.NET Framework 4.7 and C# 8.0

Lesson 01 : Introduction to .NET Framework 4.7

Capgemini

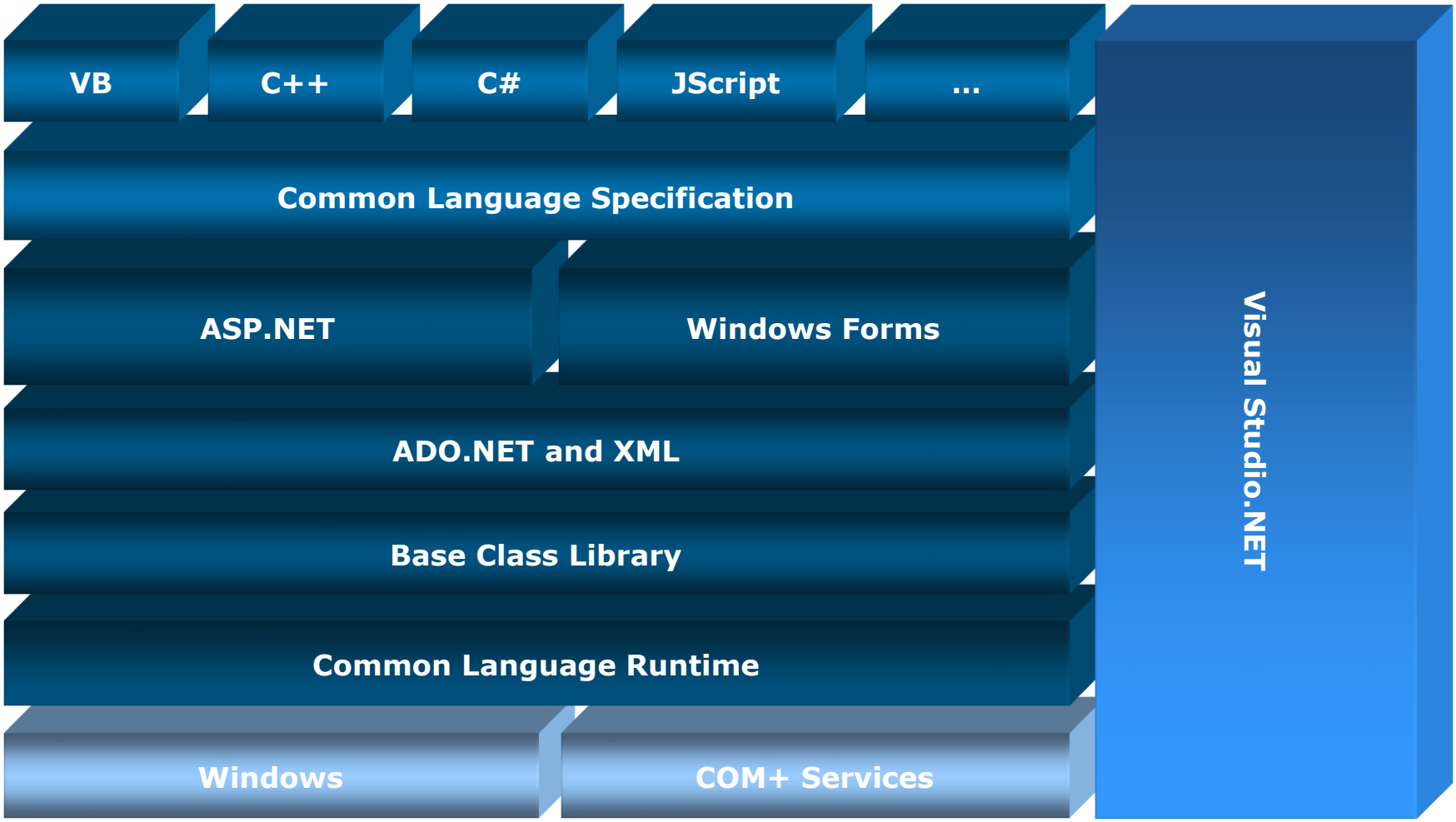
Lesson Objectives

- In this lesson, you will learn:
  - The .NET Framework
  - Features of Common Language Runtime
  - .NET Framework class Library
  - Managed Execution process
  - Understand the .NET Framework 4.7 stack
  - Know the new productivity enhancements in VS2019

