Glossary

.NET Framework

It is an internal Windows component that supports the execution of applications created by using various programming languages, such as Visual C#, Visual Basic, Visual F#, and Visual C++. These programming languages are a part of Visual Studio .NET. The .NET Framework consists of a virtual execution system called the common language runtime (CLR) and a set of class libraries.

CLR

CLR stands for common language runtime, which is one of the most important components of .NET Framework, better known as the runtime. It provides functionalities, such as memory management, exception handling, debugging, security, thread execution, code execution, code safety, verification, and compilation. CLR can host a variety of languages and ensures interoperability between their code. CLR also supports services that the application uses to access various resources, such as collections, arrays, and operating system folders. The runtime automatically releases the resources when they are no longer in use. This automatic memory management resolves the issue of memory leaks and invalid memory references.

CTS

CTS stands for Common Type System, which is an integral part of the runtime and helps to support cross-language communication. It specifies certain guidelines for declaring, using, and managing types at runtime.

Managed code

Code that is executed directly by the CLR. Therefore, the applications that are created using managed code automatically have CLR services, such as type checking, security, and automatic garbage collection. These services help to provide platform and language independence for managed code applications.

Metadata

A metadata is the self description of a program in the binary format. In other words, metadata contains the information of the classes, methods, and other elements used in a program. This information is stored in a CLR Portable Executable (PE) file or in the memory. When you compile the code in a PE file, the metadata is inserted into one part of the file, while the code is converted into IL and inserted into the other part of the file. The metadata describes every data type and member of your program. When the code is in the run mode, the CLR loads the metadata into the memory and finds information about the classes and members.

Global Assembly Cache

Global Assembly Cache (GAC) is the machine-wide code cache in a computer, which is the centre place for registering assemblies. You should note that the assemblies must be made sharable by registering them in the global assembly cache, only when needed; otherwise, they must be kept private. Moreover, it is not mandatory to install the assemblies in the GAC in order to make them accessible to the Component Object Model (COM) interop or unmanaged code. The COM interop is a service that enables .NET Framework objects to communicate with COM objects.

Framework Class Library (FCL)

It is a huge library of reusable types meant to be used by managed codes. It is an object-oriented library that is used in component-based applications. The FCL is made up of a hierarchy of namespaces that expose classes, structures, interfaces, enumerations, and delegates.

Windows Forms

Windows Forms is the graphical representation of any window displayed in an application. It is included as a part of Microsoft's .NET Framework. You can create Windows Forms applications in any CLR supported language. A Windows Form is used either to accept input from a user or to display information to the user. In Windows Forms, you add controls to the forms and raise events, such as mouse-click, which is handled by event handlers in an application.

ASP.NET

ASP.NET is a Web development model, which is used to deliver interactive, data-driven Web applications over the Internet. It also consists of a large number of controls, such as text boxes, buttons, and labels for assembling, configuring, and manipulating code to create Hyper Text Markup Language (HTML) pages. You can create ASP.NET application using any CLR compliant language, such as Visual Basic and C#.

ADO.NET

Microsoft ActiveX Data Objects.NET (ADO.NET), a part of Microsoft .NET Framework, allows your application to easily communicate with file-based and server-based data sources. ADO.NET libraries appear under the `System.Data` namespace in the .NET Framework. The most important use of ADO.NET is to communicate with the data and provide data access services to the .NET Framework. It provides access to various kinds of data, such as XML, relational, and application data. In addition to data access, it supports creation of front-end database clients and middle-tier business objects.

WF

WF is a technology introduced by Microsoft, which provides a programming model for building workflow based applications on Windows. It includes activities, workflow runtime, workflow designer, and a rules engine. WF introduced in Framework 3.0 and we see lots of improvement in the Framework 3.5 and 4.0 version.

WPF

Microsoft's WPF (formerly code-named as Avalon) provides the base for building applications and a clear separation between the user interface and the business logic. WPF helps in building interfaces that include documents, media, two and three-dimensional graphics, animations, and Web-like characteristics. It helps in creating Windows client applications with visually stunning user experiences. You can use WPF for creating both standalone and browser-hosted applications. Some examples of WPF applications are the Yahoo Messenger, the New York Times Reader, and the Contoso Healthcare Sample Application. WPF introduces Extensible Application Markup Language (XAML), which is a language based on XML, to design interfaces of WPF based applications. Even Microsoft comes with Expression studio that uses XAML for developing user interface for any Web- or Windows-based application.

WCF

WCF (formerly code-named as Indigo) is one of the new technologies introduced by Microsoft in .NET Framework 3.0 for building and running connected systems. WCF is a service-oriented technology for developing applications. The service-oriented design results in a distributed system that runs between the services and clients. If you are familiar with concepts, such as Web services, remoting, distributed transactions, and message queuing, then understanding WCF becomes easier since it is an enhanced technology for providing the same functions with better features with minimum time taken to develop the distributed system.

WCF-based applications are interoperable with any process as these communicate through Simple Object Access Protocol (SOAP) messages. When a WCF process connects with a non-WCF process, it uses XML-based encoding for SOAP messages, but when it connects with another WCF process, the SOAP messages are encoded into a binary format.

