

## **UNIT IV: ASP.NET WITH C#**

### **UNIVERSITY QUESTIONS WITH ANSWER**

← **Q. No 1: List and Explain ADO.NET objects. (Nov 18)**

ADO.NET includes many objects we can use to work with data. Some important objects of ADO.NET are:

Connection:

- To interact with a database, we must have a connection to it. The connection helps identify the database server, the database name, user name, password, and other parameters that are required for connecting to the data base.
- A connection object is used by command objects so they will know which database to execute the command on.

Command :

- The command object is one of the basic components of ADO .NET. The Command Object uses the connection object to execute SQL queries.
- The queries can be in the Form of Inline text, Stored Procedures or direct Table access. An important feature of Command object is that it can be used to execute queries and Stored Procedures with Parameters.
- If a select query is issued, the result set it returns is usually stored in either a DataSet or a DataReader object.

DataReader :

- Many data operations require that we only get a stream of data for reading. The data reader object allows us to obtain the results of a SELECT statement from a command object. For performance reasons, the data returned from a data reader is a fast forward-only stream of data.
- This means that we can only pull the data from the stream in a sequential manner this is good for speed, but if we need to manipulate data, then a DataSet is a better object to work with.

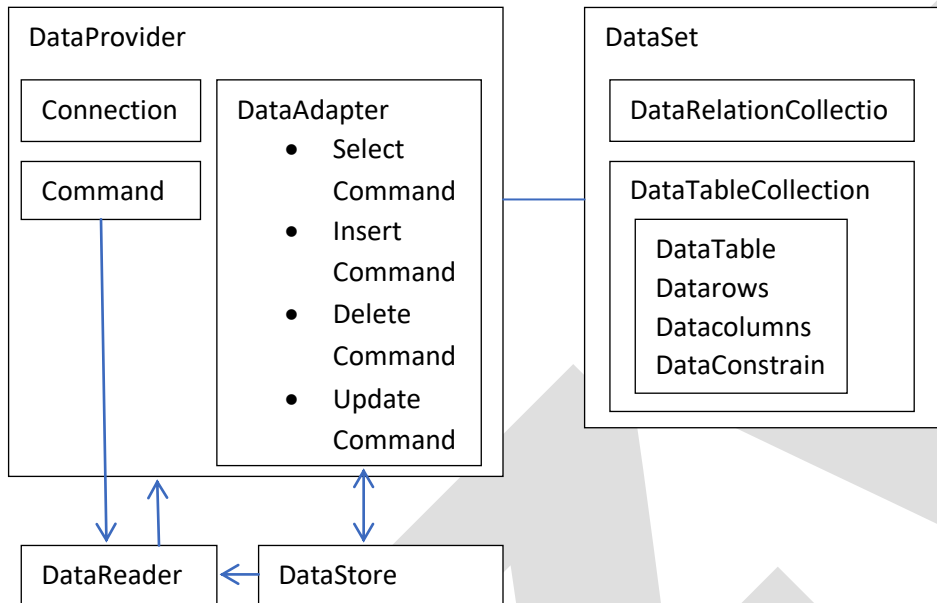
DataSet :

- DataSet objects are in-memory representations of data. They contain multiple Datatable objects, which contain columns and rows, just like normal database tables. We can even define relations between tables to create parent-child relationships.
- The DataSet is specifically designed to help manage data in memory and to support disconnected operations on data, when such a scenario make sense.

DataAdapter:

## B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE

- The data adapter fills a DataSet object when reading the data and writes in a single batch when persisting changes back to the database.
- A data adapter contains a reference to the connection object and opens and closes the connection automatically when reading from or writing to the database.



**Q.No 2: What is DataReader in ADO.NET? Explain with example? OR Give details about DataReader with example?(Nov 18,May19)**

1. DataReader provides an easy way for the programmer to read data from a database as if it were coming from a stream.
2. The DataReader is the solution for forward streaming data through ADO.NET.
3. DataReader is also called a firehose cursor or forward read-only cursor because it moves forward through the data.
4. The DataReader not only allows us to move forward through each record of database, but it also enables us to parse the data from each column.
5. The DataReader class represents a data reader in ADO.NET.
6. DataReader is like a forward only recordset.
7. It fetches one row at a time so very less network cost compare to DataSet(Fethces all the rows at a time).
8. DataReader is readonly so we can't do any update or transaction on them.
9. DataReader will be the best choice where we need to show the data to the user which requires no transaction.
10. As DataReader is forward only so we can't fetch data randomly.
11. NET Data Providers optimizes the DataReader to handle huge amount of data.
12. Performance is good.
13. DataReader is a connection oriented architecture.

## B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE

---

Example:

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Data.SqlClient;

namespace CommandTypeEnumeration
{
    class Program
    {
        static void Main(string[] args)
        {
            // Create a connection string
            string ConnectionString = "Integrated Security = SSPI; " +
                "Initial Catalog= Northwind; " + " Data source = localhost; ";
            string SQL = "SELECT * FROM Customers";

            // create a connection object
            SqlConnection conn = new SqlConnection(ConnectionString);

            // Create a command object
            SqlCommand cmd = new SqlCommand(SQL, conn);
            conn.Open();

            // Call ExecuteReader to return a DataReader
            SqlDataReader reader = cmd.ExecuteReader();
            Console.WriteLine("customer ID, Contact Name, " + "Contact Title, Address ");
            Console.WriteLine("=====");

            while (reader.Read())
            {
                Console.Write(reader["CustomerID"].ToString() + ", ");
                Console.Write(reader["ContactName"].ToString() + ", ");
                Console.Write(reader["ContactTitle"].ToString() + ", ");
                Console.WriteLine(reader["Address"].ToString() + ", ");
            }

