**.Net Interview Questions updated on Aug 2018**

**1.What is .NET?**

NET is an integral part of many applications running on Windows and provides common functionality for those applications to run. This download is for people who need .NET to run an application on their computer. For developers, the .NET Framework provides a comprehensive and consistent programming model for building applications that have visually stunning user experiences and seamless and secure communication.

**2.How many languages .NET is supporting now?**

When .NET was introduced it came with several languages.  
VB.NET,  
C#,  
COBOL  
and  
Perl, etc.

**3. What is an IL?**

Intermediate Language is also known as MSIL (Microsoft Intermediate Language) or CIL (Common Intermediate Language). All .NET source code is compiled to IL. IL is then converted to machine code at the point where the software is installed, or at run-time by a Just-In-Time (JIT) compiler.

**4. What is code access security (CAS)?**

Code access security (CAS) is part of the .NET security model that prevents unauthorized access of resources and operations, and restricts the code to perform particular tasks.

**5. What is Difference between NameSpace and Assembly?**

Assembly is physical grouping of logical units, Namespace, logically groups classes.  
Namespace can span multiple assembly.

**6. Mention the execution process for managed code.**

A) Choosing a language compiler  
B) Compiling the code to MSIL  
C) Compiling MSIL to native code  
D) Executing the code.

**7. What is Microsoft Intermediate Language (MSIL)?**

The .NET Framework is shipped with compilers of all .NET programming languages to develop programs. There are separate compilers for the Visual Basic, C#, and Visual C++ programming languages in .NET Framework. Each .NET compiler produces an intermediate code after compiling the source code. The intermediate code is common for all languages and is understandable only to .NET environment. This intermediate code is known as MSIL.

**8. What is managed extensibility framework?**

Managed extensibility framework (MEF) is a new library that is introduced as a part of .NET 4.0 and Silverlight 4. It helps in extending your application by providing greater reuse of applications and components. MEF provides a way for host application to consume external extensions without any configuration requirement.

**9. Which method do you use to enforce garbage collection in .NET?**

The System.GC.Collect() method.

**10. What is the difference between int and int32**.

There is no difference between int and int32. System.Int32 is a .NET Class and int is an alias name for System.Int32.

**11. What are tuples?**

Tuple  is a fixed-size collection that can have elements of either same or different data types. Similar to arrays, a user must have to specify the size of a tuple at the time of declaration. Tuples are allowed to hold up from 1 to 8 elements and if there are more than 8 elements, then the 8th element can be defined as another tuple. Tuples can be specified as parameter or return type of a method.

**12. What is the full form of ADO?**

The full form of ADO is ActiveX Data Object.

**13. What are the two fundamental objects in ADO.NET?**

DataReader and DataSet are the two fundamental objects in ADO.NET.

**14. What is the meaning of object pooling?**

Object pooling is a concept of storing a pool (group) of objects in memory that can be reused later as needed. Whenever, a new object is required to create, an object from the pool can be allocated for this request; thereby, minimizing the object creation. A pool can also refer to a group of connections and threads. Pooling, therefore, helps in minimizing the use of system resources, improves system scalability, and performance.

**15. Mention the namespace that is used to include .NET Data Provider for SQL server in .NET code.**

The System.Data.SqlClient namespace.

**16. Which architecture does Datasets follow?**

Datasets follow the disconnected data architecture.

**17. What is the role of the DataSet object in ADO.NET?**

One of the major component of ADO.NET is the DataSet object, which always remains disconnected from the database and reduces the load on the database.

**18. Which property is used to check whether a DataReader is closed or opened?**

The IsClosed property is used to check whether a DataReader is closed or opened. This property returns a true value if a Data Reader is closed, otherwise a false value is returned.

**19. Name the method that needs to be invoked on the DataAdapter control to fill the generated DataSet with data?**

The Fill() method is used to fill the dataset with data.

**20. What are the pre-requisites for connection pooling?**

There must be multiple processes to share the same connection describing the same parameters and security settings. The connection string must be identical.

**21. Which adapter should you use, if you want to get the data from an Access database?**

OleDbDataAdapter is used to get the data from an Access database.

