**LEVEL-3:DIFFICULT**

**C# .NET**

**70 INTERVIEW QUESTIONS**

**AND**

**ANSWERS**

**Done by Babika**

**KloudOne( 3rd Batch)**

**Q-1) What are three common acronyms used in .NET, and what do they stand for?**

Three frequently used acronyms in .NET are IL, CIL and CLI:

* **IL** stands for Intermediate Language, which is an object-oriented programming language that is a partially compiled code that .NET developers will then compile to native machine code.
* **CIL** stands for Common Intermediate Language, formerly known as Microsoft Intermediate Language (MSIL). This is another programming language that .NET developers use, and it represents the lowest possible level for a language that humans can still read.
* **CLI** stands for Common Language Infrastructure. This is a compiled code library that Microsoft developed as an open specification. Developers use CLI for security, versioning and deployment purposes.

**Q-2) Why do we use MSMQ?**

**Microsoft Message Queuing**, or MSMQ, is technology for asynchronous messaging. Whenever there's need for two or more applications (processes) to send messages to each other without having to immediately know results, MSMQ can be used. MSMQ can communicate between remote machines, even over Internet. It's free and comes with Windows, but is not installed by default.

This mainly addresses the common use case of asynchronous message processing: you have a service Service1 that communicates (send messages) with another part of your software architecture, say Service2.

Main problem: what if Service2 becomes suddenly unavailable? Will messages be lost? If you use MSMQ it won't: Service1 will send messages into a queue, and Service2 will dequeue when it is available.

MSMQ will resolve following common issues:

* Temporary unavailability of a service: messages are persisted on the disk and will be dequeued when the service becomes available again, so **no messages are lost**
* As it's fully asynchronous, it'll help a lot in case of **punctual peak load**: your Service2 won't die under the heavy load, it'll just dequeue and process messages, one after one.

**Q-3) What is .NET Standard?**

* .NET Standard solves the code sharing problem for .NET developers across all platforms by bringing all the APIs that you expect and love across the environments that you need: desktop applications, mobile apps & games, and cloud services
* .NET Standard is a set of APIs that all .NET platforms have to implement. This unifies the .NET platforms and prevents future fragmentation.
* .NET Standard 2.0 will be implemented by .NET Framework, .NET Core, and Xamarin. For .NET Core, this will add many of the existing APIs that have been requested.
* .NET Standard 2.0 includes a compatibility shim for .NET Framework binaries, significantly increasing the set of libraries that you can reference from your .NET Standard libraries.
* .NET Standard will replace Portable Class Libraries (PCLs) as the tooling story for building multi-platform .NET libraries.

**Q-4) What’s the difference between .NET and Laravel?**

Both have similar difference like,

* + - Languages (e.g. C# support in .NET versus Laravel’s support for PHP)
    - Security
    - Processing overheads
    - .NET’s integration with Visual Studio
    - Third-party libraries
    - Open-source community support

**Q-5) Explain the difference between Task and Thread in .NET?**

* **Thread** represents an actual OS-level thread, with its own stack and kernel resources. Thread allows the highest degree of control; you can Abort() or Suspend() or Resume() a thread, you can observe its state, and you can set thread-level properties like the stack size, apartment state, or culture. ThreadPool is a wrapper around a pool of threads maintained by the CLR.
* The **Task class** from the Task Parallel Library offers the best of both worlds. Like the ThreadPool, a task does not create its own OS thread. Instead, tasks are executed by a TaskScheduler; the default scheduler simply runs on the ThreadPool. Unlike the ThreadPool, Task also allows you to find out when it finishes, and (via the generic Task) to return a result.

**Q-6) What is the code to send an email from an ASP.NET application?**

|  |
| --- |
| mail message = new mail();  message.From = "abc@gmail.com";  message.To = "xyz@gmail.com";  message.Subject = "Test";  message.Body = "hello";   SmtpMail.SmtpServer = "localhost";  SmtpMail.Send(message); |

**Q-7) What are the event handlers that we have for the Global.asax file?**

* **Application Events:**

Application\_Start, Application\_End, Application\_AcquireRequestState, Application\_AuthenticateRequest, Application\_AuthorizeRequest, Application\_BeginRequest, Application\_Disposed, Application\_EndRequest, Application\_Error, Application\_PostRequestHandlerExecute, Application\_PreRequestHandlerExecute, Application\_PreSendRequestContent, Application\_PreSendRequestHeaders, Application\_ReleaseRequestState, Application\_ResolveRequestCache, Application\_UpdateRequestCache

* **Session Events:**

Session\_Start, Session\_End

**Q-8) How can we apply themes to an ASP.NET application?**

We can use the web.config file to specify the themes,

|  |
| --- |
| <cofiguration>  <system.web>  <pages theme="windows"/>  </system.web>  </configuration> |

**Q-9) What is the appSettings section in the web.config file?**

If we want to set the user-defined values for the whole applications, we can use the appSettings block in the web.config file. For example the code below uses the ConnectionString throughout the project for the database connection:

|  |
| --- |
| <em><configuration>  <appsettings>  <add key= "ConnectionString" value="server=local; pwd=password; database=default" />  </appSettings></em> |

**Q-10) What is JSON data, and what is one way that .NET developers can work with JSON?**

JSON (JavaScript Object Notation) provides developers with a way to organize and store data so it’s easy to access and read. JSON is important for developers because it allows them to manipulate JSON feeds from other sites and to load them more quickly and easily than via SML/RSS feeds. Json.NET provides a way for .NET developers to define classes that parse objects and arrays from JSON text. You can also use Json.NET if you need to serialize value types into JSON text. Json.NET runs on .NET2, .NET3 and .NET4.

**Q-11) When should you use .NET Web Forms over ASP.NET MVC?**

Traditionally, the .NET Framework has been based on Web Forms. This was essentially an effort to create web services using Microsoft’s existing Visual Studio Tools without forcing developers to learn new scripting languages. Web Forms still allows developers to create quick and simple applications, and some legacy systems may still run as Web Forms.

[ASP.NET MVC](https://docs.microsoft.com/en-us/previous-versions/aspnet/dd381412(v=vs.108)) is increasingly the standard for contemporary developers , MVC’s most important feature is that it allows applications to be broken down into discrete models, views and controllers, making them much easier to test during development.

**Q-12) Illustrate Serialization?**

A process that involves converting some code into its binary format is known as a serialization in C#. In doing so it gives the flexibility where code can be stored easily and or written to a disk or some other storage device. Serialization is used when there is a strict need for not losing the original code.