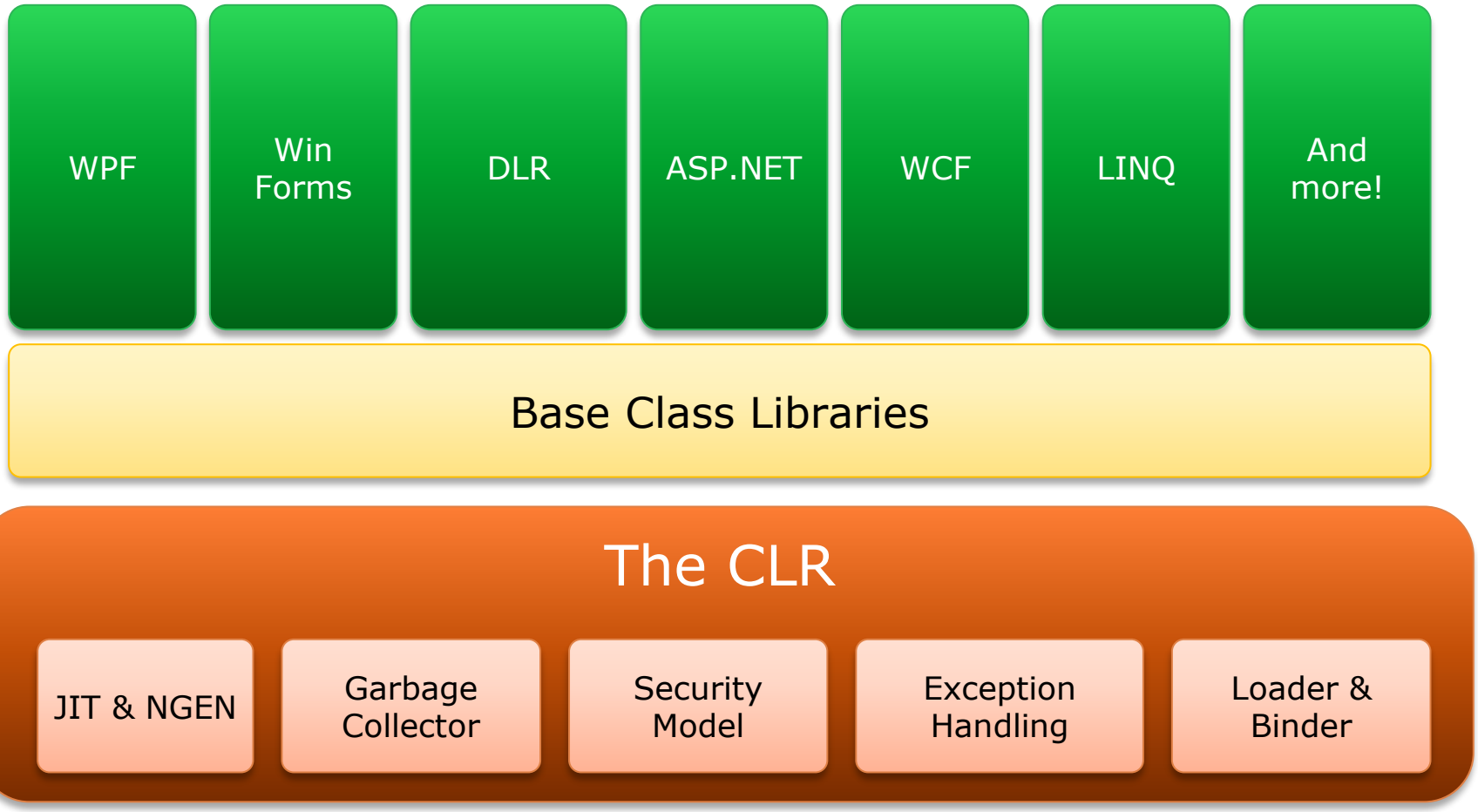
Introduction

- The .NET Platform is used to develop enterprise applications based on industry standards.
- The .NET platform was introduced to offer a much more powerful, more flexible, and simpler programming model than COM
- .NET Framework is a fully managed, protected, simplified, feature rich application execution environment
  - It is OS independent and hardware independent
  - Extensively uses Industry standards like HTTP, SOAP, XML, XSD.
  - Easily maintainable due to simplified deployment and version management
  - A platform to build Web services, Multi Threaded Applications, Windows Services, Rich Internet Applications as well as Mobile Application.
  - Provides seamless integration to a wide variety of languages.

Diagrammatic Representation



The .NET Framework 4.7



Concept of .NET

- .NET implementation is language agnostic
- .NET implementation enables multi-language development
- Multi-language development works with
  - Common Type System (CTS)
  - Common Language Specification (CLS)

Common Type System (CTS)

- The common type system defines how types are declared, used, and managed in the common language runtime.
- CTS is an important part of the runtime's support for cross-language integration
- The common type system performs the following functions:
  - Establishes a framework that helps enable cross-language integration, type safety, and high-performance code execution.
  - Provides an object-oriented model that supports the complete implementation of many programming languages.
  - Defines rules that languages must follow, which helps ensure that objects written in different languages can interact with each other.
  - Provides a library that contains the primitive data types (such as Boolean, Byte, Char, Int32, and UInt64) used in application development.

Common Language Specification (CLS)

- CLS defines a set of features that are needed by many common applications.
- It also provides a sort of recipe for any language that is implemented on top of .NET on what it needs to support.
- CLS is a subset of the CTS.
- To fully interact with other objects written in any language, objects must expose to callers only those features that are common to all languages.
- This common set of features is defined by the Common Language Specification (CLS), which is a set of rules that apply to generated assemblies.
- Some rules from CLS
  - Array should be start with zero index
  - Interface can have properties, events and virtual methods. It cannot have static methods, fields, methods with implementation
  - Constructor of derived class must call constructor of base class

Class Libraries

- Class libraries are the shared library concept for .NET.
- Class libraries are described using the .NET Assembly file format.
- There are three types of class libraries that you can use:
  - Platform-specific class libraries
    - have access to all the APIs in a given platform (for example, .NET Framework, Xamarin iOS), but can only be used by apps and libraries that target that platform.
  - Portable class libraries
    - have access to a subset of APIs, and can be used by apps and libraries that target multiple platforms.
  - .NET Standard class libraries
    - are a merger of the platform-specific and portable library concept into a single model that provides the best of both.

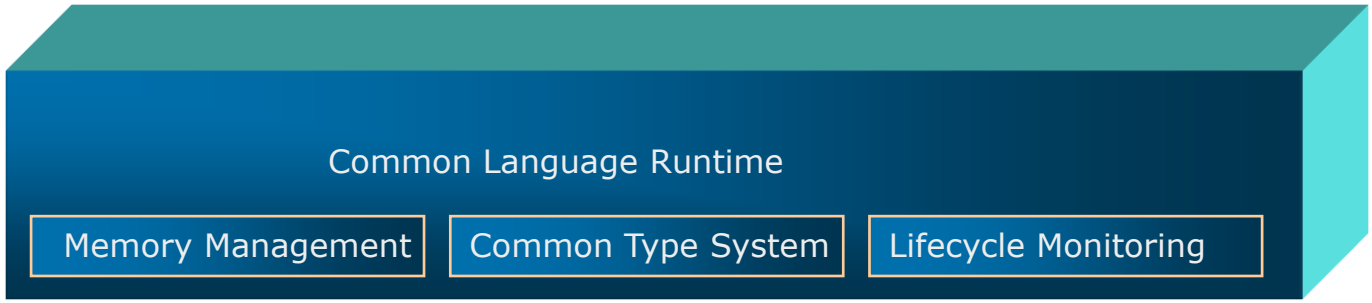
Concept of .NET

- .NET Framework class library is object-oriented collection of reusable classes.
- You use them to develop applications like:
  - Command-line applications
  - Graphical User Interface (GUI) based Desktop applications
  - Web Applications
  - Web Services
  - Distributed Applications

Concept of .NET

- It includes the following components:
  - Common Language Runtime (CLR)
  - Rich Class Libraries (FCL/BCL)
    - For e.g.: API for ADO.NET, XML, Threading, Windows Development etc.

