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.NET Core is a newer version of the .NET framework and it is a general-purpose, cost-free, open-source development platform developed by Microsoft.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.

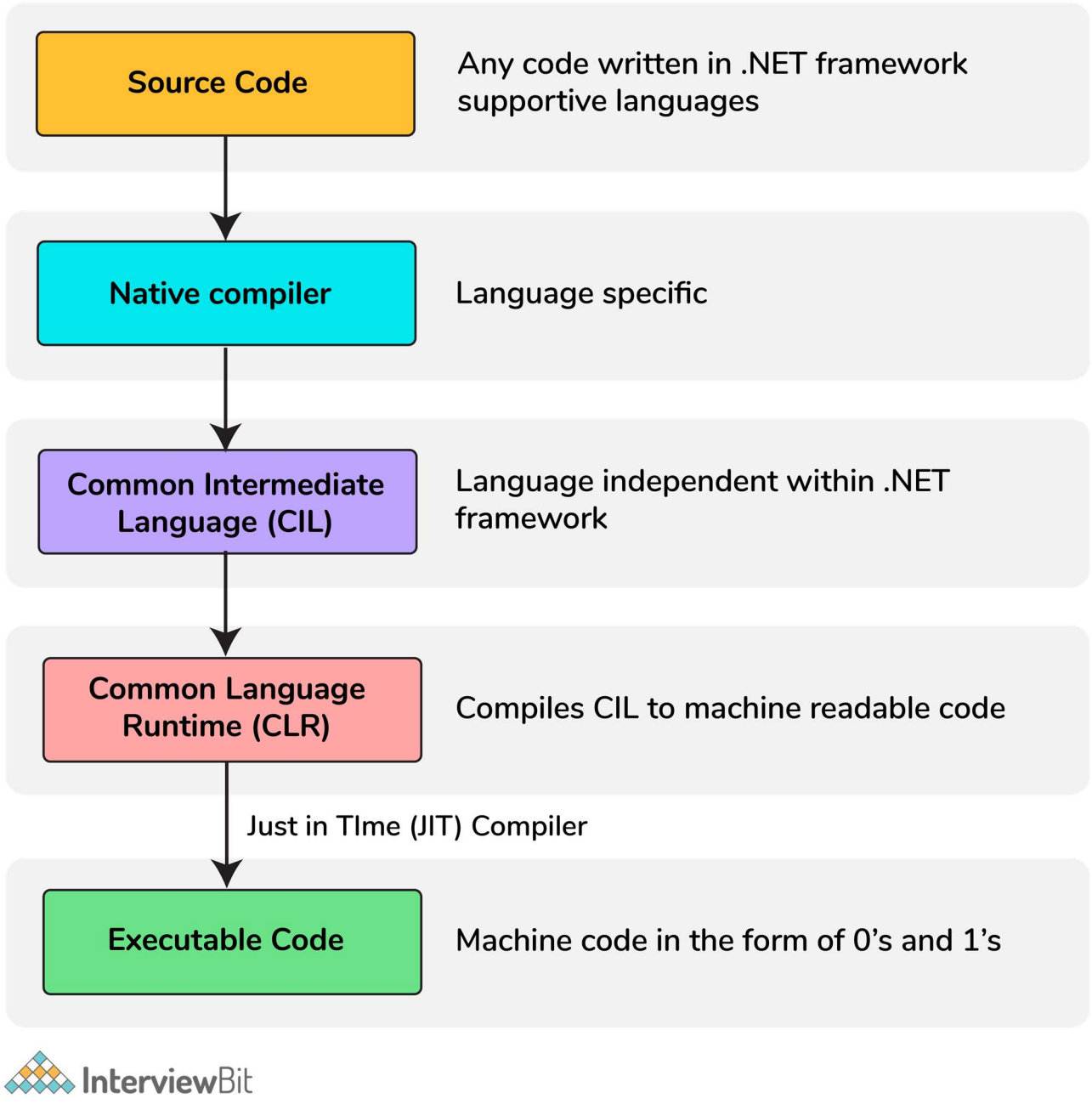
We have classified them into the following sections:

- .NET Interview Questions: Freshers and Experienced
- .NET Core Interview Questions: Freshers and Experienced

.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.



How .NET works?

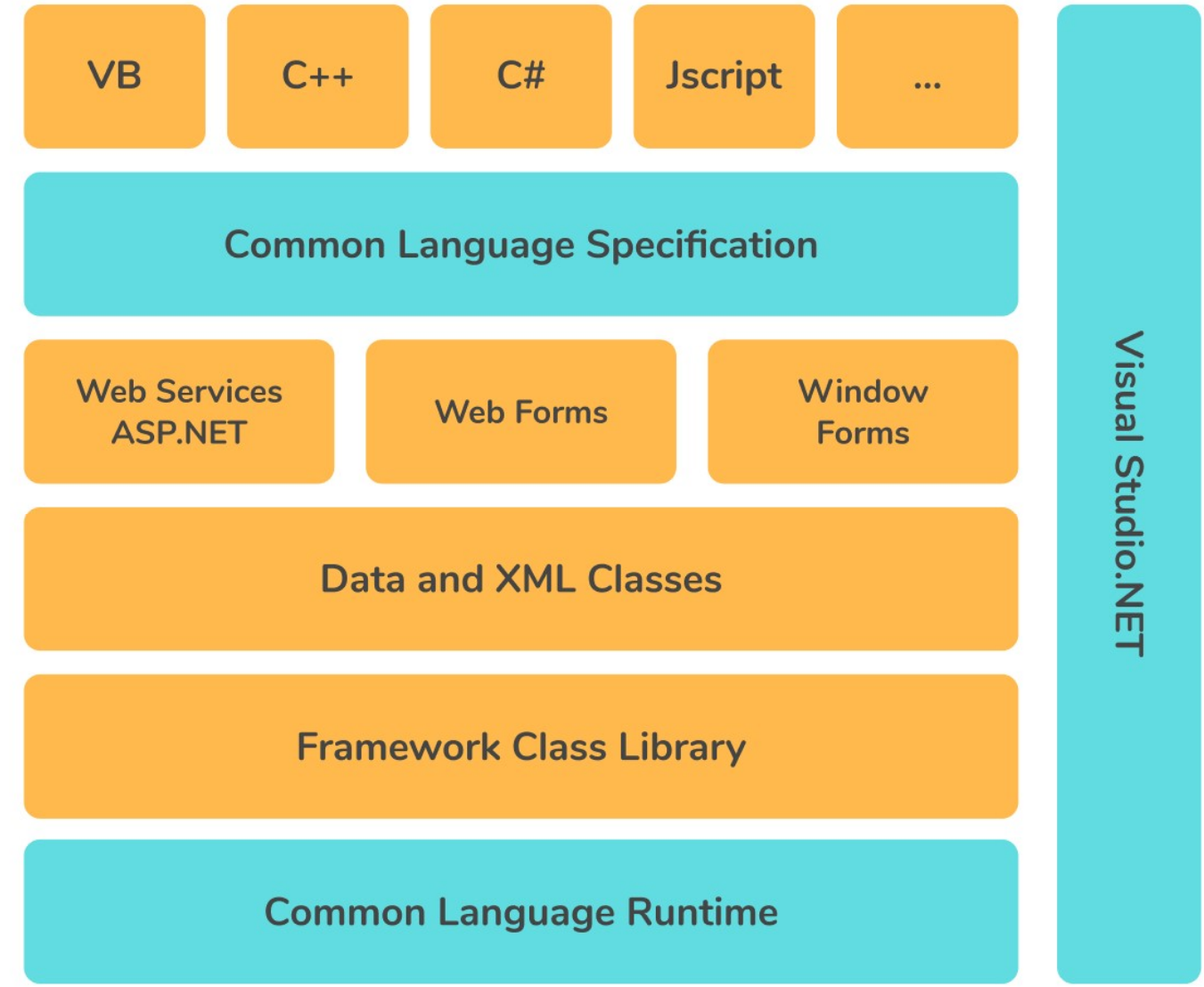
2. Explain about major components of the .NET framework.

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.



InterviewBit

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.

Managed Code	Unmanaged Code
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.

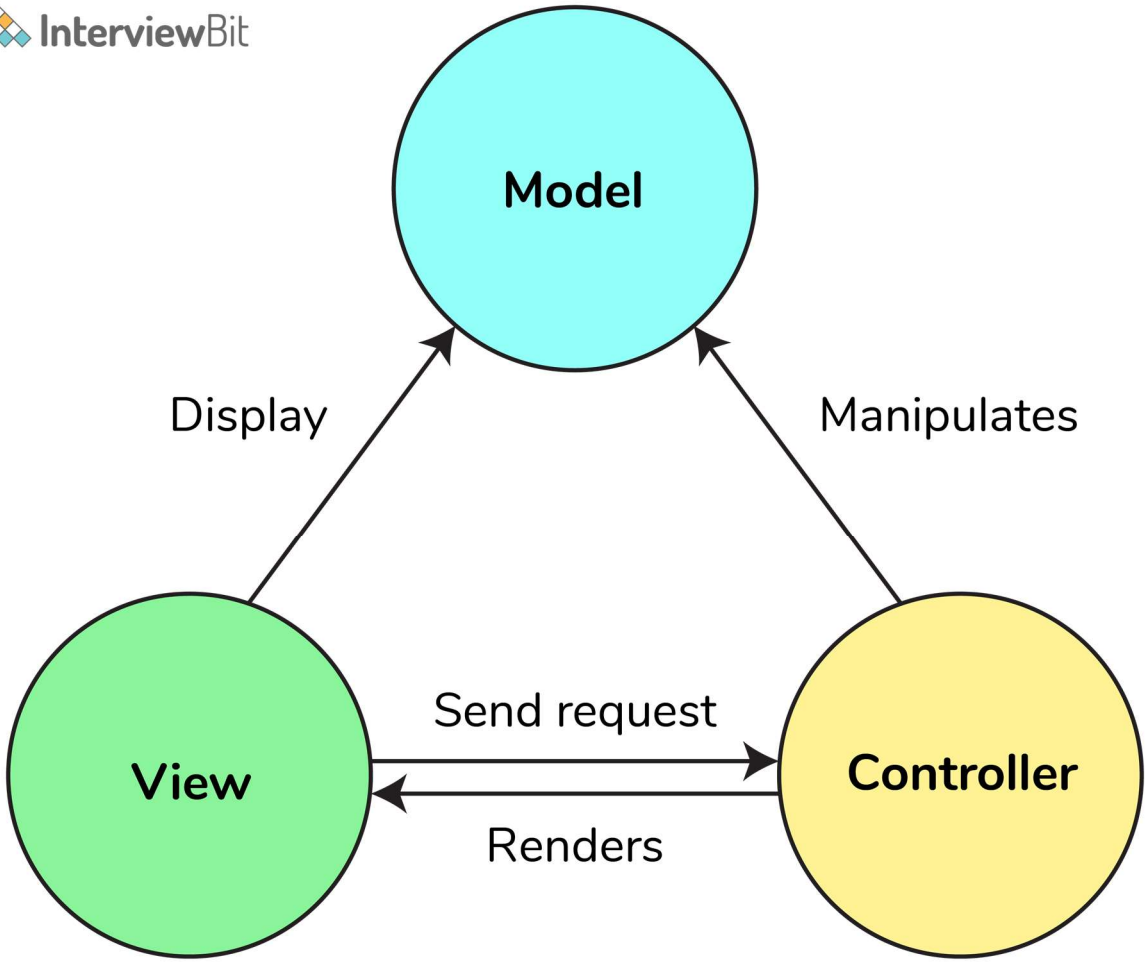


Fig: Components of MVC

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;
database=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.
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

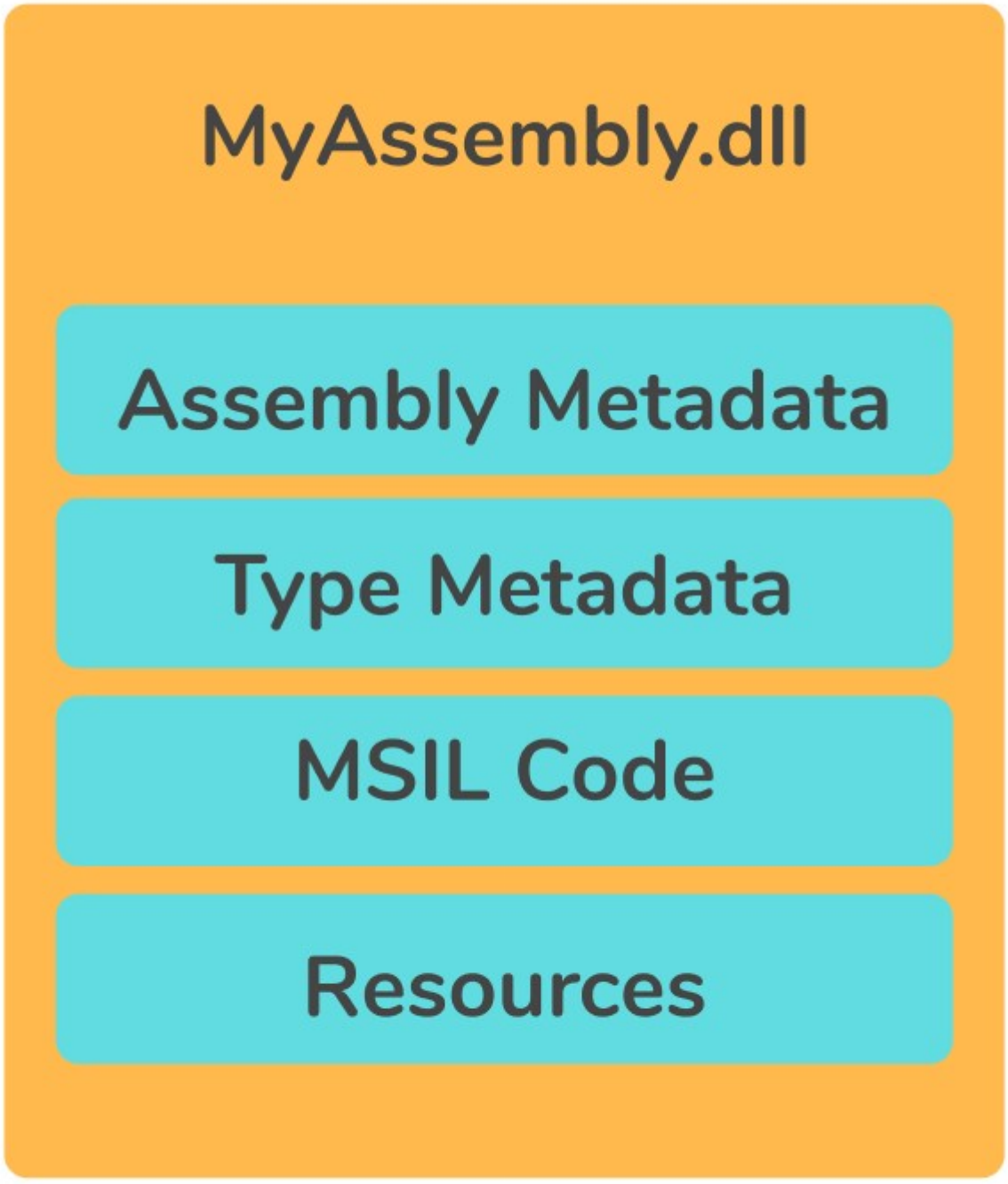
33. What are MDI and SDI?

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.

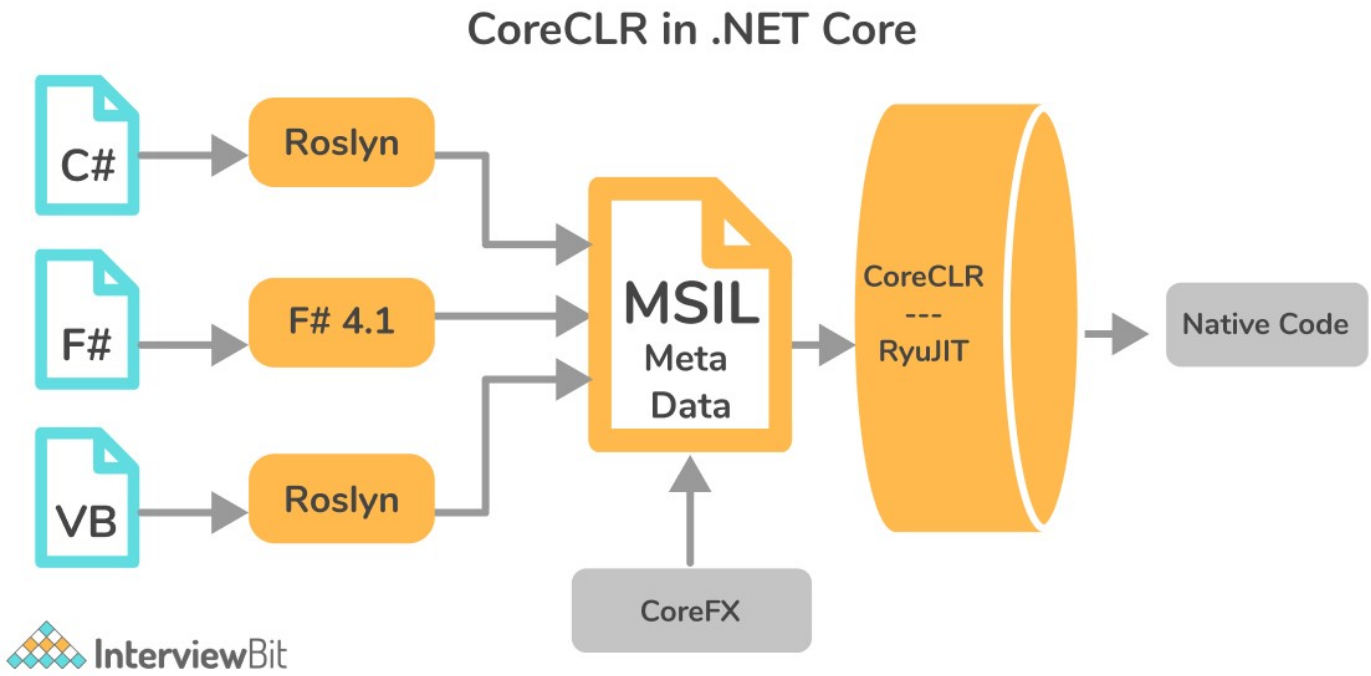
37. What are C# and F#?

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.

