Jack N @ GitHub

Full stack engineer, focus on: Angular/React, node.js/.Net

2021 Top 100 C#/.NET Interview Questions And Answers

2021-04-22 | 2021-04-23

> https://jackniu81.github.io/2021/04/22/2021-Top-100-C-NET-Interview-Questions-And-Answers/

1. What is the difference between “dispose” and “finalize” variables in C#?

1. Dispose - This method uses interface – “IDisposable” interface and it will free up both managed and unmanaged codes like – database connection, files etc.
2. Finalize - This method is called internally unlike Dispose method which is called explicitly. It is called by garbage collector and can’t be called from the code.

2. What does “Jagged Arrays” mean?

Answer: Jagged arrays refer to the array that has an element of type array. The dimensions and the sizes of the elements are different. An array of arrays is the other name of the jagged array.

3. What is a Class and an Object?

A ‘Class’ is an encapsulation of methods and properties that are used to represent an entity in real-time. Class brings all of the instances together in a single unit. An ‘Object’ is an instance of a Class, or a block of allocated memory that can be stored in the form of Variables, Array or a Collection.

4. Explain Code compilation in C#

Code compilation has four steps which include:

1. Compiling source code to Managed code by C# compiler
2. Executing the assembly by CLR
3. Combining the new code into assemblies
4. Loading the Common Language Runtime (CLR)

5. What is the difference between Virtual method and Abstract method?

A Virtual method must always have a default implementation. An Abstract method does not have an implementation. An override keyword is not necessary here, though it can be used.

6. What is a Hashtable in C#?

A Hashtable is a collection that stores (Keys, Values) pairs. Here, the Keys are used to find the storage location and is immutable and cannot have duplicate entries in a Hashtable. The .Net Framework has provided a Hash Table class that contains all the functionality required to implement a hash table without any additional development. The hash table is a general-purpose dictionary collection. Each item within the collection is a DictionaryEntry object with two properties: a key object and a value object. These are known as Key/Value. When items are added to a hash table, a hash code is generated automatically. This code is hidden from the developer. Access to the table’ s values is achieved using the key object for identification. As the items in the collection are sorted according to the hidden hash code, the items should be considered to be randomly ordered.

7. What is LINQ in C#?

LINQ stands for Language Integrated Query. LINQ is a data querying methodology that provides querying capabilities to .NET languages with a syntax similar to a SQL query.

LINQ has a great power of querying on any source of data. The data source could be collections of objects, database or XML files. We can easily retrieve data from any object that implements the IEnumerable interface.

Advantages of LINQ

- LINQ offers an object-based, language-integrated way to query over data no matter where that data came from. So through LINQ, we can query a database and XML as well as collections.
- Compile-time syntax checking.

8. What are Indexers in C#?

C# introduces a new concept known as Indexers which are used for treating an object as an array. The indexers are usually known as smart arrays in C#. They are not an essential part of object-oriented programming.

Defining an indexer allows you to create classes that act as virtual arrays. Instances of that class can be accessed using the [] array access operator.

9. Difference between the Equality Operator (==) and Equals() Method in C#

Both the == Operator and the Equals() method are used to compare two value type data items or reference type data items. The Equality Operator (==) is the comparison operator and the Equals() method compares the contents of a string. The == Operator compares the reference identity while the Equals() method compares only contents.

10. What’ s the Difference between the Is and As operator in C#

- “is” operator: In C# language, we use the “is” operator to check the object type. If two objects are of the same type, it returns true, else it returns false.

- “as” operator: The “as” operator behaves in a similar way as the “is” operator. The only difference is it returns the object if both are compatible with that type. Else it returns a null.

11. What are Different Ways a Method can be Overloaded?

Method overloading is a way to achieve compile-time polymorphism where we can use a method with the same name but different signatures. For example, the following code example has a method volume with three different signatures based on the number and type of parameters and return values.

12. What is Serialization?

Serialization converts a code to its binary format using a process. After it is converted to bytes, it can be easily stored and written to a disk. Serializations are useful so that the original form of the code isn’t lost and it can be retrieved later on.

13. What are the different types of Delegates?

The different types of Delegates are: Single Delegate, Multicast Delegate and Generic Delegate.

14. How to use extension methods?

An extension method is a static method of a static class, where the “this” modifier is applied to the first parameter. The type of the first parameter will be the type that is extended.