            //Release resources
            reader.Close();
            conn.Close();
        }
    }
}
```

## B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE

### Q.No 3: Explain SqlDataSource in ADO.NET.(Nov 18)

1. The SqlDataSource control represents a connection to a relational database such as SQL Server or Oracle database, or data accessible through OLEDB or Open Database Connectivity (ODBC).
2. Connection to data is made through two important properties ConnectionString and ProviderName.
3. It enables to work with Microsoft SQL Server,OLE DB,ODBC,or Oracle Databases
4. When it used with SQL Server,supports advanced caching capabilities.
5. The control also supports sorting, filtering, and paging when data is returned as a DataSet object.

The following code snippet provides the basic syntax of the control:

```
<asp:SqlDataSource runat="server" ID="MySqlSource"
ProviderName='<%%$ ConnectionStrings:LocalNWind.ProviderName%>'
ConnectionString='<%%$ ConnectionStrings:Empconstr %>'
SelectionCommand= "SELECT * FROM EMPLOYEES" />
<asp:GridView ID="GridView1" runat="server" DataSourceID="MySqlSource"/>
```

Configuring various data operations on operations on the underlying data depends upon the various properties (property groups) of the datasource control.

Properties of SqlDataSource control used for programming interface of the control.

Property Group	Description
DeleteCommand, DeleteParameters, DeleteCommandType	Gets or sets the SQL statement,parameters,and type for deleting rows in the underlying data.
FilterExpression, FilterParameters	Gets or sets the data filtering string and parameters.
InsertCommand, InsertParameters, InsertCommandType	Gets or sets the SQL statement, parameters, and type for inserting rows in the underlying database.
SelectCommand, SelectParameters, SelectCommandType	Gets or sets the SQL statement, parameters, and type for retrieving rows from the underlying database.
SortParameterName	Gets or sets the name of an input parameter that the command's sorted procedure will use to sort data.
UpdateCommand, UpdateParameters, UpdateCommandType	Gets or sets the SQL statement, parameters, and type for updating rows in the underlying data store.

The following code snippet shows a data source control enabled for data manipulation:

```
<asp:SqlDataSource runat="server" ID= "MySqlSource"
ProviderName='<%%$ ConnectionStrings:LocalNWind.ProviderName%>'
ConnectionString=' <%%$ ConnectionStrings:Empconstr %>'
```

## B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE

SelectCommand= "SELECT \* FROM EMPLOYEES"

UpdateCommand="UPDATE EMPLOYEES SET LASTNAME=@lname"

DeleteCommand="DELETE FROM EMPLOYEES WHERE EMPLOYEEID=@eid"

FilterExpression ="EMPLOYEEID>10">

.....

.....

<asp:SqlDataSource>

### Q.No. 4: What is a GridView control? Explain with an example?(Nov 18)

1. A recurring task in software development is to display tabular data.
2. ASP.NET provides a number of tools for showing tabular data in a grid, including the GridView control.
3. With the GridView control, you can display,edit,and delete data from many different kinds of data sources, including databases,XML files, and business objects that expose data.
4. You can use the GridView control to do the following:
  - Automatically bind to add display data from a data source control.
  - Select,sort,page through,edit and delete data from a data source control.
5. Additionally, you can customize the appearance and behaviour of the GridView control by doing the following:
  - Specifying custom columns and styles.
  - Utilizing templates to create custom user interface(UI) elements.
  - Adding your own code to the functionality of the GridView control by handling events.
6. The GridView control is the successor to the DataGrid control.
7. The GridView control display database records in a HTML table with each record in a table row and each field in a column.

#### Code: (GridView Source Code)

```
<asp:GridViewID="GridView1"runat="server" AllowPaging="True" AllowSorting="True" AutoGenerateColumns="False"DataKeyNames="Product_id"DataSourceID="SqlDataSource1"Height="162px"Width="327px"></asp:GridView>
```

Example:

```
using System;  
using System.Data;  
using System.Data.SqlClient;  
using System.Web;  
using System.Web.UI;  
using System.Web.UI.WebControls;
```

```
publicpartialclassstudentdetails : System.Web.UI.Page  
{  
protectedvoid Page_Load(object sender, EventArgs e)
```

## B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE

---

```
{  
  
}  
protected void Button1_Click(object sender, EventArgs e)  
{  
    SqlConnection conn = new SqlConnection();  
    conn.ConnectionString = ("Data Source=hp;Initial Catalog=studentinfo;Integrated  
Security=True");  
  
    string query = "select * from students";  
  
    try  
    {  
        conn.Open();  
        SqlCommand cmd = new SqlCommand(query, conn);  
        SqlDataAdapter da = new SqlDataAdapter();  
        da.SelectCommand = cmd;  
  
