Introduction to .NET Framework

.NET is a software framework which is designed and developed by Microsoft. The first version of .Net framework was 1.0 which came in the K2. In easy words, it is a virtual machine for compiling and executing programs written in different languages like C#, VB.Net etc.

It is used to develop Form-based applications, Web-based applications, and Web services. There is a variety of programming languages available on the .Net platform, VB.Net and C# being the most common ones. It is used to build applications for Windows, phone, web etc. It provides a lot of functionalities and also supports industry standards.

.NET Framework supports more than 60 programming languages in which 11 programming languages are designed and developed by Microsoft. The remaining Non-Microsoft Languages which are supported by .NET Framework are not designed and developed by Microsoft.

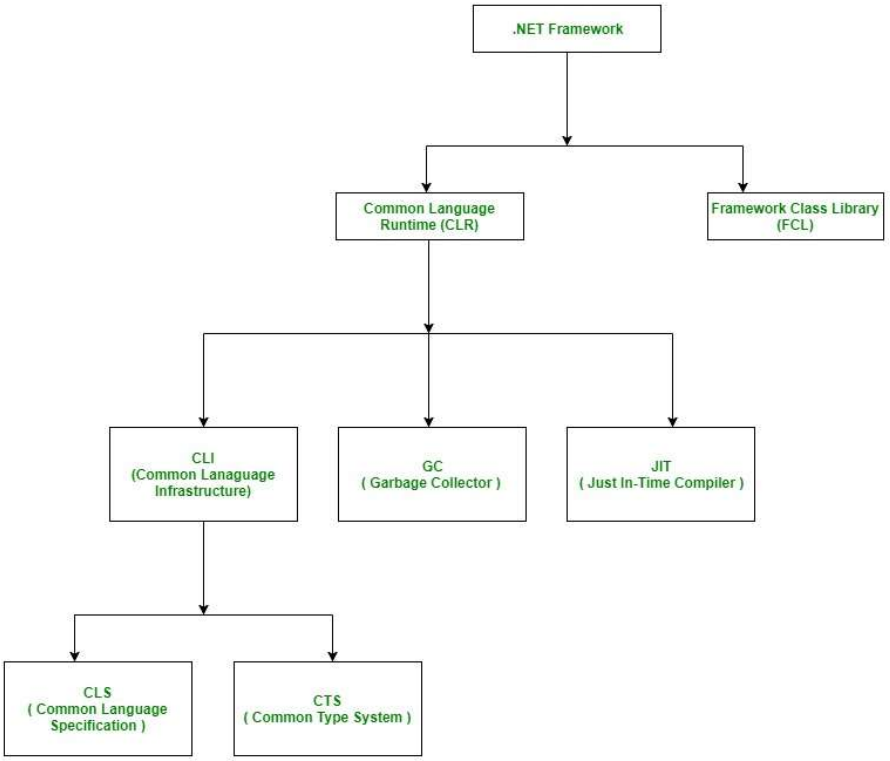
11 Programming Languages supporting .NET which are designed and developed by Microsoft are:

- C#.NET
- VB.NET
- C++.NET
- J#.NET
- F#.NET
- JSCRIPT.NET
- WINDOWS POWERSHELL
- IRON RUBY
- IRON PYTHON
- C OMEGA
- ASML(Abstract State Machine Language)

Main Components of .NET Framework

Common Language Runtime (CLR): CLR is the basic and Virtual Machine component of the .NET Framework. It is the run-time environment in the .NET Framework that runs the codes and helps in making the development process easier by providing the various services such as remoting, thread management, type-safety, memory management, robustness etc.. Basically, it is responsible for managing the execution of .NET programs regardless of any .NET programming language. It also helps in the management of code, as code that targets the runtime is known as the Managed Code and code doesn't target to runtime is known as Unmanaged code.

Framework Class Library (FCL): It is the collection of reusable, object-oriented class libraries and methods etc. that can be integrated with CLR. Also called the Assemblies. It is just like the header files in C/C++ and packages in the java. Installing .NET framework basically is the installation of CLR and FCL into the system. Below is the overview of .NET Framework.



