What is Microsoft .Net Framework?

The .Net framework is a software development platform developed by Microsoft. The framework was meant to create applications, which would run on the Windows Platform. The first version of the .Net framework was released in the year 2002.

The version was called .Net framework 1.0. The .Net framework has come a long way since then, and the current version is 4.7.1.

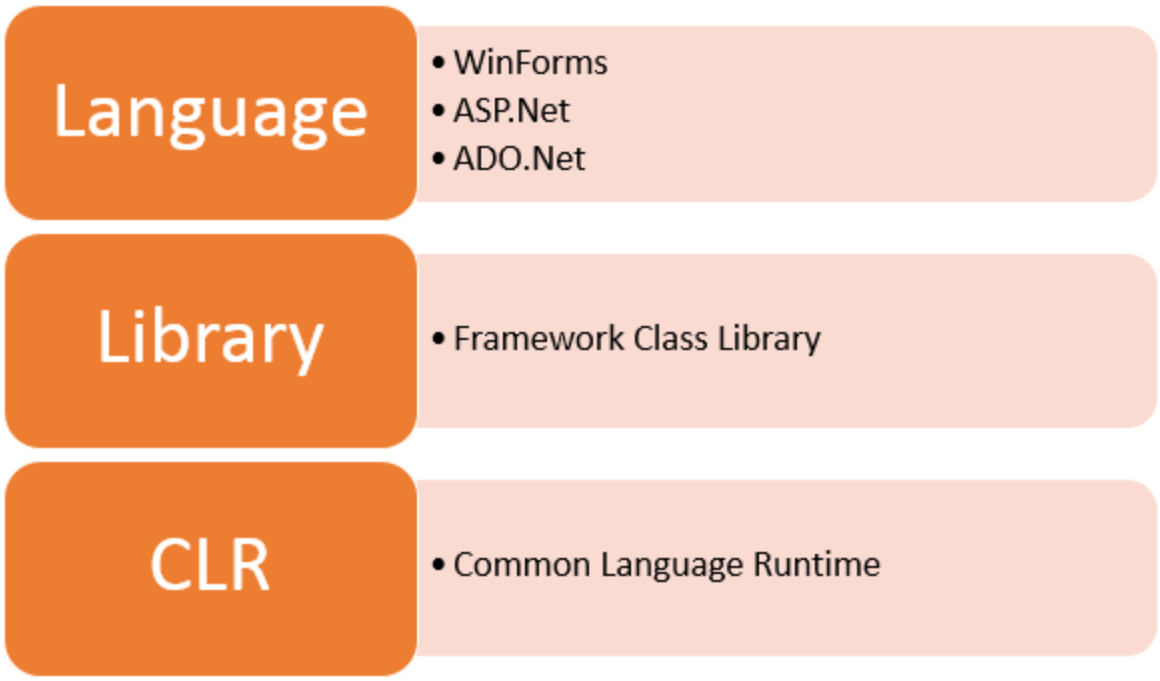
The .Net framework can be used to create both - **Form-based** and **Web-based** applications. Web services can also be developed using the .Net framework.

The framework also supports various programming languages such as Visual Basic and C#. So developers can choose and select the language to develop the required application. In this chapter, you will learn some basics of the .Net framework.

In this tutorial, you will learn-

.Net Framework Architecture

The basic architecture of the .Net framework is as shown below.



.net framework architecture diagram

.NET Components

The architecture of the .Net framework is based on the following key components;

1. Common Language Runtime

The "Common Language Infrastructure" or CLI is a platform on which the .Net programs are executed.

The CLI has the following key features:

- Exception Handling - Exceptions are errors which occur when the application is executed.

  Examples of exceptions are:

  - If an application tries to open a file on the local machine, but the file is not present.

