Top .NET Interview Questions (2022) - InterviewBit

36:36 Estimated 7685 Words EN Language

Introduction to .NET and .NET Core Framework

.NET framework is developed by Microsoft, provides an environment to run, debug and deploy code onto web services and applications by using tools and functionalities like libraries, classes, and APIs. This framework uses object-oriented programming.

You can use different languages like C#, Cobol, VB, F#, Perl, etc. for writing .NET framework applications. This Framework supports services, websites, desktop applications, and many more on Windows. It provides functionalities such as generic types, automatic memory management, reflection, concurrency, etc. These functionalities will help to make the development easier and efficiently build high-quality web as well as client applications.

.NET Core is a newer version of the .NET framework and it is a general-purpose, cost-free, open-source development platform developed by Microsoft. .NET Core is a cross-platform framework that runs an application on different operating systems such as Windows, Linux, and macOS operating systems. This framework can be used to develop various kinds of applications like mobile, web, IoT, cloud, microservices, machine learning, game, etc.

Characteristics of .NET Core:

- **Free and open-source**: .NET Core source code project can be obtained from Github. It is free and licensed under the MIT and Apache licenses.
- **Cross-platform**: .NET Core is supported by different operating systems like Windows, macOS, and Linux.
- **Sharable**: A single consistent API model that is written in .NET Standard will be used by .NET Core and is common for all the .NET applications. The same library or API can be used on multiple platforms with different languages.
- **Friendly**: The .NET Core is compatible with .NET Framework, Mono, and Xamarin, through .NET Standard. It also supports working with different Web frameworks and libraries such as Angular, React, and JavaScript.
- **Fast**: .NET Core 3.0 is faster compared to the .NET Framework, .NET Core 2.2 and previous versions. It is also much faster than other server-side frameworks like Node.js and Java Servlet.

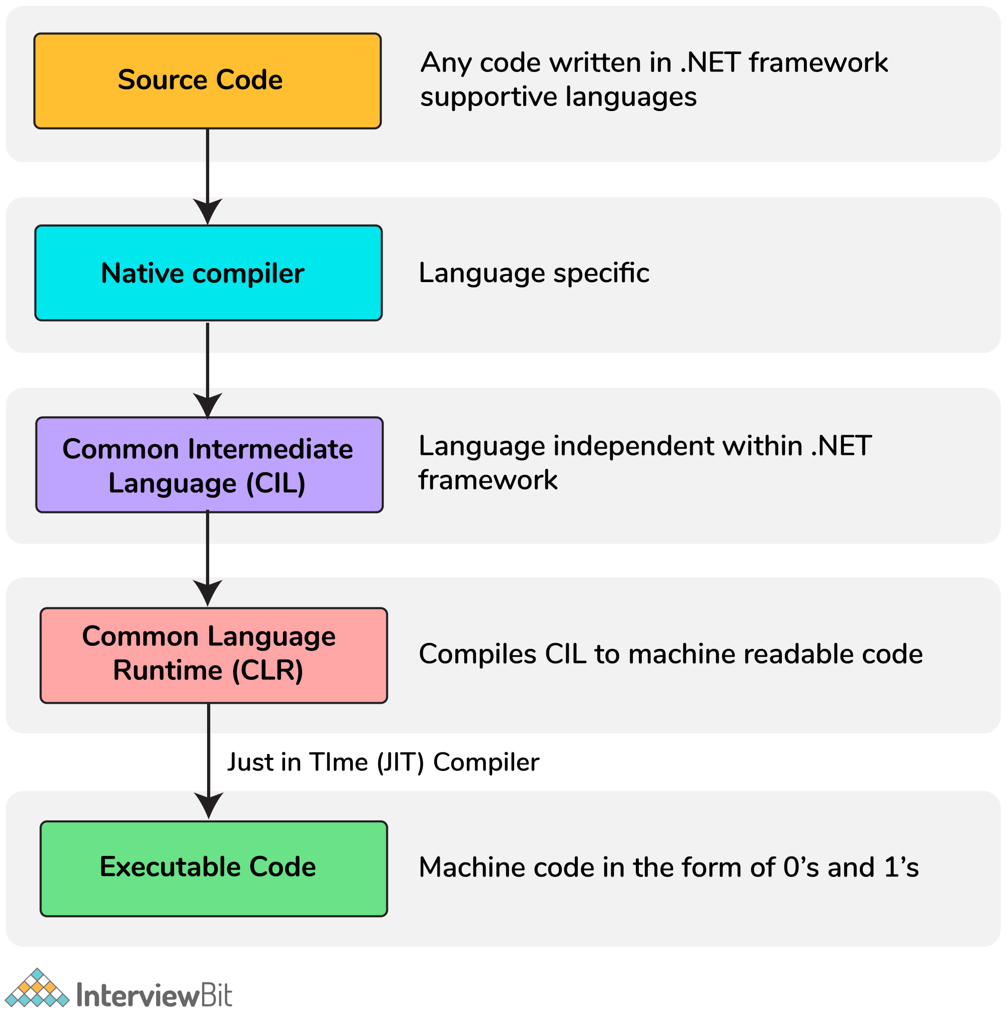
This article covers the most frequently asked .NET and .NET Core questions asked in interviews.

We have classified them into the following sections:

.NET Interview Questions

1. How does the .NET framework work?

- .NET framework-based applications that are written in supportive languages like C#, F#, or Visual basic are compiled to Common Intermediate Language (CIL).
- Compiled code is stored in the form of an assembly file that has a .dll or .exe file extension.
- When the .NET application runs, Common Language Runtime (CLR) takes the assembly file and converts the CIL into machine code with the help of the Just In Time(JIT) compiler.
- Now, this machine code can execute on the specific architecture of the computer it is running on.


Source Code
Any code written in .NET framework supportive languages
Native compiler
Language specific
Common Intermediate Language (CIL)
Language independent within .NET framework
Common Language Runtime (CLR)
Compiles CIL to machine readable code
Just in TIme (JIT) Compiler
Executable Code
Machine code in the form of 0's and 1's
InterviewBit

How .NET works?

2. Explain about major components of the .NET framework.

The major components .NET framework are given below:

- **Common Language Runtime(CLR):**
  - It is an execution engine that runs the code and provides services that make the development process easier.
  - Services provided by CLR are memory management, garbage collection, type safety, exception handling, security, and thread management. It also makes it easier for designing the applications and components whose objects interact across the languages.

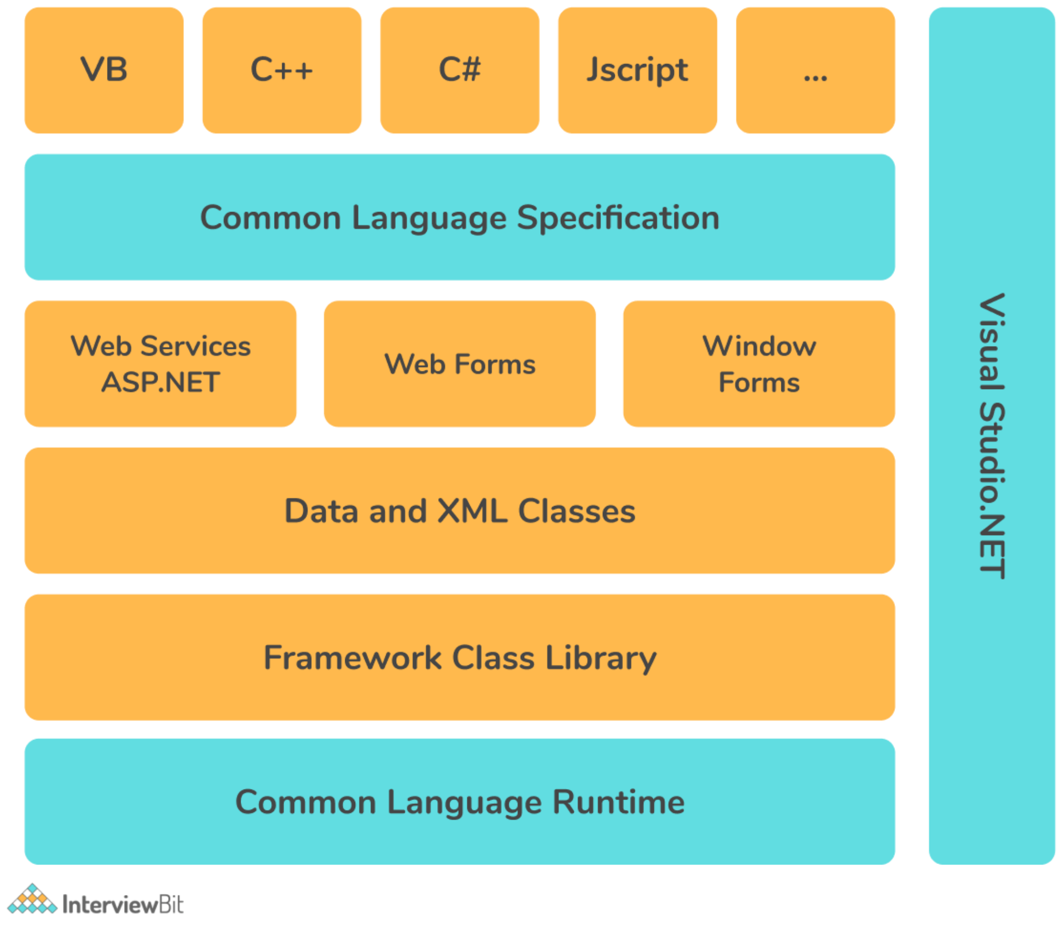
- The programs written for the .NET Framework are executed by the CLR regardless of programming language. Every .NET Framework version is having CLR.