        DataSet myDS = new DataSet();  
        da.Fill(myDS, "Student");  
        GridView1.DataSource = myDS;  
        GridView1.DataBind();  
    }  
    catch (Exception ex)  
    {  
        Label1.Text = "Problem in connection or in database";  
    }  
    finally  
    {  
        conn.Close();  
    }  
}
```

**Q.No 5: What is the application services provided in ASP.NET? Explain.(Nov 18)**

1. ASP.NET application services are built-in Web services that provide access to features such as forms authentication, roles, and profile properties.
2. These services are part of a service-oriented architecture (SOA), in which an application consists of one or more services provided on the server, and one or more clients.
3. For more information about SOA, see Understanding ServiceOriented Architecture on the MSDN Web site.
4. An important feature of ASP.NET application services is that they are available to a variety of client applications, not just ASP.NET Web applications.

## B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE

5. ASP.NET application services are available to any client that is based on the .NET Framework. In addition, any client application that can send and receive messages in SOAP format can use ASP.NET application services.
6. This topic contains the following information.
  - Scenarios
  - Background
  - Examples
  - Class Reference
  - Additional Resources

1. Client applications for ASP.NET application services can be of different types and can run on different operating systems. These include these following types of clients:

- AJAX clients. These are ASP.NET Web pages (.aspx files) that run in the browser and that access application services from client script. AJAX clients typically use JSON format to exchange data.

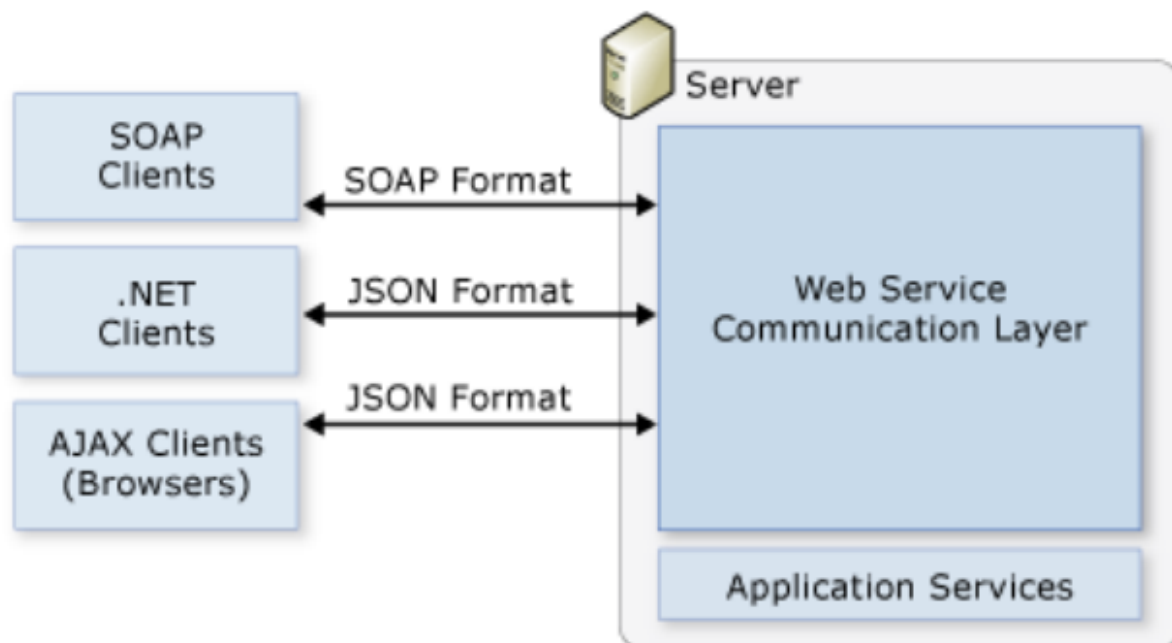
AJAX.

- .NET Framework clients. These are .NET Framework Windows applications that access application services over HTTP by using the provider model infrastructure, and that use JSON protocol to exchange data..

- SOAP clients. These are clients that can access application services through SOAP 1.1. This is useful for clients that are running on other operating systems or using other technologies, such as Java applications.

The following illustration shows how different clients communicate with the services.

Web Services communication



Background



## **B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE**

---

The application services provided by ASP.NET enable client applications to access and share information that is part of a Web application. ASP.NET makes available the following application services:

- **Authentication service.** This service enables you to let users log onto an application. The service accepts user credentials and returns an authentication ticket (cookie).
- **Roles service.** This service determines the application-related roles for an authenticated user, based on information that is made available by an ASP.NET roles provider. This can be useful if you want to provide specific UI or enable access to specific resources, depending on the user's role.
- **Profile service.** This service provides per-user information as a user profile that is stored on the server. This enables your application to access a user's settings at different times and from different client UI components.

### **Application Service Clients**

This section provides details about the types of client applications that can use ASP.NET application services and some information about how the client communicates with an application service.

#### **AJAX Clients**

AJAX clients (AJAX-enabled ASP.NET Web pages) exchange data with application services over HTTP by using POST requests. The data is packaged in JSON format. The client application communicates with the application services through client-script proxy classes. The proxy classes are generated by the server and downloaded to the browser as part of any page that calls an application service.

ASP.NET application services exchange data with .NET Framework clients over HTTP by using POST requests. The data is packaged in JSON format. The client application communicates with the application services by using the .NET Framework provider model. For ASP.NET application services, the provider model refers to the .NET Framework client types and the related membership providers that store and retrieve user's credentials from a data source. For example, this includes the `SqlMembershipProvider` class. The communication between client and server is synchronous. For more information, see *Client Application Services Overview*. The application services are implemented by the types that are defined in the `System.Web.ClientServices.Providers` namespace. To access an application service, a .NET Framework client application must be configured appropriately. The server configuration is the same as the one used for the application services in AJAX.

#### **SOAP Clients**

You can access ASP.NET authentication, profile, and roles services from any client application on any operating system that can use SOAP 1.1 protocol. ASP.NET application services are built on the Windows Communication Foundation (WCF) and exchange data with the client in SOAP format. For more information, see *XML Web Services Infrastructure* on the MSDN Web site. Communication between the client and the application services is performed by using proxy classes that run in the client and that represent the application service. You can generate proxy classes that support ASP.NET application services by using the Service Model Metadata Utility Tool (`svcutil.exe`) tool.



## **B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE**

---

### **Q.No 6: Differentiate between FormView and DetailsView.(Nov 18)**

- The DetailsView control is easier to work with.
- The FormView control provides more control over the layout.