Diagrammatic Representation

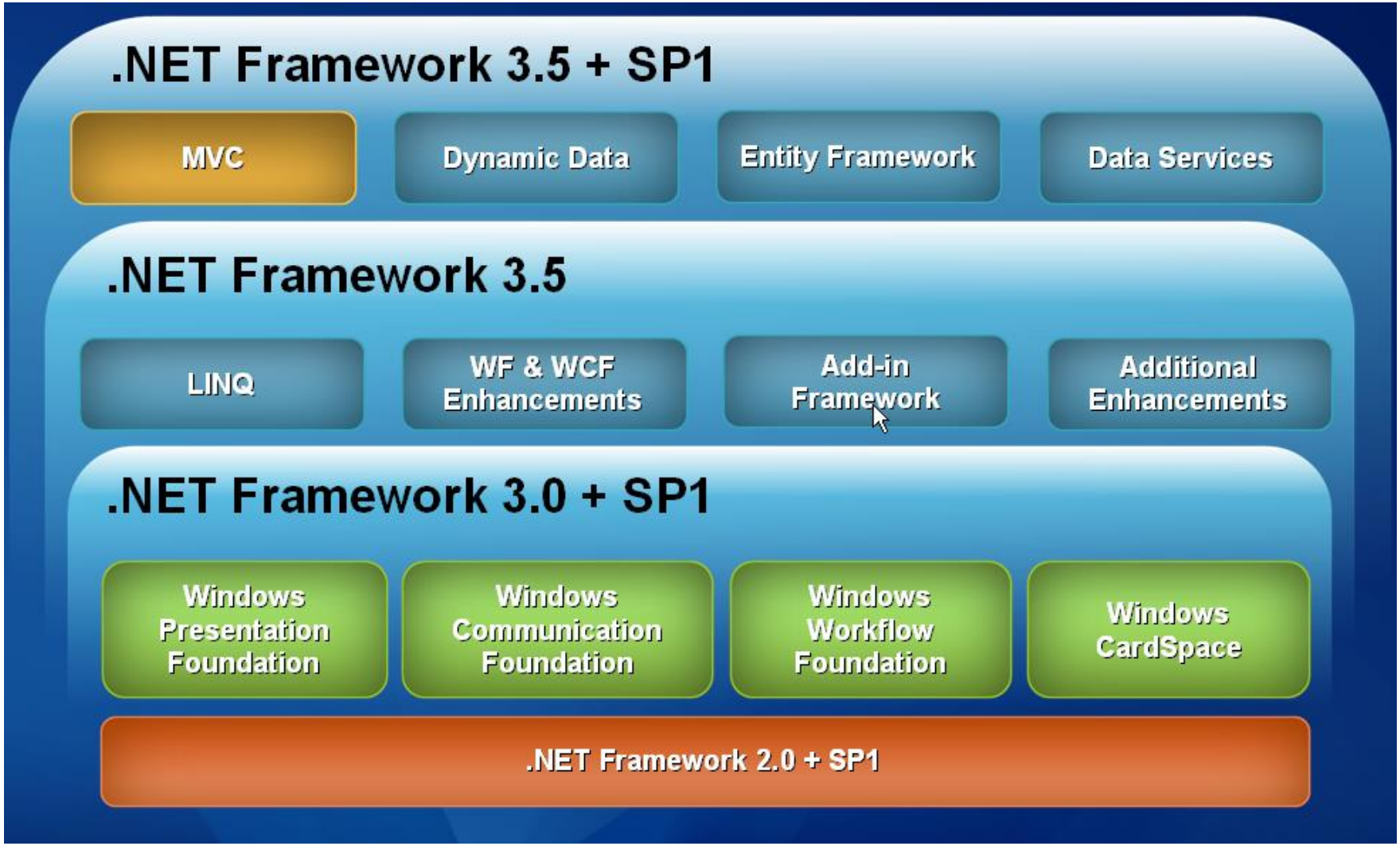


- Role of CLR:
  - CLR manages the following:
    - Running of code
    - Memory management
    - Compilation of code
    - Provides garbage collection, error handling
    - Code access security for semi-trusted code

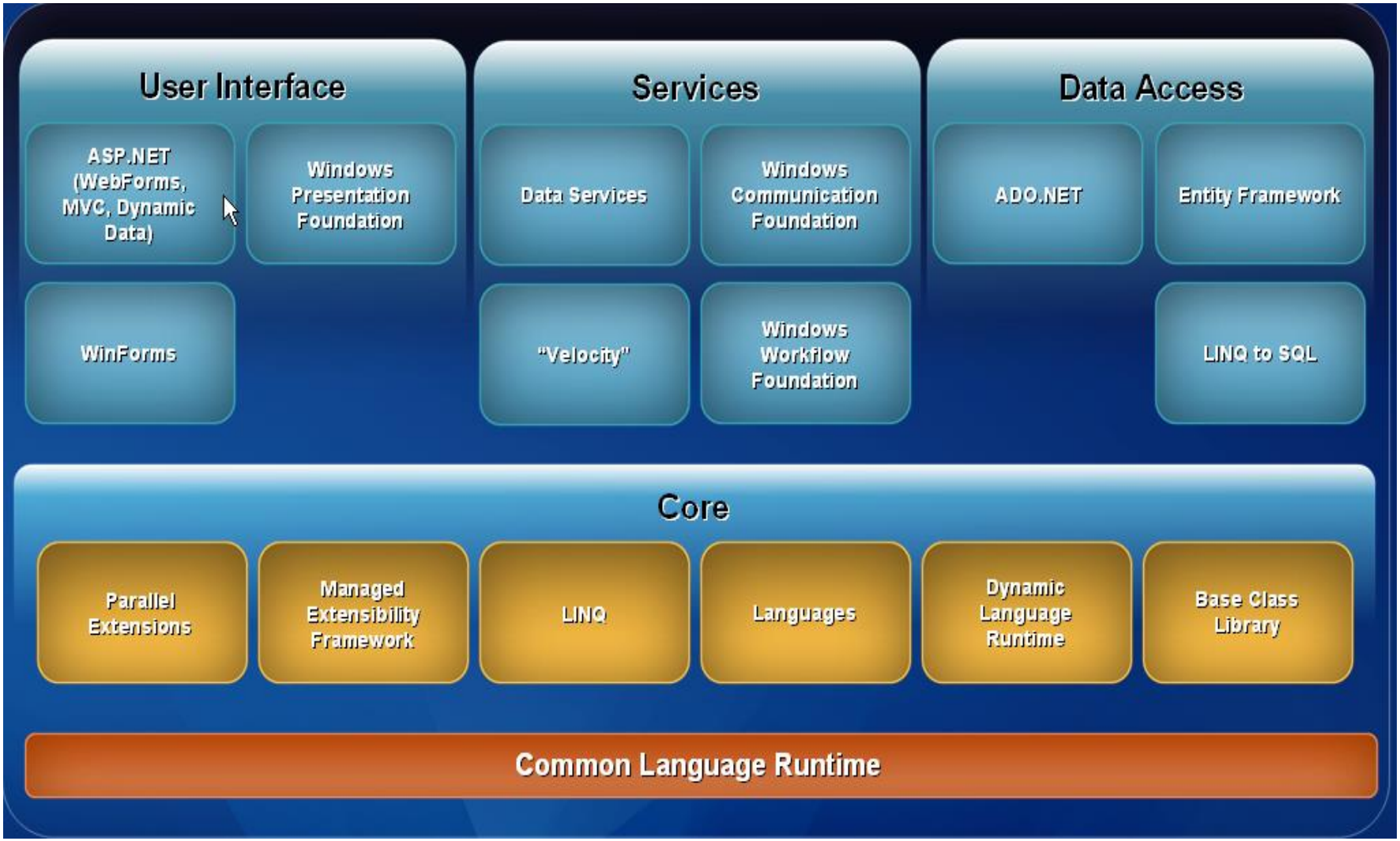
.NET Evolution

Year	CLR	.NET	Visual Studio	C#	VB
2002	1.0	1.0	2002	1.0	7.0
2003	1.1	1.1	2003	1.0	7.0
2005	2.0	2.0	2005	2.0	8.0
2006	2.0	3.0	.NET 3.0 ext.	2.0	8.0
2007	2.0	3.5	2008	3.0	9.0
2010	4	4	2010	4.0	10.0
2012	4	4.5	2012	5.0	11.0
2015	4	4.5.2	2015	6.0	12.0
2017	4	4.6.2	2017	6.0	12.0
2018	4	4.7.2	2017	7.0	13.0
2019	4	4.8	2019	8.0	14.0

