Biyani Institute of Science & Mangement

Model Paper-2019

MCA-402-A .Net Framework and Programming in ASP.Net

1. Answer each part in one line **10*1=10**

a) What is MSIL?

b) Define CLR

c) What is adRotator?

d) Define Tree View control in asp.net

e) What is caching?

f) What is localization & Globalization?

g) What is postback? How to use it.

h) Define UDDI

i) What is SOAP protocol?

j) Name the classes belongs to disconnected architecture of ADO.NET

2. Answer each part in 50 words **5*3=15**

a) What is WCF Explain

b) Explain Web Services

c) Why authentication and authorization is needed?

d) What is Namespace? Explain

e) What is Boxing and Unoxing?

3. Answer each part in 150 words **5*4=20**

a) Explain Exception Handling in asp.net

b) Explain dataproviders in asp.net

c) How to create crystal report? Explain.

d) What is custom control and user control? Explain

e) Explain any four standard control in asp.net

4. **2*10=20**

a) Explain Rich web controls in detail.

b) Explain ADO.NET in details

5. What is WCF and Web service Explain? Write down the difference between them. **15**

OR

5. Explain Session and application state management. **15**

Solution

Ans 1. One Line answers

a) **What is MSIL?**

MSIL stands for Microsoft Intermediate Language. We can call it as Intermediate Language (IL) or Common Intermediate Language (CIL). During the compile time , the compiler convert the source code into Microsoft Intermediate Language (MSIL) .Microsoft Intermediate Language (MSIL) is a CPU-independent set of instructions that can be efficiently converted to the native code.

b) **Define CLR**

The Common Language Runtime (**CLR**), the virtual machine component of Microsoft's .NET framework, manages the execution of .NET programs. Just-in-time compilation converts the managed code (compiled intermediate language code), into machine instructions which are then executed on the CPU of the computer.

c) **What is adRotator?**

The **AdRotator** is one of the rich web server control of **asp**.**net**. **AdRotator** control is used to display a sequence of advertisement images as per given priority of image. **Adrotator** control display the sequence of images, which is specified in the external XML file.

d) **Define Tree View control in asp.net**

The TreeView Web control is useful to display hierarchical data in a tree structure. A TreeView is a collection of TreeNode objects.

e) **What is caching?**

Caching technique allows to store/**cache** page output or application data on the client. - The **cached** information is used to serve subsequent requests that avoid the overhead of recreating the same information.

f) **What is localization & Globalization?**

Globalization is the process of designing the application in such a way that it can be used by users from across the globe (multiple cultures). **Localization**, on the other hand, is the process of customization to make our application behave as per the current culture and locale.

g) **What is postback? How to use it.**

PostBack is the name given to the process of submitting an**ASP**.**NET** page to the server for processing. **PostBack** is done if certain credentials of the page are to be checked against some sources (such as verification of username and password using database).

h) Define UDDI

UDDI is an XML-based standard for describing, publishing, and finding web services. **UDDI** stands for Universal Description, Discovery, and Integration. ...**UDDI** is an open industry initiative, enabling businesses to discover each other and define how they interact over the Internet.

i) What is SOAP protocol?

SOAP is an XML-based protocol for accessing web services over HTTP. ... **SOAP** is known as the Simple Object Access Protocol, but in later times was just shortened to **SOAP** v1.2. **SOAP** is a protocol or in other words is a definition of how web services talk to each other or talk to client applications that invoke them.

j) Name the classes belongs to disconnected architecture of ADO.NET

- DataSet Class
- DataTable Class
- DataColumn Class
- DataRow Class

2. 50 Words Answers

a) What is WCF Explain

WCF stands for Windows Communication Foundation. It is basically used to create a distributed and interoperable Application. WCF Applications came into the picture in .Net 3.0 Framework. This is a framework, which is used for creating Service oriented Applications. You can send the data asynchronously from one end point to another. I think you all know about Web Service and are thinking, if we already have Web Services and accessing on a remote basis then why did WCF come into picture?

There are so many reasons for it but now I am giving you an answer in two words- one is protocol and the other is hosting. Thus, I will explain these two words later in this article.