Extension methods are only in scope when you explicitly import the namespace into your source code with a using directive.

15. What is the difference between String and StringBuilder in C#?

StringBuilder and string are both used to string values, but both have many differences on the bases of instance creation and also in performance.

16. What are sealed classes in C#?

Sealed classes are used to restrict the inheritance feature of object-oriented programming. Once a class is defined as a sealed class, the class cannot be inherited.

In C#, the sealed modifier is used to define a class as sealed. In Visual Basic .NET the Not Inheritable keyword serves the purpose of the sealed class. If a class is derived from a sealed class then the compiler throws an error.

If you have ever noticed, structs are sealed. You cannot derive a class from a struct.

17. What is a Delegate? Explain.

A Delegate is a variable that holds the reference to a method. Hence it is a function pointer or reference type. All Delegates are derived from System.Delegate namespace. Both Delegate and the method that it refers to can have the same signature.

Declaring a delegate: public delegate void AddNumbers(int n);
After the declaration of a delegate, the object must be created by the delegate using the new keyword.

```
AddNumbers an1 = new AddNumbers(number);
```

The delegate provides a kind of encapsulation to the reference method, which will internally get called when a delegate is called.

```
public delegate int myDel(int number);
public class Program
{
    public int AddNumbers(int a)
    {
        int Sum = a + 10;
        return Sum;
    }
    public void Start()
```

```
10      {
11          myDel DelgateExample = AddNumbers;
12      }
13  }
```

In the above example, we have a delegate myDel which takes an integer value as a parameter. Class Program has a method of the same signature as the delegate, called AddNumbers().

If there is another method called Start() which creates an object of the delegate, then the object can be assigned to AddNumbers as it has the same signature as that of the delegate.

18. What are Events?

Events are user actions that generate notifications to the application to which it must respond. The user actions can be mouse movements, keypress and so on.

Programmatically, a class that raises an event is called a publisher and a class which responds/receives the event is called a subscriber. Event should have at least one subscriber else that event is never raised.

Delegates are used to declare Events.

```
1  Public delegate void PrintNumbers();
2  Event PrintNumbers myEvent;
```

19. How to use Delegates with Events?

Delegates are used to raise events and handle them. Always a delegate needs to be declared first and then the Events are declared.

20. What do Multicast Delegates mean?

A Delegate that points to more than one method is called a Multicast Delegate. Multicasting is achieved by using + and += operator.

21. Explain Publishers and Subscribers in Events.

Publisher is a class responsible for publishing a message of different types of other classes. The message is nothing but Event as discussed in the above questions.

From the Example in Q #32, Class Patient is the Publisher class. It is generating an Event deathEvent, which is received by the other classes.

Subscribers capture the message of the type that it is interested in. Again, from the Example of Q#32, Class Insurance and Bank are Subscribers. They are interested in event deathEvent of type void.

22. What are Synchronous and Asynchronous operations?

Synchronization is a way to create a thread-safe code where only one thread can access the resource at any given time. The asynchronous call waits for the method to complete before continuing with the program flow.

Synchronous programming badly affects the UI operations when the user tries to perform time-consuming operations since only one thread will be used. In Asynchronous operation, the method call will immediately return so that the program can perform other operations while the called method completes its work in certain situations.

In C#, Async and Await keywords are used to achieve asynchronous programming. Look at Q #43 for more details on synchronous programming.

23. What is Reflection in C#?

Reflection is the ability of a code to access the metadata of the assembly during runtime. A program reflects upon itself and uses the metadata to inform the user or modify its behavior. Metadata refers to information about objects, methods.

The namespace System.Reflection contains methods and classes that manage the information of all the loaded types and methods. It is mainly used for windows applications, For Example, to view the properties of a button in a windows form.

The MemberInfo object of the class reflection is used to discover the attributes associated with a class.

Reflection is implemented in two steps, first, we get the type of the object, and then we use the type to identify members such as methods and properties.

To get type of a class, we can simply use,

```
1   Type mytype = myClass.GetType();
```

Once we have a type of class, the other information about the class can be easily accessed.

```
1   System.Reflection.MemberInfo Info = mytype.GetMethod("AddNumbers");
```

Above statement tries to find a method with name AddNumbers in the class myClass.

24. What is a Generic Class?

Generics or Generic class is used to create classes or objects which do not have any specific data type. The data type can be assigned during runtime, i.e when it is used in the program.