- **Framework Class Library(FCL):**
  - It has pre-defined methods and properties to implement common and complex functions that can be used by .NET applications. It will also provide types for dates, strings, numbers, etc.
  - This class library includes APIs for database connection, file reading and writing, drawing, etc.
- **Base Class Library(BCL):**
  - The Base Class Library(BCL) has a huge collection of libraries features and functions that are helpful in implementing various programming languages such as C#, F#, Visual C++, etc., in the .NET Framework.
  - BCL is divided into two parts. They are:
    - **User-defined class library:** It includes Assemblies.

      Assembly: A .NET assembly is considered as the major building block of the .NET Framework. An assembly in the CLI(Common Language Infrastructure) is a logical unit of code, which is used for security, deployment, and versioning. Assembly can be defined in two forms namely Dynamic Link Library(.dll) and executable(.exe) files.

      When compilation of the .NET program takes place, metadata with Microsoft Intermediate Language(MSIL) will be generated and will be stored in a file called Assembly.
    - **Predefined class library:** It contains namespace.

      Namespace: It is the collection of pre-defined methods and classes that are present in the .Net Framework. A namespace can be added to a .NET program with the help of “using system”, where using represents a keyword and system represents a namespace.
- **Common Type System(CTS):**
  - CTS specifies a standard that will mention which type of data and value can be defined and managed in memory during runtime.
  - It will make sure that programming data defined in different languages should interact with each other for sharing the information. For example, in VB.NET we define datatype as integer, while in C# we define int as a data type.
  - It can be used to prevent data loss when you are trying to transfer data from a type in one language to its equivalent type in another language.
- **Common Language Specification (CLS):**
  - Common Language Specification (CLS) is a subset of CTS and defines a set of rules and regulations to be followed by every .NET Framework’s language.
  - A CLS will support inter-operability or cross-language integration, which means it provides a common platform for interacting and sharing information. For example, every programming language(C#, F#, VB .Net, etc.) under the .NET framework has its own syntax. So when statements belonging to different languages get executed, a common platform will be provided by the CLS to interact and share the information.



3. What is an EXE and a DLL?

EXE and DLLs are assembly executable modules.

EXE is an executable file that runs the application for which it is designed. An EXE is produced when we build an application. Therefore the assemblies are loaded directly when we run an EXE. However, an EXE cannot be shared with the other applications.

Dynamic Link Library (DLL) is a library that consists of code that needs to be hidden. The code is encapsulated inside this library. An application can consist of many DLLs which can be shared with the other programs and applications.

4. What is CTS?

CTS stands for Common Type System. It follows a set of structured rules according to which a data type should be declared and used in the program code. It is used to describe all the data types that are going to be used in the application.

We can create our own classes and functions by following the rules in the CTS. It helps in calling the data type declared in one programming language by other programming languages.

5. Explain CLS

Common Language Specification (CLS) helps the application developers to use the components that are inter-language compatible with certain rules that come with CLS. It also helps in reusing the code among all of the .NET-compatible languages.

6. What is JIT?

JIT stands for **Just In Time**. It is a compiler that converts the intermediate code into the native language during the execution.

7. What is the difference between int and Int32?

There is no difference between int and Int32. Int32 is a type provided by the .NET framework class whereas int is an alias name for Int32 in the C# programming language.

8. Explain the differences between value type and reference type.

The main differences between value type and reference type are given below:

- A Value Type holds the actual data directly within the memory location and a reference type contains a pointer which consists of the address of another memory location that holds the actual data.
- Value type stores its contents on the stack memory and reference type stores its contents on the heap memory.
- Assigning a value type variable to another variable will copy the value directly and assigning a reference variable to another doesn't copy the value, instead, it creates a second copy of the reference.
- Predefined data types, structures, enums are examples of value types. Classes, Objects, Arrays, Indexers, Interfaces, etc are examples of reference types.

9. What is the difference between managed and unmanaged code?

The main difference between managed and unmanaged code is listed below:

Managed Code	Unmanaged Code
It is managed by CLR.	It is not managed by CLR.
.NET framework is a must for execution.	Does not require a .NET framework for the execution.
Memory management is done through garbage collection.	Runtime environment takes care of memory management.

10. Explain Microsoft Intermediate Language

MSIL is the Microsoft Intermediate Language, which provides instructions for calling methods, memory handling, storing and initializing values, exception handling, and so on.

The instructions provided by MSIL are platform-independent and are generated by the language-specific compiler from the source code. JIT compiler compiles the MSIL into machine code based on the requirement.

11. What is an assembly?

An assembly is a file that is automatically generated by the compiler which consists of a collection of types and resources that are built to work together and form a logical unit of functionality. We can also say, assembly is a compiled code and logical unit of code.

Assemblies are implemented in the form of executable (.exe) or dynamic link library (.dll) files.

12. Is ASP.NET different from ASP? If yes, explain how?

Yes, ASP.NET and ASP(Active Server Pages) both are different. Let's check how they are different from each other.

- ASP.NET uses .NET languages such as C# and VB.NET, which are compiled to Microsoft Intermediate Language (MSIL). ASP uses VBScript. ASP code is interpreted during the execution.
- ASP.NET which is developed by Microsoft is used to create dynamic web applications while ASP is Microsoft's server-side technology used to create web pages.
- ASP.NET is fully object-oriented but ASP is partially object-oriented.
- ASP.NET has full XML Support for easy data exchange whereas ASP has no built-in support for XML.
- ASP.NET uses the ADO.NET technology to connect and work with databases. ASP uses ADO technology.

13. Explain role-based security in .NET

Role-based security is used to implement security measures in .NET, based on the roles assigned to the users in the organization. In the organization, authorization of users is done based on their roles.

For example, windows have role-based access like administrators, users, and guests.

14. Explain the different types of assembly.

Assemblies are classified into 2 types. They are:

Private Assembly:

- It is accessible only to the application.
- We need to copy this private assembly, separately in all application folders where we want to use that assembly. Without copying, we cannot access the private assembly.
- It requires to be installed in the installation directory of the application.

Shared or Public Assembly:

- It can be shared by multiple applications.
- Public assembly does not require copying separately into all application folders. Only one copy of public assembly is required at the system level, we can use the same copy by multiple applications.
- It is installed in the Global Assembly Cache(GAC).

15. What is the order of the events in a page life cycle?

There are eight events as given below that take place in an order to successfully render a page:

- Page_PreInit
- Page_Init
- Page_InitComplete
- Page_PreLoad

- Page_Load
- Page_LoadComplete
- Page_PreRender
- Render

16. What is a garbage collector?

Garbage collector frees the unused code objects in the memory. The memory heap is partitioned into 3 generations:

- Generation 0: It holds short-lived objects.
- Generation 1: It stores medium-lived objects.
- Generation 2: This is for long-lived objects.

Collection of garbage refers to checking for objects in the generations of the managed heap that are no longer being used by the application. It also performs the necessary operations to reclaim their memory. The garbage collector must perform a collection in order to free some memory space.

During the garbage collection process:

- The list of live objects is recognized.
- References are updated for the compacted objects.
- The memory space occupied by dead objects is recollected. The remaining objects are moved to an older segment.

`System.GC.Collect()` method is used to perform garbage collection in .NET.

17. What is caching?

Caching means storing the data temporarily in the memory so that the data can be easily accessed from the memory by an application instead of searching for it in the original location. It increases the speed and performance efficiency of an application.

There are three types of caching:

- Page caching
- Data caching
- Fragment caching

18. Can we apply themes to ASP.NET applications?

Yes. By modifying the following code in the `web.config` file, we can apply themes to ASP.NET applications:

```
<configuration>
   <system.web>
       <pages theme="windows"/>
   </system.web>
</configuration>
```

19. Explain MVC.

MVC stands for Model View Controller. It is an architecture to build .NET applications. Following are three main **logical components of MVC**: the model, the view, and the controller.