Serialization in C# is of three types:

**1.Binary Serialization-**It is fast and demands less space, converts any code into its binary format. Serialize and restore public and non-public properties.

**2.SOAP Serialization-**Generates a complete SOAP compliant envelope used by any system by its ability to understand SOAP.

Classes under this type of Serialization reside in System.Runtime.Serialization.

**3.XML Serialization-**It serializes all the public properties to XML document. Readability being one factor, an XML document can also be manipulated in various ways. Classes under this type reside in System.sml.Serialization.

**Q-13) Define Parsing? Explain how to Parse a DateTime String?**

Parsing is a method of converting a string into another data type.

**Example:**

string text = "200";

int num = int.Parse(text);

|  |
| --- |
|  |

Where, 200 is an integer. So, the Parse method converts the string 200 into its own base type. Also, int. to parse a DateTime String let’s see a small example.

**Example:**

|  |  |
| --- | --- |
|  | string dateTime = "Aug 26,2019"; Datetime parsedvalue = Datetime.Parse(dateTime); |

**Q-14) Define Delegate?**

Delegate is a variable that holds the reference to a method.

It is a function pointer of the reference type. All Delegates are derived from the  
System.Delegate namespace. Both Delegate and the method that it refers to can have the same signature.

**Example:**

|  |  |
| --- | --- |
|  | public Delegate int myDel(int number); //declaring a delegate  public class Program  {  public int SubstractNumbers(int a) //Class Program has the method same signature as delegate called  {  int difference = a - 10;  return difference;  }  public void start()  {  myDel DelegateExample = SubstractNumbers;  }  } |

**Q-15) Distinguish between System.String and System.Text.StringBuilder classes?**

System.String is immutable. Whenever you modify the value of a string variable then a new memory is allocated to the new value and previous memory allocation is released. System.Text.StringBuilder is mutable i.e. supports the fact that a variety of operations can be performed  
without allocating a seperate memory location for the already modified string.

**Q-16) Define thread? Explain about Multithreading?**

Thread is a set of instructions that when executed enables the program to perform concurrent processing. Concurrent processing helps in doing more than one process at a time. By default, C# consists of only thread. Other threads can be created to execute the code in parallel with original thread.

Thread follows a life cycle where it starts whenever a thread is created and gets terminated immediately after the execution. The namespace it follows is System.Threading which has to be included to create threads and use its members.

Threads are created by extending the thread class. Start() method marks the beginning of the thread execution.

|  |
| --- |
| //callThread is our target method//  ThreadStart methodThread = new ThreadStart(CallThread);  Thread childThread = new Thread(methodThread);  childThread.start(); |

C# can also execute more than proceses/task at a time which is done by handling different proceses at different time labled as “multithreading”.

Several operations of the same are listed below:

* **Start**
* **Sleep**
* **Abort**
* **Suspend**
* **Resume**
* **Join**

**Q-17) Explain Async and Await?**

Async and Await keywords are mostly used for creating asynchronous methods in [C#](https://www.microsoft.com/en-in/download/details.aspx?id=8193).

**Example:**

|  |  |
| --- | --- |
|  | public async Task>CalculateCount()  {  await Task.Delay(2000);  return 1;  }  public async task mytestmethod()  {  Task> count = CalculateCount();  int result = await count;  } |

In the above-given code,

async keyword is used for method declaration.

The Count is of a task type int which calls the method CalculateCount().

CalculateCount() starts execution and calculates something.

**Q-18) What is meant by ‘Transaction’ in a database and what are the ‘Properties of Transaction’?**

Transaction can be defined as a series of operations. If the transaction is successful, all the data modifications performed in the database will be committed and saved.

**Properties of Transaction**

* Atomicity
* Consistency
* Isolation
* Durability

**a) Atomicity:** Consider a bank transaction in which an amount, say Rs 1000/- is withdrawn from the Account ’AB’. If a system failure occurs, the amount will be rollbacked to the Account ’AB’ itself. This property is termed as ‘Atomicity’. This property states that either all the changes to the data are performed or none of them are performed. This means that the transactions should be completed successfully or transactions should not begin at all.

**b) Consistency:** For Example, consider the bank transaction in which an amount of Rs 500/- is transferred from Account ‘A’ to Account ‘B’. A system failure occurs and that Rs 500/- is reduced from Account ‘A’ and at the same time Rs 500/- is not credited to Account ‘B’. Such a system is considered an **‘Inconsistent’.**‘Consistency’ states that the data in the system should be in a consistent state after a transaction is completed successfully, thus maintain the integrity of the same.

As per the above Example, the transaction will be rolled back to add Rs 500/- to Account ‘A’ and thus maintain system consistency.

**c) Isolation:** Consider the bank transaction process in which an Account ‘A’ is transferring Rs 500/- to Account ‘B’ and Account ‘C’ at the same time. Account ‘A’ having Rs 1500/- as total balance. For a ‘Non-Isolated System’ both transactions will read the balance for Account ‘A’ as Rs 1000/ instead of Rs 500/- which is incorrect. ‘Isolation’ states that the changes in data of a system made by one transaction must be isolated from the changes made by the other transaction.

For an ‘Isolated System,’ the first transaction will read a balance of Rs 1000/- and the second transaction will read a balance of Rs 500/- for Account ‘A’.

**d) Durability:**Here it states that any change made in the data by a successful transaction will be permanent. Recovery management system is responsible to maintain the durability of a system.

**Q -19) What are two types of transactions supported by ADO.net?**

**Two types of Transaction supported by ADO.net**

* **Local Transaction:** A local transaction is based on a single data source. It will be handled directly by the database**. For Example,** We import ‘System.Data.SQL client’ namespace, if we need to perform data transaction using Sqlserver. Similarly, we import ‘System.Data.Oracle client’ namespace, if we are using Oracle database.
* **Distributed Transaction:** If the user needs to perform a transaction across multiple data Servers like SQL Server, Oracle, etc he can use a distributed transaction.

**Q-20) Define Property in C#?**

**Properties** are members that provide a flexible mechanism to read, write or compute the values of private fields, in other words by the property we can access private fields. In other words we can say that a property is a return type function/method with one parameter or without a parameter. These are always public data members. It uses methods to access and assign values to private fields called accessors.

**Q-21) Why can't you specify the accessibility modifier for methods inside the interface?**

In an interface, we have virtual methods that do not have method definition. All the methods are there to be overridden in the derived class. That's why they all are public.