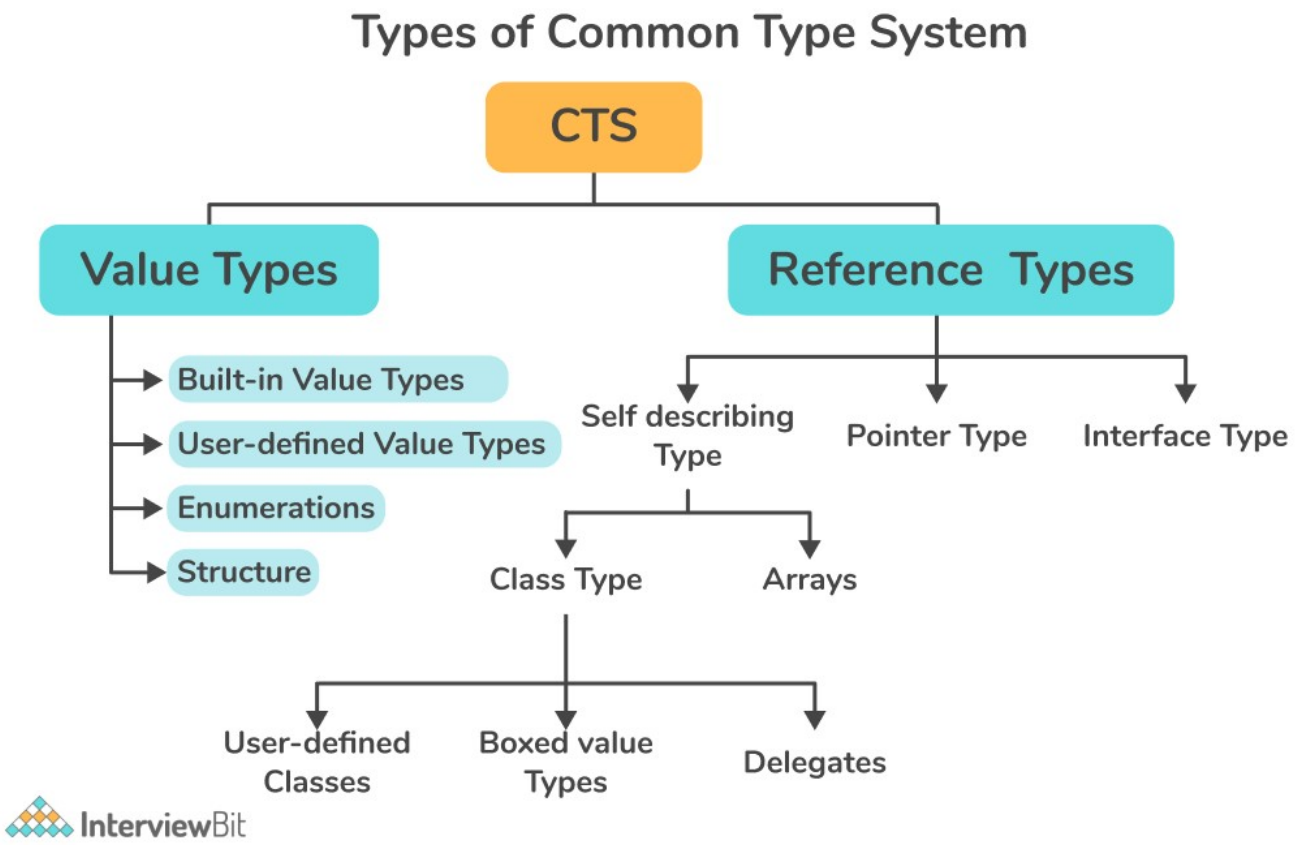
43. What is the use of generating SQL scripts in the .NET core?

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?

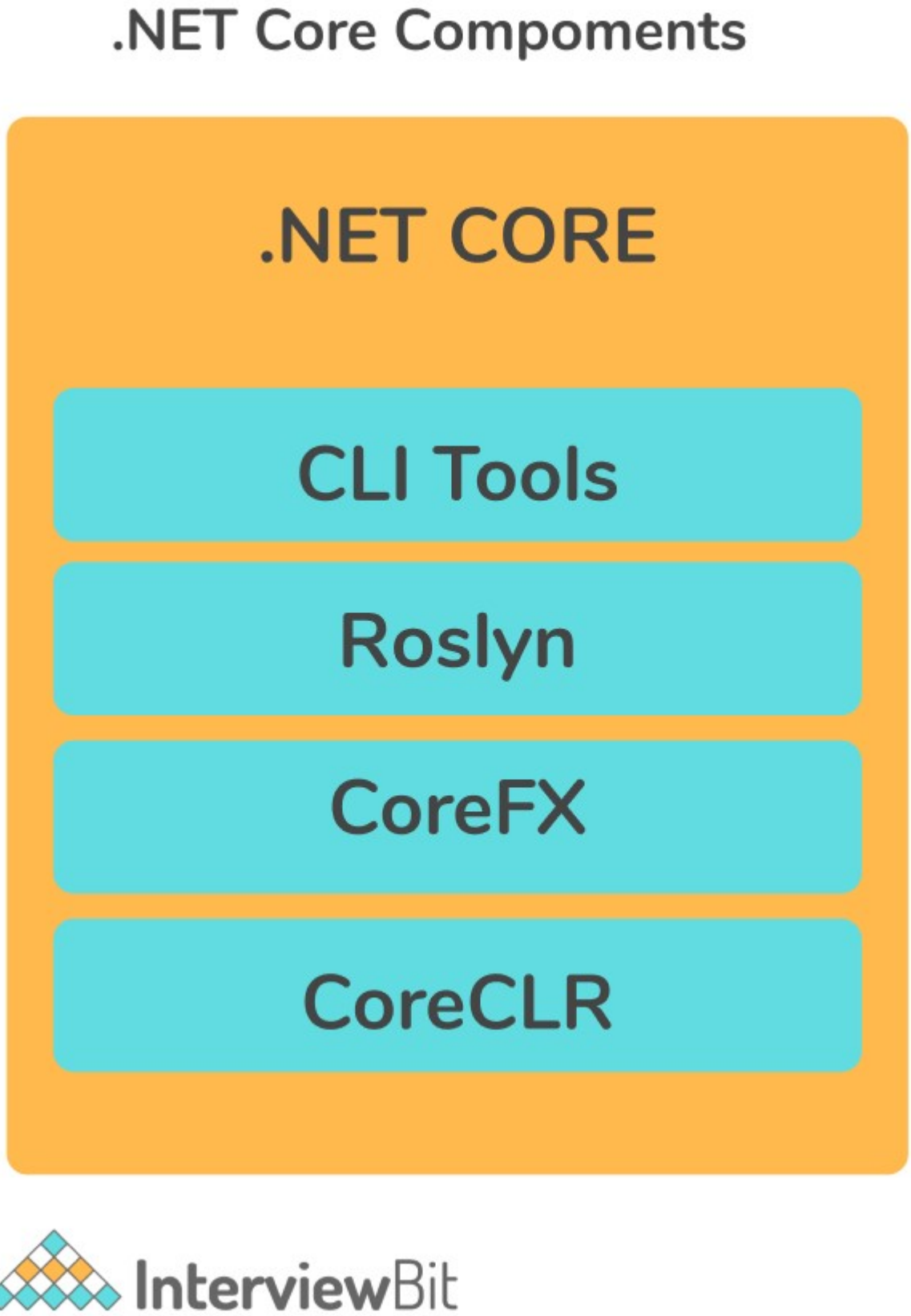
Universal Windows Platform(UWP) is one of the methods used to create client applications for Windows. UWP apps will make use of WinRT APIs for providing powerful UI as well as features of advanced asynchronous that are ideal for devices with internet connections.

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:

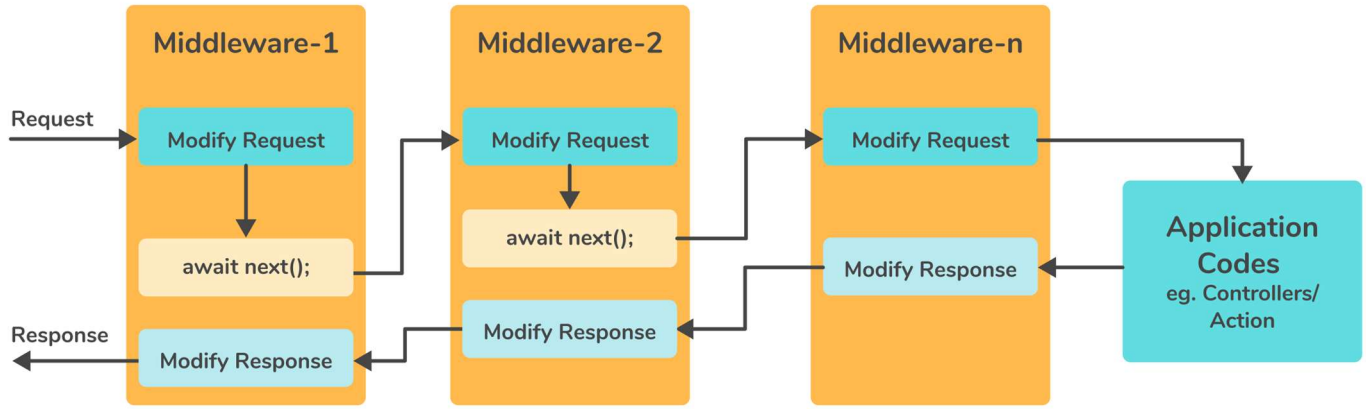


- **CLI Tools:** Command Line Interface(CLI) tools is a cross-platform tool for developing, building, executing, restoring packages, and publishing. It is also capable of building Console applications and class libraries that can run on the entire .NET framework. It is installed along with .NET Core 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

InterviewBit

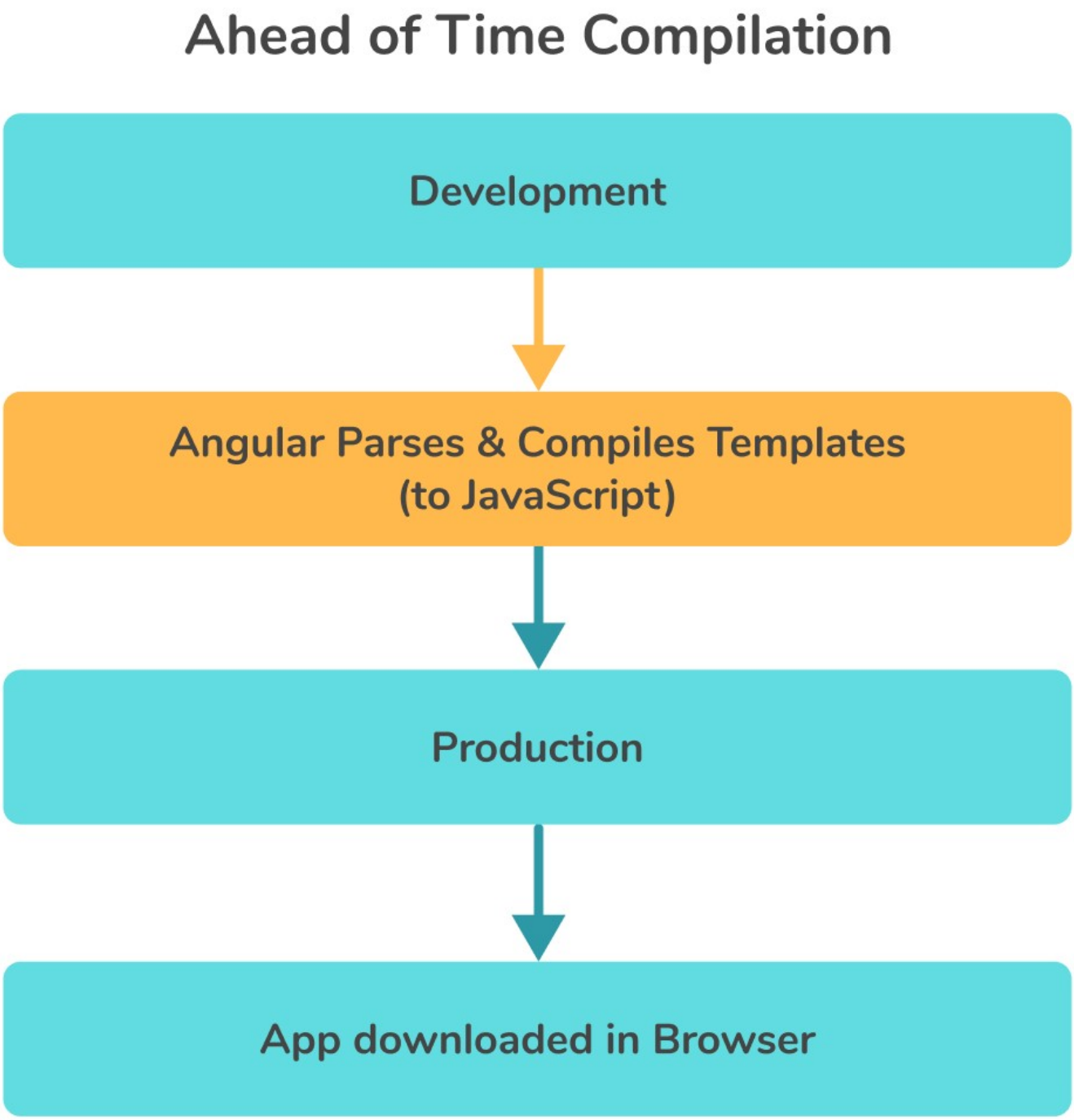
Here, there are three middlewares are associated with an ASP.NET Core web application. They can be either middleware provided by the framework, added through NuGet, or your own custom middleware. The HTTP request will be added or modified by each middleware and control will be optionally passed to the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



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="txtFirstNum" /> <br /><br />
    <span>Enter Second Number : </span> <input id="Text2" type="text"
name="txtSecondNum" /> <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.

