

# CertifyExpress



# MCAD/MCSD Exam 70-310—Developing XML Web Services and Server Components with Microsoft Visual Basic .NET and the Microsoft .NET Framework

## Abstract

This Exam Information Guide intends to provide you with information to prepare for the Microsoft MCAD/MCSD 70-310 Exam.

## What is MCAD?

*“The MCAD for Microsoft .NET credential is appropriate for professionals who use Microsoft technologies to develop and maintain department-level applications, components, Web or desktop clients, or back-end data services or work in teams developing enterprise applications.” (from Microsoft’s web site)*

MCAD candidates are required to pass two core exams and one elective exam in an area of specialization:

Core Exams (two required): Choose a language to leverage your existing skills through the .NET Framework. It can be either VB or C#. One exam focused on either Web Application Development or Windows Application Development, and another one for Web Services and Server Components.

## What is MCSD.Net?

MCSD.Net is the next generation of the MCSD certification. An entirely new set of requirements are outlined by Microsoft:

*“Microsoft Certified Solution Developer (MCSD) for Microsoft .NET candidates are required to pass four core exams and one elective exam that provide a valid and reliable measure of technical proficiency and expertise in developing and maintaining enterprise applications based on Microsoft development tools, technologies, and platforms. The elective exam provides proof of expertise with a specific Microsoft server product.”*

According to Microsoft, MCSD.NET is:

*"... appropriate for professionals who design and develop leading-edge enterprise solutions with Microsoft development tools, technologies, platforms, and the Microsoft .NET Framework. The MCSD job role includes analyzing business and technical requirements, and defining the solution architecture, as well as the tasks typically conducted by MCADs—implementing the requirements and building, deploying, and maintaining the solution."*

**The 70-310 Developing XML Web Services and Server Components with Microsoft Visual Basic .NET and the Microsoft .NET Framework exam can satisfy BOTH MCSD and MCAD.**

## Before you start

This study guide provides you with information on the many different aspects of "Microsoft MCAD/MCSD Exam 70-310". Before you proceed with this subject, please make sure you are 100% comfortable with the concept of networking, programming and development.

Do NOT rely solely on this study notes for the exam. By all means read more than one book on the subject and make sure you understand the material well enough so that you could be ready for the questions. There is no quick way to succeed for this topic. Ideally you must work things out and gain experience before even trying to sign up for the exam.

**Exam 310 tests the combination of VB.NET and XML, while exam 320 tests C# and XML.**

## Your Study Track for MCAD/MCSD 70-310

**1, Know the ins and outs of web authoring and programming using COM and VB.NET.**

The COM and COM+ Programming Primer -- by Alan Gordon; Paperback

Our Price: \$34.99 -- Or buy used from \$12.98

Applied Microsoft .NET Framework Programming -- by Jeffrey Richter; Paperback

Our Price: \$34.99 -- Or buy used from \$34.00

.NET and COM: The Complete Interoperability Guide  
by Adam Nathan (Paperback)

Programming Microsoft Visual Basic .NET (Core Reference) -- by Francesco Balena; Hardcover

.NET Enterprise Development in VB.NET: From Design to Deployment -- by Matt Reynolds, et al;  
Paperback

Professional VB.NET, 2nd Edition -- by Fred Barwell, et al; Paperback

**2, Make sure you are comfortable with the concept of web application development via XML.  
You will find the following books useful:**

Beginning XML -- by David Hunter, et al; Paperback

Learning XML

by Erik T. Ray (Paperback - February 2001)

Understanding Web Services: XML, WSDL, SOAP, and UDDI

by Eric Newcomer (Paperback)

### 3, Make sure you understand .NET thoroughly. Reference the following books:

Building Web Services and .NET Applications (Application Development)

by Lonnie Wall, Andrew Lader (Paperback)

Visual Studio .NET: The .NET Framework Black Book

by Julian Templeman, David Vitter (Paperback)