- The FormView control uses only the templates with databinding expressions to display data. The DetailsView control uses <BoundField> elements or <TemplateField> elements.
- The FormView control renders all fields in a single table row whereas the DetailsView control displays each field as a table row.  
The DetailsView control can be easier to work with, but the FormView control provides more formatting and layout options.
- The DetailsView control can use BoundField elements or TemplateField elements with templates that use data binding expressions to define bound data fields. The FormView control can use only templates with data binding expressions to display bound data.
- The DetailsView control renders each field as a table row, but the FormView control renders all the fields in a template as a single table row.
- When you bind a FormView control to a data source, the Web Forms Designer generates an Item template that includes heading text and a bound label for each column in the data source.
- The Item template is rendered whenever the FormView control is displayed in ReadOnly mode.
- The Item template uses the Eval and Bind methods to create binding expressions for the columns in the data source.
- If the data source includes Update, Delete, and Insert commands, the generated Item template will include Edit, Delete, and New buttons.
- The Web Forms Designer also generates an EditItem template and an InsertItem template, even if the data source doesn't include an Update or Insert command. For more information, see the next figure.
- You can modify a generated template so you can use CSS to control the format and layout of the data that's rendered for that template.

### **Q.No. 7: Explain the SQL Data Provider Model.(May 19)**

ADO.NET relies on the functionality in a small set of core classes. You can divide these classes into two groups.

Those that are used to contain and manage data (such as DataSet, DataTable, DataRow, and DataRelation) and those that are used to connect to a specific data source (such as Connection, Command, and DataReader).

The data container classes are completely generic.

No matter what data source you use, after you extract the data, it's stored using the same data container: the specialized DataSet class. Think of the DataSet as playing the same role as a collection or an array—it's a package for data.

The difference is that the DataSet is customized for relational data, which means it understands concepts such as rows, columns, and table relationships natively.

## **B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE**

The second group of classes exists in several flavors. Each set of data interaction classes is called an ADO.NET data provider.

Data providers are customized so that each one uses the best-performing way of interacting with its data source. For example, the SQL Server data provider is designed to work with SQL Server. Internally, it uses SQL Server's tabular data stream (TDS) protocol for communicating, thus guaranteeing the best possible performance.

If you're using Oracle, you can use an Oracle data provider.

Each provider designates its own prefix for naming classes. Thus, the SQL Server provider includes `SqlConnection` and `SqlCommand` classes, and the Oracle provider includes `OracleConnection` and `OracleCommand` classes.

Internally, these classes work quite differently, because they need to connect to different databases by using different low-level protocols.

Externally, however, these classes look quite similar and provide an identical set of basic methods because they implement the same common interfaces.

This means your application is shielded from the complexity of different standards and can use the SQL Server provider in the same way the Oracle provider uses it.

Table : ADO.NET Namespaces for SQL Server Data Access

Namespace	Purpose
<code>System.Data.SqlClient</code>	Contains the classes you use to connect to a Microsoft SQL Server database and execute commands (such as <code>SqlConnection</code> and <code>SqlCommand</code> ).
<code>System.Data.SqlTypes</code>	Contains structures for SQL Server-specific data types such as <code>SqlMoney</code> and <code>SqlDateTime</code> . You can use these types to work with SQL Server data types without needing to convert them into the standard .NET equivalents (such as <code>System.Decimal</code> and <code>System.DateTime</code> ). These types aren't required, but they do allow you to avoid any potential rounding or conversion problems that could adversely affect data.
<code>System.Data</code>	Contains fundamental classes with the core ADO.NET functionality. These include <code>DataSet</code> and <code>DataReladon</code> , which allow you to manipulate structured relational data. These classes are totally independent of any specific type of database or the way you connect to it

## **B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE**

---

### **Q. No 8: What is data Binding? Explain it's type? (May 19)**

Web Forms provides many controls that support data binding. You can connect these controls to ADO.NET components such as a DataView, DataSet, or a DataViewManager at design-time as well as at run-time. Data binding in Web Forms works in the same way as it does in the Windows Forms with a few exceptions.

ASP.NET provides a rich set of data-bound server controls. These controls are easy to use and provide a Rapid Application Development (RAD) Web development. You can categorize these controls in two groups: single-item data-bound control and multi-item data-bound controls.

Every ASP.NET web form control inherits the DataBind method from its parent Control class, which gives it an inherent capability to bind data to at least one of its properties. This is known as **simple data binding** or **inline data binding**.

Simple data binding involves attaching any collection (item collection) which implements the IEnumerable interface, or the DataSet and DataTable classes to the DataSource property of the control.

