UNIT-1 Overview of .NET Framework

What is .NET?

It is a framework.

.NET is "an open source developer platform, created by Microsoft, for building many different types of applications. You can write .NET apps in C#, F#, Visual C++, or Visual Basic."

What is framework?

A framework is a set of common prefabricated software building blocks that programmer use, extend or customize for specific computing solutions.

What is .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 4.8.1

The .Net framework can be used to create both - **Form-based (Desktop), Web-based** and **Mobile** 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.

Features of .NET

1. Rich Set Of Classes

- In c and c++ programmers uses header files like " stdio.h " , " iostream.h ", etc.
- Visual Studio.NET contains hundreds of classes and namespaces that providing variety of functionality in applications.
- For example, to work with database we need to include System. Data namespace

2. Object Oriented Programming System

- Visual Studio.NET provides fully object oriented environment.
- Visual Studio.NETSupports OOPs concepts

3. In-Built Memory Management

- Visual Studio.NET supports handling of memory on its own.
- The garbage collector (GC) takes responsibility for freeing up unused objects at regular interval

4. Multi-Language And Multi-Device Support

- Visual Studio.NET supports multiple languages like c++, C# ,vb ,J#, F#, Jscript, Etc.
- The beauty of multi-language supports lies in the fact that even though the syntax of each language is different, the basic environment for developing software is same.
- Visual Studio.NET supports multi-device
- We can create mobile or PDA or etc.

5. Faster and Easy Development For Web Applications

- ASP.NET is used for developing dynamic and database related web applications.
- ASP.NET contains rich and faster development controls for web application.

6. Xml Support

- Visual Studio.NET supports for writing, manipulating and transforming XML documents.

7. **Ease Of Development And Configuration**

- Deploying windows application especially that use COM components have always been a tedious task. Since, .NET does not require any registration as such, much of the deployment is simplified.

What is VB.NET ?

Visual Basic .NET (VB.NET) is a Microsoft object-oriented programming (OOP) language. It evolved from Visual Basic 6 (VB6) to meet an increasing need for easy web-services and web development.

"Visual" part refer to the way , which is used to create the GUI (GRAPHICAL USER INTERFACE) base applications

The **"Basic"** stands for **Beginners All-Purpose Symbolic Instruction Code.**

VB.NET

VB.NET is also known as **Visual Basic.NET**. It stands for **Visual Basic .Network Enabled Technologies.** It is a simple, high-level, object-oriented programming language developed by Microsoft in 2002. It is a successor of Visual Basic 6.0, which is implemented on the

Microsoft .NET Framework. With this language, you can develop a fully object-oriented application that is similar to an application created through another language such as C++, Java, or C#.

Feature of VB.NET

- Inheritance (object-oriented language)
- Delegates and events
- Parameterized constructors
- Method overloading/overriding
- Type-safe
- Delegates and events

Visual Basic

Visual Basic (VB) is a programming language developed by Microsoft in 1992. The purpose of this language is to develop an application that can run on different versions of the Windows operating system. A Visual Basic evolved from Basic Language; Basic language is easier to read than other languages. The final version of Visual Basic was released in 1998. Microsoft then launched a Visual Basic DotNet ('VB.NET') language, which is much better than Visual Basic in all aspects such as performance, reliability, working environment, easy to build, and debugging an application.

Features of Visual Basic

- User Interface design
- Rapid Application Development
- Using this language, you can use internet or intranet services in your application.
- It has powerful database access tools, by which you can easily develop front end applications.
- It also supports ActiveX technology, in which you can access the features of other application in system application. For example: Microsoft Word, Microsoft Excel, etc.