**Q22) What is Expression Trees and how they used in LINQ?**

An **Expression Tree** is a data structure that contains Expressions, which is basically code. So it is a tree structure that represents a calculation you may make in code. These pieces of code can then be executed by "running" the expression tree over a set of data.

In LINQ, expression trees are used to represent structured queries that target sources of data that implement IQueryable<T>. For example, the LINQ provider implements the IQueryable<T> interface for querying relational data stores. The C# compiler compiles queries that target such data sources into code that builds an expression tree at runtime. The query provider can then traverse the expression tree data structure and translate it into a query language appropriate for the data source.

**Q-23) What is marshalling and why do we need it?**

Because different languages and environments have different calling conventions, different layout conventions, different sizes of primitives (cf. char in C# and char in C), different object creation/destruction conventions, and different design guidelines. You need a way to get the stuff out of managed land an into somewhere where unmanaged land can see and understand it and vice versa. That's what marshalling is for.

**Q-24) What is the "yield" keyword used for in C#?**

**Example:**

IEnumerable<object>FilteredList()

{

Foreach(object item in FullList)

{

If(IsItemInPartialList(item)

yield return item;

}

}

**Q-25) Name some advantages of LINQ over Stored Procedures?**

**Some advantages of LINQ over sprocs:**

1. **Type safety**: I think we all understand this.
2. **Abstraction**: This is especially true with LINQ-to-Entities. This abstraction also allows the framework to add additional improvements that you can easily take advantage of. PLINQ is an example of adding multi-threading support to LINQ. Code changes are minimal to add this support. It would be MUCH harder to do this data access code that simply calls sprocs.
3. **Debugging support**: I can use any .NET debugger to debug the queries. With sprocs, you cannot easily debug the SQL and that experience is largely tied to your database vendor (MS SQL Server provides a query analyzer, but often that isn't enough).
4. **Vendor agnostic**: LINQ works with lots of databases and the number of supported databases will only increase. Sprocs are not always portable between databases, either because of varying syntax or feature support (if the database supports sprocs at all).
5. **Deployment**: It's easier to deploy a single assembly than to deploy a set of sprocs. This also ties in with #4.
6. **Easier**: You don't have to learn T-SQL to do data access, nor do you have to learn the data access API (e.g. ADO.NET) necessary for calling the sprocs. This is related to #3 and #4.

**Q-26) What is multicast delegate in C#?**

Delegate can invoke only one method reference has been encapsulated into the delegate. It is possible for certain delegate to hold and invoke multiple methods. Such delegate called **multicast delegate**. Multicast delegate also know as combinable delegate, must satisfy the following conditions:

* The return type of the delegate must be void. None of the parameters of the delegate type can be delegate type can be declared as output parameters using out keywords.
* Multicast delegate instance is created by combining two delegates, the invocation list is formed by concatenating the invocation list of two operand of the addition operation. Delegates are invoked in the order they are added.

Actually all delegates in C# are MulticastDelegates, even if they only have a single method as target. (Even anonymous functions and lambdas are MulticastDelegates even though they by definition have only single target.)

Consider:

public partial class MainPage: PhoneApplicationPage

{

public delegate void MyDelegate(int a,int b);

//constructor

public Mainpage()

{

InitializeComponent();

//Multicast delegate

MyDelegate myDel= new MyDelegate(Add);

myDel+=new MyDelegate(Multipy);

myDel(10,20);

}

public void Add(int x,int y)

{

int sum=x+y;

MessageBox.Show(sum.ToString());

}

public void Multiply(intx,int y)

{

int sum=x\*y;

MessageBox.Show(sum.ToString());

}

**Q-27) List some different ways for equality check in .Net?**

* The ReferenceEquals() method - checks if two reference type variables(classes, not structs) are referred to the same memory address.
* The virtual Equals() method. (System.Object) - checks if two objects are equivalent.
* The static Equals() method - is used to handle problems when there is a null value in the check.
* The Equals method from IEquatable interface.
* The comparison operator == - usually means the same as ReferenceEquals, it checks if two variables point to the same memory address. The gotcha is that this operator can be overrided to perform other types of checks. In strings, for instance, it checks if two different instances are equivalent.

**Q-28) Could you explain the difference between Func vs. Action vs. Predicate?**

* **Predicate**: essentially Func<T, bool>; asks the question "does the specified argument satisfy the condition represented by the delegate?" Used in things like List.FindAll.
* **Action**: Perform an action given the arguments. Very general purpose. Not used much in LINQ as it implies side-effects, basically.
* **Func**: Used extensively in LINQ, usually to transform the argument, e.g. by projecting a complex structure to one property.

**Q-29) Implement the "Where" method in C?**

**Example:**

public static IEnumerable<T>Where<T>(this IEnumerable<T>items,Predicate<T>predicate)

{

Foreach(var item in items)

{

If(predicate(item))

{

// for lazy/deffer execution plus avoid temp collection defined

yield return item;

}

}

}

**Q-30) Explain what is weak reference in C#?**

The garbage collector cannot collect an object in use by an application while the application's code can reach that object. The application is said to have a strong reference to the object.

A **weak reference** permits the garbage collector to collect the object while still allowing the application to access the object. A weak reference is valid only during the indeterminate amount of time until the object is collected when no strong references exist.

Weak references are useful for objects that use a lot of memory, but can be recreated easily if they are reclaimed by garbage collection.

**Q-31) Name some disadvantages of LINQ over sprocs?**

Some disadvantages of LINQ vs sprocs:

1. **Network traffic**: sprocs need only serialize sproc-name and argument data over the wire while LINQ sends the entire query. This can get really bad if the queries are very complex. However, LINQ's abstraction allows Microsoft to improve this over time.
2. **Less flexible**: Sprocs can take full advantage of a database's featureset. LINQ tends to be more generic in it's support. This is common in any kind of language abstraction (e.g. C# vs assembler).
3. **Recompiling**: If you need to make changes to the way you do data access, you need to recompile, version, and redeploy your assembly. Sprocs can sometimes allow a DBA to tune the data access routine without a need to redeploy anything.

Security and manageability are something that people argue about too.

1. **Security**: For example, you can protect your sensitive data by restricting access to the tables directly, and put ACLs on the sprocs. With LINQ, however, you can still restrict direct access to tables and instead put ACLs on updatable table views to achieve a similar end (assuming your database supports updatable views).
2. **Manageability**: Using views also gives you the advantage of shielding your application non-breaking from schema changes (like table normalization). You can update the view without requiring your data access code to change.