Windows CardSpace

Windows CardSpace (WCS) is a client software that helps in sharing personal information on the Internet. It helps programmers in developing websites and software that are less prone to identity related attacks, such as phishing. WCS helps in reducing the problems of traditional online security mechanisms, as it employs a separate desktop and cryptographically strong authentication, instead using user name and password, for authentication purposes. It

facilitates secure online transactions, such as online shopping, banking, and bill payment.

WCS only works with websites that accept the CardSpace logins. When you try to access the resources on the website, which accepts CardSpace, a CardSpace selection window appears. You can select the required card and access the resources on the website.

LINQ

LINQ is a component of .NET Framework 3.5. The basic function of LINQ is to add native data-querying capabilities to .NET Framework using syntax similar to SQL. LINQ allows you to define statements that interrogate a data source to yield a requested result set. In addition, it allows you to obtain and manipulate the data in a consistent manner.

Dynamic Language Runtime

The DLR feature is introduced in .NET Framework 4.0 to implement dynamic languages, such as Python and Ruby. This implies that the DLR is a runtime environment that adds services to the CLR to implement dynamic languages.

Managed Extensibility Framework

The Managed Extensibility Framework (MEF) is a new library included in the .NET Framework 4.0 version to create lightweight and extensible applications. MEF also helps in reusing the extensions not only within but across the applications also. The System.ComponentModel.Composition namespace is included in .NET Framework 4.0, which provides classes that constitute the core of MEF.

Parallel Computing

Parallel computing is a process in which a large problem is divided into small units and then these units are solved parallelly. Sometimes personal computers and workstations have two or three Central Processing Units (CPUs) that enable multiple threads to be executed simultaneously. In such a situation, a new programming model is provided by .NET Framework 4.0 that allows the developers to develop multithreaded and asynchronous code. This new programming model is known as parallel computing, which is supported by new runtime, class library types, and diagnostic tools. In parallel computing, you need to distribute the code for execution across multiple processors.

ADO.NET Entity Framework

The Entity Framework of ADO.NET allows you to focus on data through an object model instead of a relational model. The Entity Framework allows you to write less data access code, reduce maintenance, make the structure of data user friendly, and facilitate the persistence of data. You can use the Entity Framework in your application with directly using a new data provider called `EntityClient` and a new language called `Entity SQL`. The `EntityClient` uses the `EntityConnection` and `EntityCommand` objects to return a `DbDataReader`. You can also employ `Object Services` by using either `ObjectQuery` object with `Entity SQL` or LINQ to Entities.

ASP.NET AJAX

ASP.NET AJAX is an approach that works on the concept of Page Update rather than Page Replacement. The ASP.NET AJAX provides two ways to developers to build rich Web applications, server-centric and client-centric. The server-centric development approach enables a part of the Web page to be refreshed without posting the entire page to the server. This is done by using the UpdatePanel server control. The client-centric approach allows developers to use control extenders to add pre-defined client-side behavior to the new and existing ASP.NET controls. The extenders use a block of JavaScript code to add new and enhanced capabilities to ASP.NET.

Silverlight

Silverlight is a Web-based technology that allows Web designers and developers to stretch the boundaries of Web application development. It is an integration of the rich user interface of desktop applications and the transparency of other Web languages, such as HTML and JavaScript. Silverlight allows you to develop Web applications that contain high-fidelity multimedia content and eye-catching visual effects. Web designers and developers have already embraced Silverlight for building high-quality interactive Web applications.

MEF

MEF is a new technology that provides a new library to the .NET Framework 4.0. It extends a composable system with the components that are deployed by third parties.

Dynamic Data type

The dynamic data type is a new data type introduced in C# 2010 that supports the late binding. This implies that all the member functions and member variables such as fields, methods, and properties that are associated with dynamic keyword are ignored at compile time. The compiler preserves all the information about the expression, such as data type, which is used later to evaluate the expression at runtime.

DataGrid

DataGrid is a control that displays the data from a database table in a grid format. It displays the data in the form of rows and columns. In addition, it generates columns automatically, on the basis of the type of the data stored in them, by setting the ItemsSource property. You can perform various operations, such as grouping, sorting, and filtering, on the data stored in the DataGrid control. Moreover, the DataGrid control enables you to perform validation on the cell and row levels.

Calendar Control

Calendar is a control that enables you to select a date from a visual calendar. It can be used as a separate control or the drop-down part of the DatePicker control.

DatePicker

DatePicker is a control that enables you to select a date by either typing it into a text field or using a drop-down Calendar control.

The Grid Control

The most common container control is the `Grid` control. Whenever you add a new window to a WPF application, by default the window automatically includes a `Grid` control. A `Grid` control consists of a single cell. You can add additional rows and columns inside the `Grid` control. The child elements in the `Grid` control are arranged in columns and rows in the display area. A `Grid` control can also contain another `Grid` control as its child element.

The UniformGrid Control

The `UniformGrid` container control arranges the content in the Grid control in such a way that the cells in the grid remain the same size. When you add different controls on the `UniformGrid` control, all the controls get arranged in a grid pattern. The cells in the `UniformGrid` control adjust to accommodate different controls.

The Canvas Control

The `Canvas` control is the simplest container control that supports absolute positioning of its child controls by using the coordinates that are relative to the `Canvas` control. When you place a control in the `Canvas` control, you must specify one horizontal property and one vertical property to specify the starting position of the child control inside the Canvas control.