Distributed Application

It means those Applications, which do not run only on single system but can run on multiple systems, which are connected over the network. For example, a Web Service that can consume by different clients.

Interoperable

It means that an Application can consume or connect with another Application but it does not matter in which platform it is developed.

b) Explain Web Services

A web service is any piece of software that makes itself available over the internet and uses a standardized XML messaging system. XML is used to encode all communications to a web service. For example, a client invokes a web service by sending an XML message, then waits for a corresponding XML response. As all communication is in XML, web services are not tied to any one operating system or programming language—Java can talk with Perl; Windows applications can talk with Unix applications.

Web services are self-contained, modular, distributed, dynamic applications that can be described, published, located, or invoked over the network to create products, processes, and supply chains. These applications can be local, distributed, or web-based. Web services are built on top of open standards such as TCP/IP, HTTP, Java, HTML, and XML.

Web services are XML-based information exchange systems that use the Internet for direct application-to-application interaction. These systems can include programs, objects, messages, or documents.

A web service is a collection of open protocols and standards used for exchanging data between applications or systems. Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks like the Internet in a manner similar to inter-process communication on a single computer.

c) Why authentication and authorization is needed?

Authentication is the process of validating the identity of a user to allow or deny the user request. It involves accepting user credentials from the user and validating against the user. The system checks for the verification the user is legal for the system.

When the user request for specific request it comes to the IIS. The IIS authenticates the user requesting the resource. The process will decide whether to impersonate the authenticated user by the IIS or not. If the impersonation is enabled, the configuration settings in the Web.config file, the ASP.NET server impersonates the authenticated user.

Authorization is the process of ensuring users with valid identity are allowed to access the specific resources in the system. Depending on the roles assign to the user the authorization is provided. If the user has Administrator authorization, access throughout the application is possible. It can create or delete users depending on the requirement.

d) What is Namespace? Explain

Interfaces, Enumerations, Delegates etc.), example System.IO logically groups input output related features , System.Data.SqlClient is the logical group of ado.net Connectivity with Sql server related features. In Object Oriented world, many times it is possible that programmers will use the same class name, Qualifying NameSpace with class name can avoid this collision. Namespaces allow you to create a system to organize your code. A good way to organize your namespaces is via a hierarchical system. You put the more general names at the top of the hierarchy and get more specific as you go down. This hierarchical system can be represented by nested namespaces. Bellow shows how to create a nested namespace. By placing code in different sub-namespaces, you can keep your code organized.

e) What is Boxing and Unoxing?

C# Type System contains three Types , they are Value Types , Reference Types and Pointer Types. C# allows us to convert a Value Type to a Reference Type, and back again to Value Types . The operation of Converting a Value Type to a Reference Type is called Boxing and the reverse operation is called Unboxing.

Boxing

```
int Val = 1;
Object Obj = Val; //Boxing
```

The first line we created a Value Type Val and assigned a value to Val. The second line , we created an instance of Object Obj and assign the value of Val to Obj. From the above operation (Object Obj = i) we saw converting a value of a Value Type into a value of a corresponding Reference Type . These types of operation is called Boxing.

UnBoxing

```
int Val = 1;
Object Obj = Val; //Boxing
int i = (int)Obj; //Unboxing
```

The first two line shows how to Box a Value Type . The next line (int i = (int) Obj) shows extracts the Value Type from the Object . That is converting a value of a Reference Type into a value of a Value Type. This operation is called UnBoxing.

Boxing and UnBoxing are computationally expensive processes. When a value type is boxed, an entirely new object must be allocated and constructed , also the cast required for UnBoxing is also expensive computationally.

3. 150 words answer

a) **Explain Exception Handling in asp.net**

Exception Class

The Exception class is the base class from which exceptions inherit. Most exception objects are instances of some derived class of the Exception class, such as the SystemException class, the IndexOutOfRangeException class, or the ArgumentNullException class. The Exception class has properties, such as the StackTrace property, the InnerException property, and the Message property, that provide specific information about the error that has occurred.

Exception Inheritance Hierarchy

The runtime has a base set of exceptions deriving from the SystemException class that the runtime throws when an exception is encountered. Most of the classes that inherit from the Exception class, such as the IndexOutOfRangeException class and the ArgumentNullException class, do not implement additional members. Therefore, the most important information for an exception can be found in the hierarchy of exceptions, the exception name, and the information contained in the exception.