**22. What are different types of authentication techniques that are used in connection strings to connect .NET applications with Microsoft SQL Server?**

The Windows Authentication option

The SQL Server Authentication option

**23. What are the parameters that control most of connection pooling behaviors?**

Connect Timeout  
Max Pool Size  
Min Pool Size  
Pooling

**24. What is AutoPostBack?**

If you want a control to postback automatically when an event is raised, you need to set the AutoPostBack property of the control to True.

**25. What is the function of the ViewState property?**

The ASP.NET 4.0 introduced a new property called ViewStateMode for the Control class. Now you can enable the view state to an individual control even if the view state for an ASP.NET page is disabled.

**26. Which properties are used to bind a DataGridView control?**

The DataSource property and the DataMember property are used to bind a DataGridView control.

**27. What is the basic difference between ASP and ASP.NET?**

The basic difference between ASP and ASP.NET is that ASP is interpreted; whereas, ASP.NET is compiled. This implies that since ASP uses VBScript; therefore, when an ASP page is executed, it is interpreted. On the other hand, ASP.NET uses .NET languages, such as C# and VB.NET, which are compiled to Microsoft Intermediate Language (MSIL).

**28. In which event are the controls fully loaded?**

Page load event guarantees that all controls are fully loaded. Controls are also accessed in Page\_Init events but you will see that view state is not fully loaded during this event

**29. How can we identify that the Page is Post Back?**

Page object has an "IsPostBack" property, which can be checked to know that is the page posted back.

**30. Which is the parent class of the Web server control?**

The System.Web.Ul.Control class is the parent class for all Web server controls.

**31. What are the advantages of the code-behind feature?**

i)Makes code easy to understand and debug by separating application logic from HTML tags  
ii)Provides the isolation of effort between graphic designers and software engineers  
iii)Removes the problems of browser incompatibility by providing code files to exist on the Web server and supporting Web pages to be compiled on demand.

**32. Define a multilingual Web site.**

A multilingual Web site serves content in a number of languages. It contains multiple copies for its content and other resources, such as date and time, in different languages.

**33. What is IIS? Why is it used?**

Internet Information Services (IIS) is created by Microsoft to provide Internet-based services to ASP.NET Web applications. It makes your computer to work as a Web server and provides the functionality to develop and deploy Web applications on the server. IIS handles the request and response cycle on the Web server. It also offers the services of SMTP and FrontPage server extensions. The SMTP is used to send emails and use FrontPage server extensions to get the dynamic features of IIS, such as form handler.

**34. How can you register a custom server control to a Web page?**

You can register a custom server control to a Web page using the @Register directive.

**35. Which ASP.NET objects encapsulate the state of the client and the browser?**

The Session object encapsulates the state of the client and browser.

**36. Differentiate globalization and localization.**

The globalization is a technique to identify the specific part of a Web application that is different for different languages and make separate that portion from the core of the Web application. The localization is a procedure of configuring a Web application to be supported for a specific language or locale.

**37. What is ViewState?**

The ViewState is a feature used by ASP.NET Web page to store the value of a page and its controls just before posting the page. Once the page is posted, the first task by the page processing is to restore the ViewState to get the values of the controls.

**38. Which method is used to force all the validation controls to run?**

The Page.Validate() method is used to force all the validation controls to run and to perform validation.

**39. What does the Orientation property do in a Menu control?**

Orientation property of the Menu control sets the horizontal or vertical display of a menu on a Web page. By default, the orientation is vertical.

**40. Differentiate between client-side and server-side validations in Web pages.**

Client-side validations take place at the client end with the help of JavaScript and VBScript before the Web page is sent to the server. On the other hand, server-side validations take place at the server end.

**41. What is garbage collection?**

Garbage collection is a heap-management strategy where a run-time component takes responsibility for managing the lifetime of the memory used by objects. This concept is not new to .NET - Java and many other languages/runtimes have used garbage collection for some time.

**42. What is serialization?**

Serialization is the process of converting an object into a stream of bytes.Deserialization is the opposite process, i.e. creating an object from a stream of bytes. Serialization/Deserialization is mostly used to transport objects (e.g. during remoting), or to persist objects (e.g. to a file or database).