Collection

Collection allows you to store different types of data as well as add, remove, or modify the individual elements of these data types. It also provides automatic memory management and capacity expansion. Built-in collection classes had been introduced in .NET Framework 1.1, which have been retained in the latest version as well. The System.Collections namespace provides various classes, such as ArrayList and Stack, to easily manipulate the data. Some collections classes also provide the search and sort capabilities on the basis of index values or associated key values of the elements.

Generics

Generics are a type of fixed data structure, which can be reused by passing different types of parameters to it. The concept of generics has been introduced in .NET Framework 2.0 to overcome this limitation of collections. Generics permit classes, structs, interfaces, delegates, primitive types, and methods to be passed as parameters to store and modify data. In other words, generics are used to provide the reusability of code in software components. In .NET, you can create generic collection, which can handle collections of items of any data type in type safe manner by using various generic classes, such as List<T> and Dictionary<TKey>.

Thread

A thread is a sequence of instructions that must be executed in a particular order to perform a

specific task within a program. A program in execution is called a process and a thread can be a part of the executing program. A process executes in the operating system (OS); whereas, a thread executes within a process.

Therefore, threads are also called as light-weight processes. One major difference between threads and processes is that processes are fully isolated from each other. It means that the functioning of one process does not affect the functioning of another process that is running in the OS. However, threads can share memory with other threads running in the same application.

Multithreading

The Multithreading feature enables you to have more than one execution path for your application at a time. You are already aware of Windows multitasking facilities, which allow you to perform more than one task at the same time. In multitasking, you can perform multiple tasks simultaneously, such as writing an article using MSWord, listening a song in Window Media Player (WMP), and downloading a movie using the Internet Download Manager. In the same way, you can use multithreading to execute different methods of a program simultaneously.

XML

Extensible Markup Language (XML) is a simple and flexible text format widely used to exchange different types of data on the Web. XML plays a very significant role with respect to .NET Framework, as it provides a namespace, `System.Xml`, which includes classes required to work with XML. You can import this namespace to use XML in the .NET applications.

Directories

Directories are the entities that act as a store house for files. C# provides two classes, `Directory` and `DirectoryInfo`, to work with directories. The `Directory` class is one of the most important classes for working with directories. This class exposes static methods that help in creating, moving, and enumerating directories and subdirectories. The `Directory` class is also used to set date and time information of when a directory was created and accessed.

File Compression

File compression is one of the important techniques of compressing files. You can use the `GZipStream` class to compress and decompress files. This class offers the gzip data format, which uses certain industry standard algorithms for file compression and decompression without any loss of data and information.

Workflow

A workflow is a collection of actions that presents the model of a process. Each action represents an activity. These activities are stored as a model to describe a real world process. A workflow provides a way to describe the order of the execution of a long running process. In Visual Studio 2010, a workflow instance is created and maintained by an in-process runtime engine.

Host Process

A host process is an executable program that hosts a workflow. It may be a Windows application, a Web application, or a Web service application. A host process interacts with a workflow by using the WorkflowInvoker class and invokes the workflow as a method. The workflow is hosted and run within the host process. You can use Web services in the host process or remoting to enable other applications to communicate with the workflow.

Base Activity Library

The base activity library is a collection of standard actions, known as activities, used to create workflows. The action that an activity implements can be a simple action, such as a delay, or it can be a composite activity that consists of multiple child activities. The activities contained in the base activity library provide functionalities for control flow, state management, event handling, and communicating with applications and services. Each activity has an activity execution context, which represents the execution environment of the activity.

Runtime Engine

The runtime engine of WF provides the basic functionality required to execute and manage the workflow lifetime. It interacts with the host process and is responsible for executing each workflow instance. A host process can interact with multiple runtime engines simultaneously, where each engine executes multiple workflow instances.

Filtering Operators

Filtering, as the name suggests, refers to the operation of filtering the result set so that it contains only those elements that satisfy a specified given condition. The `where` clause is used as the Filtering operator in LINQ.

Projection Operators

The Projection operators are used to transform an object into a new object of a different type. You can construct a new object of a different type that is built from each object by using the Projection operators. The `Select()` and `SelectMany()` methods are the Projection operators used in LINQ.

Sorting Operators

The Sorting operator in LINQ arranges the elements of a sequence based on one or more attributes. This implies that for using the Sorting operator, first you need to specify a particular attribute and perform primary sorting on the elements. Then you need to specify the second sort criterion and sort the elements within the primary sorted group. The different Sorting operators are the `orderby`, OrderByDescending, ThenBy, ThenByDescending, and Reverse clauses.

Join Operators

The Join operators in LINQ are used to join objects in one data source with objects that share a common attribute in another data source. The `Join` and `GroupJoin` clauses act as the Join operators in LINQ. The `Join` clause implements an inner join, which is the type of join in which only those objects that have a match in other data set are returned. The `GroupJoin` clause joins two sequence based on a key selector functions and groups the results.

Grouping Operator

The Grouping operator in LINQ is used to put data into groups so that the elements in each group share a common attribute. The `group by` clause is the Grouping operator used in LINQ and it returns a sequence of `IGrouping<TKey, TElement>` objects that contain zero or more items that match the key value for the group.

Quantifier Operators

The Quantifier operators return a Boolean value if the elements of the sequence satisfy a specific condition. The clauses that are classified as the Quantifier operators are `Any`, `All`, and `Contains`. The `Any` clause determines whether any element in a sequence satisfies a condition. This clause enumerates the source sequence and returns `True` if any element satisfies the condition. The enumeration of the source sequence is terminated as soon as the result is known.

