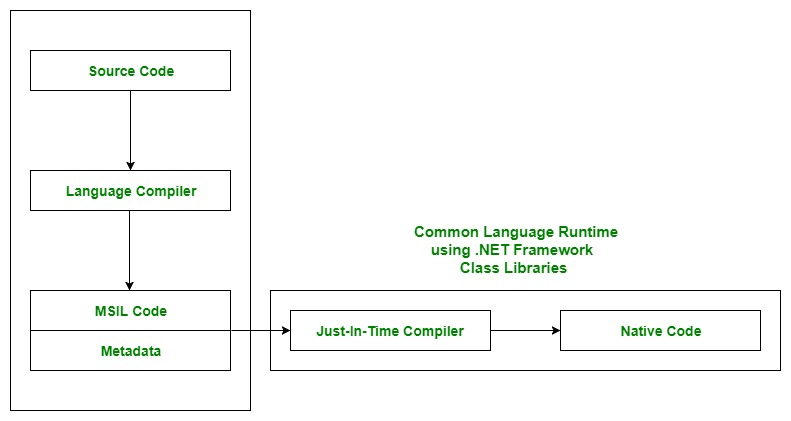
**.NET Framework Introduction Assignment**

**1.Demonstrate the process of conversion of Source code into the native machine code in .Net framework with the help of a flowchart.**

**Ans:**



Source:<https://www.geeksforgeeks.org/common-language-runtime-clr-in-c-sharp/>

Conversion involves following steps:

* .Net framework has Visual Basic, C#, Visual C++ and other compilers which translate source code into Microsoft Intermediate Language(MSIL) code and generate the required metadata.
* MSIL code is also known as Intermediate Language(IL) code or Common Intermediate Language(CIL) code.
* Now Common Language Runtime(CLR) includes a JIT compiler for converting MSIL to native code.

**2.Explain in detail the CTS and how the .net framework implements CTS.**

**Ans:**

The Common Type System(CTS) is a standard for defining and using data types in .net framework. CTS defines a collection of data types, which are used and managed by the run-time to manage cross-language integration.

CTS provides the types in .NET framework and applications are built in different programming languages so information is shared easily.

CTS support two kinds of types:

Value Types- Contain the values that need to be stored directly into the stack or allocated structure.

Reference Types- Store a reference to he values memory address and are allocated on the heap.

**Implementation:**

If make a console application in both c# and visual basic.

C# data declaration:

Double value;

Visual basic declaration:

Dim CustomerValue as Double

Now we’ll start debugging both the console application with “ILDASM” tool.

After debugging we’ll find that visual basic Double converted into float64 and same in the case of c#.

Hence we can clearly see that .NET framework is converting the data type to a Common Type System(CTS).

**3.Name at least 3 runtime services provided by CLR and explain their role in .net framework.**

**Ans:**

CLR provides multiple services to execute processes, like memory management service and

security services. CLR performs multiple tasks to manage the execution of .NET applications.

1.Automatic Memory Management:

CLR calls various predefined functions of .NET framework to allocate and deallocate memory of .Net objects.

2.Garbage Collection:

GC is used to prevent memory leaks or holes.GC of CLR automatically determines the best time to free the memory, which is allocated to an object for execution.

3.JIT compilation:

Just In TIme converts MSIl code to targeted machine code for execution.

**4.What are the differences between Library vs DLL vs .Exe? Explain.**

**Ans:**

**LIB:**

* Functions that are statically linked to a program.
* They are not shared by other programs.
* If you have two programs linked with third, then both programs will contain code in third.

**DLL:**

* Dynamic Link Library.
* The system loads a DLL into the context of an existing system.
* The purpose of a DLL is to have a collection of methods/classes which can be re-used from some other application.

**.EXE:**

* Executable file.
* When loading an executable, no export is called, but only the module entry point.
* When a system launches a new executable, a new process is created.
* The purpose of exe is to launch a separate application of its own.

**5.How does CLR in .net ensure security and type safety? Explain.**

**Ans:**

CLR (Common Language Runtime) of the .NET framework has its own secure execution model that isn’t part of any operating system nor bound to where it's running on. CLR enforces security policy based on where code is coming from rather than who the user is. This model, called code access security, makes sense in today’s environment because so much code is there all over the internet and even a trusted user doesn't know when the code the safe. So in this way .NET provides security.

TYPE safety meaning that types can interact only through protocols they define, thereby ensuring each type internal consistency. For example, C# prevents from interacting with a string type though it is was integer type.

More importantly, language supports static typing that means the language enforces type safety at compile time. This is in addition to dynamic type safety, which the .NET enforces at runtime.

Thanks and regards,

Mandeep Kumar

(Mandeep.Kumar@accoliteindia.com)