VB.NET Features

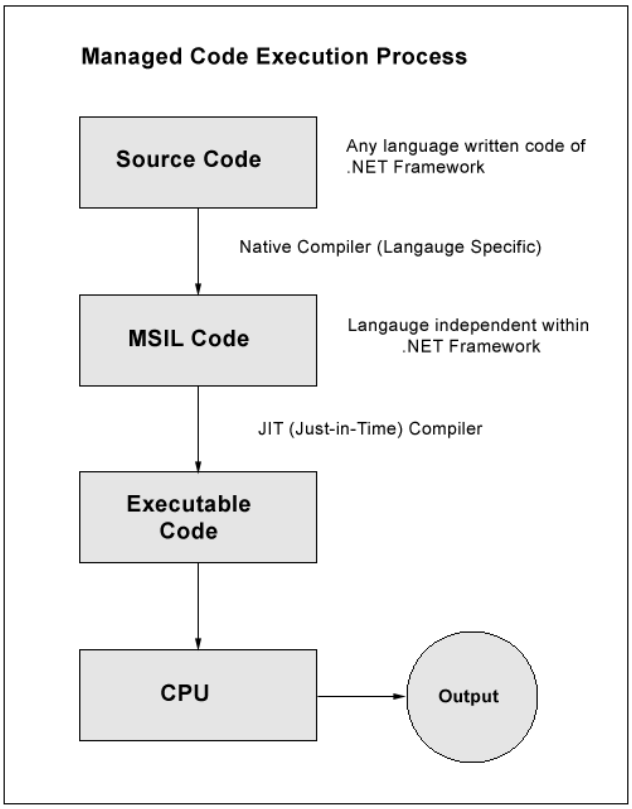
As we know, it is a high-level programming language with many features to develop a secure and robust application. These are the following features that make it the most popular programming language.

- It is an object-oriented programming language that follows various oops concepts such as abstraction, encapsulation, inheritance, and many more. It means that everything in VB.NET programming will be treated as an object.
- This language is used to design user interfaces for window, mobile, and web-based applications.
- It supports a rapid application development tool kit. In which a developer does not need to write all the codes as it can get various code automatically from its libraries. For example, when we create a form in Visual basic.net, it automatically calls events of various form in that class.
- It is not a case sensitive language like other languages such as C++, java, etc.
- It supports Boolean condition for decision making in programming.
- It also supports the multithreading concept, in which you can do multiple tasks at the same time.
- It provides simple events management in .NET application.
- A Window Form enables us to inherit all existing functionality of form that can be used to create a new form. So, in this way, it reduced the code complexity.
- It uses an external object as a **reference** that can be used in a VB.NET application.
- Automatic initialized a garbage collection.
- It follows a structured and extensible programming language for error detection and recovery.
- Conditional compilation and easy to use generic classes.
- It is useful to develop web, window, and mobile applications.
-

Difference Between VB. NET and Visual Basic

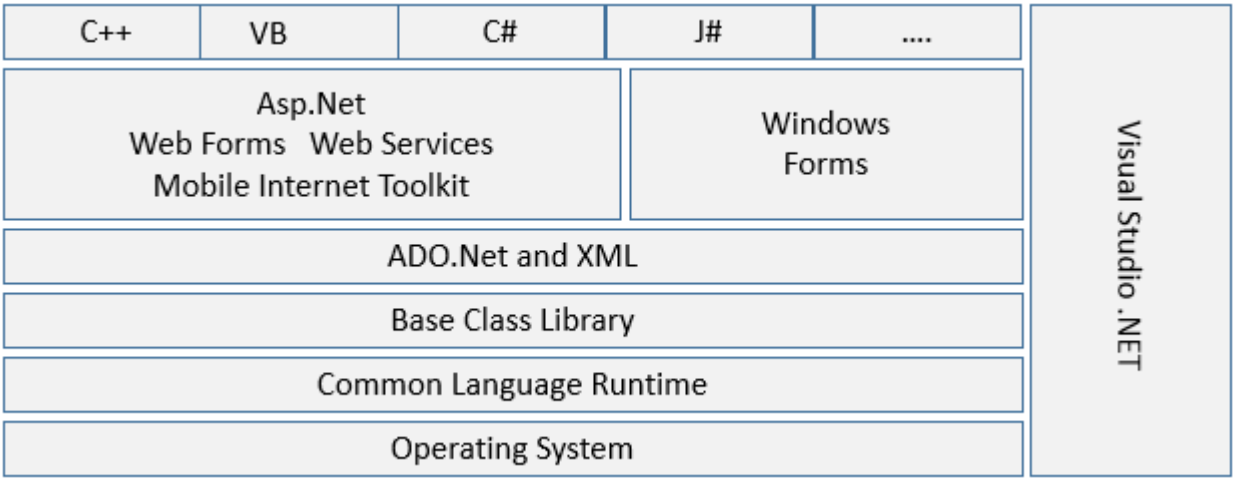
VB .NET	Visual Basic
It stands for Visual Basic. Network Enables Technology. It is also developed by Microsoft, and this language was based on the .Net Framework. Furthermore, it is specially designed for VB developers.	It is a programming language developed by Microsoft for the fastest development of a window-based operating system as well as applications.
It is a modern, fully object-oriented language that replaced VB6.	VB is the predecessor of VB.NET and was not an object-oriented language. So, it is not actively maintained.
A VB.NET uses the Common Language Runtime	Visual Basic uses the VB-Runtime

