



# Working with XML & Web Services

UNIT – 7 & 8

# XML

- ▶ XML stands for extensible Markup Language
- ▶ XML is text file that contains information organized in a structure that meets XML standards.
- ▶ XML is markup language much like HTML but both have many differences between them.
- ▶ Using XML you can create your own tags.XML is used to describe data.
- ▶ XML is platform independent.
- ▶ XML offers an effective way of defining standards for data transfer across different types of applications.
- ▶ XML can separate data from HTML.HTML is mainly used for presentation where XML focuses on data part.
- ▶ XML allows data only in well formed manner which makes easy to identify the data even multiple levels.

# XML

- ▶ In XML, you can create DTD(Data Type Definition) and XSD(XML Schema Definition) files to specify your own set of rules for user defined tags.
- ▶ In XML, you can create XSLT(XML Style sheet Transformation) files to show your XML data in particular format.
- ▶ Example :
  - ▶ `<age>35</age>`

# XML Parser

- ▶ XML Parser is small software tool that reads XML file for two purpose.
- ▶ XML Parser checks XML file is Well Formed and Verified or not.
- ▶ XML parser parses XML documents and sends data to display in a browser.
- ▶ The job of parser is to make sure that document meets the defined structures, validation and constraints.



# XML Rules

- ▶ Every XML document can have only one root element.
- ▶ Every start tag must have end tag.
- ▶ End tag can not contain any information
- ▶ If tag is empty ,the same tag can work as start and end tag.
- ▶ All attribute's value must be written between double quotes.
- ▶ All tags must be nested properly in xml file.
- ▶ Tags in XML are case sensitive.

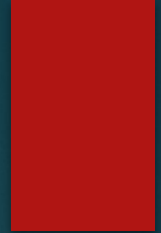
# XML Rules

- ▶ .NET has got special namespace for XML which gives you all classes to access XML features from top to bottom.  
namespace - using System.Xml
- ▶ It provides you no. of classess
- ▶ XmlDocument (used to create XML file / document)
- ▶ XmlDataDocument (used to create Data Based XML document)
- ▶ XmlNode(used to create Xml Node – element object)
- ▶ XmlComment (used to write comment in Xml doc)
- ▶ XmlElement (used to create Xml Element)
- ▶ XmlAttribute ( used to create Attribute for Xml Element)
- ▶ XmlNodeList (used to store list of node collection)

# Reading DataSets with XML

- ▶ You can read XML file / XML Data and store it into DataSet or DataTable.
- ▶ For reading XML into DataSet or DataTable, both have two important methods to read.
- ▶ ReadXml() method:
  - ▶ This method is used to read XML data from any XML file to dataset or datatable
  - ▶ It treats all main elements under root element as different tables. If you are using DataSet to read the data.
  - ▶ It automatically stores data in XML way, because you are already reading a file which is in XML format.
- ▶ ReadXmlSchema() method:
  - ▶ Before reading XML data into dataset, if you want to specify some schema rules before writing XML data to DataSet. You can use ReadXmlSchema() which allows you to specify XSD file and it restricts the data being copied into DataSet / DataTable.

# Writing DataSets with XML



- ▶ You can write DataSet or DataTable as XML file.
- ▶ For writing DataSet or DataTable as XML , both have two important methods.
- ▶ WriteXml() method:
  - ▶ This method is used to read data from DataSet and write data as XML data.
  - ▶ All Tables of DataSets are treated as Main Elements.
  - ▶ All Rows of Tables are treated as Sub Elements.
  - ▶ All Columns of Tables are treated as Data Elements which has actual data of table.
  - ▶ Of course there is only one root element as per XML rule.
- ▶ WriteXmlSchema() method:
  - ▶ In above method only data is written as Xml, but along with data if you also want to write schema information, you can use WriteXmlSchema method.