Partitioning Operators

The Partitioning operators in LINQ are used to divide an input sequence into two sections, without rearranging the elements, and then returning the result set with one of the sections that satisfies the given condition. The `Take`, `Skip`, `TakeWhile`, and `SkipWhile` clauses are referred to as the Partitioning operators.

Set Operators

The Set operators in LINQ refer to the query operations that produce a result set. The result is based on the presence or absence of the equivalent elements that are present within the same or separate collection. The `Distinct`, `Union`, `Intersect`, and `Except` clauses are classified as the Set operators.

Element Operators

The Element operators in LINQ return just one element. The `ElementAt`, `ElementAtOrDefault`, `First`, FirstOrDefault, Last, LastOrDefault, Single, and SingleOrDefault clauses are termed as the Element operators.

Aggregate Operators

The Aggregate operators compute a single value from a collection. The different Aggregate operators are `Aggregate`, `Average`, `Count`, `LongCount`, `Max`, `Min`, and `Sum`. The Aggregate clause calculates a sum value of the values in the collection.

Conversion Operators

The Conversion operators convert a collection to an array. A Conversion operator changes the type of input objects. The different Conversion operators are `ToSequence`, `ToArray`, `ToList`, `ToDictionary`, `ToLookup`, `OfType`, and `Cast`.

Generation Operators

The Generation operators, which are `DefaultIfEmpty`, `Empty`, `Range`, and `Repeat`, help in creating a new sequence of values. The `DefaultIfEmpty` clause replaces an empty collection with a default single collection. The `Empty` clause refers to an empty collection. The `Range` clause generates a collection that contains a sequence of numbers.

Zip Operator

The Zip operator is a new operator introduced in LINQ in ASP.NET 4.0, which helps in combining two collections into one. It uses the Zip() method to merge the two sequences, which are identical in length, into one.

The LinqDataSource Control

The `LinqDataSource` control enables you to use LINQ in an ASP.NET Web page by setting the properties in the markup text. You can use the `LinqDataSource` control to create a Web page that retrieves or modifies the data. You can simplify the code in a Web page by enabling the `LinqDataSource` control to automatically create the commands for interacting with the data. The `LinqDataSource` control is similar to the `SqlDataSource` and `ObjectDataSource` controls in the respect that it can be used to declaratively bind other ASP.NET controls on a page to a data source. The difference is that instead of binding directly to a database or to a generic class, the `LinqDataSource` control is designed to bind a data model that is LINQ enabled.

Anonymous Methods

Anonymous methods allow you to handle an event rather than having a separate event handler. For example, if you want to display a message box at the click of a button, you can handle it in a standard way with a delegate and an event handler or you can also perform the action using an anonymous method. To put it simply, an anonymous method is a block of code used as a parameter for the delegate. The main benefit of using anonymous method is to reduce the code that you need to write. When you are using the anonymous method, you do not need to define any static event handler. The anonymous method is defined at the time the caller is handling the event.

Lambda Expressions

In .NET Framework 3.5, a new syntax, named `Lambda` expressions, was introduced for anonymous methods. A `Lambda` expression is an anonymous function that can contain expressions and statements to create delegates or expression tree types. The `Lambda` expressions use the `Lambda` operator =>. The left-hand side of the `Lambda` operator specifies the input parameter, and the right-hand side specifies the expression or the statement, shown as follows: (input parameter) => expression

PLINQ

Parallel LINQ (PLINQ) is the parallel implementation of LINQ, in which a query can be executed by using multiple processors. PLINQ ensures the scalability of software on parallel processors in the execution environment. It is used where data grows rapidly, such as in telecom industry or where data is heterogeneous. PLINQ also supports all the operators of LINQ. In addition, you can query collections by using PLINQ. It can also run several LINQ queries simultaneously and makes use of the processors on the system. Apart from this, PLINQ uses parallel execution, by which the queries run quickly.

The GridView Control

The `GridView` control is a data bound control that displays the values of a data source in the form of a table. In this table, each column represents a field and each row represents a record. The `GridView` control exists within the `System.Web.UI.WebControls` namespace.

The DataList Control

The DataList control is a data bound control that displays data by using templates. These templates define controls and HTML elements that should be displayed for an item. The `DataList` control exists within the `System.Web.UI.WebControls` namespace.

The DetailsView Control

The `DetailsView` control is a data bound control that is used to display a single record from the associated data source in a table format, where each row of the table represents a field of the record. The `DetailsView` control uses the `DataSourceID` property to support two-way

binding; consequently, it also supports insert, update, and delete operations.

The FormView Control

The `FormView` control displays a single record from the associated data source. Each row of the table displays each field of the record. For the `FormView` control to display content, you need to create templates for the different parts of the control. You must create a template for the mode in which the control is configured, even though most of the templates are optional.

The ListView Control

The `ListView` control is a data bound control used to display data from the associated data source. The `ListView` control enables a developer to display data in any format using templates and styles. This control provides excellent customization and extensibility features. It provides the developer with control on the rendered HTML output and provides support for adding new row, and sorting.

