.NET Technology (PS02CDCA34)

Unit – 1 : The .NET Technology

- Introduction to .NET Framework
- Architecture of .NET framework
  – BCL (Base Class Library),
  – CLR (Common Language Runtime), etc.
- Types of applications supported by .NET Technology
- .NET Languages - introduction

Introduction to .NET Framework

- The **.NET Framework** is a software development platform that was introduced by Microsoft
- On 13 Feb 2002, Microsoft launched the first version of the .NET Framework, referred to as the **.NET Framework 1.0**.
- The current version of the **.Net framework is 4.8.**
- It provide a common platform to run an application that was built using the different language such as C#, VB.NET, Visual Basic, etc.
- It is also used to create a form based, console-based, mobile and web-based application or services that are available in Microsoft environment.

Introduction to .NET Framework

- IDEs (Integrated Development Environments) for .NET are:
- Visual Studio
  - Runs on Windows only.
  - Has extensive built-in functionality designed to work with .NET.
  - The Community edition is free for students, open-source contributors, and individuals.
- Visual Studio Code
  - Runs on Windows, macOS, and Linux.
  - Free and open source.
  - Extensions are available for working with .NET languages.

Architecture of .NET Framework

The architecture of the .net framework is based on .net component.

The Core of .NET Framework: FCL & CLR

- **Common Language Runtime**
  - Garbage collection
  - Language integration
  - Multiple versioning support
  - Integrated security
- **Framework Class Library (Base Class Library)**
  - Provides the core functionality:

    ASP.NET, Web Services, ADO.NET, Windows Forms, IO, XML, etc.

Architecture of .NET Framework

Common Language Runtime:

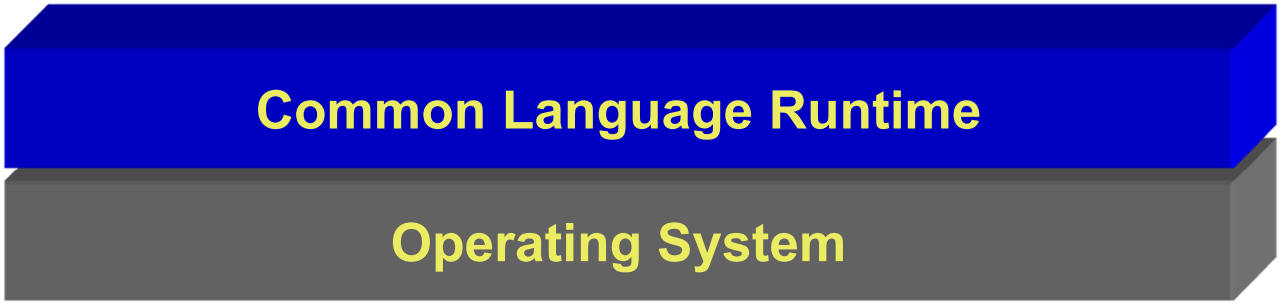
•It is works like a virtual component of the .NET Framework to executes the different languages program like c#, Visual Basic, etc.

•It helps to convert a **source code into the byte code**, and this byte code is known as **CIL (Common Intermediate Language) or MSIL (Microsoft Intermediate Language).**

•After converting into a byte code, a CLR uses a **JIT (Just – In –Time)** compiler at run time that helps **to convert a CIL or MSIL code into the machine or native code.**

•CLR manages code execution at runtime

•Memory management, thread management, etc.