(CLR) component of .Net Framework at runtime. It has better features and design implementation as compared to VB-Runtime.	environment.
It is a compiled language	It is an Interpreter based language
It does not support backward compatibility.	It supports backward compatibility.
It is a type-safe language.	It is not a type-safe language.
In VB.NET, data is handled using the ADO.net protocol.	Data Connectivity and handling are done through DAO, RDO, and ADO (ActiveX Data Object) protocol,
Object does not support default property.	The Object support default property of virtual basic.
In the VB.Net parameter are passed by a default value.	In VB, most of the parameters are passed by reference.
A Multithreaded application can be developed in VB.NET.	It does not support the multithread concept



The .NET Framework:

The **Microsoft .Net Framework** is a platform that provides tools and technologies you need to build Networked Applications as well as Distributed Web Services and Web Applications. The .Net Framework provides the necessary **compile time** and **run-time** foundation to build and run any language that conforms to the Common Language Specification (CLS).



The main two components of .Net Framework are **Common Language Runtime** (CLR) and .Net **Framework Class Library** (FCL).

1. CLR(Common Language Runtime)

The **Common Language Runtime (CLR)** is at the heart of the .NET Framework.

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

2. Microsoft Intermediate Language - 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) . During the runtime the Common Language Runtime (CLR)'s Just In Time (JIT) compiler converts the Microsoft Intermediate Language (MSIL) code into native code to the Operating System.

When a compiler produces Microsoft Intermediate Language (MSIL), it also produces Metadata. The Microsoft Intermediate Language (MSIL) and Metadata are contained in a portable executable (PE) file . Microsoft Intermediate Language (MSIL) includes instructions for loading, storing, initializing, and calling methods on objects, as well as instructions for arithmetic and logical operations, control flow, direct memory access, exception handling, and other operations

3.Net Framework Class Libraries (FCL) (Class Library)

This is also called as Base Class Library and it is common for all types of applications The .NET framework provides a set of base class libraries which provide functions and features

which can be used with any programming language which implements .NET, such as Visual Basic, C# (or course), Visual C++, etc.

The base class library contains standard programming features such as Collections, XML, DataType definitions, IO (for reading and writing to files), Reflection and Globalization to name a few. All of which are contained in the System namespace. As well, it contain some non-standard features such as LINQ, ADO.NET (for database interactions), drawing capabilities, forms and web support.

4.Data and XML

It is basically used for communication
It is set of computer software components
It is use for access data and data service
It's a part of BCL
Consist of two parts: 1. Data Provider 2. Data Set

Access relational databases
Disconnected data model
Work with XML

Web Service and User Interface

Create application's front-end – Web-based user interface, Windows GUI, Web service,etc…

- **Windows Forms** is used to create GUI for windows desktop application. Provide an integrated and unified way of developing GUI. It has a rich variety of windows control and user interface support.
  Ex. Textbox, label, button, etc…

- **Web Forms** provides tools for web application.it is a part of ASP.NET. User interfaces created with Web Forms also have built-in browser independence and state management.

- **Web Services** are small unit of code. They are design to handle limited set of task. They used XML based communicating protocol to establish connection between applications. They are independent from operating system and programming lnguages.it connects people,system and device.

  XML: standard for storing,carrying and exchanging data.

5. Common Language Specification - CLS

Common Language Specification (CLS) is a set of basic language features that .Net Languages needed to develop Applications and Services , which are compatible with the .Net Framework. When there is a situation to communicate Objects written in different .Net Complaint languages , those objects must expose the features that are common to all the languages . Common Language Specification (CLS) ensures complete interoperability among applications, regardless of the language used to create the application.