Fundamentals of Web Applications Using .Net and XML

by Eric Bell (Editor), et al (Paperback)

### 4, Review the exam objectives for 70-310:

- Creating and Managing Microsoft Windows Services, Serviced Components, .NET Remoting Objects, and XML Web Services
- Consuming and Manipulating Data
- Testing and Debugging
- Deploying Windows Services, Serviced Components, .NET Remoting Objects, and XML Web Services

## What is XML?

The Extensible Markup Language XML is:

*"... a specification developed by the W3C. XML is a pared-down version of SGML, designed especially for Web documents. It allows designers to create their own customized tags, enabling the definition, transmission, validation, and interpretation of data between applications and between organizations."*

<http://www.webopedia.com/TERM/X/XML.html>

**Remember, this 70-310 exam focuses on XML in the context of .NET.**

The XmlReader Class represents a reader that provides fast, non-cached, forward-only access to XML data.

XmlReader:

- can be advanced using any of the read methods and properties reflect the value of the current node - the current node refers to the node on which the reader is positioned
- provides read-only access to a stream of XML data.
- provides forward-only access to a stream of XML data.

XmlTextReader:

- fastest implementation of XmlReader
- checks for well-formed XML
- does not support data validation
- cannot expand general entities
- does not support default attributes

XmlValidatingReader:

- can validate data using DTDs or schemas
- can expand general entities
- supports default attributes

XmlNodeReader:

- reads XML data from an XmlNode

XmlReader class methods:

- Read XML content when the content is available in its entirety
- Find the depth of the XML element stack.
- Determine if an element is empty.
- Read and navigate attributes.
- Skip over elements and their content.

XmlReader class properties can return the following information:

- name of the current node.
- content of the current node.

XsltReader:

- another implementation that does not have a public constructor
- created as a result of calling the Transform method on the XsltTransform class
- functionality:
  - ◆ Enforces the rules that XML must be well-formed.
  - ◆ Does not validate against DTDs or schemas.
  - ◆ Does not expand default attributes.

## What is XPath?

*XPath is a language for addressing parts of an XML document, designed to be used by both XSLT and XPointer.*

<http://www.w3.org/TR/xpath>

According to W3C, XPath is the result of an effort to provide a common syntax and semantics for functionality shared between XSL Transformations [XSLT] and XPointer [XPointer]. The primary purpose of XPath is to address parts of an XML [XML] document. In support of this primary purpose, it also provides basic facilities for manipulation of strings, numbers and booleans. XPath uses a compact, non-XML syntax to facilitate use of XPath within URIs and XML attribute values. XPath operates on the abstract, logical structure of an XML document, rather than its surface syntax. XPath gets its name from its use of a path notation as in URLs for navigating through the hierarchical structure of an XML document. In addition to its use for addressing, XPath is also designed so that it has a natural subset that can be used for matching (testing whether or not a node matches a pattern); this use of XPath is described in XSLT.

XPath:

- models an XML document as a tree of nodes
- defines a way to compute a string-value for each type of node
- fully supports XML Namespaces

### You need to know how to use XPath to query XML data.

The XPathNavigator class:

- found in the System.Xml.XPath Namespace
- based on the data model described in the XML Path Language (XPath) 1.0 Recommendation
- provides the methods required to implement XPath queries over any data store
- reads data from any data store using a cursor model that allows forward and backwards movement
- functions:
  - ◆ useful for executing XPath queries
  - ◆ nodes can be selected from any store that implements the IXPathNavigable class
  - ◆ classes that already implement IXPathNavigable include XPathDocument and XmlNode, which XmlDocument and XmlDataDocument inherit from

## XPathNavigator

- Defines a read-only, random access cursor model over any store
- Supports the XPath data model
- Allows performant XSLT over any store that implements this abstract class

## XPathNodeIterator

- Iterates over a set of nodes selected by calling an XPath method

## XPathDocument

- Provides a fast and performant cache for XML document processing using XSLT.