Exception Handling Hierarchy

In an ASP.NET Web Forms application, exceptions can be handled based on a specific handling hierarchy. An exception can be handled at the following levels:

Application level

Page level

Code level

Application Level Error Handling

You can handle default errors at the application level either by modifying your application's configuration or by adding an Application_Error handler in the Global.asax file of your application.

You can handle default errors and HTTP errors by adding a customErrors section to the Web.config file. The customErrors section allows you to specify a default page that users will be redirected to when an error occurs. It also allows you to specify individual pages for specific status code errors.

However, you can trap errors that occur anywhere in your application by adding code to the Application_Error handler in the Global.asax file.

Page Level Error Event Handling

A page-level handler returns the user to the page where the error occurred, but because instances of controls are not maintained, there will no longer be anything on the page. To provide the error details to the user of the application, you must specifically write the error details to the page.

You would typically use a page-level error handler to log unhandled errors or to take the user to a page that can display helpful information.

This code example shows a handler for the Error event in an ASP.NET Web page. This handler catches all exceptions that are not already handled within try/catch blocks in the page.

After you handle an error, you must clear it by calling the ClearError method of the Server object (HttpServerUtility class), otherwise you will see an error that has previously occurred.

Code Level Error Handling

The try-catch statement consists of a try block followed by one or more catch clauses, which specify handlers for different exceptions. When an exception is thrown, the common language runtime (CLR) looks for the catch statement that handles this exception. If the currently executing method does not contain a catch block, the CLR looks at the method that called the current method, and so on, up the call stack. If no catch block is found, then the CLR displays an unhandled exception message to the user and stops execution of the program.

b) Explain dataproviders in asp.net

ADO.NET Data Providers

The classes responsible for the movement of data between the disconnected data classes in the client application and the data store are referred to as connected classes or provider classes.

The ADO.NET Framework comes with the following providers:

OLEDB: The OLEDB provider, expressed through the System.Data.OleDb namespace. You can use this provider to access SQL Server 6.5 and earlier, SyBase, DB2/400, and Microsoft Access.

ODBC: The ODBC provider, expressed through the System.Data.Odbc namespace. This provider is typically used when no newer provider is available.

SQL Server: The Microsoft SQL Server provider, expressed through the System.Data.SqlClient namespace. It Contains classes that provide functionality similar to the generic OleDb provider. The difference is that these classes are tuned for SQL Server 7 and later data access.

Prior to .NET Framework 4, Microsoft also included a functional Oracle provider with the .NET Framework. However, its classes have been marked as deprecated and obsolete in .NET Framework 4.

Core objects that make up a .NET Framework data provider:

Connection : Establishes a connection to a specific data source. The base class for all Connection objects is the DbConnection class.

Command : Executes a command against a data source. Exposes Parameters and can execute in the scope of a Transaction from a Connection. The base class for all Command objects is the DbCommand class.

DataReader : Reads a forward-only, read-only stream of data from a data source. The base class for all DataReader objects is the DbDataReader class.

DataAdapter : Populates a DataSet and resolves updates with the data source. The base class for all DataAdapter objects is the DbDataAdapter class.

c) How to create crystal report? Explain.

Open Visual Studio .NET and select a new CSharp Windows project.

Now you will get the default Form1.cs.

From the main menu in Visual Studio C# project select PROJECT-->Add New Item . Then Add New Item dialogue will appear and select Crystal Reports from the dialogue box.

Select Report type from Crystal Reports gallery.

Accept the default settings and click OK.

Next step is to select the appropriate connection to your database (here crstaldb). Here we are going to select OLEDB Connection for SQL Server to connect Crystal Reports in C#.

Select OLE DB (ADO) from Create New Connection.

csharp-crystal-reports-oledb

Select Microsoft OLE DB Provider for SQL Server.

csharp-crystal-reports-oledb-provider

The next screen is the SQL Server authentication screen for connecting to the database - crystalDB. Select your Sql Server name , enter userid , password and select your Database Name.

csharp-crystal-reports-oledb-authentication

Click next , Then the screen shows OLE DB Property values , leave it as it is , and then click finish button.

csharp-crystal-reports-oledb-finish

After you click the finish button , the next window you will get your Server name under OLEDB Connection, from there selected database name (Crystaldb) and click the tables , then you can see all your tables from your database.