**Q-32) What Is Attribute In C#?**

An attributes is a declarative tag that is used to convey information about the behaviors of various elements (classes, methods, assemblies, structures, enumerators, etc). it is access at compile time or run-time. Attributes are declare with a square brackets [] which is places above the elements.

[Obsolete(“Don’t use Old method, please use New method”, true)]

**For example**, consider the bellow class. If we call the old method it will through error message.

public class myClass

{

    [Obsolete("Don't use Old method, please use New method", true)]

    public string Old() { return "Old"; }

    public string New() { return "New"; }

}

myClass omyClass = new myClass();

omyClass.Old();

**Attributes Are Used:**

In a program the attributes are used for adding metadata, like compiler instruction or other information (comments, description, etc).

**Q-33) What is tracing in .NET?**

Tracing helps to see the information of issues at the runtime of the application. By default Tracing is disabled.  
  
Tracing has the following important features:

1. We can see the execution path of the page and application using the debug statement.
2. We can access and manipulate trace messages programmatically.
3. We can see the most recent tracing of the data.

Tracing can be done with the following 2 types.

1. **Page Level:**  
   When the trace output is displayed on the page and for the page-level tracing we need to set the property of tracing at the page level.  
     
   <%@ Page Trace="true" Language="C#"
2. **Application Level:**  
   In Application-Level tracing the information is stored for each request of the application. The default number of requests to store is 10. But if you want to increase the number of requests and discard the older request and display a recent request then you need to set the property in the web.config file.  
     
   <trace enabled="true"/>

**Q-34) What is the use of CheckBox in .NET?**

The CheckBox control is a very common control of HTML, unlike radio buttons it can select multiple items on a webpage. The CheckBox control in ASP.NET has many properties and some of them are listed below.

|  |  |
| --- | --- |
| Property | Description |
| AutoPostBack | Specifies whether the form should be posted immediately after the Checked property has changed or not. The default is false. |
| CausesValidation | Specifies if a page is validated when a Button control is clicked. |
| Checked | Specifies whether the check box is checked or not. |
| InputAttributes | Attribute names and values used for the Input element for the CheckBox control. |
| LabelAttributes | Attribute names and values used for the Label element for the CheckBox control. |
| Runat | Specifies that the control is a server control. Must be set to "server". |
| Text | The text next to the check box. |
| TextAlign | On which side of the check box the text should appear (right or left). |
| ValidationGroup | Group of controls for which the Checkbox control causes validation when it posts back to the server. |
| OnCheckedChanged | The name of the function to be executed when the Checked property has changed. |

**Q-35) Explain the new features added in version 4 of MVC (MVC4)?**

Following are features added newly ,

* Asynchronous controller task support.
* Bundling the java scripts.
* Segregating the configs for MVC routing, Web API, Bundle etc.
* Mobile templates
* Added ASP.NET Web API template for creating REST based services.
* Asynchronous controller task support.
* Bundling the java scripts.
* Segregating the configs for MVC routing, Web API, Bundle etc.

**Q-36) What is Separation of Concerns in ASP.NET MVC?**

It’s is the process of breaking the program into various distinct features which overlaps in functionality as little as possible. MVC pattern concerns on separating the content from presentation and data-processing from content.

**Q-37) What is Razor View Engine?**

Razor is the first major update to render HTML in MVC 3. Razor was designed specifically for view engine syntax. Main focus of this would be to simplify and code-focused templating for HTML generation. Below is the sample of using Razor:

@model MvcMusicStore.Models.Customer

@{ViewBag.Title = “Get Customers”;}

<div class=”cust”> <h3><em>@Model.CustomerName</em> </h3>

**Q-38) What is the meaning of Unobtrusive JavaScript?**

This is a general term that conveys a general philosophy, similar to the term REST (Representational State Transfer). Unobtrusive JavaScript doesn’t intermix JavaScript code in your page markup.

Eg : Instead of using events like onclick and onsubmit, the unobtrusive JavaScript attaches to elements by their ID or class based on the HTML5 data- attributes.

**Q-39) What is Attribute Routing in MVC?**

ASP.NET Web API supports this type routing. This is introduced in MVC5. In this type of routing, attributes are being used to define the routes. This type of routing gives more control over classic URI Routing. Attribute Routing can be defined at controller level or at Action level like ,

[Route(“{action = TestCategoryList}”)] //Controller Level

[Route(“customers/{TestCategoryId:int:min(10)}”)] // Action Level

**Q-40) How to enable Attribute Routing?**

Just add the method — “MapMvcAttributeRoutes()” to enable attribute routing as shown below

public static void RegistearRoutes(RouteCollection routes)

{

routes.IgnoareRoute(“{resource}.axd/{\*pathInfo}”);

//enabling attribute routing

routes.MapMvcAttributeRoutes();

//convention-based routing

routes.MapRoute

(

name: “Default”,

url: “{controller}/{action}/{id}”,

defaults: new { controller = “Customer”, action = “GetCustomerList”, id = UrlParameter.Optional }

);

}

**Q-41) Which are the important namespaces used in MVC?**

Below are the important namespaces used in MVC ,

System.Web.Mvc

System.Web.Mvc.Ajax

System.Web.Mvc.Html

System.Web.Mvc.Async

**Q-42) How route table has been created in ASP.NET MVC?**

Method — “RegisterRoutes()” is used for registering the routes which will be added in “Application\_Start()” method of global.asax file, which is fired when the application is loaded or started.

**Q-43) What is the difference between ViewBag and ViewData in MVC?**

ViewBag is a wrapper around ViewData, which allows to create dynamic properties. Advantage of viewbag over viewdata will be ,

In ViewBag no need to typecast the objects as in ViewData.

ViewBag will take advantage of dynamic keyword which is introduced in version 4.0. But before using ViewBag we have to keep in mind that ViewBag is slower than ViewData.

**Q-44) Explain TempData in MVC?**

TempData is again a key, value pair as ViewData. This is derived from “TempDataDictionary” class. TempData is used when the data is to be used in two consecutive requests, this could be between the actions or between the controllers. This requires typecasting in view.

**Q-45) What is Layout in MVC?**

Layout pages are similar to master pages in traditional web forms. This is used to set the common look across multiple pages. In each child page we can find — /p>

@{

Layout = “~/Views/Shared/TestLayout1.cshtml”;

}

This indicates child page uses TestLayout page as it’s master page.