In case of other data type parameter comparisons, instead of creating many overloaded methods, we can create a generic class and pass a substitute data type, i.e T. So, T acts as a datatype until it is used specifically in the Main() method.

25. What is a Jagged Array?

A Jagged array is an array whose elements are arrays. It is also called as the array of arrays. It can be either single or multiple dimensions.

int[] jaggedArray = new int[4][];

#) Name some properties of Array.

Properties of an Array include:

- Length: Gets the total number of elements in an array.
- IsFixedSize: Tells whether the array is fixed in size or not.
- IsReadOnly: Tells whether the array is read-only or not.

26. What is an Array Class?

An Array class is the base class for all arrays. It provides many properties and methods. It is present in the namespace system.

27. What is a String? What are the properties of a String Class?

A String is a collection of char objects. We can also declare string variables in c#.

string name = "C# Questions";
A string class in C# represents a string. The properties of the string class are:

- Chars get the Char object in the current String.
- Length gets the number of objects in the current String.

28. Name some properties of Thread Class.

Few Properties of thread class are:

- IsAlive – contains value True when a thread is Active.
- Name – Can return the name of the thread. Also, can set a name for the thread.
- Priority – returns the prioritized value of the task set by the operating system.
- IsBackground – gets or sets a value which indicates whether a thread should be a background process or foreground.
- ThreadState– describes the thread state.

29. What are the different states of a Thread?

Different states of a thread are:

- Unstarted – Thread is created.
- Running – Thread starts execution.
- WaitSleepJoin – Thread calls sleep, calls wait on another object and calls join on another thread.
- Suspended – Thread has been suspended.
- Aborted – Thread is dead but not changed to state stopped.
- Stopped – Thread has stopped.

30. What are Async and Await?

Async and Await keywords are used to create asynchronous methods in C#.

Asynchronous programming means that the process runs independently of main or other processes.

31. Async Keyword

Async keyword is used for the method declaration.

32. What is a Deadlock?

A Deadlock is a situation where a process is not able to complete its execution because two or more processes are waiting for each other to finish. This usually occurs in multi-threading.

Here a shared resource is being held by a process and another process is waiting for the first process to release it and the thread holding the locked item is waiting for another process to complete.

33. Explain Lock, Monitors, and Mutex Object in Threading.

- Lock keyword ensures that only one thread can enter a particular section of the code at any given time. `lock(ObjA)` means the lock is placed on ObjA until this process releases it, no other thread can access ObjA.

- Mutex is also like a lock but it can work across multiple processes at a time. WaitOne() is used to lock and ReleaseMutex() is used to release the lock. But Mutex is slower than lock as it takes time to acquire and release it.

- Monitor.Enter and Monitor.Exit implements lock internally. a lock is a shortcut for Monitors. `lock(objA)` internally calls.

```
Monitor.Enter(ObjA);
try
{
}
Finally {
    Monitor.Exit(ObjA));
}
```

34. What is a Race Condition?

Race condition occurs when two threads access the same resource and are trying to change it at the same time. The thread which will be able to access the resource first cannot be predicted.

If we have two threads, T1 and T2, and they are trying to access a shared resource called X. And if both the threads try to write a value to X, the last value written to X will be saved.

35. What is Thread Pooling?

Thread pool is a collection of threads. These threads can be used to perform tasks without disturbing the primary thread. Once the thread completes the task, the thread returns to the pool.

`System.Threading.ThreadPool` namespace has classes that manage the threads in the pool and its operations.

36. What is Serialization?

Serialization is a process of converting code to its binary format. Once it is converted to bytes, it can be easily stored and written to a disk or any such storage devices. Serializations are mainly useful when we do not want to lose the original form of the code and it can be retrieved anytime in the future.

Any class which is marked with the attribute `[Serializable]` will be converted to its binary form.

The reverse process of getting the C# code back from the binary form is called Deserialization.

To Serialize an object we need the object to be serialized, a stream that can contain the serialized object and namespace System.Runtime.Serialization can contain classes for serialization.

37. What are the types of Serialization?

The different types of Serialization are:

- XML serialization – It serializes all the public properties to the XML document. Since the data is in XML format, it can be easily read and manipulated in various formats. The classes reside in System.sml.Serialization.
- SOAP – Classes reside in System.Runtime.Serialization. Similar to XML but produces a complete SOAP compliant envelope that can be used by any system that understands SOAP.
- Binary Serialization – Allows any code to be converted to its binary form. Can serialize and restore public and non-public properties. It is faster and occupies less space.

