C Sharp Interview Questions and Answers

Introduction

C# programming is a must to know for creating high-performance and secure applications in the .NET environment. This list of C-Sharp interview questions and answers provides an in-depth review of C# programming fundamentals and advanced topics like Object-Oriented Programming (OOPs), LINQ, Asynchronous Programming (Async/Await), and Garbage Collection. This ensures that you can clearly communicate the importance of creating scalable and maintainable code for desktop, web, and mobile applications.
Are you ready to become a certified .NET programmer? Develop your technical skills with our professional C Sharp Programming Certification Course in Chennai and start creating world-class applications.

List of C Sharp Interview Questions for Freshers

1. Explain C#
2. Define CLR
3. Define the JIT compiler process
4. Explain Garbage Collection in C#

5. List the types of classes in C#
6. Differentiate struct and class in C#
7. Explain Enum in C#
8. Explain properties in C#
9. Define Reflection in C#
10. Explain Jagged Array

Get started with our C Sharp course syllabus.

C Sharp Interview Questions and Answers for Freshers

1. Explain C#.

Microsoft produced C Sharp, which operates on the .Net Framework, making it a contemporary programming language founded on OOP ideas. The C# programming language is easy to learn for people proficient in C, C++, or Java.

2. Define CLR.

CLR stands for Common Language Runtime is the basic and virtual machine component of the .Net framework.

.NET has CLR as the runtime environment to make the development process easier by offering various services like remoting, type-safety, robustness, thread management, memory management, and so on.

3. Define the JIT compiler process.

JIT is the Just In Time compiler that is part of CLR (Common Language Runtime), and it is responsible for managing the execution of .Net programs.

This is a language-specific compiler that converts source code into intermediate language and then into machine code.

4. Explain Garbage Collection in C#.

The garbage collection of the .Net Framework is used for automatic memory management. Whenever a class object is created at runtime, the memory space will be allocated to it in the heap memory.

5. List the types of classes in C#.

The various types of classes in C# are abstract class, partial class, sealed class, and static class.

6. Differentiate struct and class in C#.

A class is a user-defined prototype for which objects will be created. It combines fields and methods into a single unit. A struct, on the other hand, is a collection of variables of different data types under a single unit.

7. Explain Enum in C#.

“Enum,” also known as enumeration, is a data type used to assign names or string values to integral constants that make a program easy to read and maintain. The main purpose of using Enum is to define the user-defined data types that are known as enumerated data types.

8. Explain properties in C#.

Properties are the unique types of class members that offer a flexible mechanism to read, write, or compute the value of a private field. It enables data to be accessed easily to help promote the flexibility and safety of methods.

9. Define Reflection in C#.

Reflection is the process of identifying the metadata of types, fields, and methods in the code. It enables the user to obtain data on the loaded assemblies and the elements within them, like methods, value types, and classes.

10. Explain Jagged Array.

A jagged array is an array of arrays whose member arrays can be of various sizes. The length of every array index will be varied, and the elements of the jagged array are reference types and initialized to null by default.

Learn from scratch with our C Sharp tutorial for beginners.

List of C Sharp Interview Questions for Experienced

1. Differentiate late binding and early binding in C#.
2. Define indexers in .Net.
3. Explain boxing and unboxing in C#.
4. Explain partial classes in C#.
5. Explain delegates in C#.
6. Explain sealed classes in C# with syntax.
7. What are the most commonly used types of exceptions?
8. Explain the Singleton design pattern in C#.
9. Explain the ways to implement a singleton design pattern in C#.
10. Describe Events in C#.

Prepare your career for the best C Sharp salary for freshers in India.

C Sharp Technical Interview Questions and Answers for Experienced

1. Differentiate late binding and early binding in C#.

The C# compiler performs the binding with the help of the .NET Framework when the object is assigned to an object variable of a specific type.

It has two different types of bindings, as follows:

- **Early binding:** Early binding is fast and easy to code, as it recognizes and checks the methods and properties during compile time. The limitation of early binding is that it increases the number of runtime errors.
- **Late binding:** Late binding is performed by using virtual methods, and the compiler doesn't know what kind of object it is or what methods it holds, as

the objects are dynamic. Late binding performs more slowly as it needs lookups at runtime.

2. Define indexers in .Net.

Indexers are smart arrays in C# that allow the instances of a class to be indexed in the same manner as an array.