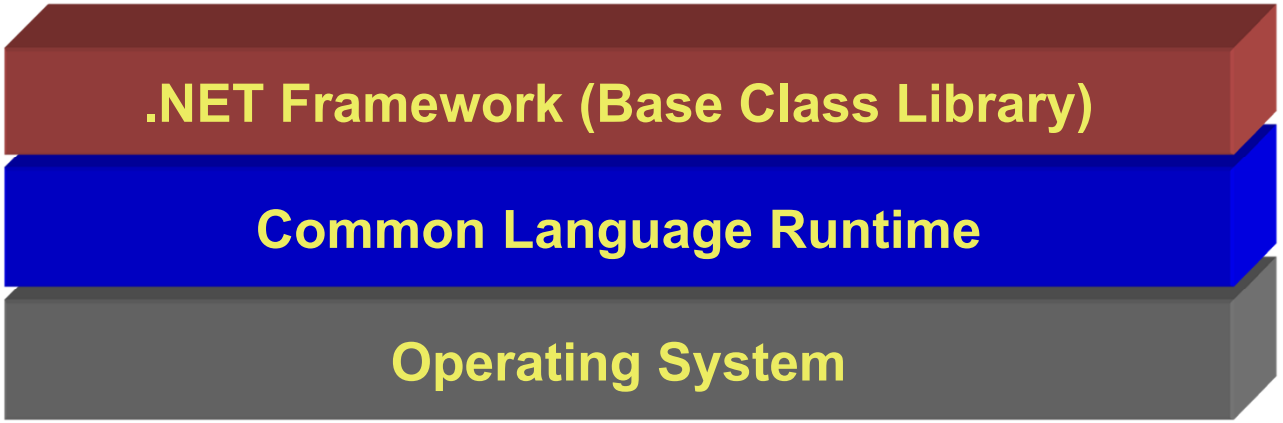
Architecture of .NET Framework

Framework Class Library (Base Class Library)

–The base class library has a rich collection of libraries features and functions that help to implement many programming languages in the .NET Framework, such as C #, F #, Visual C ++, and more.

–It contains either the DLL (Dynamic Link Library) or exe (Executable) file.

–It is the collection of predefined class and method that present in .Net.



