Q. Why .Net is known as multilingual application development technology.

**Introduction**

A application is said to be multilingual if it can be deployed in many different languages. With .NET, all of the languages Visual Basic .Net, C# and J# compile to common Intermediate language (IL). This make all languages interoperable. Microsoft creates java byte code, which is low-level language with a simple syntax, which can be very quickly translated into native machine code.

**CLR**

.Net framework is multilingual application because of CLR.

CLR is key of .NET framework. Code running under the control of the CLR is often termed as managed code.

Main task of CLR is to convert compiled code into native code.

.NET framework has one or more compiler for e.g. VB.NET, c#, c++, JScript or any third party compiler as COBOL

Any one of these compiler will convert your source code into Microsoft Intermediate Language (MSIL). Main reason of .NET to be multilingual is that you can compile your code from IL and this compiled code will be interoperable with the code that has been compiled to IL from another language.

It simply means that you can create pages in different languages (like c#, vb.net, j# etc.) and once all of these pages are compiled they all can be used in single application. Let us understand this point clearly with example.

Let us consider situation where customer needs an application to be ready in 20 days. For completing application in 20 days we want 30 developer who all knows specific language but we have 15 developer who knows c# and 15 developers who knows vb.net. In this situation, if we don’t use .NET then we need to hire more 15 developer of c# or vb.net which is difficult and costly solution. Now if we use .NET then we can use c# and vb.net language in same application. This is possible because once c# code is compiled from IL it become interoperable with vb.net code which is compiled from IL.

Then JIT (Just In Time) of CLR convert this MSIL code into native code using metadata which is then executed by OS.

CLR stands for common language runtime. Common language runtime provide other services as memory management, thread management, remoting and other security as CTS and CLS

CLR is layer between operating system and .net language, which use CTS and CLS to create code.

**CTS**

CTS stands for common type system. CTS define rules that common language runtime follows when we are declaring, using and managing type. **CTS** deal with **data type.**

.Net support many languages and every language have its own data type. One languages cannot understand data types of other language.

For example: When we are creating application in C #, we have int and when we are creating application in vb.net, we have integer. Here CTS come in light, after compilation CTS convert int and integer into int32 structure.

**CLS**

CLS stands for common language specification.

CLS is subset of CTS and it declare all the rule and restriction that all language under .NET framework must follow.

The language, which follow these rules, are known as CLS complaint.

For example: We can use multiple inheritance in c++ but when we use the same code in c# it create problem because c# does not support multiple inheritance. Therefore, CLS restrict multiple inheritance for all language.

One another rule is that you cannot have member with same name and different case.

In c# add() and Add() are different because it is case sensitive but problem arise when we use this code into vb.net because it is not case-sensitive and it consider add() and Add() as same.