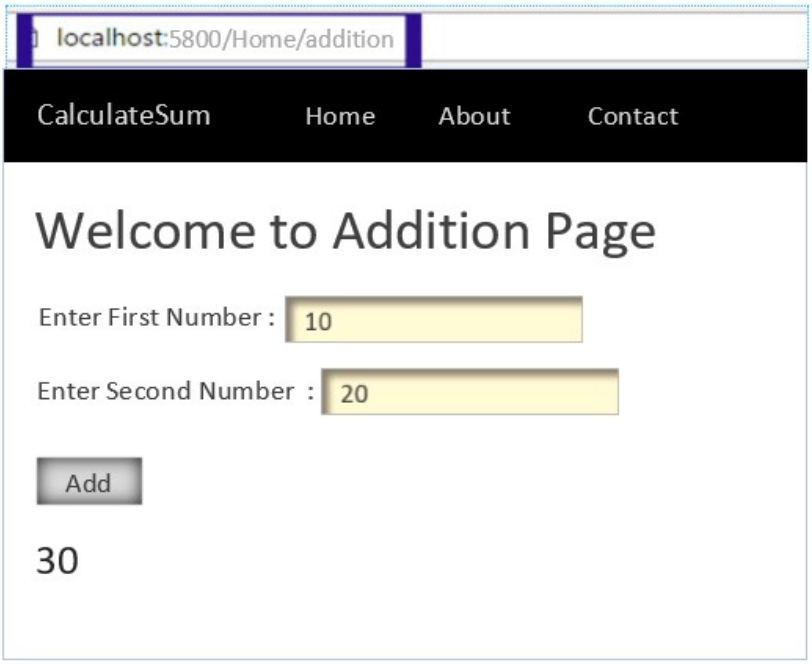
5. Open the `HomeController.cs` and write the following code onto it:

```
 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["txtFirstNum"].ToString());
```

```
            int number2 =
Convert.ToInt32(HttpContext.Request.Form["txtSecondNum"].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:



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.

ASP.NET is a web application framework developed by Microsoft which was released as part of the .NET framework. It makes it easy to build dynamic web applications. Some other examples of similar web application frameworks include Ruby on Rails (Ruby), Django (Python), and Express (JavaScript).

ASP.NET Core is a **cross-platform**, **high-performance**, and **open-source** web application framework. Microsoft released the first version of ASP.NET Core in 2016. It enables developers to create modern, cloud-enabled applications.

- **Cross-Platform:** The main advantage of ASP.NET Core is that it's not tied to a Windows operating system, like the legacy ASP.NET framework. You can develop and run production-ready ASP.NET Core apps on Linux or a Mac.
- **High Performance:** It's designed from scratch, keeping performance in mind. It's now one of the fastest web application frameworks.
- **Open Source:** Finally, it's open-source and actively contributed by thousands of developers all over the world.

Both the ASP.NET and ASP.NET Core run on C#, an object-oriented, general-purpose programming language. ASP.NET Core inherits many concepts and features from its ASP.NET heritage, but it's fundamentally a new framework.

Though Microsoft is going to support the legacy ASP.NET framework, it's not going to develop it actively. The new ASP.NET Core framework will include all the new features and enhancements. Going forward, Microsoft is recommending developers to build all the new web applications with ASP.NET Core instead of the legacy ASP.NET framework.

In this article, we will focus on both ASP.NET and ASP.NET Core interview questions. To limit the article's scope, we assume that you have programmed in the C# programming language. A basic understanding of common object-oriented concepts and front-end technologies such as HTML, CSS, and JavaScript is also expected.

We have divided the interview questions into two sections. The basic interview questions cover the fundamentals and focus on understanding the application structure of a basic ASP.NET project. Then, we cover the more advanced concepts such as dependency injection, routing, and model binding in the advanced interview questions.

Basic ASP.NET Interview Questions

1. What is a web application?

A Web application is a software that the users can access through a web browser such as Chrome or Firefox. The browser makes an HTTP request for a specific URL for the web application. The web application server intercepts and processes the request to build a dynamic HTML response sent to the user. Some examples of popular web applications include StackOverflow, Reddit, Google, etc.

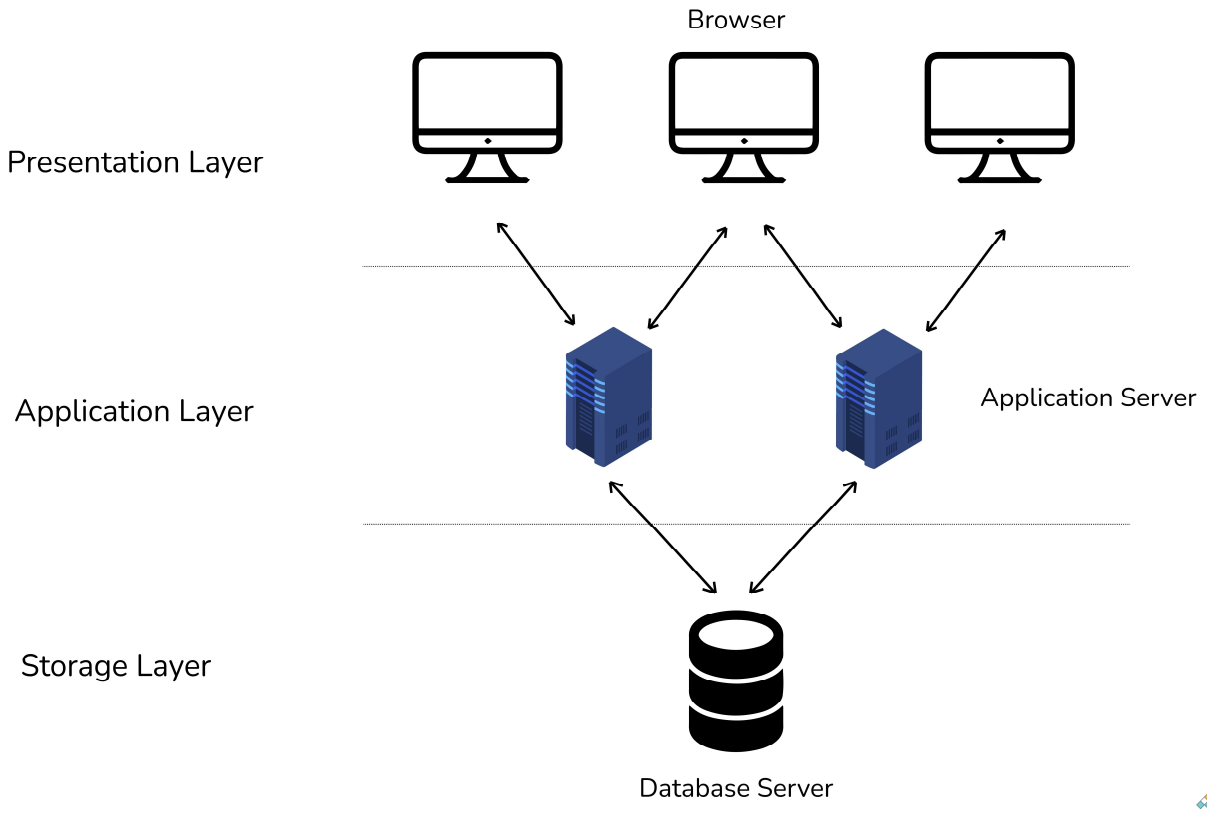
A web application is different from a typical website. A website is static. When you go to the website, it returns an HTML page without doing any processing to build the contents of that

HTML page. You will see the same page if you reload the browser. In contrast, a web application might return a different response each time you visit.

For example, let's say you ask a question on Stack Overflow. Initially, you will see only your question when you visit the URL. However, if another user answers your question, the browser will display that answer on your next visit to the same URL.

A web application consists of multiple separate layers. The typical example is a three-layered architecture made up of presentation, business, and data layers. For example, the browser (presentation) talks to the application server, which communicates to the database server to fetch the requested data.

The following figure illustrates a typical Web application architecture with standard components grouped by different areas of concern.



2. What is a web application framework, and what are its benefits?

Learning to build a modern web application can be daunting. Most of the web applications have a standard set of functionality such as:

- Build a dynamic response that corresponds to an HTTP request.
- Allow users to log into the application and manage their data.
- Store the data in the database.
- Handle database connections and transactions.
- Route URLs to appropriate methods.
- Supporting sessions, cookies, and user authorization.
- Format output (e.g. HTML, JSON, XML), and improve security.

Frameworks help developers to write, maintain and scale applications. They provide tools and libraries that simplify the above recurring tasks, eliminating a lot of unnecessary complexity.

3. What are some benefits of ASP.NET Core over the classic ASP.NET?

- **Cross-Platform:** The main advantage of ASP.NET Core is that it's not tied to a Windows operating system, like the legacy ASP.NET framework. You can develop and run production-ready ASP.NET Core apps on Linux or a Mac. Choosing an open-source operating system like Linux results in significant cost-savings as you don't have to pay for Windows licenses.
- **High Performance:** It's also designed from scratch, keeping performance in mind. It's now one of the fastest web application frameworks.
- **Open Source:** Finally, it's open-source and actively contributed by thousands of developers all over the world. All the source code is hosted on GitHub for anyone to see, change and contribute back. It has resulted in significant goodwill and trust for Microsoft, notwithstanding the patches and bug-fixes and improvements added to the framework by contributors worldwide.
- **New Technologies:** With ASP.NET Core, you can develop applications using new technologies such as Razor Pages and Blazor, in addition to the traditional Model-View-Controller approach.

4. When do you choose classic ASP.NET over ASP.NET Core?

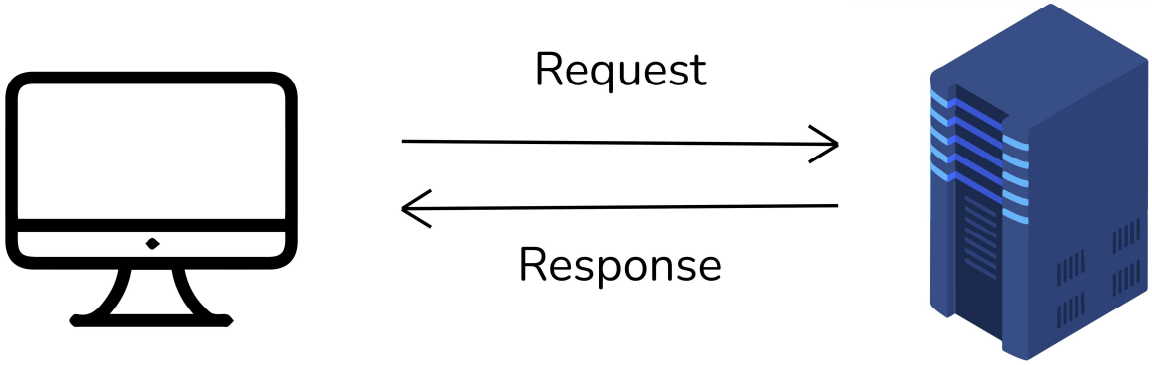
Though it's a better choice in almost all the aspects, you don't have to switch to ASP.NET Core if you are maintaining a legacy ASP.NET application that you are happy with and that is no longer actively developed.
ASP.NET MVC is a better choice if you:

- Don't need cross-platform support for your Web app.
- Need a stable environment to work in.
- Have nearer release schedules.
- Are already working on an existing app and extending its functionality.
- Already have an existing team with ASP.NET expertise.

5. Explain how HTTP protocol works?

Hypertext Transfer Protocol (HTTP) is an application-layer protocol for transmitting hypermedia documents, such as HTML. It handles communication between web browsers and web servers. HTTP follows a classical client-server model. A client, such as a web browser, opens a connection to make a request, then waits until it receives a response from the server.

HTTP is a protocol that allows the fetching of resources, such as HTML documents. It is the foundation of any data exchange on the Web, and it is a client-server protocol, which means requests are initiated by the recipient, usually the Web browser.

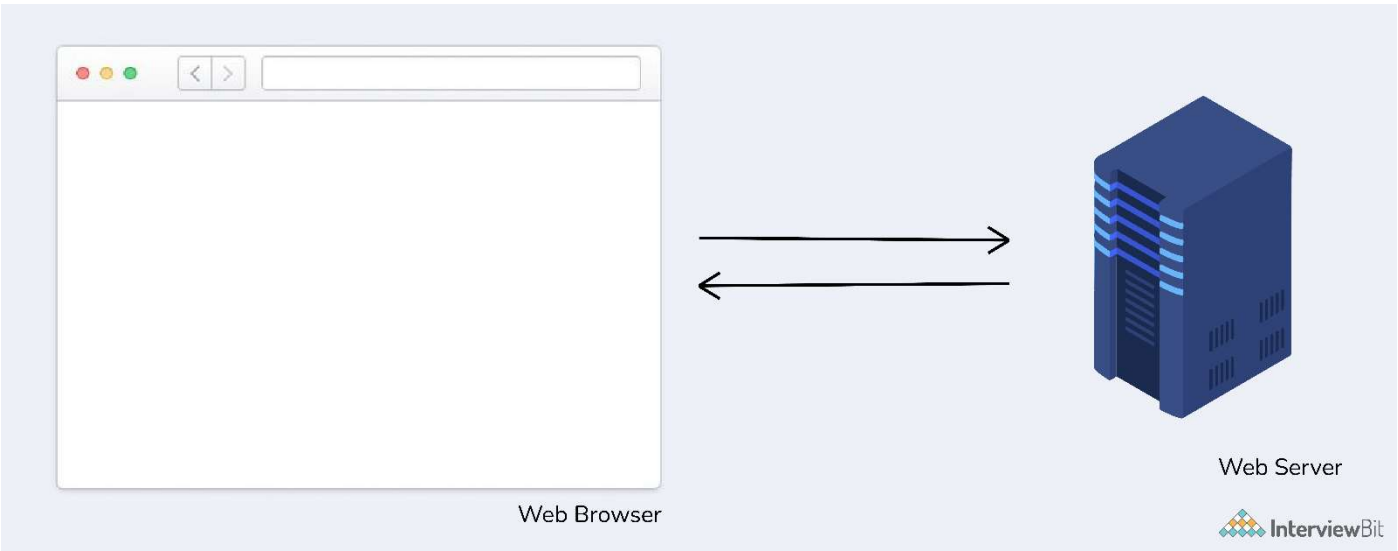


InterviewBit

6. What is a web server?

The term web server can refer to both hardware or software, working separately or together.

On the hardware side, a web server is a computer with more processing power and memory that stores the application's back-end code and static assets such as images and JavaScript, CSS, HTML files. This computer is connected to the internet and allows data flow between connected devices.

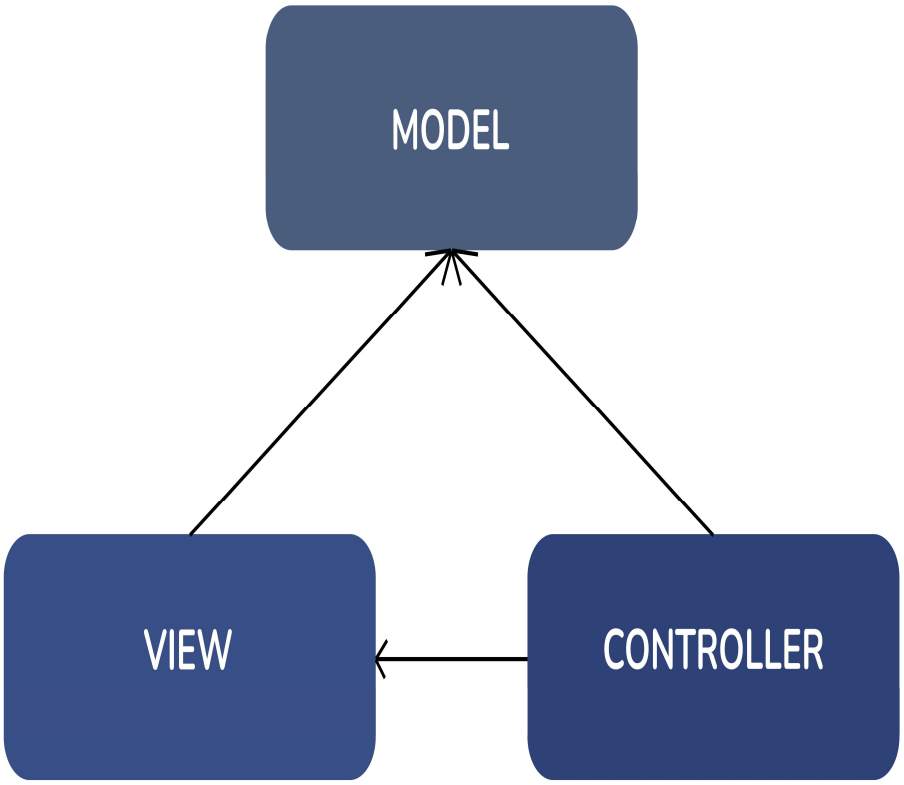


On the software side, a web server is a program that accepts HTTP requests from the clients, such as a web browser, processes the request, and returns a response. The response can be static, i.e. image/text, or dynamic, i.e. calculated total of the shopping cart.

Popular examples of web servers include Apache, Nginx, IIS.

7. What is the MVC pattern?

The **Model-View-Controller** (MVC) is an architectural pattern that separates an application into three main logical components: the **model**, the **view**, and the **controller**. It's different to note that this pattern is unrelated to the layered pattern we saw earlier. MVC pattern operates on the software side, while the layered pattern dictates how and where we place our database and application servers.



In an application that follows the MVC pattern, each component has its role well specified. For example, model classes only hold the data and the business logic. They don't deal with HTTP requests. Views only display information. The controllers handle and respond to user input and decide which model to pass to which view. This is known as the separation of responsibility. It makes an application easy to develop and maintain over time as it grows in complexity.

Though Model-View-Controller is one of the oldest and most prominent patterns, alternate patterns have emerged over the years. Some popular patterns include MVVM (Model-View-ViewModel), MVP (Model-View-Presenter), MVA (Model-View-Adapter).

8. Explain the role of the various components of the MVC pattern?

Model: Represents all the data and business logic that the user works within a web application. In ASP.NET, the model is represented by C# classes that hold the data and the related logic that operates on that data. The 'Models' directory stores the model classes.

For example, a model class representing a blog post might look like this:

```
// Models/Post.cs
namespace app.Models
{
  public class Post
  {
      public int ID { get; set; }

      public string Title { get; set; }

      public string Body { get; set; }
  }
}
```

View: Represents all the UI logic of the application. In a web application, it represents the HTML that's sent to the user and displayed in the browser.

One important thing to remember is that this HTML is not static or hard-coded. It's generated dynamically by the controller using a model's data. In ASP.NET, the 'Views' directory contains the views in files ending with the .cshtml file extension.

To continue our example of a blog post, a view to render a post might be:

```
// Views/Post.cshtml

<div class="post">
  <div class="title">
      <a href="/posts/@post.ID">@post.Title</a>
  </div>

  <div class=body>
      @Html.Raw(post.Body)
  </div>
</div>
```

Controller: Acts as an interface between Model and View. It processes the business logic and incoming requests, manipulates data using the Model, and interacts with the Views to render the final output.

In ASP.NET, these are C# classes that form the glue between a model and a view. They handle the HTTP request from the browser, then retrieve the model data and pass it to the view to dynamically render a response. The 'Controllers' directory stores the controller classes.

A PostController that builds the view for the post by fetching the Post model will be:

```
// Controllers/PostController
namespace app.Controllers
{
  public class PostsController : BaseController
  {
      public IActionResult Post(int id)
      {
          // Get the post from the database
          Post post = _service.Get(id);

          // Render the post.cshtml view, by providing the post model
          return View(post);
```

```
        }
    }
}
```

9. What is the purpose of the .csproj file?

The project file is one of the most important files in our application. It tells .NET how to build the project.

All .NET projects list their dependencies in the .csproj file. If you have worked with JavaScript before, think of it like a package.json file. The difference is, instead of a JSON, this is an XML file.

When you run dotnet restore, it uses the .csproj file to figure out which NuGet packages to download and copy to the project folder (check out the next question to learn more about Nuget).

The .csproj file also contains all the information that .NET tooling needs to build the project. It includes the type of project you are building (console, web, desktop, etc.), the platform this project targets, and any dependencies on other projects or 3rd party libraries.

Here is an example of a .csproj file that lists the Nuget packages and their specific versions.

```
<Project Sdk="Microsoft.NET.Sdk.Web">
<PropertyGroup>
  <TargetFramework>net5.0</TargetFramework>
</PropertyGroup>

<ItemGroup>
  <PackageReference Include="AWSSDK.S3" Version="3.5.6.5" />
  <PackageReference Include="Microsoft.EntityFrameworkCore.Sqlite" Version="5.0.1" />
  <PackageReference Include="Microsoft.Extensions.Caching.Memory" Version="5.0.0" />
  <PackageReference Include="Npgsql" Version="5.0.1.1" />
  <PackageReference Include="Serilog" Version="2.10.0" />
</ItemGroup>
</Project>
```

In the above example,

- The SDK attribute specifies the type of .NET project.
- TargetFramework is the framework this application will run on, .NET 5 in this case.
- The PackageReference element includes the NuGet packages. The Version attribute specifies a version of the package we want.

10. What is NuGet package manager?

Software developers don't write all code from scratch. They rely on libraries of code written by other developers. Any modern development platform must provide a mechanism where developers can download and use existing libraries, often called packages. For example, the JavaScript ecosystem has NPM (Node Package Manager), where developers can find and use libraries written by other JavaScript developers.

NuGet is a package manager for the .NET ecosystem. Microsoft developed it to provide access to thousands of packages written by .NET developers. You can also use it to share the code you wrote with others.

A typical web application developed using ASP.NET relies on many open source NuGet packages to function. For example, Newtonsoft.Json is a very popular package (with 91,528,205 downloads at the time of writing) used to work with JSON data in .NET.

11. What is the purpose of the Program class?

Program.cs class is the entry point of our application. An ASP.NET application starts in the same way as a console application, from a **static void Main()** function.

This class configures the web host that will serve the requests. The host is responsible for application startup and lifetime management, including graceful shutdown.

At a minimum, the host configures a server and a request processing pipeline. The host can also set up logging, configuration, and dependency injection.

12. What is the purpose of the Startup class?

This class handles two important aspects of your application, namely service registration, and middleware pipeline.

Services are C# classes that your application depends on for providing additional functionality, both used by the framework and your application. Examples include logging, database, etc. These services must be registered to be instantiated when your app is running and when it needs them.

The middleware pipeline is the sequence in which your application processes an HTTP request (the next question explains the concept of Middleware in detail).

Startup class contains two methods: ***ConfigureServices()*** and ***Configure()***. As the name suggests, the first method registers all the services that the application needs. The second method configures the middleware pipeline.

13. What is the purpose of the wwwroot folder?

The wwwroot folder contains static files and compiled assets, such as JavaScript, CSS, and images that your web application needs. Wwwroot is the only folder in the entire project that's exposed as-is to the browser.

14. What is the purpose of the appsettings.json file?

Appsettings.json contains all of the application's settings, which allow you to configure your application behavior.

Here is an example of an appsettings.json file.