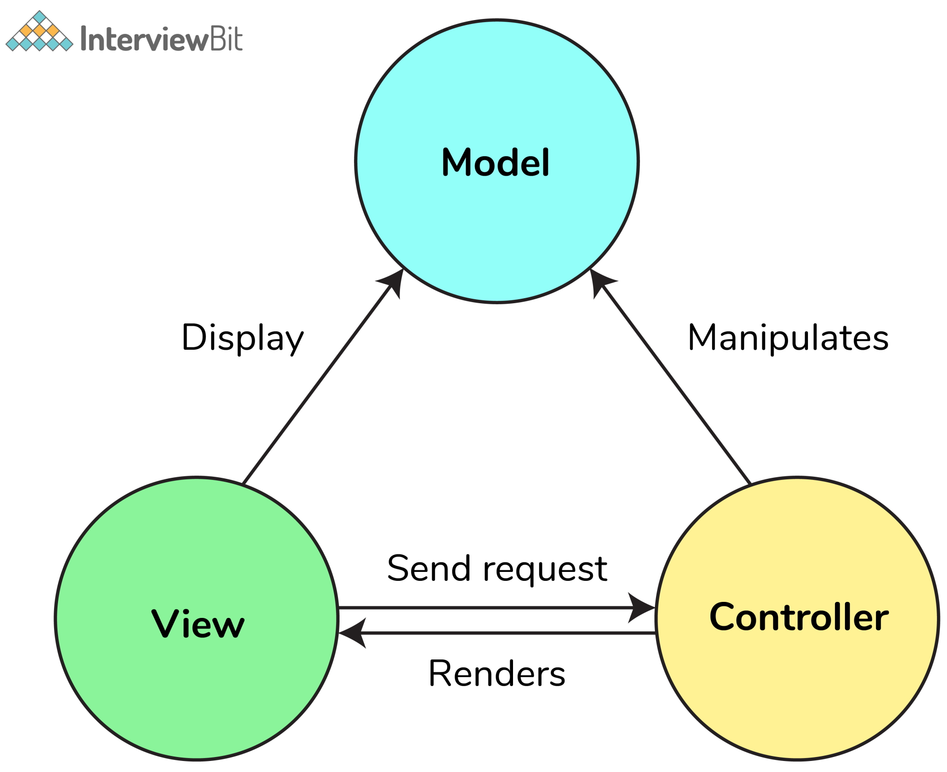


Fig: Components of MVC

Components of MVC

Model: They hold data and its related logic. It handles the object storage and retrieval from the databases for an application. For example:
A Controller object will retrieve the employee information from the database.
It manipulates employee data and sends back to the database or uses it to render the same data.

View: View handles the UI part of an application. They get the information from the models for their display. For example, any employee view will include many components like text boxes, dropdowns, etc.

Controller: They handle the user interactions, figure out the responses for the user input and also render the final output. For instance, the Employee controller will handle all the interactions and inputs from the Employee View and update the database using the Employee Model.

20. What is cross-page posting?

Whenever we click on a submit button on a webpage, the data is stored on the same page. But if the data is stored on a different page and linked to the current one, then it is known as a cross-page posting. Cross-page posting is achieved by `POSTBACKURL` property.

To get the values that are posted on this page to which the page has been posted, the FindControl method can be used.

21. What is a delegate in .NET?

A delegate is a .NET object which defines a method signature and it can pass a function as a parameter.

Delegate always points to a method that matches its specific signature. Users can encapsulate the reference of a method in a delegate object.

When we pass the delegate object in a program, it will call the referenced method. To create a custom event in a class, we can make use of delegate.

22. What are security controls available on ASP.NET?

Following are the five security controls available on ASP.NET:

- `<asp: Login>` Provides a login capability that enables the users to enter their credentials with ID and password fields.
- `<asp: LoginName>` Used to display the user name who has logged-in.
- `<asp: LoginView>` Provides a variety of views depending on the template that has been selected.
- `<asp: LoginStatus>` Used to check whether the user is authenticated or not.
- `<asp: PasswordRecovery>` Sends an email to a user while resetting the password.

23. What is boxing and unboxing in .NET?

Boxing is the process of converting a value type into a reference type directly. Boxing is implicit.

Unboxing is the process where reference type is converted back into a value type. Unboxing is explicit.

An example is given below to demonstrate boxing and unboxing operations:

```
int a = 10;       // a value type
object o = a;      // boxing
int b = (int)o;    // unboxing
```

24. What is MIME in .NET?

MIME stands for Multipurpose Internet Mail Extensions. It is the extension of the e-mail protocol which lets users use the protocol to exchange files over emails easily.

Servers insert the MIME header at the beginning of the web transmission to denote that it is a MIME transaction.

Then the clients use this header to select an appropriate ‘player’ for the type of data that the header indicates. Some of these players are built into the web browser.

25. What is the use of manifest in the .NET framework?

Manifest stores the metadata of the assembly. It contains metadata which is required for many things as given below:

- Assembly version information.
- Scope checking of the assembly.

- Reference validation to classes.
- Security identification.

26. Explain different types of cookies available in ASP.NET?

Two types of cookies are available in ASP.NET. They are:

- **Session Cookie:** It resides on the client machine for a single session and is valid until the user logs out.
- **Persistent Cookie:** It resides on the user machine for a period specified for its expiry. It may be an hour, a day, a month, or never.

27. What is the meaning of CAS in .NET?

Code Access Security(CAS) is necessary to prevent unauthorized access to programs and resources in the runtime. It is designed to solve the issues faced when obtaining code from external sources, which may contain bugs and vulnerabilities that make the user’s system vulnerable.

CAS gives limited access to code to perform only certain operations instead of providing all at a given point in time. CAS constructs a part of the native .NET security architecture.

28. What is the appSettings section in the web.config file?

We can use the appSettings block in the web.config file, if we want to set the user-defined values for the whole application. Example code given below will make use of ConnectionString for the database connection throughout the project:

```
<em>
   <configuration>
       <appSettings>
           <add key= "ConnectionString" value="server=local; pwd=password; databa
se=default"  />
       </appSettings>
   </configuration>
</em>
```

29. What is the difference between an abstract class and an interface?

The main difference between an abstract class and an interface are listed below:

Abstract Class	Interface
Used to declare properties, events, methods, and fields as well.	Fields cannot be declared using interfaces.
Provides the partial implementation of functionalities that must be implemented by inheriting classes.	Used to declare the behavior of an implementing class.
Different kinds of access modifiers like private, public, protected, etc. are supported.	Only public access modifier is supported.
It can contain static members.	It does not contain static members.

Abstract Class	Interface
Multiple inheritances cannot be achieved.	Multiple inheritances are achieved.

30. What are the types of memories supported in the .NET framework?

Two types of memories are present in .NET. They are:

Stack: Stack is a stored-value type that keeps track of each executing thread and its location. It is used for static memory allocation.

Heap: Heap is a stored reference type that keeps track of the more precise objects or data. It is used for dynamic memory allocation.

31. Explain localization and globalization.

Localization is the process of customizing our application to behave as per the current culture and locale.

Globalization is the process of designing the application so that it can be used by users from across the globe by supporting multiple languages.

32. What are the parameters that control the connection pooling behaviors?

There are 4 parameters that control the connection pooling behaviours. They are:

- Connect Timeout
- Min Pool Size
- Max Pool Size
- Pooling

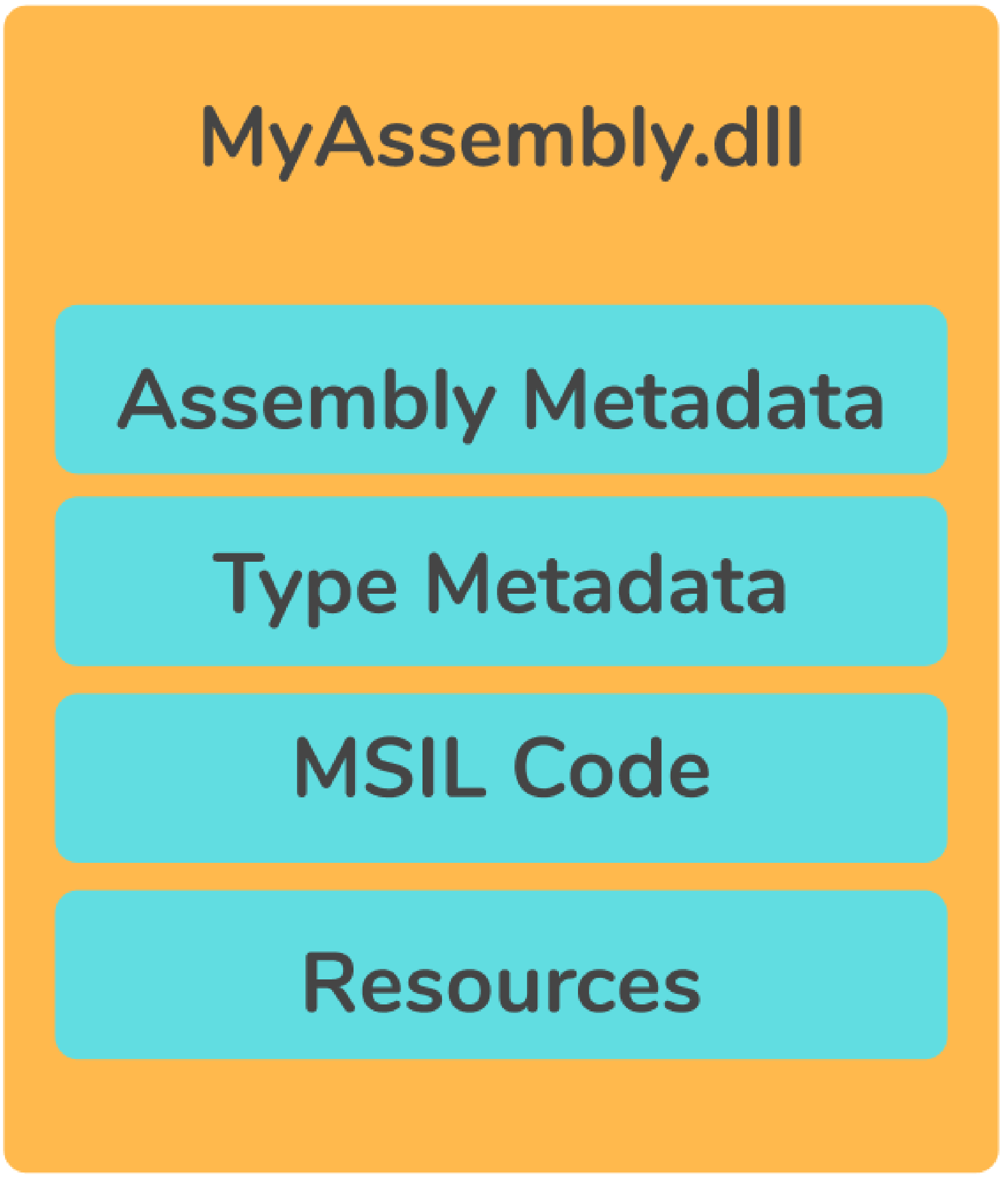
33. What are MDI and SDI?

MDI (Multiple Document Interface): An MDI allows you to open multiple windows, it will have one parent window and as many child windows. The components are shared from the parent window like toolbar, menubar, etc.

SDI (Single Document Interface): SDI opens each document in a separate window. Each window has its own components like a toolbar, menubar, etc. Therefore it is not constrained to the parent window.

34. Explain the different parts of an Assembly.

The different parts of an assembly are:



The different parts of an assembly are:

- **Manifest** – Every static or dynamic assembly holds a data collection that gives details about how the elements in the assembly relate to each other. An assembly manifest consists of complete metadata required to specify version requirements and security identity of an assembly, and also the metadata required for defining the assembly scope and resolving references to classes and resources.
  The assembly manifest will be stored in either a standalone PE(Portable Executable) file that holds only assembly manifest information, or in a PE file (a .exe or .dll) with MSIL(Microsoft intermediate language) code.
- **Type Metadata** – Metadata gives you additional information such as types, type names, method names, etc about the contents of an assembly. Metadata will be automatically generated by the

Compilers from the source files and the compiler will embed this metadata within target output files like .exe, .dll, or a .netmodule(in the case of multi-module assembly).

- **MSIL** – Microsoft Intermediate Language(MSIL) is a code that implements the types. It includes instructions to load, store, initialize, and call the methods on objects. Along with this, it also includes instructions for control flow, direct memory access, arithmetic and logical operations, exception handling, etc. This is generated by the compiler using one or more source code files. During the runtime, the JIT(Just In Time) compiler of CLR(Common Language Runtime) converts the MSIL code into native code to the Operating System.
- **Resources** – Resources can be a list of related files such as .bmp or .jpg files. These resources are static, which means they do not change during run time. Resources are not executable items.

.NET Core Interview Questions

35. What is .NET core?

.NET Core can be said as the newer version of the .NET Framework. It is a cost-free, general-purpose, open-source application development platform provided by Microsoft. It is a cross-platform framework because it runs on various operating systems such as Windows, Linux, and macOS. This Framework can be used to develop applications like mobile, web, IoT, machine learning, game, cloud, microservices, etc.

It consists of important features like a cross-platform, sharable library, etc., that are necessary for running a basic .NET Core application. The remaining features are supplied in the form of NuGet packages, that can be added to your application according to your needs. Like this we can say, the .NET Core will boost up the performance of an application, decreases the memory footprint, and becomes easier for maintenance of an application. It follows the modular approach, so instead of the entire .NET Framework installation, your application can install or use only what is required.

36. What is Dot NET Core used for?

- .NET Core is useful in the server application creations, that run on various operating systems like Windows, Mac, and Linux. Using this, developers can write libraries as well as applications in C#, F#, and VB.NET in both runtimes.
- Generally, it is used for cloud applications or for modifying large enterprise applications into microservices.
- .NET Core 3.0 supports cross-development between WPF, UWP, and Windows Forms.
- .NET Core supports microservices, which permits cross-platform services to work with the .NET Core framework including services developed with .NET Framework, Ruby, Java, etc.
- .NET Core's features like lightweight, modularity, and flexibility make it easier to deploy .NET Core applications in containers. These containers can be deployed on any platform, Linux, cloud, and Windows.

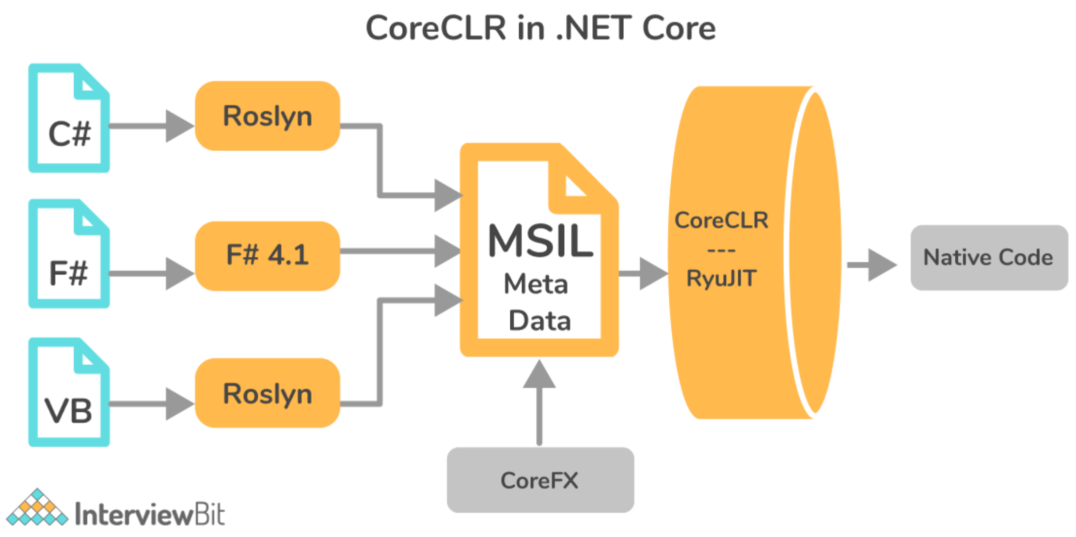
37. What are C# and F#?

C# is a general-purpose and object-oriented programming language from Microsoft that runs on the .NET platform. It is designed for CLI(Common Language Infrastructure), which has executable code and a runtime environment that allows for the usage of different high-level languages on various computer platforms and architectures. It is mainly used for developing web applications, desktop applications, mobile applications, database applications, games, etc.

F# is an open-source, functional-first, object-oriented and, cross-platform programming language that runs on a .NET platform and is used for writing robust, succinct, and performant code. We can say that F# is data-oriented because here code involves transforming data with functions. It is mainly used in making scientific models, artificial intelligence research work, mathematical problem solving, financial modelling, GUI games, CPU design, compiler programming, etc.

38. What is CoreCLR?

CoreCLR is the run-time execution engine provided by the .NET Core. It consists of a JIT compiler, garbage collector, low-level classes, and primitive data types. .NET Core is a modular implementation of .NET, and can be used as the base stack for large scenario types, ranging from console utilities to web applications in the cloud.



Here, various programming languages will be compiled by respective compilers(Roslyn can compile both C# and VB code as it includes C# and VB compilers) and Common Intermediate Language(CIL) code will be generated. When the application execution begins, this CIL code is compiled into native machine code by using a JIT compiler included within CoreCLR. This CoreCLR is supported by many operating systems such as Windows, Linux, etc.