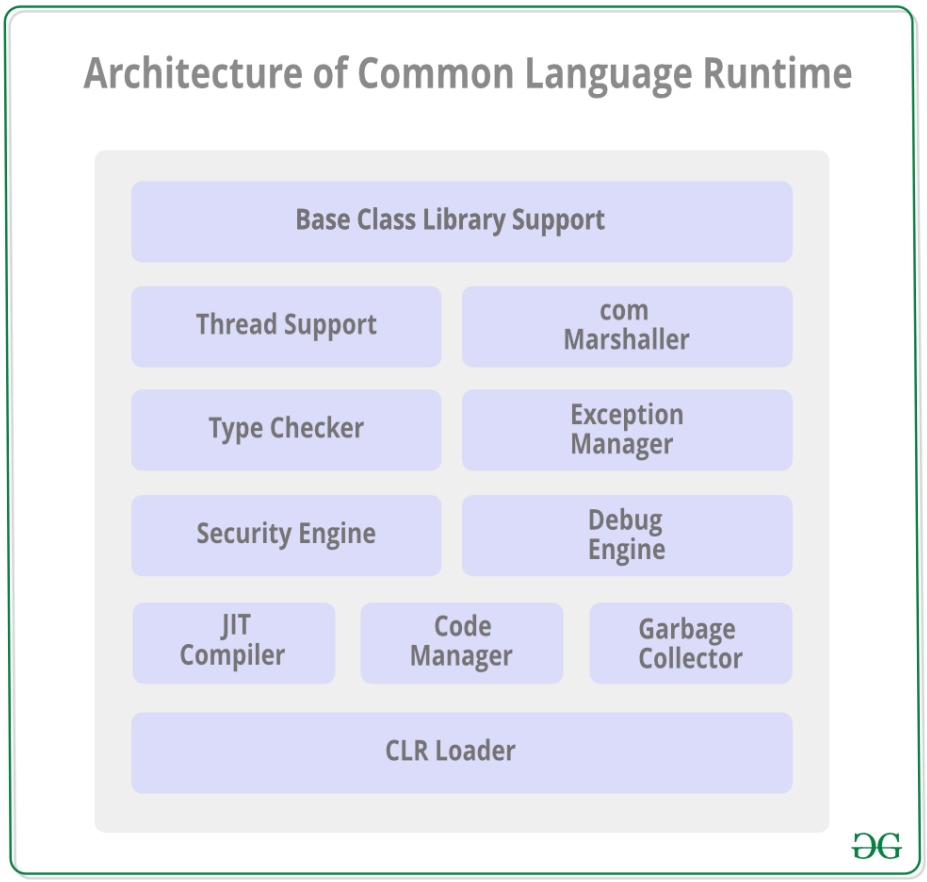
Common Language Specification (CLS) defines a subset of Common Type System (CTS) . Common Type System (CTS) describes a set of types that can use different .Net languages have in common , which ensure that objects written in different languages can interact with each other.

6. Common Type System - CTS

Common Type System (CTS) describes a set of types that can be used in different .Net languages in common . That is , the Common Type System (CTS) ensure that objects written in different .Net languages can interact with each other. For Communicating between programs written in any .NET complaint language, the types have to be compatible on the basic level .

These types can be Value Types or Reference Types . The Value Types are passed by values and stored in the stack. The Reference Types are passed by references and stored in the heap.

