Programming using C#

GUI Application Development - GAD

Programming in C#

- C# is a simple, modern, general-purpose, object-oriented programming language developed by Microsoft within its .NET initiative led by Anders Hejlsberg.

- Should not be your first programming class.
  - Assume you know C++ and/or Java and basic object-oriented or component-based programming.
- Requires (lots of) practice / reading.

History of C#

- Developed by Microsoft.
- Based on Java and C++, but has many additional extensions.
- Java and C# are both being updated to keep up with each other.
- Cross-development with Visual Basic, Visual C++, and many other .NET languages.

Implementation of Microsoft's New Vision

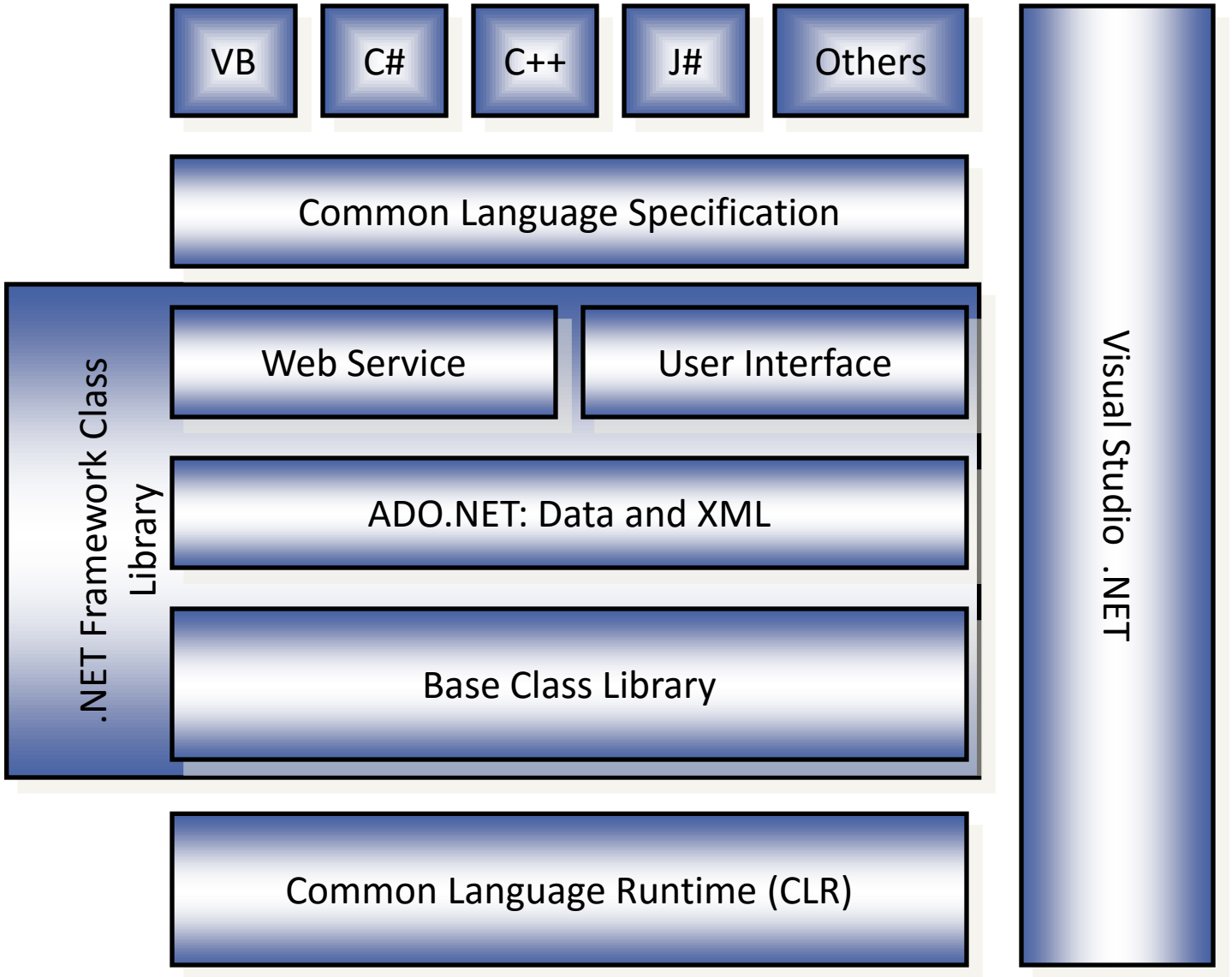
Application Foundation

.NET Framework Introduction

The .Net framework is a revolutionary platform that helps you to write the following types of applications:

- Windows applications
- Web applications
- Web services

.NET Framework Architecture



Goals of .NET

Interoperability between programming languages

So far

- millions of lines of code in C++, Fortran, Visual Basic, ...
- very limited interoperability

Under .NET

- binary compatibility between more than 20 languges (C#, C++, VB.NET, Java, Eiffel, Fortran, Cobol, ML, Haskell, Pascal, Oberon, Perl, Python, ...)

class in VB.NET

```
Public Class A
   Public x As Integer
   Public Sub Foo() ...
End Class
```

subclass in C#

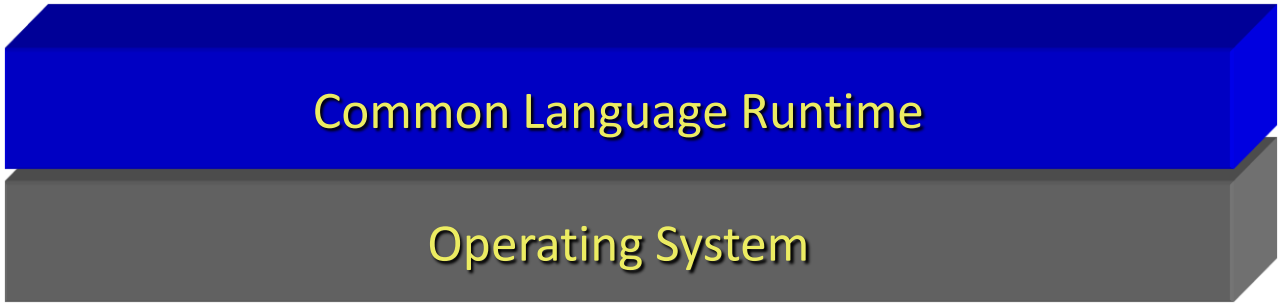
```
class B : A {
   public string s;
   public void Bar() {...}
}
```

used in Eiffel

```
class Client feature
   obj: B;
   ...
   create obj;
   obj.Bar;
   ...
end
```