```
{
 "Logging": {
     "LogLevel": {
          "Default": "Information",
    "Microsoft": "Warning",
    "Microsoft.Hosting.Lifetime": "Information"
      }
  },
"ConnectionStrings": {
    "AppConnection": ""
},
"AWS": {
  "Profile": "local-test-profile",
  "Region": "us-west-2"
},
"AllowedHosts": "*"
}
```

15. What is IIS?

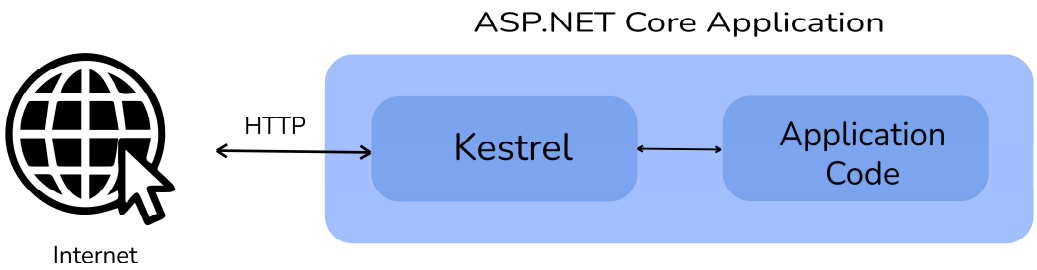
IIS stands for Internet Information Services. It is a powerful web server developed by Microsoft. IIS can also act as a load balancer to distribute incoming HTTP requests to different application servers to allow high reliability and scalability.

It can also act as a reverse proxy, i.e. accept a client's request, forward it to an application server, and return the client's response. A reverse proxy improves the security, reliability, and performance of your application.

A limitation of IIS is that it only runs on Windows. However, it is very configurable. You can configure it to suit your application's specific needs.

16. What is Kestrel?

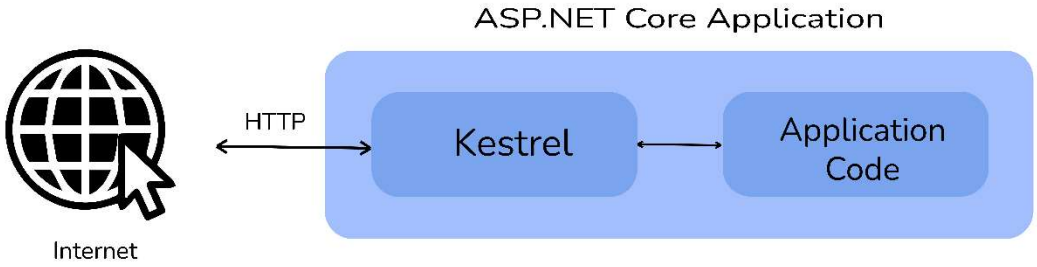
Kestrel is an open-source, cross-platform web server designed for ASP.NET Core. Kestrel is included and enabled by default in ASP.NET Core. It is very light-weight when compared with IIS.



InterviewBit

Kestrel can be used as a web server processing requests directly from a network, including the Internet.

Though Kestrel can serve an ASP.NET Core application on its own, Microsoft recommends using it along with a reverse proxy such as IIS, Nginx, or Apache, for better performance, security, and reliability.



InterviewBit

17. What is the difference between IIS and Kestrel? Why do we need two web servers?

The main difference between IIS and Kestrel is that Kestrel is a cross-platform server. It runs on Windows, Linux, and Mac, whereas IIS only runs on Windows.

Another essential difference between the two is that Kestrel is fully open-source, whereas IIS is closed-source and developed and maintained only by Microsoft.

IIS is very old software and comes with a considerable legacy and bloat. With Kestrel, Microsoft started with high-performance in mind. They developed it from scratch, which allowed them to ignore the legacy/compatibility issues and focus on speed and efficiency.

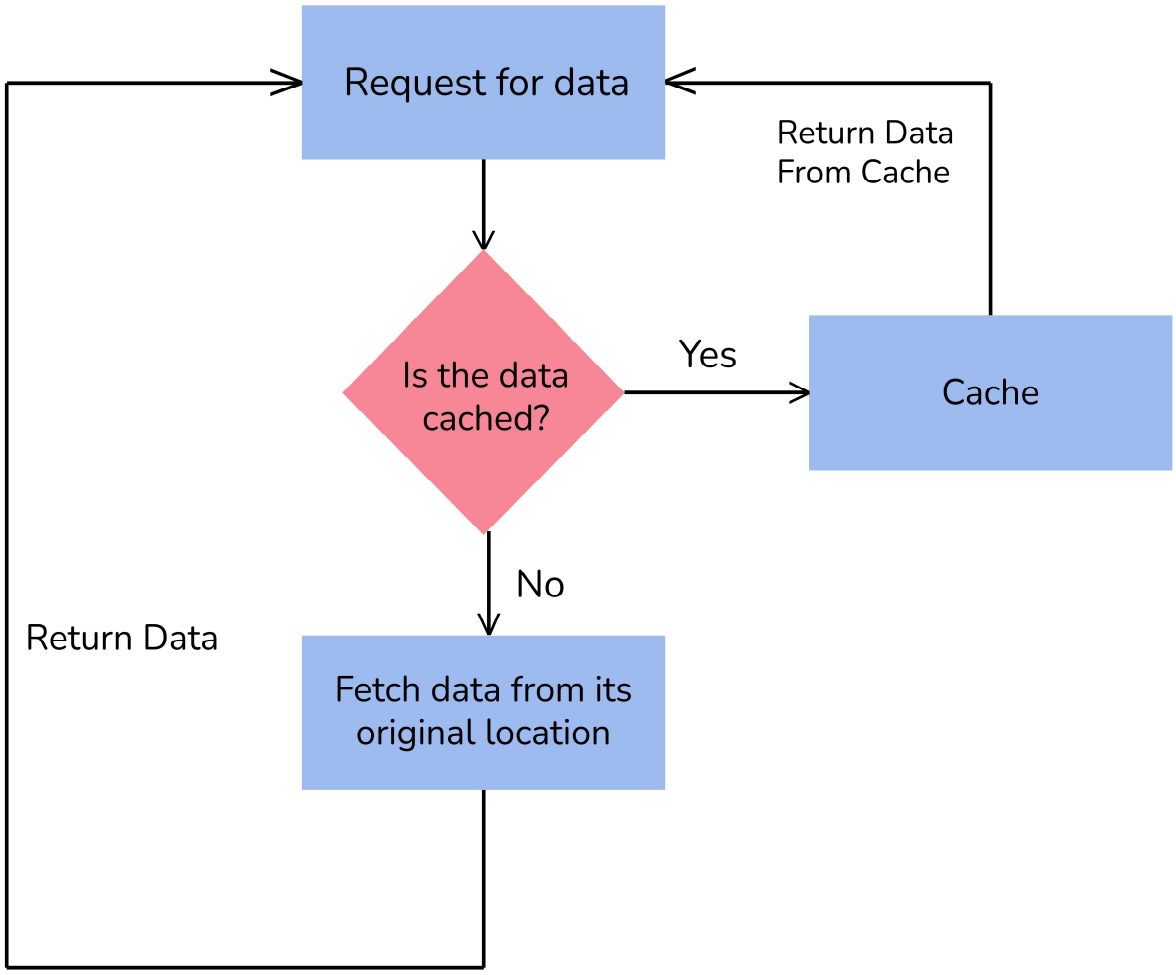
However, Kestrel doesn't provide all the rich functionality of a full-fledged web server such as IIS, Nginx, or Apache. Hence, we typically use it as an application server, with one of the above servers acting as a reverse proxy.

18. What is caching?

Caching is the process of storing data in a temporary storage location that is quicker to access than the original location of the data so that it can be accessed more quickly when the same data is needed next time.

Caching improves the scalability and performance of your application. It does this by reducing the work required to fetch the data. Caching is useful for data that doesn't change frequently and is expensive to create and retrieve.

ASP.NET provides a set of caching features out of the box. You can use the IMemoryCache interface for simple use cases. It represents a cache stored in the web server's memory. ASP.NET also supports distributed caching, which is a cache shared by multiple app servers, with Redis.



InterviewBit

Advanced ASP.NET Interview Questions

19. What is model binding in ASP.NET?

Controllers and views need to work with data that comes from HTTP requests. For example, routes may provide a key that identifies a record, and posted form fields may provide model properties. The process of converting these string values to .NET objects could be complicated and something that you have to do with each request. Model binding automates and simplifies this process.

The model binding system fetches the data from multiple sources such as form fields, route data, and query strings. It also provides the data to controllers and views in method parameters and properties, converting plain string data to .NET objects and types in the process.

Example:
Let's say you have the following action method on the PostsController class:

```
[HttpGet("posts/{id}")]
public ActionResult<Post> GetById(int id, bool archivedOnly)
```

And the app receives a request with this URL:

```
http://yourapp.com/api/Posts/5?ArchivedOnly=true
```

After the routing selects the action method, model binding executes the following steps.

- Locate the first parameter of GetByID, an integer named id, look through the available sources in the HTTP request and find id = "5" in route data.
- Convert the string "5" into an integer 5.
- Find the next parameter of GetByID, a boolean named archivedOnly.
- Look through the sources and find "ArchivedOnly=true" in the query string. It ignores the case when matching the parameters to the strings.
- Convert the string "true" into boolean true.

Some other examples of attributes include:
1. [FromQuery] - Gets values from the query string.
2. [FromRoute] - Gets values from route data.
3. [FromForm] - Gets values from posted form fields.
4. [FromBody] - Gets values from the request body.
5. [FromHeader] - Gets values from HTTP headers.

20. What is an Action Method?

An action method is a method in a controller class with the following restrictions:

1. It must be public. Private or protected methods are not allowed.
2. It cannot be overloaded.
3. It cannot be a static method.

An action method executes an action in response to an HTTP request.

For example, here is an example of an Index() action method on the PostController. It takes an id as an input and returns an IActionResult, which can be implemented by any result classes (see the following question).

```
public class PostController : Controller
{
  public IActionResult Index(int id)
  {

  }
}
```

21. What are the different types that implement the IActionResult interface?

ASP.NET Core has many different types of IActionResult:

- ViewResult—Generates an HTML view.
- RedirectResult—Sends a 302 HTTP redirect response to send a user to a specified URL automatically.
- RedirectToRouteResult—Sends a 302 HTTP redirect response to automatically send a user to another page, where the URL is defined using routing.

- FileResult—Returns a file as the response.
- ContentResult—Returns a provided string as the response.
- StatusCodeResult—Sends a raw HTTP status code as the response, optionally with associated response body content.
- NotFoundResult—Sends a raw 404 HTTP status code as the response.

22. What's the HTTPContext object? How can you access it within a Controller?

HTTPContext encapsulates all HTTP-specific information about an individual HTTP request. You can access this object in controllers by using the **ControllerBase.HttpContext** property:

```
public class HomeController : Controller
{
  public IActionResult About()
  {
      var pathBase = HttpContext.Request.PathBase;

      ...

      return View();
  }
}
```

23. What is dependency injection?

Dependency injection is a design pattern that helps to develop loosely coupled code. This pattern is used extensively in ASP.NET.

Dependency injection means **providing** the objects that an object needs (its dependencies) in that object's constructor instead of requiring the object to construct them.

Dependency injection reduces and often eliminates unnecessary dependencies between objects that don't need to know each other. It also helps in testing by mocking or stubbing out the dependencies at runtime.

24. Explain how dependency injection works in ASP.NET Core?

ASP.NET Core injects instances of dependency classes by using the built-in IoC (Inversion-of-Control) container. This container is represented by the IServiceProvider interface that supports constructor injection.

The types (classes) managed by the container are called services. To let the IoC container automatically inject our services, we first need to register them with the IoC container in the Startup class.

ASP.NET Core supports two types of services, namely framework and application services. **Framework** services are a part of ASP.NET Core framework such as ILoggerFactory, IApplicationBuilder, IHostingEnvironment, etc. In contrast, a developer creates the **application services** (custom types or classes) specifically for the application.

25. What is a cookie?

A cookie is a small amount of data that is persisted across requests and even sessions. Cookies store information about the user. The browser stores the cookies on the user's computer. Most browsers store the cookies as key-value pairs.

Write a cookie in ASP.NET Core:

```
Response.Cookies.Append(key, value);
```

Delete a cookie in ASP.NET Core

```
Response.Cookies.Delete(somekey);
```

26. Explain the concept of middleware in ASP.NET Core?

In general, middleware is plumbing software that facilitates communication flow between two components. In a web application framework, middleware provides common services and capabilities to the application outside of the default framework.

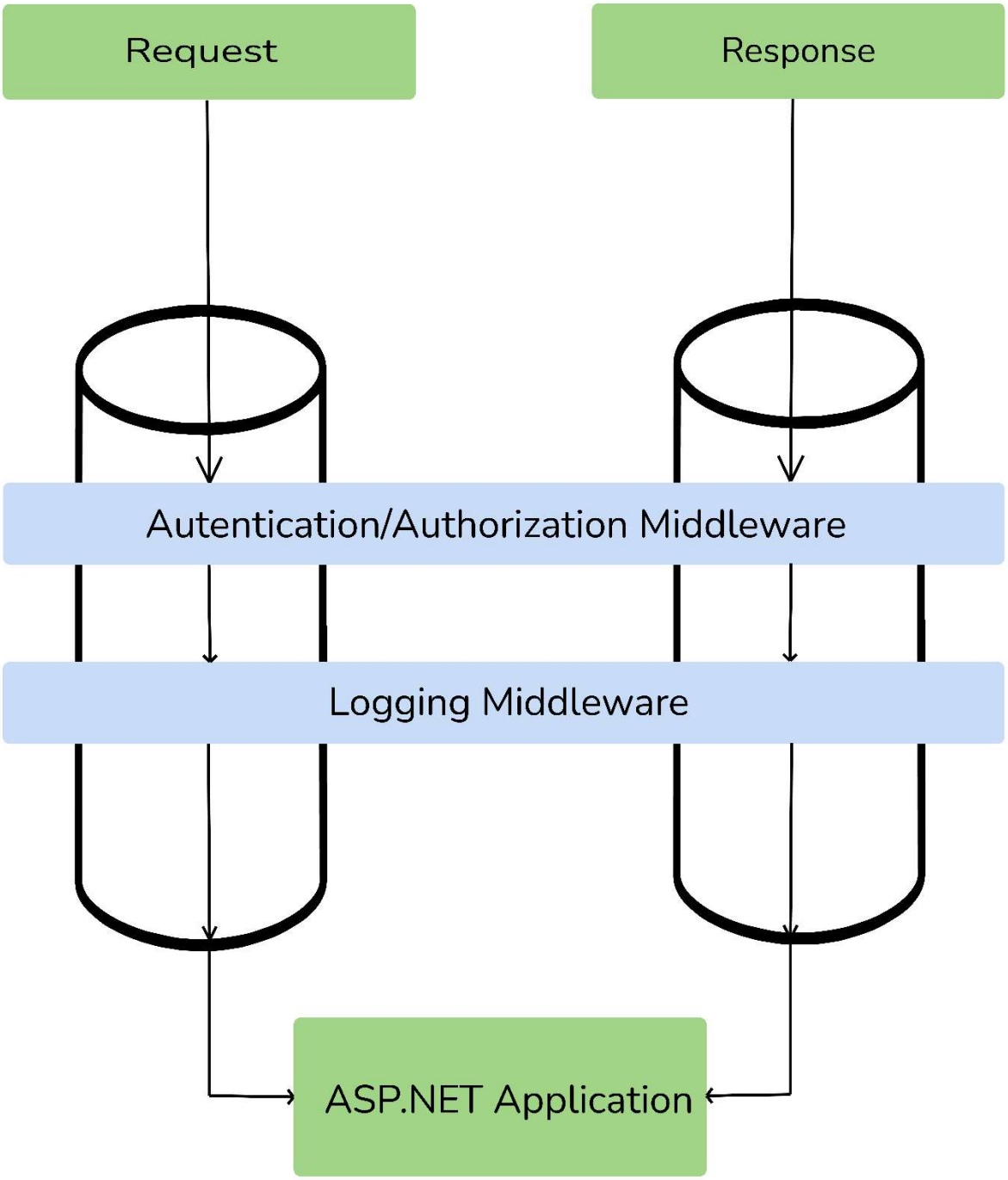
In ASP.NET Core, middleware refers to the C# classes that manipulate an HTTP request when it comes in or an HTTP response when it's on its way out. For example,

Generate an HTTP response for an incoming HTTP request
Intercept and make changes to an incoming HTTP request and pass it on to the next piece of middleware.
Intercept and make changes to an outgoing HTTP response, and pass it on to the next piece of middleware.

One of the most common use cases for middleware is to deal with concerns that affect your entire application. These aspects of your application need to occur with every request, regardless of the specific path in the request or the resource requested. These include things like logging, security, authentication, authorization, etc.

For example, logging middleware logs when a request comes in and passes it to another piece of middleware. Some other common middleware uses include database middleware, error handling middleware, and authentication/authorization middleware.

In each of these examples, the middleware receives a request, modifies it, and then passes it to the next middleware piece in the pipeline. Subsequent middleware uses the details added by the earlier middleware to handle the request in some way. The following diagram illustrates this.



InterviewBit

27. What is routing, and how can you define routes in ASP.NET Core?

Routing is the process of mapping an incoming HTTP request to a specific method in the application code. A router maps the incoming requests to the route handler. It takes in a URL as an input and deconstructs it to determine the controller and action method to route the request.

A simple routing pattern, for example, might determine that the **/posts/show** URL maps to the **Show** action method on the **PostsController**.

There are two ways to define routes in an ASP.NET Core MVC application.

- Conventional Routing
- Attribute-Based Routing.

We can use both Conventional Routing and Attribute Routing in an application.

28. Explain how conventional routing works?

As the name suggests, conventional routing uses predefined conventions to match the incoming HTTP request to a controller's action method. It handles the most general cases that a typical web application needs, but you can modify it to suit your specific needs.

For example, the Configure() method in the Startup.cs class contains the following code that sets up the conventional routing.