# Web Services

- ▶ Web Services are group of Web Methods where each Web Methods gives you some particular functionality.
- ▶ You have already developed many methods in C# / VB.NET.
- ▶ Web Methods are same as those methods, but all web methods are combined under a single Web Service is uploaded / kept on Web Server so that you can use its functionality through various Web Methods.
- ▶ Web Services has no. of web methods that provide different functionality which can be accessed by different types of application. You can use web services under your Windows Application as well as Web Application too.
- ▶ Web Services can be used by different applications regardless of programming languages, operating systems, hardware platform which uses them.

# Web Services



- ▶ An application which uses Web Services is called Web Service Client as it is used particular web service.
- ▶ You may not use all methods of web services. it depends on your requirements which web method you want to use from web service.

# Standards & Protocols for Web Service

- ▶ Various standards or protocols that work behind web service :
  - ▶ HTTP (Hyper Text Transport Protocol)
  - ▶ XML (eXtensible Markup Language)
  - ▶ SOAP (Simple Object Access Protocol)
  - ▶ WSDL (Web Services Description Language)
  - ▶ UDDI (Universal Description Discovery and Integration)

# HTTP



- ▶ we are already aware about HTTP is used over the internet as internet protocol.
- ▶ Using HTTP we can send and resources over the internet.
- ▶ Using Http Request obj we can send resources over server.
- ▶ Using Http Response obj we can receive response from server.
- ▶ When we use web service, we need to use resources that web services used / consumed.



# XML



- ▶ Web methods are like normal methods you can use using any programming language.
- ▶ Web methods take arguments and return the values.
- ▶ The arguments and return values use XML language so that they can be platform independent.
- ▶ XML contains standard data representation format that can be used on any platform.

# SOAP

- ▶ **SOAP** ( Simple Object Access Protocol) is a message protocol that allows distributed elements of an application to communicate.
- ▶ **SOAP** can be carried over a variety of lower-level protocols, including the web-related Hypertext Transfer Protocol (HTTP).
- ▶ Using SOAP , XML data is transferred over HTTP from one location to another location.
- ▶ SOAP is a standard communication protocol for interchanging information in a structured format in distributed environment.
- ▶ When a client application makes a request for a web method, a SOAP packet is created. This SOAP packet contains the name of the Web Method to be invoked and the parameters to be passed to Web Methods in an XML format.
- ▶ When SOAP packet arrives at the Web Server on which the Web Service resides, the Web Method name and its parameters are extracted from the SOAP packet. After web method is invoked , execute and again answer value is returned back using SOAP packet in reverse.

# WSDL

- ▶ **WSDL** stands for **Web Services Description Language**; **WSDL** is used to describe web services.
- ▶ To able to use Web Service , the developers need to know the methods exposed by Web Service and parameters to be passed to these methods.
- ▶ As we have already discussed that Web Services are platform independent .
- ▶ so, explanation of Web Methods i.e. its parameters, return values and body, everything should also be in standard format so that it can be shared among all platforms.
- ▶ This is achieved by using XML based description language called WSDL.
- ▶ WSDL is language which is used to describe Web Service and its Methods.