The.NET Framework 3.5 Layer



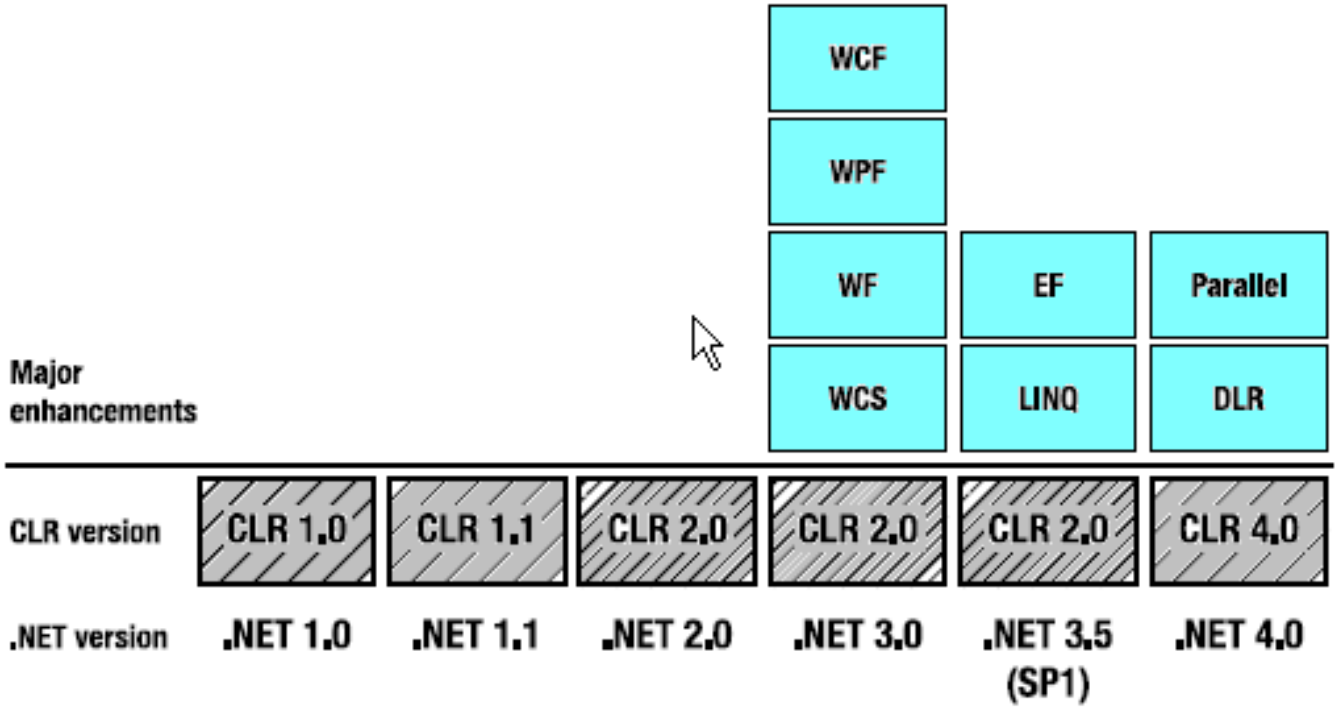
.NET Framework 4.0



CLR and BCL Changes

- The last two releases of.NET (3.0 and 3.5) have been additive releases building on top of the functionality available in CLR version 2.0
- .NET 4.0 however has a new version of the CLR
- Installing .NET 4.0 will not affect existing .NET applications running on previous versions of the framework

New CLR 4.0

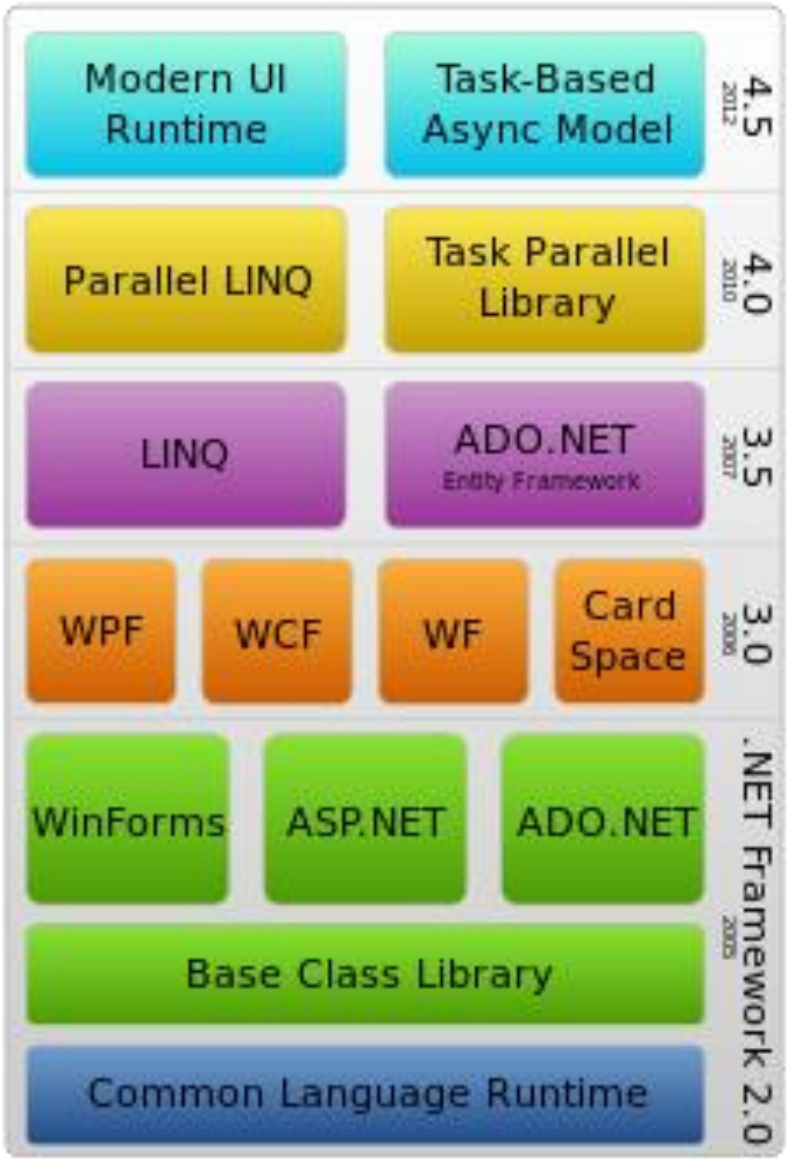

WCF
WPF
WF
EF
Parallel
WCS
LINQ
DLR
Major enhancements
CLR version
CLR 1.0
CLR 1.1
CLR 2.0
CLR 2.0
CLR 2.0
CLR 4.0
.NET version
.NET 1.0
.NET 1.1
.NET 2.0
.NET 3.0
.NET 3.5 (SP1)
.NET 4.0