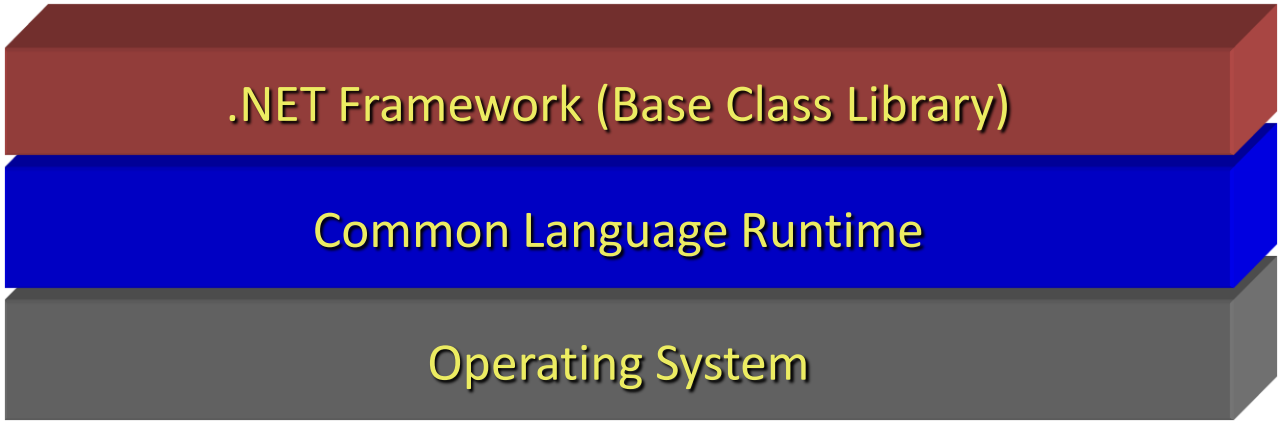
Common Language Runtime

- CLR manages code execution at runtime
- Memory management, thread management, etc.