# WSDL

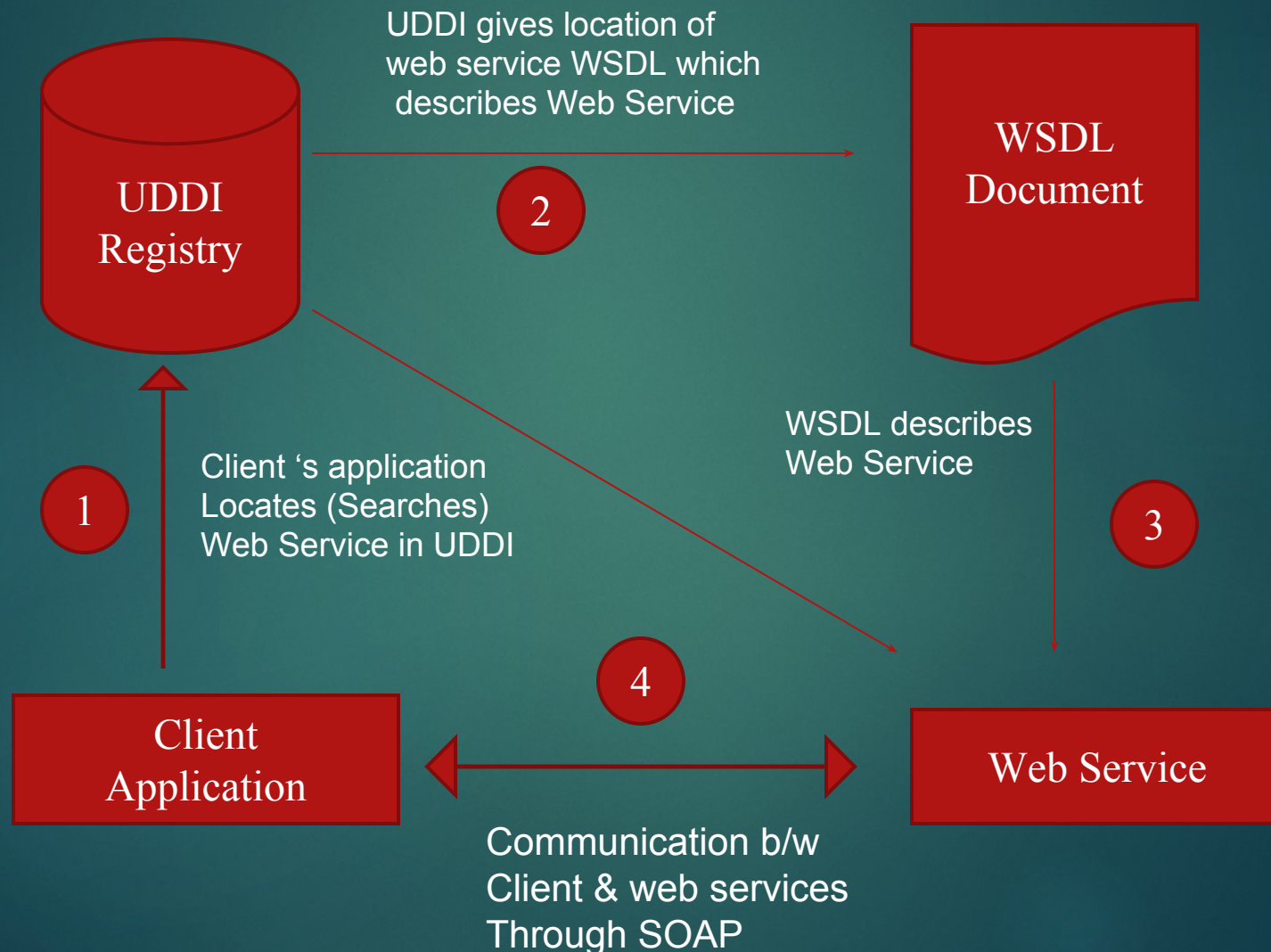
- ▶ WSDL is a markup language that describes a Web Services and all its Web Methods.
- ▶ WSDL contains following information about particular web services and its methods:
  - ▶ All the Web Services which are specified / created under particular Website.
  - ▶ The purpose of each Web Service.
  - ▶ The types of parameters and return values for each Web Method under Web Service.
  - ▶ The format to access each web service method.
  - ▶ Most importantly, URL at which a Web Service can be accessed.
- ▶ Finally you can see that WSDL is used to describe and locate Web Services.




# UDDI

- ▶ **Universal Description, Discovery, and Integration (UDDI)** is an XML-based registry for business internet services.
- ▶ UDDI is a directory service where different companies can register and search for their web services.
- ▶ UDDI provides standard mechanism to register and discover a Web Service.
- ▶ UDDI is a place (directory) for storing information about Web Services. The client who wants to access some web service, need to first find the particular web service and then need to use it. The place where you can find all the registered web services , is UDDI.
- ▶ When you register Web Service under UDDI, its registered and the URL of Web Service is stored under UDDI, which can be accessed by client after searching it.
- ▶ UDDI communicates via SOAP.