Maturation of Existing Technologies - .NET 4

- Windows Workflow Foundation (WF) and Windows Communication Foundation (WCF) are much more closely integrated
- WF has a radical change with a much-improved designer, introduction of new activities, and easier customization
- WCF becomes simpler to use and also introduces new service discovery functionality
- WPF has some great additions, with an improved designer, multi-touch, and Windows 7 task bar support
- Entity Framework architecture is improved
- Enhancements in ASP.NET

.NET Framework 4.5 Core Enhancement

- Windows 8 Support
- Making Asynchronous programming as easy as possible
- Improved Memory Management with parallel background Garbage Collection and better concurrency.
- NuGet – A preferred Distribution and update Mechanism
- Supporting emerging standards and Technologies



The .NET Framework Stack

Why .NET 4.5 and VS2012

- VS2012 and .NET 4.5 Themes:
  - Changes and Improvements in the BCL
  - Maturation of existing technologies
  - Making Async Programming as Easy as Possible
  - Supporting Emerging Standards and Technologies
  - Promoting NuGet as the Preferred Distribution and Update Mechanism
  - Making It Easy to Develop on Different Devices and Other Platforms

.NET Framework 4.6.0 – New Features

- Open-source
- Transferring source to GitHub
- CLR, Just-In-Time Compiler (JIT), Garbage Collector (GC), and core .NET base class libraries
- .NET Core Framework on Linux and OSX (Mac)
- .NET Compiler Platform ("Roslyn") provides open source C# and Visual Basic compilers with rich code analysis APIs
- 64-bit JIT Compiler for Managed Code

.NET Framework 4.6.0 – Built-In Support for IoC & DI

- Dependency Injection - Technique whereby one object supplies the dependencies of another object.
  - A dependency is an object that can be used (a service).
  - An injection is the passing of a dependency to a dependent object (a client) that would use it.
- DI Allows the Removal of Hard-Coded Dependencies
  - Which makes it possible to change them, at rum-time or compile-time.
- Built-In Support for Dependency Injection in ASP.NET Core
  - ASP.NET Core applications can leverage built in framework support for implementing dependency injection

.NET Framework 4.6.1 – New Features

- Cryptography: Support for X509 certificates containing ECDSA
- ADO.NET Improvement
- WPF Improvement
- Profiling
- (NGEN) PDBs

.NET Framework 4.6.2 – New Features

- ASP.NET Area Features
  - Improved support for localized error messages in data annotation validators
  - Async support for session-state store providers
  - Async support for output-cache providers
- Character Encoding - Unicode Standard, Version 8.0.0.
- Cryptography - Support for X509 certificates containing FIPS 186-3 DSA
- ADO.NET SQLClient - Connection pooling and timeouts with Azure SQL databases

.NET Framework 4.6.2 – New Features (Cont.…)

- Windows Communication Foundation
  - WCF transport security support for certificates stored using CNG
  - Better support for multiple daylight-saving time adjustment rules by the DataContractJsonSerializer class
  - Support for preserving a UTC time when serializing and deserializing with the XMLSerializer class
  - NetNamedPipeBinding best match
  - SSL 3.0 is not a default protocol

.NET Framework 4.7 – New Features (Cont.…)

- Windows Presentation Framework
  - Soft keyboard support
  - Group sorting
  - Per-monitor DPI
- ClickOnce improvement - Support TLS 1.1 & TLS 1.2
- Converting Windows Forms and WPF apps to UWP apps

.NET CORE

- NET Core is a general-purpose development platform maintained by Microsoft and the .NET community.
- It is cross-platform, supporting Windows, macOS and Linux, and can be used in device, cloud, and embedded/IoT scenarios.

.NET Core is composed of

- A .NET Runtime which provides a type system, assembly loading, a garbage collector, native interop and other basic services.
- The 'dotnet' app host, which is used to launch .NET Core apps. It selects the runtime and hosts the runtime, provides an assembly loading policy and launches the app. The same host is also used to launch SDK tools in much the same way.
- A set of framework libraries, which provide primitive data types, app composition types and fundamental utilities

