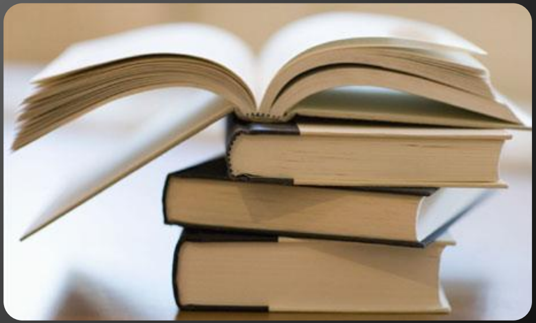
.NET Framework Overview

.NET Framework, CLR, MSIL, Assemblies, CTS, etc.

Abdallah MOUJAHID
PMP®, COBIT® V5 , ITIL® V3, ISO 27002

Table of Contents

Table of Contents (2)



.NET Framework

Microsoft's Platform for
Application Development

What is the .NET Platform?

- The .NET platform
  - Microsoft's platform for software development
  - Unified technology for development of almost any kind of applications
    - GUI / Web / mobile / server / cloud / etc.
- .NET platform versions
  - .NET Framework
  - .NET Compact Framework

What is .NET Framework?

- .NET Framework
  - An environment for developing and executing .NET applications
  - Unified programming model, set of languages, class libraries, infrastructure, components and tools for application development
  - Environment for controlled execution of managed code
- It is commonly assumed that
  - .NET platform == .NET Framework

.NET Framework Components

- Common Language Runtime (CLR)
  - Environment for controlled execution of programmed code – like a virtual machine
  - Executes .NET applications
- Framework Class Library (FCL)
  - Standard class library for .NET development
  - Delivers basic functionality for developing: XML, ADO.NET, LINQ, ASP.NET, WPF, WCF, WWF, Silverlight, Web services, Windows Forms, ...
- SDK, compilers and tools

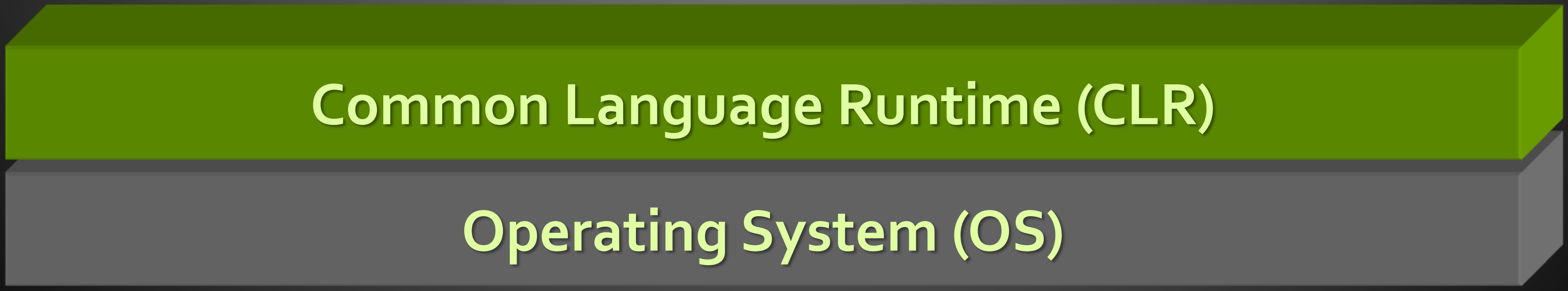
.NET Framework Architecture

- The OS manages the resources, the processes and the users of the machine
- Provides to the applications some services (threads, I/O, GDI+, DirectX, COM, COM+, MSMQ, IIS, WMI, …)
- CLR is a separate process in the OS

Operating System (OS)

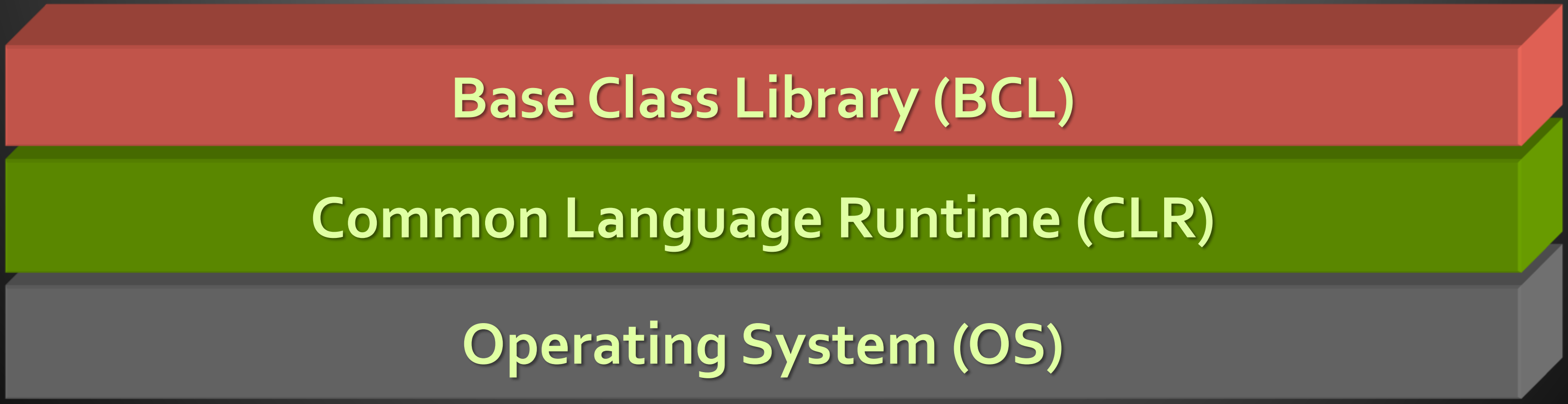
.NET Framework Architecture (2)

- CLR manages the execution of the.NET code
- Manages the memory, concurrency, security, ...



