**Microsoft started development on the .NET Framework in the late 1990s originally under the name of Next Generation Windows Services (NGWS).**

1. **.Net Framework is a part of .NET platform.**
2. **.Net Framework support multiple languages.**
3. **.Net Framework provides the necessary compile-time and runtime foundation to build and run .Net based applications.**

**1) Components of .Net Framework?**

**- CLR (Common Language Runtime)**

**- Base Class Library**

**- Language Compilers.**

**2) What is CLI (Common Language Infrastructure)?**

**CLI enables an application program written in any of several commonly-used programming languages to be run on any operating system using a common runtime program rather than a language-specific on. CLI provides a virtual execution environment.**

**It is an open specification developed by Microsoft that describes the executable code and runtime environment that allows multiple high-level languages to be used on different computer platform without being rewritten for specific architectures.**

**3) What is CTS (Common Type System)?**

***CTS are common in-build data types of .NET framework which can be used by only .NET compliant languages.***

**The language interoperability and .NET Class Framework are not possible without all the languages sharing the same data types. This means is that an “int” should mean the same in VB, VC++, C# and all other .NET compliant languages. Same idea follows for all the other data types. This is achieved through introduction of CTS.**

**It performs the following functions:**

* **Establishes a framework that enables cross-language integration, type safety and high performance code execution.**
* **Provides an object-oriented model that supports the complete implementation of my programming languages.**

**4) What is CLS (Common Language Specification)?**

***CLS is a set of rules to design any language to support .NET Framework.***

**CLS is subset of CTS. CLS is actually a set of restrictions on the CTS defines not only type allowed in external calls, but the rules for using them, depending on the goal of the user.**

**5) What is MSIL (Microsoft Intermediate Language)?**

**A .NET programming language does not compile into executable code instead it compiles into an intermediate code called MSIL. The MSIL code is then send to the CLR that converts the code to machine language which is then run on the host machine. MSIL is similar to Java Byte code.**

**6) What is CLR (Common Language Runtime)?**

It is a very important part of .NET Framework. At the base level, it is the infrastructure that executes applications and allows them to interact with other parts of the Framework. It also provides important capabilities in optimizing, securing and providing many robust capabilities such as application deployment and side-by-side execution.

Services provided by the CLR (MCA-VCA-SEA):

M – Managing memory of managed objects.

C – Conversion of IL to Native code.

A – Application memory isolation.

**V – Verification of type safety.**

**C – Code management (loading and execution).**

**A – Access to metadata (enhanced type information).**

**S – Support for developer services.**

**E – Exception handling, including cross-language exceptions.**

**A – Automation of object layout.**

**7) Components of CLR?**

1. **Class Library Support**
2. Thread Support
3. Type Checker
4. **Security Engine**
5. **Debug Engine**
6. **Exception Manager**
7. **COM Marshaller**
8. **MSIL to Native Compiler**
9. **Code Manager**
10. **Garbage Collector**
11. **Class Loader**

**8) What is COM Marshaller?**

It performs marshalling of data when data passes between managed and unmanaged execution environments. Marshalling manages the different representations of data across different execution environments. **It performs the necessary conversions in data formats between managed and unmanaged code.**

**9) What is Managed Code?**

1-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.

2-Managed Code running under a CLR cannot be accessed outside the runtime environment.

3-It offers services like garbage collection, run-time type checking, reference checking etc. By using managed code you can avoid many typical programming mistakes that lead to security holes and unstable applications.

**10) What is Unmanaged Code?**

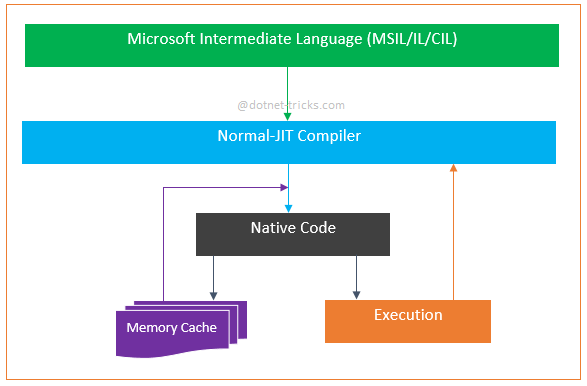
Unmanaged code compiles straight to machine code and directly executed by the Operating System. The generated code runs natively on the host processor and the processor directly executes the code generated by the compiler. It is always compiled to target a specific architecture and will only run on the intended platform. So, if you want to run the same code on different architecture then you will have to recompile the code using that particular architecture.

**All code compiled by traditional C/C++ compilers are Unmanaged Code.**

**11) What are the different Types of JIT Complier?**

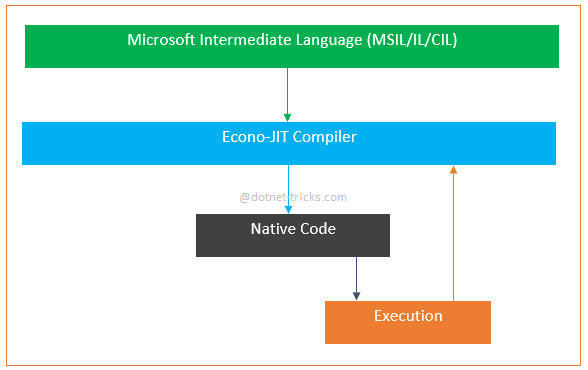
**Normal JIT**

**This complies only those methods that are called at runtime. These methods are compiled only first time when they are called, and then they are stored in memory cache. This memory cache is commonly called as JITTED. When the same methods are called again, the complied code from cache is used for execution.**



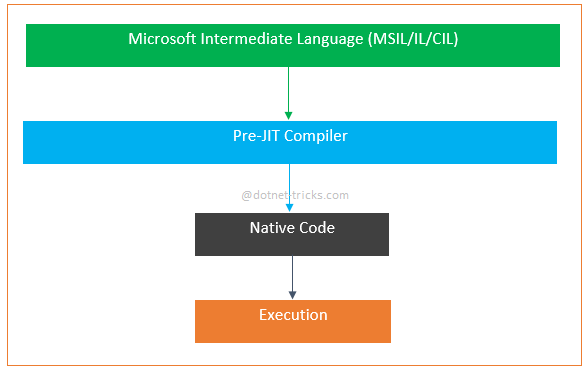
**Econo JIT**

**This complies only those methods that are called at runtime and removes them from memory after execution.**



**Pre JIT**

**This complies entire MSIL code into native code in a single compilation cycle. This is done at the time of deployment of the application.**



**12) What do code contracts do?**

**Provide a way to specify preconditions, postconditions**, and object invariants in your code.

**Preconditions:** are requirements that must be met before entering a method or property. **Postconditions:** describe expectations at the time the method or property get executed.

**13) GAC (Global Assembly Cache)**

The Global Assembly Cache (GAC) is a folder in Windows directory to store the .NET assemblies that are specifically designated to be shared by all applications executed on a system. Assemblies can be shared among multiple applications on the machine by registering them in global Assembly cache (GAC). GAC is a machine wide a local cache of assemblies maintained by the .NET Framework.

**To install**

gacutil -i [Path][Assembly Name].dll

**To uninstall**

gacutil -u [Assembly Name], Version=1.0.0.0, PublickeyToken=7896a3567gh