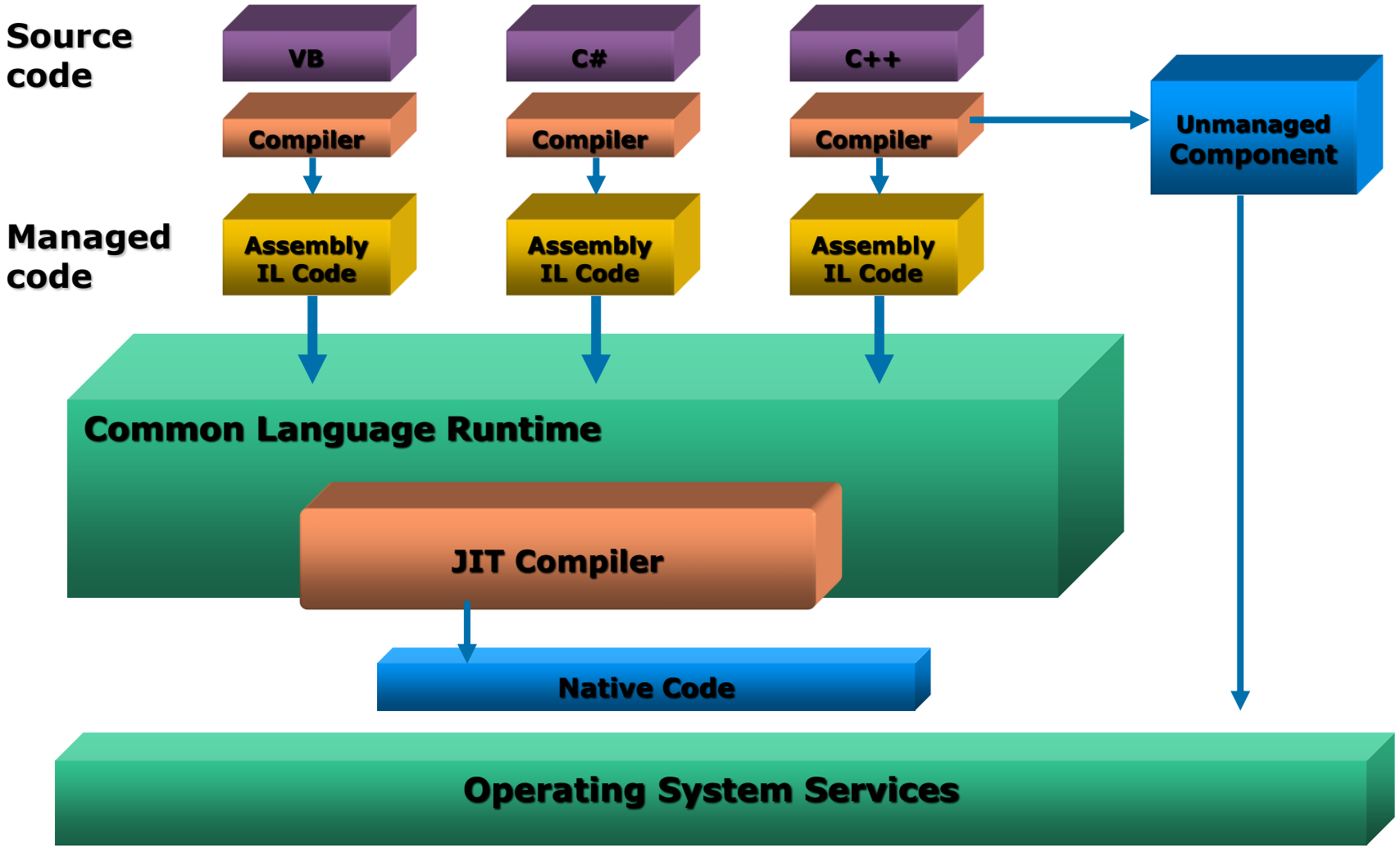
.NET Core Evolution

Year	.NET Core	Visual Studio
2016	.NET Core 1.0	Visual Studio 2015 Update 3
2016	.NET Core 1.1	Visual Studio 2017 Version 15.0
2017	.NET Core 2.0	Visual Studio 2017 Version 15.3
2018	.NET Core 2.1	Visual Studio 2017 Version 15.7
2018	.NET Core 2.2	Visual Studio 2019 Version 16.0
2019	.NET Core 3.0	Visual Studio 2019 Version 16.3
2019	.NET Core 3.1	Visual Studio 2019 Version 16.4

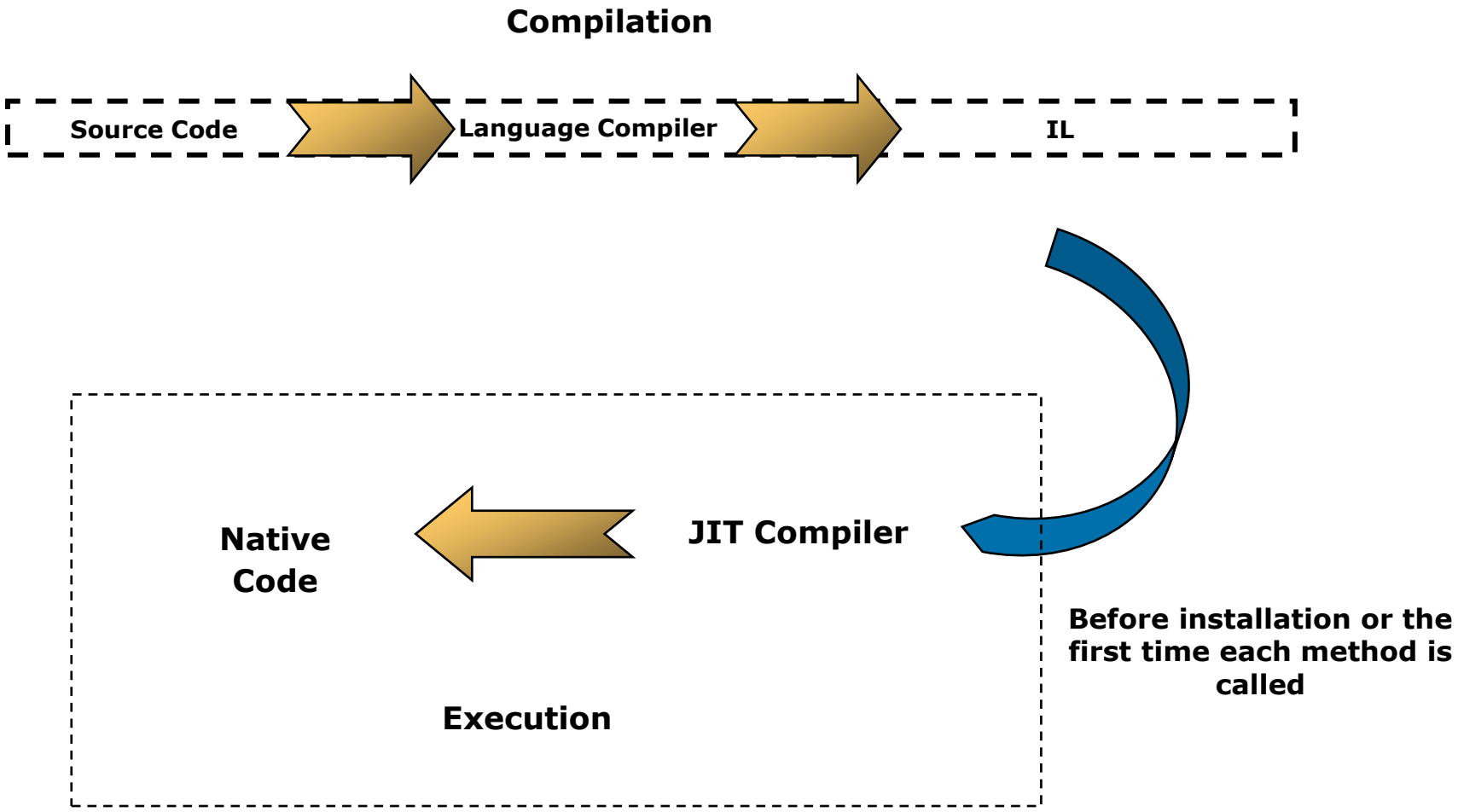
Difference in .NET Framework & .NET CORE