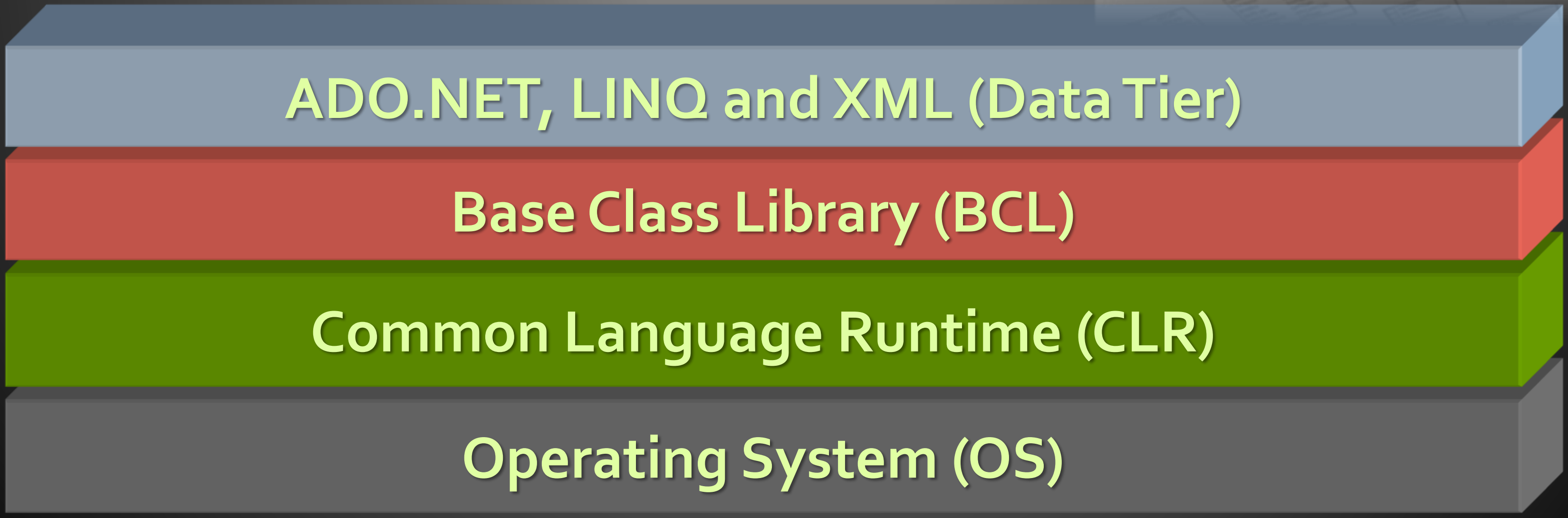
.NET Framework Architecture (3)

- Rich object-oriented library with fundamental classes
- Input-output, collections, text processing, networking, security, multi-threading, …



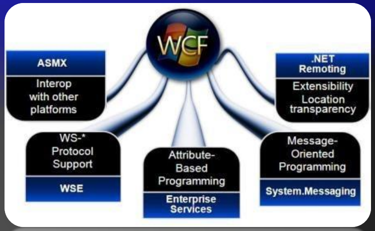
.NET Framework Architecture (4)

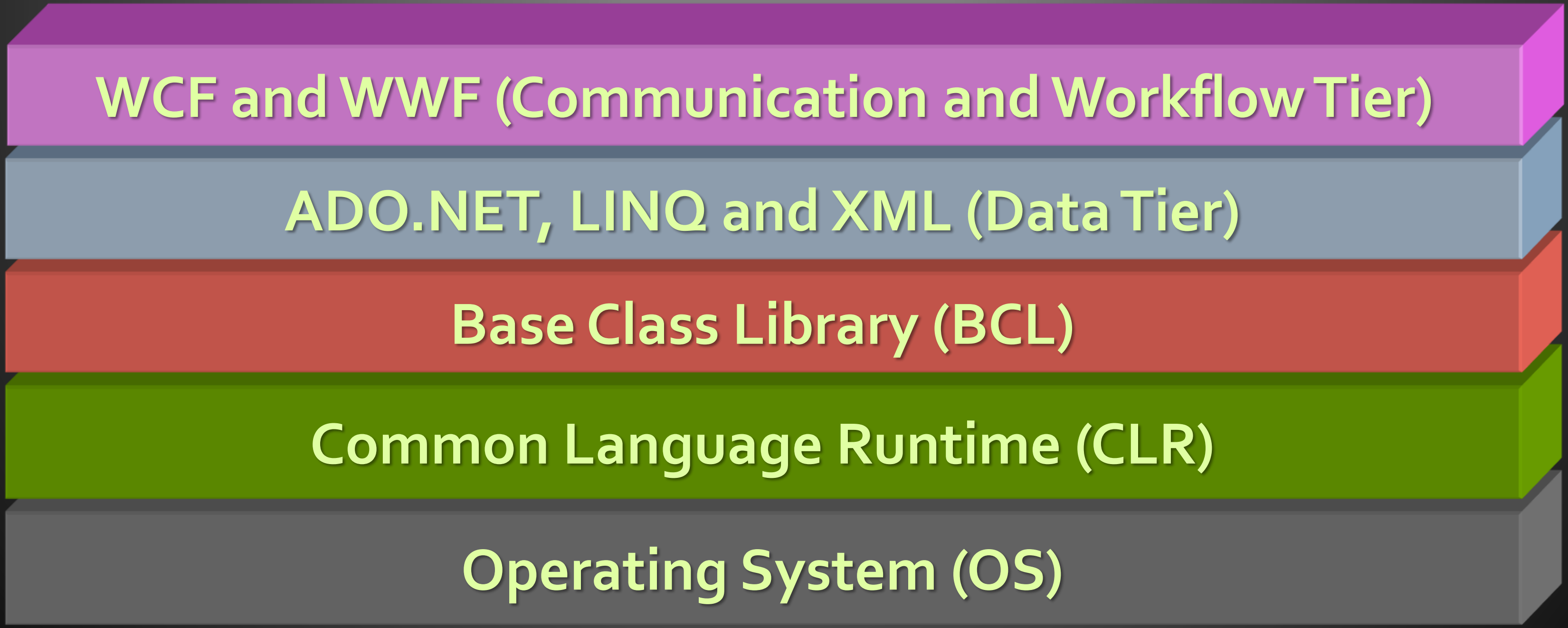
- Database access
- ADO.NET, LINQ, LINQ-to-SQL and Entity Framework
- Strong XML support


