# C#.NET / VB.NET interview questions

Explain the elements of the .NET Framework.

a. CLR (Common Language Runtime): It is a common managed environment where all the .net programs run. Supports multiple languages and has the garbage collector.

b. .Net Framework Class Libraries: For each source code compiler (VB.NET, C#.NET, etc.), there is a minimum set of coding standards that must be met. The minimum set of coding standards that must be met to compile .NET code into MSIL code is known as CLS - Common Language Specification. The role of the Common Language Specification is to ensure that all generated code (MSIL) that meets the minimum set of coding standards can operate successfully within the .NET framework. THE CTS (Common Type System) handles conversion of programming-language data types into .NET compatible (MSIL) data types. The implicit benefit of the CTS is the reduction of development time when attempting to coordinate data types between two sets of different programming-language code.

c. Data and XML: Support for disconnected programming model and XML.

d. XML webservices: creating webservices for distributed architecture.

e. Webforms: Provides support and functionality for Web based UI.

f. Windows forms: Provides support and functionality for Windows based UI.

What is assembly manifest? What is the information it provides.

Assembly Manifest is a file that contains data that describes how the elements present inside an assembly are connected to each other. The assembly manifest contains assembly metadata to define the scope of the assembly and resolve references to resources and classes.

Information provided by Assembly Manifest:

a. Assembly Name
b. Version Number
c. Culture
d. Strong name
e. List of files inside the assembly
f. Reference information

Explain how a .NET application is compiled and executed

Any code written in any .NET complaint languages when compiled, converts into MSIL (Microsoft Intermediate Language) code in form of an assembly through CLS, CTS. IL is the language that CLR can understand. On execution, this IL is converted into binary code by CLR's just in time compiler (JIT) and these assemblies or DLL are loaded into the memory.

Describe the .NET base class library
.NET's Base class library exists in order to encapsulate huge number of common functions and makes them easily accessible to the developer. .NET base class library provides the functionality like ADO.NET, XML, Threading, IO, Security, Diagnostics, Resources, Globalization, collections etc. It serves as the main point of interaction between developer and runtime.

Explain the difference between value types and reference types

Value Type:
- a. Stores the data.
- b. The value of value types is stored on the managed stack.
- c. One variable can have just one value.
- d. They are lighter objects.

Reference Type:
- a. Stores the reference to the data.
- b. A reference type is allocated on the heap.
- c. several variables can reference the same data
- d. They are heavier objects.

Explain the importance of Imports and Using Statements.

Import statement: creates a property on the global object with the name supplied as namespace and initializes it to contain the object that corresponds to the namespace being imported. Any properties created using the import statement cannot be assigned to, deleted, or enumerated. All import statements are executed when a script starts.

Using statements: mainly defines the namespaces whose objects will be used in the form. This clearly solves 2 purposes: Defines all the namespaces that will be used in a form. Secondly, reduces the hassle for the programmer to type the name of namespace again and again while using classes/objects that belong to the namespace.

Explain the difference between a class and a structure

Class:
- a. It is reference type.
- b. Null value can be assigned to a variable in a class
- c. It can have destructor.
- d. All variables in classes are by default private.
- e. Good to be used from architecture view as it provides high flexibility.

Structure:
- a. It is value type.
- b. Null value assignment is not feasible here.
- c. Cannot have destructor.
- d. All variables in structures are public.
- e. Good to be used for simple data structures.

Explain how garbage collection manages the reclamation of unused memory.

The garbage collector assumes that all objects in the managed heap are garbage. It starts walking the roots and builds a graph of all objects reachable from the roots recursively. It stops when it attempts to add an object to the graph that it previously added. The graph contains the set of all objects that are reachable from the application's roots. Any object/s that is not in the graph is not accessible by the application, and is considered garbage. Collection only occurs when the heap is full. In such a case, each and every garbage object calls the Finalize method and reclaims the unused memory. 55. Explain how garbage collection deals with circular references.