# Diagram to show how each standards communication with each other to access Web Service



- 
- ▶ First of all client application searches for the web application. we have already discussed that after creating services, they are registered under UDDI.
  - ▶ So when any client's application wants to access any web service, it searches for web service under UDDI.
  - ▶ If Web Service is registered , UDDI gives its location in a format of WSDL document.
  - ▶ We have already discussed that WSDL is XML based descriptor which describes web application.
  - ▶ Location of WSDL is given by UDDI.

# Creating a Web Service

- ▶ **Step (1)** : Select File -> New -> Web Site in Visual Studio, and then select ASP.NET Web Service.
- ▶ **Step (2)** : A web service file called Service.asmx and its code behind file, Service.cs is created in the App\_Code directory of the project.
- ▶ **Step (3)** : Change the names of the files to StockService.asmx and StockService.cs.
- ▶ **Step (4)** : The .asmx file has simply a WebService directive on it:  


```
<%@ WebService Language="C#" CodeBehind="~/App_Code/StockService.cs" Class="StockService" %>
```
- ▶ **Step (5)** : Open the StockService.cs file, the code generated in it is the basic Hello World service.




# Sample code of A Web service class


- ▶ `using System;`
- ▶ `using System.Collections;`
- ▶ `using System.ComponentModel;`
- ▶ `using System.Data;`
- ▶ `using System.Linq;`
- ▶ `using System.Web;`
- ▶ `using System.Web.Services;`
- ▶ `using System.Web.Services.Protocols;`
- ▶ `using System.Xml.Linq;`

- ▶ namespace StockService
- ▶ {
- ▶ // <summary>
- ▶ // Summary description for Service1 // <summary> [WebService(Namespace = "http://tempuri.org")]
- ▶ // [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1\_1)] [ToolboxItem(false)] // To allow this Web Service to be called from script, // using ASP.NET AJAX, uncomment the following line. // [System.Web.Script.Services.ScriptService]
- ▶ public class Service1 : System.Web.Services.WebService
- ▶ {
- ▶ [WebMethod]
- ▶ public string HelloWorld()
- ▶ { return "Hello World"; }
- ▶ }
- ▶ }


- 
- ▶ WSDL describes Web Service and its embedded Web Methods.
  - ▶ WSDL describes argument list(parameters) and returns values of each web method within web service. So that client can get the list of each of web methods.
  - ▶ We discussed that UDDI gives location of web service and description in form of WSDL.
  - ▶ Now finally when client gets physical location of WSDL, client can access Web Service via SOAP. SOAP is communication protocol which allows you to access web service and its Web Methods.

- 
- ▶ **Step (6)** : Change the code behind file to add the two dimensional array of strings for stock symbol, name and price and two web methods for getting the stock information.






```
[WebMethod]
public double GetPrice(string symbol)
{
    //it takes the symbol as parameter and returns price
    for (int i = 0; i < stocks.GetLength(0); i++)
    {
        if (String.Compare(symbol, stocks[i, 0], true) == 0)
            return Convert.ToDouble(stocks[i, 2]);
    }
    return 0;
}
```



```
[WebMethod]
public string GetName(string symbol)
{
    // It takes the symbol as parameter and
    // returns name of the stock
    for (int i = 0; i < stocks.GetLength(0); i++)
    {
        if (String.Compare(symbol, stocks[i, 0], true) == 0) return stocks[i,
1];
    }
    return "Stock Not Found";
}
}
```

**Step (7) :** Running the web service application gives a web service test page, which allows testing the service methods.



StockService Web Service

## StockService

The following operations are supported. For a formal definition, please review the [Service Description](#).