The Repeater Control

The `Repeater` control is a data bound control that is used to display repeated list of items from the associated data source. It is a single Web control that allows splitting markup tags across the templates. The `Repeater` control does not provide support to in-built layout or styles. Therefore, while working with the `Repeater` control, you have to explicitly declare all layouts, formatting, and style.

The DataPager Control

The `DataPager` control is used to provide paging functionality to the data bound controls, such as the `ListView` control.

The Chart Control

The Chart control enables you to present complex data in a simpler way. It helps you to analyse the data in an easier way with the help of charts.

The QueryExtender Control

The QueryExtender control is a data source control that helps in retrieving only those records from the database which meet the specific criteria or in other words, to filter the records. Filtering of records helps in displaying the records in various ways on the Web page without affecting the dataset.

The SqlDataSource Control

The `SqlDataSource` control is a data source control that allows a server control, such as the `GridView` control, to access data located in a relational database, such as Microsoft SQL Server and Oracle. This control also generates a connection string to interact with the data in an ASP.NET page. These connection strings are generated through the `Configure Data Source` wizard.

The AccessDataSource Control

The `AccessDataSource` control is a data source control that allows a Web server control to access a Microsoft Access database. This control does not support connection strings, but the `DataFile` property of this control allows you to specify the Access file (`.mdb` file) that you want to use to access the data.

The ObjectDataSource Control

The `ObjectDataSource` control represents the business objects and allows you to use the `ObjectDataSource` control in conjunction with a data-bound control to display, edit, and sort the data on a Web page with little or no code. This control is used to create Web applications that rely on the middle-tier objects to manage data.

The XmlDataSource Control

The XmlDataSource control allows a data bound control to bind the data from a XML document. This control also supports `XPath` expressions that allow you to return only certain nodes from the XML document.

The EntityDataSource Control

The EntityDataSource control helps you to connect with ADO.NET Entity Data Model (EDM) to perform various database operations, such as create, insert, remove, and update.

The SiteMapDataSource Control

The `SiteMapDataSource` control allows you to work with the data stored in the `SiteMap` configuration file. This control also provides with the capability to customize site navigation by using site map data with data server controls, irrespective of the fact that controls are not navigation controls.

Data binding

Data binding refers to the process of binding Windows or Windows Presentation Foundation (WPF) controls to data sources. It is done to display and access the data stored in a data source directly by an application. In other words, the control bound to a data source displays the information and data stored in that data source. For example, to display the names of the employees of a company in a ListBox control, you need to bind this control to the table containing the required data. In Windows Forms applications, you can bind controls not only to data sources, but also to any programming construct that holds the data. For example, you can bind control properties to an array of values or an Excel file. These control properties can be bound to a data source to retrieve data from it.

Database

Database in simple terms means a collection of structured information. It stores information in an organized and structured manner to facilitate the retrieval as and when required. Databases have become more complex over the years; however, the fundamental concept is still a simple one. DataTable consists of DataRow and DataColumn and stores data in the table row format. The DataTable is the central object of the ADO.NET library and similar to a table in a database. You should note that the DataTable objects are case-sensitive. For example, the EmployeeDetails table is not the same as the employeedetails table. A DataSet can contain two DataTable objects that have the same TableName property and different Namespace property values. The NewRow() method is used to add a row to a DataTable. The maximum number of rows that a DataTable can contain is fixed at 16,777,216. The DataTable also contains a collection of Constraint objects that is used to ensure the integrity of data.

DataView

DataView represents a customized view of DataTable for sorting, filtering, searching, editing, and navigation. A DataView allows you to create a view on a DataTable to see a subset of data based on a preset condition specified in the RowStateFilter property. A DataView can be used to present a subset of data from the DataTable.

DataColumn

DataColumn consists of a number of columns that comprises a DataTable. A DataColumn is the essential building block of the DataTable. The DataType property of DataColumn determines the kind of data that a column holds. You can also bind a control to a particular column in the DataTable.

DataRow

DataRow represents a row in the DataTable. You can use the DataRow object and its properties and methods to retrieve, evaluate, insert, delete, and update the values in the DataTable. You can use the NewRow() method of the DataTable to create a new DataRow and the Add() method to add the new DataRow to the DataTable. You can also delete DataRow by calling the Remove() method.

DataRelation

DataRelation allows you to specify relations between various tables. In simple words, a DataRelation is used to relate two DataTable objects to each other through DataCoulmn objects. The relationships are created between matching columns in the parent and child tables. For this, the DataType value of both columns must be the same. The DataRelation objects are contained in a DataRelationCollection, which you can access through the Relations property of the DataSet.

DataReader

In ADO.NET, `DataReader` is used to sequentially read data from a data source. A `DataReader` allows to access information on a single row of data at a given time having a single record loaded in the memory, regardless of the size of the result set. A `Command` object is required to retrieve data from the database using the `DataReader`. After creating the `Command` object, create a `DataReader` by calling the `Command.ExecuteReader` to retrieve the rows from a data source. You should use the `Read()` method of the `DataReader` object to obtain a row from the results of the query. The `DataReader` is a good choice when you are retrieving large amounts of data because data does not store in the cache memory. Call the `Close()` method every time you finish using the `DataReader` object. This is necessary, because if your `Command` contains return values,

they will not be available until you close the `DataReader` object.

EDM

Entity Data Model (EDM) defines the conceptual entities that can be read in a serialized form using a DataReader.