**Q-46) What is the App Domain Concept in ASP.NET?**

ASP.NET introduces the concept of an Application Domain which is known as AppDomain for short. It can be considered as a lightweight process which is both a container and boundary. The .NET runtime uses an AppDomain as a container for code and data, just like the operating system uses a process as a container for code and data. As the operating system uses a process to isolate misbehaving code, the .NET runtime uses an AppDomain to isolate code inside a secure boundary.  
  
The CLR can allow the multiple .NET applications to run in a single AppDomain. Mulitple Appdomains can exist in Win32 process.

How to create AppDomain

AppDomains are created using the CreateDomain method. AppDomain instances are used to load and execute assemblies (Assembly). When an AppDomain is no longer in use, it can be unloaded.

public class MyAppDomain: MarshalByRefObject

public string GetInfo()

{

return AppDomain.CurrentDomain.FriendlyName;

}

}

public class MyApp

{

public static void Main()

{

AppDomain apd = AppDomain.CreateDomain("Rajendrs Domain");

MyAppDomain apdinfo = (MyAppDomain) apd.CreateInstanceAndUnwrap(Assembly.GetCallingAssembly()

.GetName()

.Name, "MyAppDomain");

Console.WriteLine("Application Name = " + apdinfo.GetInfo());

}

}

**Q-47) What is Query String in ASP?**

 A QueryString is a collection of characters input to a computer or web browser. A Query String is helpful when we want to transfer a value from one page to another. When we need to pass content between the HTML pages or aspx Web Forms in the context of ASP.NET, a Query String is Easy to use and the Query String follows a separating character, usually a Question Mark (?). It is basically used for identifying data appearing after this separating symbol. A Query String Collection is used to retrieve the variable values in the HTTP query string. If we want to transfer a large amount of data then we can't use the Request.QueryString. Query Strings are also generated by form submission or can be used by a user typing a query into the address bar of the browsers.  
  
**Syntax:**  
  
Request.QueryString(variable)[(index).count]

**Q-48) What is master page in ASP.NET?**

The extension of MasterPage is '.master'. MasterPage cannot be directly accessed from the client because it just acts as a template for the other Content Pages. In a MasterPage we can have content either inside ContentPlaceHolder or outside it. Only content inside the ContentPlaceHolder can be customized in the Content Page. We can have multiple masters in one web application. A MasterPage can have another MasterPage as Master to it. The MasterPageFile property of a webform can be set dynamically and it should be done either in or before the Page\_PreInit event of the WebForm. Page.MasterPageFile = "MasterPage.master". The dynamically set Master Page must have the ContentPlaceHolder whose content has been customized in the WebForm.

A master page is defined using the following code,  
  
<%@ master language="C#" %>  
  
Adding a MasterPage to the Project

1. Add a new MasterPage file (MainMaster.master) to the Web Application.
2. Change the Id of ContentPlaceHolder in <Head> to "cphHead" and the Id "ContentPlaceHolder1" to "cphFirst".
3. Add one more ContentPlaceHolder (cphSecond) to Master page.
4. To the master page add some header, footer and some default content for both the content place holders.

<form id="form1" runat="server"> Header...

<br />

<asp:ContentPlaceHolder id="cphFirst" runat="server"> This is First Content Place Holder (Default) </asp: ContentPlaceHolder>

<br />

<asp:ContentPlaceHolder ID="cphSecond" runat="server">

This is the second Content Place Holder (Default).

</asp:ContentPlaceHolder>

<br /> Footer...

</form>

**Q-49) Types of authentication and authorization in ASP.NET?**

There are three ways of doing authentication and authorization in ASP.NET:

**1.Windows authentication:** In this methodology ASP.NET web pages will use local windows users and groups to authenticate and authorize resources.

**2.Forms Authentication:** This is a cookie based authentication where username and password are stored on client machines as cookie files or they are sent through URL for every request. Form-based authentication presents the user with an HTML-based Web page that prompts the user for credentials.

**3.Passport authentication:** Passport authentication is based on the passport website provided by the Microsoft .So when user logins with credentials it will be reached to the passport website ( i.e. hotmail,devhood,windows live etc) where authentication will happen. If Authentication is successful it will return a token to your website.

**Anonymous access:**

If you do not want any kind of authentication then you will go for Anonymous access.

In 'web.config' file set the authentication mode to 'Windows' as shown in the below code snippets.

<authentication mode="Windows"/>

We also need to ensure that all users are denied except authorized users. The below code snippet inside the authorization tag that all users are denied. '?' indicates any unknown user.

<authorization>

<deny users="?"/>

</authorization>

**Q-50) What is the web API in ASP.NET?**

It is a framework provided by Microsoft for writing HTTP services. There are many frameworks available to build HTTP based services. They follow a common guideline of international standardization but with different flavors.  
  
For example, all framework must adhere to these status codes-

* 1xx - Informational Message
* 2xx - Successful
* 3xx - Redirection
* 4xx - Client Error
* 5xx - Server Error

Features

* It is light weight and thus good for small devices also like tablets, smart phones.
* No tedious & extensive configuration like WCF REST is required.
* MediaTypeFormatter makes easy to configure your APIs response type in single line (JSON, XML and so on).
* IIS Hosting dependency is no more and it can be hosted in application too.
* Easy and simple control with HTTP features such as Caching, Versioning, request/response headers and its various content formats.
* It support content-negotiation (deciding the best response data format that client can accept).

**Q-51)What is the code behind and Inline Code?**

**Code Behind:**  
  
Code Behind refers to the code for an ASP.NET Web page that is written in a separate class file that can have the extension of .aspx.cs or .aspx.vb depending on the language used. Here the code is compiled into a separate class from which the .aspx file derives. You can write the code in a separate .cs or .vb code file for each .aspx page. One major point of Code Behind is that the code for all the Web pages is compiled into a DLL file that allows the web pages to be hosted free from any Inline Server Code.  
  
**Inline Code:**  
  
Inline Code refers to the code that is written inside an ASP.NET Web Page that has an extension of .aspx. It allows the code to be written along with the HTML source code using a <Script> tag. It's major point is that since it's physically in the .aspx file it's deployed with the Web Form page whenever the Web Page is deployed.  
  