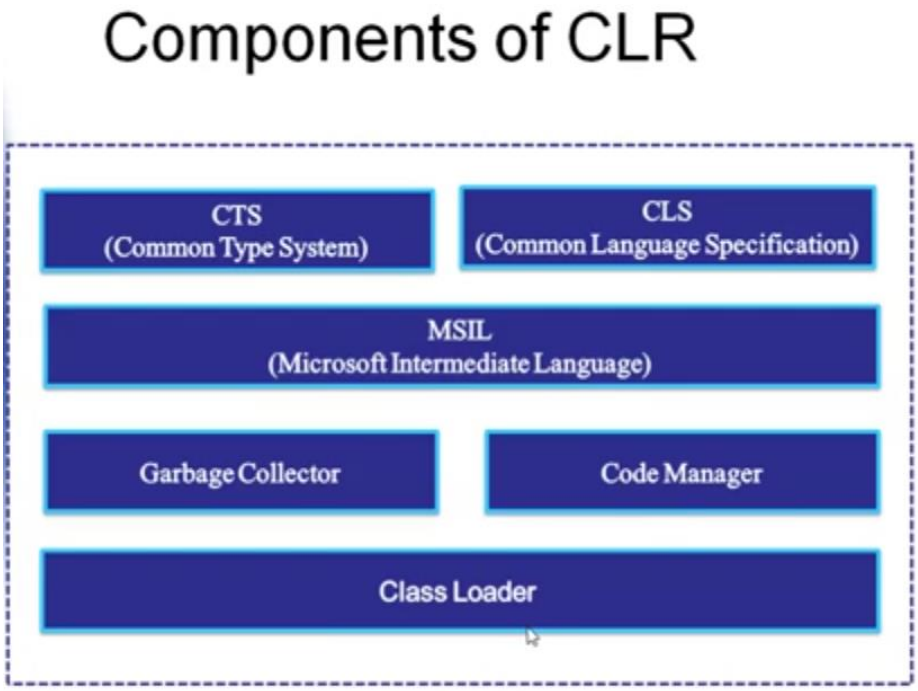
Architecture of Common Language Runtime (CLR)



- **Base Class Library Support:** The Common Language Runtime provides support for the base class library. The BCL contains multiple libraries that provide various features such as *Collections*, *I/O*, *XML*, *DataType definitions*, etc. for the multiple *.NET* programming languages.
- **Thread Support:** The CLR provides thread support for managing the parallel execution of multiple threads. The *System.Threading class* is used as the base class for this.
- **COM Marshaller:** Communication with the COM (Component Object Model) component in the .NET application is provided using the COM marshaller. This provides the COM interoperability support.
- **Type Checker:** Type safety is provided by the type checker by using the Common Type System (CTS) and the Common Language Specification (CLS) that are provided in the CLR to verify the types that are used in an application.
- **Exception Manager:** The exception manager in the CLR handles the exceptions regardless of the *.NET Language* that created them. For a particular application, the catch block of the exceptions are executed in case they occur and if there is no catch block then the application is terminated.
- **Security Engine:** The security engine in the CLR handles the security permissions at various levels such as the code level, folder level, and machine level. This is done using the various tools that are provided in the *.NET* framework.

- **Debug Engine:** An application can be debugged during the run-time using the debug engine. There are various ICorDebug interfaces that are used to track the managed code of the application that is being debugged.
- **JIT Compiler:** The JIT compiler in the CLR converts the Microsoft Intermediate Language (MSIL) into the machine code that is specific to the computer environment that the JIT compiler runs on. The compiled MSIL is stored so that it is available for subsequent calls if required.
- **Code Manager:** The code manager in CLR manages the code developed in the .NET framework i.e. the managed code. The managed code is converted to intermediate language by a language-specific compiler and then the intermediate language is converted into the machine code by the Just-In-Time (JIT) compiler.
- **Garbage Collector:** Automatic memory management is made possible using the garbage collector in CLR. The garbage collector automatically releases the memory space after it is no longer required so that it can be reallocated.
- **CLR Loader:** Various modules, resources, assemblies, etc. are loaded by the CLR loader. Also, this loader loads the modules on demand if they are actually required so that the program initialization time is faster and the resources consumed are lesser.

❖ COMPONENTS OF CLR:



1.Common Type System (CTS)

It describes set of data types that can be used in different .Net languages in common. (i.e), CTS ensures that objects written in different .Net languages can interact with each other.

For Communicating between programs written in any .NET complaint language, the types have to be compatible on the basic level.

The common type system supports two general categories of types:

Value types:

Passed by value and data store in stack. In built data type
Ex. Dim a as integer

Reference types:

Passed by reference and data store in heap.
Ex. Dim obj as new oledbconnection

2.Common Language Specification (CLS)

It is a sub set of CTS and it specifies a set of rules that needs to be adhered or satisfied by all language compilers targeting CLR. It helps in cross language inheritance and cross language debugging.