Entity SQL

Entity SQL defines a common SQL based query language that is extended to express queries in terms of EDM concepts. It is supported by both the EntityClient and Object Services.

EntityClient

Provides a gateway for the queries of the entity-level queries, which is queried through a common Entity SQL language.

Object Services

Object Services allow you to interact with a conceptual layer through a set of CLR classes. Object Services also provide support for Entity Framework by providing services, such as state management, identity resolution, and query building support.

Stored Procedures

Stored procedure is a precompiled set of SQL commands that are stored on a database. It is compiled once and used repeatedly by client applications without sending it to the database server.

Triggers

Trigger is a special type of stored procedure that is fired when you modify the data of a database table using one or more data modification operations, UPDATE, INSERT, or DELETE. Note that a trigger can never be called or executed, but it is fired automatically when data is modified in the associated tables.

User-Defined Aggregate Functions

In SQL, an aggregate function is a function that performs a calculation on a set of values and returns a single computed value. It summarizes the data in a table. SQL Server provides some built-in aggregate functions, such as SUM, MAX, and COUNT.

POCO

Plain-Old Common Language Runtime Objects (POCO), which allows the developers to create classes that are not only simple, but also can be included in the Entity Framework. In the previous versions of .NET Framework, the classes that derived from a model are inherited from the EntityObject class. This was a hindrance for the developers if they wanted to create simple classes that are not bound to a Framework. POCO helps the Entity Framework to use custom data classes along with the data model without making any changes to the data classes. In simpler language, if you have a class called Employee with name, address, and phone number, then the Entity Framework will not change the class while implementing it. This implies that the class will not include EntityKey or any other property, which requires you to implement any interface specific to Entity Framework while creating an application.

Data modeling

Data modeling is a technique to structure and organize data that are needed to support an application. For example, an entity in an application must be represented by data, such as customers, orders, and products, whereas, a relationship specifies the logical connection among the entities.

Entity Type

An entity type in EDM is the design template for a data type in the application domain. In a schema, the entity type has a name, properties, and a key that identifies the data type when instantiated by an application code.

EntityClient Provider

The EntityClient provider is an ADO.NET-managed provider that supports retrieving data described in an EDM. Entities are combined together with other related entities that can have different types of relationships among them. The EntityClient provider allows a user to query data described in a conceptual model. It is capable of returning results such as scalar results, result sets, and object graphs.

ADO.NET Metadata

ADO.NET Entity Framework uses ADO.NET metadata that provides infrastructure and type hierarchy to exhibit the conceptual entities, fundamental tables, and CLR classes according to EDM. ADO.NET uses the `System.Data.Metadata.Edm` namespace to

manage the metadata services for Entity Framework.

Windows Installer Deployment Technology

Windows Installer, also popular as Microsoft Windows Installer (MSI), is a software installation and configuration service that is used to install, maintain, or remove a software application on Windows operating systems. This service contains the information of every application that it installs. The Windows Installer service verifies an application at three levels—product, feature, and component. The product level contains the complete application as a product that a user can install; the feature level contains one or more functionalities of the product; and the component level contains one or more files that are logically related in the application.

ClickOnce Deployment Technology

ClickOnce is a deployment technology that allows you to publish Windows-based applications on a Web server or network file share location. Visual Studio 2010 provides the full support for the ClickOnce deployment technology to publish and update the applications. The ClickOnce deployment technology allows you to deploy the self-updating Windows-based applications. It also supports the deployment of WPF XAML Browser Applications (XBAPs).

XAML BrowserApplications (XBAPs)

XBAP is a Windows technology used to create rich Internet applications just similar to WPF applications. Note that WPF is a Windows application that does not support Web browsers; whereas an XBAP application can be easily host on a Web browser.

File System Editor

The File System editor allows you to add the project outputs, files, assemblies, and other items to a deployment project and specify the target location where these files will be deployed. It displays a standard set of deployment folders by default. You can add your own folders and special folders with the editor. In addition, you can specify the desktop and Start menu shortcuts for the application within the File System editor. The files that are part of the deployment must be referenced from within the File System editor.

Registry Editor

The Registry editor specifies registry keys and data values that you want to add to the registry of the target computer. When you open the Registry editor for the first time, it displays a standard set of registry keys.

File Types Editor

The File Types editor establishes association between files and applications. For example, when you double-click the file having the .doc extension, it opens in MS Word.

User Interface Editor

The User Interface editor specifies properties for a set of predefined dialog boxes. It is helpful when you want to add extra information that is displayed during the installation process. The User Interface editor is divided in two sections, Install and Administrative Install. The Install section contains the dialog boxes that are displayed when the user runs the installer; whereas the Administrative Install section contains the dialog boxes that are displayed when the system administrator uploads the installer to a network location.

Custom Actions Editor

The Custom Actions editor allows you to define custom actions that take place at the end of the installation process on the target computer. Custom actions are created in advance and must be compiled as a .dll or .exe file, or added to the project as a script or assembly.

Launch Conditions Editor

The Launch Conditions editor allows you to specify certain conditions that must be met to continue the installation. For example, if you want to install an application on a specific version of the Windows operating system, the installation first checks the presence of specific conditions and runs the process only when the conditions are met. Otherwise, the installation is terminated at that point.