From the tables list double click the Product table then you can see the Product table will come in the right side list.

csharp-crystal-reports-product

Click Next Button

Select all fields from Product table to the right side list .

csharp-crystal-reports-all-tables

Click Finish Button. Then you can see the Crystal Reports designer window in your C# project. In the Crystal Reports designer window you can see the selected fields from Product table. You can arrange the field Objects and design of the screen according your requirements.

csharp-crystal-report-designer-window

Now the designing part is over and the next step is to call the Crystal Reports in your C# application and view it through Crystal Reports Viewer control in C#.

Select the default form (Form1.cs) you created in C# and drag a button and a CrystalReportViewer control to your form .

csharp-crystal-reports-viewer

After you drag the CrystalReportViewer to your form , it will look like the following picture.

csharp-crystal-reports-form

You have to include CrystalDecisions.CrystalReports.Engine in your C# Source Code.

```
using CrystalDecisions.CrystalReports.Engine;
```

d) What is custom control and user control? Explain

User Controls

User controls behaves like miniature ASP.NET pages or web forms, which could be used by many other pages. These are derived from the System.Web.UI.UserControl class. These controls have the following characteristics:

They have an .ascx extension.

They may not contain any <html>, <body>, or <form> tags.

They have a Control directive instead of a Page directive.

To understand the concept, let us create a simple user control, which will work as footer for the web pages. To create and use the user control, take the following steps:

Create a new web application.

Right click on the project folder on the Solution Explorer and choose Add New Item. Add New Item

Select Web User Control from the Add New Item dialog box and name it footer.ascx. Initially, the footer.ascx contains only a Control directive.

(1) The Register directive specifies a tag name as well as tag prefix for the control.

<%@ Register Src="~/footer.ascx" TagName="footer" TagPrefix="Tfooter" %>

(2) The following tag name and prefix should be used while adding the user control on the page:

<Tfooter:footer ID="footer1" runat="server" />

Custom Controls

Custom controls are deployed as individual assemblies. They are compiled into a Dynamic Link Library (DLL) and used as any other ASP.NET server control. They could be created in either of the following way:

By deriving a custom control from an existing control

By composing a new custom control combing two or more existing controls.

By deriving from the base control class.

To understand the concept, let us create a custom control, which will simply render a text message on the browser. To create this control, take the following steps:

Create a new website. Right click the solution (not the project) at the top of the tree in the Solution Explorer.

In the New Project dialog box, select ASP.NET Server Control from the project templates.

project templates

The above step adds a new project and creates a complete custom control to the solution, called ServerControl1. In this example, let us name the project CustomControls. To use this control, this must be added as a reference to the web site before registering it on a page. To add a reference to the existing project, right click on the project (not the solution), and click Add Reference.

Select the CustomControls project from the Projects tab of the Add Reference dialog box. The Solution Explorer should show the reference.

Custom Controls

To use the control on a page, add the Register directive just below the @Page directive:

```
<%@ Register Assembly="CustomControls"  Namespace="CustomControls"  TagPrefix="ccs" %>
```

Further, you can use the control, similar to any other controls.

```
<form id="form1" runat="server">
  <div>
    <ccs:ServerControl1 runat="server" Text = "I am a Custom Server Control" />
  </div>
</form>
```

e) Explain any four standard control in asp.net

Button Controls

ASP.NET provides three types of button control:

Button : It displays text within a rectangular area.

Link Button : It displays text that looks like a hyperlink.

Image Button : It displays an image.

When a user clicks a button, two events are raised: Click and Command.

Basic syntax of button control:

```
<asp:Button ID="Button1" runat="server" onclick="Button1_Click" Text="Click" / >
```

Common properties of the button control:

Property	Description
Text	The text displayed on the button. This is for button and link button controls only.
ImageUrl	For image button control only. The image to be displayed for the button.
AlternateText	For image button control only. The text to be displayed if the browser cannot display the image.
CausesValidation	Determines whether page validation occurs when a user clicks the button. The default is true.
CommandName	A string value that is passed to the command event when a user clicks the button.
CommandArgument	A string value that is passed to the command event when a user clicks the button.
PostBackUrl	The URL of the page that is requested when the user clicks the button.