39. What is the purpose of webHostBuilder()?

WebHostBuilder function is used for HTTP pipeline creation through `webHostBuilder.Use()` chaining all at once with `WebHostBuilder.Build()` by using the builder pattern. This function is provided by `Microsoft.AspNet.Hosting namespace`. The Build() method's purpose is building necessary services and a `Microsoft.AspNetCore.Hosting.IWebHost` for hosting a web application.

40. What is Zero Garbage Collectors?

Zero Garbage Collectors allows you for object allocation as this is required by the Execution Engine. Created objects will not get deleted automatically and theoretically, no longer required memory is never reclaimed.

There are two main uses of Zero Garbage Collectors. They are:

- Using this, you can develop your own Garbage Collection mechanism. It provides the necessary functionalities for properly doing the runtime work.
- It can be used in special use cases like very short living applications or almost no memory allocation(concepts such as No-alloc or Zero-alloc programming). In these cases, Garbage Collection overhead is not required and it is better to get rid of it.

41. What is CoreFx?

CoreFX is the set of class library implementations for .NET Core. It includes collection types, console, file systems, XML, JSON, async, etc. It is platform-neutral code, which means it can be shared across all platforms. Platform-neutral code is implemented in the form of a single portable assembly that can be used on all platforms.

42. What is the IGCToCLR interface?

IGCToCLR interface will be passed as an argument to the InitializeGarbageCollector() function and it is used for runtime communication. It consists of a lot of built-in methods such as RestartEE(), SuspendEE(), etc.

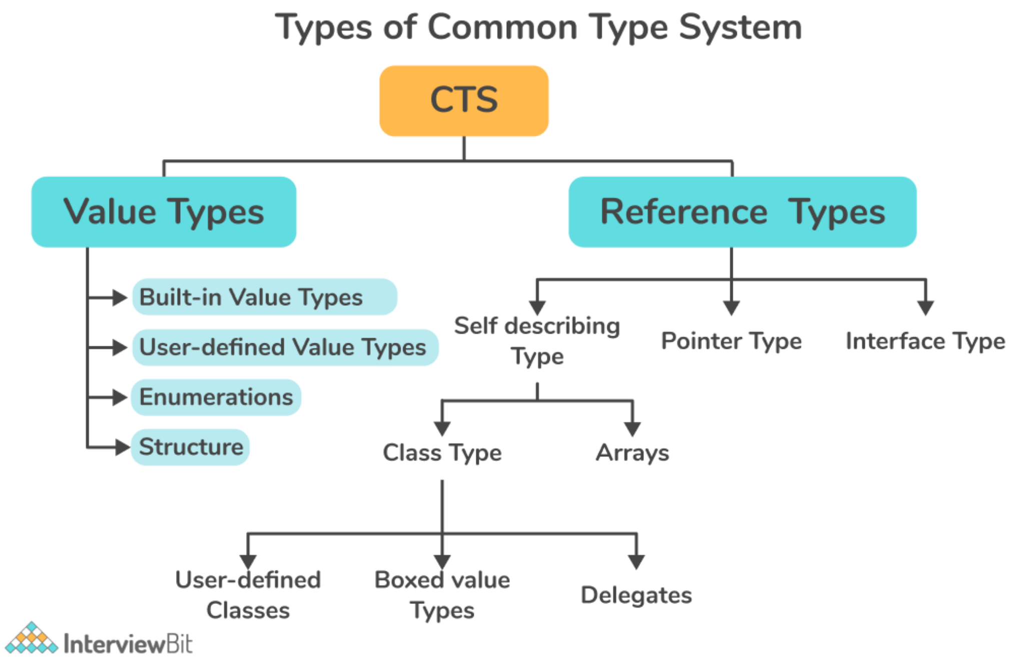
43. What is the use of generating SQL scripts in the .NET core?

It's useful to generate a SQL script, whenever you are trying to debug or deploy your migrations to a production database. The SQL script can be used in the future for reviewing the accuracy of data and tuned to fit the production database requirement.

44. Explain about types of Common Type System(CTS).

Common Type System(CTS) standardizes all the datatypes that can be used by different programming languages under the .NET framework.

CTS has two types. They are:



1. **Value Types:** They contain the values that are stored on a stack or allocated inline within a structure. They are divided into :
   - Built-in Value Types - It includes primitive data types such as Boolean, Byte, Char, Int32, etc.
   - User-defined Value Types - These are defined by the user in the source code. It can be enumeration or structure.
   - Enumerations - It is a set of enumerated values stored in the form of numeric type and are represented by labels.

- Structures - It defines both data(fields of the structure) and the methods(operations performed on that data) of the structure. In .NET, all primitive data types like Boolean, Byte, Char, DateTime, Decimal, etc., are defined as structures.

2. **Reference Types:** It Stores a reference to the memory address of a value and is stored on the heap. They are divided into :
    - Interface types - It is used to implement functionalities such as testing for equality, comparing and sorting, etc.
    - Pointer types - It is a variable that holds the address of another variable.
    - Self-describing types - It is a data type that gives information about themselves for the sake of garbage collectors. It includes arrays(collection of variables with the same datatype stored under a single name) and class types(they define the operations like methods, properties, or events that are performed by an object and the data that the object contains) like user-defined classes, boxed value types, and delegates(used for event handlers and callback functions).

45. Give the differences between .NET Core and Mono?

.NET Core	Mono
.Net Core is the subset of implementation for the .NET framework by Microsoft itself.	Mono is the complete implementation of the .Net Framework for Linux, Android, and iOS by Xamarin.
.NET Core only permits you to build web applications and console applications.	Mono permits you to build different application types available in .NET Framework, including mobile applications, GUI-enabled desktop apps, etc.
.NET Core does not have the built-in capability to be compiled into WebAssembly-compatible packages.	Mono has the built-in capability to be compiled into WebAssembly-compatible packages.
.NET Core is never intended for gaming. You can only develop a text-based adventure or relatively basic browser-based game using .NET Core.	Mono is intended for the development of Games. Games can be developed using the Unity gaming engine that supports Mono.

46. What is Transfer-encoding?

Transfer-encoding is used for transferring the payload body(information part of the data sent in the HTTP message body) to the user. It is a hop-by-hop header, that is applied not to a resource itself, but to a message between two nodes. Each multi-node connection segment can make use of various Transfer-encoding values.

Transfer-encoding is set to "Chunked" specifying that Hypertext Transfer Protocol's mechanism of Chunked encoding data transfer is initiated in which data will be sent in a form of a series of "chunks". This is helpful when the amount of data sent to the client is larger and the total size of the response will not be known until the completion of request processing.

47. Whether 'debug' and 'trace' are the same?

No. The Trace class is used for debugging as well as for certain build releases. It gives execution plan and process timing details. While debug is used mainly for debugging. Debug means going through the program code flow during execution time.

Debug and trace allow for monitoring of the application for errors and exceptions without VS.NET IDE.

----

48. What is MSBuild in the .NET Core?

MSBuild is the free and open-source development platform for Visual Studio and Microsoft. It is a build tool that is helpful in automating the software product creation process, along with source code compilation, packaging, testing, deployment, and documentation creation. Using MSBuild, we can build Visual Studio projects and solutions without the need of installing the Visual Studio IDE.

In the Universal Windows Platform(UWP) app, if you open the folder named project, you will get to see both files namely `project.json` and `*.csproj`. But if you open our previous Console application in .NET Core, you will get to see `project.json` and `*.xproj` files.