## XPathExpression

- Encapsulates a compiled XPath expression.

## IXPathNavigable Interface

- Creates an XPathNavigator class
- Classes that implement it can provide the ability to create navigators using the CreateNavigator method

## What is XSD Schema?

The purpose of a schema is to define a class of XML documents. As said by Kurt Cagle at <http://www.vbxml.com/xml/articles/xsd/>,

*"Modern XML technology rests on three distinct legs: the XML specification itself, XSLT/XPath, and XML Schema. Schemas are significant not only in defining XML structures but also in providing data type capabilities to XML, adding a measure of object oriented programming support, and giving an infrastructure that can be used to support internationalization and personalization of presentation."*

There are currently a number of different schema language representations. They include:

- Document Type Definitions(DTD)
- XML Data Reduced(XDR)
- Simple Object XML(SOX)

MSDN defines the XML Schema definition language XSD as something that:

*“...enables you to define the structure and data types for XML documents. An XML Schema defines the elements, attributes, and data types that conform to the World Wide Web Consortium (W3C) XML Schema Part 1: Structures Recommendation for the XML Schema Definition Language. The W3C XML Schema Part 2: Datatypes Recommendation is the recommendation for defining data types used in XML schemas. The XML Schema Reference (XSD) is based on the W3C 2001 Recommendation specifications for Datatypes and for Structures.”*

The schema element contains:

- type definitions – simpleType and complexType
- attribute
- element declarations
  - allows for the definition of new data types using the simpleType and complexType elements

simpleType

- can be used as the content (textOnly) of an element or attribute.
- cannot contain elements or have attributes.

complexType

- for elements that can contain attributes and elements
- can contain elements
- can have attributes

XML Schema data types and their corresponding .NET Framework type support (extracted from MSDN for your reference):

XML Schema (XSD) type	.NET Framework type
anyURI	System.Uri
base64Binary	System.Byte[]
Boolean	System.Boolean
Byte	System.SByte
Date	System.DateTime
dateTime	System.DateTime
decimal	System.Decimal
Double	System.Double
duration	System.TimeSpan
ENTITIES	System.String[]
ENTITY	System.String
Float	System.Single
gDay	System.DateTime
gMonthDay	System.DateTime
gYear	System.DateTime
gYearMonth	System.DateTime
hexBinary	System.Byte[]
ID	System.String
IDREF	System.String
IDREFS	System.String[]
int	System.Int32
integer	System.Decimal
language	System.String
long	System.Int64
month	System.DateTime
Name	System.String
NCName	System.String
negativeInteger	System.Decimal
NMTOKEN	System.String
NMTOKENS	System.String[]
nonNegativeInteger	System.Decimal
nonPositiveInteger	System.Decimal
normalizedString	System.String
NOTATION	System.String
positiveInteger	System.Decimal
QName	System.Xml.XmlQualifiedName
short	System.Int16
string	System.String
time	System.DateTime
timePeriod	System.DateTime
token	System.String
unsignedByte	System.Byte
unsignedInt	System.UInt32
unsignedLong	System.UInt64
unsignedShort	System.UInt16

## What is SOM?

- SOM = XML Schema Object Model
- An XML schema is a powerful and complex tool for providing a way to define the structure of XML documents, by specifying the elements that can be used in the documents, as well as the structure and types that these elements must follow in order to be valid for that specific schema.
- A schema is typically with an .xsd file name extension to describe the contents of XML documents using valid XML: elements and attributes
- Schemas provide several advancements over document type definitions (DTDs):
  - ◆ Additional data types are available
  - ◆ Can create custom data types.
  - ◆ Uses XML syntax.
  - ◆ Supports object-oriented concepts such as polymorphism and inheritance.
- provides a set of classes in the System.Xml.Schema namespace that allow you to:
  - ◆ read a schema from a file
  - ◆ programmatically create a schema in memory that can be compiled and validated or written to a file.