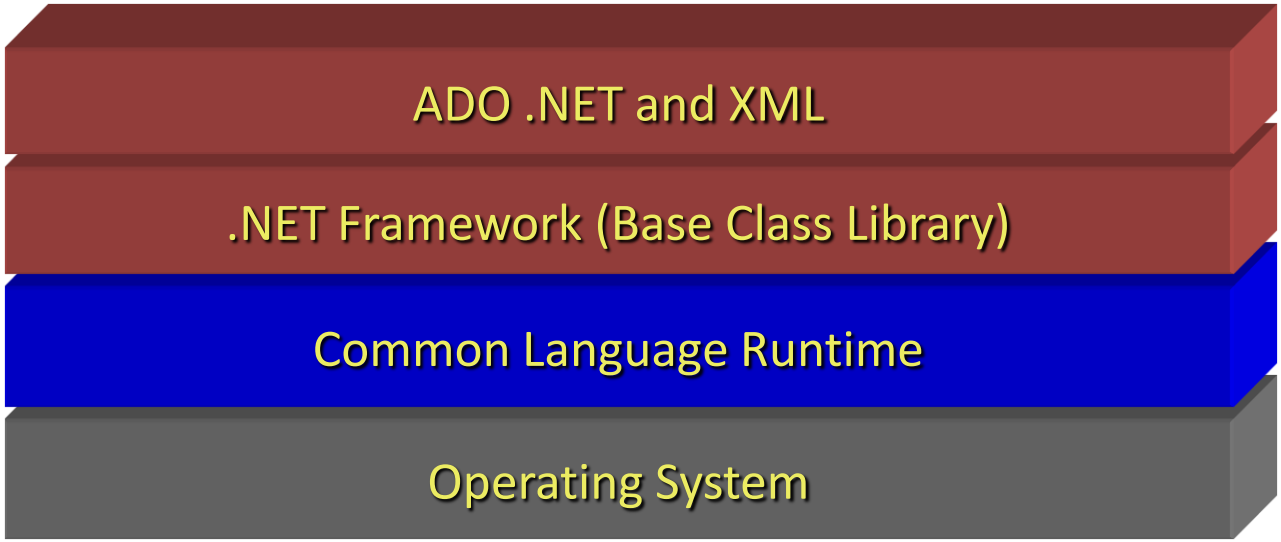
Base Class Library

- Object-oriented collection of reusable types
- Collections, I/O, Strings, …



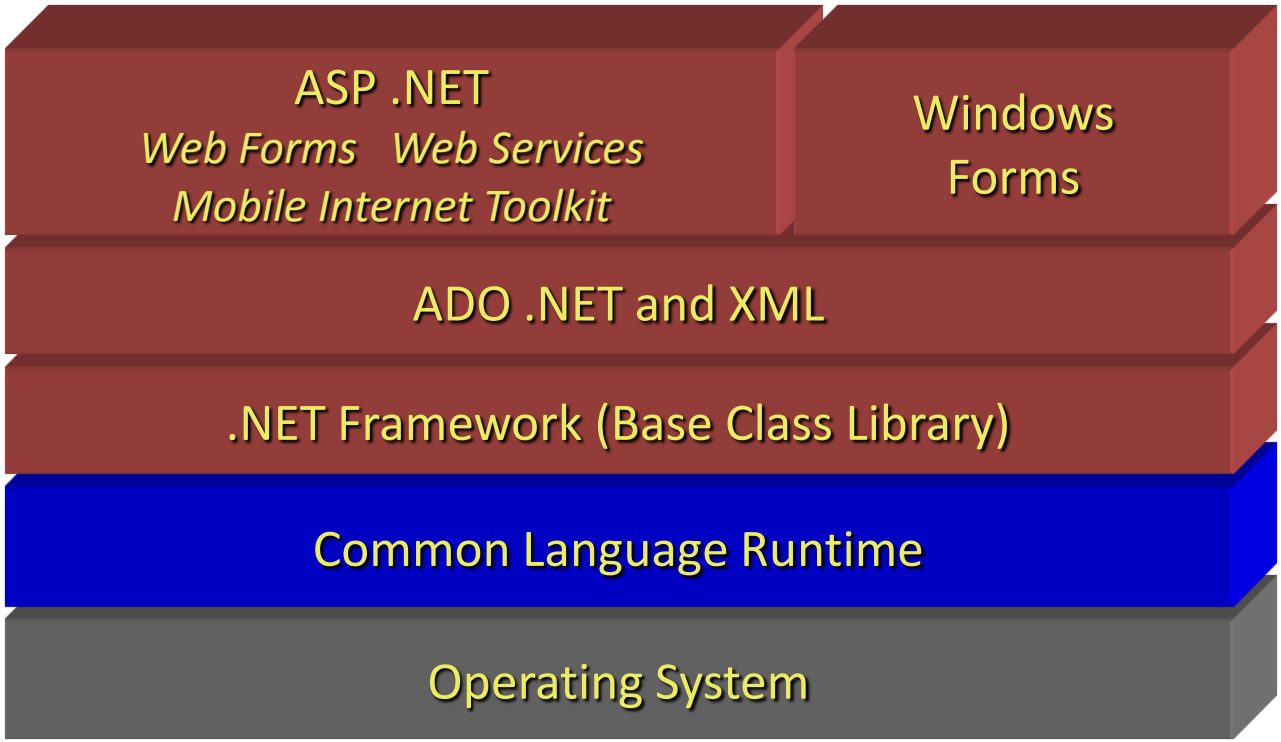
Data Access Layer

- Access relational databases
- Disconnected data model
- Work with XML



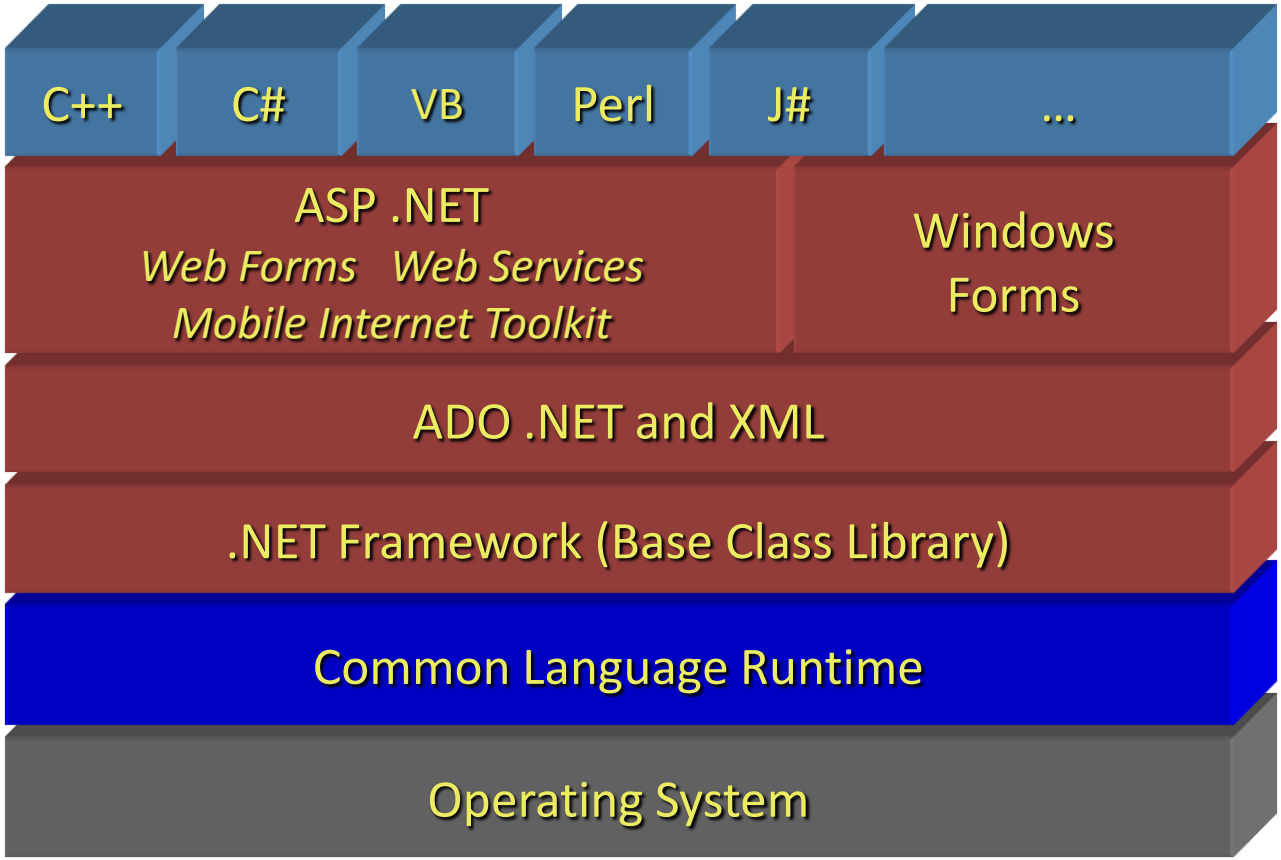
Asp.NET & Windows Forms

- Create application's front-end – Web-based user interface, Windows GUI, Web services, …