The array access operator ([]) can be implemented in the instance of this class to access an indexer that the user creates for a class that functions like a virtual array.

Syntax:

```
element_type this [int index]
{
        get //the get accessor
        {
        //return the value defined by the index
        }
        set //the set accessor
        {
        //set the value defined by the index
        }
}
```

3. Explain boxing and unboxing in C#.

Boxing and unboxing are used to allow a unified view of the type system in which a value of any type is treated as an object.

- Boxing is the process of converting a value type, like char or int, into a reference type object.
- Unboxing is the process of converting the reference type back into the value type, and it is an explicit conversion process.

4. Explain partial classes in C#.

The partial class is a unique feature in C# that provides a special ability for implementing the functionality of a single class into multiple files, and all the files will be combined into a single class file when the application is compiled.

The partial class is created using a partial keyword, and the following is the syntax to create a partial class.

```
public partial class_name

{

        //codes to execute

}
```

5. Explain delegates in C#.

Delegates in C# are objects that refer to a method, or they are reference type variables that hold references to the methods.

It is a type that defines references to methods with a particular parameter list for the return type and calls the method in a program for execution when it is required.

6. Explain sealed classes in C# with syntax.

Sealed classes are implemented in the C# program to restrict users from inheriting the class, and a class can be sealed using the sealed keyword.

It tells the compiler that the class is sealed and can’t be extended. No class will be derived from a sealed class.

```
sealed class class_name

{
```

```
        //data members

        //methods

        ….

    }
```

A method is sealed, and the method can't be overridden. It can be sealed in the classes where they have been inherited. It has to be declared as a virtual class in its base class if you want to declare a method as sealed.

7. What are the most commonly used types of exceptions?

An exception is an error that happens at runtime and uses the C# exception handling subsystem in a structured and controlled manner that handles runtime errors.

The main aim of exception handling is to automate the error handling code, as it defines standard exceptions for common program errors like divide-by-zero or index-out-of-range.

- ArrayTypeMismatchException comes when the Type of value stored is incompatible with the type of the array.
- DivideByZeroException comes when the user tries to divide an integer value by zero.
- IndexOutOfRangeException occurs when an array index is out-of-bounds if an exception occurred.
- InvalidCastException is an invalid runtime cast
- An OutOfMemoryException occurs when insufficient free memory exists to continue program execution.

- OverflowException is an arithmetic overflow that occurs.
- A NullReferenceException is a type of exception that was made to operate on a null reference, similar to an object.

8. Explain the Singleton design pattern in C#.

The singleton design pattern is a common design pattern for a class that has one instance in the program that provides global access to it. It is a class that allows only one instance of itself to be made and gives simple access to the instance.

There are various approaches and methods to perform a singleton design in C#, and they are as follows:

- Private and parameterized single constructor.
- Sealed classes.
- Static variable to hold a reference to the single instance.
- Public and static methods of getting the reference to the made instance.

9. Explain the ways to implement a singleton design pattern in C#.

Users can implement a singleton design pattern in C# in the following ways:

- No thread-safe singleton
- Thread-safety singleton
- Thread-safety singleton using Double-Check locking
- Thread-safe without a lock
- Using .Net 4’s lazy <T> type.

10. Describe Events in C#.

Delegates and events are similar to each other, with an event created using a delegate serving as a notification to inform them when an action has taken place.

They expand the set of programming tasks where C# can be implemented. It is the main feature of C# to automate notifications.

Events are members of a class, and they will be declared using the event keyword. It can be applied as follows:

event event-delegate event-name;

The ‘event-delegate’ is the name that the delegates used to support the event, and the ‘event-name’ is the name for the particular event object that is being declared in the C# program.

Conclusion

For a candidate to succeed in a C# interview, he/she needs to prove his/her high-level understanding of OOPs concepts, LINQ, and Asynchronous programming. Employers want developers who can go beyond the basic syntax of the programming language and effectively use Memory Management (GC), Delegates, and Generics, all of which are part of the .NET ecosystem. Your ability to clearly explain the differences between Interface and Abstract classes, as well as Exception Handling, will prove your worth as a developer for building high-scale enterprise-level software solutions. As the requirements of the software development industry change towards Cloud-Native development, your understanding of .NET Core and Microservices will make you stand out as a high-level software architect.
Are you ready to become a certified .NET expert? Do you want to learn the fundamentals of building enterprise-level software? Then, take the best C# & .NET Masterclass in our software training institute in Chennai.