Explain how garbage collection deals with circular references.

The .Net runtime knows about all the references between the objects. It can identify all the circular references that are reachable from the root and hence finalize them to free them all at once if and when needed.

Explain the process of creating a menu using the MainMenu component.

MainMenu component is a component that allows the display of Menus at runtime on a form.
Process of creating Menu using MainMenu Component:

a. Add MainMenu component on Windows Form.
b. Menu designer allows deciding the structure of the main menu by selecting the Type Here area and adding the Menu Items to be displayed on the menu.
c. Add functionality to Menu Items as required.

Explain the process of creating a context menu using the ContextMenu component

ContextMenu component provides the users with the ability to access some very frequently used commands. Context menu works by right click of mouse. They mainly provide access to commands particular to the control that has been clicked upon.

Process for creating context menus:

a. Open the windows form application.
b. Select ContextMenu component from toolbox.
c. A menu is added. Click on Type here and type in new Menu Items to be placed on the Menu.
d. Provide the functionality.
e. Associate the context menu with the form or the control it is supposed to be related to.

What is a delegate? Explain how to create it.

A delegate declares a ref type that references a named of anonymous method. Delegates are secure and type-safe. Consider them as type safe function pointers.

```
public delegate void Del<T>(T item);

Del<int> d1 = new Del<int>(Notify);
```

Explain how to declare and raise events from your application.

Declare Events: “Event” keyword is used to declare an event.

```
public delegate void MyCustomHandler(object o, MyEventArgse);

public class MyEventArgs: EventArgs
{
   public readonly int Age;

   public MyEventArgs(int age)
   {
      Age = age;
   }

}

public class MyCustomListener
{
   public void Show(object o, MyEventArgs e)
   {
      Console.WriteLine(
         "Age is {0}",
         e.Age);
   }
}
```

Describe how to implement event handlers and associate them with events.

```
public class MyClass
{
   public static event MyCustomHandler MyEvent;

   public static void Main()
   {
      MyCustomListener mcll = new MyCustomListener();
      MyEvent += new MyCustomHandler(mcl1.Show);
      GetAge();
   }

   public static void OnMyEvent(MyEventArgse)
   {
      if(MyEvent!=null)
         MyEvent(new object(),e);
   }

   public static void GetAge()
   {
      MyEventArgse1 = new MyEventArgs(25);
      OnMyEvent(e1);
```

```
    }

}
```

What is Break Mode? How to set breakpoints?

Break mode is the state of an application when the execution gets paused and allows the developer to edit the value in the current state. To attain a break mode we can do any of the following steps:

a. Selecting Break from the Run menu (Ctrl+Break) or pressing the pause button.
b. Reaching to break point.

Setting up the break points:

a. Go to the line where you need to mark the breakpoint.
b. Click with mouse on left corner margin of that line.
c. Another way is to press F9

Describe how to step through code in .NET.

Steps to step through the code in .NET:

a. Start the program in debug mode.
b. When the first breakpoint is reached then step through can be done in one of the two ways:
    i. Press F10 to move to next line.
    ii. Select debug menu and click on step over. This would step over the breakpoint to next level.
c. Other options are: “Step Into” and “Step Out”.

Describe the debugging windows available in .NET.

Debug->Windows:
Breakpoints: displays a list of all the breakpoints and where they are. Shows condition when that breakpoint will be hit if a condition exists and the Hit Count shows the number of times that breakpoint has been hit.

Output: Displays the status messages for various features in the IDE. It shows the output form a list of objects throughout debug mode of the application.

Immediate: This window allows the programmer to write code and verify values through programming while debugging the application. It helps in checking the values of objects/controls etc, and the conditions throughout the application in debug mode.

What are Debug and Trace classes? Explain how to use them to display error classes.

Both are used to help programmers find errors, occurring events and flow of code. In release mode, however, debug class is disabled and only Trace class is valid to trap things in a live application.
Both have assert functions to validate values.