**Q -52) What are the differences between ASP.NET HttpHandler and HttpModule?**

The user requests for a resource on web server. The web server examines the file name extension of the requested file, and determines which ISAPI extension should handle the request. Then the request is passed to the appropriate ISAPI extension. For example when an .aspx page is requested it is passed to ASP.NET page handler. Then Application domain is created and after that different ASP.NET objects like Httpcontext, HttpRequest, HttpResponse are created. Then instance of HttpApplication is created and also instance of any configured modules. One can register different events of HttpApplication class like BeginRequest, AuthenticateRequest, AuthorizeRequest, ProcessRequest etc.  
  
**HTTP Handler:**  
  
HTTP Handler is the process which runs in response to a HTTP request. So whenever user requests a file it is processed by the handler based on the extension. So, custom http handlers are created when you need to special handling based on the file name extension. Let's consider an example to create RSS for a site. So, create a handler that generates RSS-formatted XML. Now bind the .rss extension to the custom handler.

**HTTP Modules:**  
  
HTTP Modules are plugged into the life cycle of a request. So when a request is processed it is passed through all the modules in the pipeline of the request. So generally http modules are used for,

* Security: For authenticating a request before the request is handled.
* Statistics and Logging: Since modules are called for every request they can be used for gathering statistics and for logging information.
* Custom header: Since response can be modified, one can add custom header information to the response.

**Q -53) What is the difference between ASP.NET Web API and WCF?**

The ASP. NET Web API is a framework that uses the HTTP services and makes it easy to provide the response to the client request. The response depends on the request of the clients. The Web API builds the HTTP services, and handles the request using the HTTP protocols. The request may be GET, POST, DELETE, PUT. We can also say that the ASP. NET Web API:

* Is an HTTP service.
* Is designed for reaching the broad range of clients.
* Uses the HTTP application.

We use the ASP. NET Web API for creating the REST ful (Representational State Transfer) services.  
  
The following are some important points of the ASP. NET Web API,

* The ASP. NET Web API supports the MVC application features that are controller, media formatters, routing etcetera.
* It is a platform for creating the REST services.
* It is a framework for creating the HTTP services.
* Responses can be formatted by the APIs MediaTypeFormatter into the Java Script Object Notation (JSON) and Extencible Markup Language (XML) formats.

**Q -54) What is the PostBack property in ASP.NET?**

If we create a web Page, which consists of one or more Web Controls that are configured to use AutoPostBack (every Web controls will have their own AutoPostBack property), the ASP.NET adds a special JavaScipt function to the rendered HTML Page. This function is named \_doPostBack() . When Called, it triggers a PostBack, sending data back to the web Server.  
  
ASP.NET also adds two additional hidden input fields that are used to pass information back to the server. This information consists of ID of the Control that raised the event and any additional information if needed. These fields will empty initially as shown below,

<input type="hidden" name="\_\_EVENTTARGET" id="\_\_EVENTTARGET" value="" />

<input type="hidden" name="\_\_EVENTARGUMENT" id="\_\_EVENTARGUMENT" value="" />

The following actions will be taken place when a user changes a control that has the AutoPostBack property set to true:

1. On the client side, the JavaScript \_doPostBack function is invoked, and the page is resubmitted to the server.
2. ASP.NET re-creates the Page object using the .aspx file.
3. ASP.NET retrieves state information from the hidden view state field and updates the controls accordingly.
4. The Page.Load event is fired.
5. The appropriate change event is fired for the control. (If more than one control has been changed, the order of change events is undetermined.)
6. The Page.PreRender event fires, and the page is rendered (transformed from a set of objects to an HTML page).
7. Finally, the Page.Unload event is fired.
8. The new page is sent to the client.

**Q-55) What are the Navigations techniques in ASP.NET?**

Navigation can cause data loss if it not properly handled. We do have many techniques to transfer data from one page to another but every technique has its own importance and benefits.  
  
We will discuss the following techniques in this article.

* Response.Redirect
* Server.Transfer
* Server.Exceute
* Cross page posting

**Q -56) What is WebParts in ASP.NET?**

ASP.NET 2.0 incorporates the concept of WEB PARTS in itself and we can code and explore that as easily as we have done with the other controls in the previous sessions.  
  
We can compose web parts pages from "web parts", which can be web controls, user controls.

**Component of Web Parts:**  
  
The web parts consist of different components like,

* Web Part Manager
* Web Part Zone
* CatLog Part
* CatLog Zone
* Connections Zone
* Editor Part
* Editor Zone

**Web Part Zone:**

* Web Part Zone can contain one or more Web Part controls.
* This provides the layout for the Controls it contains. A single ASPX page can contain one or more Web Part Zones.
* A Web Part Control can be any of the controls in the toolbox or even the customized user controls.

[Q-57](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)**[) Describe two uses of the “using” statement during the operation of C#](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled14)**

[The “using” statement in C# is used:](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

* [To import a namespace. The following coding is coded:](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[using System;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[using System. Collections.Generic;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[using System. ComponentModel;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[using System. Data;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[using System. Drawing;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[using System. Linq;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[using System. Text;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[using System. Threading, Tasks;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[using System. Windows. Forms;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[using System. Data. OLEDB;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

* [To close a running connection efficiently. The following coding is used:](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)
* [using System;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)
* [using System. Collections.Generic;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)
* [using System. ComponentModel;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)
* [using System. Data;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)
* [using System. Drawing;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)
* [using System. Linq;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)
* [using System. Text;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)
* [using System. Threading, Tasks;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)
* [using System. Windows. Forms;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)
* [using System. Data. OLEDB;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[namespace WindowsFormsApp1](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[{](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[public partial class Form1: Form](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[{](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[public Form1()](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[{](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[InitializeComponent();](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[}](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[private void Form1\_Load(object sender, EventArgs e)](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[{](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[String constr = “Provider=Microsoft.ACE.OLEDB.12.0;} +](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[“Data Source=\*\*\*” +](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[“Jet OLEDB: Database Password=\*\*\*”;](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[using (OLEDBConnection con = new OLEDBConnection (constr))](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[{](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[con.Open();](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[MessageBox.Show(“Connection is Opened!);](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[}](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[}](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[}](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

[}](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled9)

**[Q-58) Differentiate between Response.Expires and Response.ExpiresAbsolute?](https://www.onlineinterviewquestions.com/ado-net-interview-questions/" \l "collapseUnfiled12)**

Response.Expires is a property that is specific to the minutes that a particular page has in caches right from the time of the request that has been placed from the server. While on the other hand, Response.ExpiresAbsolute is a property that helps in providing the exact time at which a particular page cache has expired.