Common language specification Rules:

It describes the minimal and complete set of features to produce code that can be hosted by CLR. It ensures that products of compilers will work properly in .NET environment.

Sample Rules:

1. Representation of text strings
2. Internal representation of enumerations
3. Definition of static members and this is a subset of the CTS which all .NET languages are expected to support.
4. Microsoft has defined CLS which are nothing but guidelines that language to follow so that it can communicate with other .NET languages in a seamless manner.

3. MSIL (Microsoft Intermidiate Language)

- 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.

- MSIL includes instructions for loading, storing, initializing, and calling methods on objects, as well as instructions for arithmetic and logical operations, control flow, direct memory access, exception handling, and other operations.

- Combined with metadata and the common type system ,MSIL allows for true cross-language integration.
- **Use**: You can run your program wherever without copy the source code. You can just copy the IL file from your machine (Dll or .exe). It can run everywhere.

4. Garbage Collection

It is the mechanism that allows the computer to detect when an object can no longer be accessed

It automatically free up the memory used by the object

One of the advantages of the CLR is automatic memory management that uses the garbage collection mechanism.

It manages the allocation and release of memory for an application.

We don't have to write code to perform memory management tasks when you developed managed application.

5. Code Manager

Code manager invokes class loader for execution of .net application

.NET supports two kind of coding

1. Managed Code
2. Unmanaged Code

Managed Code

The resource, which is within your application domain is, managed code. The resources that are within domain are faster.

The code, which is developed in .NET framework, is known as managed code. This code is directly executed by CLR with help of managed code execution. Any language that is written in .NET Framework is managed code.

Unmanaged Code

The code, which is developed outside .NET, Framework is known as unmanaged code.

Applications that do not run under the control of the CLR are said to be unmanaged, and certain languages such as C++ can be used to write such applications, which, for example, access low - level functions of the operating system. Background compatibility with code of VB, ASP and COM are examples of unmanaged code.

Unmanaged code is executed with help of wrapper classes.

Wrapper classes are of two types: **CCW (COM Callable Wrapper)** and **RCW (Runtime Callable Wrapper).**

Services provided by CLR

1. **Code loading and execution**

   It is the process followed by the CLR from loading MSIL, converting it into machine code, memory management. Managed code execution also handles JIT compilation, handles exceptions, and confirms type safety
   and security issues.

2. **Application memory isolation**

   The CLR manages memory for managed code
   a. All allocations of objects and buffers made from a *Managed Heap*
   b. Unused objects and buffers are cleaned up automatically through *Garbage Collection*
   Some of the worst bugs in software development are not possible with managed code
   c. Leaked memory or objects
   d. References to freed or non-existent objects

e. Reading of un-initialized variables

3. **Verification of type safety**

   It referred as the strongly typed feature. It allows to access memory only in authorized ways makes sure that it is within the bounds. It ensures that code cannot perform operations that are invalid for an object.

4. **Conversion of IL to native code**

   NET languages are compiled to an Intermediate Language (IL).IL is also known as MSIL or CIL
   CLR compiles IL in just-in-time (JIT) manner – each function is compiled just before execution
   into the native binary code of the machine the code is executed on.

5. **Access to metadata**

   A set of data that describes and gives information about other data or data about data is known as metadata. Clr can access the metadata.

Assemblies

It is the primary building block of .net framework.
It consists of DLL or EXE (executable) file
Smallest deployable unit in the CLR
Have unique version number
No version conflicts (known as DLL hell)
.NET assemblies contain the definition of types, versioning information for the type, meta-data, and manifest.

There are two kinds of assemblies in .NET;

1. Private

   Private assemblies are used only by single application, and is stored in that application's install directory (or a subdirectory therein.).Their names need to be unique within the application that uses it.