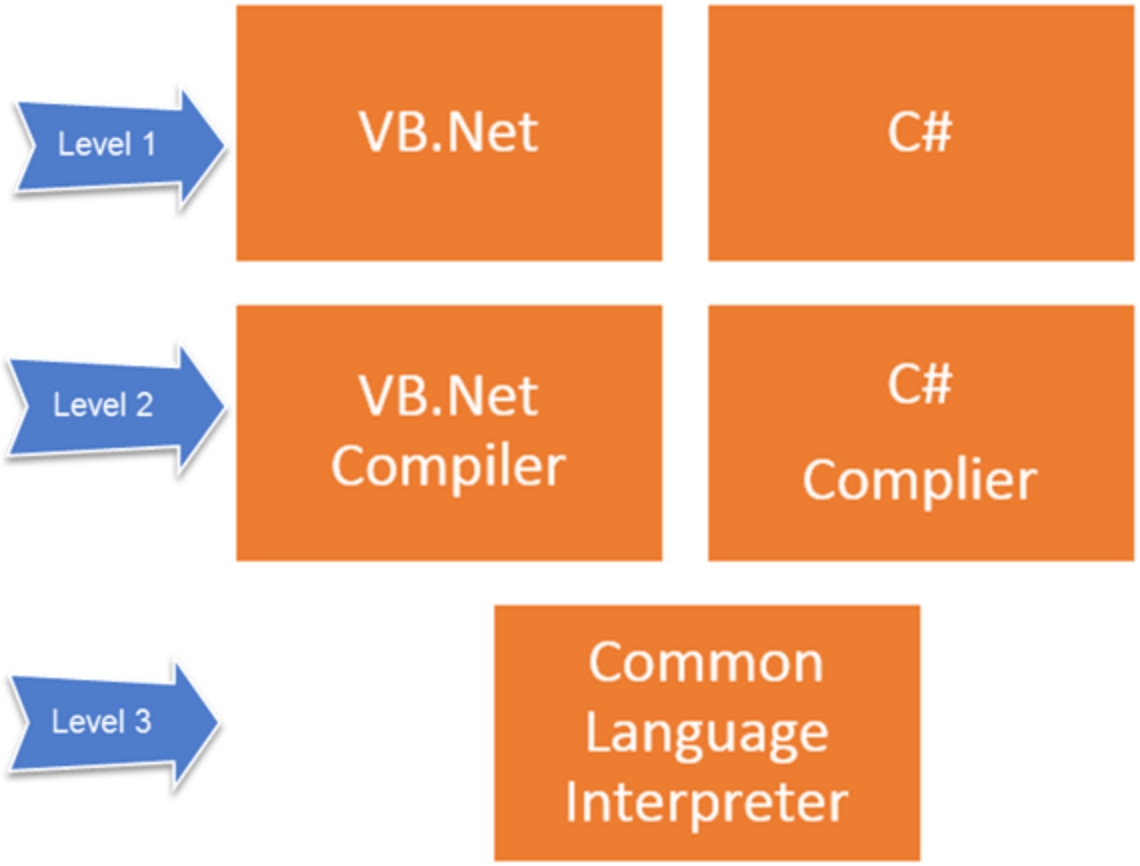
    - If the application tries to fetch some records from a database, but the connection to the database is not valid.
- Garbage Collection - Garbage collection is the process of removing unwanted resources when they are no longer required.

  Examples of garbage collection are

    - A File handle which is no longer required. If the application has finished all operations on a file, then the file handle may no longer be required.
    - The database connection is no longer required. If the application has finished all operations on a database, then the database connection may no longer be required.
- Working with Various programming languages –

As noted in an earlier section, a developer can develop an application in a variety of .Net programming languages.

1. Language - The first level is the programming language itself, the most common ones are VB.Net and C#.
2. Compiler – There is a compiler which will be separate for each programming language. So underlying the VB.Net language, there will be a separate VB.Net compiler. Similarly, for C#, you will have another compiler.
3. Common Language Interpreter – This is the final layer in .Net which would be used to run a .net program developed in any programming language. So the subsequent compiler will send the program to the CLI layer to run the .Net application.



2. Class Library

The .NET Framework includes a set of standard class libraries. A class library is a collection of methods and functions that can be used for the core purpose.

For example, there is a class library with methods to handle all file-level operations. So there is a method which can be used to read the text from a file. Similarly, there is a method to write text to a file.

Most of the methods are split into either the System.* or Microsoft.* namespaces. (The asterisk * just means a reference to all of the methods that fall under the System or Microsoft namespace)

A namespace is a logical separation of methods. We will learn these namespaces more in detail in the subsequent chapters.

3. Languages

The types of applications that can be built in the .Net framework is classified broadly into the following categories.

- WinForms – This is used for developing Forms-based applications, which would run on an end user machine. Notepad is an example of a client-based application.
- ASP.Net – This is used for developing web-based applications, which are made to run on any browser such as Internet Explorer, Chrome or Firefox.
    - The Web application would be processed on a server, which would have Internet Information Services Installed.
    - Internet Information Services or IIS is a Microsoft component which is used to execute anAsp.Net application.
    - The result of the execution is then sent to the client machines, and the output is shown in the browser.
- ADO.Net – This technology is used to develop applications to interact with Databases such as Oracle or Microsoft SQL Server.

Microsoft always ensures that .Net frameworks are in compliance with all the supported Windows operating systems.

.Net Framework Design Principle

The following design principles of the .Net framework is what makes it very relevant to create .Net based applications.

