.Net Deployment Interview Questions And Answers Guide.

Global Guideline.

https://globalguideline.com/

.Net Deployment Job Interview Preparation Guide.

Question # 1

Explain Can the validation be done in the server side? Or this can be done only in the Client side?

Answer:-

Client side is done by default. Server side validation is also possible in .NET. We can switch off the client side and server side can be done only in .NET

Read More Answers.

Question # 2

Explain How to manage pagination in a page using .NET?

Answer:-

Using pagination option in DataGrid control is available in .NET. We have to set the number of records for a page, then it takes care of pagination by itself automatically.

Read More Answers.

Question # 3

ASP.NET interview questions?

Answer:-

1. What is a static class?
2. What is static member?
3. What is static function?
4. What is static constructor?
5. How can we inherit a static variable?
6. How can we inherit a static member?
7. Can we use a static function with a non-static variable?
8. How can we access static variable?
9. Why main function is static?
10. How will you load dynamic assembly? How will create assesblies at run time?
11. What is Reflection?
12. If I have more than one version of one assemblies, then how will I use old version (how/where to specify version number?) in my application?
13. How do you create threading in.NET? What is the namespace for that?
14. What do you mean by Serialize and MarshalByRef?
15. What is the difference between Array and LinkedList?
16. What is Asynchronous call and how it can be implemented using delegates?
17. How to create events for a control? What is custom events? How to create it?
18. If you want to write your own dot net language, what steps you will you take care?
19. Describe the diffeerence between inline and code behind - which is best in a loosely coupled solution?
20. How dot net compiled code will become platform independent?
21. Without modifying source code if we compile again, will it be generated MSIL again?
22. How does you handle this COM components developed in other programming languages in.NET?
23. How CCW (Com Callable Wrapper) and RCW (Runtime Callable Wrappers) works?
24. What are the new thee features of COM+ services, which are not there in COM (MTS)?
25. What are the differences between COM architecture and.NET architecture?
26. Can we copy a COM dll to GAC folder?
27. What is Shared and Repeatable Inheritance?
28. Can you explain what inheritance is and an example of when you might use it?
29. How can you write a class to restrict that only one object of this class can be created (Singleton class)?
30. What are virtual destructures?
31. What is close method? How its different from Finalize and Dispose?
32. What is Boxing and UnBoxing?
33. What is check/uncheck?
34. What is the use of base keyword? Tell me a practical example for base keyword's usage?
35. What are the different.NET tools which you used in projects?
36. What will do to avoid prior case?
37. What happens when you try to update data in a dataset in.NET while the record is already deleted in SQL Server as backend?
38. What is concurrency? How will you avoid concurrency when dealing with dataset?

