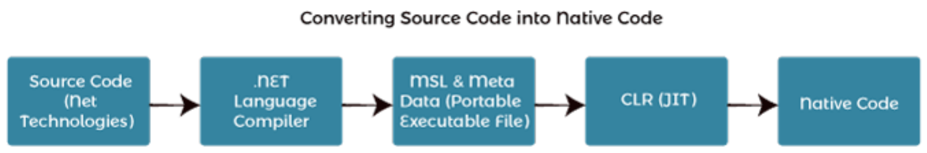
<u>Common Language Runtime(CLR)</u>

Major component of .net

Executive engine that handles running applications.

Executive engine: executing the compiling code in a managing environment.

CLR is a run time environment that manages and executes code written in any .Net programming language.


Converting Source Code into Native Code
Source Code (Net Technologies)
.NET Language Compiler
MSL & Meta Data (Portable Executable File)
CLR (JIT)
Native Code

MSIL : Microsoft Intermediate language. Comparable to byte code in java. Platform Independent.

MSIL + Metadata ---> Manifest file.

Metadata : members and types required by CLR to execute the MSIL code.

CLR has JIT compiler which converts the MSIL into Native code(platform dependent).

<u>Components of CLR</u>

1) Common Type System

   Provides guidlines for declaring,using and managing data types in run time.

   Types referred to data type supported by languages

   Provides common type system to all the languages.

   Supports the object oriented programming concepts. This ensures that object-oriented features work consistently across languages.

   Helps in writing language independent code.

   CTS categorises types into:

   - Value Types: int i=30; //value allocated by stack
   - Reference types : String i="Hello"; //memory allocated at heap and address at stack.

2) Common Language Specification

Sub part of CTS

The CLS specifies that only certain types of data types, method signatures, and language features should be used to ensure that code can be consumed by any CLS-compliant language.

CLS is a set of rules within CTS that languages must follow to be considered CLS-compliant

3) JIT Compiler

Converts MSIL code into machine level code.

Three Types:

Pre : compiles entire MSIL code into native code before execution

Econo : compiles only those parts of MSIL code required during execution and removes those parts that are not required anymore.

Normal : compiles only those parts of MSIL code required during execution but places them in cache for future use.

4) Garbage Collection

Works as an automatic memory manager

Manage memory by automatically allocating memory according to the requirement.

Allocates heap memory of objects

reclaims the memory allocated to them for future use.

Also ensures the safety of objects by not allowing one object to use the content of another object.

5) Metadata

Data about data

Binary info about the program

Stored along with the MSIL code or in memory along with the CLR portable file (executable file)

Loaded into memory for proper interpretation of code and related information used.

Helps to implement code in language neutral manner.

6) Assemblies

Combination of bothe executable code(MISL) and metadata.

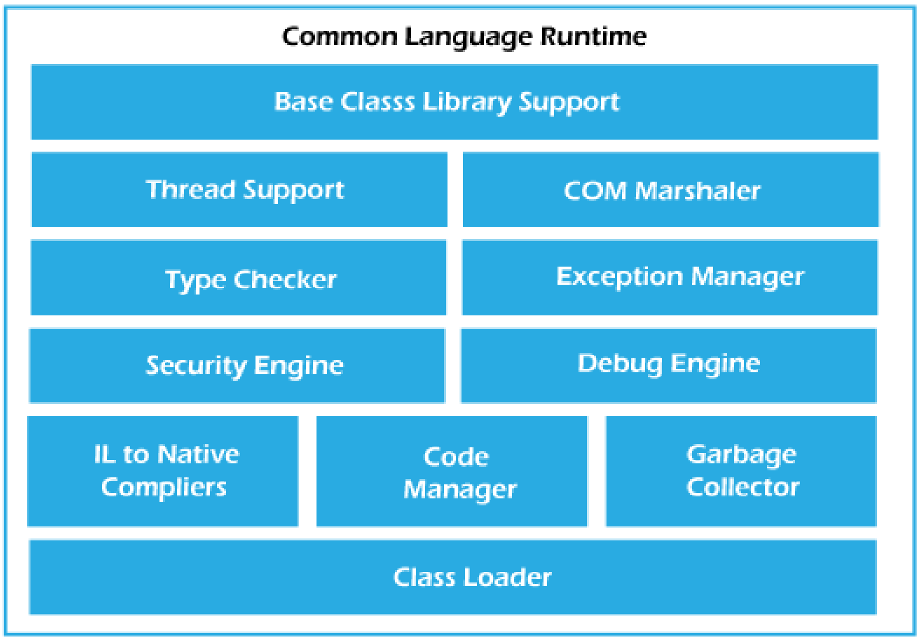
<u>**Functions of CLR**</u>

- It converts the program into native code.
- Handles Exceptions
- Provides type-safety
- Memory management
- Provides security
- Improved performance
- Language independent
- Platform independent
- Garbage collection
- Provides language features such as inheritance, interfaces, and overloading for object-oriented programs.

Managed code : The code that runs the CLR

Unmanaged code : the code that runs outside the CLR

<u>CLR Structure</u>



Components of the Common Language runtime/Architecture of CLR

1) Base Class Library Support

It is a class library that supports classes for the .NET application.

Plays a crucial role in providing comprehensive set of classes, interfaces and buildings that form a foundation.

It abstracts platform specific details, helps the developers to write code without being concerned about underlying operating system.

includes classes for exception handling, enabling developers to handle errors and unexpected situations in a consistent manner across different applications.

Defines wide range of data types and collection classes that are fundemental to data manipulation and storage.

Support File I/O

Supports Multithreading

2) COM Marshaler

provides communication between the COM objects and the application.

COM : common object model. Binary interface standard for software applications. It provides a way for software components to communicate and interact with software components regardless of the language they are written in.

3) Code manager

Managing the code during execution.

Allocate memory to objects.

4) Type Checker

Checks the type of the variable declared.

5) Debug Engine

Find and remove bugs from the code.

6) Class Loader

Load all classes at runtime.

Source code will be compiled and converted into Intermediate Language(very secure)

We can send the intermediate language code to anyone, then they can execute the code but can't steal our code.

For example, If we send the intermediate language to client and that is loaded, where there will be CLR in client machine, then the CLR links that with class loader and that class loader links with all the libraries and classes required for the execution of the class.

The main task of class loader is to load data in the runtime.

Source code ----> compiled to Intermediate language by compiler ----> Using CLR ‘s class loader all the libraries and class (data) loaded ----> Using the JIT compiler the data is converted to machine code (native code) ----> project runs on system.

7) Exception Manager

   Exception : logically the program is correct, but due to the input the program results in inconsistencies.

   an event or occurrence that disrupts the normal flow of a program's instructions.

   Exception Manager handles the exception of both managed(intermediate code) and unmanaged code(native or assembly code).

8) Thread Support

   Multiprogramming : Multiple programs running parallel

   Multithreading : a process having multiple sub parts(threads) and are executing parallely

   Thread support helps in performing the multi threading

9) Security Checker

   Some memory of the database should not be accessed.

   The data where access is not granted shouldnt be available to everyone.

   This task is done by security checker.

10) Garbage Collector

   Garbage in a program : the variables , objects which are unused anymore

   Collects the information about all the objects which are unused and removes from memory.

   Objects created are stored in heap, garbage collector periodically checks the heap for unused objects and removes them from the memory when they are unused.

11) IL to Native compilers

   converting Intermediate Language (IL) code, also known as Common Intermediate Language (CIL), to native machine code is performed by the Just-In-Time (JIT) compiler in the context of the Common Language Runtime (CLR) in the .NET framework.