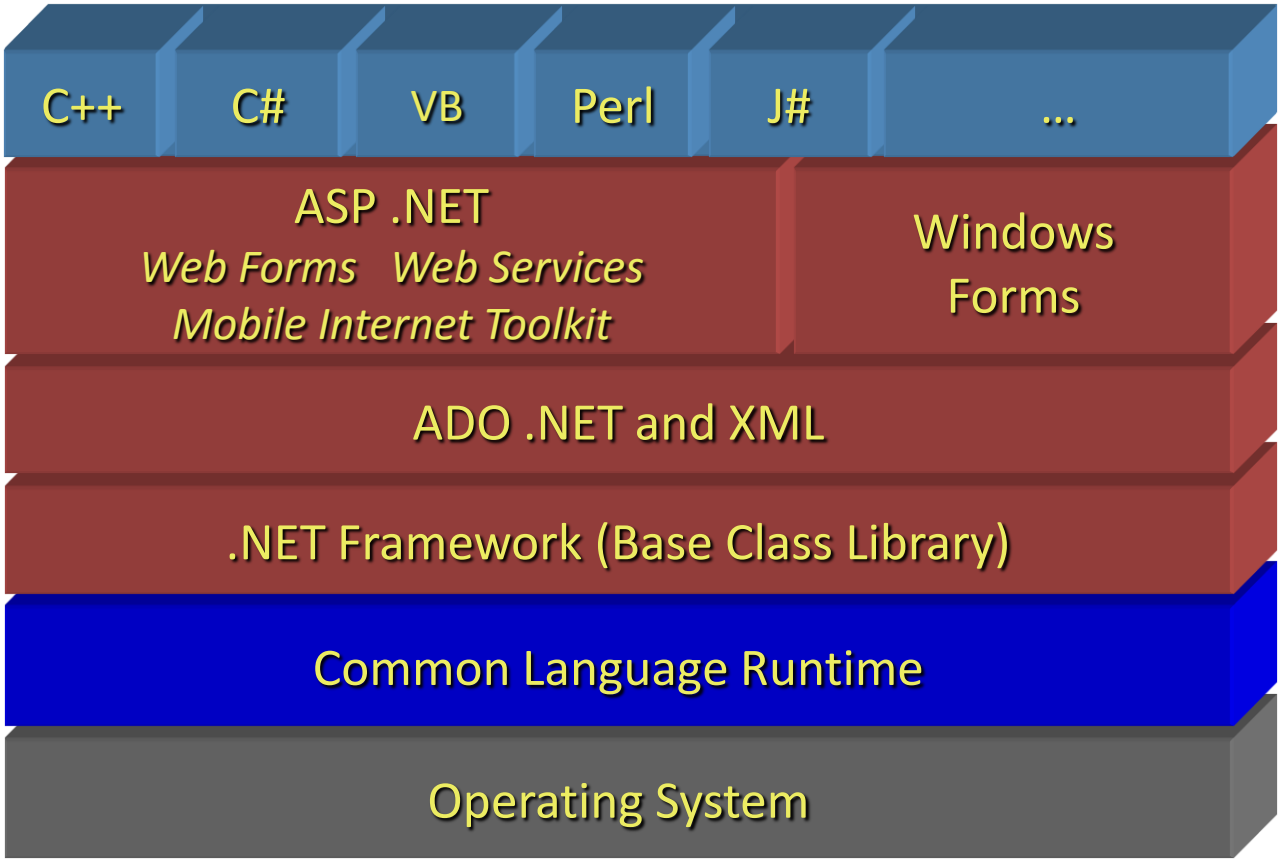


Programming Languages

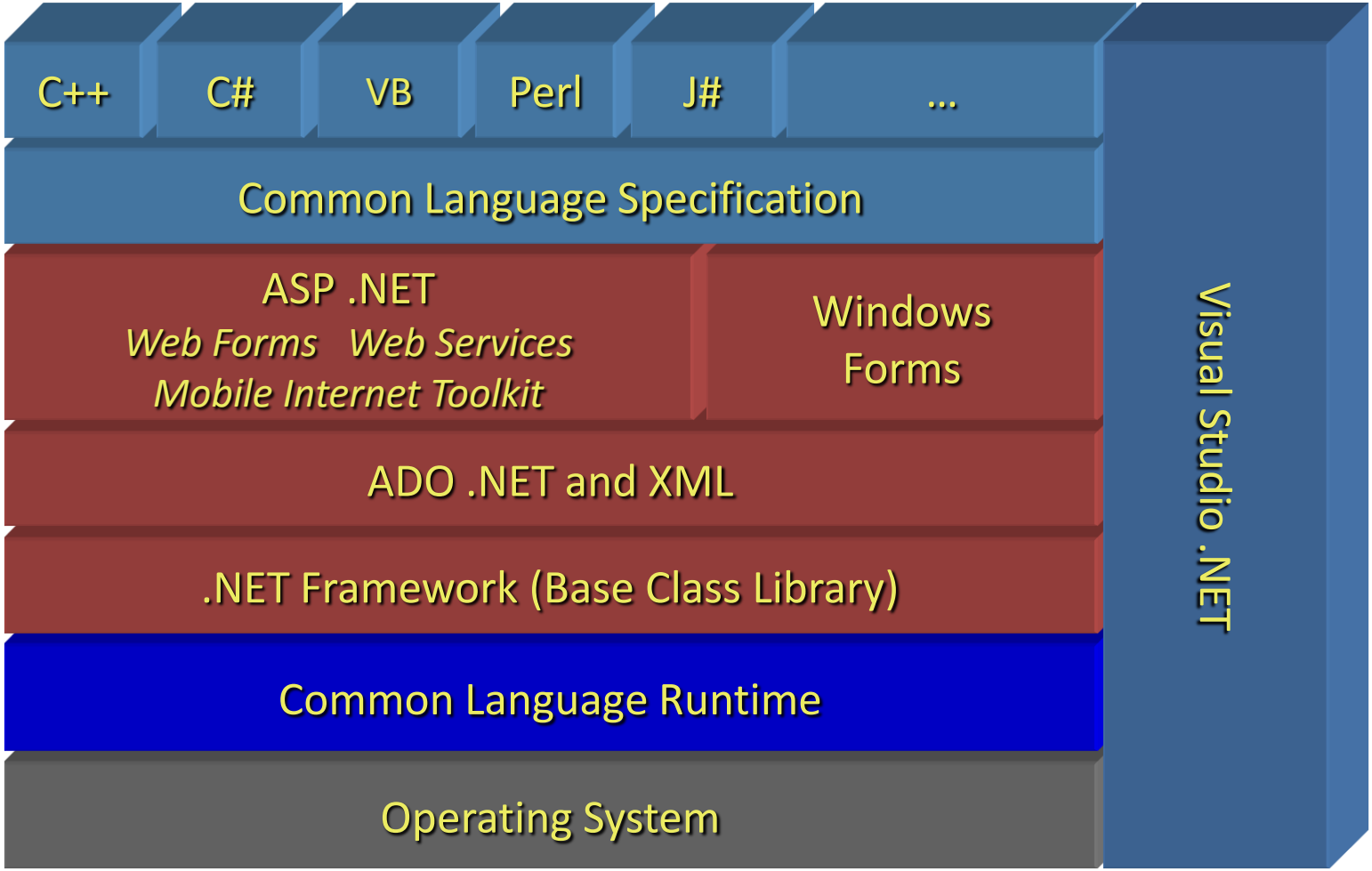
– Use your favorite language



Common Language Specification



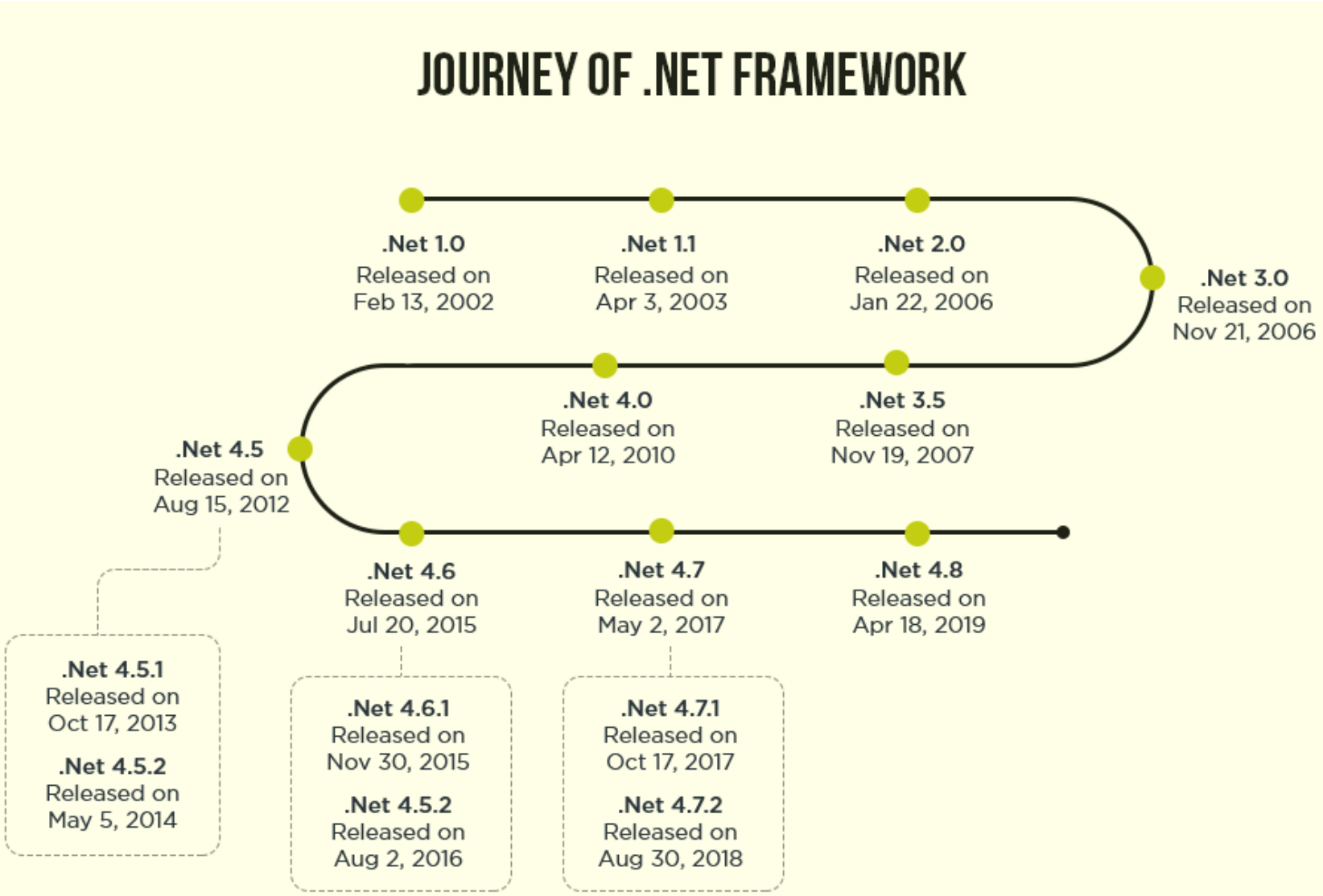
Visual Studio .NET



A Look Back…

CLR version	.NET framework version	Most important changes
1.0	1.0	Initial release: console, Windows Forms, Windows Services, Web Forms, Web Services
1.1	1.1	IPv6, ASP.NET mobile controls, ADO.NET ODBC
2.0	2.0	Generics, edit and continue, 64-bit runtime
	3.0	WPF, WCF, WF
	3.5	LINQ, ASP.NET AJAX, WF services, cryptography
	3.5 SP1	.NET Client Profile, EF
4	4	DLR, PCL, TAP, ASP.NET MVC
	4.5	Async I/O, ASP.NET Web API, ASP.NET Web Pages