## What is SOAP?

The Simple Object Access Protocol SOAP:

*"...provides a way for applications to communicate with each other over the Internet, independent of platform. Unlike OMG's IIOP, SOAP piggybacks a DOM onto HTTP (port 80) in order to penetrate server firewalls, which are usually configured to accept port 80 and port 21 (FTP) requests."*

<http://www.webopedia.com/TERM/S/SOAP.html>

The relationship between SOAP and XML - SOAP relies on XML to define the format of the information and then adds the necessary HTTP headers to send it.

## What is DOM?

The Document Object Model, the specification for how objects in a Web page. It:

- defines what attributes are associated with each object
- defines how the objects and attributes can be manipulated
- defines how to dynamically change the appearance of Web pages

You need to know how to access an XML file by using DOM and an XmlReader.

# What is .NET?

According to webopedia.com,

*"It is a Microsoft operating system platform that incorporates applications, a suite of tools and services and a change in the infrastructure of the company's Web strategy. The objective of .NET is to bring users into the next generation of the Internet by conquering the deficiencies of the first generation and giving users a more enriched experience in using the Web for both personal and business applications. This is Microsoft's most ambitious undertaking since the release of Windows 3.0."*

The four main principles of .NET are:

- erasing boundaries between applications and the Internet.
- allowing software rental as a hosted service over the Internet.
- allowing user access to their information on the Internet from any device, anytime, anywhere.
- introducing new ways to interact with application data.

Common Language Runtime:

- CLR
- forms the foundation of the .NET Framework
- manages the execution of .NET code
- verifies that code is type-safe
- enforces Code Access Security
- ensure version compatibility between application components
- manages memory and threads.
- includes the just-in-time compilers
- all code created to target .NET is referred to as "managed code"

Microsoft Intermediate Language:

- MSIL
- CPU-independent
- tokenized set of instructions
- when executed, invoke the CLR

.NET compilers:

- VB.NET
- C++ with Managed Extension
- JScript.NET
- C#

## Common Type System:

- CTS
- defines a subset of common data types
- defines how new data types can be declared
- allows all languages to share a basic set of types and rules for extending types
- allows .NET to support cross language integration

## Assemblies:

- building blocks of .NET applications
- portable executable DLL or EXE
- self-describing with the help of meta data
- contain the following elements:
  - ◆ PE Header
  - ◆ Unmanaged stub
  - ◆ Meta data
  - ◆ MSIL code
  - ◆ Resources

## .NET Framework Class Library

- an object-oriented library of classes and other types
- comes with the .NET Framework SDK
- classes are organized hierarchically in namespaces
- supports all .NET applications, regardless of the language or the application type
- provides access to system functionality
- implements the classes upon which .NET Web Services are built

**This exam tests heavily on .NET remoting object.**

## What is .NET Remoting?

MSDN describes .NET Remoting as a framework that

*"... provides a rich and extensible framework for objects living in different AppDomains, in different processes, and in different machines to communicate with each other seamlessly. .NET Remoting offers a powerful yet simple programming model and runtime support for making these interactions transparent. In this article we will take a look at the different building blocks of the Remoting architecture, as well as explore some of the common scenarios in which .NET Remoting can be leveraged."*

## .NET Remoting Objects:

- Single Call
  - ◆ service one and only one request coming in
  - ◆ useful in scenarios where the objects are required to do a finite amount of work
  - ◆ usually not required to store state information
  - ◆ cannot hold state information between method calls
  - ◆ can be configured in a load-balanced fashion
  
- Singleton Objects
  - ◆ service multiple clients
  - ◆ share data by storing state information between client invocations
  - ◆ useful in cases in which data needs to be shared explicitly between clients
  - ◆ useful in which the overhead of creating and maintaining objects is substantial
  
- CAO
  - ◆ Client-activated objects
  - ◆ server-side objects
  - ◆ activated upon request from the client
  - ◆ When the client submits a request for a server object using "new" operator, an activation request message is sent to the remote application
  