Trace.WriteLine(variable value or comment).

Debug.WriteLine(variable value or comment).

We could create error handlers to trap unhandled exceptions trapped by Trace and Debug class.

Describe how to create Trace Listeners and log Trace output.

```
 [Conditional("TRACE")]
public static void InitializeUnhandledExceptionHandler()
{
   AppDomain.CurrentDomain.UnhandledException +=
          new UnhandledExceptionEventHandler(CutomExceptionHandler);
}
public static void CustomExceptionHandler(object sender,
                             UnhandledExceptionEventArgs args)
{
   Exception e=(Exception) args.ExceptionObject;
   Trace.WriteLine("Exception: "+e.Message+"\n"+e.GetType() +
               "\nStack Trace:\n"+e.StackTrace);
   MessageBox.Show(
      "An error has occurred:"+e.Message+"\nin: "+e.GetType(),
      "Fatal",
      MessageBoxButtons.OK,
      MessageBoxIcon.Stop,
      MessageBoxDefaultButton.Button1);
   Trace.Close();
   Process.GetCurrentProcess().Kill();
}

[Conditional("DEBUG")]
public static void TrapDebug(string str)
{
   Debug.WriteLine("Debug error: "+str);
}
```

What are Trace switches? Describe how to create and use Trace switches.

Trace switches allow us to filter, enable/disable the outputs through Trace. We can configure them through the config file. 3 types of trace switches:
BooleanSwitch: Enable/Disable trace statements.
TraceSwitch and SourceSwitch: used for trapping particular Trace levels.

```
BooleanSwitch dataSwitch =
  new BooleanSwitch("Comment", "module1");
TraceSwitch generalSwitch =
  new TraceSwitch("comment",
  "module1");
```

Explain how to configure Trace switches in the application's .config file.

switches are configured using the .config file

```
<system.diagnostics>
```

```
   <switches>
      <add name="MyTraceSwitch" value="1" />
      <add name="TraceSwitch2" value="1" />
   </switches>
</system.diagnostics>
```

both are on.

Explain how exceptions are handled by the common language runtime in .NET.

The CLR uses a technique generally referred to as a two-pass exception review process. What this means is that the CLR will process an exception in two passes. In the first pass, the CLR will determine if there is a handler for the exception. This is done by reviewing the entries in the SEH table; specifically it looks at the Try Offset and Try Length flags to see if the exception occurred within a guarded block, and if so, whether the Flags entry dictates that a handler exists for this type of occurrence. Let's assume that the CLR did find a handler during the first pass. At that point the CLR begins a second pass of the SEH table during which it will work through the execution phase of the exception management process. So we can divide the two passes into a discovery pass, in which we determine whether there is a handler in this method context to handle the exception; and an execution pass, in which we actually execute the handler and any special rules.
When code throws an exception, the CLR looks up the call stack looking for a catch filter to handle the exception. When it finds the relevant catch block, before executing the code, it will execute all code in all finally blocks - starting from the try block that threw the exception and stopping with the catch filter that matches the exception. when the CLR encounters an exception for a method it will use the descriptors in the SEH table to determine how to handle the exception, which code block is affected, and what handler should be invoked.

Describe the different types of user-authored controls in NET.

User authored controls are which not part of the .net framework library. It includes both custom controls and user controls.

a. Custom Controls: They look similar to ASP.NET controls. They can be created in one of the 3 ways:-
    a. Deriving a custom control from existing custom control.
    b. Making a composite custom control by combining 2 or more existing controls
    c. By creating a new control from scratch by deriving the control from its base class.

b. User Controls: enables a part of ASP.NET page to be reused. The reusable part is in form of a control with the extension .ascx. They look like to be a group of ASP.NET controls which can be used over and over again.

Explain with code sample how to create an inherited control.