38. What is an XSD file?

An XSD file stands for XML Schema Definition. It gives a structure for the XML file. It means it decides the elements that the XML should have and in what order and what properties should be present. Without an XSD file associated with XML, the XML can have any tags, any attributes, and any elements.

Xsd.exe tool converts the files to the XSD format. During Serialization of C# code, the classes are converted to XSD compliant format by xsd.exe.

39. What is an Object and a Class?

- Class is an encapsulation of properties and methods that are used to represent a real-time entity. It is a data structure that brings all the instances together in a single unit.

- Object is defined as an instance of a Class. Technically, it is just a block of memory allocated that can be stored in the form of variables, array or a collection.

40. What are the fundamental OOP concepts?

The four fundamental concepts of Object-Oriented Programming are:

- Encapsulation: Here, the internal representation of an object is hidden from the view outside the object' s definition. Only the required information can be accessed whereas the rest of the data implementation is hidden.
- Abstraction: It is a process of identifying the critical behavior and data of an object and eliminating the irrelevant details.
- Inheritance: It is the ability to create new classes from another class. It is done by accessing, modifying and extending the behavior of objects in the parent class.
- Polymorphism: The name means, one name, many forms. It is achieved by having multiple methods with the same name but different implementations.

41. What is Managed and Unmanaged code?

Managed code is a code that is executed by CLR (Common Language Runtime) i.e all application code is based on .Net platform. It is considered as managed because of the

.Net framework which internally uses the garbage collector to clear up the unused memory.

Unmanaged code is any code that is executed by application runtime of any other framework apart from .Net. The application runtime will take care of memory, security and other performance operations.

42. What is an Interface?

Interface is a class with no implementation. The only thing that it contains is the declaration of methods, properties, and events.

43. Explain Get and Set Accessor properties?

Get and Set are called Accessors. These are made use by Properties. The property provides a mechanism to read, write the value of a private field. For accessing that private field, these accessors are used.

1. Get Property is used to return the value of a property
2. Set Property accessor is used to set the value.

44. What is a Thread? What is Multithreading?

A Thread is a set of instructions that can be executed, which will enable our program to perform concurrent processing. Concurrent processing helps us do more than one operation at a time. By default, C# has only one thread. But the other threads can be created to execute the code in parallel with the original thread.

Thread has a life cycle. It starts whenever a thread class is created and is terminated after the execution. System.Threading is the namespace which needs to be included to create threads and use its members.

Threads are created by extending the Thread Class. Start() method is used to begin thread execution.

```
1  //CallThread is the target method//
2   ThreadStart methodThread = new ThreadStart(CallThread);
3   Thread childThread = new Thread(methodThread);
4   childThread.Start();
```

C# can execute more than one task at a time. This is done by handling different processes by different threads. This is called MultiThreading.

There are several thread methods that are used to handle multi-threaded operations: Start, Sleep, Abort, Suspend, Resume and Join.

45. What is “using” statement in C#?

“Using” keyword denotes that the particular namespace is being used by the program.

For Example, using System

Here, System is a namespace. The class Console is defined under System. So, we can use the console.writeline (“….”) or readline in our program.

46. Explain Abstraction.

Abstraction is one of the OOP concepts. It is used to display only the essential features of the class and hide unnecessary information.

Abstraction helps in knowing what is necessary and hiding the internal details from the outside world. Hiding of the internal information can be achieved by declaring such parameters as Private using the private keyword.

47. What are C# I/O classes? What are the commonly used I/O classes?

C# has System.IO namespace, consisting of classes that are used to perform various operations on files like creating, deleting, opening, closing, etc.

Some commonly used I/O classes are:

- File – Helps in manipulating a file.
- StreamWriter – Used for writing characters to a stream.
- StreamReader – Used for reading characters to a stream.
- StringWriter – Used for reading a string buffer.
- StringReader – Used for writing a string buffer.
- Path – Used for performing operations related to the path information.

48. What is StreamReader/StreamWriter class?

StreamReader and StreamWriter are classes of namespace System.IO. They are used when we want to read or write charact90, Reader-based data, respectively.

Some of the members of StreamReader are: Close(), Read(), Readline().