	4.5+	64-bit edit and continue, async debugging, HTTP/2, 64-bit JIT

.NET Frameworks



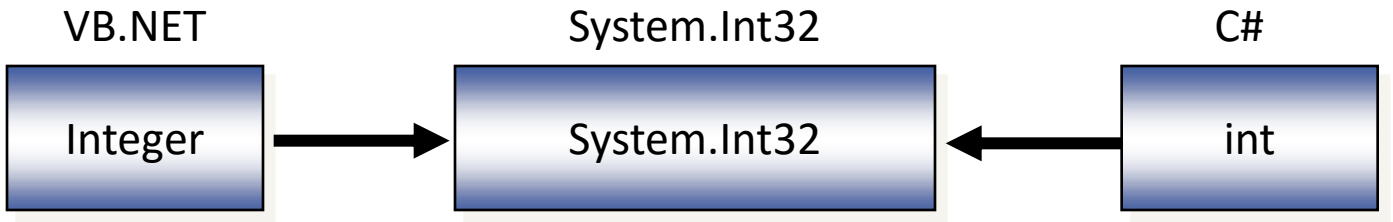
.NET Framework Components

Following are the major components of .NET Framework:

- Common Language Specification (CLS)
- .NET Framework Languages
- .NET Framework Base Class Library (BCL - FCL)
- Common Language Runtime (CLR)

Common Language Specification (CLS)

- An Important goal of .NET Framework is to support multiple languages.
- The Common Language Specification is an agreement among languages.
- The CLS defines the minimum standards that .NET languages must confirm.
- Common Language Specification provides a series of basic rules that are required for language integration.