39. One user deleted one row after that another user through his dataset was trying to update same row. What will happen? How will you avoid this problem?
40. How do you merge two datasets into the third dataset in a simple manner?
41. If you are executing these statements in commandObject. â€œSelect * from Table1; Select * from Table2? How you will deal result set? 42. How do you sort a dataset.
43. If a dataset contains 100 rows, how to fetch rows between 5 and 15 only?
44. What is the use of Parameter object?
45. How to generateXML from a dataset and vice versa?
46. How do you implement locking concept for dataset?
47. How will you do Redo and Undo in TextBox control?
48. How to implement DataGrid in.NET? How would you make a combo-box appear in one column of a DataGrid? What are the ways to show data grid inside a data grid for a master details type of tables? If we write any code for DataGrid methods. what is the access specifier used for that methods in the code behind file and why?
49. How can we create Tree control in asp.NET?
50. Write a program in C# to find the angle between the hours and minutes in a clock?
51. Write a program to create a user control with name and surname as data members and login as method and also the code to call it.
52. How can you read 3rd line from a text file?
53. Explain the code behind wors and contrast that using the inline style.
54. Explain different types of HTML, Web and server controls.
55. What are the differences between user control and server control?
56. How server form post-back works?
57. Can the action attribute of a server-side tag be set to a value and if not how can you possibly pass data from a form page to a subsequent page?
58. How would ASP and ASP.NET apps run at the same time on the same server?
59. What are good ADO.NET object to replace to ADO Recordset object.
60. Explain the differences between Server-side code and Client-side code.
61. What type of code(server or client) is found in a Code-Behind class?
62. Should validation (did the user enter a real date) occur server-side or client-side? Why?
63. What does the â€œEnableViewStateâ€• property do? Why would I want it on or off?
64. What is the difference between Server.Transfer and response.Redirect? Why?
65. Can you give an example of when it would be appropriate to use a web service as opposed to a non-serviced.NET component?
66. Let's say I have an existing application written using VB6 and this application utilizes Windows 2000 COM+ transaction services. How would you approach migrating this application to.NET?
67. If I am developing an application that must accomodate multiple security levels though secure login and my ASP.NET web application is spanned across three web-servers (using round-robin load balancing). What would be the best approach to maintain login-in state for the users?
68. What are ASP.NET web forms? How is this technology different than what is available though ASP(1.0-3.0)?
69. How does VB.NET achieve polymorphism?
70. How does C# achieve polymorphism?
71. Can you explain what is Inheritance and an example in VB.NET and C# of when you might use it?
72. Describe difference between inline and code-behind?
73. What is loosely coupled solution in.NET?
74. What is diffgram?
75. Where would you use an iHTTPModule and what are the limitations of any approach you might take in implementing one?
76. What are the Advantages and DisAdvantages of viewstate?
77. Describe session handling in a webform, how does it work and what are the limitations?
78. How would you get ASP.NET running in Apache web servers? Explain it's limitations.
79. What is MSIL and why should my developers need an appreciation of it if at all?
80. Which methos do you invoke on the DataAdapter control to load your generated dataset with data?
81. Can you edit data in Repeater control? How?
82. Which template must you provide, in order to display data in a Repeater control?
83. How can you provide an alternating color scheme in a Repeater control?
84. What property must you set, and what method must you call in your code, in order to bind the data from some data source to the repeater control?
85. What base class do all web forms inherit from?
86. What method do you use to explicitly kill a user's session? How?
87. How do you turn off cookies for one page in your site? Give an example.
88. Which two properties are on every validation control?
89. What tags do you need to add within the asp:datagrid tags to bind columns manually? Give an example.
90. How do you create a permanent cookie?
91. What tag do you use to add a hyperlink column to the dataGrid?
92. What is the standard you use to wrap up a call to a Web Service?
93. Which method do you use to redirect the user to another page without performing a round trip to the client? How?
94. What is the transport protocol you use to call a Seb Service SOAP?
95. What does WSDL stand for?
96. What property do you have to set to tell the grid which page to go to when using the Pager object?
97. Where on the Internet would you look for Web Services?
98. What tags do you need to add within the asp:datagrid tags to bind columns manually? How?
99. Which property on a Combo Box do you set with a column name, prior to setting the DataSource, to display data in the combo box?
100. How is a property designated as read-only?
101. Which control would you use if you needed to make sure the values in two different controls matched?
111. Differences between DLL and EXE?
112. Can an assembly have EXE?
113. Can a DLL be changed to an EXE?
114.. Compare & contrast rich client (smart clients or Windows-based) & browser-based Web application
115. Compare Client server application with n-Tier application
116.. Can a try block have more than one catch block?
117. Can a try block have nested try blocks?
118. How do you load an assembly at runtime?
119. If I am writing in a language like VB or C++, what are the procedures to be followed to support .NET?
120. How do you view the methods and members of a DLL?
121. What is shadowing?
122. What are the collections you've used?
123. What's the use of formatters in .NET?
124. How is Threading done in .NET?
125. Differences between Namespace, Class, Assembly?

Read More Answers.

Question # 4

What is your Observations between VB.NET and VC#.NET?

