CLR

Common language runtime.

There are 2 compilation process

1.Source code to intermediate code(c# compile do this work)

2.For running intermediate code to native code (we need clr for monitoring it)

Code manager:

It means managing the code during execution.

Providing the memory to objects(variables,method etc)done by code manager

Type check:

Checking the types of variables is take care of it.

Debug engine:

The engine which is finding the error.

Find and remove bugs from code

Class loader:

Who is loading the class

Two compilations source to intermediate language or Microsoft il (msil)or manage code it is very secure for example we send this exe file to client sothen can run but don’t access the original code. When we double click the exeit starts and give output then clr having class loader then we run through clr

Class loader loads the data into run tym. This jit compile convert to native code and then project runs.

Exception manager:

Errors or not exceptions "exception" refers to an event or occurrence that disrupts the normal flow of a program's execution. Exceptions are typically unexpected or exceptional situations that can arise during the runtime of a program and require special handling to prevent the program from crashing or producing incorrect results.

Exceptions handle the expection in manage code(source to intermediate) languages are .net,java and unmanage code(source directly to native code) languages support c,vb.

Thread support:

Multiple programming:

When ever we are opening multiple programs of different tasks(playing game,music listening,drawing)

Multiple threading:

Ex: google chrome has suparts called tabs they are called threads.

Doing the diff types oftasks which are not linked are called multiprogramming.

Doing the similar types of tasks which are interlinked called threads dealing with it is multiple threading

Making the program where we have multiple threads ther execute parallely taken care by thread support,supports multithreading.

Security check:

Intermediate code(it checks access to some only)

Meomory(every body don’t have access to it

It restrict access to system resourses memory should be totally secured.

Garbage collector

Automatic memory management by checking continuously heap.

The things which are unnecessary

In Heap memory objects are stored if 3rd and 1 st objects are not required the garbage collector checks the heap it automatically take that obj free from memory

Definition of CLR:

CLR provides a framework for developing and deploying .NET applications, including a set of libraries, called the .NET Framework Class Library, which provides access to a wide range of functionality, such as input/output operations, networking, database connectivity, and user interface design.

Internally CLR includes the JIT(Just-In-Time) compiler which converts the MSIL code to machine code which further executed by CPU. CLR also uses the .NET Framework class libraries. Metadata provides information about the programming language, environment, version, and class libraries to the CLR by which CLR handles the MSIL code.

**Components of CLR:**

Common Language Specification (CLS):

It is responsible for converting the different .NET programming language syntactical rules and regulations into CLR understandable format.it has Language Interoperability(LI) means providing execution support to other programming languages also in .NET framework.

LI can be implemented in two ways:

manage code(source to intermediate) languages are .net,java

unmanage code(source directly to native code) languages support c,vb.

Common Type System (CTS):

Every programming language has its own data type system, so CTS is responsible for understanding all the data type systems of .NET programming languages and converting them into CLR understandable format which will be a common format.

There are 2 Types of CTS

Value Types: Value Types will store the value directly into the memory location. These types work with stack mechanisms only. CLR allows memory for these at Compile Time.

Reference Types: Reference Types will contain a memory address of value because the reference types won’t store the variable value directly in memory. These types work with Heap mechanism. CLR allot memory for these at Runtime.

JIT compiler:

The MSIL code is converted into native code in the second compilation process. This process is called JIT compilation. There are three types of JIT compilers -Pre, Econo, and Normal. Pre JIT Compiler compiles entire MSIL code into native code before execution. Econo JIT Compiler compiles only those parts of MSIL code required during execution and removes those parts that are not required anymore. Normal JIT Compiler also compiles only those parts of MSIL code required during execution but places them in cache for future use.

Metadata:

A Metadata is a binary information about the program, either stored in a CLR Portable Executable file (PE) along with MSIL code or in the memory. During the execution of MSIL, metadata is also loaded into memory for proper interpretation of classes and related. Information used in code. So, metadata helps implement code in a language-neutral manner or achieve language interoperability.

### Assemblies:

An assembly is a fundamental unit of physical code grouping. It consists of the assembly manifest, metadata, MSIL code, and a set of resources like image files. It is also considered a basic deployment unit, version control, reuse, security permissions, etc.

Class loader:

A class loader is a component of a programming language runtime system that is responsible for loading classes into the memory during the program's execution. In languages like Java and .NET (Common Language Runtime), class loaders play a crucial role in dynamic class loading and linking.

Base class Library:

The Base Class Library is a set of reusable classes, interfaces, and value types provided by a runtime environment or a framework. It serves as a foundation for application development, providing common functionality that developers can leverage without having to write the code from scratch.

It includes classes for I/O operations, data access, networking, threading, and more.

Namespaces like System, System.IO, System.Collections, etc., contain fundamental classes.

MSIL to Native Compilers:

When you compile a C# program, it is translated into an intermediate language called Microsoft Intermediate Language (MSIL). The CLR then just-in-time compiles this MSIL to native machine code during runtime. There isn't a separate "MSIL to native compiler" that you explicitly invoke; it's handled by the CLR.

You can view MSIL using tools like Ildasm (IL Disassembler) in Visual Studio.

Some scenarios may involve converting MSIL to native code ahead of time (AOT compilation).

Tools like NGEN (Native Image Generator) in .NET allow developers to create native images from MSIL, which can improve application startup performance.

COM Object:

A COM object is an instance of a component that follows the Component Object Model (COM) specifications.

COM objects are typically written in languages like C++ and expose a set of interfaces that define their functionality.

Managed Proxy (COM Callable Wrapper - CCW):

C# is a managed language, and it operates in a managed runtime environment, such as the Common Language Runtime (CLR).

The CLR provides a way to interact with unmanaged COM objects by creating a managed proxy, known as the COM Callable Wrapper (CCW).

The CCW acts as a bridge between the managed C# code and the unmanaged COM object.

COM (Component Object Model) is a binary-interface standard for software components introduced by Microsoft.

COM marshaler is responsible for handling the communication between managed and unmanaged code, especially when dealing with COM objects. IDisposable Interface for Cleanup:

When dealing with COM objects, it's crucial to release resources explicitly, as the garbage collector doesn't manage the lifetime of COM objects. Implementing the IDisposable interface allows you to release resources properly.