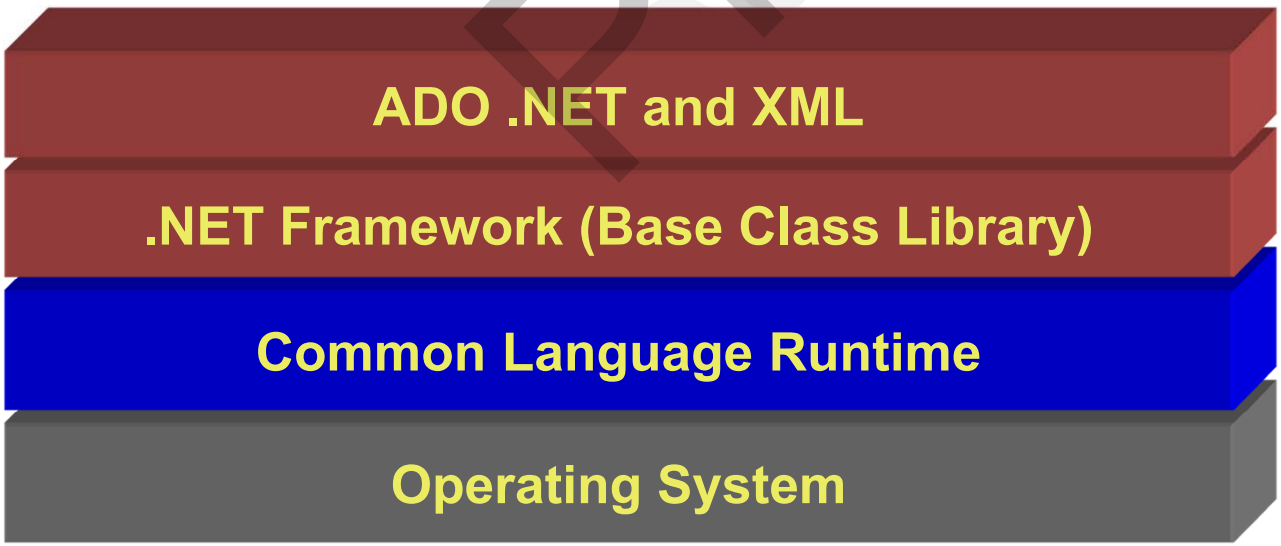
Architecture of .NET Framework

Data Access Layer

– Access relational databases

– Connected and Disconnected data model

– Work with XML



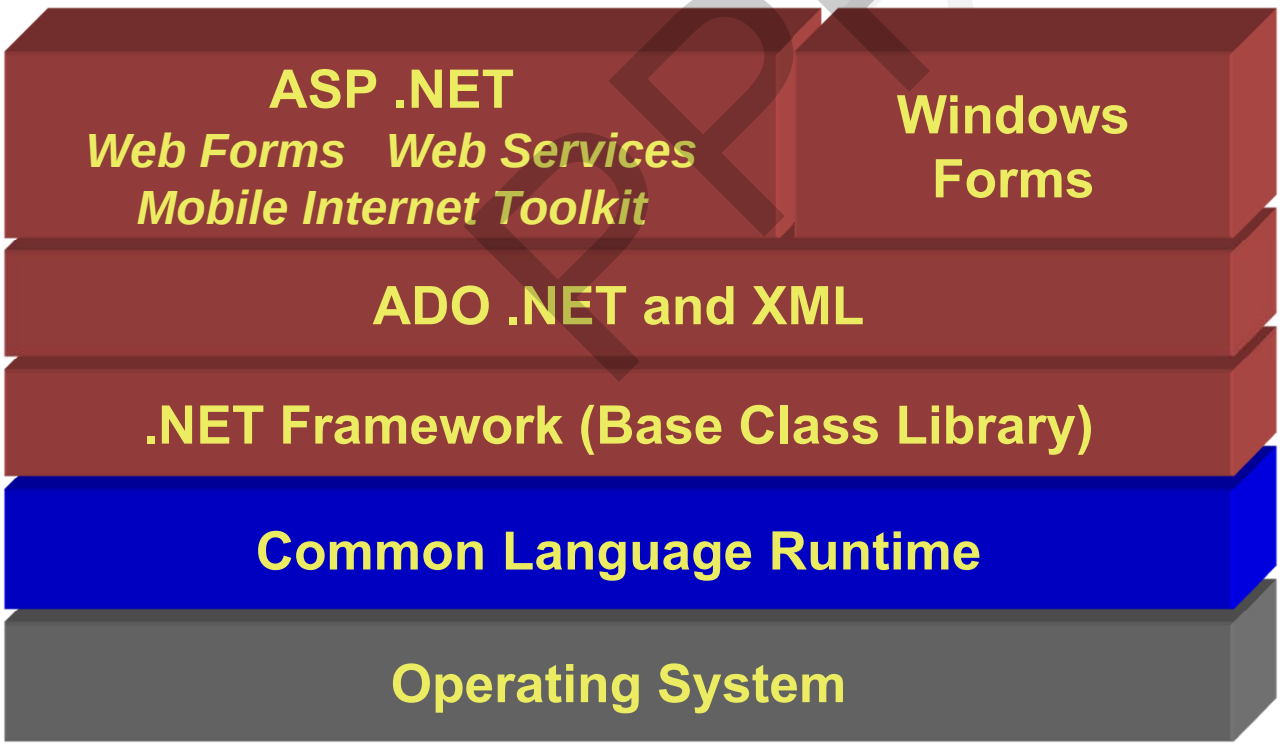
Architecture of .NET Framework

ASP.NET & Windows Forms

– Create application's front-end

– Web-based user interface,

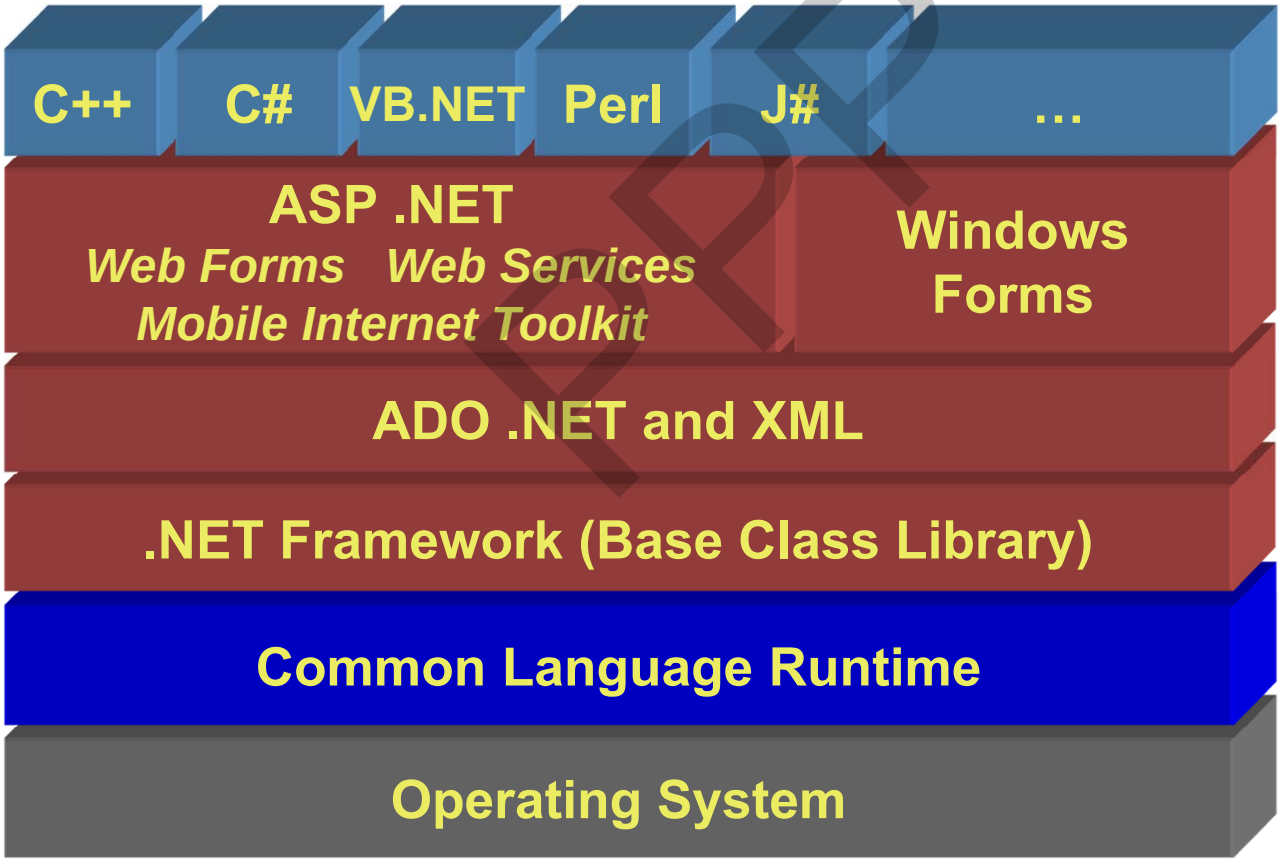
– Windows GUI, Web services etc.



Architecture of .NET Framework

Programming Languages

– Use your favorite language



Common Type System (CTS)

- **CTS** (Common Type System) **and CLS** (Common Language Specification) **are parts of .NET CLR** and are responsible for type safety with in the code. Both allow cross-language communication and type safety.
- **All .NET languages have the same primitive data types.**
- **Every language provides its own keywords for Data Types.**

  E.g.