Answer:-

Choosing a programming language depends on your language experience and the scope of the application you are building. While small applications are often created using only one language, it is not uncommon to develop large applications using multiple languages.
For example, if you are extending an application with existing XML Web services, you might use a scripting language with little or no programming effort. For client-server applications, you would probably choose the single language you are most comfortable with for the entire application. For new enterprise applications, where large teams of developers create components and services for deployment across multiple remote sites, the best choice might be to use several languages depending on developer skills and long-term maintenance expectations.
The .NET Platform programming languages - including Visual Basic .NET, Visual C#, and Visual C++ with managed extensions, and many other programming languages from various vendors - use .NET Framework services and features through a common set of unified classes. The .NET unified classes provide a consistent method of accessing the platform's functionality. If you learn to use the class library, you will find that all tasks follow the same uniform architecture. You no longer need to learn and master different API architectures to write your applications.
The .NET Platform programming languages - including Visual Basic .NET, Visual C#, and Visual C++ with managed extensions, and many other programming languages from various vendors - use .NET Framework services and features through a common set of unified classes. The .NET unified classes provide a consistent method of accessing the platform's functionality. If you learn to use the class library, you will find that all tasks follow the same uniform architecture. You no longer need to learn and master different API architectures to write your applications.
In most situations, you can effectively use all of the Microsoft programming languages. Nevertheless, each programming language has its relative strengths and you will want to understand the features unique to each language. The following sections will help you choose the right programming language for your application.
Visual Basic .NET
Visual Basic .NET is the next generation of the Visual Basic language from Microsoft. With Visual Basic you can build .NET applications, including Web services and ASP.NET Web applications, quickly and easily. Applications made with Visual Basic are built on the services of the common language runtime and take advantage of the .NET Framework.
Visual Basic has many new and improved features such as inheritance, interfaces, and overloading that make it a powerful object-oriented programming language. Other new language features include free threading and structured exception handling. Visual Basic fully integrates the .NET Framework and the common language runtime, which together provide language interoperability, garbage collection, enhanced security, and improved versioning support. A Visual Basic support single inheritance and creates Microsoft intermediate language (MSIL) as input to native code compilers.
Visual Basic is comparatively easy to learn and use, and Visual Basic has become the programming language of choice for hundreds of thousands of developers over the past decade. An understanding of Visual Basic can be leveraged in a variety of ways, such as writing macros in Visual Studio and providing programmability in applications such as Microsoft Excel, Access, and Word.
Visual Basic provides prototypes of some common project types, including:
â€¢ Windows Application.
â€¢ Class Library.
â€¢ Windows Control Library.
â€¢ ASP.NET Web Application.
â€¢ ASP.NET Web Service.
â€¢ Web Control Library.
â€¢ Console Application.
â€¢ Windows Service.
â€¢ Windows Service.
Visual C# .NET
Visual C# (pronounced C sharp) is designed to be a fast and easy way to create .NET applications, including Web services and ASP.NET Web applications. Applications written in Visual C# are built on the services of the common language runtime and take full advantage of the .NET Framework.
C# is a simple, elegant, type-safe, object-oriented language recently developed by Microsoft for building a wide range of applications. Anyone familiar with C and similar languages will find few problems in adapting to C#. C# is designed to bring rapid development to the C++ programmer without sacrificing the power and control that are a hallmark of C and C++. Because of this heritage, C# has a high degree of fidelity with C and C++, and developers familiar with these languages can quickly become productive in C#. C# provides intrinsic code trust mechanisms for a high level of security, garbage collection, and type safety. C# supports single inheritance and creates Microsoft intermediate language (MSIL) as input to native code compilers.
C# is fully integrated with the .NET Framework and the common language runtime, which together provide language interoperability, garbage collection, enhanced security, and improved versioning support. C# simplifies and modernizes some of the more complex aspects of C and C++, notably namespaces, classes, enumerations, overloading, and structured exception handling. C# also eliminates C and C++ features such as macros, multiple inheritance, and virtual base classes. For current C++ developers, C# provides a powerful, high-productivity language alternative.
Visual C# provides prototypes of some common project types, including:
â€¢ Windows Application.
â€¢ Class Library.
â€¢ Windows Control Library.
â€¢ ASP.NET Web Application.
â€¢ ASP.NET Web Service.
â€¢ Web Control Library.
â€¢ Console Application.
â€¢ Windows Service.