- App-Models
- API
- SubSystems
- Platforms
- Open Source

CLR: Execution Model



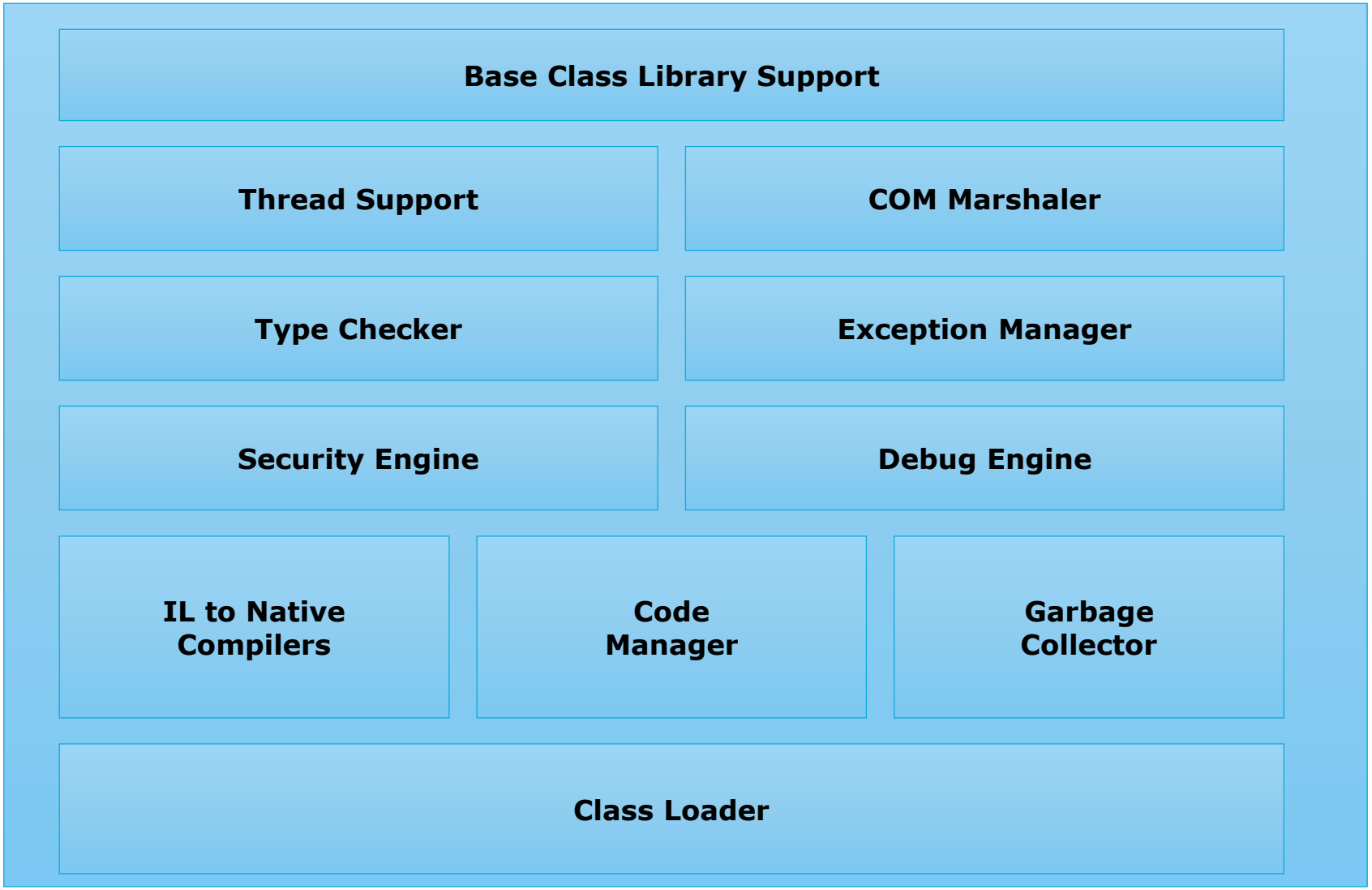
Diagrammatic Representation



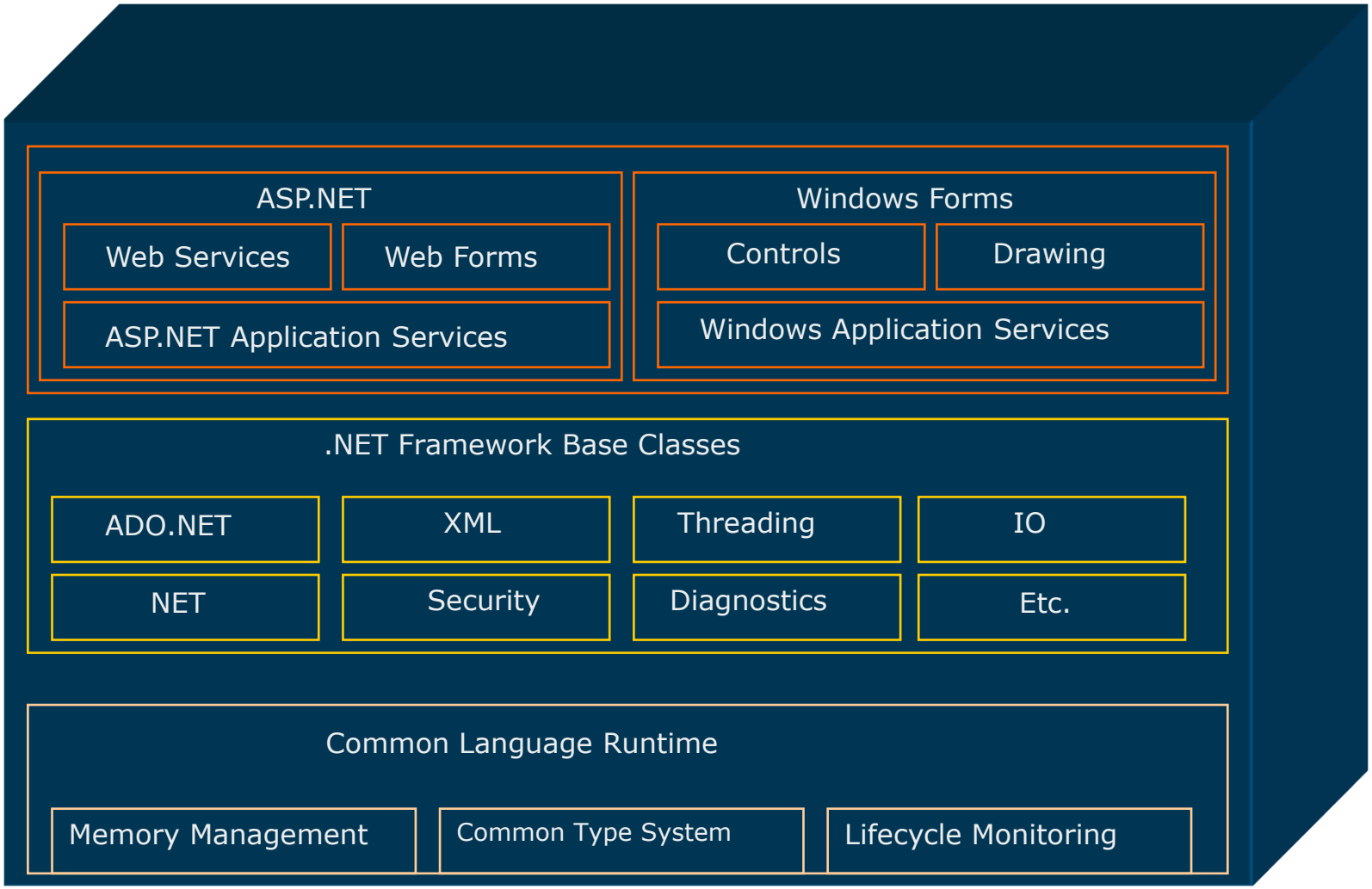
Concept of .NET

- Assembly
  - When you compile an application, the CIL code created is stored in an *assembly*.
  - Assemblies include both executable application files that you can run directly from Windows without the need for any other programs (these have a .exe file extension) and libraries (which have a .dll extension) for use by other applications.
  - It is defined as the Smallest Unit of Deployment , Versioning and Sharing
- IL
  - The CPU independent Set of Binary Instructions generated by .NET Language Compiler
- Managed Code
  - Code written using the .NET Framework is *managed* when it is executed (a stage usually referred to as *runtime*).
  - This means that the CLR looks after your applications by managing memory, handling security, allowing cross-language debugging, and so on.

Common Language Runtime

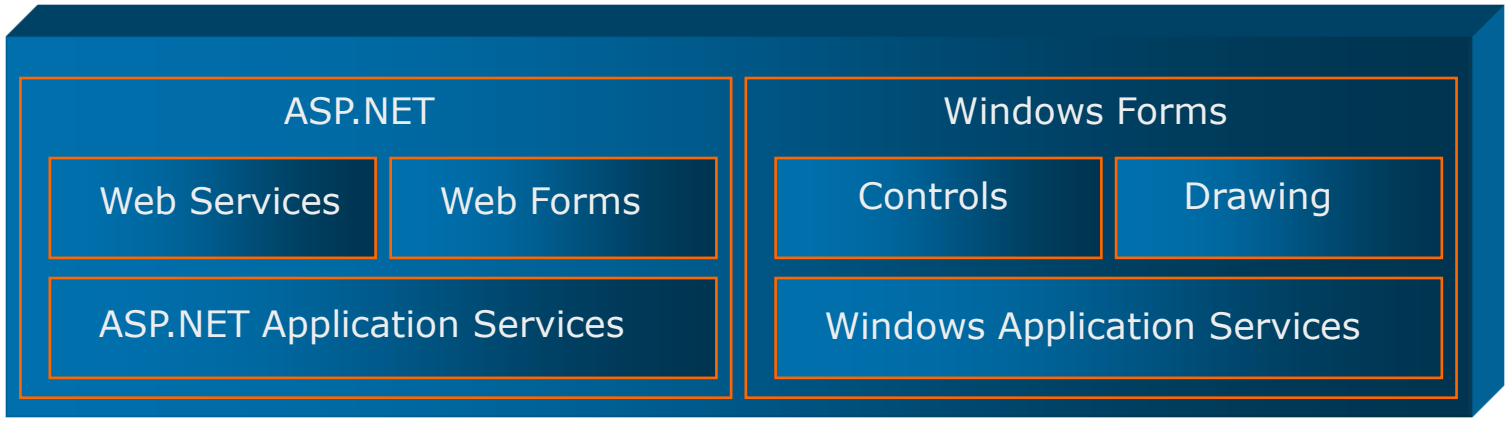


Diagrammatic Representation



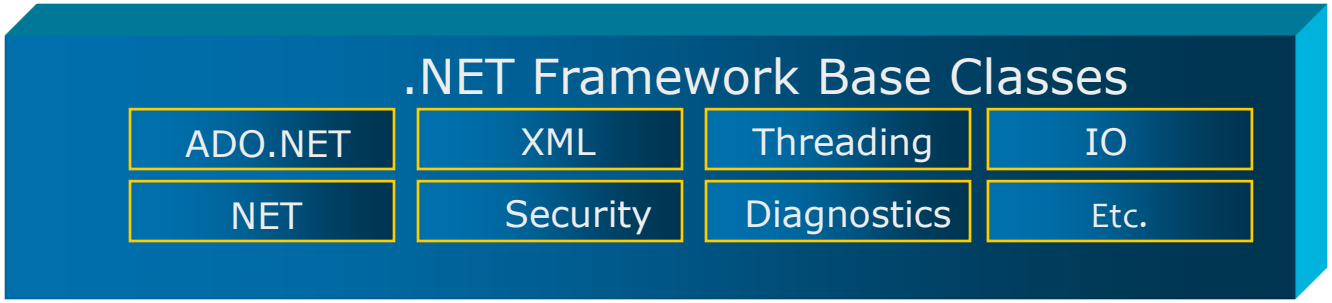
Explanation

- The top layer consists of user and program interfaces.
- Windows Forms, WPF can be used for Desktop Development
- Asp. Net Web Forms for the Web Development.



Explanation

- The middle layer consists of classes for next generation of system services.
- These are ADO.NET and XML as well as many others.
- Under the control of Framework, all components are universally available.
- Web Services, Remoting could be used.



Explanation

- .NET Framework provides simplicity during deployment:
  - No registration required