- Passing Objects with .NET Remoting:
  - ◆ As parameters in method calls
  - ◆ Return Value of method calls
  - ◆ Values resulting from property
  - ◆ Values resulting from field access of a .NET component
    - objects that are Marshal By Value (MBV) - complete copy of the object is made when the object is passed
    - objects that are Marshal By Reference (MBR) - reference to the object is made when passing

## Creating and testing a XML Web Service Application in VB.NET – the steps:

\* [Information in the section is compiled from material in KB article 3090138](#)

Create a new XML web service:

- Invoke Visual Studio.NET. On the File menu, click New and then click Project.
- Under Project types click Visual Basic Projects , then click ASP.NET Web Service under Templates . Name the project TestService.
- In Solution Explorer, change the name of Service1.asmx to Services.asmx.
- Open Services.asmx in the visual designer. In the Properties window, change the Name property of the Service1 class to Services.
- Save the project.

Create the XML Web Service Methods:

- Open Services.asmx in the code editor.
- Add the following code within the Services class definition to create various Web methods (code extracted from Microsoft web site at <http://support.microsoft.com/directory/article.asp?ID=kb;en-us:Q309013&SD=MSDN> ):

```
<WebMethod()> Public Function GetMessage() As String
    Return "Today is the day"
End Function
```

```
<WebMethod()> _
Public Function SendMessage(ByVal message As String) As String
    Return "Message received as: " & message
End Function
```

```
<WebMethod()> _
Public Function ReverseMessageFunction(ByVal message As String) As String
    Return StrReverse(message)
End Function
```

```
<WebMethod()> Public Sub ReverseMessageSub(ByRef message As String)
    message = StrReverse(message)
End Sub
```

- Save and build the project.

Test the Services with Visual Studio .NET:

- In Solution Explorer, right-click Services.asmx and then click View in Browser.
- Click the hyperlink for the method that you want to test.
- Fill in any requested message parameter values.
- Click Invoke.
- View the resulting XML and close the results window.
- Click the Back button to return to the method list and repeat the steps for the remaining Web methods.
- Close the built-in browser.

Create the Test Client Application:

- On the File menu, click Add Project, and then click New Project .
- Select Visual Basic Console Application , and then name the project TestHarness.
- On the Project menu, click Add Web Reference .
- In the Address field, type `http://localhost/TestService/Services.asmx` , and then click Go .
- Click Add Reference to finish creating the Web reference.
- In Solution Explorer, right-click localhost in the Web References folder, click Rename , and then change the name to WebService. This becomes the namespace that is used within the test application to refer to the Services class.

Create the Test Code:

- Open Module1.vb and locate the Sub Main procedure.
- Paste the following code in the file to call the appropriate Web methods:

```
Dim strValue As String = "This is my test message"  
Dim myService As New WebService.Services()  
Console.WriteLine(myService.GetMessage)  
Console.WriteLine(myService.SendMessage(strValue))  
Console.WriteLine(myService.ReverseMessageFunction(strValue))  
myService.ReverseMessageSub(strValue)  
Console.WriteLine(strValue)
```

(code extracted from Microsoft web site at

<http://support.microsoft.com/directory/article.asp?ID=kb;en-us;Q309013&SD=MSDN>

Test the Client Application:

- Create a breakpoint on the following line:  
*Console.WriteLine(myService.GetMessage)*
- In Solution Explorer, right-click the TestHarness project, and then click Set as StartUp Project .
- On the Debug menu, click Start and wait for the program to enter debug mode.
- On the Debug menu, click Windows , and then click Locals . Use the Locals window to view the value of the strValue variable during the debugging to observe any changes that are made to the variable.
- On the Debug toolbar, use Step Into to step through each line of code from the TestHarness client into the XML Web service.
- Confirm that the output in the console window is as expected.
- When the program ends, remove the breakpoint and close Visual Studio .NET.