Members of StreamWriter are: Close(), Write(), Writeline().

```
Class Program1
{
    using(StreamReader sr = new StreamReader("C:\ReadMe.txt")
    {
        //----------------code to read-------------------//
    }
    using(StreamWriter sw = new StreamWriter("C:\ReadMe.txt"))
    {
        //-------------code to write------------------//
    }
}
```

49. What is a Destructor in C#?

Destructor is used to clean up the memory and free the resources. But in C# this is done by the garbage collector on its own. System.GC.Collect() is called internally for cleaning up. But sometimes it may be necessary to implement destructors manually.

For Example:

```
~Car()
{
    Console.writeline("….");
}
```

50. What is an Abstract Class?

An Abstract class is a class which is denoted by abstract keyword and can be used only as a Base class. This class should always be inherited. An instance of the class itself cannot be created. If we do not want any program to create an object of a class, then such classes can be made abstract.

Any method in the abstract class does not have implementations in the same class. But they must be implemented in the child class.

For Example:

```
abstract class AB1
{
    Public void Add();
}
Class childClass : AB1
{
    childClass cs = new childClass ();
    int Sum = cs.Add();
}
```

All the methods in an abstract class are implicitly virtual methods. Hence, the virtual keyword should not be used with any methods in the abstract class.

51. How can one use nullable types in .Net?

Nullable types are defined as the types which can either take the normal value or the null value.

52. How to use nullable types in .Net?

Value types can take either their normal values or a null value. Such types are called nullable types.

```
Int? someID = null;
If(someID.HasVAlue)
{
}
```

53. What is the difference between Continue and Break Statement?

Break statement breaks the loop. It makes the control of the program to exit the loop. Continue statement makes the control of the program to exit only the current iteration. It does not break the loop.

54. What is the difference between finally and finalize block?

finally block is called after the execution of try and catch block. It is used for exception handling. Regardless of whether an exception is caught or not, this block of code will be executed. Usually, this block will have a clean-up code.

finalize method is called just before garbage collection. It is used to perform clean up operations of Unmanaged code. It is automatically called when a given instance is not subsequently called.

55. What is an Array? Give the syntax for a single and multi-dimensional array?

An Array is used to store multiple variables of the same type. It is a collection of variables stored in a contiguous memory location.

For Example:

```
double numbers = new double[10];
```

```
2   int[] score = new int[4] {25,24,23,25};
```

A single dimensional array is a linear array where the variables are stored in a single row. Above example is a single dimensional array.

Arrays can have more than one dimension. Multidimensional arrays are also called rectangular arrays.

For Example, `int[,] numbers = new int[3,2] { {1,2} ,{2,3},{3,4} };`

56. What are Boxing and Unboxing?

Converting a value type to reference type is called ‘Boxing’. Explicit conversion of the same reference type that is created by boxing back to value type is called ‘Unboxing’.

57. What is an Array used for?

An Array is used to store multiple variables of the same type and is a collection of variables stored in a contiguous memory location.

58. What is an Escape Sequence? Name the sequences in C

An Escape sequence is denoted by a backslash (). The backslash indicates that the character that follows it should be interpreted literally or that it is a special character. An escape sequence is considered as a single character.

59. What are the basic String Operations?

The basic String Operations are: Concatenate, Modify, Search, Compare.

60. What is an Escape Sequence? Name some String escape sequences in C#.

An Escape sequence is denoted by a backslash (). The backslash indicates that the character that follows it should be interpreted literally or it is a special character. An escape sequence is considered as a single character.

String escape sequences are as follows:

- \n – Newline character
- \b – Backspace
- \ – Backslash
- \’ – Single quote
- \’ ’ – Double Quote

61. In C#.Net, what do indexers mean?

In C#, indexers are also known as smart arrays. They allow the instances of the class that are to be indexed in a similar as that of the array.

62. What are Regular expressions? Search a string using regular expressions?

Regular expression is a template to match a set of input. The pattern can consist of operators, constructs or character literals. Regex is used for string parsing and replacing the character string.

For Example:

- matches the preceding character zero or more times. So, a*b regex is equivalent to b, ab, aab, aaab and so on.