Text Boxes and Labels

Text box controls are typically used to accept input from the user. A text box control can accept one or more lines of text depending upon the settings of the TextMode attribute.

Label controls provide an easy way to display text which can be changed from one execution of a page to the next. If you want to display text that does not change, you use the literal text.

Basic syntax of text control:

<asp:TextBox ID="txtstate" runat="server" ></asp:TextBox>

Common Properties of the Text Box and Labels:

Property Description

TextMode Specifies the type of text box. SingleLine creates a standard text box, MultiLIne creates a text box that accepts more than one line of text and the Password causes the characters that are entered to be masked. The default is SingleLine.

Text The text content of the text box.

MaxLength The maximum number of characters that can be entered into the text box.

Wrap It determines whether or not text wraps automatically for multi-line text box; default is true.

ReadOnly Determines whether the user can change the text in the box; default is false, i.e., the user can not change the text.

Columns The width of the text box in characters. The actual width is determined based on the font that is used for the text entry.

Rows The height of a multi-line text box in lines. The default value is 0, means a single line text box.

The mostly used attribute for a label control is 'Text', which implies the text displayed on the label.

Check Boxes and Radio Buttons

A check box displays a single option that the user can either check or uncheck and radio buttons present a group of options from which the user can select just one option.

To create a group of radio buttons, you specify the same name for the GroupName attribute of each radio button in the group. If more than one group is required in a single form, then specify a different group name for each group.

If you want check box or radio button to be selected when the form is initially displayed, set its Checked attribute to true. If the Checked attribute is set to true for multiple radio buttons in a group, then only the last one is considered as true.

Basic syntax of check box:

<asp:CheckBox ID= "chkoption" runat= "Server">

</asp:CheckBox>

Basic syntax of radio button:

<asp:RadioButton ID= "rdboption" runat= "Server">

</asp: RadioButton>

Common properties of check boxes and radio buttons:

Property	Description
Text	The text displayed next to the check box or radio button.
Checked	Specifies whether it is selected or not, default is false.
GroupName	Name of the group the control belongs to.

Basic syntax for an image control:

<asp:Image ID="Image1" runat="server">

It has the following important properties:

Property	Description
AlternateText	Alternate text to be displayed in absence of the image.
ImageAlign	Alignment options for the control.
ImageUrl	Path of the image to be displayed by the control.

Ans 4. Long Answer Type questions

a) **Explain Rich web controls in detail.**

Rich Controls

ASP.NET provides large set of controls. These controls are divided into different categories, depends upon their functionalities. The followings control comes under the rich controls category.

FileUpload control is used to browse and upload files. After the file is uploaded, you can store the file on any drive or database. FileUpload control is the combination of a browse button and a text box for entering the filename.

The FileUpload control supports the following important properties.

FileBytes: It returns the contents of uploaded file as a byte array

FileContent: You can get the uploaded file contents as a stream.

FileName: Provides the name of uploaded file.

Calendar control

Calendar control provides you lots of property and events. By using these properties and events you can perform the following task with calendar control.

Select date.

Selecting a day, a week or a month.

Customize the calendar's appearance.

The Calendar control supports three important events:

Event Description

SelectionChanged This event is fired when you select a day, a week or an entire month.

DayRender This event is fired when each data cell of the calendar control is rendered.

VisibleMonthChangedIt is raised when user changes a month.

AdRotator control

AdRotator control is used to display different advertisements randomly in a page. The list of advertisements is stored in either an XML file or in a database table. Lots of websites uses AdRotator control to display the advertisements on the web page.

To create an advertisement list, first add an XML file to your project.

Important properties of AdRotator control.

ImageUrl: The URL of the image that will be displayed through AdRotator control.

NavigateUrl: If the user clicks the banner or ad then the new page is opened according to given URL.

AlternateText: It is used for displaying text instead of the picture if picture is not displayed. It is also used as a tooltip.

Impressions: It is a number that sets how frequently an advertisement will appear.