Steps to create inherited Control:-
a. Create a new project.
b. Add a custom control to the project.
c. Change the name of the class you need to inherit the control from the base class. E.g. inherit the class from System.Windows.Forms.Button if the control s to be inherited from a button class.

d. Implement the control with custom properties and featured needed by the control.
e. Override the OnPaint method if the control's appearance needs to be changed.
f. Save the build the control
g. Reference you control into another or the same project and use the control.

Explain with code sample how to create a user control.

Steps to create a User control:

a. Select a project
b. Right click and add a new item (User Control - .ascx) to the selected project.
c. Add @Control Directive
d. Add all the controls that you want to be displayed on the User control as a part of one or more web pages.
e. Write the code for all the tasks to be performed by the user control.
f. Create accessor methods for the outside world accessing this user control.

Using the User control:

a. Register the control on the webpage it needs to be used by putting @Register directive.
b. Specify the following attributes to the register directive:
    a. TagPrefix: defines the namespace in which the control would reside
    b. TagName: defines the name with which control is referred
    c. Src: Path of where the control is kept.
c. Control is then used on the page using the following code:
   <TagPrefix:TagName />

Explain with code sample how to create a custom control.

Steps to create a custom control:

a. Create a new project.
b. Add a custom control to the project.
c. Change the name of the class you need to inherit the control from the base class. E.g. inherit the class from System.Windows.Forms.Button if the control s to be inherited from a button class.
d. Implement the control with custom properties and featured needed by the control.
e. Override the OnPaint method if the control's appearance needs to be changed.
f. Save the build the control
g. Reference you control into another or the same project and use the control.

Describe the .NET Framework architecture.

.Net framework has two components:

1. .Net framework class library
2. Common language runtime.

FCL facilitates the types through CTS which are common to all the supported languages.
The CLS ensures that all languages are interoperable. This ensures that all code is managed .i.e. code which is converted to MSIL.
The CLR has the class loader that load the MSIL code of an application into runtime, which is then converted into native code by the JIT complier. The CLR manages code and provide services such as memory management, threading, remoting, type safety, security, Exception handling etc.

What is the managed execution process?

Managed execution process is a process where CLR executes the managed code. The steps involved in this process are:

a. Choosing the right compiler
b. Compiling the code to MSIL. This also generates the required metadata.
c. Compile the MSIL ode to native machine code.
d. Executing the code with the variety of services available.

What are assemblies? Describe the types of assemblies.

Assembly is a compiled output of program which are used for easy deployment of an application. They are executables in the form of exe or dll. It also is a collection of resources that were used while building the application and is responsible for all the logical functioning.
Types of assemblies:

a. Private Assemblies: are accessible by a single application. They reside within the application folder and are unique by name. They can be directly used by copying and pasting them to the bin folder.

b. Shared Assemblies: are shared between multiple applications to ensure reusability. They are placed in GAC.

c. Satellite Assemblies: are the assemblies to provide the support for multiple languages based on different cultures. These are kept in different modules based on the different categories available.

Explain the role of assemblies in .NET.

Assemblies are main building blocks. An assembly maybe defined as a unit of deployment. A single assembly is a collection of types, and resources. The CLR does not understand any types that are outside assemblies. The CLR executes the code in assemblies as they contain MSIL code. They define type, version and security boundaries.

Assemblies in .Net are a solution to the Dll hell problem as one can use different versions of same assembly in different applications at the same time. To make a shared assembly, we need to register it with GAC where as private assemblies reside in applications directory.

What are windows services? How are they differ from other .NET application?

Windows services are a way to create continuously running applications in the background. They don't interfere with other applications and can be run whenever a machine starts. They can be paused if and when needed and quietly run in the background without the need of any user intervention. Windows services can be configured to run under specific user accounts. They run under their own windows sessions and are ideal for tasks that need to be performed periodically or for monitoring requirements.
Main difference between windows services and other .Net applications lies in the fact that they run in their own windows session without any user intervention in the background.