**43. Where do you add an event handler?**  
It's the Attributesproperty, the Add function inside that property. e.g.btnSubmit.Attributes.Add("onMouseOver","someClientCode();")

**44. What do you mean by authentication and authorization?**

Authentication is the process of validating a user on the credentials(username and password) and authorization performs after authentication. After Authentication a user will be verified for performing the various tasks, It access is limited it is known as authorization.

**45. What is portable executable (PE) ?**

The file format used for executable programs and for files to be linked together to form executable programs

**46. Differences between DLL and EXE?**

.exe

1.These are outbound file.  
2.Only one .exe file exists per application.  
3..Exe cannot be shared with other applications.  
  
.dll  
1.These are inbound file .  
2.Many .dll files may exists in one application.  
3. .dll can be shared with other applications.

**47. What is shadowing?**

Shadowing is either through scope or through inheritance. Shadowing through inheritance is hiding a method of a base class and providing a new implementation for the same. This is the default when a derived class writes an implementation of a method of base class which is not declared as overridden in the base class. This also serves the purpose of protecting an implementation of a new method against subsequent addition of a method with the same name in the base class.’shadows’ keyword is recommended although not necessary since it is the default.

**48. What is Method Overriding? How to override a function in C#?**

An override method provides a new implementation of a member inherited from a base class. The method overridden by an override declaration is known as the overridden base method. The overridden base method must have the same signature as the override method.  
Use the override modifier to modify a method, a property, an indexer, or an event. You cannot override a non-virtual or static method. The overridden base method must be virtual, abstract, or override.

**49. Differences between dataset.clone and dataset.copy?**

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

**50. What is the managed and unmanaged code in .net?**

The .NET Framework provides a run-time environment called the Common Language Runtime, which manages the execution of code and provides services that make the development process easier. Compilers and tools expose the runtime's functionality and enable you to write code that benefits from this managed execution environment. Code that you develop with a language compiler that targets the runtime is called managed code; it benefits from features such as cross-language integration, cross-language exception handling, enhanced security, versioning and deployment support, a simplified model for component interaction, and debugging and profiling services.

**51. Whats an assembly?**

Assemblies are the building blocks of .NET Framework applications; they form the fundamental unit of deployment, version control, reuse, activation scoping, and security permissions. An assembly is a collection of types and resources that are built to work together and form a logical unit of functionality. An assembly provides the common language runtime with the information it needs to be aware of type implementations. To the runtime, a type does not exist outside the context of an assembly.

**52. How do you create a permanent cookie?**

Setting the Expires property to MinValue means that the Cookie never expires.

**53. What’s a Windows process in .NET?**

Windows process is an application that’s running and had been allocated memory in .NET

**54. What is Delegation in .NET?**

A delegate acts like a strongly type function pointer. Delegates can invoke the methods that they reference without making explicit calls to those methods.  
Delegate is an entity that is entrusted with the task of representation, assign or passing on information. In code sense, it means a Delegate is entrusted with a Method to report information back to it when a certain task (which the Method expects) is accomplished outside the Method's class.

**55. What is Serialization in .NET?**

The serialization is the process of converting the objects into stream of bytes.  
they or used for transport the objects(via remoting) and persist objects(via files and databases)

**56. Difference between Class And Interface in .NET?**

Class is logical representation of object. It is collection of data and related sub procedures with definition.  
Interface is also a class containing methods which is not having any definitions.  
Class does not support multiple inheritance. But interface can support

**57. Can any object be stored in a Viewstate in .NET?**

An object that either is serializable or has a TypeConverter defined for it can be persisted in ViewState.

**58 What is the use of ErrorProvider Control in .NET?**

The ErrorProvider control is used to indicate invalid data on a data entry form. Using this control, you can attach error messages that display next to the control when the data is invalid, as seen in the following image. A red circle with an exclamation point blinks, and when the user mouses over the icon, the error message is displayed as a tooltip.

**59. How do you validate the controls in an ASP .NET page?**

Using special validation controls that are meant for validation of any controle.  
We have Range Validator, Email Validator in .NET to validate any control.

**60. How to manage pagination in a page using .NET?**

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.

**61. Explain the difference between a class and an object.**

In short, a class is the definition of an object, and an object is instance of a class.