Keyword: It is used to filter ads or identifies a group of advertisement.

MultiView control

MultiView control can be used when you want to create a tabbed page. In many situations, a web form may be very long, and then you can divide a long form into multiple sub forms. MultiView control is made up of multiple view controls. You can put multiple ASP.NET controls inside view controls. One View control is displayed at a time and it is called as the active view. View control does not work separately. It is always used with a Multiview control.

Wizard Control

This control is same as MultiView control but the main difference is that, it has inbuilt navigation buttons.

The wizard control enables you to design a long form in such a way that you can work in multiple sub form. You can perform the task in a step by step process. It reduces the work of developers to design multiple forms. It enables you to create multi step user interface. Wizard control provides with built-in previous/next functionality.

The Wizard control can contains one or more WizardStep as child controls. Only one WizardStep is displayed at a time. WizardStep control has an important property called as StepType. The StepType property determines the type of navigation buttons that will be displayed for that step. The possible values are:

The StepType associated with each WizardStep determines the type of navigation buttons that will be displayed for that step. The StepTypes are:

Start:

Step:

Finish:

Complete:

Auto:

Drag the Wizard control on the web page from toolbox, you will get the following code.

```
<asp:Wizard ID="Wizard1" runat="server" Height="75px" Width="140px">
    <WizardSteps>
        <asp:WizardStep runat="server" title="Step 1">
        </asp:WizardStep>
        <asp:WizardStep runat="server" title="Step 2">
        </asp:WizardStep>
    </WizardSteps>
</asp:Wizard>
```

You can put WizardStep according to application need.

Important events of Wizard control are as follows:

ActiveStepChanged:

CancelButtonClick:

FinishButtonClick:

NextButtonClick:

PreviousButtonClick:

b) Explain ADO.NET in details

ADO.NET provides a bridge between the front end controls and the back end database. The ADO.NET objects encapsulate all the data access operations and the controls interact with these objects to display data, thus hiding the details of movement of data.

The following figure shows the ADO.NET objects at a glance:

ADO.Net objects

The DataSet Class

The dataset represents a subset of the database. It does not have a continuous connection to the database. To update the database a reconnection is required. The DataSet contains DataTable objects and DataRelation objects. The DataRelation objects represent the relationship between two tables.

Following table shows some important properties of the DataSet class:

Properties Description

CaseSensitive Indicates whether string comparisons within the data tables are case-sensitive.

Container Gets the container for the component.

DataSetName Gets or sets the name of the current data set.

DefaultViewManager Returns a view of data in the data set.

DesignMode Indicates whether the component is currently in design mode.

EnforceConstraints Indicates whether constraint rules are followed when attempting any update operation.

Events Gets the list of event handlers that are attached to this component.

ExtendedProperties Gets the collection of customized user information associated with the DataSet.

HasErrors Indicates if there are any errors.

IsInitialized Indicates whether the DataSet is initialized.

Locale Gets or sets the locale information used to compare strings within the table.

Namespace Gets or sets the namespace of the DataSet.

Prefix Gets or sets an XML prefix that aliases the namespace of the DataSet.

Relations Returns the collection of DataRelation objects.

Tables Returns the collection of DataTable objects.

The following table shows some important methods of the DataSet class:

Methods Description

AcceptChanges Accepts all changes made since the DataSet was loaded or this method was called.

BeginInit Begins the initialization of the DataSet. The initialization occurs at run time.

Clear Clears data.

Clone Copies the structure of the DataSet, including all DataTable schemas, relations, and constraints. Does not copy any data.

Copy Copies both structure and data.

CreateDataReader() Returns a DataTableReader with one result set per DataTable, in the same sequence as the tables appear in the Tables collection.

CreateDataReader(DataTable[]) Returns a DataTableReader with one result set per DataTable.

EndInit Ends the initialization of the data set.

Equals(Object) Determines whether the specified Object is equal to the current Object.

Finalize Free resources and perform other cleanups.

GetChanges Returns a copy of the DataSet with all changes made since it was loaded or the AcceptChanges method was called.

GetChanges(DataRowState) Gets a copy of DataSet with all changes made since it was loaded or the AcceptChanges method was called, filtered by DataRowState.