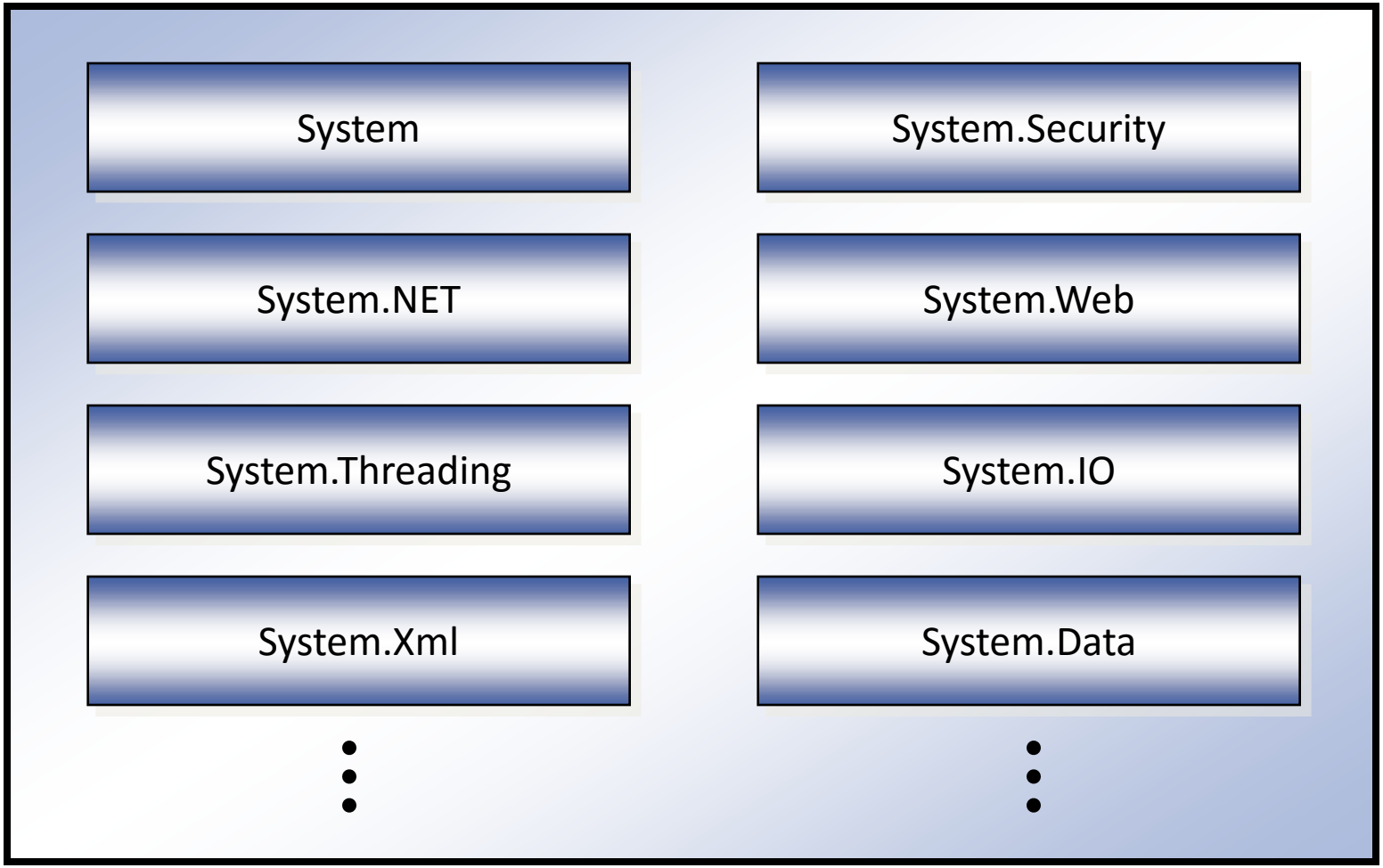


Base Class Libraries

- The common classes that are used in many programs
  - using System;
  - using System.Collections.Generic;
  - using System.Linq;
  - using System.Text;
  - using System.Threading.Tasks;

  - using System.Data;
  - using System.Drawing;
  - using System.Windows.Forms;