ADO.NET, LINQ and XML (Data Tier)
Base Class Library (BCL)
Common Language Runtime (CLR)
Operating System (OS)

.NET Framework Architecture (5)

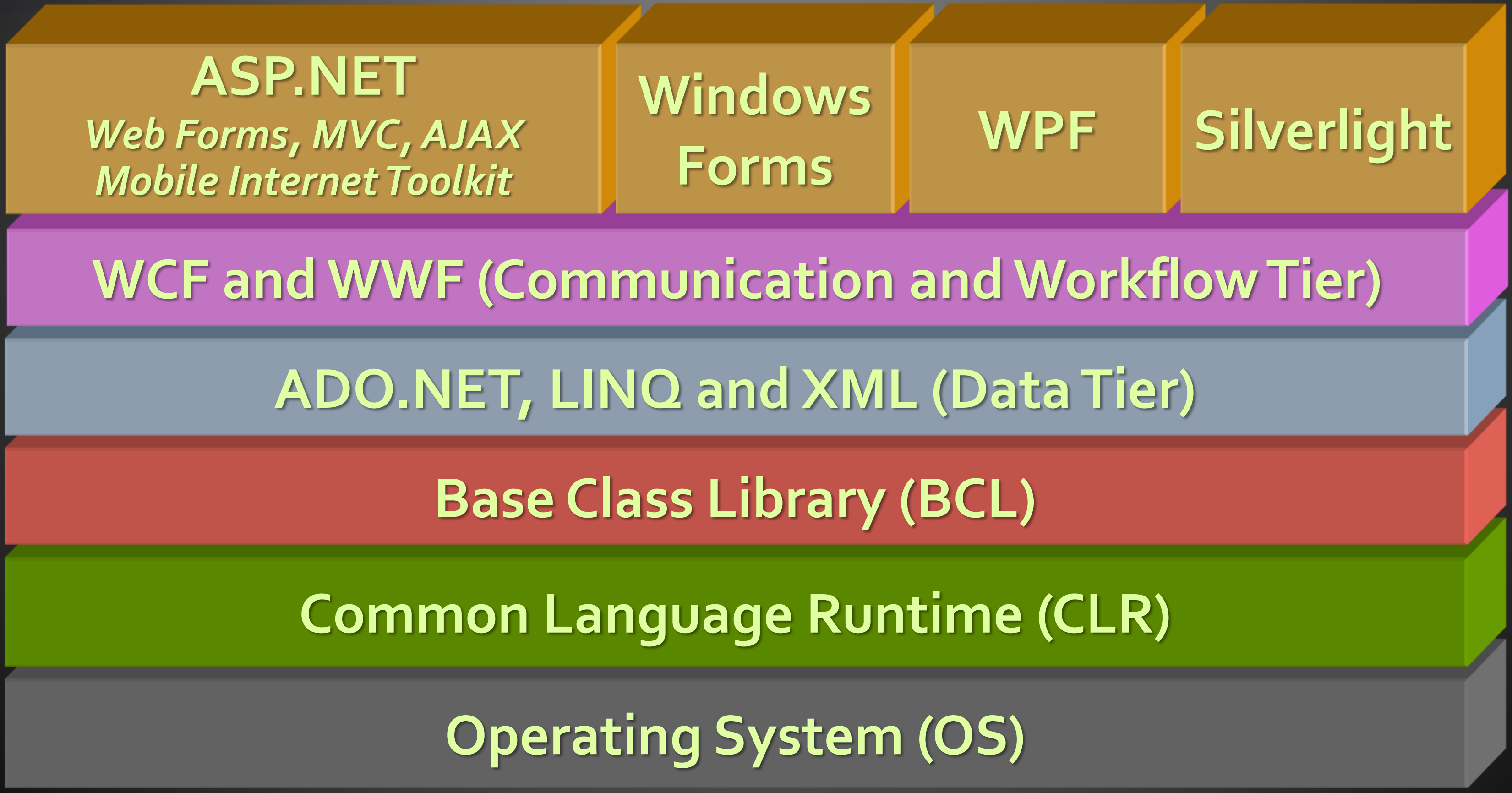
- Windows Communication Foundation (WCF) and Windows Workflow Foundation (WWF) for the SOA world


WCF
ASMX
Interop with other platforms
.NET Remoting
Extensibility Location transparency
WS-* Protocol Support
WSE
Attribute-Based Programming
Enterprise Services
Message-Oriented Programming
System.Messaging


WCF and WWF (Communication and Workflow Tier)
ADO.NET, LINQ and XML (Data Tier)
Base Class Library (BCL)
Common Language Runtime (CLR)
Operating System (OS)