Read More Answers.

Question # 5

How to debug failed assembly binds?

Answer:-

Use the Assembly Binding Log Viewer (fuslogvw.exe) to find out the paths searched.
Using Binding LogViewer.

Read More Answers.

Question # 6

Explain What is an interface and what is an abstract class? Please, expand by examples of using both. Explain why?

Answer:-

In a interface class, all methods are abstract without implementation where as in an abstract class some methods we can define concrete. In interface, no accessibility modifiers are allowed. An abstract class may have accessibility modifiers. Interface and abstract class are basically a set of rules which u have to follow in case u r using them(inheriting them).
Abstract classes are closely related to interfaces. They are classes that cannot be instantiated, and are frequently either partially implemented, or not at all

implemented. One key difference between abstract classes and interfaces is that a class may implement an unlimited number of interfaces, but may inherit from only one abstract (or any other kind of) class. A class that is derived from an abstract class may still implement interfaces. Abstract classes are useful when creating components because they allow you specify an invariant level of functionality in some methods, but leave the implementation of other methods until a specific implementation of that class is needed. They also version well, because if additional functionality is needed in derived classes, it can be added to the base class without breaking code.
Abstract Classes
An abstract class is the one that is not used to create objects. An abstract class is designed to act as a base class (to be inherited by other classes). Abstract class is a design concept in program development and provides a base upon which other classes are built. Abstract classes are similar to interfaces. After declaring an abstract class, it cannot be instantiated on it's own, it must be inherited. Like interfaces, abstract classes can specify members that must be implemented in inheriting classes. Unlike interfaces, a class can inherit only one abstract class. Abstract classes can only specify members that should be implemented by all inheriting classes.
An interface looks like a class, but has no implementation. They're great for putting together plug-n-play like architectures where components can be interchanged at will. Think Firefox Plug-in extension implementation. If you need to change your design, make it an interface. However, you may have abstract classes that provide some default behavior. Abstract classes are excellent candidates inside of application frameworks.
One additional key difference between interfaces and abstract classes (possibly the most important one) is that multiple interfaces can be implemented by a class, but only one abstract class can be inherited by any single class.
Some background on this: C++ supports multiple inheritance, but C# does not. Multiple inheritance in C++ has always be controversial, because the resolution of multiple inherited implementations of the same method from different base classes is hard to control and anticipate. C# decided to avoid this problem by allowing a class to implement multiple interfaces, which do not contain method implementations, but restricting a class to have at most a single parent class. Although this can result in redundant implementations of the same method when different classes implement the same interface, it is still an excellent compromise.
Another difference between interfaces and abstract classes is that an interface can be implemented by an abstract class, but no class, abstract or otherwise, can be inherited by an interface.
What is an Abstract class?
An abstract class is a special kind of class that cannot be instantiated. So the question is why we need a class that cannot be instantiated? An abstract class is only to be sub-classed (inherited from). In other words, it only allows other classes to inherit from it but cannot be instantiated. The advantage is that it enforces certain hierarchies for all the subclasses. In simple words, it is a kind of contract that forces all the subclasses to carry on the same hierarchies or standards.
What is an Interface?
An interface is not a class. It is an entity that is defined by the word Interface. An interface has no implementation; it only has the signature or in other words, just the definition of the methods without the body. As one of the similarities to Abstract class, it is a contract that is used to define hierarchies for all subclasses or it defines specific set of methods and their arguments. The main difference between them is that a class can implement more than one interface but can only inherit from one abstract class. Since C# doesn't support multiple inheritance, interfaces are used to implement multiple inheritance.

Read More Answers.

Question # 7

What is assemblies in .NET?