**For instance:**

* Response.Expires is set to 10 minutes and the page is directed to stay in cache for 10 minutes from the time it has been requested.
* Response.ExpiresAbsolute gives information like 13 January 15:48:30. This time specifies when the page was in cache.

**Q-59) What is Connection Pooling in ADO.NET?**

Connection pooling is the ability of reusing your connection to the database. This means if you enable Connection pooling in the connection object, actually you enable the re-use of the connection to more than one user.

ADO.NET uses a technique called connection pooling, which minimizes the cost of repeatedly opening and closing connections. Connection pooling reuses existing active connections with the same connection string instead of creating new connections when a request is made to the database. It involves the use of a connection manager that is responsible for maintaining a list, or pool, of available connections for a given connection string. Several pools exist if different connection strings ask for connection pooling.

**Example of Pooling:**

connection.ConnectionString = sqlConnectString + "Connection Timeout=30;Connection Lifetime=0;Min Pool Size=0;Max Pool Size=100;Pooling=true;";

//Open connection

A Connection String in the Web.Config file with connection pooling option:

<connectionStrings>

   <clear />

   <add name="sqlConnectionString" connectionString="Data Source=mySQLServer;Initial Catalog=myDatabase;Integrated Security=True;Connection Timeout=15;Connection Lifetime=0;Min Pool Size=0;Max Pool Size=100;Pooling=true;" />

</connectionStrings>

SQL Server connection string pooling attributes:

* **Connection Lifetime:**Length of time in seconds after creation after which a connection is destroyed. The default is 0, indicating that connection will have the maximum timeout.
* **Connection Reset:**Specifies whether the connection is reset when removed from the pool. The default is true.
* **Enlist:**Specifies whether the connection is automatically enlisted in the current transaction context of the creation thread if that transaction context exists. The default is true.
* **Load Balance Timeout:** Length of time in seconds that a connection can remain idle in a connection pool before being removed.
* **Max Pool Size:** Maximum number of connections allowed in the pool. The default is 100.
* **Min Pool Size:**Minimum number of connections maintained in the pool. The default is 0.
* **Pooling:**When true, the connection is drawn from the appropriate pool, or if necessary, created and added to the appropriate pool. The default is true.

**Q-60) What is SqlCommand Object?**

 The SqlCommand carries the SQL statement that needs to be executed on the database. SqlCommand carries the command in the CommandText property and this property will be used when the SqlCommand calls any of its execute methods.

* 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.

The three important methods exposed by the SqlCommand object is shown below:

* ExecuteScalar
* ExecuteNonQuery
* ExecuteReader

**ExecuteScalar** is useful for returning a single value from the database. For example, using this method we can retrieve a sum of sales made by a specific product, total number of records in the employee table, unique id by supplying filtering conditions and so on. Since this method performs faster we do not need to go for the Reader method just to retrieve a single scalar value.

**ExecuteNonQuery** is useful for performing data manipulation on the database. Simply, the ExecuteNonQuery is for executing the DML statements. The return value of the ExecuteNonQuery is an integral value that represents the number of rows affected by the Operation.

**ExecuteReader** is used when we need to retrieve rows and columns of data using the SQL select statements. As the data retrieved is a table of data, ExecuteReader returns SqlDataReader. We should iterate through this object to get the required values.

**Q-61) What is the SqlCommandBuilder?**

**Answer**: CommandBuilder helps you to generate update, delete, and insert commands on a single database table for a data adapter. Similar to other objects, each data provider has a command builder class. The OleDbCommandBuilder, SqlCommonBuilder, and OdbcCommandBuilder classes represent the CommonBuilder object in the OleDb, Sql, and ODBC data providers.

**Creating a Command Builder Object:**

Creating a CommonedBuider object is pretty simply. You pass a DataAdapter as an argument of the CommandBuilder constructor. For example,

// Create a command builder object

SqlCommandBuilder builder = **new** SqlCommandBuilder(adapter);

**Q-62) What is ExecuteScalar method in ADO.NET?**

**ExecuteScalar Method**

The ExecuteScalar method of the SqlCommand object is useful for retrieving a single value from the database. In our example, we need to retrieve the total number of records in the Titles table of the Pubs database. Since the total number of records is a single scalar value, the Execute Scalar method is used. The following is the code and its explanation:

**private** **void** frmSqlCommand\_Load(**object** sender, EventArgs e)

{

 //Sample 03: Open Database Connection

String con\_string = Properties.Settings.Default.ConStrPubs;

pubs\_db\_connection = **new** SqlConnection(con\_string);

pubs\_db\_connection.Open();

//Sample 04: Form the Command Object

 SqlCommand cmd = **new** SqlCommand();

cmd.CommandText = "Select Count(\*) as Count from Titles";

cmd.Connection = pubs\_db\_connection;

//Sample 05: Execute the Command & retrive scalar value

lblTotal.Text = System.Convert.ToString(cmd.ExecuteScalar());

}

**Q-63) What are Parameters in ADO.NET?**

Parameters in conjunction with SelectCommand to help you Select data for the DataSet.

The OleDbType describes the type information for the parameter. It consists of everything from strings to Global Unique Identifiers (GUIDs). SQL data provider has SqlDbType, and ODBC data provider has an ODBC type. These type names and definitions differ depending upon the provider you're using. For example, the Money type is the same in ODBC and Sqldata providers, but is called Currency in OleDb data providers.

**The OLEDB Parameter Class properties**

|  |  |
| --- | --- |
| Property | Description |
| DbType | Represents the DbType of the parameter. |
| Direction | Represents the direction of a parameter. A parameter can be input-only, output-only, bi-directional, or a stored procedure. |
| IsNullable | Represents whether a parameter accepts null values. |
| OleDbType | Represents the OleDbType of the parameter. |
| ParameterName | Represents the name of the parameter. |
| Precision | Represents the maximum number of digits used to represent the Value property. |
| Scale | Represents the decimal places to which Value is resolved. |
| Size | Represents the maximum size in bytes a column can store. |
| SourceColumn | Represents the source column mapped to the DataSet. |
| SourceVersion | Represents the DataRowversion. |
| Value | Represents the Value of the parameter. |

**Creating a parameter**

this.oleDbDeleteCommand2.Parameters.Add(newSystem.Data.OleDb.OleDbParameter("ContactName", System.Data.OleDb.OleDbType.char, 30, System.Data.ParameterDirection.Input, false, (( system.Byte)(0)),((System.Byte)(0)), "Contact Name", System.Data.DataRowVersion.Original, null));

**Q-64) Explain the properties and methods of Command Object?**

The command object is one of the basic components of ADO .NET.