On the other hand, some controls can bind records, lists, or columns of data into their structure through a DataSource control. These controls derive from the BaseDataBoundControl class. This is called **declarative data binding**.

### **Simple Data Binding**

Simple data binding involves the read-only selection lists. These controls can bind to an array list or fields from a database. Selection lists takes two values from the database or the data source; one value is displayed by the list and the other is considered as the value corresponding to the display.

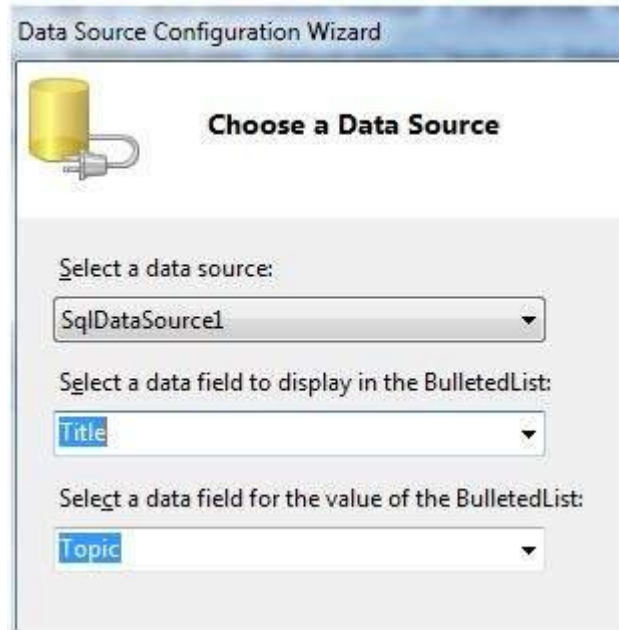
Let us take up a small example to understand the concept. Create a web site with a bulleted list and a SqlDataSource control on it. Configure the data source control to retrieve two values from your database (we use the same DotNetReferences table as in the previous chapter).

Choosing a data source for the bulleted list control involves:

- Selecting the data source control
- Selecting a field to display, which is called the data field
- Selecting a field for the value

## B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE

---



When the application is executed, check that the entire title column is bound to the bulleted list and displayed.

### Declarative Data Binding

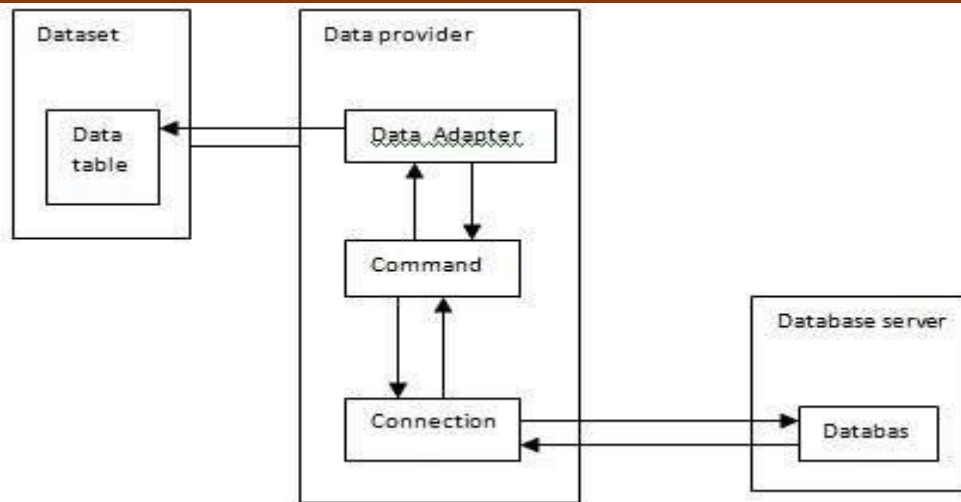
We have already used declarative data binding in the previous tutorial using GridView control. The other composite data bound controls capable of displaying and manipulating data in a tabular manner are the DetailsView, FormView, and RecordList control.

In the next tutorial, we will look into the technology for handling database, i.e, ADO.NET.

However, the data binding involves the following objects:

- A dataset that stores the data retrieved from the database.
- The data provider, which retrieves data from the database by using a command over a connection.
- The data adapter that issues the select statement stored in the command object; it is also capable of update the data in a database by issuing Insert, Delete, and Update statements.

## B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE



### Q.No 9: Write short note on Data Source Control ?(May 19)

A data source control interacts with the data-bound controls and hides the complex data binding processes. These are the tools that provide data to the data bound controls and support execution of operations like insertions, deletions, sorting, and updates.

Each data source control wraps a particular data provider-relational databases, XML documents, or custom classes and helps in:

- Managing connection
- Selecting data
- Managing presentation aspects like paging, caching, etc.
- Manipulating data

There are many data source controls available in ASP.NET for accessing data from SQL Server, from ODBC or OLE DB servers, from XML files, and from business objects.

Based on type of data, these controls could be divided into two categories:

- Hierarchical data source controls
- Table-based data source controls

The data source controls used for hierarchical data are:

XMLDataSource - It allows binding to XML files and strings with or without schema information

SiteMapDataSource - It allows binding to a provider that supplies site map information.

Data source controls	Description
SqlDataSource	It represents a connection to an ADO.NET data provider that returns SQL data, including data sources accessible via OLEDB and ODBC.
ObjectDataSource	It allows binding to a custom .Net business object that returns data.
LinqdataSource	It allows binding to the results of a Linq-to-SQL query (supported by ASP.NET 3.5 only).
AccessDataSource	It represents connection to a Microsoft Access

## B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE

database

- **The SqlDataSource Control :-**

The SqlDataSource control represents a connection to a relational database such as SQL Server or Oracle database, or data accessible through OLEDB or Open Database Connectivity (ODBC). Connection to data is made through two important properties ConnectionString and ProviderName.

- **The ObjectDataSource Control :-**

The ObjectDataSource Control enables user-defined classes to associate the output of their methods to data bound controls. The programming interface of this class is almost same as the SqlDataSource control.

- **The AccessDataSource Control :-**

The AccessDataSource control represents a connection to an Access database. It is based on the SqlDataSource control and provides simpler programming interface.

- **The LinqDataSource control :-**

The LinqDataSource control enables you to use LINQ in an ASP.NET Web page by setting properties in markup text. The LinqDataSource control uses LINQ to SQL to automatically generate the data commands.

### Q.No :10 Explain the ways of formatting GridView Data for display(May 19)

The GridView is an extremely flexible grid control for showing data in a basic grid consisting of rows and columns. It has selection, paging and editing feature, and it is extensible through templates. The great advantage of Gridview over Datagrid is its support for code free scenarios. In GridView you can do many things without writing any code like paging and selection.

FormattingFields:

To format the grid view you have to ensure that dates, currency and other number values are in good format. Grid View has property "DataFormatString" to apply formatting. You can change colors, fonts, borders and alignment of grid. Each BoundField column provides a DataFormatString property that you can use to configure the numbers and dates using a format string.

Format strings are generally made up of a placeholder and format indicator, which are wrapped inside curly brackets, like this:

{0:C}

Here 0 shows the value that will be formatted and the letter indicates a predetermined format style. In this case C means currency format which formats a number as a dollar.