We can look at the class as a template of the object: it describes all the properties, methods, states and behaviors that the implementing object will have. As mentioned, an object is an instance of a class, and a class does not become an object until it is instantiated. There can be more instances of objects based on the one class, each with different properties.

**62. Explain the difference between managed and unmanaged code.**

Managed code is a code created by the .NET compiler. It does not depend on the architecture of the target machine because it is executed by the CLR (Common Language Runtime), and not by the operating system itself. CLR and managed code offers developers few benefits, like garbage collection, type checking and exceptions handling.

On the other hand, unmanaged code is directly compiled to native machine code and depends on the architecture of the target machine. It is executed directly by the operating system. In the unmanaged code, the developer has to make sure he is dealing with memory usage and allocation (especially because of memory leaks), type safety and exceptions manually.

In .NET, Visual Basic and C# compiler creates managed code. To get unmanaged code, the application has to be written in C or C++.

**63. Explain the difference between the while and for loop. Provide a .NET syntax for both loops.**

Both loops are used when a unit of code needs to execute repeatedly. The difference is that the for loop is used when you know how many times you need to iterate through the code. On the other hand, the while loop is used when you need to repeat something until a given statement is true.

The syntax of the while loop in C# is:

while (condition [is true])

{

// statements

}

The syntax of the for loop in C# is:

for (initializer; condition; iterator)

{

// statements

}

**64. Explain the difference between boxing and unboxing. Provide an example.**

Boxing is the process of converting a value type to the type object, and unboxing is extracting the value type from the object. While the boxing is implicit, unboxing is explicit.

Example (written in C#):

int i = 13;

object myObject = i; // boxing

i = (int)myObject; // unboxing

**65. Discuss the difference between constants and read-only variables.**

While constants and read-only variable share many similarities, there are some important differences:

* Constants are evaluated at compile time, while the read-only variables are evaluated at run time.
* Constants support only value-type variables, while read-only variables can hold reference-type variables.
* Constants should be used when the value is not changing during run time, and read-only variables are used mostly when their actual value is unknown before run time.
* Read-only variables can only be initialised at the time of declaration or in a constructor.

**66. Explain what LINQ is.**

LINQ is an acronym for Language Integrated Query, and was introduced with Visual Studio 2008. LINQ is a set of features that extends query capabilities to the .NET language syntax by adding sets of new standard query operators that allow data manipulation, regardless of the data source. Supported data sources are: .NET Framework collections, SQL Server databases, ADO.NET Datasets, XML documents, and any collection of objects that support IEnumerable or the generic IEnumerable<T> interface, in both C# and Visual Basic. In short, LINQ bridges the gap between the world of objects and the world of data.

**67. Discuss what garbage collection is and how it works. Provide a code example of how you can enforce garbage collection in .NET.**

Garbage collection is a low-priority process that serves as an automatic memory manager which manages the allocation and release of memory for the applications. Each time a new object is created, the common language runtime allocates memory for that object from the managed Heap. As long as free memory space is available in the managed Heap, the runtime continues to allocate space for new objects. However, memory is not infinite, and once an application fills the Heap memory space, garbage collection comes into play to free some memory. When the garbage collector performs a collection, it checks for objects in the managed Heap that are no longer being used by the application and performs the necessary operations to reclaim the memory. Garbage collection will stop all running threads, it will find all objects in the Heap that are not being accessed by the main program and delete them. It will then reorganize all the objects left in the Heap to make space and adjust all the Pointers to these objects in both the Stack and the Heap.

To enforce garbage collection in your code manually, you can run the following command (written in C#):

System.GC.Collect();

**68. What do the following acronyms in .NET stand for: IL, CIL, MSIL, CLI and JIT?**

IL, or Intermediate Language, is a CPU independent partially compiled code. IL code will be compiled to native machine code using current environmental properties by Just-In-Time compiler (JIT). JIT compiler translates the IL code to an assembly code and uses the CPU architecture of the target machine to execute a .NET application. In .NET, IL is called Common Intermediate Language (CIL), and in the early .NET days it was called Microsoft Intermediate Language (MSIL).