Answer:-

Assemblies are similar to dll files. Both has the reusable pieces of code in the form of classes/ functions. Dll needs to be registered but assemblies have its own metadata.
Assembly is a single deployable unit that contains information about the implementation of classes, structures and interfaces. it also stores the information about itself called metadata and includes name and verison of the assembly, security information, information about the dependencies and the list of files that constitute the assembly.
Assembly also contains namespaces. In the .Net Framework, applications are deployed in the form of assemblies.
An assembly is a single deploy able unit that contains all the information about the implementation of :
- classes
- structures and
- interfaces
An assembly stores all the information about itself. This information is called METADATA and include the name and the verison number of the assembly, security information, information about the dependencies and a lost of files that constitute the assembly.
All the application developed using the .NET framework are made up of assemblies.
Namespaces are also stored in assemblies
In the Microsoft .NET framework an assembly is a partially compiled code library for use in deployment, versioning and security. In the Microsoft Windows implementation of .NET, an assembly is a PE (portable executable) file. There are two types, process assemblies (EXE) and library assemblies (DLL). A process assembly represents a process which will use classes defined in library assemblies. In version 1.1 of the CLR classes can only be exported from library assemblies; in version 2.0 this restriction is relaxed. The compiler will have a switch to determine if the assembly is a process or library and will set a flag in the PE file. .NET does not use the extension to determine if the file is a process or library. This means that a library may have either .dll or .exe as its extension.
The code in an assembly is partially compiled into CIL, which is then fully compiled into machine language at runtime by the CLR.
An assembly can consist of one or more files. Code files are called modules. An assembly can contain more than one code module and since it is possible to use different languages to create code modules this means that it is technically possible to use several different languages to create an assembly. In practice this rarely happens, principally because Visual Studio only allows developers to create assemblies that consist of a single code module.

Read More Answers.

Question # 8

What is the base class of Button control in .NET?

Answer:-

Listing from visual studio .net > Button Class System.Object
System.MarshalByRefObject
System.ComponentModel.Component
System.Windows.Forms.Control
System.Windows.Forms.ButtonBase
System.Windows.Forms.Button

Read More Answers.

Question # 9

Explain Attributes in dot NET?

Answer:-

Attributes are declarative tags in code that insert additional metadata into an assembly. There exist two types of attributes in the .NET Framework: Predefined attributes such as Assembly Version, which already exist and are accessed through the Runtime Classes; and custom attributes, which you write yourself by extending

the System.Attribute class.

Read More Answers.

Question # 10

Explain the top .NET class that everything is derived from?

Answer:-

System.Object.

Read More Answers.

Question # 11

What is view state in .NET?

Answer:-

The web is stateless. But in ASP.NET, the state of a page is maintained in the in the page itself automatically. How? The values are encrypted and saved in hidden controls. this is done automatically by the ASP.NET. This can be switched off / on for a single control

Read More Answers.

Question # 12

What is .NET?

Answer:-

NET is essentially a framework for software development.It is similar in nature to any other software development framework (J2EE etc) in that it provides a set of runtime containers/capabilities, and a rich set of pre-built functionality in the form of class libraries and APIs
The .NET Framework is an environment for building, deploying, and running Web Services and other applications. It consists of three main parts: the Common Language Runtime, the Framework classes, and ASP.NET.

Read More Answers.

Question # 13

Explain the difference between VB and VB.NET?

Answer:-

Now VB.NET is object-oriented language. The following are some of the differences:
Data Type Changes
The .NET platform provides Common Type System to all the supported languages. This means that all the languages must support the same data types as enforced by common language runtime. This eliminates data type incompatibilities between various languages. For example on the 32-bit Windows platform, the integer data type takes 4 bytes in languages like C++ whereas in VB it takes 2 bytes. Following are the main changes related to data types in VB.NET:
. Under .NET the integer data type in VB.NET is also 4 bytes in size.
. VB.NET has no currency data type. Instead it provides decimal as a replacement.
. VB.NET introduces a new data type called Char. The char data type takes 2 bytes and can store Unicode characters.
. VB.NET do not have Variant data type. To achieve a result similar to variant type you can use Object data type. (Since every thing in .NET including primitive data types is an object, a variable of object type can point to any data type).
. In VB.NET there is no concept of fixed length strings.
. In VB6 we used the Type keyword to declare our user-defined structures. VB.NET introduces the structure keyword for the same purpose.
Declaring Variables
Consider this simple example in VB6:
Dim x,y as integer

Read More Answers.