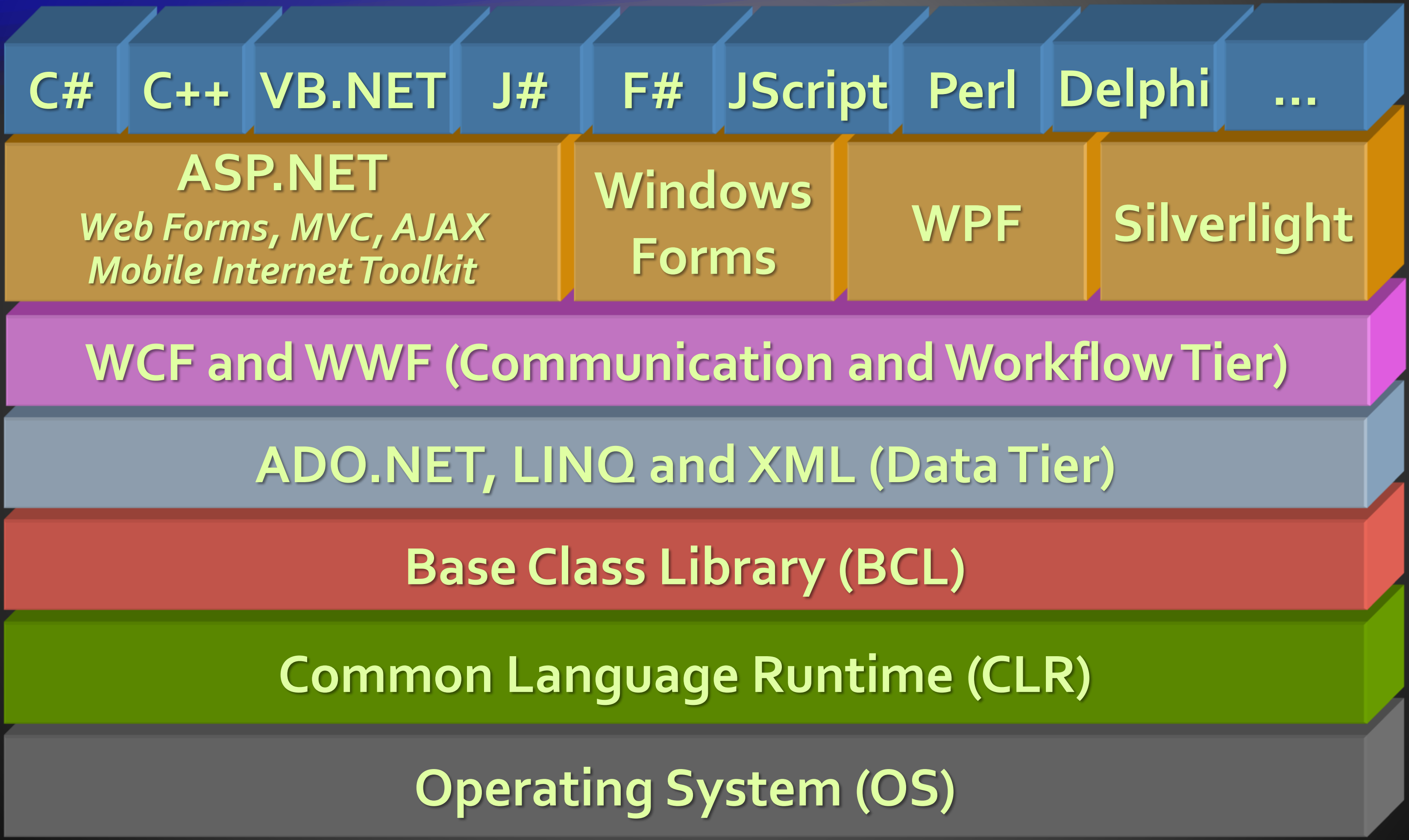
.NET Framework Architecture (6)

- User interface technologies: Web based, Windows GUI, WPF, Silverlight, mobile, …