CTS and CLS are parts of .NET CLR and are responsible for type safety within the code. Both allow cross-language communication and type safety. In this article, I would like to expose the relationship between these two.

CLS

CLS stands for Common Language Specification and it is a subset of CTS. It defines a set of rules and restrictions that every language must follow which runs under the .NET framework. The languages which follow these set of rules are said to be CLS Compliant. In simple words, CLS enables cross-language integration or Interoperability.

For Example

1. if we talk about C# and VB.NET then, in C# every statement must have to end with a semicolon. it is also called a statement Terminator, but in VB.NET each statement should not end with a semicolon(;).

Explanation of the above Example

So these syntax rules which you have to follow from language to language differ but CLR can understand all the language Syntax because in .NET each language is converted into MSIL code after compilation and the MSIL code is language specification of CLR.

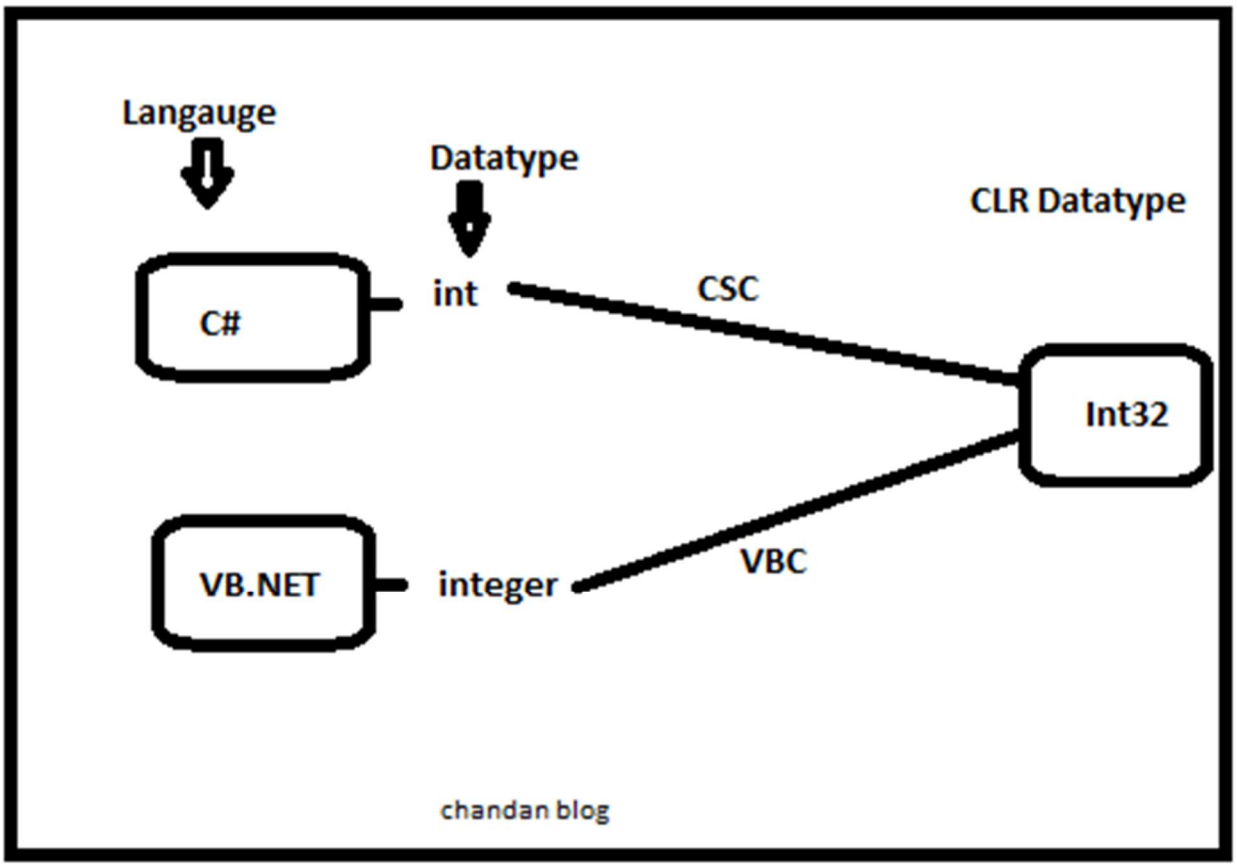
CTS

Common Type System (CTS) describes the datatypes that can be used by managed code. CTS defines how these types are declared, used and managed in the runtime. It facilitates cross-language integration, type safety, and high-performance code execution. The rules defined in CTS can be used to define your own classes and values.