GetDataSetSchema Gets a copy of XmlSchemaSet for the DataSet.

GetObjectData Populates a serialization information object with the data needed to serialize the DataSet.

GetType Gets the type of the current instance.

GetXML Returns the XML representation of the data.

GetXMLSchema Returns the XSD schema for the XML representation of the data.

HasChanges() Gets a value indicating whether the DataSet has changes, including new, deleted, or modified rows.

HasChanges(DataRowState) Gets a value indicating whether the DataSet has changes, including new, deleted, or modified rows, filtered by DataRowState.

IsBinarySerialized Inspects the format of the serialized representation of the DataSet.

Load(IDataReader, LoadOption, DataTable[]) Fills a DataSet with values from a data source using the supplied IDataReader, using an array of DataTable instances to supply the schema and namespace information.

Load(IDataReader, LoadOption, String[]) Fills a DataSet with values from a data source using the supplied IDataReader, using an array of strings to supply the names for the tables within the DataSet.

Merge() Merges the data with data from another DataSet. This method has different overloaded forms.

ReadXML() Reads an XML schema and data into the DataSet. This method has different overloaded forms.

ReadXMLSchema(0) Reads an XML schema into the DataSet. This method has different overloaded forms.

RejectChangesRolls back all changes made since the last call to AcceptChanges.

WriteXML() Writes an XML schema and data from the DataSet. This method has different overloaded forms.

WriteXMLSchema() Writes the structure of the DataSet as an XML schema. This method has different overloaded forms.

The DataTable Class

The DataTable class represents the tables in the database. It has the following important properties; most of these properties are read only properties except the PrimaryKey property:

Properties	Description
ChildRelations	Returns the collection of child relationship.
Columns	Returns the Columns collection.
Constraints	Returns the Constraints collection.
DataSet	Returns the parent DataSet.
DefaultView	Returns a view of the table.
ParentRelations	Returns the ParentRelations collection.
PrimaryKey	Gets or sets an array of columns as the primary key for the table.
Rows	Returns the Rows collection.

The following table shows some important methods of the DataTable class:

Methods	Description
AcceptChanges	Commits all changes since the last AcceptChanges.
Clear	Clears all data from the table.

GetChanges Returns a copy of the DataTable with all changes made since the AcceptChanges method was called.

GetErrors Returns an array of rows with errors.

ImportRows Copies a new row into the table.

LoadDataRow Finds and updates a specific row, or creates a new one, if not found any.

Merge Merges the table with another DataTable.

NewRow Creates a new DataRow.

RejectChanges Rolls back all changes made since the last call to AcceptChanges.

Reset Resets the table to its original state.

Select Returns an array of DataRow objects.

The DataRow Class

The DataRow object represents a row in a table. It has the following important properties:

Properties Description

HasErrors Indicates if there are any errors.

Items Gets or sets the data stored in a specific column.

ItemArrays Gets or sets all the values for the row.

Table Returns the parent table.

The following table shows some important methods of the DataRow class:

Methods Description

AcceptChanges Accepts all changes made since this method was called.

BeginEdit Begins edit operation.

CancelEdit Cancels edit operation.

Delete Deletes the DataRow.

EndEdit Ends the edit operation.

GetChildRows Gets the child rows of this row.

GetParentRow Gets the parent row.

GetParentRows Gets parent rows of DataRow object.

RejectChanges Rolls back all changes made since the last call to AcceptChanges.

The DataAdapter Object

The DataAdapter object acts as a mediator between the DataSet object and the database. This helps the Dataset to contain data from multiple databases or other data source.

The DataReader Object

The DataReader object is an alternative to the DataSet and DataAdapter combination. This object provides a connection oriented access to the data records in the database. These objects are suitable for read-only access, such as populating a list and then breaking the connection.

DbCommand and DbConnection Objects

The DbConnection object represents a connection to the data source. The connection could be shared among different command objects.

The DbCommand object represents the command or a stored procedure sent to the database from retrieving or manipulating data.

5. What is WCF and Web service Explain? Write down the difference between them.

Ans.

A Web service, in the context of .NET, is a component that resides on a Web server and provides information and services to other network applications using standard Web protocols such as HTTP and Simple Object Access Protocol (SOAP).