    - Code is completely self-describing
    - Simply copy components to app dir
  - Zero-impact install
    - Installing one app will not affect another
  - Side-by-side execution
    - Multiple component versions can co-exist

Explanation

- Namespaces are the way by which .NET avoids name clashes between classes.
- .NET requires all types to be defined in a namespace.
- The System namespace is the root namespace that contains the fundamental types of the .NET Framework.
  - This namespace contains the Object class that is the root of the inheritance hierarchy, primitive and extended types, and many other classes.

Demo

- Demo on how the various .NET framework versions are installed on a machine.

Summary

- In this lesson, you have learnt about:
  - Microsoft .NET and its main components
    - Development tools and .NET languages
    - .NET Enterprise servers
    - The Microsoft .NET Framework
    - The .NET Framework class library, Common Language Runtime, and web services
  - The Common Language Runtime
    - The components of the Common Language Runtime
    - The concept of managed code, which includes compiler-generated code in Microsoft Intermediate Language, metadata, as well as Just-in-Time compiling into the native, platform-dependent code.

Review Questions

- Question 1: The ___ is the foundation of the .NET Framework.
- Question 2: Code that targets the runtime is known as ___; code that does not target the runtime is known as ___.
- Question 3: In .NET, the applications are Compiled to a common language called ___.
- Question 4: ___ component manages the allocation and release of memory for the application, and automatically reclaims unused memory.