OR we can also understand like,

CTS deals with the data type. So here we have several languages and each and every language has its own data type and one language data type cannot be understandable by other languages but .NET Framework language can understand all the data types.

C# has an **int** data type and VB.NET has **Integer** data type. Hence a variable declared as an int in C# and Integer in VB.NET, finally after compilation, uses the same structure Int32 from CTS.



Note

All the structures and classes available in CTS are common for all .NET Languages and the purpose of these is to support language independence in .NET. Hence it is called CTS.

Introduction to ASP .NET

ASP.NET is a web development platform, which provides a programming model, a comprehensive software infrastructure and various services required to build up robust web applications for PC, as well as mobile devices.

ASP.NET works on top of the HTTP protocol, and uses the HTTP commands and policies to set a browser-to-server bilateral communication and cooperation.

ASP.NET is a part of Microsoft .Net platform. ASP.NET applications are compiled codes, written using the extensible and reusable components or objects present in .Net framework. These codes can use the entire hierarchy of classes in .Net framework.

The ASP.NET application codes can be written in any of the following languages:

- C#
- Visual Basic.Net
- Jscript
- J#

ASP.NET is used to produce interactive, data-driven web applications over the internet. It consists of a large number of controls such as text boxes, buttons, and labels for assembling, configuring, and manipulating code to create HTML pages.

ASP.NET Web Forms Model

ASP.NET web forms extend the event-driven model of interaction to the web applications. The browser submits a web form to the web server and the server returns a full markup page or HTML page in response.

All client side user activities are forwarded to the server for stateful processing. The server processes the output of the client actions and triggers the reactions.

Now, HTTP is a stateless protocol. ASP.NET framework helps in storing the information regarding the state of the application, which consists of:

- Page state
- Session state

The page state is the state of the client, i.e., the content of various input fields in the web form. The session state is the collective information obtained from various pages the user visited and worked with, i.e., the overall session state. To clear the concept, let us take an example of a shopping cart.

User adds items to a shopping cart. Items are selected from a page, say the items page, and the total collected items and price are shown on a different page, say the cart page. Only HTTP cannot keep track of all the information coming from various pages. ASP.NET session state and server side infrastructure keeps track of the information collected globally over a session.

The ASP.NET runtime carries the page state to and from the server across page requests while generating ASP.NET runtime codes, and incorporates the state of the server side components in hidden fields.

This way, the server becomes aware of the overall application state and operates in a two-tiered connected way.

The ASP.NET Component Model

The ASP.NET component model provides various building blocks of ASP.NET pages. Basically it is an object model, which describes:

- Server side counterparts of almost all HTML elements or tags, such as <form> and <input>.
- Server controls, which help in developing complex user-interface. For example, the Calendar control or the Gridview control.

ASP.NET is a technology, which works on the .Net framework that contains all web-related functionalities. The .Net framework is made of an object-oriented hierarchy. An ASP.NET web application is made of pages. When a user requests an ASP.NET page, the IIS delegates the processing of the page to the ASP.NET runtime system.

The ASP.NET runtime transforms the .aspx page into an instance of a class, which inherits from the base class page of the .Net framework. Therefore, each ASP.NET page is an object and all its components i.e., the server-side controls are also objects.

References:

Page 1-2: Introduction to .NET Framework - Tutorialspoint.dev
Page 3-4: What Are CTS And CLS In .NET (c-sharpcorner.com)
Page 5-6: ASP.NET - Introduction (tutorialspoint.com)