELEMENT ...>
Attributes <IATTLIST ...>
Entities <IENTITY>
Comments <!-- ... -->
  
```

### Internal DTD (Document Type Definition)

```
<?xml version="1.0"?>
<!DOCTYPE WEATHER [
  <!ELEMENT WEATHER (DEGREES, DATE)>
  <!ELEMENT DEGREES (#PCDATA)>
  <!ELEMENT DATE (#PCDATA)>
]>
```

```
< WEATHER >
  <DEGREES>20.5</DEGREES>
  <DATE>1999-08-17</DATE>
</ WEATHER >
```

### External DTD (Document Type Definition)

```
<<!DOCTYPE WEATHER SYSTEM "weather.dtd">
<!DOCTYPE catalog PUBLIC "-//weather//DTD Standard //EN" "http://www.weather.com/dtd/weather.dtd">
```

### Namespaces

Namespaces are declared in the prolog of the XML document

```
<?xml:namespace ns="http://invent/schema/ns" prefix="inv"?>
```

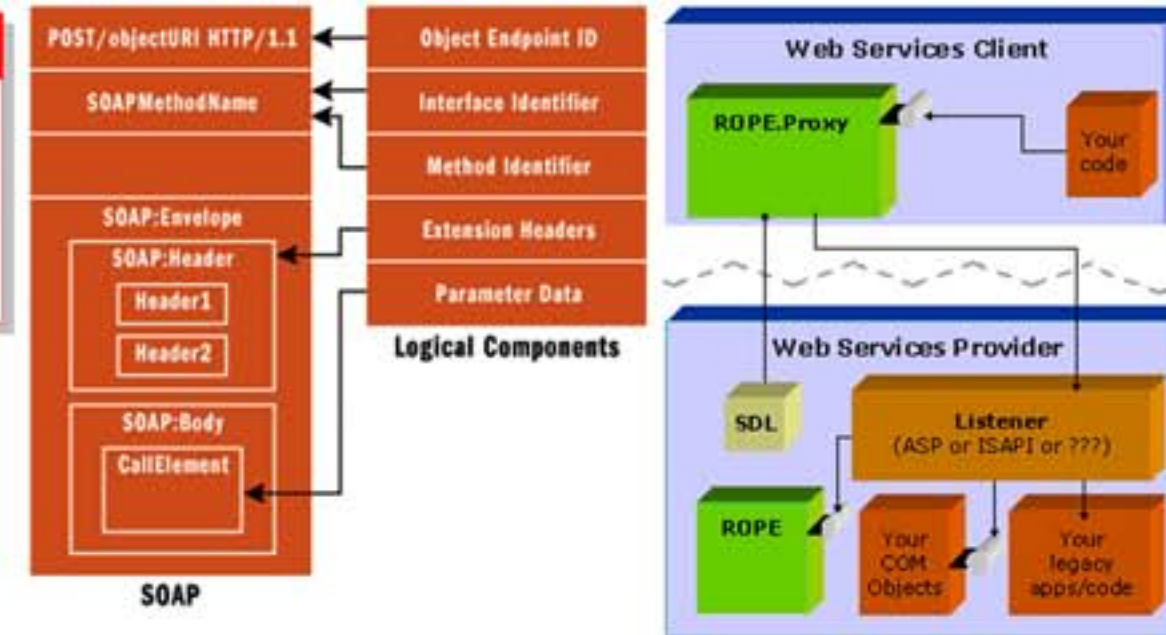
"inv" is namespace prefix that is used to make document tags unique

```
<?xml version="1.0"?>
<?xml:namespace ns=http://orders/schemaA/ns prefix="a"?>
<?xml:namespace ns=http://shipping/schemaB/nsprefix="b"?>
<example>
  <a:quantity>100</a:quantity >
  <b:quantity>50</b:quantity >
</example>
```

### Apply Style Sheet

```

'XML variables
Dim source As new MSXML.DomDocument
Dim srcroot As new MSXML.IXMLDOMElement
'XSL variables
Dim style As new MSXML.DomDocument
Dim styleroot As new MSXML.IXMLDOMElement
'Load both documents and get roots
source.load( "xmldata.xml" )
Set srcroot = xmldoc.documentElement
style.load( "xsldata.xsl" )
Set styleroot = xsldoc.documentElement
'Transformed data as string to response
Response.write srcroot.transformNode( styleroot
  
```



### SOAP

POST/objectURI HTTP/1.1

SOAPMethodName

SOAP:Envelope

SOAP:Header

Header1

Header2

SOAP:Body

CallElement

Logical Components

Object Endpoint ID

Interface Identifier

Method Identifier

Extension Headers

Parameter Data

Proxy

LoadServicesDescription ( )

ServicesDescription

Timeout

WireTransfer

ClearHeaders ( )

GetHeader ( )

GetPageByURI ( )

PostDataToURI ( )

SetHeader ( )

DataReceived

DataSent

Port

StatusCode

StatusText

Timeout

SOAPPackager

CDATAize ( )

ClearBody ( )

ClearHeaders ( )

Escape ( )

GetBody ( )

GetFirstParameterName ( )

GetMethodResult ( )

GetMethodStruct ( )

GetNameSpace ( )

GetNextParameterName ( )

GetParameter ( )

GetPayload ( )

GetSOAPHeader ( )

GetStruct ( )

GetStructElementCount ( )

isMethodAvailable ( )

LoadServicesDescription ( )

Reset ( )

SetFaultPayload ( )

SetParameter ( )

SetPayload ( )

SetPayloadData ( )

SetSOAPHeader ( )

ValidateDataAsType ( )

FaultActor

FaultCode

FaultDetail

FaultPayload

FaultString

GetServiceDescriptors

ServicesDescription

ServiceDescriptors

Add ( )

CollectionType

Count

Item

SDMethodInfo

CreateParameterArray ( )

FillParameterArray ( )

ComplexityType

Endpoint

FillParameterArraySize

InputStructure

Name

OutputStructure

ParameterOrder

ReturnType

SDEndpointInfo

Type

SDParameterInfo

Direction

Name

Type

### SOAP Request (Example)

```

POST /SomeAction HTTP/1.1
HOST: www.my-url.com
Content-Type: text/xml
Content-Length: nnnn
SOAPAction: SomeAction

<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/_
  envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/_
  soap/encoding/">
  <SOAP-ENV:Header>
    <!-- The SOAP header element is optional... -->
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <!-- Serialized object information... -->
  </SOAP-ENV:Body>
  <!-- Optional sub-elements... -->
</SOAP-ENV:Envelope>
```

### SOAP Response (Example)

```

200 OK
Content-Type: text/xml
Content-Length: nnnn
SOAPAction: SomeAction

Response content goes here ...
```

### SOAP Tools

X-Ray	XML Editor by archiTAG.
XML Authority	Schema maintenance, www.extensibility.com
BizTalk Editor	Schema maintenance
BizTalk Mapper	Map records (element) and fields (attribute)
BizTalk	BizTalk Management Desk BizTalk Server Administration Console BizTalk Document Tracking BizTalk Workflow Designer