```
<asp:BoundField DataField="Price" HeaderText="Price" DataFormatString="{0:C}"
HtmlEncode="false" />
```



## B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE

---

Here we are going to discuss about few format strings, if you want to know in details then you can search on msdn.

Numeric Format Strings :

Currency {0:C} - \$1,234.50 Brackets indicate negative values(\$1,234.50). Currency sign is locale specific (?1,234.50).

Scientific (Exponential) {0:E} - 1.234.50E+004

Percentage {0:P} - 35.5%

Fixed Decimal {0:F?} - Depends on the number of decimal places you set {0:F3} would be 123.400. {0:F0} would be 123.

Time and Date Format Strings:

<asp:BoundField DataField="DOB" HeaderText="DOB" DataFormatString="{0:MM/dd/yy}" HtmlEncode="false" />

Short Date {0:d} - M/d/yyyy (11/21/2003)

Long Date {0:D} - dddd, MMMM dd, yyyy (Saturday, March 21, 2001)

Long Date and Short Time {0:f} - dddd, MMMM dd, yyyy HH:mm aa (Saturday, March 21, 2003 11:00 AM)

Long Date and Long Time {0:F} - dddd, MMMM dd, yyyy HH:mm:ss aa (Saturday, March 21, 2003 11:00:20 AM)

ISO Sortable Standard {0:s} - yyyy-MM-dd HH:mm:ss (2003-01-21 11:00:21)

Month and Day {0:M} - MMMM dd (March 21)

General {0:G} - M/d/yyyy HH:mm:ss aa (depends on local specific setting) (10/21/2003 11:00:21 AM)

Styles : you can set eight GridView styles.

Header Style: Set the header row style that contains column titles if you do ShowHeader property true.

RowStyle: Set the style of every data row.

AlternatingRowStyle: Set the style of every alternate row in gridview.

SelectedRowStyle: Set the style of currently selected row.

EditRowStyle: Set the style of row that is in edit mode. This formatting acts in addition to the RowStyle formatting.

EmptyDataRowStyle: Set the style that is used fro the single row in the special case where the bound data object contains no rows.

FooterStyle: Set the style of the footer row at the bottom of the GridView, if you choose ShowFooter property true.

PagerStyle: Set the style of the row with the page links if you enable AllowPaging property true.

Example:



## B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE

```
<asp:GridView ID="GridView1" runat="server" AllowPaging="True" ShowFooter="True">
```

```
<RowStyle BackColor="Gray" Font-Italic="True" />
```

```
<EmptyDataRowStyle BackColor="Yellow" />
```

```
<PagerStyle BackColor="#FFC0C0" Font-Italic="True" Font-Underline="True" />
```

```
<SelectedRowStyle BackColor="#00C000" Font-Bold="True" Font-Italic="True" />
```

```
<EditRowStyle BackColor="#0000C0" />
```

```
<AlternatingRowStyle BackColor="Red" BorderColor="Green" BorderStyle="Dashed" BorderWidth="1px" Font-Bold="True" />
```

```
<FooterStyle BackColor="#00C0C0" />
```

```
<HeaderStyle BackColor="#00C000" Font-Bold="True" Font-Italic="True" Font-Underline="True" />
```

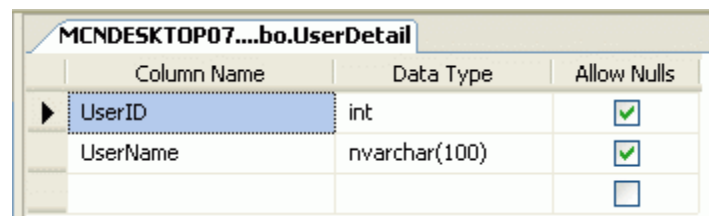
```
</asp:GridView>
```

### Q.No 11: Write a short note on selecting grid view row (May 19)

A GridView allows us to select only a single row at a time. The sample makes use of the database. We will be pulling data from the UserDetails table.

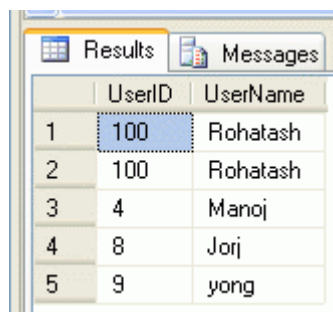
Creating Table in SQL Server Database

Now create a table named UserDetails with the columns UserID and UserName. The table looks as below.



	Column Name	Data Type	Allow Nulls
►	UserID	int	<input checked="" type="checkbox"/>
	UserName	nvarchar(100)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Now insert data into the table.



	UserID	UserName
1	100	Rohatash
2	100	Rohatash
3	4	Manoj
4	8	Jorj
5	9	yong

First of all drag the GridView control from the Data controls menu. It will add the GridView control's HTML source code as given above.

1. `<asp:GridView ID="GridView2" runat="server"></asp:GridView>`

## B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE

You will need to set the Bound column in order to see the text in the cells of the GridView control.

.aspx source code

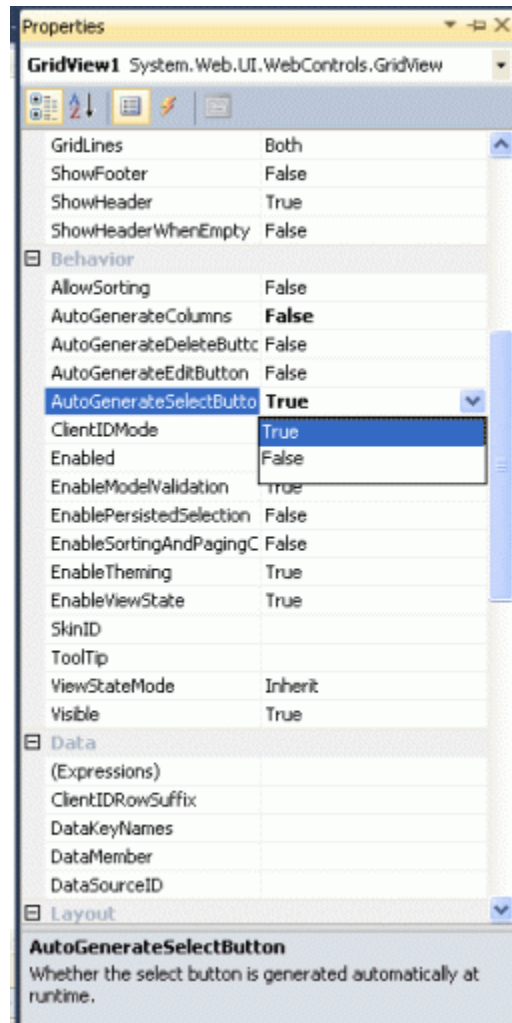
```
1. <%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs
   " Inherits="WebApplication120.WebForm1" %>
2. <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml
   1-transitional.dtd">
3. <html xmlns="http://www.w3.org/1999/xhtml">
4. <head runat="server">
5.   <title></title>
6. </head>
7. <body>
8.   <form id="form1" runat="server">
9.     <div>
10.      UserID:
11.      <asp:TextBox ID="TextBoxUserID" runat="server"></asp:TextBox>
12.      <br />
13.      UserName:
14.      <asp:TextBox ID="TextBoxUserName" runat="server"></asp:TextBox>
15.      <br />
16.      <br />
17.      <asp:GridView ID="GridView1" runat="server" AutoGenerateSelectButton="True"
         OnSelectedIndexChanged="GridView1_SelectedIndexChanged"
18.        BackColor="#DEBA84" BorderColor="#DEBA84" BorderStyle="None" Border
         Width="1px"
19.        CellPadding="3" CellSpacing="2" AutoGenerateColumns="False">
20.        <FooterStyle BackColor="#F7DFB5" ForeColor="#8C4510" />
21.        <HeaderStyle BackColor="#A55129" Font-Bold="True" ForeColor="White" />
22.        <PagerStyle ForeColor="#8C4510" HorizontalAlign="Center" />
23.        <RowStyle BackColor="#FFF7E7" ForeColor="#8C4510" />
24.        <SelectedRowStyle BackColor="#738A9C" Font-
         Bold="True" ForeColor="White" />
25.        <SortedAscendingCellStyle BackColor="#FFF1D4" />
26.        <SortedAscendingHeaderStyle BackColor="#B95C30" />
27.        <SortedDescendingCellStyle BackColor="#F1E5CE" />
28.        <SortedDescendingHeaderStyle BackColor="#93451F" />
29.        <Columns>
30.          <asp:BoundField HeaderText="UserID" DataField="UserID" />
31.          <asp:BoundField HeaderText="UserName" DataField="UserName" />
32.        </Columns>
33.      </asp:GridView>
34.    </div>
35.  </form>
```

## B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE

36. </body>

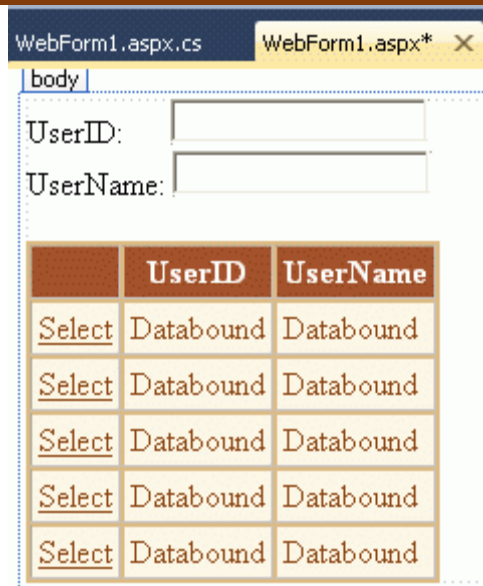
37. </html>

In the above GridView Properties set 'AutoGenerateSelectButton=True'. See the following image of a GridView after setting 'AutoGenerateSelectButton=True'. Select the GridView and press F4 for the property window.



See the Design view of your GridView. You will find a Hyperlink with a text as 'Select'.

## B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE



The screenshot shows a web application window titled 'WebForm1.aspx.cs' and 'WebForm1.aspx\*'. The page contains a login form with two input fields: 'UserID:' and 'UserName:'. Below the form is a table with three columns: 'UserID', 'UserName', and an unnamed column. The unnamed column contains a 'Select' button for each row. The table has five rows of data, all showing 'Databound' for both 'UserID' and 'UserName'.

	UserID	UserName
Select	Databound	Databound
Select	Databound	Databound
Select	Databound	Databound
Select	Databound	Databound
Select	Databound	Databound

Now add the following namespace.

1. using System.Data.SqlClient;
2. using System.Data;

Now write the connection string to connect to the database.

1. string strConnection = "Data Source=.; uid=sa; pwd=wintellect;database=Rohatash;"

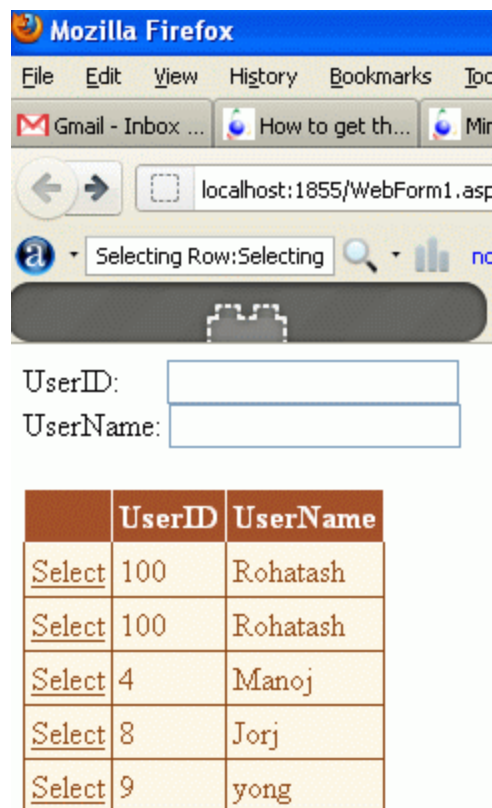
Now double-click on the page and write the following code for binding the data with the GridView.

1. using System;
2. using System.Collections.Generic;
3. using System.Linq;
4. using System.Web;
5. using System.Web.UI;
6. using System.Web.UI.WebControls;
7. using System.Data.SqlClient;
8. using System.Data;
9. namespace WebApplication120
10. {
11. public partial class WebForm1 : System.Web.UI.Page
12. {
13. protected void Page\_Load(object sender, EventArgs e)
14. {
15. show();
16. }
17. private void show()

## B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE

```
18.    {
19.    {
20.        SqlConnection con = new SqlConnection("Data Source=.; uid=sa; pwd=wintell
    ect;database=Rohatash;");
21.        string strSQL = "Select * from UserDetail";
22.        SqlDataAdapter dt = new SqlDataAdapter(strSQL, con);
23.        DataSet ds = new DataSet();
24.        dt.Fill(ds, "UserDetail");
25.        con.Close();
26.        GridView1.DataSource = ds;
27.        GridView1.DataBind();
28.    }
29.    }
30. }
31. }
```

Now run the application.



	UserID	UserName
Select	100	Rohatash
Select	100	Rohatash
Select	4	Manoj
Select	8	Jorj
Select	9	yong

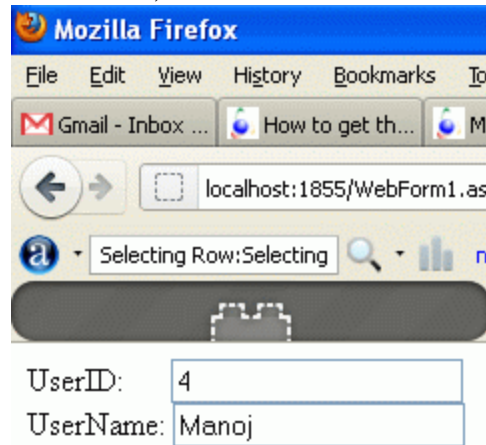
### Selecting Row

Selecting the row event is fired when you make a click on the select link. If you need any particular item in that row you can easily select it using the cells property. In the Gridview, double-Click on the SelectedIndexChanged Event and write the following code:

## B.Sc - IT/CS DEPARTMENT OF RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE

```
1. protected void GridView1_SelectedIndexChanged(object sender, EventArgs e)
2. {
3.     TextBoxUserID.Text = GridView1.SelectedRow.Cells[1].Text;
4.     TextBoxUserName.Text = GridView1.SelectedRow.Cells[2].Text;
5. }
```

Now run the application and select a row; that will show the selected row data in the TextBoxes.



	UserID	UserName
Select	100	Rohatash
Select	100	Rohatash
Select	4	Manoj
Select	8	Jorj
Select	9	yong