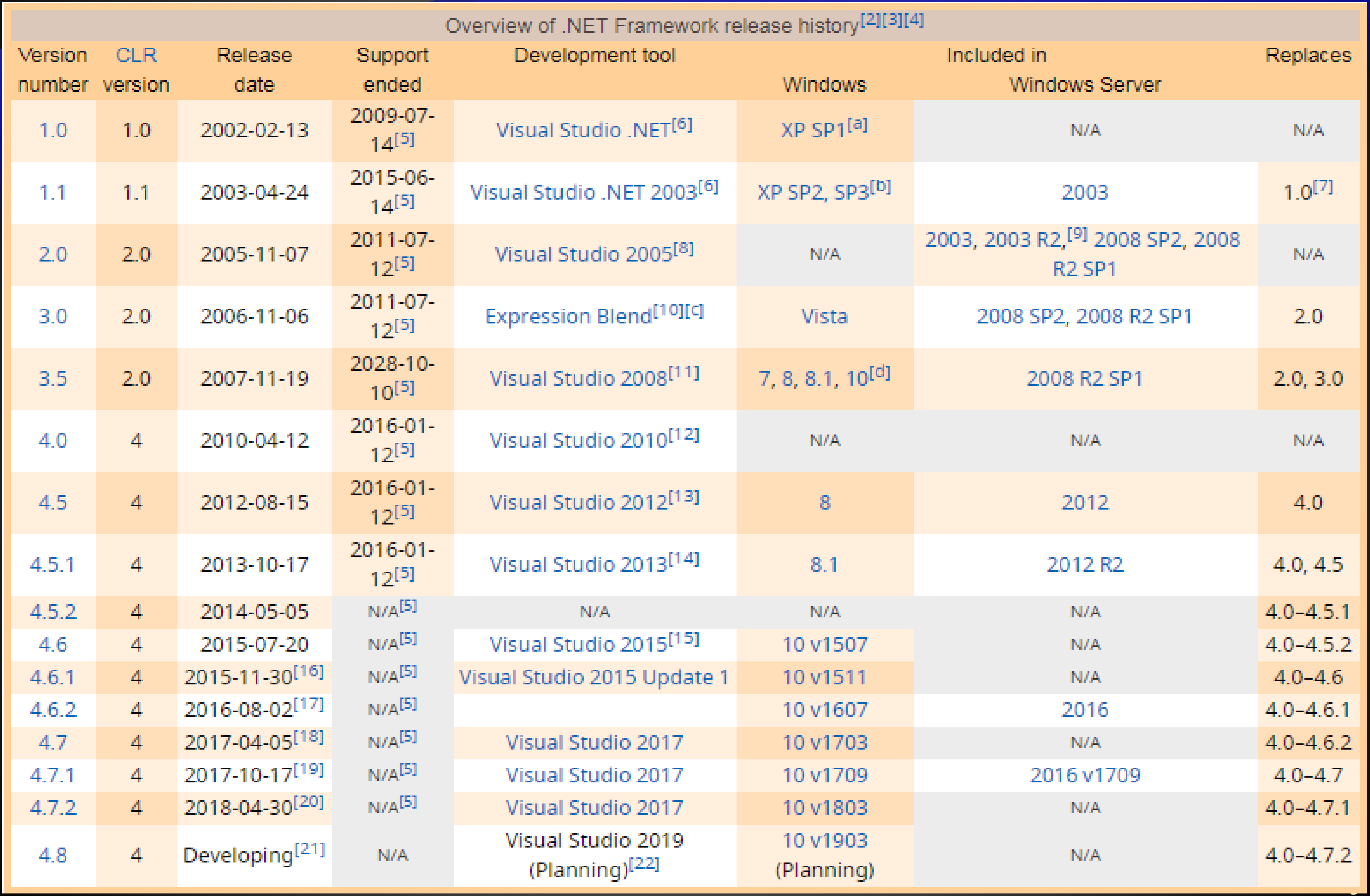

ASP.NET
Web Forms, MVC, AJAX
Mobile Internet Toolkit
Windows Forms
WPF
Silverlight
WCF and WWF (Communication and Workflow Tier)
ADO.NET, LINQ and XML (Data Tier)
Base Class Library (BCL)
Common Language Runtime (CLR)
Operating System (OS)

.NET Framework Architecture (7)

- Programming language on your flavor!

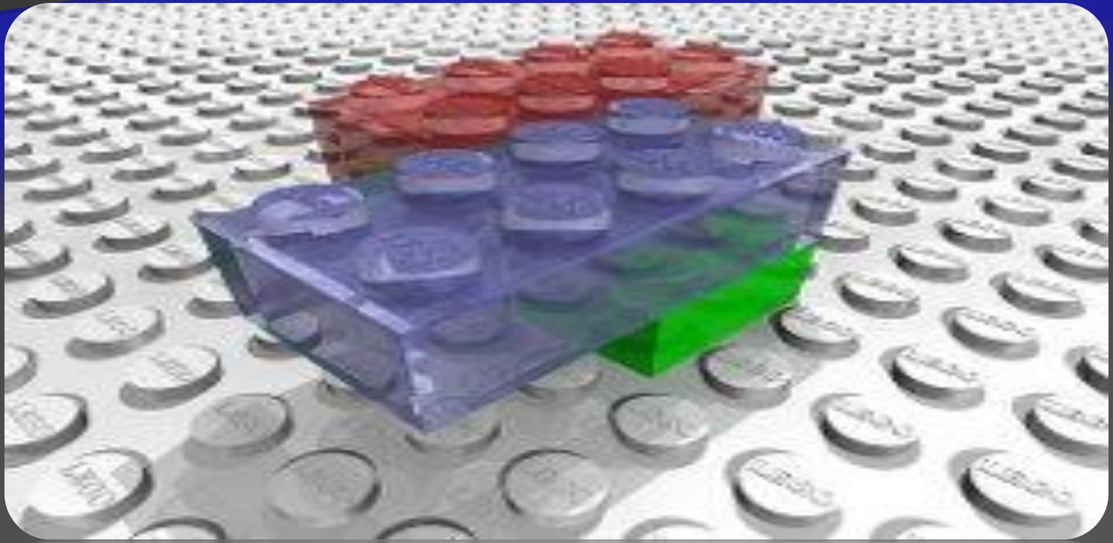


.NET Framework Versions



Overview of .NET Framework release history[2][3][4]