2. Shared

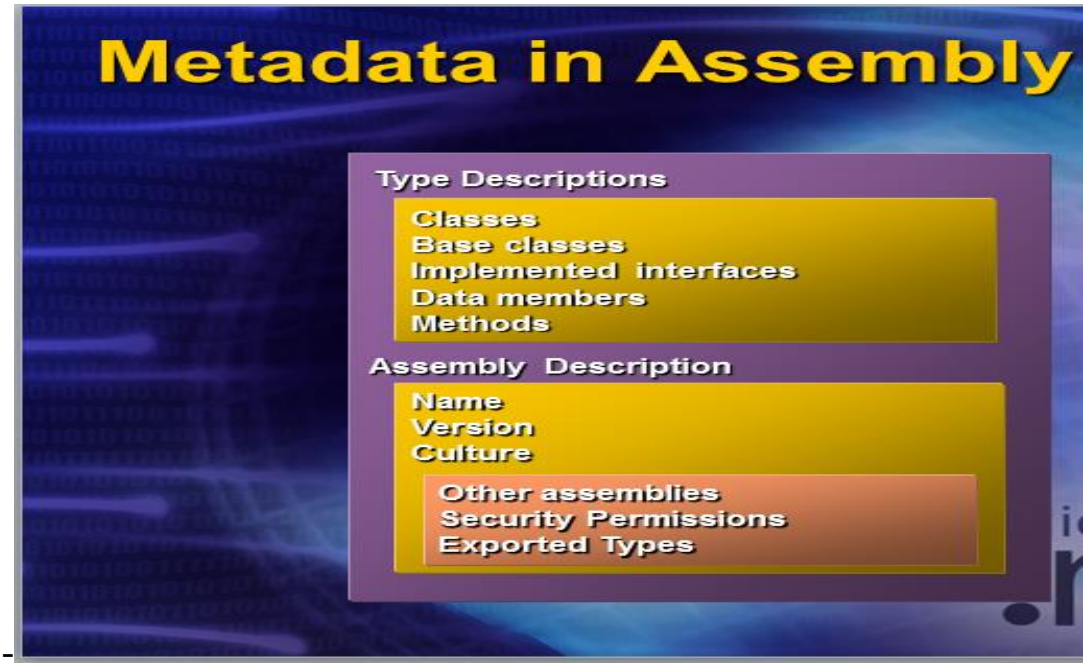
Shared assemblies are one that can be referenced by more than one application.in order to share an assembly; the assembly must be explicitly built for this purpose by giving it a cryptographically strong name

DLL hell

Before some time, if we install an application then dll of that application get stored in the registry, then if we install other application that has same name .dll that means previously installed .dll get overwrite by the same name new .dll. Ok for newly installed application but previously installed application cant get execute further. This is big problem in context of version of same application. This is Dell-Hell problem.
OR
Dll Hell refers to a set of problems caused when multiple applications attempt to share a common component like a dynamic link library (DLL).The reason for this issue was that the version information about the different components of an application was not recorded by the system.

Metadata

- Metadata stored within the assembly
- .net records the information about compiled classes as metadata
- It means data about data
- A .net language compiler will generate the metadata and store this in the assembly
- The .net platform program compiled into .NET PE (portable executable) files
- The header section of every .NET PE file contains a special new section for metadata
- Metadata is nothing but a description of every namespace, class, method, property, etc.
- The CLR uses the metadata to,
    - Locate classes
    - Load classes
    - Generate native class
    - Provide security



ILDASM

It's a one kind of software which used to view basic information about the project

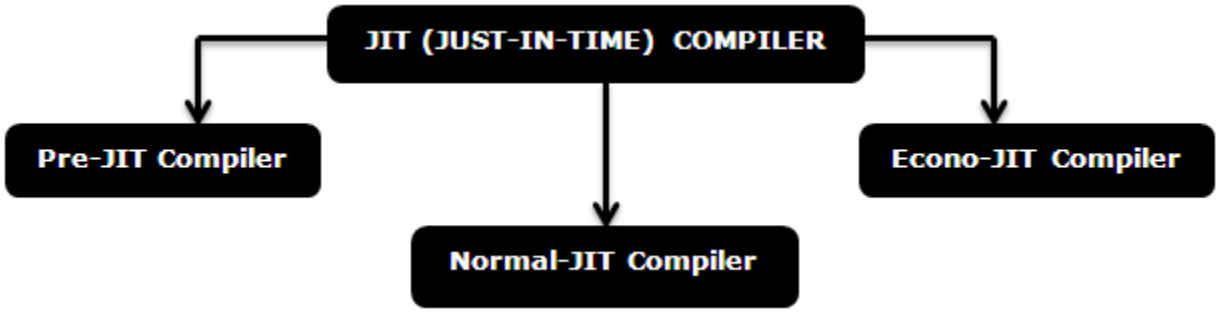
ILDASM.exe can opens any .NET framework .exe or .dll assembly

It displays the MSIL code, namespaces, methods, fields, events, interfaces, etc,

Ildasm.exe is used to examine native .net framework assemblies.