  –**In C# int** a

  –**In VB integer** a
- But internally all the languages which run under .NET framework use the classes and structures available in CTS.
- **An *int* in C# is the same as an *integer* in VB.NET**
- **After compilation, use the same structure Int32 from CTS.**
- When communicating between modules written in any .NET language, the types are guaranteed to be compatible on the binary level
- All the structures and classes available in CTS are common for all .NET Languages & it support **language independence in .NET.**

Architecture of .NET Framework

Common Language Specification

- **It is a subset of CTS** (Common Type System)
- It defines a set of rules and regulations which should be followed by every language that comes under the .NET framework.

- A CLS language should be cross-language integration or interoperability.

- **E.g.** C# language terminate each statement with semicolon, whereas in VB.NET it is not end with semicolon but when these statements execute in .NET Framework, it provides a common platform to interact & share information with each other.

Architecture of .NET Framework

Common Language Specification

E.g.

Rule is that you cannot use multiple inheritance within .NET Framework.

•**C++ supports multiple inheritance** but; when you will try to use that C++ code within C#, it is not possible because **C# doesn't supports multiple inheritance**.

E.g.

Rule is that you cannot have members with same name with case difference only i.e. you cannot have add() and Add() methods.

•This easily works in **C#** because it **is case-sensitive** but when you will try to use that C# code in VB.NET, it is not possible because **VB.NET is not case-sensitive.**

Types of applications supported by .NET Technology

- Console Applications
  - A **console application** is a program designed to be used via a text-only computer interface, such as a text terminal, the command line interface of some operating systems (Unix, DOS, etc.)
- Windows GUI Applications (Windows Forms)
  - **Windows Forms** is a UI framework for building **Windows** desktop **apps**. It provides one of the most productive ways to create desktop **apps** based on the visual designer provided in Visual Studio. Functionality such as drag-and-drop placement of visual controls makes it easy to build desktop **apps**.
- Windows Presentation Foundation (WPF) Applications
  - It is a UI framework that creates desktop client **applications**. The **WPF** development platform supports a broad set of **application** development features, including an **application** model, resources, controls, graphics, layout, data binding, documents, and security.

- ASP.NET Applications
- Windows services
- Service-oriented Applications using Windows Communication Foundation (WCF)
- Workflow-enabled Applications using Windows Workflow Foundation (WF)