1. The Command Object uses the connection object to execute SQL queries.
2. The queries can be in the form of Inline text, Stored Procedures or direct Table access.
3. An important feature of Command object is that it can be used to execute queries and Stored Procedures with Parameters.
4. If a select query is issued, the result set it returns is usually stored in either a DataSet or a DataReader object.

**Associated Properties of SqlCommand class**

|  |  |  |
| --- | --- | --- |
| Property | Type of Access | Description |
| Connection | Read/Write | The SqlConnection object that is used by the command object to execute SQL queries or Stored Procedure. |
| CommandText | Read/Write | Represents the T-SQL Statement or the name of the Stored Procedure. |
| CommandType | Read/Write | This property indicates how the CommandText property should be interpreted. The possible values are:  1. Text (T-SQL Statement) 2. StoredProcedure (Stored Procedure Name) 3. TableDirect |
| CommandTimeout | Read/Write | This property indicates the time to wait when executing a particular command.  **Default Time for Execution of Command is 30 Seconds.**  The Command is aborted after it times out and an exception is thrown. |

Now, let us have a look at various execute methods that can be called from a Command Object.

|  |  |
| --- | --- |
| Property | Description |
| ExecuteNonQuery | This method executes the command specifies and returns the number of rows affected. |
| ExecuteReader | The ExecuteReader method executes the command specified and returns an instance of instance of SqlDataReader class. |
| ExecuteScalar | This method executes the command specified and returns the first column of first row of the result set. The remaining rows and column are ignored |
| ExecuteXMLReader | This method executes the command specified and returns an instance of XmlReader class. This method can be used to return the result set in the form of an XML document |

**Q-65) What is clone() method of DataSet?**

The clone method copy the structure of DataSet. Means it copy only schema not full records of DataSet.

**For Example**,

Take another DataGridView and one button in your project and write the following code on button click event.

private void btnclone\_Click(object sender, EventArgs e)

{

   DataSet daset = ds.Clone();

   dataGridView2.DataSource = daset.Tables[0];

}

**Q-66) What is the Copy() method of DataSet?**

Copy the whole records with structure of DataSet.

**For Example,**

Take a button and set it's text as copy and write the following code on click event.

private void btncopy\_Click(object sender, EventArgs e)

{

   DataSet daset = ds.Copy();

   dataGridView2.DataSource = daset.Tables[0];

}

**Q-67) What is the HasChanges() method of DataSet?**

This method return boolean value to show whether record of DataSet has changed or not. It returns true if any changes made and false if no changes performed.

**For Example:**

Take a button and set it's text as "HasChanges" and write the following code on button click.

private void btnHasChanges\_Click(object sender, EventArgs e)

{

if(ds.HasChanges())

{

MessageBox.Show("Changes Has Made");

}

if(!ds.HasChanges())

{

MessageBox.Show("No Change");

}

}

**Q-68) What are the Connection object properties and Connection class members?**

The Connection class has a connection string that opens a connection to the database. The connection string will vary depending upon the provider used. The connection strings typically contain a group of property-value pair to describe how to connect to a database. For an OleDbConnection, you have properties such as Provider and DataSource.

|  |  |
| --- | --- |
| Property | Description |
| ConnectionString | Represent the connection string. |
| ConnectionTimeOut | Waiting time while establishing a connection. |
| DataBase | Name of the current database. |
| DataSource | Location of the file name of the data source. |
| Provider | Name of the OLE DB provider. This property is not available for Sql and ODBC data providers. |
| State | Current state of the connection of type ConnectionState. (Table 5-17 describes the ConnectionState). |
| PacketSize | Size of network packets. Available to only Sql data providers. |
| ServerVersion | SQL server version. Available to only Sql data providers. |
| WorkStationId | Database client ID. Available to only Sql data providers. |

The connection can have different states such as open, closed, connecting, and so on. The ConnectionType enumeration defines the members of the ConnectionState.

The connection Class Members

|  |  |
| --- | --- |
| **Method** | **Description** |
| BeginTransaction | Begins database transaction. |
| ChangeDatabase | Changes databases for an open connection. |
| Close | Closes an opened connection. |
| CreateCommand | Creates and return a Command object depends on the data providers. For example, OleDb Connection returns OleDbCommand, and SqlConnection returns SqlCommand. |
| Open | Open a new connection. |
| ReleaseObjectPool | Represents that the connection pooling can be cleared when the provider is released. Available only for Ole Db data providers. |

**Q-69) How DataSet objects in ADO.NET replace the ADO Recordset object?**

DataSet is good for ADO.NET objects to replace the ADO Recordset object:

* DataSet can hold multiple tables at a time.
* It allows data access to easily read or write data operations in / from the database.
* DataSet data stores in local system.
* It holds multiple rows at a time.
* It uses more memory.
* DataSet maintain relation.
* It bind data from the database

**DataSet representation in .aspx.cs code,**

protected void BindDataSet()

{

SqlConnection con = new SqlConnection("your database connection string ");

con.Open();

SqlCommand cmd = new SqlCommand("Write your query or procedure ", con);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds);

grid.DataSource = ds;

grid.DataBind();

}

**Q- 70) What is Transactions and Concurrency in ADO.NET?**

 Transactions: ADO.NET providers a transaction class that represents a transaction. All data providers provide their own version of the transaction class. The IDbTransaction interface implements the basic functionality of the transaction class. All data provider-specific classes implement this namespace.

**Methods of the Transaction Class**

|  |  |
| --- | --- |
| Method | Description |
| Commit | Commits the transaction to the database |
| Rollback | Rollbacks a transaction to the previous database state |
| Begin(IsolationLevel) | Begins a nested database transaction passing the isolation level |

**Concurrency in ADO.NET**

The ADO.NET model assumes that the optimistic concurrency is the default concurrency because of its disconnected nature of data. A user reads data in a data through a data adapter, and data is available to user as a local copy of the data. The server database is available to all other users.

Another way of handling optimistic concurrency that you may be familiar with is by checking to see if a timestamp on the data source row has changed or the row version number has changed on the row being updated.

Pessimistic locking on the database isn't really supported by the data providers because the connection to the database is not kept open, so you must perform all locking with business logic on the DataSet.