Web Services

- Web services are a standardized way for developing interoperable applications.
- Web services use open standards and protocols like **HTTP**, **XML** and **SOAP.**
- Hyper Text Transfer Protocol (HTTP) is the protocl widely used by web services to send and receive messages.
- The messaging protocol is **SOAP.** SOAP stands for Simple Object Access Protocol. SOAP messages are in XML format.
- Webservice is a class that is decorated with **[WebService]** attribute and inherits from **System.Web.Services.WebService** base class.
- The **[WebService]** attribute tells that the this class contains the code for a web service.
- **WebService Namespace** is used to uniquely identify your web service on the internet from other services that are already there on the Web.
- WebService Namespace can be any string, but it is common to give it a company's internet domain name as they are usually unique. Something like [WebService(Namespace = "http://pragimtech.com/webservices")]
- It is not mandatory for a web service to inherit from **System.Web.Services.WebService** base class. However, if the web service has to use **ASP.NET session or application state** objects, then inheriting from **System.Web.Services.WebService** base class will provide direct access to these asp.net objects.
- To allow a web service to be called from Javascript, using ASP.NET AJAX, then decorate the web service class with **[System.Web.Script.Services.ScriptService]** attribute.
- What is WSDL and what is it's purpose?
  The **WSDL** document formally defines a web service. It contains

1. All the methods that are exposed by the web service
2. The parameters and their types
3. The return types of the methods

- How is a proxy class generated?

A proxy class is genrated using the **WSDL** (Web Service Description Language) document of the web service.

- What is the use of a proxy class?

The proxy class will then serialize the parameters, prepares a SOAP request message and sends it to the web service. The web service executes the method and returns a SOAP response message to the proxy.
The proxy class will then deserialize the SOAP response message and hands it the client application. We don't have to serialize or deserialize dot net CLR objects to and from SOAP format. The proxy class takes care of serialization and deserialization and makes the life of a developer much easier.

WebMethod attribute properties

- **BufferResponse -** This is a boolean property. Default is true. When this property is true, the response of the XML Web service method is not returned to the client until either the response is completely serialized or the buffer is full.
  On the other hand, when this property is false, the response of the XML Web service method is returned to the client as it is being serialized.
- **CacheDuration -** Use this property, if you want to cache the results of a web service method. This is an integer property, and specifies the number of seconds

that the response should be cached. The response is cached for each unique parameter.

WebMethod overloading in asp.net web services

- Method overloading allows a class to have multiple methods with the same name, but, with a different signature. So, in C# methods can be overloaded based on the number, type(int, float etc) and the kind(Value, Ref or Out) of parameters.
- **ebMethods** in a web service can also be **overloaded**. Notice that the **Add**() functions below are **overloaded based on the number of parameters.** The first method adds 2 numbers and the second one adds 3 numbers.

```
[WebMethod]
public int Add(int firstNumber, int secondNumber)
{
    return firstNumber + secondNumber;
}

[WebMethod]
public int Add(int firstNumber, int secondNumber, int thirdNumber)
{
    return firstNumber + secondNumber + thirdNumber;
}
```

When we build the solution, it builds successfully. But when we run the web service and try to view the service page we get an exception at runtime - Both Int32 Add(Int32, Int32, Int32) and Int32 Add(Int32, Int32) use the message name 'Add'. Use the MessageName property of the WebMethod custom attribute to specify unique message names for the methods.

- To fix this use **MessageName** property of the **WebMethod** attribute. MessageName property is used to uniquely identify the individual XML Web service methods.

```
[WebMethod(MessageName="Add2Numbers")]
public int Add(int firstNumber, int secondNumber)
{
    return firstNumber + secondNumber;
}
```

- Is method overloading possible in web services?
  Yes, use **MessageName** property of WebMethod attribute.

Calling asp.net web service from javascript using ajax

- Could you explain how an object-oriented application such as ASP.NET web application can make use of web services to access a relational database ?
  ASP.NET Web application calls the web service, and the web service has ado.net code to retrieve data from a relational database.
  Just like an asp.net web application, a web service can work with any type of datastore (Relational databases, xml file, text file, access, excel etc)
- **Where does ADO.NET and XML webservices come in architecture?**
  Most of the real time applications have **3 or more layers.** The most common layers in many of the applications are as follows
  1. User Interface of Presentation Layer - Contains only the UI logic
  2. Business Logic Layer - Contains logic to validate business rules

3. Data Access Layer - Performs the database CRUD operations

With respect to the above architecture
Web services belong to Business Logic Layer
ADO.NET belong to Data Access Layer

- **How to call a webservice from the web form without reloading the entire web page?**
  OR
  How to call a webservice without full page postback?
  Call the web service using ASP.NET AJAX, which allows partial page postback. With partial page postback, only specific portion of the page is updated without reloading the entire page which is better for performance and avoids screen flickers.