BCL/FCL Namespaces

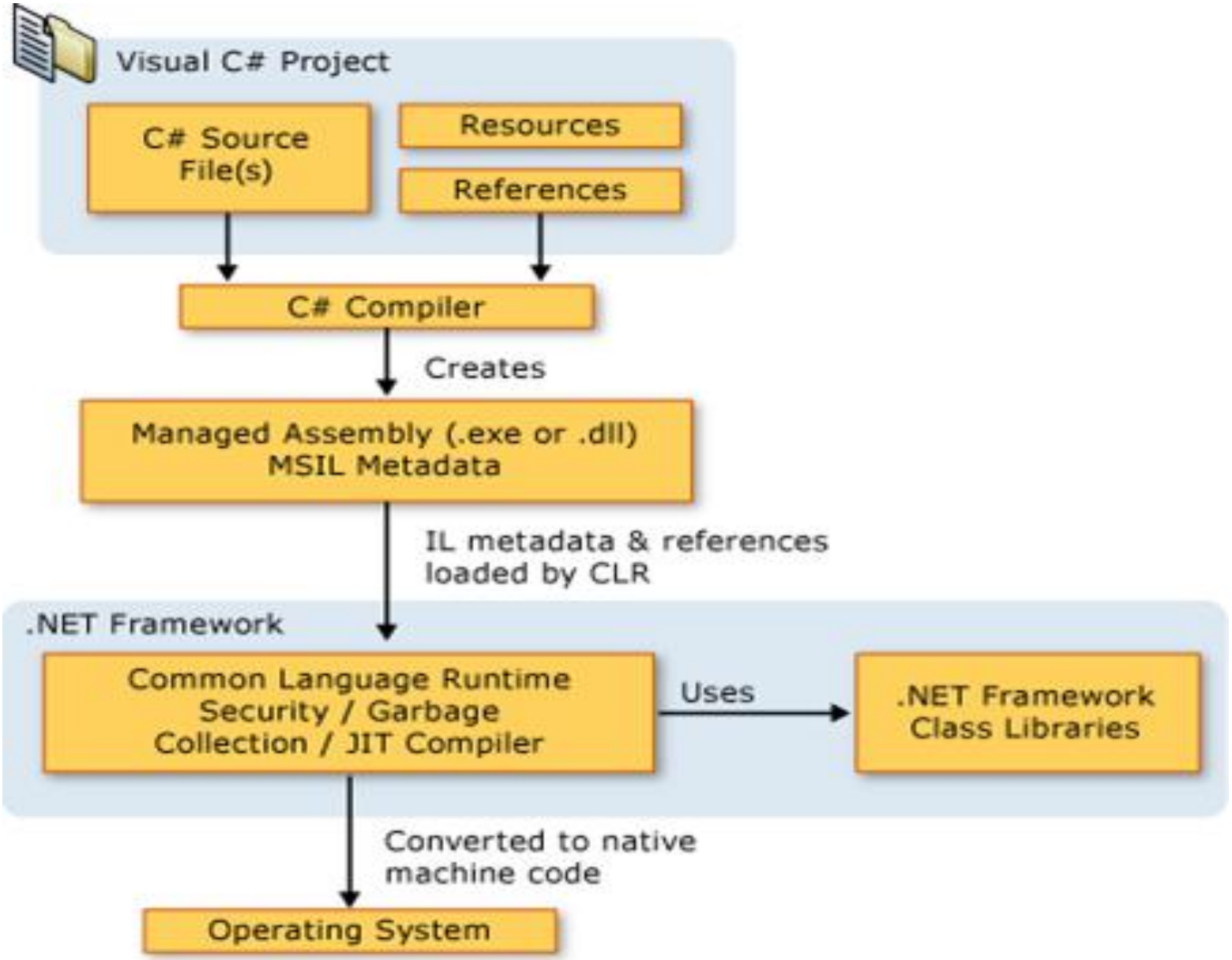


Common Language Runtime

- C# is executed indirectly through an abstract computer architecture called the CLR.

- Before the execution, C# programs are compiled to an intermediate language which is called as Microsoft Intermediate Language(MSIL) or bytecode.

CLR and JIT compiling



Microsoft Intermediate Language (MSIL)

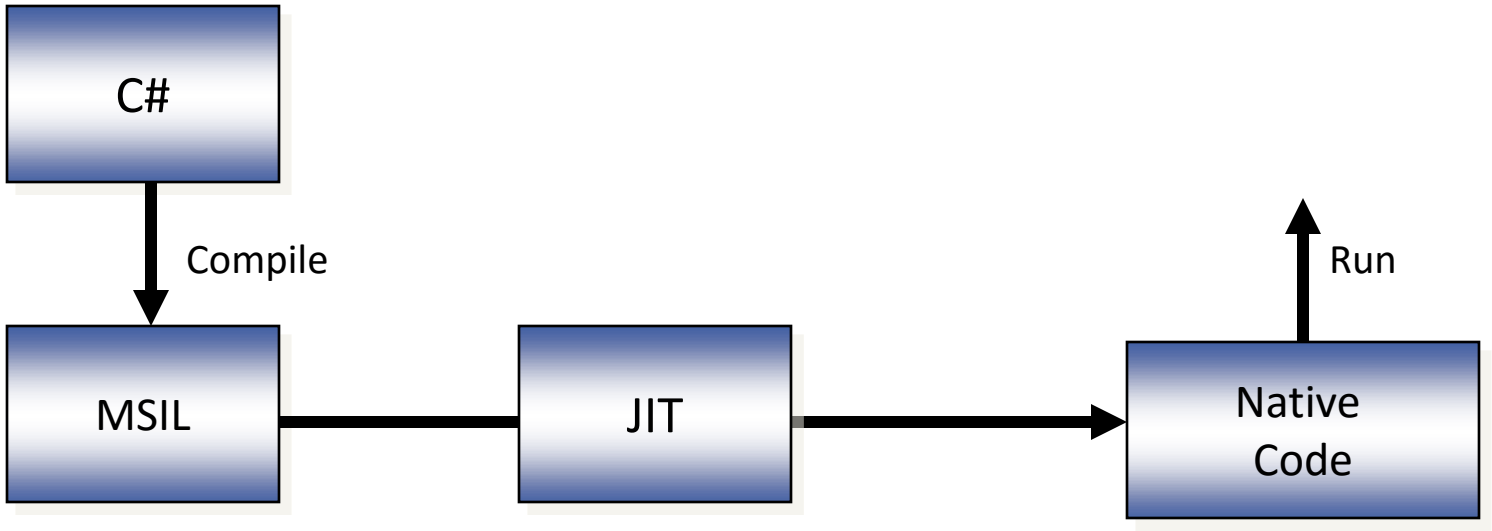
When a .NET application is compiled it is converted from the language it was written in (VB.NET, C#, J# etc) to a Managed Module.

This Managed Module contain MSIL which is direct compiled form of your code and metadata.

MSIL is a low level set of instructions understood by Common Language Runtime.

Compilation and Execution of .NET Assembly

When execution starts, the Just-In-Time (JIT) compiler of CLR compiles the IL code into native code and now native code is loaded in memory to execute.



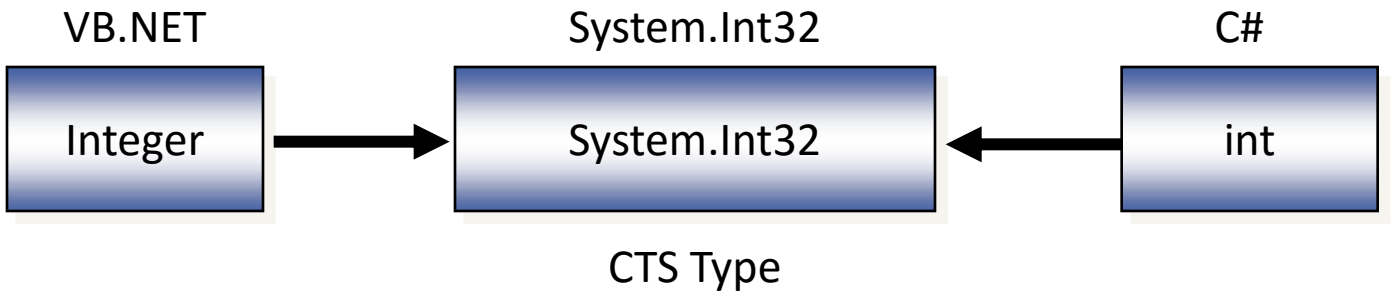
Common Type System (CTS)

.NET Framework also defines CTS which defines what types are allowed to run inside the framework.