49. What are Universal Windows Platform(UWP) Apps in .Net Core?

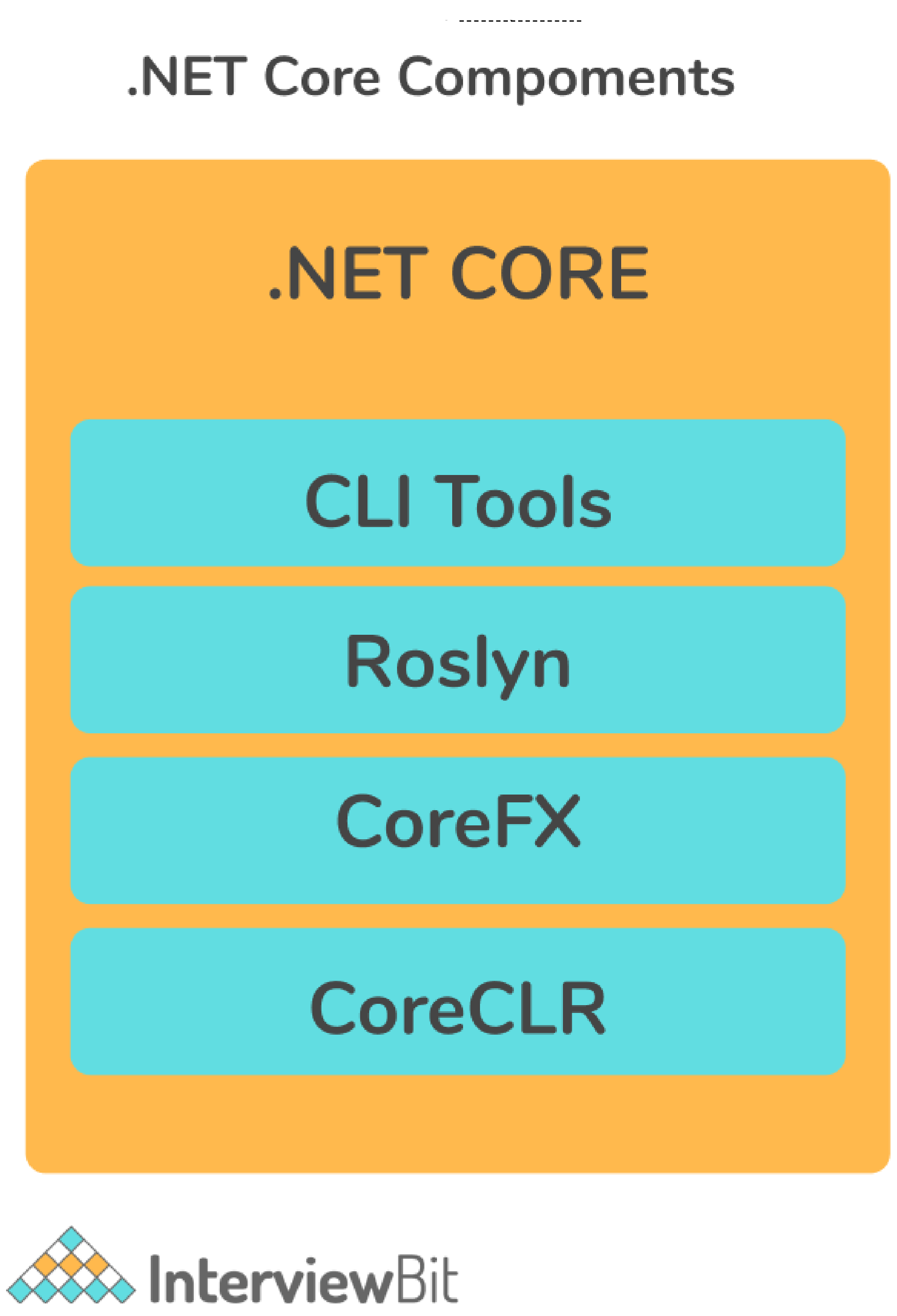
Universal Windows Platform(UWP) is one of the methods used to create client applications for Windows. UWP apps will make use of WinRT APIs for providing powerful UI as well as features of advanced asynchronous that are ideal for devices with internet connections.

Features of UWP apps:

- Secure: UWP apps will specify which resources of device and data are accessed by them.
- It is possible to use a common API on all devices(that run on Windows 10).
- It enables us to use the specific capabilities of the device and adapt the user interface(UI) to different device screen sizes, DPI(Dots Per Inches), and resolutions.
- It is available on the Microsoft Store on all or specified devices that run on Windows 10.
- We can install and uninstall these apps without any risk to the machine/incurring “machine rot”.
- Engaging: It uses live tiles, user activities, and push notifications, that interact with the Timeline of Windows as well as with Cortana’s Pick Up Where I Left Off, for engaging users.
- It can be programmable in C++, C#, Javascript, and Visual Basic. For UI, you can make use of WinUI, HTML, XAML, or DirectX.

50. Explain about .NET Core Components.

The .NET Core Framework is composed of the following components:



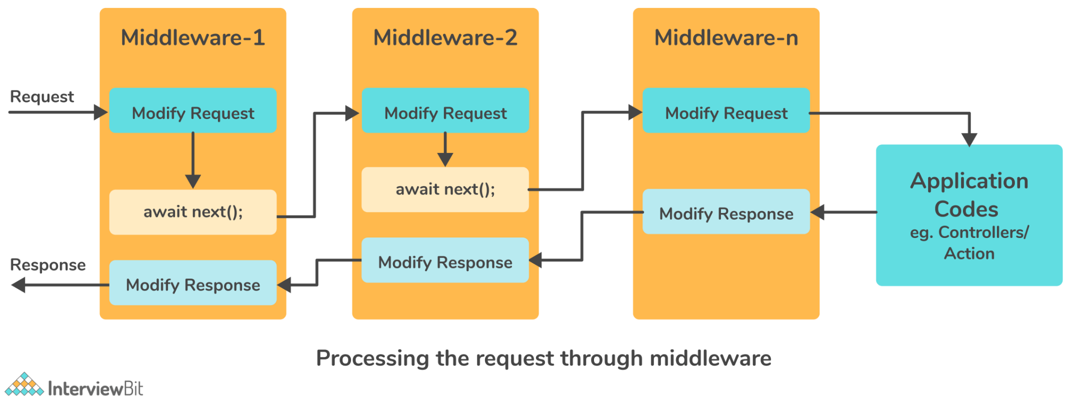
- **CLI Tools:** Command Line Interface(CLI) tools is a cross-platform tool for developing, building, executing, restoring packages, and publishing. It is also capable of building Console applications and class libraries that can run on the entire .NET framework. It is installed along with .NET Core SDK for the selected platforms. So it does not require separate installation on the development machine. We can verify for the proper CLI installation by typing `dotnet` on the command prompt of Windows and then pressing Enter. If usage and help-related texts are displayed, then we can conclude that CLI is installed properly.
- **Roslyn(.NET Compiler platform):** It is a set of an open-source language compiler and also has code analysis API for the C# and Visual Basic (VB.NET) programming languages. Roslyn exposes

modules for dynamic compilation to Common Intermediate Language(CLI), syntactic (lexical) and semantic code analysis, and also code emission.

- **CoreFX:** CoreFX is a set of framework libraries. It consists of the new BCL(Base Class Library) i.e. `System.*` things like `System.Xml`, `System.Collections`, etc.
- **CoreCLR:** A JIT(Just In Time) based CLR (Command Language Runtime). CoreCLR is the runtime implementation that runs on cross-platform and has the GC, RyuJIT, native interop, etc.

51. What is middleware in .NET core?

- Middleware is software assembled into an application pipeline for request and response handling. Each component will choose whether the request should be passed to the next component within the pipeline, also it can carry out work before and after the next component within the pipeline.
- For example, we can have a middleware component for user authentication, another middleware for handling errors, and one more middleware for serving static files like JavaScript files, images, CSS files, etc.
- It can be built-in into the .NET Core framework, which can be added through NuGet packages. These middleware components are built as part of the configure method's application startup class. In the ASP.NET Core application, these Configure methods will set up a request processing pipeline. It contains a sequence of request delegates that are called one after another.
- Normally, each middleware will handle the incoming requests and passes the response to the next middleware for processing. A middleware component can take the decision of not calling the next middleware in the pipeline. This process is known as short-circuiting the pipeline or terminating the request pipeline. This process is very helpful as it avoids unnecessary work. For example, if the request is made for a static file such as a CSS file, image, or JavaScript file, etc., these static files middleware can process and serve the request and thus short-circuit the remaining pipeline.



Processing the request through middleware

Here, there are three middlewares are associated with an ASP.NET Core web application. They can be either middleware provided by the framework, added through NuGet, or your own custom middleware. The HTTP request will be added or modified by each middleware and control will be optionally passed to the next middleware and a final response will be generated on the execution of all middleware components.