XCopy Deployment

XCopy is a term used to deploy an application by copying a set of assemblies to a folder on a target computer and then executing the application on the target computer. The application starts executing on the target computer by using its assembly file, which is a self-description file that

contains all the information about the application. The XCopy deployment does not make any impact on the target system by way of configuring the components and registering entries, and therefore known as *zero-impact* installation.

Web Services Directory

The Web service directory provides a central location for Web service providers. In other words, it is a place where Web service users (consumers) can easily locate services offered by other companies and organizations. It is difficult to search a Web service without using any resource. Web service directories provide those centralized locations where you can publish information about your Web service. A Web service can be a directory for other Web services and can provide the result of the query to the Web service client.

Web Services Discovery

The Web service discovery provides the capability to locate Web services. This component locates documents that describe a particular Web service, through WSDL. If you want to use a Web service in your website, you first need to know where the Web service is located. On the other hand, if you want other people to use your Web service, they also need to know where your Web service is located.

Web Services Description

The Web service description provides the description of a Web service in the XML format, known as WSDL. A WSDL document defines message formats, such as HTTP-GET and HTTP-POST, to facilitate communication between a Web service and a client. A message pattern determines how the messages can be exchanged between a client and a service.

Web Services Wire Format

The Web service wire format provides an interface to transmit data between a Web service and its users through protocols, such as HTTP-GET and HTTP-POST. A Web service can use any Remote Procedure Call (RPC) protocol, such as DCOM or Common Object Request Broker Architecture (CORBA), to transmit data. However, these types of protocols are not recommended for creating universally available Web services. Therefore, you need to work with frequently used protocols, such as Hypertext Transfer Protocol (HTTP) and SOAP, to transfer data between a Web service and its users.

Simple Object Access Protocol

SOAP, one of the key elements in a Web service, is a protocol used for messaging. It is completely XML-based. SOAP specifications are standardized by W3C and are used by most companies that develop Web services. SOAP provides a complete set of rules for messages, called SOAP envelopes, as well as the rules for issues, such as data encoding, message handling, and message binding, to other protocols, such as HTTP. One of the great advantages of using a SOAP message is that it is not restricted to any particular protocol. However, the HTTP protocol is most commonly used since it is not restricted to any one operating system.

The WebService Directive

The WebService directive defines Web service-specific attributes, such as Namespace and Description attributes, used by the ASP.NET compilers and parsers.

Silverlight

Silverlight is a Web-based technology that allows Web designers and developers to stretch the boundaries of Web application development. It is an integration of the rich user interface (UI) of desktop applications and the transparency of other Web languages, such as HTML and JavaScript. Silverlight allows you to develop Web applications that contain high-fidelity multimedia content and eye-catching visual effects. Web designers and developers have already embraced Silverlight for building high-quality interactive Web applications.

The Login Control

The Login control provides a user interface, which authenticates users and grant them access to the desired services or resources on the basis of the credentials, such as user name and password, filled in by them. As it can restrict the access to other Web pages, the Login control is generally placed in the Home page of a website.

The LoginView Control

The **LoginView** control is a Web server control, which is used to display two different views of a Web page of any website, depending on whether

the user has logged on to a Web page as a registered user or a visitor. The `LoginView` control provides a way of altering the look of the page or showing different content to different groups of users. This control has the built-in functionality to gather the current user's status and roles.

The LoginName Control

The `LoginName` control is responsible for displaying the names of all the authenticated users. It uses the `Page.User.Identity.Name` namespace to return the value for the user's name. If no user is logged in, then this control is not displayed on the page. The `LoginName` control is provided by the `LoginName` class.

The LoginStatus Control

The LoginStatus control specifies whether or not a particular user has logged on to the website. The text displayed on this control changes according to the login status of the user. When the user is not logged in, this control displays the Login text as a hyperlink that facilitates the user in navigating to the login page. To do this, the LoginStatus control uses the authentication section of the web.config file and retrieves the URL of the login page. The `LoginStatus` control displays the Logout text as hyperlink, when the user is logged on to the website, and deletes the authentication information as soon as the user clicks the Logout link.

The PasswordRecovery Control

The PasswordRecovery control is used to recover or reset the forgotten password of a user. This control does not display the password on the browser, but sends the password as an e-mail message to the e-mail address specified at the time of registration. This control also uses the Membership service to create or reset the password.

The ChangePassword Control

The ChangePassword control allows the users to change their password. This control prompts users to provide the current password first and then set the new password. If the old password is not correct, the new password cannot be set. This control also facilitates in sending e-mails to users about the new password. The `ChangePassword` control is provided by the `ChangePassword` class.

The CreateUserWizard Control

The CreateUserWizard control uses the Membership service to create a new user in the Membership data store. This control is an extended version of the already existing Wizard control. The CreateUserWizard control is provided by the CreateUserWizard class and can be customized by using templates and style properties.

The WebPartManager Control

The `WebPartManager` control acts as a central point that manages all the other Web Parts controls, functionality and events on a Web page. This control coordinates the interaction between Web Parts controls and the events that occur on a Web page. The WebPartManager control manages the personalization states of Web Parts controls on a Web page.

The ProxyWebPartManager Control

As you know, a Web Parts application (an application using Web Parts controls) can contain a single `WebPartManager` control for each Web page, which manages all the Web Parts controls on that page.