CLI, or Common Language Infrastructure, is an open specification developed by Microsoft. It is a compiled code library used for deployment, versioning, and security. In .NET there are two CLI types: process assemblies (EXE) and library assemblies (DLL). CLI assemblies contain code in CIL, and as mentioned, during compilation of CLI programming languages, the source code is translated into CIL code rather than into platform or processor specific object code.

To summarize:

1. When compiled, source code is first translated to IL (in .NET, that is CIL, and previously called MSIL).
2. CIL is then assembled into a bytecode and a CLI assembly is created.
3. Before code execution, CLI code is passed through the runtime’s JIT compiler to generate native machine code.
4. The computer’s processor executes the native machine code.

**69. Explain the difference between the Stack and the Heap.**

The short answer would be: in the Stack are stored value types (types inherited from System.ValueType), and in the Heap are stored reference types (types inherited from System.Object).

We can say the Stack is responsible for keeping track of what is actually executing and where each executing thread is (each thread has its own Stack). The Heap, on the other hand, is responsible for keeping track of the data, or more precise objects.

**70. Explain what inheritance is, and why it’s important.**

Inheritance is one of the most important concepts in object-oriented programming, together with encapsulation and polymorphism. Inheritance allows developers to create new classes that reuse, extend, and modify the behavior defined in other classes. This enables code reuse and speeds up development. With inheritance, developers can write and debug one class only once, and then reuse that same code as the basis for the new classes. The class whose members are inherited is called the base class, and the class that inherits those members is called the derived class. By default, all classes in .NET are inheritable.

**71. Explain the differences between an Interface and an Abstract Class in .NET.**

An **interface** merely declares a contract or a behavior that implementing classes should have. It may declare only properties, methods, and events with no access modifiers. All the declared members must be implemented.

An **abstract class** provides a partial implementation for a functionality and some abstract/virtual members that must be implemented by the inheriting entities. It can declare fields too.

Neither interfaces nor abstract classes can be instantiated.

**72. Explain the differences between an Interface and an Abstract Class in .NET.**

In LINQ, **deferred execution** simply means that the query is not executed at the time it is specified. Specifically, this is accomplished by assigning the query to a variable. When this is done, the query definition is stored in the variable but the query is not executed until the query variable is iterated over. For example:

DataContext productContext = new DataContext();

var productQuery = from product in productContext.Products

where product.Type == "SOAPS"

select product; // Query is NOT executed here

foreach (var product in productQuery) // Query executes HERE

{

Console.WriteLine(product.Name);

}

You can also force **immediate execution** of a query. This can be useful, for example, if the database is being updated frequently, and it is important in the logic of your program to ensure that the results you’re accessing are those returned at the point in your code where the query was specified. Immediate execution is often forced using a method such as Average, Sum, Count, List, ToList, or ToArray. For example:

DataContext productContext = new DataContext();

var productCountQuery = (from product in productContext.Products

where product.Type == "SOAPS"

select product).Count(); // Query executes HERE

**73. What is a delegate in .NET?**

A delegate in .NET is similar to a function pointer in C or C++. Using a delegate allows the programmer to encapsulate a reference to a method inside a delegate object. The delegate object can then be passed to code which can call the referenced method, without having to know at compile time which method will be invoked. In addition, we could use delegate to create custom event within a class. For example,

public delegate void FooDelegate();

class FooClass

{

// custom event

public event FooDelegate FooEvent;

}

FooClass FooObj = new FooClass()

FooObj.FooEvent += new FooDelegate();

**74. Why can’t you specify access modifiers for items in an interface?**

It is always public

**75. When break is used inside two nested for loops, control comes out of which loop, the inner or the outer for loop? (I.e. does it break from all the present loops?)**

It breaks from the inner loop only.

**76. You would know that System.Object is the parent class of all .NET classes; In other words all types in .NET (whether implicit, explicit, or user-created) derive from the System.Object class. What are the various methods provided to System.Object’s deriving classes/types?**

System.Object provides the following important methods, among others:

1. ToString—Returns a string that represents the current object
2. both overrides of Equals(object), Equals(object, object)
3. GetHashCode
4. Finalize
5. GetType
6. ReferenceEquals
7. MemberwiseClone

Most of these methods provide the basic implementation required of any type that a developer will work with in the .NET stack.