52. Differentiate .NET Core vs .NET framework.

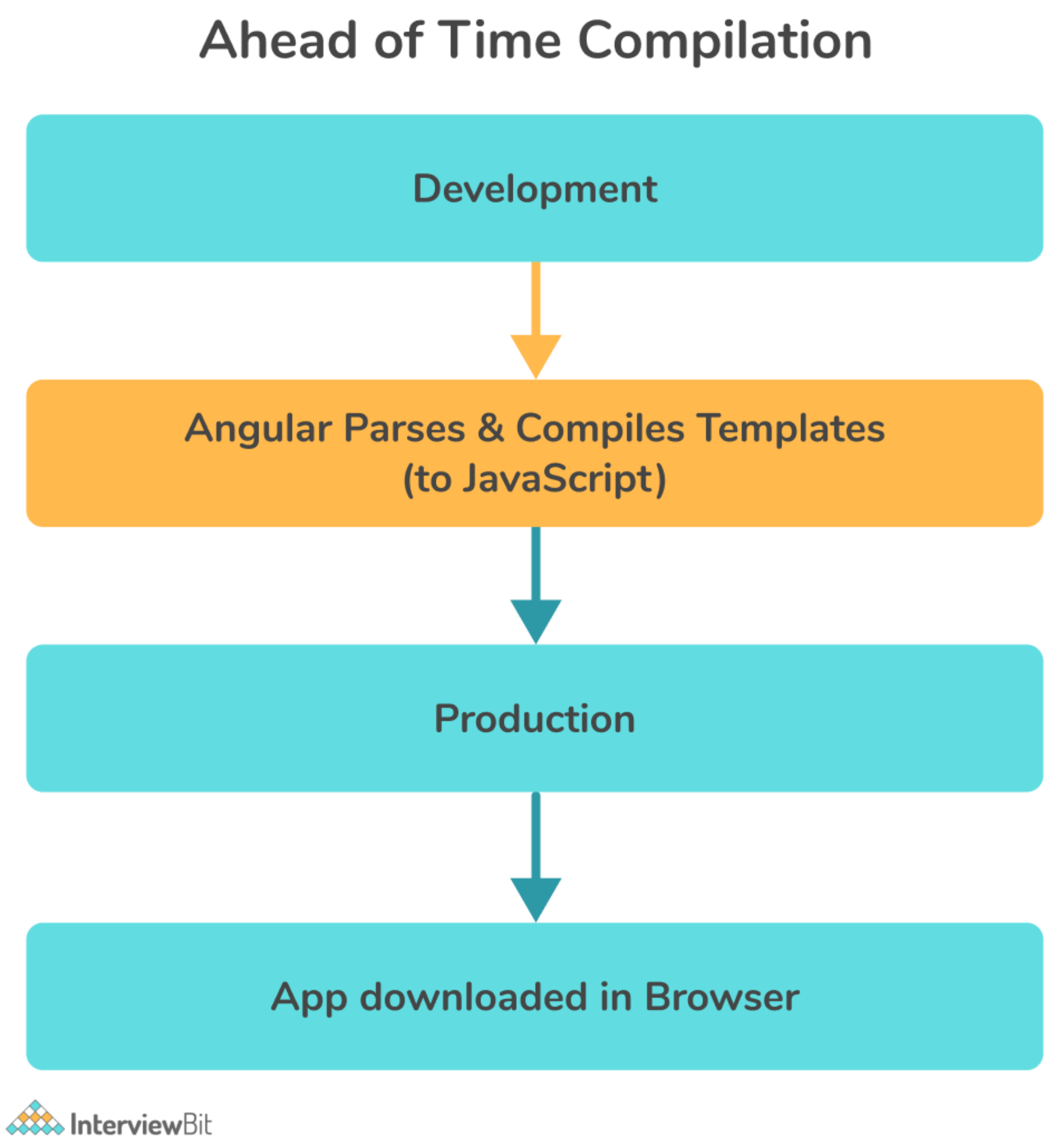
Features	.NET Core	.NET framework

Features	.NET Core	.NET framework
Compatibility	It works based on the principle of “build once, run anywhere”. It is cross-platform, so it is compatible with different operating systems such as Linux, Windows, and Mac OS.	This framework is compatible with the Windows operating system only. Even though, it was developed for supporting software and applications on all operating systems.
Installation	Since it is cross-platform, it is packaged and installed independently of the OS.	It is installed in the form of a single package for Windows OS.
Application Models	It does not support developing the desktop application and it focuses mainly on the windows mobile, web, and windows store.	It is used for developing both desktop and web applications, along with that it also supports windows forms and WPF applications.
Performance and Scalability	It provides high performance and scalability.	It is less effective compared to .Net Core in terms of performance as well as scalability of applications.
Support for Micro-Services and REST Services	It supports developing and implementing the micro-services and the user is required to create a REST API for its implementation.	It does not support the microservices’ development and implementation, but it supports REST API services.
Packaging and Shipping	It is shipped as a collection of Nugget packages.	All the libraries that belong to the .Net Framework are packaged and shipped all at once.
Android Development	It is compatible with open-source mobile app platforms like Xamarin, via .NET Standard Library. Developers can make use of tools of Xamarin for configuring the mobile application for particular mobile devices like Android, iOS, and Windows phones.	It does not support the development of mobile applications.
CLI Tools	For all supported platforms, it provides lightweight editors along with command-line tools.	This framework is heavy for CLI(Command Line Interface) and developers usually prefer to work on the lightweight CLI.
Deployment Model	Updated version of the .NET Core gets initiated on one machine at a time, which means it gets updated in new folders/directories in the existing application without affecting it. Thus, we can say that .NET Core has a very good flexible deployment model.	When the updated version is released, it is deployed only on the Internet Information Server at first.

53. Explain Explicit Compilation (Ahead Of Time compilation).

- Ahead-of-time(AOT) compilation is the process of compiling a high-level language into a low-level language during build-time, i.e., before program execution. AOT compilation reduces the workload during run time.
- AOT provides faster start-up time, in larger applications where most of the code executes on startup. But it needs more amount of disk space and memory or virtual address space to hold both IL(Intermediate Language) and precompiled images. In this case, the JIT(Just In Time) Compiler will do a lot of work like disk I/O actions, which are expensive.

- The explicit compilation will convert the upper-level language into object code on the execution of the program. Ahead of time(AOT) compilers are designed for ensuring whether the CPU will understand line-by-line code before doing any interaction with it.
- Ahead-of-Time (AOT) compilation happens only once during build time and it does not require shipping the HTML templates and the Angular compiler into the bundle. The source code generated can begin running immediately after it has been downloaded into the browser, earlier steps are not required. The AOT compilation will turn the HTML template into the runnable code fragment. AOT will analyze and compile our templates statically during build time.


Ahead of Time Compilation
Development
Angular Parses & Compiles Templates (to JavaScript)
Production
App downloaded in Browser
InterviewBit

Benefits of AOT Compilation:

- Application size is smaller because the Compiler itself isn’t shipped and unused features can be removed.
- Template the parse errors that are detected previously(during build time)
- Security is high (not required to dynamically evaluate templates)
- Rendering of a component is faster (pre-compiled templates)
- For AOT compilation, some tools are required to accomplish it automatically in the build process.

54. What is MEF?

The MEF(Managed Extensibility Framework) is a library that is useful for developing extensible and lightweight applications. It permits application developers for using extensions without the need for configuration. It also allows extension developers for easier code encapsulation and thus avoiding fragile hard dependencies. MEF will let you reuse the extensions within applications, as well as across the applications. It is an integral part of the .NET Framework 4. It improves the maintainability, flexibility, and testability of large applications.

55. In what situations .NET Core and .NET Standard Class Library project types will be used?

.NET Core library is used if there is a requirement to increase the surface area of the .NET API which your library will access, and permit only applications of .NET Core to be compatible with your library if you are okay with it.

.NET Standard library will be used in case you need to increase the count of applications that are compatible with your library and reduce surface area(a piece of code that a user can interact with) of the .NET API which your library can access if you are okay with it.

56. What is CoreRT?

- CoreRT is the native runtime for the compilation of .NET natively ahead of time and it is a part of the new .NET Native (as announced in April 2014).
- It is not a virtual machine and it does not have the facility of generating and running the code on the fly as it doesn’t include a JIT. It has the ability for RTTI(run-time type identification) and reflection, along with that it has GC(Garbage Collector).
- The type system of the CoreRT is designed in such a way that metadata for reflection is not at all required. This feature enables to have an AOT toolchain that can link away unused metadata and can identify unused application code.

57. What is .NET Core SDK?

.NET Core SDK is a set of tools and libraries that allows the developer to create a .NET application and library for .NET 5 (also .NET Core) and later versions. It includes the .NET CLI for building applications, .NET libraries and runtime for the purpose of building and running apps, and the dotnet.exe(dotnet executable) that runs CLI commands and runs an application. Here's the link to download.

58. What is Docker?

- Docker is an open-source platform for the development of applications, and also for shipping and running them. It allows for separating the application from the infrastructure using containers so that software can be delivered quickly. With Docker, you will be able to manage the infrastructure in the same ways you used to manage your applications.
- It supports shipping, testing, and deploying application code quickly, thus reducing the delay between code writing and running it in production.
- The Docker platform provides the ability of packaging and application execution in a loosely isolated environment namely container. The isolation and security permit you for running multiple containers at the same time on a given host. Containers are lightweight and they include every necessary thing required for running an application, so you need not depend on what is currently installed within the host.

----

59. What is Xamarin?

- Xamarin is an open-source platform useful in developing a modern and efficient application for iOS, Android, and Windows with .NET. It is an abstraction layer used to manage the communication of shared code with fundamental platform code.
- Xamarin runs in a managed environment that gives benefits like garbage collection and memory allocation.
- Developers can share about 90% of their applications over platforms using Xamarin. This pattern permits developers for writing entire business logic in a single language (or reusing existing app code) but accomplish native performance, look and feel on each platform. The Xamarin applications can be written on Mac or PC and then they will be compiled into native application packages, like a .ipa file on iOS, or .apk file on Android.

60. How can you differentiate ASP.NET Core from .NET Core?

.NET Core is a runtime and is used for the execution of an application that is built for it. Whereas ASP.NET Core is a collection of libraries that will form a framework for developing web applications. ASP.NET Core libraries can be used on .NET Core as well as on the “Full .NET Framework”.