1. Interoperability - The .Net framework provides a lot of backward support. Suppose if you had an application built on an older version of the .Net framework, say 2.0. And if you tried to run the same application on a machine which had the higher version of the .Net framework, say 3.5. The application would still work. This is because with every release, Microsoft ensures that older framework versions gel well with the latest version.
2. Portability- Applications built on the .Net framework can be made to work on any Windows platform. And now in recent times, Microsoft is also envisioning to make Microsoft products work on other platforms, such as iOS and Linux.
3. Security - The .NET Framework has a good security mechanism. The inbuilt security mechanism helps in both validation and verification of applications. Every application can explicitly define their security mechanism. Each security mechanism is used to grant the user access to the code or to the running program.
4. Memory management - The Common Language runtime does all the work or memory management. The .Net framework has all the capability to see

those resources, which are not used by a running program. It would then release those resources accordingly. This is done via a program called the "Garbage Collector" which runs as part of the .Net framework.

The garbage collector runs at regular intervals and keeps on checking which system resources are not utilized, and frees them accordingly.

5. Simplified deployment - The .Net framework also have tools, which can be used to package applications built on the .Net framework. These packages can then be distributed to client machines. The packages would then automatically install the application.

MSIL:

MSIL stands for Microsoft Intermediate Language. We can call it as Intermediate Language (IL) or Common Intermediate Language (CIL). During the compile time , the compiler convert the source code into Microsoft Intermediate Language (MSIL) .Microsoft Intermediate Language (MSIL) is a CPU-independent set of instructions that can be efficiently converted to the native code.

Common Language Runtime (CLR)

The Common Language Runtime (CLR) is an Execution Environment . It works as a layer between Operating Systems and the applications written in .Net languages that conforms to the Common Language Specification (CLS). The main function of Common Language Runtime (CLR) is to convert the Managed Code into native code and then execute the Program. The Managed Code compiled only when it needed, that is it converts the appropriate instructions when each function is called . The Common Language Runtime (CLR) 's Just In Time (JIT) compilation converts Intermediate Language (MSIL) to native code on demand at application run time.

During the execution of the program ,the Common Language Runtime (CLR) manages memory, Thread execution, Garbage Collection (GC) , Exception Handling, Common Type System (CTS), code safety verifications, and other system services. The CLR (Common Language Runtime) defines the Common Type System (CTS), which is a standard type system used by all .Net languages . That means all .NET programming languages uses the same representation for common Data Types , so Common Language Runtime (CLR) is a language-independent runtime environment . The Common Language Runtime (CLR) environment is also referred to as a managed environment, because during the execution of a program it also controls the interaction with the Operating System. In the coming section you can see what are the main functions of Common Language Runtime (CLR).

What is Assembly

It is the unit of deployment for the Microsoft .NET framework and takes the form of an executable (.exe) file or dynamic-link library (DLL). Assembly is logical unit of code. All the .NET assemblies contain the definition of types, versioning information for the type, meta-data, and manifest. Assemblies are a collection of Single-File and Multiple-File. An assembly is the primary unit of deployment, security, and version control in the .NET Framework. The .NET Framework (version 2.0) includes 51 assemblies.

Before we can use a class contained in an assembly in our application, we must add a reference to the assembly.

- ☐ mscorlib.dll
- ☐ System.dll
- ☐ System.Configuration.dll
- ☐ System.Web.dll
- ☐ System.Data.dll
- ☐ System.Web.Services.dll
- ☐ System.Xml.dll

There are two types of assemblies:

- **Private:** A private assembly can be used by only a single application.
- **Shared:** A shared assembly can be used by all applications located on the same server.(also called

strong named assemblies) are copied to a single location (usually the Global assembly cache) GAC.

Advantages of using assemblies

1.It increase the overall Performance
2.It can use the N-tier Architecture
3.Used for business Logic
4.Better user interaction
5.Easy to manage the code

dynamic link library

A dynamic link library (DLL) is a collection of small programs that can be loaded when needed by larger programs and used at the same time. The small program lets the larger program communicate with a specific device, such as a printer or scanner. It is often packaged as a DLL program, which is usually referred to as a DLL file. DLL files that support specific device operation are known as device drivers.

A DLL file is often given a ".dll" file name suffix. DLL files are dynamically linked with the program that uses them during program execution rather than being compiled into the main program.

The advantage of DLL files is space is saved in random access memory (RAM) because the files don't get loaded into RAM together with the main program. When a DLL file is needed, it is loaded and run. For example, as long as a user is editing a document in Microsoft Word, the printer DLL file does not need to be loaded into RAM. If

the user decides to print the document, the Word application causes the printer DLL file to be loaded and run.

A program is separated into modules when using a DLL. With modularized components, a program can be sold by module, have faster load times and be updated without altering other parts of the program. DLLs help operating systems and programs run faster, use memory efficiently and take up less disk space.

Namespaces

Namespaces are used to organize the classes. It helps to control the scope of methods and classes in larger **.Net** programming projects. In simpler words you can say that it provides a way to keep one set of names(like class names) different from other sets of names. The biggest advantage of using namespace is that the class names which are declared in one namespace will not clash with the same class names declared in another namespace. It is also referred as **named group of**

classes having common features. The members of a namespace can