.NET Web services provide asynchronous communications for XML applications that operate over a .NET communications framework. They exist so that users on the Internet can use applications that are not dependent on their local operating system or hardware and are generally browser-based.

Web services are designed to provide the messaging infrastructure necessary for communication across platforms using industry standards. Web services also use asynchronous communication to address the latency issue that arises due to requests from remote locations across the Internet. This allows the execution of background tasks for the client (such as responding to user interactions) until the actual completion of the Web service request.

Advantages of WCF

1) WCF is interoperable with other services when compared to .Net Remoting where the client and service have to be .Net.

2) WCF services provide better reliability and security in compared to ASMX web services.

3) In WCF, there is no need to make much change in code for implementing the security model and changing the binding. Small changes in the configuration will make your requirements.

4) WCF has integrated logging mechanism, changing the configuration file settings will provide this functionality. In other technology developer has to write the code.

Difference between WCF and Web service

Web service is a part of WCF. WCF offers much more flexibility and portability to develop a service when comparing to web service. Still we are having more advantages over Web service; following table provides detailed difference between them.

It can be hosted in IIS whereas

It can be hosted in IIS, windows activation service, Self-hosting, Windows service

Programming

[WebService] attribute has to be added to the class

Whereas [ServiceContract] attribute has to be added to the class

Model

[WebMethod] attribute represents the method exposed to client whereas [OperationContract] attribute represents the method exposed to client

Operation

One-way, Request- Response are the different operations supported in web service whereas One-Way, Request-Response, Duplex are different type of operations supported in WCF

XML

System.Xml.serialization name space is used for serialization whereas System.Runtime.Serialization namespace is used for serialization

Encoding

XML 1.0, MTOM(Message Transmission Optimization Mechanism), DIME, Custom whereas XML 1.0, MTOM, Binary, Custom

Transports

Can be accessed through HTTP, TCP, Custom whereas Can be accessed through HTTP, TCP, Named pipes, MSMQ,P2P, Custom

Security

Security, Reliable messaging, Transactions whereas A WCF Service is composed of three components parts viz,

1) Service Class - A WCF service class implements some service as a set of methods.

2) Host Environment - A Host environment can be a Console application or a Windows Service or a Windows Forms application or IIS as in case of the normal asmx web service in .NET.

3) Endpoints - All communications with the WCF service will happen via the endpoints. The endpoint is composed of 3 parts (collectively called as ABC's of endpoint) as defines below:

Address: The endpoints specify an Address that defines where the endpoint is hosted. It’s basically url.

OR

5. Explain Session and application state management.

Session State

Session state is generally used for storing application data such as inventory, supplier list, customer record, or shopping cart. It can also keep information about the user and his preferences, and keep the track of pending operations.

The session state object is created from the HttpSessionState class, which defines a collection of session state items.

The HttpSessionState class has the following properties:

Properties	Description
SessionID	The unique session identifier.
Item(name)	The value of the session state item with the specified name. This is the default property of the HttpSessionState class.
Count	The number of items in the session state collection.
TimeOut	Gets and sets the amount of time, in minutes, allowed between requests before the session-state provider terminates the session.

The HttpSessionState class has the following methods:

The session state object is a name-value pair to store and retrieve some information from the session state object.

Application State

Application State is mostly used to store hit counters and other statistical data, global application data like tax rate, discount rate etc. and to keep the track of users visiting the site.

The HttpApplicationState class has the following properties:

Properties	Description
Item(name)	The value of the application state item with the specified name. This is the default property of the HttpApplicationState class.

Count	The number of items in the application state collection.

The HttpApplicationState class has the following methods:

Methods	**Description**
Add(name, value)	Adds an item to the application state collection.
Clear	Removes all the items from the application state collection.
Remove(name)	Removes the specified item from the application state collection.
RemoveAll	Removes all objects from an HttpApplicationState collection.
RemoveAt	Removes an HttpApplicationState object from a collection by index.
Lock()	Locks the application state collection so only the current user can access it.
Unlock()	Unlocks the application state collection so all the users can access it.

Application state data is generally maintained by writing handlers for the events:

- Application_Start
- Application_End
- Application_Error
- Session_Start
- Session_End