Searching a string using Regex:

```
1  static void Main(string[] args)
2  {
3      string[] languages = { "C#", "Python", "Java" };
4      foreach(string s in languages)
5      {
6          if(System.Text.RegularExpressions.Regex.IsMatch(s,"Python"))
```

```
7           {
8               Console.WriteLine("Match found");
9           }
10      }
11  }
```

The above example searches for "Python" against the set of inputs from the languages array. It uses Regex.IsMatch which returns true in case if the pattern is found in the input. The pattern can be any regular expression representing the input that we want to match.

63. State the difference between direct cast and ctype.

The difference between direct cast and ctype is that direct cast is used for the conversion of type of an object that requires the run time which is similar to the specified type in the direct cast. Whereas ctype is used for converting the conversion which is defined for the expression and the type.

64. How can one use the singleton design pattern in C#?

The singleton design pattern is used in C# when the class has one instance and the access is provided globally.

65. What are the basic String Operations? Explain.

Some of the basic string operations are:

- Concatenate: Two strings can be concatenated either by using a System.String.Concat or by using + operator.
- Modify: Replace(a,b) is used to replace a string with another string. Trim() is used to trim the string at the end or at the beginning.
- Compare: System.StringComparison() is used to compare two strings, either a case-sensitive comparison or not case sensitive. Mainly takes two parameters, original string, and string to be compared with.

- Search: StartWith, EndsWith methods are used to search a particular string.

66. What is Parsing? How to Parse a Date Time String?

Parsing converts a string into another data type.

For Example:

```
string text = “500”;
int num = int.Parse(text);
```

500 is an integer. So, the Parse method converts the string 500 into its own base type, i.e int.

Follow the same method to convert a DateTime string.

```
string dateTime = “Jan 1, 2018”;
DateTime parsedValue = DateTime.Parse(dateTime);
```

67. What are extension methods in C#?

Extension methods enable you to add methods to existing types without creating a new derived type, recompiling, or otherwise modifying the original type.

An extension method is a special kind of static method, but they are called as if they were instance methods on the extended type.

68. What is the difference between the Virtual method and the Abstract method?

The Virtual method must always have a default implementation. However, it can be overridden in the derived class, although it is not mandatory. It can be overridden using the override keyword.

An Abstract method does not have an implementation. It resides in the abstract class. It is mandatory that the derived class implements the abstract method. An override

keyword is not necessary here though it can be used.

69. What are the types of errors in C#?

Answer: Following are the two types of error in C#:

1. Compile-time error
2. Run time error

70. What is the difference between method overriding and method overloading?

In method `overriding`, we change the method definition in the derived class that changes the method behavior.

Method `overloading` is creating a method with the same name within the same class having different signatures.

71. 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.

72. What is an object pool in .NET?

An object pool is a container having objects ready to be used. It tracks the object that is currently in use, total number of objects in the pool. This reduces the overhead of creating and re-creating objects.

73. What is the difference between read-only and constant?

The difference between read-only and constant is that read-only is used during run time when the value has to be assigned. Constant variables are used during compilation time

for declaration and initialization.

74. What are partial classes?

A partial class is only used to split the definition of a class in two or more classes in the same source code file or more than one source file. You can create a class definition in multiple files, but it will be compiled as one class at run time. Also, when you create an instance of this class, you can access all the methods from all source files with the same object.

Partial Classes can be created in the same namespace. It isn't possible to create a partial class in a different namespace. So use the "partial" keyword with all the class names that you want to bind together with the same name of a class in the same namespace.

75. Can "this" be used in a static method?

No, "this" cannot be used in a static method as static methods are used for either static variables or static methods.

76. What is IEnumerable<> in C#?

IEnumerable is the parent interface for all non-generic collections in System.Collections namespace like ArrayList, HastTable etc. that can be enumerated. For the generic version of this interface as IEnumerable which a parent interface of all generic collections class in System.Collections.Generic namespace like List<> and more.

In System.Collections.Generic.IEnumerable have only a single method which is GetEnumerator() that returns an IEnumerator. IEnumerator provides the power to iterate through the collection by exposing a Current property and Move Next and Reset methods if we don't have this interface as a parent so we can't use iteration by foreach loop or can't use that class object in our LINQ query.

77. What is the difference between late binding and early binding in C#?

Early Binding and Late Binding concepts belong to polymorphism in C#. Polymorphism is the feature of object-oriented programming that allows a language to use the same name in different forms. For example, a method named Add can add integers, doubles, and decimals.