```
app.UseEndpoints(endpoints =>
{
  endpoints.MapControllerRoute(
      name: "default",
      pattern: "{controller=Home}/{action=Index}/{id?}");
});
```

This code creates a single route named 'default'. The route template pattern '{controller=Home}/{action=Index}/{id?}' is used to match an incoming URL such as **/Posts/Archived/5** to the **Archived(int id)** action method on the PostsController, passing 5 for the id parameter. By default, the router uses the Index method on the HomeController.

29. Explain how attribute-based routing works?

Attribute routing is an alternative routing strategy for conventional routing. It is often used to create REST API endpoints for web services. It uses a set of attributes to map action methods directly to route templates.

Attribute routing directly defines the routes on action methods. We can also use these attributes on the controllers. It enables us to get fine-grained control over what routes map to which actions. With attribute routing, the controller and action names play no part in determining the action method.

For example, we use attributes Blog and Home to map an incoming URL such as **myapp.com/blog/post/3** to the **Show** method on the **PostsController**.

```
[Route("blog")]
public class PostsController : Controller
{
  [HttpGet("post/{id:int}")]
  public IActionResult Show(int id = 0)
  {
      Post post = new Post()
      {
          ID = id
      };
```

```
        return View("Show", post);
    }

    [HttpGet("edit/{id:int}")]
    public IActionResult Edit(int id)
    {
        Post postToEdit = _service.Get(id);

        return View("Edit", postToEdit);
    }
}
```

In the above example, the attribute **[Route("blog")]** is placed on the controller, whereas the route **[HttpGet("post/{id:int}")]** is placed on the action method. A controller route applies to all actions in that controller. For example, the second **["edit/{id:int}"]** route matches the url **myapp.com/blog/edit/3**.

In addition to the above route templates, ASP.NET Core provides the following HTTP verb templates.

- [HttpGet]
- [HttpPost]
- [HttpPut]
- [HttpDelete]
- [HttpHead]
- [HttpPatch]

30. What is a RESTful Web Service or a Web API?

Not all web applications return an HTML view as a response to an HTTP request. Sometimes, a client only wants some data from your application, and it wants to handle how that data will be formatted.

For example, let's say your application supports both web and mobile interfaces. Instead of writing two separate projects which return HTML and mobile views, you can write a single application that only returns the specific data that the clients need. Once the clients receive this data, they format it accordingly. The web client renders the HTML using view templates and JavaScript, and the mobile clients generate the appropriate mobile view for its specific platform.

An application might also need to communicate with another application to fetch the data that it needs. For example, when you go to Amazon.com, it communicates with hundreds of other services and applications to retrieve data and renders the final HTML page you see.

Such back-end applications, which provide data, are commonly known as RESTful web services. REST protocol uses verbs like **GET, POST, PUT, DELETE** to communicate between multiple applications. The client and server can be written in different languages and technologies and still work together without knowing about each other, as long as each side knows the format of the data that is getting sent.

ASP.NET Core supports creating RESTful services, also known as web APIs, using C#. A Web API consists of one or more controllers that derive from ControllerBase class.

```
[PostController]
[Route("[controller]")]
public class PostController : ControllerBase
```

An MVC controller derives from the **Controller** class. However, a Web API controller should derive from the ControllerBase class. The reason is that Controller derives from ControllerBase and provides additional support for views, which you don't need for web API requests.

That said, you can use a controller for both rendering views and data. That is, it can act as both an MVC controller and a Web API controller. In this case, you can derive the controller from the Controller class.

31. What is Entity Framework?

Most applications require storing and retrieving data. Usually, we store this data in a database. Working with databases can often be rather complicated. You have to manage database connections, convert data from your application to a format the database can understand, and handle many other subtle issues.

The .NET ecosystem has libraries you can use for this, such as ADO.NET. However, it can still be complicated to manually build SQL queries and convert the data from the database into C# classes back and forth.

EF, which stands for Entity Framework, is a library that provides an object-oriented way to access a database. It acts as an object-relational mapper, communicates with the database, and maps database responses to .NET classes and objects.

Entity Framework (EF) Core is a lightweight, open-source, and cross-platform version of the Entity Framework.

Here are the essential differences between the two:

Cross-platform:

- We can use EF Core in cross-platform apps that target .NET Core.
- EF 6.x targets .NET Framework, so you're limited to Windows.

Performance:

- EF Core is fast and lightweight. It significantly outperforms EF 6.x.

Features:

- EF Core has some features that EF 6.x doesn't have (batching statements, client-side key generation, in-memory database for testing)
- EF 6.x is much more feature-rich than EF Core. EF Core is missing some headline features at the time of writing, such as lazy-loading and full server-side Group By. However, it is under active development, so those features will no doubt appear soon.

32. Conclusion

In this article on ASP.NET interview questions, we learned about the legacy ASP.NET framework and its modern alternative, that is ASP.NET Core. The article explored a broad range of basic and advanced questions that an interviewer would ask in a job interview for a junior/intermediate developer role.

MVC (full form **Model View Controller**) is a software architecture or application design model containing 3 interconnected verticals or portions. These 3 portions are the **model** (data associated with the application), the **view** (which is the user interface of an MVC application), and the **controller** (the processes that are responsible for handling the input).

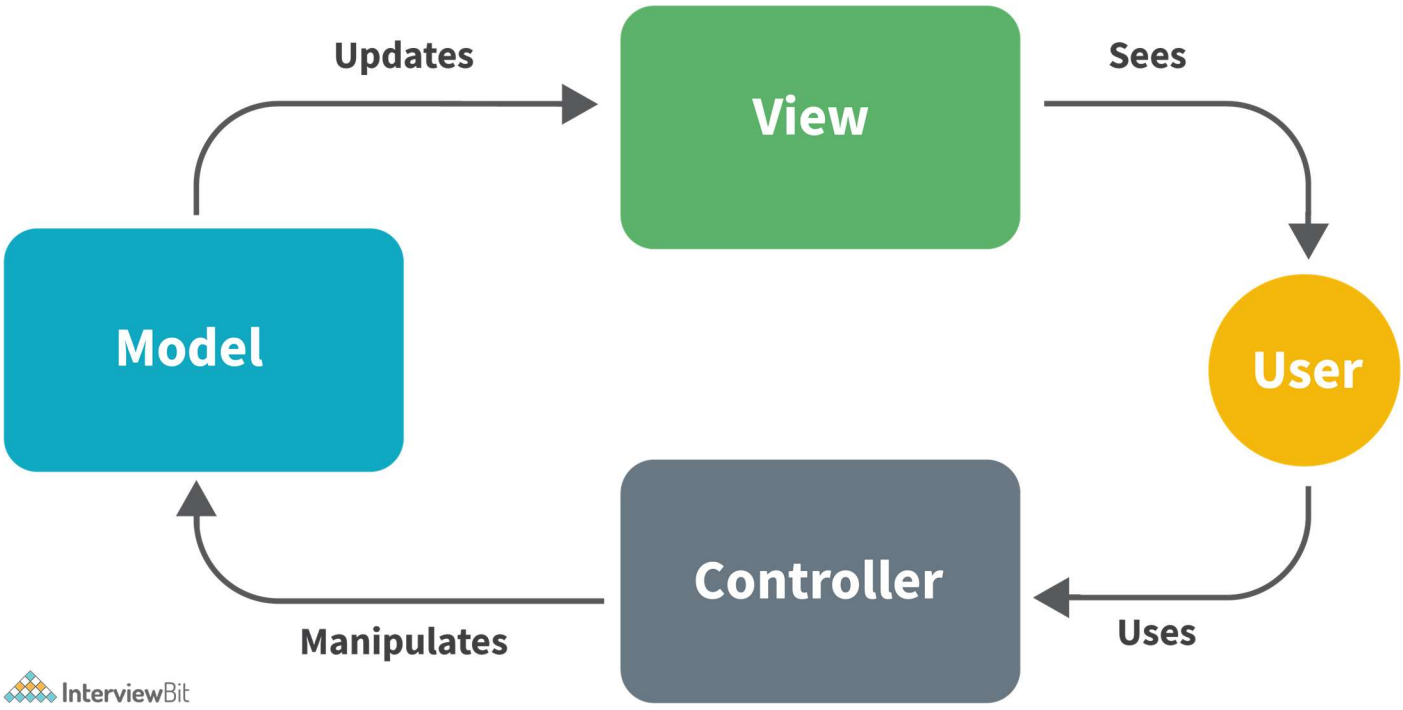
The MVC model is normally used to develop modern applications with user interfaces. It provides the central pieces for designing a desktop or mobile application, as well as modern web applications.

In this article, you will find a collection of real-world MVC interview questions with inline answers that are asked in top tech companies. So, here we go!

MVC Interview Questions and Answers

1. Explain Model, View and Controller in Brief.

- A model can be defined as the data that will be used by the program. Commonly used examples of models in MVC are the database, a simple object holding data (such as any multimedia file or the character of a game), a file, etc.
- A view is a way of displaying objects (user interfaces) within an application. This is the particular vertical through which end users will communicate.
- A controller is the third vertical which is responsible for updating both models and views. It accepts input from users as well as performs the equivalent update. In other words, it is the controller which is responsible for responding to user actions.



2. What are the different return types used by the controller action method in MVC?

The various return types of controller action methods in MVC are:

- View Result
- JSON Result
- Content Result
- Redirect Result
- JavaScript Result

3. Name the assembly in which the MVC framework is typically defined.

In the `System.Web.Mvc`, the MVC framework is usually defined.

4. Explain the MVC Application life cycle.

Web applications usually have 2 primary execution steps. These are:

- Understanding the request.
- Sending an appropriate response based on the type of request.

The same thing can be related to MVC applications also whose life cycle has 2 foremost phases:

- For creating a request object.
- For sending the response to any browser.

5. What are the various steps to create the request object?

In order to create a request object, we have to go through 4 different steps. These are:

- Step 1: Fill the route.
- Step 2: Fetch the route.
- Step 3: Create a request context.
- Step 4: Create a controller instance.

6. Explain some benefits of using MVC?

Some common **benefits of MVC** are:

- **Support of multiple views**: Since there is a separation of the model from its view, the user interface (UI) gets the capability to implement multiple views of the same data concurrently.
- **Faster development process**: MVC has the ability to provide rapid and parallel development. This means that while developing an application, it is more likely that one programmer will perform some action on the view and in parallel, another can work on creating the application's business logic.
- **SEO-friendly development**: The platform of MVC can support the SEO-friendly development of web pages or web applications.
- **More Control**: The MVC framework (of ASP.NET) offers additional control over HTML, CSS and JavaScript than that of traditional WebForms.
- **Lightweight**: MVC framework does not make use of View State which eventually minimises the requested bandwidth to some extent.

7. Explain in brief the role of different MVC components?

The different MVC components have the following roles:

- **Presentation**: This component takes care of the visual representation of a particular abstraction in the application.
- **Control**: This component takes care of the consistency and uniformity between the abstraction within the system along with their presentation to the user. It is also responsible for communicating with all other controls within the MVC system.
- **Abstraction**: This component deals with the functionality of the business domain within the application.

8. How will you maintain the sessions in MVC?

The sessions of an MVC can be maintained in 3 possible ways:

- viewdata
- temp data and
- view bag

9. What do you mean by partial view of MVC?

A partial view can be defined as a portion of HTML that is carefully injected into an existing DOM. Partial views are commonly implemented for componentizing Razor views, making them simpler to build and update. Controller methods can also directly return the partial views.

10. Explain in brief the difference between adding routes in a webform application & an MVC application?

We make use of the MapPageRoute() which is of the RouteCollection class for adding routes in a webform application. Whereas, the MapRoute() method is used for adding routes to an MVC application.

11. How will you define the 3 logical layers of MVC?

The 3 logical layers of MVC can be defined as follows:

- Model logic acts as a business layer.
- View logic acts as a display layer.
- Controller logic acts as input control.

12. What is the use of ActionFilters in MVC?

ActionFilters are used for executing the logic while MVC action is executed. Furthermore, action filters permit the implementation of pre and post-processing logic and action methods.

13. How to execute any MVC project? Explain its steps.

For executing an MVC project, the steps followed are:

- Receive the first request for the application.

- Then, the routing is performed.
- Then, the MVC request handler is created.
- After that, the controller is created and executed.
- Then, the action is invoked.
- Then, the results are executed.

14. What is the concept of routing in MVC?

MVC routing can be defined as a pattern-matching scheme that is used for mapping incoming requests of browsers to a definite MVC controller action.

15. What are the 3 important segments for routing?

The 3 important segments for routing are:

- ControllerName.
- ActionMethodName.
- Parameter.

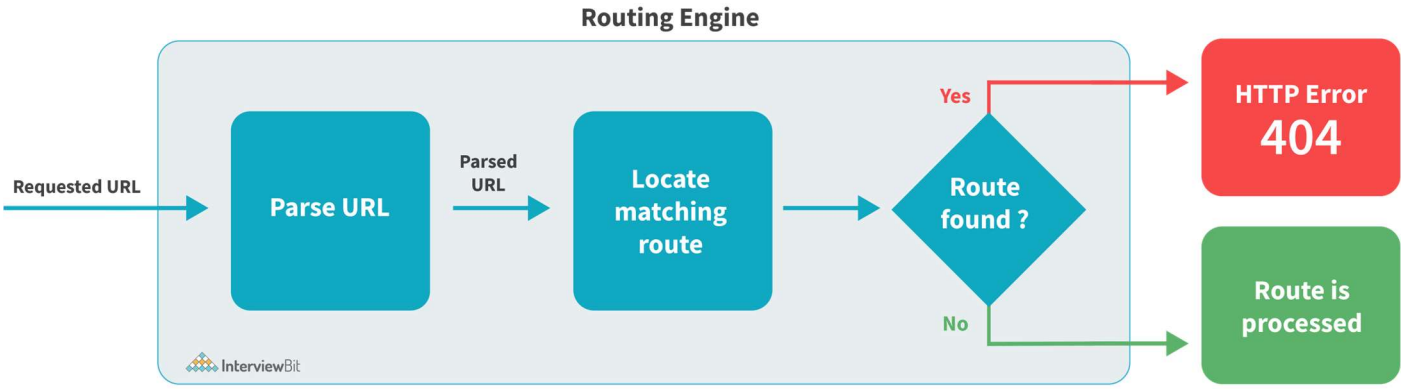
16. What are the different properties of MVC routes?

MVC routes are accountable for governing which controller method will be executed for a given URL. Thus, the URL comprises the following properties:

- **Route Name**: It is the URL pattern that is used for mapping the handler.
- **URL Pattern:** It is another property containing the literal values as well as variable placeholders (known as URL parameters).
- **Defaults:** This is the default parameter value assigned at the time of parameter creation.
- **Constraints:** These are used for applying against the URL pattern for more narrowly defining the URL matching it.

17. How is the routing carried out in MVC?

The RouteCollection contains a set of routes that are responsible for registering the routes in the application. The RegisterRoutes method is used for recording the routes in the collection. The URL patterns are defined by the routes and a handler is used which checks the request matching the pattern. The MVC routing has 3 parameters. The **first parameter** determines the name of the route. The **second parameter** determines a specific pattern with which the URL matches.
The **third parameter** is responsible for providing default values for its placeholders.



18. How will you navigate from one view to another view in MVC? Explain with a hyperlink example.

We will make use of the ActionLink method which will help us to navigate from one view to another. Here is an example of navigating the Home controller by invoking the Go to Home action. He is how we can code it:

```
<%=Html.ActionLink("Home","GoTo Home")%>
```

19. Explain the 3 concepts in one line; Temp data, View, and Viewbag?

We can briefly describe Temp data, View, and Viewbag as:

- **Temp data:** This is used for maintaining the data when there is a shift of work from one controller to another.
- **View data:** This is used for maintaining the data when we will shift from a controller to a view within an application.
- **View Bag:** This acts as a view data's dynamic wrapper.

20. Mention & explain the different approaches you will use to implement Ajax in MVC?

There are 2 different approaches to implementing Ajax in MVC. These are:

- **jQuery:** This is a library written using JavaScript for simplifying HTML-DOM manipulation.
- **AJAX libraries:** Asynchronous JavaScript and XML libraries are a set of web development libraries written using JavaScript and are used to perform common operations.

21. How will you differentiate between ActionResult and ViewResult?

Some common differentiation between ActionResult and ViewResult is:

ActionResult	ViewResult
It becomes effective if you want to derive different types of views dynamically.	It is not so effective in deriving different types of views dynamically.

ActionResult	ViewResult
It is an abstract class, meaning it has methods and variables without the implementation body of instruction.	This has been derived from the ActionResult class.
JsonResult, ViewResult, and FileStreamResult are some examples of its derived class.	This class do not have its own derived class.

22. What is Spring MVC?

The **Spring MVC** or Spring Web MVC can be defined as a framework that provides a "Model View Controller" (MVC) architecture in the application as well as ready components implemented for developing adjustable and adaptable web applications. It is actually a Java-based framework intended to build web applications. It works on the Model-View-Controller design approach. This framework also makes use of all the elementary traits of a core Spring Framework such as dependency injection, lightweight, integration with other frameworks, inversion of control, etc. Spring MVC has a dignified resolution for implementing MVC in Spring Framework with the use of DispatcherServlet.

23. Explain briefly what you understand by separation of concern.

Separation of Concerns can be defined as one of the core features as well as benefits of using MVC and is supported by ASP.NET. Here, the MVC framework offers a distinct detachment of the different concerns such as User Interface (UI), data and the business logic.

24. What is TempData in MVC?

TempData can be defined as a dictionary object used for storing data for a short period of time. This is the MVC's TempDataDictionary class which acts as a Controller base-class's instance property. TempData has the ability to preserve data for an HTTP request.

25. Define Output Caching in MVC?

Output Caching is an approach used for improving the performance of an MVC application. It is used for enabling its users to cache the data sent back by the controller method so that the data used earlier does not get generated each time while invoking the same controller method. It has advantages to use Output Caching as it cuts down database server round trips, minimizes server round trips as well as reduces the network traffic.

26. Why are Minification and Bundling introduced in MVC?

Two new techniques have been included in MVC, known as Bundling and minification, whose primary function is to progress the request load time. It advances the load time by dipping the number of requests sent to the server as well as reducing the requested asset's (JavaScript and CSS) size.

27. Describe ASP.NET MVC?

The term **ASP.NET MVC** can be defined as a web application framework that is very lightweight and has high testable features. ASP.NET supporting MVC uses 3 separate components in its application. These are the Model, View, and Controller.

28. Which class will you use for sending the result back in JSON format in MVC?

For sending back the result in JSON format in any MVC application, you have to implement the "JSONRESULT" class in your application.

29. Make a differentiation between View and Partial View?

The major differentiation between View and Partial View is as follows:

View	Partial View
The view is not as lightweight as that of the Partial view.	Partial view, as the name suggests, is lightweight than View.
The view has its own layout page.	The partial view does not have its own layout page.
The Viewstart page is rendered just before rendering any view.	A partial view is designed particularly for rendering within the view.
The view can have markup tags of HTML such as HTML, head, body, title, meta, etc.	The partial view does not contain any markup.

30. Define the concept of Filters in MVC?

There are situations where I want to implement some logic either prior to the execution of the action method or right after it. In that scenario, the Action Filters are used. Filters are used to determine the logic needed for executing before or after the action method gets executed. Action methods make use of the action filters as an attribute.

Different types of MVC action filters are:

- Action filter (that implements the IActionFilter).
- Exception filter (that implements the IExceptionFilter attribute).
- Authorization filter (that implements the IAuthorizationFilter).
- Result filter (that implements the IResultFilter).

31. Mention the significance of NonActionAttribute?

The various public methods that are associated with the controller class are considered to be the action method. For preventing the default method, you have to allocate its public method with NonActionAttribute.

32. What is used to handle an error in MVC?

Error handling is usually done using Exception handling, whether it's a Windows Forms application or a web application. The HandleError attribute is used, which helps in providing built-in exception filters. The HandleError attribute of ASP.NET can be functional over the action method as well as Controller at its global level.

Example of implementation:

```
public static void RegGlobalFilters(Global_FilterCollection filt)
{
    filt.Add(new HandleErrorAttribute());
}
protected void Application_Start()
{
    AreaRegn.RegisterAllAreas();
    RegGlobalFilters(Global_Filters.Filters);
    RegisterRoutes(Route_Table.Routes);
}
```

33. Define Scaffolding in MVC?

Scaffolding can be defined as an ASP.NET's code-generation framework used in web applications. Scaffolding is used in developing MVC applications when anyone wants to rapidly enhance the code that intermingles with the application's data model. Scaffolding can also lower the quantity of time for developing a standard data operation in the application.

34. When multiple filters are used in MVC, how is the ordering of execution of the filters done?

The order in which filters are used:

- First, the authorization filters are executed.
- Followed by the Action filters.
- Then, the response filters are executed.
- Finally, the exception filters.

35. What is ViewStart?

A new layout called _ViewStart is introduced by the Razor View Engine that is applied to all views automatically. ViewStart is executed at the very beginning followed by the start rendering as well as other views.

Example:

```
@ {
  Layout = "~/ Views/ Shared/ _
  file.cshtml";
}
<html>
  <head>
  <meta name="viewport" />
  <title> InitialView </title> </head>
  <body>  ….
  </body>
</html>
```

36. Which type of filters are executed in the end while developing an MVC application?

In the end, while developing an MVC application, the "Exception Filters" are executed.

37. Mention the possible file extensions used for razor views?

The different file extensions that are used by razor views are:

- **.cshtml**: When your MVC application is using C# as the programming language.
- **.vbhtml**: When your MVC application is using VB is the programming language.

38. Explain briefly the two approaches of adding constraints to an MVC route?

For adding constraints to an MVC route, the 2 different approaches are:

- By making use of regular expressions.
- By making use of objects that implement the "IRouteConstraint" interface.

39. Point out the different stages a Page life cycle of MVC has?

The different steps or stages of the page life-cycle of MVC are:

- Initialization of app.
- Routing.
- Instantiate the object followed by executing the controller.
- Locate as well as invoke the controller action.
- Instantiating and then rendering the view.

40. Explain briefly the use of ViewModel in MVC?

ViewModel can be defined as a plain class having different properties. It is used for binding a view that is strongly typed. ViewModel consists of various validation rules for defining the properties of practising data annotation.

41. Define Default Route in MVC?

The default Route of project templates in MVC includes a generic route that makes use of the given URL resolution for breaking the URL based on the request into 3 tagged segments. **URL**: "{controller} / {action} / {id}"

42. Explain briefly the GET and POST Action types?

- The **GET Action Type** is implemented for requesting the data from a particular resource. Using these GET requests, a developer can pass the URL (that is compulsory).
- The **POST Action Type** is implemented for submitting the data that needs to be handled to a certain resource. Using these POST requests, a developer can move with the URL, which is essential along with the data.

43. What are the rules of Razor syntax?

The primary rules for creating Razor are:

- The block of Razor codes is enclosed within @{ ... }.
- Variables and functions of inline expressions start with @ symbol.
- The 'var' keyword is used for declaring variables.
- Razor code statements are terminated with a semicolon.
- C# files have .cshtml as file extension.

44. How can you implement the MVC forms authentication?

Authentication in forms is added in order to include a layer of security to access the user for a specific service. This authentication is done by verifying the user's identity through the credentials such as username with password or email with a password.

The code snippet will look something like this:

```
<system.web>
    <authentication mode = "Forms" >
        <formsloginUrl = "Login.aspx" protection = "All" timeout = "30" name =
".ASPXAUTH" path = "/" requireSSL = "false" defaultUrl = "default.aspx" cookieless =
"UseDeviceProfile" />
    </authentication>
</system.web>
```

45. What are the areas of benefits in using MVC?

The area of benefits of using MVC is:

- Unit testing becomes much easier.
- It permits its users to shape views, models, and controllers into 3 distinct operational sections within an application.
- It becomes easy to assimilate with other areas produced by another application.

46. Point out the two instances where you cannot use routing or where routing is not necessary

The 2 situations where routing is not used or not necessary are:

- When there is a physical file matching the URL pattern.
- When any routing gets disabled in any particular URL pattern.

47. How will you explain the concept of RenderBody and RenderPage of MVC?

RenderBody can be considered as a ContentPlaceHolder of web forms. It is available on the layout page and will be responsible for rendering the child pages/views. On the other hand, the layout page contains a single RenderBody() method. Multiple RenderPage() can reside within the Layout page.

Introduction to ADO.NET

ADO.NET is a technology used for data and is provided by the Microsoft .NET Framework. It is a part of the .NET framework that supports the communication between relational and non-relational systems with the help of a set of software components. It supports disconnected architecture using which programmers are allowed to access data and data services from a database without depending on the data source.

ADO.NET is comprised of a group of built-in classes that are useful for establishing the database connection, for gaining access to XML, relational data, and application data, and for retrieval of a result. It can be used in various programming languages such as Visual Basic.NET, Visual C++, etc., that are supported by the .NET framework.

Advantages of ADO.NET

ADO.NET has various advantages which can be categorized into the following categories:

- **Interoperability:** It provides the ability to communicate across heterogeneous environments, once the connection has been established between them.
- **Scalability:** It provides the ability to serve an increasing number of clients without reducing the performance of the system. So we can say that ADO.NET is highly scalable because it is flexible enough to be easily expanded when there is a requirement for the same.
- **Productivity:** It provides the ability to rapidly develop robust applications for data access using rich and extensible component object models provided by the ADO.NET.
- **Performance:** An improvement over earlier ADO.NET versions because of the disconnected data model. It can establish connections quickly to fetch data without any delay.

Scope of ADO.NET

ADO.NET being one of the products of Microsoft, it is good enough to position itself strongly in the market. ADO.NET has massive community support, so it is definitely having a large scope ahead. You could learn ADO.NET along with hands-on experience on the .Net frame

work in order to have a good scope. Any full-stack developer who has a better grip over both front-end and back-end technology can precisely learn ADO.NET.

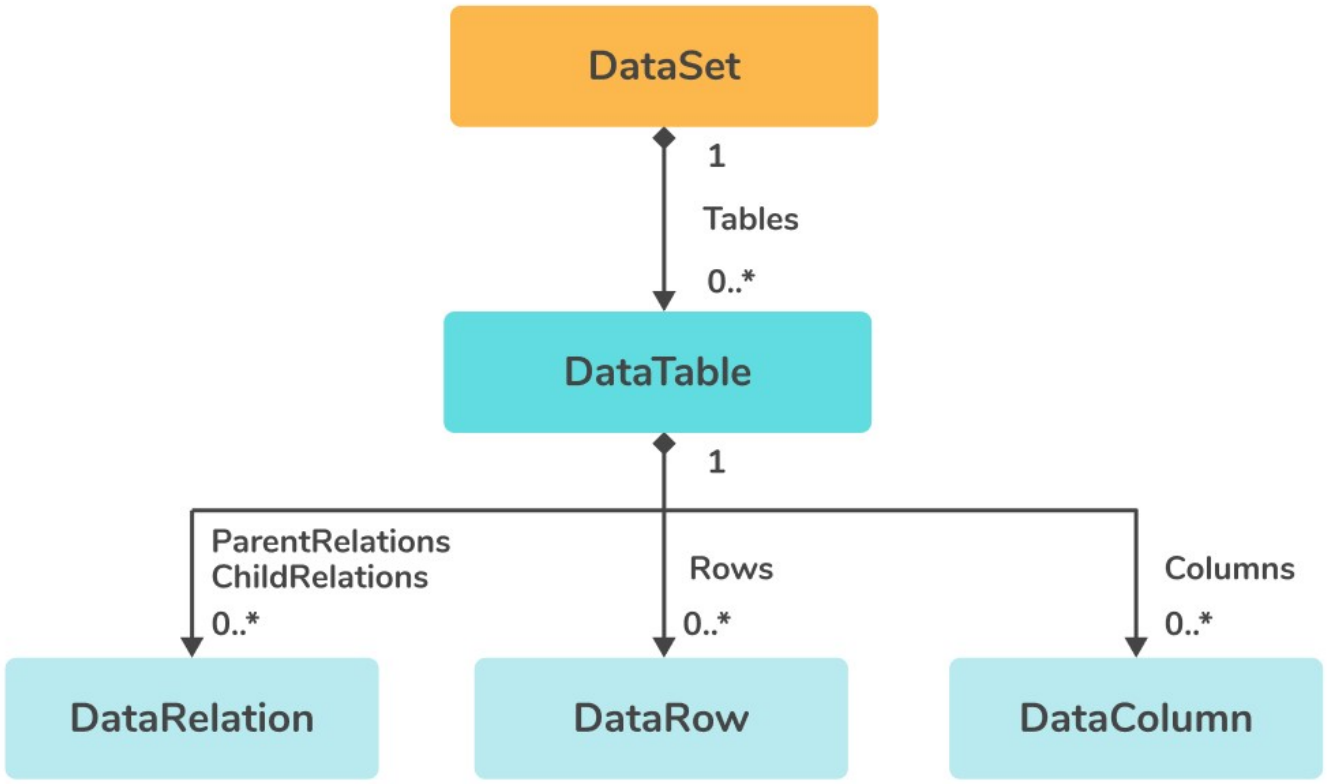
ADO.NET Interview Questions for Freshers

1. What is ADO.NET?

- ADO.NET stands for ActiveX Data Object, it is a part of the .NET Framework by Microsoft. ADO.NET framework provides a set of classes that are used to handle data communication with data sources such as XML files and databases (such as SQL, Oracle, MySQL, MS Access, etc.).
- ADO.NET can separate mechanisms for data connectivity, data access, and data manipulation.
- It has introduced the disconnected architecture, in which data can be stored in a DataSet. ADO.NET has providers for database connection, commands for execution, and result retrieval.
- The ADO.NET classes are stored in the DLL named `System.Data.dll`.
- Various applications like ASP.NET applications, console applications, windows applications, etc., will use ADO.NET for database connection, command execution, and retrieval of data.

2. What is DataSet in ADO.NET?

- The DataSet is a collection of database tables(row and column format) that contain the data. It is helpful for fetching the data without any need for Data Source interaction, that is why it is called a disconnected data access method.
- It is an in-memory data store that can contain multiple tables at the same time. DataRelation objects can be used to relate these tables.
- For creating a DataSet object, ADO.NET provides a DataSet class that consists of constructors and methods to carry out data-related operations.
- It can be used with various data sources, with XML data, or to manage the application's local data. The DataSet will include related tables, data constraints, and relationships among the tables.



Structure Of A DataSet

InterviewBit

3. Give the differences between ADO and ADO.NET.

ADO	ADO.NET
It is Component Object Modelling(COM) based.	It is Common Language Runtime(CLR) based.
It works in connected mode to access the data store.	It does require an active connection, works in disconnected mode to access the data store.
It uses the RecordSet object to access and store data from the data sources.	It uses a DataSet object to access and store data from the data sources.
It provides a feature of locking.	It does not provide a feature of locking.
Data is stored in binary form.	Data is stored in XML.
It does not support XML integration.	It supports XML integration.
Using a single connection instance, it is not possible to send multiple transactions.	Using a single connection instance you can send multiple transactions.
We can create only client-side cursors.	Both client-side and server-side cursors can be created.
It supports sequential row access in a RecordSet.	Non-sequential data access is supported in DataSet by using a collection-based hierarchy.

ADO	ADO.NET
It will make use of SQL JOINs and UNIONs for combining data from multiple tables. It is not possible to fetch records from multiple tables independently.	It will make use of DataRelational objects to combine data from multiple tables without the help of JOINs and UNIONs. Therefore records from multiple tables are maintained independently.

You can download a PDF version of Ado Net Interview Questions.

Download PDF

---

4. What is a DataAdapter in ADO.NET?

- A DataAdapter is used to access data from a data source by functioning as a bridge between DataSet and a data source. DataAdapter class includes an SQL command set and a database connection. It is helpful to fill the DataSet and resolve changes to the data source.
- The DataAdapter will make use of the Connection object that belongs to the .NET Framework data provider for connecting with a data source. Along with that, it will also use Command objects to retrieve data from the data source as well as to resolve changes to the data source.
- DataAdapter properties that permit the user to control the database are the Select command, Update command, Insert command, and Delete command.
- Example code for the usage of DataAdapter:

```
using System;
using System.Data.SqlClient;
using System.Data;
namespace DataAdapterExample
{
    public partial class DataAdapterDemo : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            using (SqlConnection conn = new SqlConnection("data source=.;
database=items; integrated security=SSPI"))
            {
                SqlDataAdapter da = new SqlDataAdapter("Select * from items", conn);
                DataSet s = new DataSet();
                da.Fill(s);
                GridView1.DataSource = s;
                GridView1.DataBind();
            }
        }
    }
}
```

Here, DataAdapter will receive the data from the items table and fill the DataSet, which will be later used to display the information retrieved from the items database.

5. Explain the difference between ADO.NET and ASP.NET.

ADO.NET(ActiveX Data Objects)	ASP.NET(Active Server Pages)
ADO.NET is a Library within the .NET framework.	ASP.NET is a Framework.
It is a technology useful for accessing data from databases.	It is a technology useful for the creation of dynamic web pages.
Here, data can be converted into XML format.	Here, We can write our code into VB.Net, C#, ASP.Net, etc.
It is used to develop reliable and scalable database applications with high performance for client-server applications.	It is used to create dynamic web pages, web applications, websites, and web services.

6. Explain about DataSet types in ADO.NET.

DataSet can be said as a collection of database tables(row and column format) that holds the data. There are two types of DataSet in ADO.NET. They are:

1. **Typed DataSet:** A typed DataSet is derived from the DataSet base class and can be created by selecting the DataSet option provided by Visual Studio. It will be created as an XML schema(.xsd file) that contains DataSet structure information such as rows, columns, and tables. Data from the database is moved into a dataset and from the dataset to another component in the XML format.
2. **Untyped DataSet:** Untyped DataSet does not have an associated XML schema with it. Users are supposed to add columns, tables, and other elements to it. Properties can be set during design time or can add them during run time.

Example program for the usage of DataSet:

```
using System;
using System.Data.SqlClient;
using System.Data;
namespace DataSetDemo
{
    public partial class DataSetExample : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            using (SqlConnection conn = new SqlConnection("data source=.;
database=employee; integrated security=SSPI"))
            {
                SqlDataAdapter da = new SqlDataAdapter("Select * from employee",
conn);
                DataSet d = new DataSet();
                da.Fill(d);
                GridView1.DataSource = d;
                GridView1.DataBind();
            }
        }
    }
}
```

Here, DataSet will be filled by DataAdapter that receives data from the employee table. This DataSet will be used to display the information received from the employee database.

7. Explain the difference between DataTable and DataSet.

DataTable	DataSet
DataTable consists of a single database table that is placed within a memory.	DataSet consists of a collection of multiple database tables which is placed within a memory.
It has a row and column collection.	It has a database table collection.
It allows fetching only a single TableRow at a time.	It allows fetching multiple TableRows at a time.
It is a single database table, so there will not be any relation with other tables.	It represents a collection of DataTable objects, so there might be a relation between them to obtain a particular result.
In this, DataSource objects are not serialized.	In this, DataSource objects are serialized.
UniqueConstraint and ForeignKeyConstraint objects are not available enforcing data integrity.	UniqueConstraint and ForeignKeyConstraint objects are available for enforcing data integrity.

8. What are the different namespaces available in ADO.NET?

Various namespaces available under ADO.NET is given below:

1. **System.Data:** It contains the definition for rows, columns, relations, views, tables, constraints, and databases.
2. **System.Data.SqlClient:** It is a collection of classes that are helpful in connecting to a Microsoft SQL Server database such as SqlConnection, SqlCommand, SqlDataAdapter, etc.
3. **System.Data.Odbc:** It consists of classes that are required for connecting with most Odbc Drivers. These classes include OdbcConnection, OdbcCommand.
4. **System.Data.OracleClient:** It has classes required for connection with an Oracle database, OracleConnection, OracleCommand.

9. What is object pooling?

Object pooling is a repository of the objects in memory that can be reused later without creating them. This object pooling reduces the burden of creating objects when it is required. Whenever there is a requirement of an object, the object pool manager will process the request and serve accordingly. It is designed for optimizing the use of limited resources so that the demands of client requests will be fulfilled.

10. Differentiate DataSet and DataReader.

DataSet	DataReader
DataSet provides read/write access to data, so we can update the data.	DataReader provides read-only access to data, so we can't update the data.

DataSet	DataReader
It has a disconnected architecture, which means the data obtained from the database can be accessed even after the database connection was closed.	It has a connected architecture, which means to access the data retrieved from the database, the connection must be opened.
It supports various database tables from different databases.	It supports only a single table from a single database.
It provides slower access to data due to overhead.	It provides faster access to data.
Both forward and backward scanning of data is possible.	Only forward scanning of data is possible.

11. What are the different execute() methods available in ADO.NET?

Different execute() methods supported by SqlCommandObject in ADO.NET is given below:

- **ExecuteScalar():** This method returns only a single value from the first row and first column of the ResultSet after the execution of the query. Even if ResultSet is having more than one row or column, all those rows and columns will be ignored. If the ResultSet is empty, it will return NULL.
- **ExecuteNonQuery()**: This method returns the number of rows affected by the execution of a query. This method is not useful to return the ResultSet.
- **ExecuteReader()**: This method returns an object of DataReader which is a read-only and forward-only ResultSet. It needs a live connection with the Data Source. We cannot directly instantiate the DataReader object. A valid DataReader object can be created with the help of the ExecuteReader() method.
- **ExecuteXmlReader()**: This method builds an object of the XmlReader class and will return the ResultSet in the form of an XML document. This method is made available in SQL Server 2000 or later.

12. What is a transaction in ADO.NET? Explain the types of transactions available in ADO.NET.

In ADO.NET, transactions are used when you want to bind several tasks together and execute them in the form of a single unit. The transaction provides data consistency by ensuring either all of the database operations will be succeeded or all of them will be failed. For example, consider an application that performs two tasks. First, it updates an item_order table with order information. Second, it updates an item_inventory table that holds inventory information, where a number of items ordered will be debited. If any one of the tasks fails, then both updates must be rolled back.

Two types of transactions supported by ADO.NET are as follows:

- **Local Transaction:**
  - A local transaction is a single-phase transaction that is directly handled by the database. Every .NET Framework data provider has its own Transaction object for bringing out local transactions.
  - For example, if we want to perform a transaction using SQL Server database, we import a `System.Data.SqlClient` namespace. Similarly, to perform an Oracle transaction, import

the `System.Data.OracleClient` namespace. A `DbTransaction` class will be used for writing code that is independent of the provider and that requires transactions.

- **Distributed Transaction:**
  - A distributed transaction is coordinated by a transaction monitor and will make use of fail-safe mechanisms like two-phase commit for transaction resolution. This transaction will affect multiple resources.
  - If the user can make use of a distributed transaction, if he wants to do a transaction across multiple data servers such as Oracle, SQL Server, etc.
  - If you want a distributed transaction to commit, all participants must guarantee that data modification made will be permanent. Changes must remain unchanged even if the system crash or other unforeseen events occur. Even if a single participant will make this guarantee fail, then the entire transaction will fail, and updates made to data within the transaction scope are rolled back.

13. Explain the difference between OLEDB (Object Linking and Embedding DataBase) and ODBC (Open DataBase Connectivity).

OLEDB	ODBC
An API(Application Programming Interface) that allows accessing data from different sources in a uniform manner.	It is an API for accessing DBMS (DataBase Management System).
It supports both relational and non-relational databases.	It supports only relational databases.
It is procedural-based.	It is component-based.
It is easier to deploy.	It is difficult to deploy.
It gives a higher performance on loading and extracting the data.	It performs less compared to OLE DB on loading and extraction of data.
`OleDbConnection = New OleDbConnection(connetionString)` is used to make connection with OLE DB data source.	`resource odbc_connect(string datasource , string username , string password , [int cursor_type ])` is used to make a connection to an ODBC data source. On success, this function will return a connection resource handle that is helpful in accessing the database using subsequent commands.

14. What is data binding in ADO.NET?

- Data binding in ADO.NET is the process through which user interface (UI) controls of a client application are configured to update or fetch data from data sources like a database or XML document. Using data binding, the user will be able to bind values to the particular control.
- There are two types of data binding based on the type of binding offered:
  1. **Simple data binding:** It is the process of binding the control with only one value in the dataset. The controls such as label, text box will be made bound to the control using the control properties.

2. **Complex data binding:** It is the method of binding the component with the Database. The controls can be a Dropdown list, GridView, or combo box. One or more than one value can be displayed from the dataset using the complex data binding.

15. What is Connection pooling?

The task of grouping database connections in the cache memory is to make them available whenever there is a requirement of connection. Opening a new database connection every time is a time-consuming process. Connection pooling allows you to reuse existing and active database connections, whenever there is a need, and thus increases the application performance. By setting the pooling property into true or false in the connection string, we can enable or disable the connection pooling in the application. It is enabled by default in every application.

16. What is DataTable in ADO.NET?

DataTable in ADO.NET represents a single table in a DataSet that has in-memory relational data. The data within DataTable is local to the .NET framework-based application to which it belongs but can be populated using a DataAdapter from different data sources such as Microsoft SQL Server. The DataTable class belongs to the `System.Data` namespace within the library of .NET Framework.

DataTable can be represented in `.aspx.cs` code as given below:

```
protected void DataTableExample()
{
    SqlConnection conn = new SqlConnection("Write the database connection string");
    conn.Open();
    SqlCommand cd = new SqlCommand("Write the query or procedure", conn);
    SqlDataAdapter d = new SqlDataAdapter(cd);
    DataTable dt = new DataTable();
    d.Fill(dt);
    grid.DataSource = dt;
    grid.DataBind();
}
```

The SQL connection and SQL command object will be created. We pass the SQL query to the object of the SQL command class. A new data table object will be created by using the DataTable class and it is filled with data using a data adapter.

17. Name some of the properties and methods provided by the DataReader in ADO.NET?

Some of the properties provided by the DataReader are as follows:

- **Depth:** It represents the depth of nesting for a row.
- **FieldCount:** It gives the total column count in a row.
- **Item:** It obtains the column value in a native format.
- **RecordsAffected:** It gives the number of transaction affected rows.
- **IsClosed:** It represents whether a data reader is closed.
- **VisibleFieldCount:** It is used to obtain the number of unhidden fields in the SqlDataReader.

Some of the methods provided by the DataReader are as follows:

- **Read():** This method reads a record from the SQL Server database.
- **Close():** It closes a SqlDataReader object.
- **NextResult():** It moves the data reader to the next result during the time of batch transactions.
- **Getxxx():** Various types of Getxxx() methods such as GetBoolean(Int32), GetChar(Int32), GetFloat(Int32), GetDouble(Int32), etc., are provided by the DataReader. These methods will read a value of a particular data type from a column. For example, GetFloat() will return a column value as a Float and GetChar as a character.

18. What are the conditions for connection pooling?

The conditions for connection pooling are:

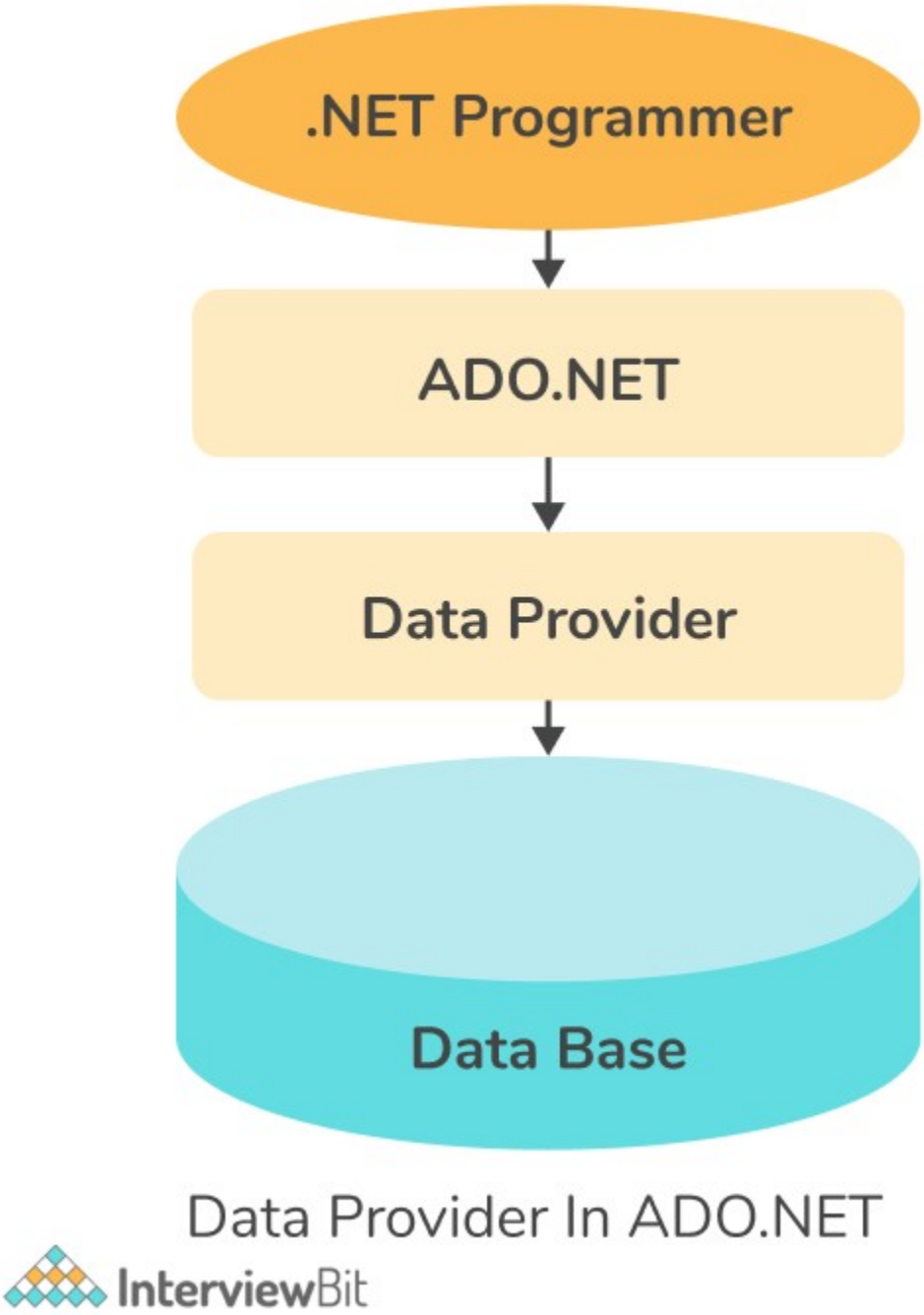
- There must be several processes with the same parameters and security settings so that they can share the same connection.
- The connection string should be identical.

19. What are the data providers in ADO.NET?

Data providers are used to transferring the data between the client application and the data store. It encapsulates the database-specific details. Data providers are helpful for database connection, data retrieval, storing the data in a dataset, reading the retrieved data, and updating the database.

The data providers that comes along with the ADO.NET Framework are:

- **OLE DB:** The OLEDB provider is available under `System.Data.OleDb` namespace. This provider can be used to access Microsoft Access, DB2/400, SyBase, and SQL Server 6.5 and earlier.
- **ODBC:** The ODBC provider is available under `System.Data.Odbc` namespace. This provider is used when there will not be any newer provider is available.
- **SQL Server:** The Microsoft SQL Server provider is available under `System.Data.SqlClient` namespace. Classes available under this provider will provide the same functionality as the generic OLEDB provider.



20. Why Stored Procedure is used in ADO.NET?

The reasons for using Stored Procedures in ADO.NET are given below:

- For improved performance
- For security reasons
- Easier to use and maintain
- Lesser Network Traffic
- Execution time is less

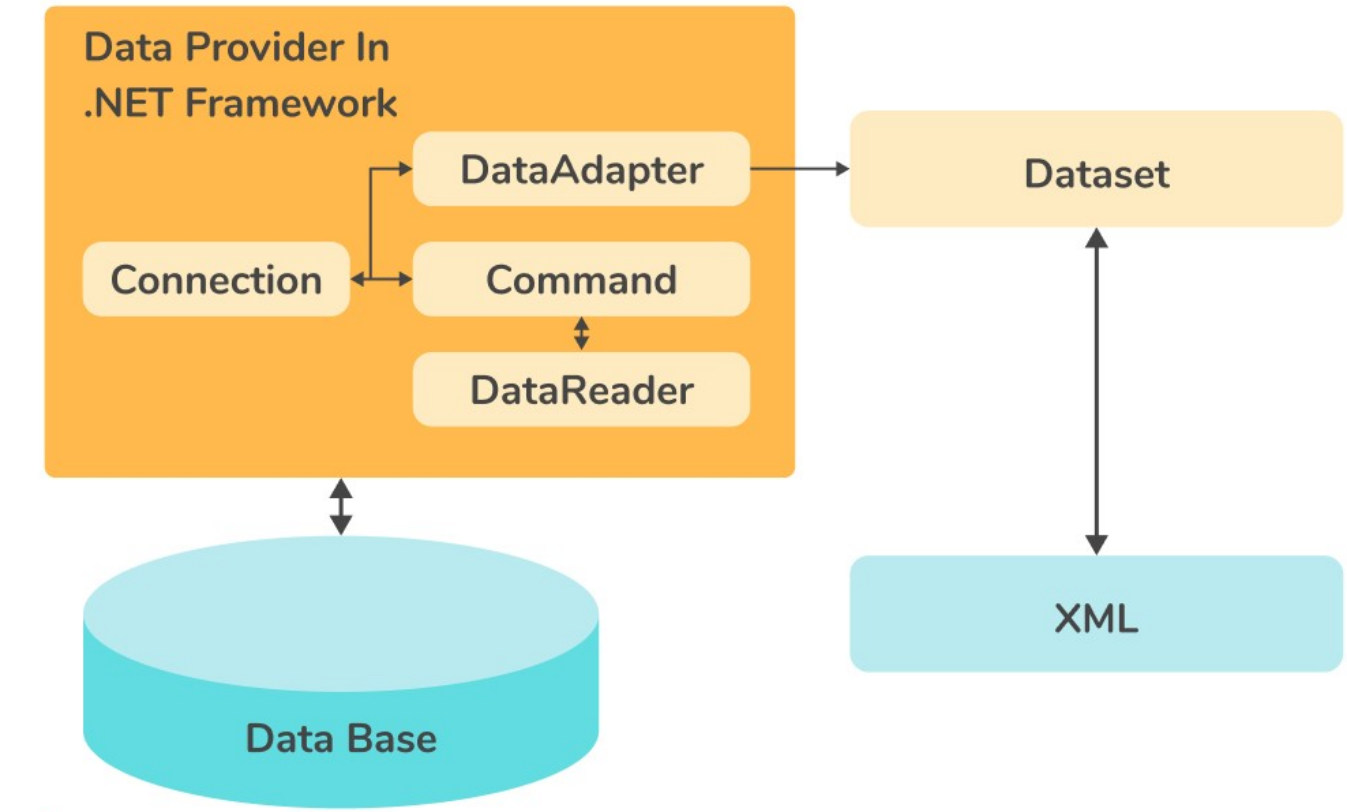
ADO.NET Interview Questions for Experienced

21. Explain ADO.NET Architecture.

ADO.NET is based on an Object Model where data residing in the database is accessed using a data provider. It is a technology of data access given by the Microsoft .Net Framework, which helps to communicate between relational and non-relational systems using a common group of components.

The components of ADO.NET architecture are:

- **Data Provider:** It provides data to all the applications that perform the database updates. The application can access data through the DataSet or DataReader object. A data provider is a having group of components such as Command, Connection, DataReader, and DataAdapter objects. Command and Connection objects are the necessary components irrespective of the operations like Insert, Delete, Select, and Update.
- **Connection:** The connection object is needed to connect with the database such as SQL Server, MySQL, Oracle, etc. To create a connection object, you must know about where the database is located(Ex: IP address or machine name, etc.) and the security credentials(Ex: user name and password-based authentication or windows authentication).
- **Command:** The command object is the component where you will write the SQL queries. Then by using the command object, execute the queries over the connection. By using the command object and SQL queries, you will be able to fetch the data or send the data to the database.
- **DataReader:** DataReader is a connected read-only RecordSet that is helpful in reading the records in the forward-only mode.
- **DataAdapter:** The DataAdapter acts as a bridge between the dataset and command object. It receives the data from the command object and puts it into the data set.
- **DataSet:** The DataSet is a disconnected RecordSet that can be browsed in both forward and backward directions. We can also update the data using the dataset. DataSet is filled by using DataAdapter.
- **DataView Class:** A DataView allows you to create various views of data from DataTable, which can be used for data-binding applications. Using this, you can display the table with different order of sorting or you can filter the data based on a filter expression or by row state, etc.
- **XML:** It is possible to create an XML representation of a dataset. In the dataset's XML representation, data is represented in XML format and the database schema is represented in XML Schema Definition(XSD) language.

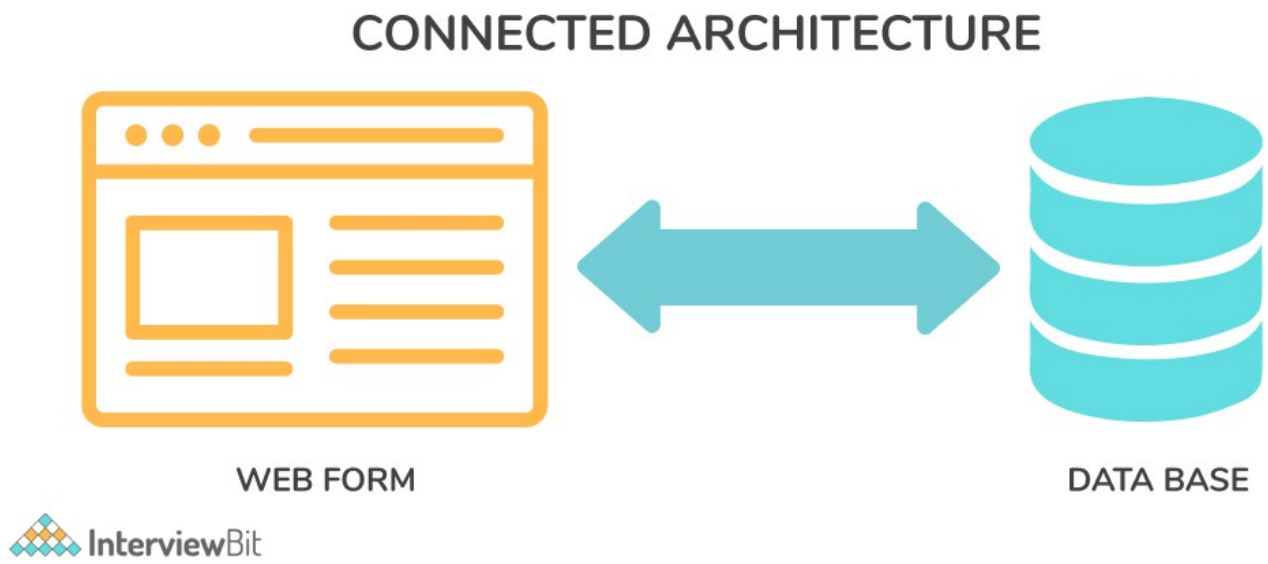


InterviewBit

22. Briefly explain connected and disconnected architecture of ADO.NET.

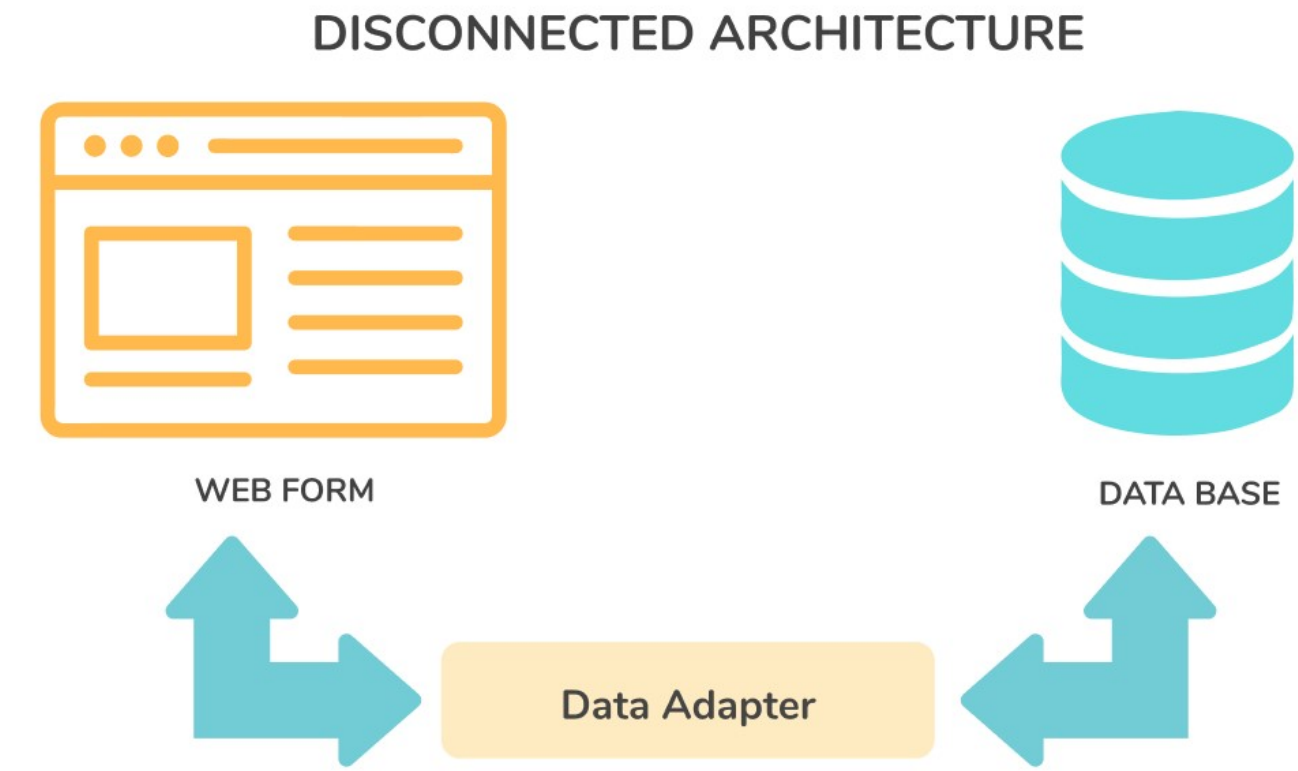
Connected Architecture:

- In connected architecture, the connection must be kept open for accessing the data retrieved from the database. Connected architecture is based on Connection, DataReader, Command, and Transaction classes.
- You constantly visit the database for any CRUD (Create, Read, Update, and Delete) operation you want to do. This will create high traffic to the database, but this is usually faster as you are doing only smaller transactions.
- DataReader can be said as a Connected Architecture as it holds the connection open until it fetches all the rows one by one.



Disconnected Architecture:

- In disconnected architecture, even if the database connection is closed, data retrieved from the database can be accessed. Disconnected architecture is based on classes connection, CommandBuilder, DataAdapter, DataSet, and DataView.
- Here, we retrieve and store a recordset from the database so that you can perform many CRUD (Create, Read, Update, and Delete) operations on the data within memory, it will be re-synchronized when you reconnect with the database.
- DataSet is a Disconnected Architecture because all records are brought at once and holding the database connection alive is not necessary.



InterviewBit

23. Explain about ExecuteScalar() in ADO.NET.

- A single value from the first row and first column of the ResultSet will be returned by ExecuteScalar() method on query execution.
- If the ResultSet is having multiple rows or columns, all those rows and columns will be ignored except the first row and first column. If the ResultSet is empty, this function will return NULL.
- The best situation to use ExecuteScalar() method is when we are using functions such as COUNT(), SUM(), etc., as it uses only a few resources compared to the ExecuteReader() method.
- Example:

```
public void ExecuteScalarExample()
{
    SqlConnection con = new SqlConnection();
    con.ConnectionString =
ConfigurationManager.ConnectionStrings["conString"].ConnectionString;
    try
    {
        SqlCommand cd = new SqlCommand();
        cd.Connection = con;
        cd.CommandText = "SELECT SUM(SALARY) FROM EMPLOYEE";
        cd.CommandType = CommandType.Text;
        con.Open();
        Int32 SalaryTotal = Convert.ToInt32(cd.ExecuteScalar());
        MessageBox.Show("Total Salary of the employee is : " +
SalaryTotal.ToString());
        cd.Dispose();
        con.Dispose();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
```

```
    }
}
```

Here, we create an object of the class SqlConnection and SqlCommand. We pass SQL Statement to the object of SqlCommand class, which returns a single value. When ExecuteScalar() function gets executed, a single value will be returned, i.e, the total salary of employees. This value will be displayed using a message box.

24. Explain about ADO.NET objects.

There are seven main objects in ADO.NET. They are:

1. **DataSet:** It is available under both `System.Data.ADO` and the `System.Data.SQL` namespaces. DataSet is a database cache built-in memory for using it in disconnected operations. It holds the complete collection of tables, constraints, and relationships.
2. **SQLDataSetCommand:** It represents a stored procedure or a database query that can be used to populate the DataSet object. It corresponds to the ADO's Command object-provided functionalities.
3. **SQLCommand:** It represents a stored procedure or a T-SQL statement that will be executed by SQL Server. It corresponds to another set of functionalities provided by the ADO's Command object.
4. **SQLParameter:** It can be used to pass parameters to the object of SQLCommand or SQLDataSetCommand class. When you are passing a parameter for SQLCommand using SQLParameter, SQLParameter will represent a parameter that can be used by T-SQL statement or stored procedure. Whenever a parameter has been passed for SQLDataSetCommand using SQLParameter, SQLParameter will represent a column from a result set.
5. **SQLConnection:** It represents an open connection to the data source like SQL Server. This object is similar to the standard Connection object in ADO.
6. **SQLDataReader:** It reads a forward-only stream of data from a SQL Server database. It works with an open database connection.
7. **SQLError:** It collects runtime warnings and error conditions related information that will be encountered by an ADO.NET application. It corresponds to ADO's Error object.

25. What are the different authentication techniques used to connect with MS SQL Server?

Before performing any task in the database, SQL Server will authenticate. Two types of authentication techniques are:

- **Windows Authentication:** This default authentication is provided only through Windows domain accounts. This SQL Server security model is strongly integrated with Windows, so it is also referred to as integrated security. Particular Windows users and group accounts are allowed to login into SQL Server. Windows users who are already been authenticated or logged onto Windows do not have to provide additional credentials.
  The below-given `SqlConnection.ConnectionString` specifies Windows authentication without any need of providing a user name or password by the user.

```
C#
"Server=MSSQL1;Database=Institute;Integrated Security=true;
```

- **SQL Server and Windows Authentication Mode(Mixed-mode):** Authentication will be provided with the help of the Windows and SQL Server Authentication combination. User name and password pair will be maintained within SQL Server. In order to use this mixed-mode authentication, you need to create SQL Server logins that are stored in SQL Server. After that, you can supply the user name and password to SQL Server at run time.

The below-given ConnectionString specifies Mixed mode authentication:

```
C#
"Persist Security Info=False;User ID=Harsh;Password=xyz@123;Initial
Catalog=Institute;Server=MySqlServer"
```

26. What is Response.Expires and Response.ExpiresAbsolute property?

- `Response.Expires` property is specific to the minutes that a particular page stays in the cache for the specific time from the time it has been requested. For example, if `Response.Expires` value is set to 5 minutes, then the page is instructed to be in cache for 5 minutes from the time it has been requested.
- `Response.ExpiresAbsolute` property helps to provide the proper time at which a specific page cache has been expired. For example, `Response.ExpiresAbsolute` provides information like 14 March 15:40:15. This time tells about when the page was in cache.

27. How to load multiple tables into a dataset?

```
DataSet ds=new DataSet();
SqlConnection con=new SqlConnection("connection_string");
SqlDataAdapter da=new SqlDataAdapter("select * from Employee1",con);
da.Fill(ds.Tables.Add());
da=new SqlDataAdapter("select * from Employee2",con);
da.Fill(ds.Tables.Add());
```

After tables have been added into a DataSet, the below-given code tells about how to make use of the DataSet tables. If you decide to use the first table in a dataset or to copy the table data into a data table, then follow the below-given code:

```
DataTable dt=new DataTable();
dt=ds.Tables[0];
```

The above code can be used to add the required number of tables in a dataset. This ensures connection-less access to data. As the dataset is filled with multiple tables, every time we want to query the data the database connection is not required. It also makes sure about the reusability of data.

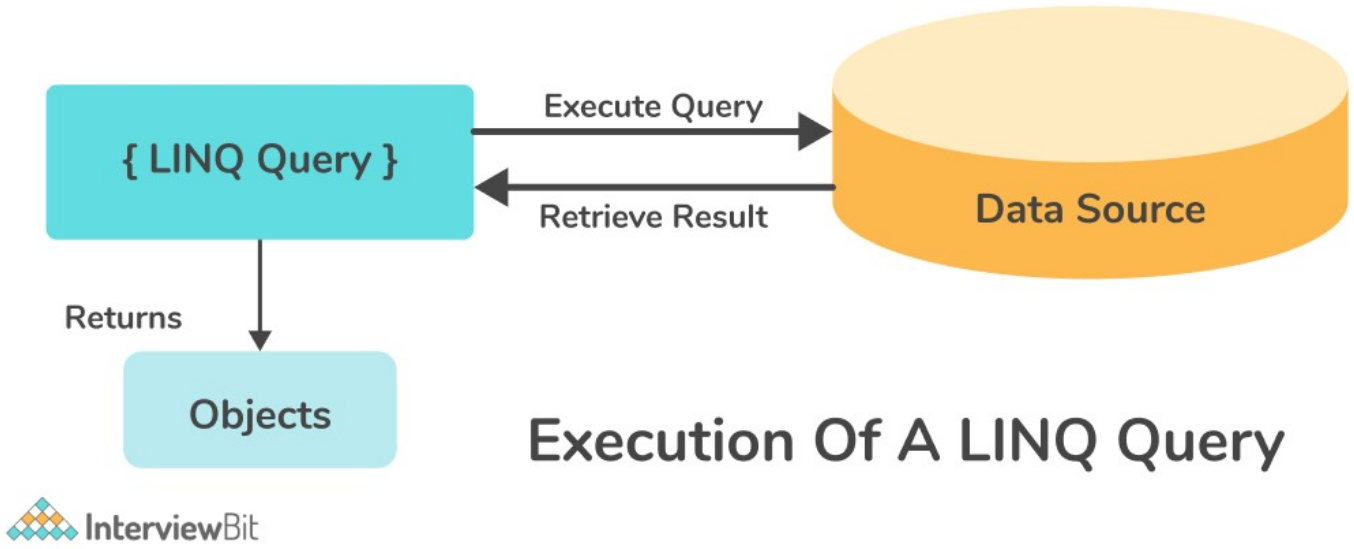
28. What is the difference between connected and disconnected architecture in ADO.NET?

Connected architecture	Disconnected architecture
It is connection-oriented.	It is not connection-oriented.
DataReader is a connected architecture.	DataSet is a disconnected architecture.

Connected architecture	Disconnected architecture
High speed and performance are given by connected methods.	Disconnected methods are low in speed and performance.
Data persistence is not possible using DataReader.	Data persistence is possible using DataSet.
It carries the single table data.	It carries data from multiple tables.
We can't update the data as it is read-only.	Here we can update the data.

29. What is LINQ?

- LINQ(Language Integrated Query) is a structured query syntax that helps the programmers and testers to retrieve data from various data sources such as Collections, XML Docs, ADO.NET DataSet, web service, MS SQL Server, etc.
- It is integrated with C# or VB.NET and it eliminates the mismatch between different programming languages and databases. It provides a single querying interface for various data source types.
- An object will be returned as a result of LINQ query execution. It will allow you to use an object-oriented approach on the result set and there is no need to worry about the transformation of different result formats into objects.



30. How can you identify whether any changes are made to the DataSet object since the time it was last loaded?

The DataSet object has two methods to track down the changes:

- **GetChanges():** It returns the DataSet object that has been changed since it was loaded or since the execution of the AcceptChanges() method.
- **HasChanges():** It indicates if any modifications were made since from the time the DataSet object was loaded or after a method call to the AcceptChanges() was made.

Use the RejectChanges() method, if you want to reverse the entire changes since from the time the DataSet object was loaded.

31. What is the difference between Dataset.Clone() and DataSet.Copy() methods?

- The method **Clone()** copies only the DataSet structure. The copied structure will have all the constraints, relations, as well as DataTable schemas used by the DataSet. It does not copy the data stored in the DataSet.
- The **Copy()** method copies the DataSet structure along with the data in the DataSet. The original data will not be affected.

32. Which methods are provided to add or remove rows from the DataTable object?

The collection of rows for the DataTable object has been defined by the DataRowCollection class. DataRowCollection class has the method NewRow() for adding a new DataRow to DataTable. This method creates a new row that implements the similar schema that is applied to the DataTable.

The methods provided by the DataRowCollection object are given below:

- **Add()**- It adds a newly created row into DataRowCollection.
- **Remove()**- It deletes the object DataRow from DataRowCollection.
- **RemoveAt()**- It deletes a row for which location is marked by an index number.

33. How to make SQL Server connection in ADO.NET?

Consider the below example where a connection to the SQL Server has been established. An employee database will be used to connect. The C# code will be:

```
using (SqlConnection con = new SqlConnection(connectionString))
{
  con.Open();
}
```

Using block will be useful in closing the connection automatically. It is not required to explicitly call the close() method, because using block will do this implicitly when the code exits the block.

```
// ConnectionExample.cs

using System;
using System.Data.SqlClient;
namespace ConsoleApplicationExample
{
    class ConnectionExample
    {
        static void Main(string[] args)
        {
            new Program().ConnectingMethod();
        }
        public void ConnectingMethod()
        {
            using (
                    // Creating Connection
                    SqlConnection conn = new SqlConnection("data source=.;
database=employee; integrated security=SSPI")
                )
            {
                conn.Open();
                Console.WriteLine("Connection Has Been Successfully Established.");
```

```
            }
        }
    }
}
```

Output:

```
Connection Has Been Successfully Established.
Press any key to continue...
```

On execution, if the connection has been established, a message will be displayed on an output window.

If the connection is not created with the help of using a block, a connection must be closed explicitly.

34. What is serialization? Write an example program to serialize a DataSet.

Serialization is the method of converting an object into a byte stream which can be stored as well as transmitted over the network. The advantage of serialization is that data can be transmitted in a cross-platform environment across the network and also it can be saved in a storage medium like persistent or non-persistent.

The code for serializing a DataSet is:

```
using System;
using System.Data;
using System.Data.SqlClient;
using System.Xml.Serialization;
using System.IO;
public partial class Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        SqlConnection conn = new SqlConnection("Data Source=data_source_name;Initial
Catalog=employee;Integrated Security=True");  //Create connection object
        SqlDataAdapter da = new SqlDataAdapter("select * from emp", conn);
//DataAdapter creation
        DataSet s = new DataSet();
        da.Fill(s);
        FileStream fObj = new FileStream("C:\\demo.xml", FileMode.Create);   //
Create a XML file
        XmlSerializer sObj = new XmlSerializer(typeof(DataSet));
        sObj.Serialize(fObj, s);  //Serialization of a DataSet
        fObj.Close();
    }
}
```

In the above given example, the database name is employee and, the table name is emp. The data in a DataSet will be serialized and stored in a demo.xml file by using Serialize() method.

35. Give an example code to fill the GridView by using the object of DataTable during runtime.

```
using System;
using System.Data;
```

```
public partial class Default : System.Web.UI.Page
{
     protected void Page_Load(object sender, EventArgs e)
     {
          GridView gridView1=new GridView();       //Create GridView object
          DataTable t = new DataTable("Employee"); // Create the table object
          DataColumn c = new DataColumn();        //Creating table column
          DataRow r;                              //Instance of row
          c.ColumnName = "EmpID";                 //Heading of the coloumn
          c.DataType = Type.GetType("System.Int32"); //Set the data type of EmpID as
an Integer
          t.Columns.Add(c);                      //Adding a column to data table
          c = new DataColumn();
          c.ColumnName = "EmpName";
          c.DataType = Type.GetType("System.String"); //Set the type of EmpName as
String
          t.Columns.Add(c);
          for (int i = 0; i < 5; i++)         //This code will create 5 rows
          {
               r = t.NewRow();
               //Add Column values
               r["EmpID"] = i;
               r["EmpName"] = "Employee " + i;
               t.Rows.Add(r);
          }
          gridView1.DataSource = t;  //Set gridView1 Datasource as DataTable t
          gridView1.DataBind();         //Bind Datasource to gridview
     }
}
```

Output:

EmpID	EmpName
0	Employee 0
1	Employee 1
2	Employee 2
3	Employee 3
4	Employee 4

Conclusion

ADO.NET is a brilliant technology that was developed by Microsoft on the framework of .NET. The primary role and responsibility of ADO.NET technology is to setup a bridge between backend language and your database. A good experience of this technology will be of great use from a development point of view.

ADO.NET technology will definitely help in your career growth as it has quite a good scope. Also learning this interesting technology will always be great fun. ADO.NET along with the knowledge of databases will definitely be exceptional from a growth perspective.