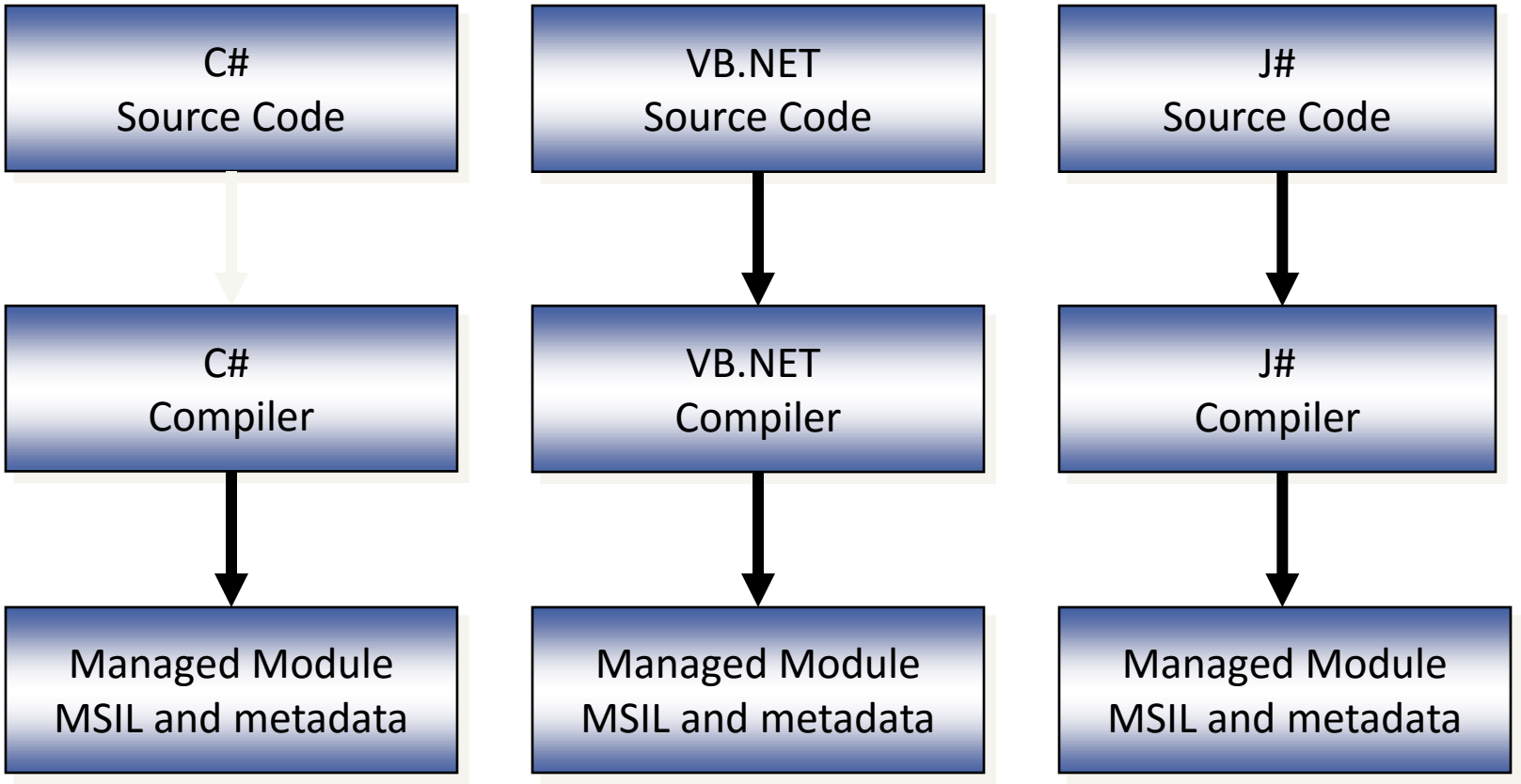
The CTS provides a wide range of types and operations that are found in many programming languages.

The CTS provides a framework for cross-language integration.

Due to this there is no difference between Integer in VB.NET and int in C#, they are System.Int32 according to CTS.



Compiling Source into Managed Module



Microsoft Visual Studio .NET

- Development tool that contains a rich set of productivity and debugging features
  - Supports managed and unmanaged applications
  - Supports C#, C++, VB.NET, …
  - Many useful tools and wizards
  - Windows Forms Designer
  - ASP.NET Web Forms Designer
  - Web Services support
  - SQL Server integration with ADO.NET and XML

Microsoft Visual Studio .NET

- **Integrated Development Environment (IDE) For C#**

- Microsoft provides the following development tools for C# programming:
  - Microsoft Visual Studio 2015

C# Program Structure

- A C# program basically consists of the following parts:
  - Class Libraries
  - Namespace declaration
  - A class
  - Class methods
  - Class attributes
  - A Main method
  - Statements & Expressions
  - Comments

C# Hello World Example

```
using System;
namespace HelloWorldApplication
{
    class HelloWorld
    {
        static void Main(string[] args)
        {
            /* my first program in C# */
            Console.WriteLine("Hello World");
            Console.ReadKey();
        }
    }
}
```

C# Hello World Example

First line: **using System;** - the **using** keyword is used to include the **System** namespace in the program. A program generally has multiple **using** statements.

The next line has the **namespace** declaration. A **namespace** is a collection of classes. The *HelloWorldApplication* namespace contains the class *HelloWorld*.

The next line has a **class** declaration, the class *HelloWorld*, contains the data and method definitions that program uses. Classes generally would contain more than one method. *HelloWorld* class has only one method **Main**.

C# Hello World Example

The next line defines the **Main** method, which is the **entry point** for all C# programs. The **Main** method states what the class will do when executed

The next line /*...*/ will be ignored by the compiler and it has been put to add additional **comments** in the program.

The Main method specifies its behavior with the statement **Console.WriteLine("Hello World");**

The last line **Console.ReadKey();** is for the VS.NET Users. This makes the program wait for a key press.

C# Hello World Example

- C# is case sensitive.
- All statements and expression must end with a semicolon (;).
- The program execution starts at the Main method.
- Unlike Java, file name could be different from the class name.