Question # 14

What is smart navigation in .NET?

Answer:-

The cursor position is maintained when the page gets refreshed due to the server side validation and the page gets refreshed.

Read More Answers.

Question # 15

What is CLR in .NET?

Answer:-

CLR(Common Language Runtime) is the main resource of .Net Framework. it is collection of services like garbage collector, exception handler, jit compilers etc. with the CLR cross language integration is possible.
The .NET Framework provides a runtime environment which runs the code and provides services that make the development process easier. This runtime environment in .NET Framework is known as Common Language Runtime (CLR). The CLR sits at the very heart of managed code. Common Language Runtime is the generalized multi-language, reflective execution engine on which code originally written in various languages runs. At a higher level, CLR is simply an engine that takes in Intermediate Language (IL) instructions, translates them into machine instructions, and executes them. Although the common language runtime provides many standard runtime services, managed code is never interpreted. A feature called just-in-time (JIT) compiling enables all managed code to run in the native machine language of the system on which it is executing. The CLR shares much in common with a traditional operating system.
Quote:
Managed code is the term used for any code that is running on .NET Framework.
The CLR provides the infrastructure that enables managed code to execute as well provides variety of services during execution. When a method, for which IL has been generated, is called for the first time the CLR compiles the IL into native code that is specific to the processor the Environment it is running on (This process is known as Just in Time Compilation or JIT). If the same method is called next time, the existing JIT compiled code is reused. During execution managed code receives variety of services from the runtime environment.
Quote:

When compiling to managed code, the compiler translates your source code into Microsoft intermediate language (MSIL), which is a CPU-independent set of instructions that can be efficiently converted to native code. Intermediate Language is a binary assembly language that is compiled at runtime down to whatever machine language is appropriate for the host CPU. This runtime compilation is called Just-In-Time Compiling or JIT-compiling.

Read More Answers.

Question # 16

How Viewstate is being formed and how it is stored on client in .NET?

Answer:-

The type of ViewState is System.Web.UI.StateBag, which is a dictionary that stores name/value pairs. ViewState is persisted to a string variable by the ASP.NET page framework and sent to the client and back as a hidden variable. Upon postback, the page framework parses the input string from the hidden variable and populates the ViewState property of each control. If a control uses ViewState for property data instead of a private field, that property automatically will be persisted across round trips to the client. (If a property is not persisted in ViewState, it is good practice to return its default value on postback.)

Read More Answers.

Question # 17

When displaying fonts, what is the difference between pixels, points and ems?

Answer:-

A pixel is the lowest-resolution dot the computer monitor supports. Its size depends on user's settings and monitor size. A point is always 1/72 of an inch. An em is the number of pixels that it takes to display the letter M.

Read More Answers.

Question # 18

Explain What is a Manifest in .NET?

Answer:-

An assembly manifest contains all the metadata needed to specify the assembly's version requirements and security identity, and all metadata needed to define the scope of the assembly and resolve references to resources and classes. The assembly manifest can be stored in either a PE (Portable Executable) file (an .exe or .dll) with Microsoft intermediate language (MSIL) code or in a standalone PE (Portable Executable) file that contains only assembly manifest information. The following table shows the information contained in the assembly manifest. The first four items the assembly name, version number, culture, and strong name information make up the assembly's identity.
Assembly name: A text string specifying the assembly's name.
Version number: A major and minor version number, and a revision and build number. The common language runtime uses these numbers to enforce version policy.
Culture: Information on the culture or language the assembly supports. This information should be used only to designate an assembly as a satellite assembly containing culture- or language-specific information. (An assembly with culture information is automatically assumed to be a satellite assembly.) Strong name information: The public key from the publisher if the assembly has been given a strong name. List of all files in the assembly:
A hash of each file contained in the assembly and a file name. Note that all files that make up the assembly must be in the same directory as the file containing the assembly manifest.
Type reference information: Information used by the runtime to map a type reference to the file that contains its declaration and implementation. This is used for types that are exported from the assembly.
Information on referenced assemblies: A list of other assemblies that are statically referenced by the assembly. Each reference includes the dependent assembly's name, assembly metadata (version, culture, operating system, and so on), and public key, if the assembly is strong named.