78. What is the Constructor Chaining in C#?

Constructor chaining is a way to connect two or more classes in a relationship as Inheritance. In Constructor Chaining, every child class constructor is mapped to a parent class Constructor implicitly by base keyword, so when you create an instance of the child class, it will call the parent’ s class Constructor. Without it, inheritance is not possible.

79. What’ s the difference between the Array.CopyTo() and Array.Clone()?

The Array.Clone() method creates a shallow copy of an array. A shallow copy of an Array copies only the elements of the Array, whether they are reference types or value types, but it does not copy the objects that the references refer to. The references in the new Array point to the same objects that the references in the original Array point to.

The CopyTo() static method of the Array class copies a section of an array to another array. The CopyTo method copies all the elements of an array to another one-dimension array. The code listed in Listing 9 copies contents of an integer array to an array of object types.

80. Can Multiple Catch Blocks be executed in C#?

We can use multiple catch blocks with a try statement. Each catch block can catch a different exception. The following code example shows how to implement multiple catch statements with a single try statement.

81. What are Value types and Reference types in C#?

In C#, data types can be of two types, value types, and reference types. Value type variables contain their object (or data) directly. If we copy one value type variable to another then we are actually making a copy of the object for the second variable. Both of them will independently operate on their values, Value type data types are stored on a stack and reference data types are stored on a heap.

In C#, basic data types include int, char, bool, and long, which are value types. Classes and collections are reference types.

82. How do you use the “using” statement in C#?

There are two ways to use the using keyword in C#. One is as a directive and the other is as a statement. Let’s explain!

1. using Directive

   Generally, we use the using keyword to add namespaces in code-behind and class files. Then it makes available all the classes, interfaces and abstract classes and their methods and properties on the current page. Adding a namespace can be done in the following two ways:
2. Using Statement

   This is another way to use the using keyword in C#. It plays a vital role in improving performance in Garbage Collection.

83. What are Anonymous Types in C#?

Anonymous types allow us to create new types without defining them. This is a way of defining read-only properties in a single object without having to define each type explicitly. Here, Type is generated by the compiler and is accessible only for the current block of code. The type of properties is also inferred by the compiler.

We can create anonymous types by using “new” keyword together with the object initializer.

84. Explain “static” keyword in C#?

“Static” keyword can be used for declaring a static member. If the class is made static then all the members of the class are also made static. If the variable is made static then it will have a single instance and the value change is updated in this instance.

85. What are the different types of classes in C#?

The different types of class in C# are

1. Partial class – Allows its members to be divided or shared with multiple .cs files. It is denoted by the keyword ‘Partial’.
   Sealed class – It is a class that cannot be inherited. To access the members of a sealed class, we need to create the object of the class. It is denoted by the keyword ‘Sealed’.
2. Abstract class – It is a class whose object cannot be instantiated. The class can only be inherited. It should contain at least one method. It is denoted by the keyword ‘abstract’.
3. Static class – It is a class which does not allow inheritance. The members of the class are also static. It is denoted by the keyword ‘static’. This keyword tells the compiler to check for any accidental instances of the static class.

86. What is Managed and Unmanaged code?

Managed code is a code that is executed by CLR (Common Language Runtime). It is called ‘managed code’ because of the .Net framework which uses the garbage collector internally to clear up unused memory. ‘__Unmanaged code’ is any code that is executed by the application runtime of any other framework apart from .Net. The application runtime will take care of security, memory and other performance operations.

87. Explain Namespaces in C#

Namespaces are used to organize large code projects. "System" is the most widely-used namespace in C#.

88. Explain Polymorphism

In programming, polymorphism means the same method but different implementations. It contains two types, Compile-time and Runtime. Compile time polymorphism is accomplished by operator overloading. Runtime polymorphism is accomplished by overriding. An example would be: a class has a method Void Add(), polymorphism is accomplished by Overloading the method, that is, void Add(int a, int b), void Add(int add) are all overloaded methods.

89. How is Exception Handling implemented in C#?

Exception handling is done using four keywords in C#:

1. Try – contains a block of code that checks an exception.
2. Catch – is a program that catches an exception with the help of exception handler.
3. Finally – is a block of code written to execute even if an exception is not caught.
4. Throw – Throws an exception when a problem occurs.

.NET # 面试题 # C#

‹ 安装umi后，使用umi提示不是内部或者外部命令 前端进阶（1）Web前端性能优化 ›