Version number	CLR version	Release date	Support ended	Development tool	Included in: Windows	Included in: Windows Server	Replaces
1.0	1.0	2002-02-13	2009-07-14[5]	Visual Studio .NET[6]	XP SP1[a]	N/A	N/A
1.1	1.1	2003-04-24	2015-06-14[5]	Visual Studio .NET 2003[6]	XP SP2, SP3[b]	2003	1.0[7]
2.0	2.0	2005-11-07	2011-07-12[5]	Visual Studio 2005[8]	N/A	2003, 2003 R2,[9] 2008 SP2, 2008 R2 SP1	N/A
3.0	2.0	2006-11-06	2011-07-12[5]	Expression Blend[10][c]	Vista	2008 SP2, 2008 R2 SP1	2.0
3.5	2.0	2007-11-19	2028-10-10[5]	Visual Studio 2008[11]	7, 8, 8.1, 10[d]	2008 R2 SP1	2.0, 3.0
4.0	4	2010-04-12	2016-01-12[5]	Visual Studio 2010[12]	N/A	N/A	N/A
4.5	4	2012-08-15	2016-01-12[5]	Visual Studio 2012[13]	8	2012	4.0
4.5.1	4	2013-10-17	2016-01-12[5]	Visual Studio 2013[14]	8.1	2012 R2	4.0, 4.5
4.5.2	4	2014-05-05	N/A[5]	N/A	N/A	N/A	4.0–4.5.1
4.6	4	2015-07-20	N/A[5]	Visual Studio 2015[15]	10 v1507	N/A	4.0–4.5.2
4.6.1	4	2015-11-30[16]	N/A[5]	Visual Studio 2015 Update 1	10 v1511	N/A	4.0–4.6
4.6.2	4	2016-08-02[17]	N/A[5]		10 v1607	2016	4.0–4.6.1
4.7	4	2017-04-05[18]	N/A[5]	Visual Studio 2017	10 v1703	N/A	4.0–4.6.2
4.7.1	4	2017-10-17[19]	N/A[5]	Visual Studio 2017	10 v1709	2016 v1709	4.0–4.7
4.7.2	4	2018-04-30[20]	N/A[5]	Visual Studio 2017	10 v1803	N/A	4.0–4.7.1
4.8	4	Developing[21]	N/A	Visual Studio 2019 (Planning)[22]	10 v1903 (Planning)	N/A	4.0–4.7.2



Common Language Runtime (CLR)

The Heart of .NET Framework

Common Language Runtime (CLR)

- Managed execution environment
  - Controls the execution of managed .NET programming code
- Something like virtual machine
  - Like the Java Virtual Machine (JVM)
- Not an interpreter
  - Compilation on-demand is used
    - Known as Just In Time (JIT) compilation

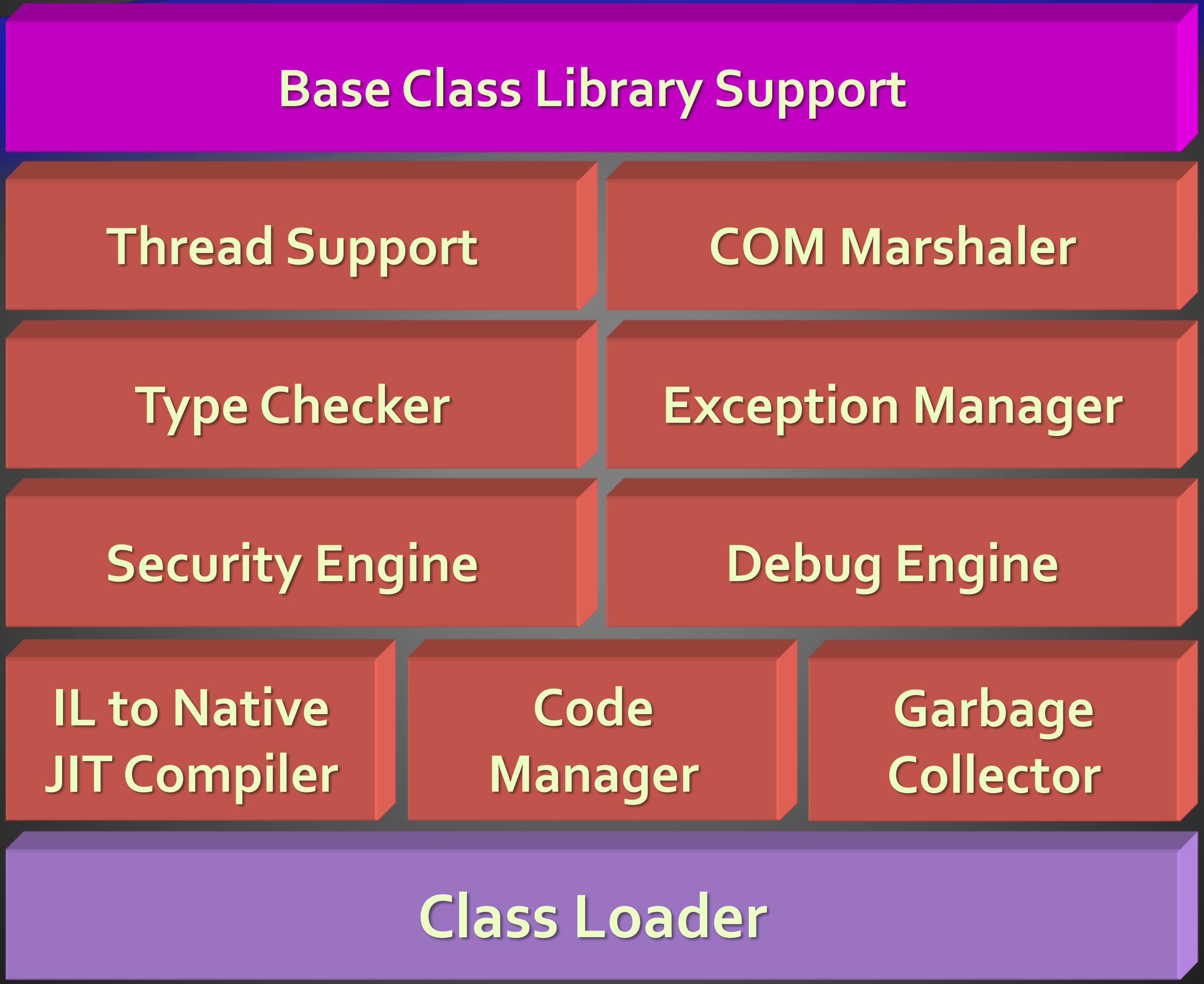
Responsibilities of CLR

- Execution of the IL code and the JIT compilation
- Managing memory and application resources
- Ensuring type safety
- Interaction with the OS
- Managing security
    - Code access security
    - Role-based security