- [GetName](#)
- [GetPrice](#)
- [HelloWorld](#)

---

**This web service is using <http://tempuri.org/> as its default namespace.**

**Recommendation: Change the default namespace before the XML Web service is made public.**

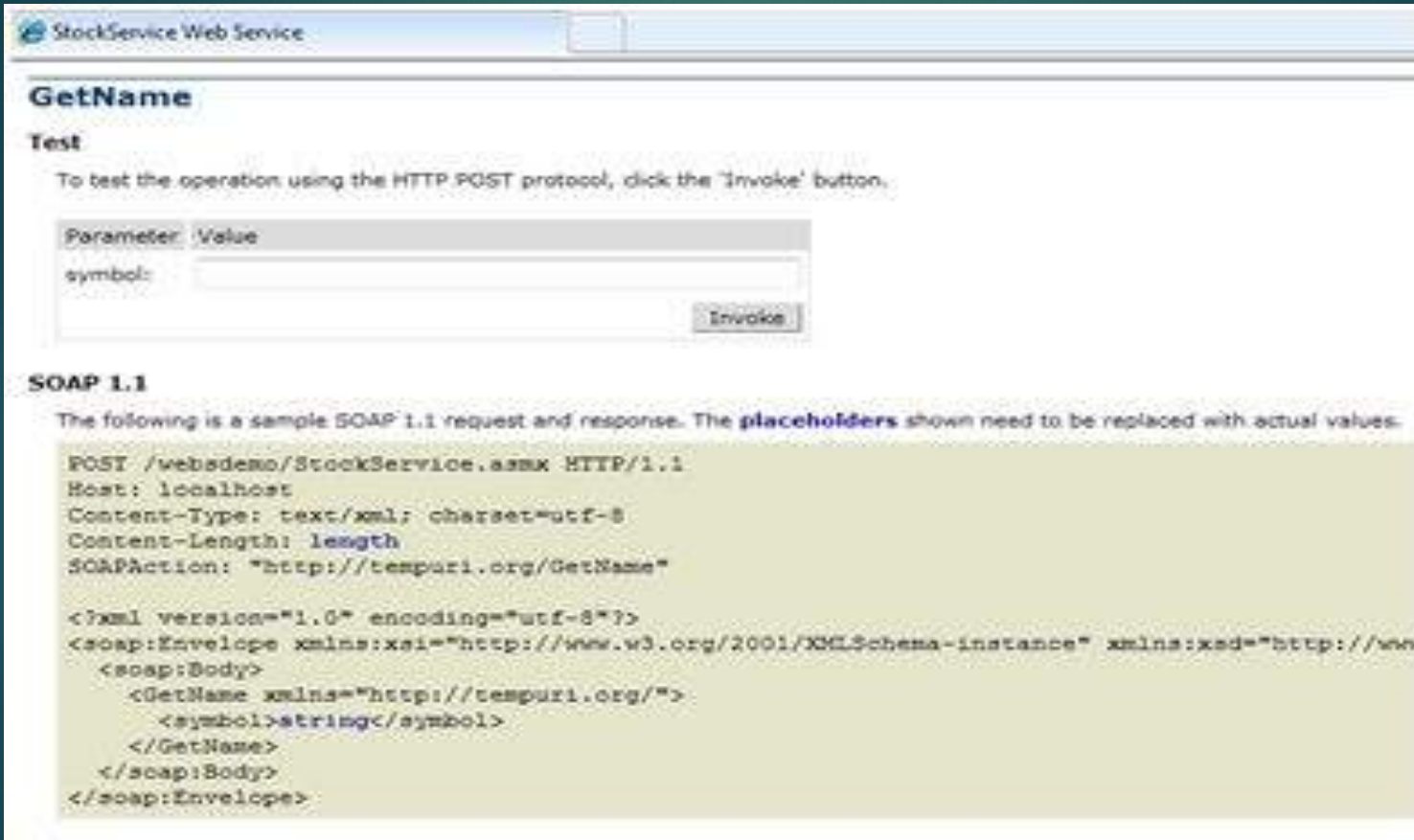
Each XML Web service needs a unique namespace in order for client applications to distinguish it from other services on the Web. <http://msdn.microsoft.com/en-us/library/aa304142.aspx> XML Web services should use a more permanent namespace.