Just - In- Time (JIT)Compiler

It's a smart compiler. JIT does not compile whole program each time and every time . it compiles only that portion in which functions are called during that time if native code is already present then that data will not again compiled. If changes are made then it is possible that it will generates MSIL to native.



- **Pre-JIT COMPILER**

  Pre-JIT compiles complete source code into native code in a single compilation cycle. This is done at the time of deployment of the application.

- **Econo-JIT COMPILER:**

  Econo-JIT compiles only those methods that are called at runtime. However, these compiled methods are

  removed when they are not required.

- **Normal-JITCOMPILER:**

  Normal-JIT compiles only those methods that are called at runtime. These methods are compiled the first time they are called, and then they are stored in cache. When the same methods are called again, the compiled code from cache is used for execution.

Microsoft .Net Namespaces

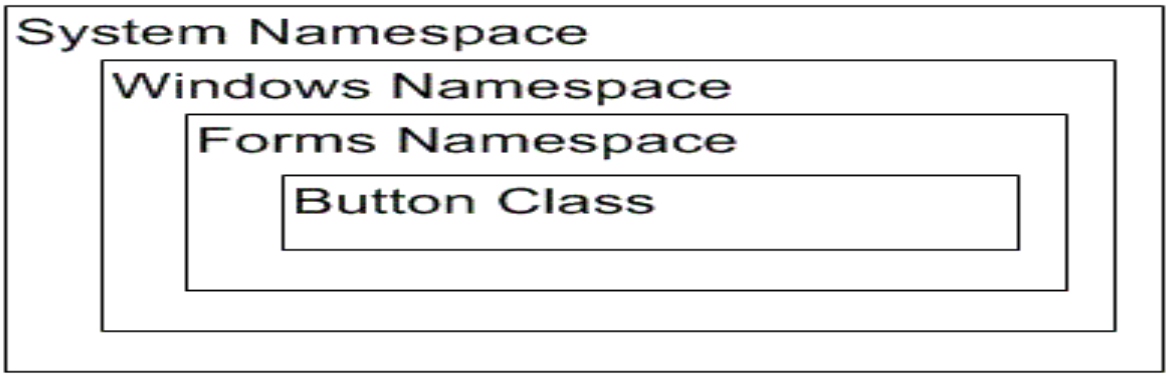
Namespaces are the way to organize .NET Framework Class Library into a logical grouping according to their functionality, usability as well as category they should belong to, or we can say Namespaces are logical grouping of types for the purpose of identification.

The .NET Framework Class Library (FCL) is a large collection of thousands of Classes. These Classes are organized in a hierarchical tree. The System Namespaces is the root for types in the .NET Framework.

Common Namespaces

- System namespace contains fundamental classes
- System.Data namespace contains classes supplying data access capabilities
  - ADO.NET
- System.Drawing provides access to the Graphical Device Interface allowing shapes to be drawn to forms and printers
- System.IO provides access to files
- System.Windows.Forms supplies the capabilities to create forms and control instances on those forms

For example, the Button type is contained in the System.Windows.Forms namespace.



Net Assembly Manifest

An Assembly Manifest is a file that containing Metadata about .NET Assemblies. Assembly Manifest contains a collection of data that describes how the elements in the assembly relate to each other. It describes the relationship and dependencies of the components in the Assembly, versioning information, scope information and the security permissions required by the Assembly.

The Assembly Manifest can be stored in Portable Executable (PE) file with Microsoft Intermediate Language (MSIL) code. You can add or change some information in the Assembly Manifest by using assembly attributes in your code. The Assembly Manifest can be stored in either a PE file (an .exe or .dll) with Microsoft Intermediate Language (MSIL) code or in a standalone PE file that contains only assembly manifest information. Using ILDasm, you can view the manifest information for any managed DLL.