Responsibilities of CLR (2)

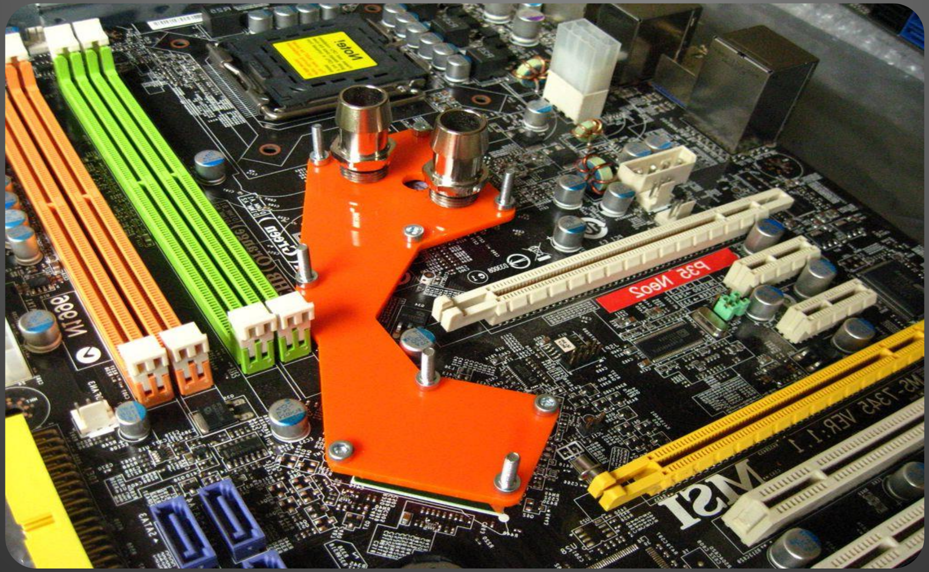
- Managing exceptions
- Managing concurrency – controlling the parallel execution of application threads
- Managing application domains and their isolation
- Interaction with unmanaged code
- Supporting debug / profile of .NET code

CLR Architecture



Managed and Unmanaged Code

What is the Difference?



Managed Code

- CLR executed code is called managed code
- Represents programming code in the low level language MSIL (MS Intermediate Language)
- Contains metadata
  - Description of classes, interfaces, properties, fields, methods, parameters, etc.
- Programs, written in any .NET language are
  - Compiled to managed code (MSIL)
  - Packaged as assemblies (`.exe` or `.dll` files)

Managed Code (2)

- Object-oriented
- Secure
- Reliable
  - Protected from irregular use of types (type-safe)
- Allows integration between components and data types of different programming languages
- Portable between different platforms
  - Windows, Linux, Max OS X, etc.

Memory Management

- CLR manages memory automatically
  - Dynamically loaded objects are stored in the managed heap
  - Unusable objects are automatically cleaned up by the garbage collector
- Some of the big problems are solved
  - Memory leaks
  - Access to freed or unallocated memory
- Objects are accessed through a reference

Intermediate Language (MSIL)

Intermediate Language (MSIL, IL, CIL)

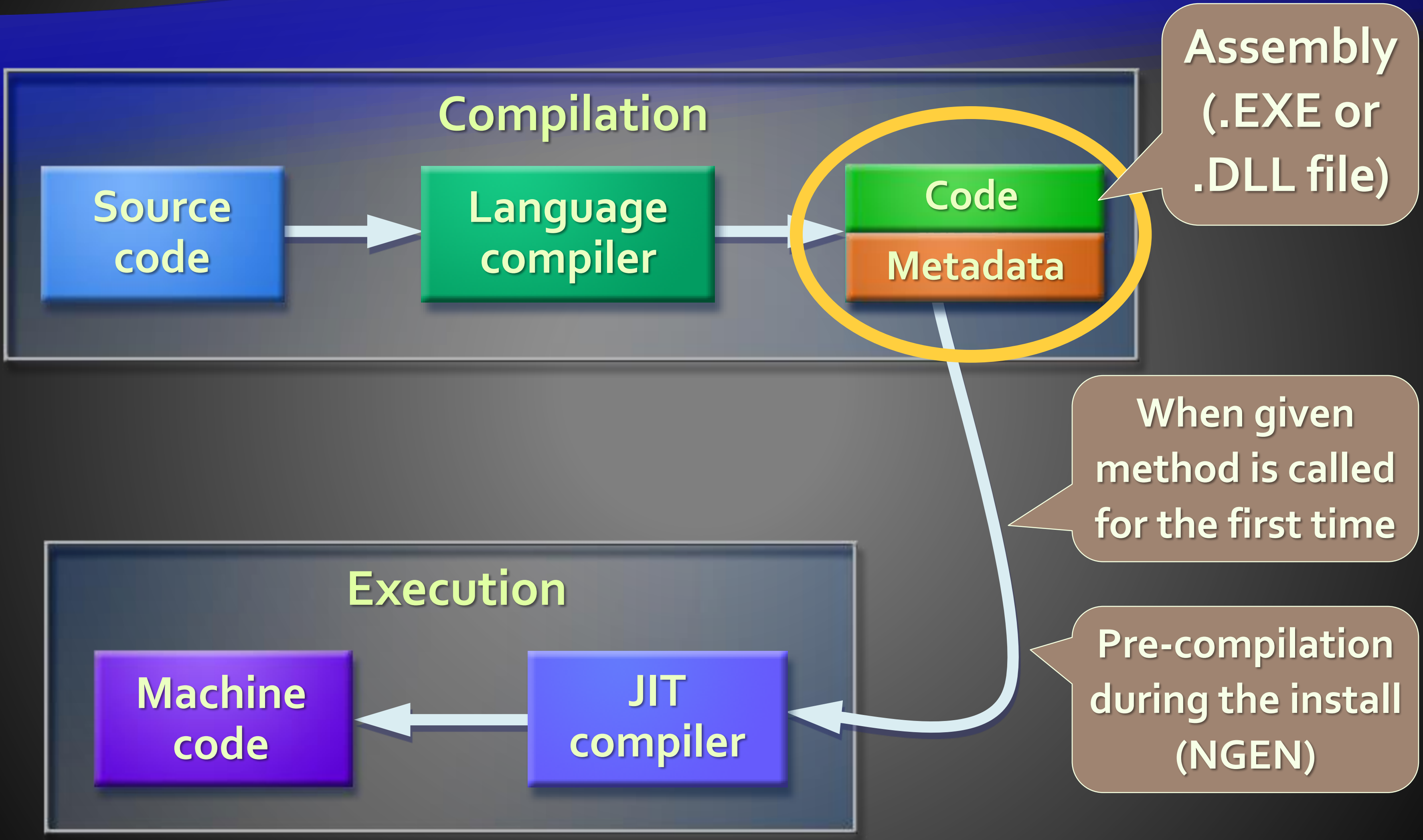
- Low level language (machine language) for the .NET CLR
- Has independent set of CPU instructions
  - Loading and storing data, calling methods
  - Arithmetic and logical operations
  - Exception handling
  - Etc.