An application using the tools and libraries of ASP.NET Core is normally referred to as “ASP.NET Core Application”, which in theory doesn’t say whether it is built for .NET Framework or .NET Core. So an application of “ASP.NET Core” can be considered as a “.NET Core Application” or a “.NET Framework Application”.

61. Write a program to calculate the addition of two numbers.

The steps are as follows:

1. You need to create a new ASP.NET Core Project “CalculateSum”. Open Visual Studio 2015, goto **File–> New–> Project**. Select the option Web in Left Pane and go for the option **ASP.NET Core Web Application** (.NET Core) under the central pane. Edit the project name as “CalculateSum” and click on OK.

2. In the template window, select Web Application and set the Authentication into “No Authentication” and click on OK.

3. Open “Solution Explorer” and right-click on the folder “Home” (It is Under Views), then click on **Add New Item**. You need to select **MVC View Page Template** under ASP.NET Section and rename it as “addition.cshtml” and then click on the **Add** button.

4. Open addition.cshtml and write the following code:

```
 @{
    ViewBag.Title = "Addition Page";
}

<h1>Welcome to Addition Page</h1>

<form asp-controller="Home" asp-action="add" method="post">

    <span>Enter First Number : </span> <input id="Text1" type="text" name="txtFir
stNum" /> <br /><br />
    <span>Enter Second Number : </span> <input id="Text2" type="text" name="txtSe
condNum" /> <br /><br />
```

```
    <input id="Submit1" type="submit" value="Add" />
</form>

<h2>@ViewBag.Result</h2>
```

Here, we have created a simple form that is having two text boxes and a single Add Button. The text boxes are named as `txtFirstNum` and `txtSecondNum`. On the controller page, we can access these textboxes using:
`<form asp-controller="Home" asp-action="add" method="post">`

This form will indicate all the submissions will be moved to HomeController and the method add action will be executed.

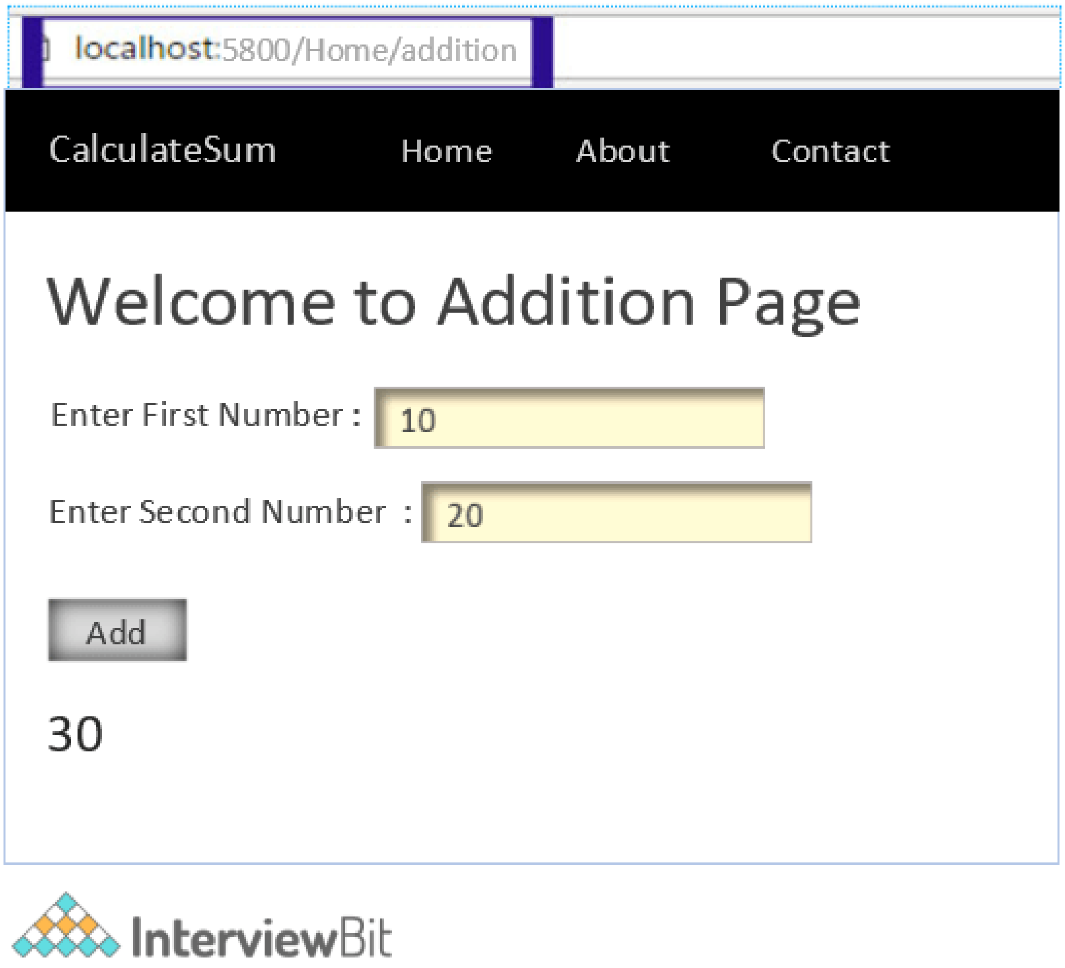
5. Open the `HomeController.cs` and write the following code onto it:

```
 using System;
using Microsoft.AspNetCore.Mvc;

namespace CalculateSum.Controllers
{
    public class HomeController : Controller
    {
        public IActionResult Index()
        {
            return View();
        }

        public IActionResult About()
        {
            ViewData["Message"] = "Application description page.";
            return View();
        }

        public IActionResult Contact()
        {
            ViewData["Message"] = "Contact page.";
            return View();
        }

        public IActionResult Error()
        {
            return View();
        }

        public IActionResult addition()
        {
            return View();
        }

        [HttpPost]
        public IActionResult add()
        {
            int number1 = Convert.ToInt32(HttpContext.Request.Form["txtFirstNu
m"].ToString());
            int number2 = Convert.ToInt32(HttpContext.Request.Form["txtSecondNu
m"].ToString());
            int res = number1 + number2;
```

```
                ViewBag.Result = res.ToString();
                return View("addition");
            }
        }
    }
```

In this program, we have added two IAction Methods addition() and add(). Addition() method will return the addition view page and add() method obtains input from the browser, processes it, and results will be kept in ViewBag.Result and then returned to the browser.

Now, press Ctrl+F5 for running your program. This will launch an ASP.NET Core website into the browser. Add `/Home/addition` at the end of the link and then hit on enter. The output format is given below:


localhost:5800/Home/addition

CalculateSum Home About Contact

Welcome to Addition Page

Enter First Number : 10

Enter Second Number : 20

Add

30

Conclusion

The .NET is a full-stack software development framework, which is essentially used to build large enterprise-scale and scalable software applications. The .NET framework has wide scope in the market. It is a flexible and user-friendly framework, that goes well along with other technologies.

The .NET Core was developed in response to the surge in Java popularity. The .NET Core is normally used in low-risk projects. Some of the .NET components can be used in .NET core applications (but not the other way around). This article mainly concentrates on the framework concepts of .Net and .NET Core. We are sure that it would give you sufficient information and a fair knowledge of the common questions that will be asked during an interview.

Useful Resources:

.Net Developer: Career Guide

ASP. NET Interview

C# Interview

.NET MCQ

System.Int32

System.Int8

System.Int16

System.Int64

Integer type to double

Value type to a reference type

Reference type to a value type

Double type to integer

No return type for events

Integer

String

Double

There is no such block as finally

Yes

No

Both catch and finally block will be executed.

Static

Serial

Local

Private

Value types

All data types in .net

Reference types

Communication between multiple languages

System.Web

System.IO

System.Object

System.File

GC.Run() method

GC.Collection() method

GC.Finalize() method

GC.Collect() method

Response.Output.Write() allows you to buffer output

Response.Output.Write() allows you to flush output

Response.Output.Write() allows you to write formatted output

Response.Output.Write() allows you to stream output

.NET Core 3.1

.NET Core 3.0

.NET 6

None of the above

Developing the applications

Shipping the applications

Running the applications

All of the above

CLI tools

Roslyn

CoreFX and CoreCLR

All of the above

Mix technologies across the service boundary

Debugging

Creating and deploying a container

None of the above

Windows only

Cross-platform

Linux only

MacOS only

Toolset for the development of applications

Compiler

Foundational class libraries for .NET Core

Docker container

Garbage Collector

JIT Compiler

Primitive data types

All of the above

.NET Compiler platform

Container

Library

None of the above

use of a memory-mapped file

queries that use LINQ(Language-Integrated Query)

to configure an assembly

immutable collections

NuGet package

LINQ

System

Dapper