.NET Languages

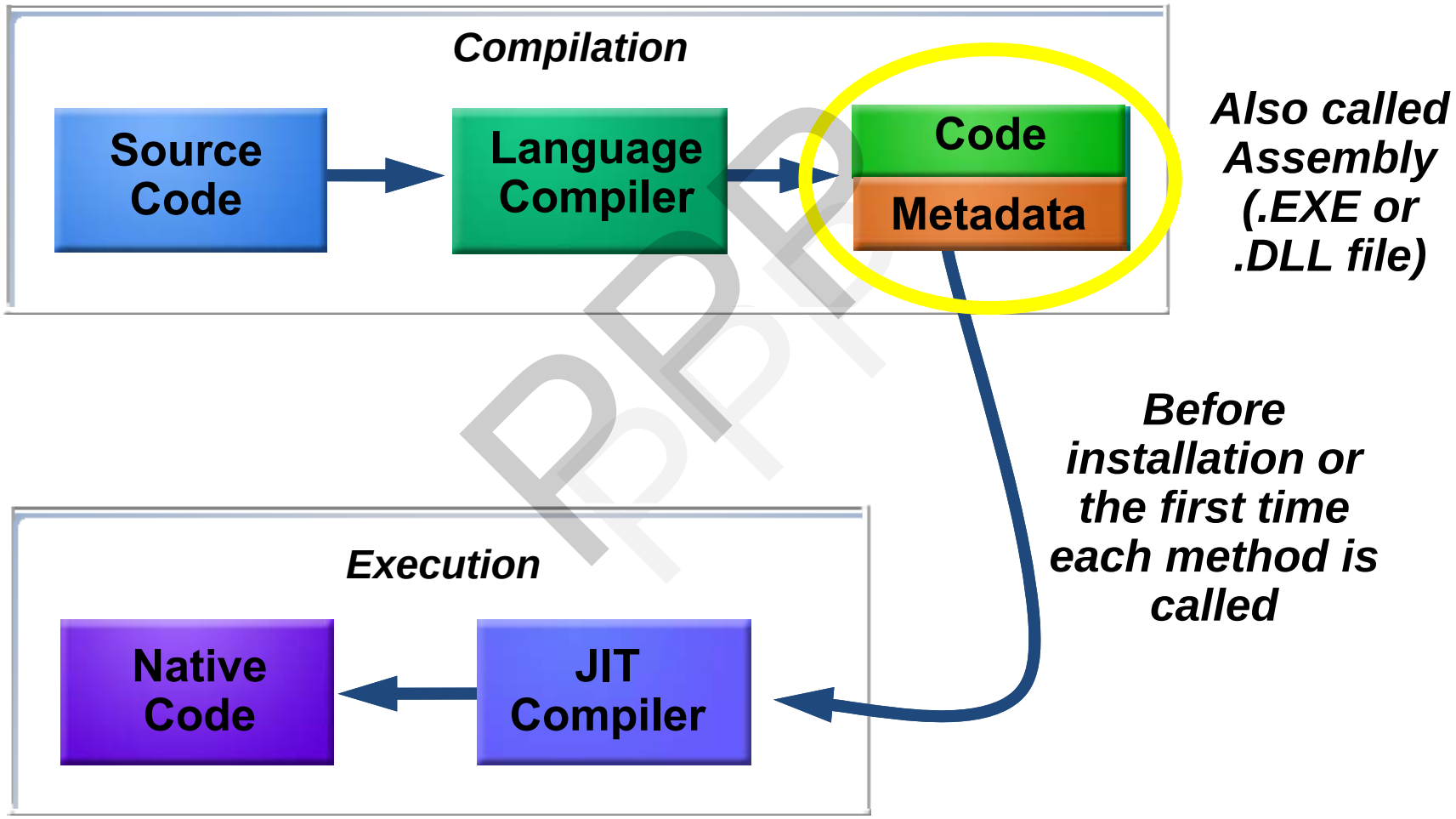
- Languages provided by Microsoft
  - C++, C#, J#, VB.NET, JScript
- Third-parties languages
  - Perl, Python, Pascal, APL, COBOL, Eiffel, Haskell, ML, Oberon, Scheme, Smalltalk…
- Advanced multi-language features
  - Cross-language inheritance and exceptions handling

C# Language – Example

```
using System;

class HelloWorld
{
  public static void main()
  {
    Console.WriteLine("Hello, world!");
  }
}
```

Code Compilation and Execution



Microsoft .NET languages

- **C#** - Microsoft's flagship .NET Framework language which bears similarities to the C++ and Java languages.
- **Visual Basic .NET** - A completely redesigned version of the Visual Basic language for the .NET Framework. This also includes Visual Basic 2005 (v8.0).
  - **VBx**, a dynamic version of Visual Basic .NET that runs on top of the Dynamic Language Runtime.
- **C++/CLI** and the deprecated **Managed C++** - A managed version of the C++ language.
- **J#** - A Java and J++ .NET transitional language.
- **JScript .NET** - A compiled version of the JScript language.
- **Windows PowerShell** - An interactive command line shell/scripting language which provides full access to the .NET Framework.
- **IronPython** - A .NET implementation of the Python programming language developed by Jim Hugunin at Microsoft.
- **IronRuby** - A dynamically compiled version of the Ruby programming language targeting the .NET Framework.
- **F#**, a member of the ML programming language family.

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
- VS.NET is not part of the .NET Framework
  - Not necessary to build or run managed code
  - The .NET Framework SDK includes command line compilers

VS.NET – Single Development Environment & Skill Set

- From Visual Studio.NET you can:
  - Write code
  - Design user interface
  - Study documentation
  - Debug
  - Test
  - Deploy
- Same tools for all languages
- Same tools for all platforms