The WebPartZone Control

The `WebPartZone` control is used to create a region on the Web page where Web Parts or server controls can be relocated, maximized, or minimized according to the requirements specified by a user. The `WebPartZone` control is an object of the `WebPartZone` class that permits you to create zones on a Web page. After the zone is created, you can place any server control in the zone.

The CatalogZone Control

You can personalize a website by adding the catalog functionality, which allows you to manage Web Parts controls efficiently. You can create a catalog-enabled Web page by adding a `CatalogZone` control to it. A `CatalogZone` control helps to create a catalog of Web Parts controls. This allows a user to add or update the controls easily on a Web page.

The DeclarativeCatalogPart Control

The `DeclarativeCatalogPart` control is one of the three controls (the other controls are `PageCatalogPart` and `ImportCatalogPart`) used together with the

CatalogZone control. This control is used to modify the functionality of a Web page. This control displays a list of Web Parts controls and other server controls, when the website is in the CatalogDisplayMode mode. A user can add controls from this catalog onto the Web page.

The PageCatalogPart Control

The PageCatalogPart control is used with the CatalogZone control to add a specific functionality to the catalog section of a Web page. This control allows users to restore previously deleted WebPart controls of a Web page. This is the only way users have to restore the deleted Web Parts controls. The PageCatalogPart control adds a list of check boxes to the catalog zone controls corresponding to each deleted Web Parts control. To restore these controls on the Web page, users simply need to select the required check box and a Web part zone where the Web Parts control is to be added, and click the Add button.

The ImportCatalogPart Control

The ImportCatalogPart control imports the description file of a Web Parts control or server control that is used as a Web Parts control. This enables the server control to be added to a Web page with pre-assigned settings. The description file enables users to share the settings of Web Parts controls.

The EditorZone Control

Till now, you must have got some idea of important Web Parts controls. Now, suppose you want to change the appearance of a Web page. To do so, you have to go to the properties of the Web page to make the changes at design time. However, instead of this traditional approach, you can use a Web Parts control called EditorZone. The EditorZone control is one of the primary editing controls used to change the appearance, format, and structure of Web pages containing Web Parts controls at run time. In addition, you can use the EditorZone control to change the behavior and content of Web Parts controls.

The LayoutEditorPart Control

You can use the LayoutEditorPart control to create a full-fledged layout editor for Web Parts controls, which can have their respective LayoutEditorPart controls associated with them. To change the layout of a Web Parts control, simply make the required changes in the LayoutEditorPart control associated with the Web Parts control.

The AppearanceEditorPart Control

The AppearanceEditorPart control allows users to edit the properties associated with the appearance of a Web Parts control on the Web page. This control resides in an EditorZone zone and is visible only when a Web Parts control is selected for editing.

The PropertyGridEditorPart Control

The PropertyGridEditorPart control provides a UI that allows users to edit the custom properties of a Web Parts control on a Web page. This control resides in the EditorZone control on the Web page. The PropertyGridEditorPart control is visible only when the Web page is in the edit mode or when the user selects a Web Parts control for editing.

The BehaviorEditorPart Control

The BehaviorEditorPart control, as the name suggests, provides options that enable users to modify the behavior of Web Parts controls. This control usually remains hidden and is visible only when a Web page containing Web Parts controls is in the edit mode.

The ConnectionsZone Control

You can dynamically connect Web Parts controls with each other that reside in the WebPartZoneBase zone with the help of the ConnectionsZone control. However, to do this, the Web Parts controls must be enabled for such type of dynamic connection. When you add the ConnectionsZone control to a Web page, a new mode, the connection mode, appears in the Select mode list of the Web page. The ConnectionsZone control is an object of the ConnectionsZone class.

The TreeView Control

The TreeView control is used for logically displaying the data in a hierarchical structure, similar to Windows explorer. You can use this control when you need to display the navigation menu for displaying the files and folders on the Web page. You can display an Extensible Markup

Language (XML) document and database records in a tree structure.

The Menu Control

The `Menu` control is another Navigation control, which is also used to display site navigation information. The `Menu` control can, however, display the site structure vertically as well as horizontally. It can be used as a databound control, for example, binding the menu control with a `SiteMapDataSource` control. It can also retrieve the data to be displayed from the items that are added to the `Items` collection of the Menu control at runtime. The Menu control supports binding data with hierarchical data source, such as XML files.

The SiteMapPath Control

The `SiteMapPath` control is also known as the breadcrumb Navigation control. Breadcrumb allows you to display the current page's context within the entire structure of a Web site. It also allows you to display links as a path back to the home page. It displays information regarding the user's current page along with the entire hierarchy of the pages, thereby enabling the user to navigate back to some other pages in the hierarchy.

WCF Data Services

WCF Data Services allow you to create and consume the OData services for the Web applications. OData services are based on the Open Data protocol, which is a new data-sharing standard. WCF Data Services provide client libraries that are supported by various technologies, such as .NET, Silverlight, Asynchronous JavaScript and XML (AJAX), Hypertext Preprocessor (PHP), and Java. In addition, these services support OData in SQL Server 2008, Windows Azure Storage, Excel 2010, and SharePoint 2010. The Data Services framework enables you to create flexible WCF Data Services that can be easily integrated with the Web applications. These WCF Data Services use Uniform Service Identifiers (URIs) to display data in various formats, such as JavaScript Object Notation (JSON) and Representational State Transfer (REST).