Read More Answers.

Question # 19

What is typical about a Windows process in regards to memory allocation in dot NET?

Answer:-

Each process is allocated its own block of available RAM space, no process can access another process' code or data. If the process crashes, it dies alone without taking the entire OS or a bunch of other applications down.

Read More Answers.

Question # 20

Explain security measures exist for .NET Remoting in System.Runtime.Remoting?

Answer:-

None. Security should be taken care of at the application level. Cryptography and other security techniques can be applied at application or server level.

Read More Answers.

Question # 21

Which dll is required to translate XML to SQL in Internet Information Server (IIS)?

Answer:-

Microsoft.data.sqlxml.dll used to translate XML to SQL using Internet Information Server IIS

Read More Answers.

Question # 22

Explain How ASP .NET different from ASP?

Answer:-

Scripting is separated from the HTML, Code is compiled as a DLL, these DLLs can be executed on the server.

Read More Answers.

Question # 23

Explain about .NET assemblies?

Answer:-

Assemblies are the smallest units of versioning and deployment in the .NET application. Assemblies are also the building blocks for programs such as Web services, Windows services, serviced components, and .NET Remoting applications.
Assemblies are also the building blocks for programs such as Web services, Windows services, serviced components, and .NET Remoting applications.
Types of Assemblies:
Private, Public/Shared

Read More Answers.

Question # 24

Explain What are possible implementations of distributed applications in .NET?

Answer:-

NET Remoting and ASP.NET Web Services. If we talk about the Framework Class Library, noteworthy classes are in System.Runtime.Remoting and System.Web.Services.

Read More Answers.

Microsoft .Net Technologies Most Popular Interview Topics.

1 : Dot Net Frequently Asked Interview Questions and Answers Guide.

2 : C# (Sharp) Programming Language Frequently Asked Interview Questions and Answers Guide.

3 : VB .Net Frequently Asked Interview Questions and Answers Guide.

4 : ADO.Net Entity Framework Frequently Asked Interview Questions and Answers Guide.

5 : ASP.Net Frequently Asked Interview Questions and Answers Guide.

6 : ADO.NET Frequently Asked Interview Questions and Answers Guide.

7 : ASP Programming Frequently Asked Interview Questions and Answers Guide.

8 : Crystal Reports Frequently Asked Interview Questions and Answers Guide.

9 : .Net Architecture Frequently Asked Interview Questions and Answers Guide.

10 : ASP.NET 2.0 Frequently Asked Interview Questions and Answers Guide.

About Global Guideline.

Global Guideline is a platform to develop your own skills with thousands of job interview questions and web tutorials for fresher's and experienced candidates. These interview questions and web tutorials will help you strengthen your technical skills, prepare for the interviews and quickly revise the concepts. Global Guideline invite you to unlock your potentials with thousands of **Interview Questions with Answers** and much more. Learn the most common technologies at Global Guideline. We will help you to explore the resources of the World Wide Web and develop your own skills from the basics to the advanced. Here you will learn anything quite easily and you will really enjoy while learning. Global Guideline will help you to become a professional and Expert, well prepared for the future.

* This PDF was generated from https://GlobalGuideline.com at **November 29th, 2023**

* If any answer or question is incorrect or inappropriate or you have correct answer or you found any problem in this document then don't hesitate feel free and e-mail us we will fix it.

You can follow us on FaceBook for latest Jobs, Updates and other interviews material.
www.facebook.com/InterviewQuestionsAnswers

Follow us on Twitter for latest Jobs and interview preparation guides
https://twitter.com/InterviewGuide

Best Of Luck.

Global Guideline Team
https://GlobalGuideline.com
Info@globalguideline.com