Your XML Web service should be identified by a namespace that you control. For example, you can use your company's Internet domain FQDNs. They need not point to actual resources on the Web. (XML Web service namespaces are URIs.)

For XML Web services created using ASP.NET, the default namespace can be changed using the `WebService` attribute's `Namespace` property. Below is a code example that sets the namespace to "<http://microsoft.com/webservices/>":

```
C#  
[WebService(Namespace="http://microsoft.com/webservices/")]  
public class MyWebService {  
    // implementation  
}
```

Step (8) : Click on a method name, and check whether it runs properly.



The screenshot shows a web browser window titled "StockService Web Service". Below the title bar, there is a section for the "GetName" method. Under the heading "Test", a text box explains: "To test the operation using the HTTP POST protocol, click the 'Invoke' button." Below this, there is a table with two columns: "Parameter" and "Value". The first row has "symbol:" in the "Parameter" column and an empty text input field in the "Value" column. To the right of the input field is an "Invoke" button. Below the table, there is a section titled "SOAP 1.1" with a text box explaining: "The following is a sample SOAP 1.1 request and response. The placeholders shown need to be replaced with actual values." Below this text is a code block containing a sample SOAP 1.1 request.

**GetName**

**Test**

To test the operation using the HTTP POST protocol, click the 'Invoke' button.

Parameter	Value
symbol:	<input type="text"/>

**SOAP 1.1**

The following is a sample SOAP 1.1 request and response. The **placeholders** shown need to be replaced with actual values.


```
POST /webdemo/StockService.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/GetName"


<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www
  <soap:Body>
    <GetName xmlns="http://tempuri.org/">
      <symbol>string</symbol>
    </GetName>
  </soap:Body>
</soap:Envelope>
```




Step (9) : For testing the GetName method, provide one of the stock symbols, which are hard coded, it returns the name of the stock




- 
- ▶ For using the web service, create a web site under the same solution.
  - ▶ This could be done by right clicking on the Solution name in the Solution Explorer.
  - ▶ The web page calling the web service should have a label control to display the returned results and two button controls one for post back and another for calling the service.
  - ▶ The content file for the web application is as follows:

- 
- ▶ `<h3>Using the Stock Service</h3>`
  - ▶ `<br /> <br />`
  - ▶ `<asp:Label ID="lblmessage" runat="server"></asp:Label>`
  - ▶ `<br /> <br />`
  - ▶ `<asp:Button ID="btnpostback" runat="server"  
onclick="Button1_Click" Text="Post Back"  
style="width:132px" />`
  - ▶ `<asp:Button ID="btnservice" runat="server"  
onclick="btnservice_Click" Text="Get Stock"  
style="width:99px" />`

- 
- ▶ using System; using System.Collections; using System.Configuration; using System.Data; using System.Linq; using System.Web; using System.Web.Security; using System.Web.UI; using System.Web.UI.HtmlControls; using System.Web.UI.WebControls; using System.Web.UI.WebControls.WebParts; using System.Xml.Linq; //this is the proxy





- ▶ using localhost;
- ▶ namespace wsclient
- ▶ {
- ▶ public partial class \_Default : System.Web.UI.Page {
- ▶ protected void Page\_Load(object sender, EventArgs e)
- ▶ {
- ▶ if (!IsPostBack)
- ▶ {
- ▶ lblmessage.Text = "First Loading Time: " +  
DateTime.Now.ToLongTimeString } else { lblmessage.Text =  
"PostBack at: " + DateTime.Now.ToLongTimeString(); } }



- ▶ `protected void btnservice_Click(object sender, EventArgs e)`
- ▶ `{`
- ▶ `StockService proxy = new StockService();`
- ▶ `lblmessage.Text = String.Format("Current SATYAM Price:{0}", proxy.GetPrice("SATYAM").ToString());`
- ▶ `}`
- ▶ `}`
- ▶ `}`