- MSIL is converted to instructions for the current physical CPU by the JIT compiler

Sample MSIL Program

```
.method private hidebysig static void Main() cil managed
{
  .entrypoint
  // Code size       11 (0xb)
  .maxstack  8
  ldstr     "Hello, world!"
  call      void
    [mscorlib]System.Console::WriteLine(string)
  ret
} // end of method HelloWorld::Main
```

Compilation and Execution


Compilation
Source code
Language compiler
Code
Metadata
Assembly (.EXE or .DLL file)
When given method is called for the first time
Pre-compilation during the install (NGEN)
Execution
Machine code
JIT compiler

QUIZZ

- Quelles sont les principales composantes du Framework .NET?
- C'est quoi le CLR?
- Comment peut on expliquer le caractère multi-langage de la plateforme .NET?
- C'est le FCL?
- Expliquer le processus de compilation et exécution dans la Plateforme .NET?
- Quelle est la différence entre un code « Managed » et « Unmanaged » language?

.NET Applications

Assemblies, Metadata and Applications

.NET Assemblies

- .NET assemblies:
  - Self-containing .NET components
    - Stored in .DLL and .EXE files
  - Contain list of classes, types and resources
  - Smallest deployment unit in CLR
  - Have unique version number
- .NET deployment model
  - No version conflicts (forget the "DLL hell")
  - Supports side-by-side execution of different versions of the same assembly

Metadata in the Assemblies

- Metadata in the .NET assemblies
  - Data about data contained in the assembly
  - Integral part of the assembly
  - Generated by the .NET languages compiler
  - Describes all classes, their class members, versions, resources, etc.

Metadata in Assemblies

Type Description

Classes, interfaces, inner types, base classes, implemented interfaces, member fields, properties, methods, method parameters, return value, attributes, etc.

Assembly Description

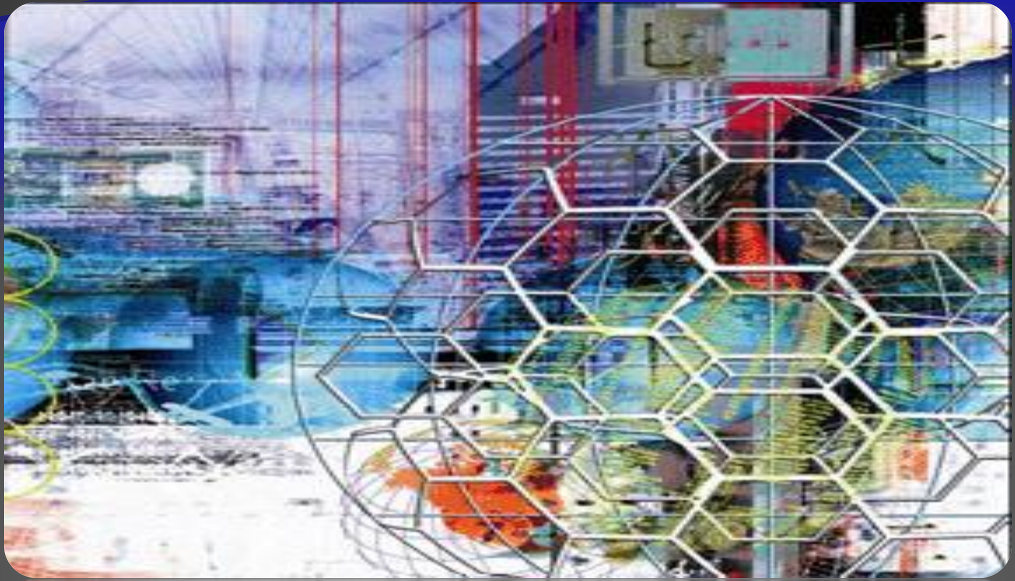
Name
Version
Localization

[digital signature]

Dependencies on other assemblies
Security permissions
Exported types

.NET Applications

- Configurable executable .NET units
- Consist of one or more assemblies
- Installed by "copy / paste"
  - No complex registration of components
- Different applications use different versions of common assemblies
  - No conflicts due to their "strong name"
- Easy installation, un-installation and update



Common Language Infrastructure

How .NET Supports Multiple Languages?

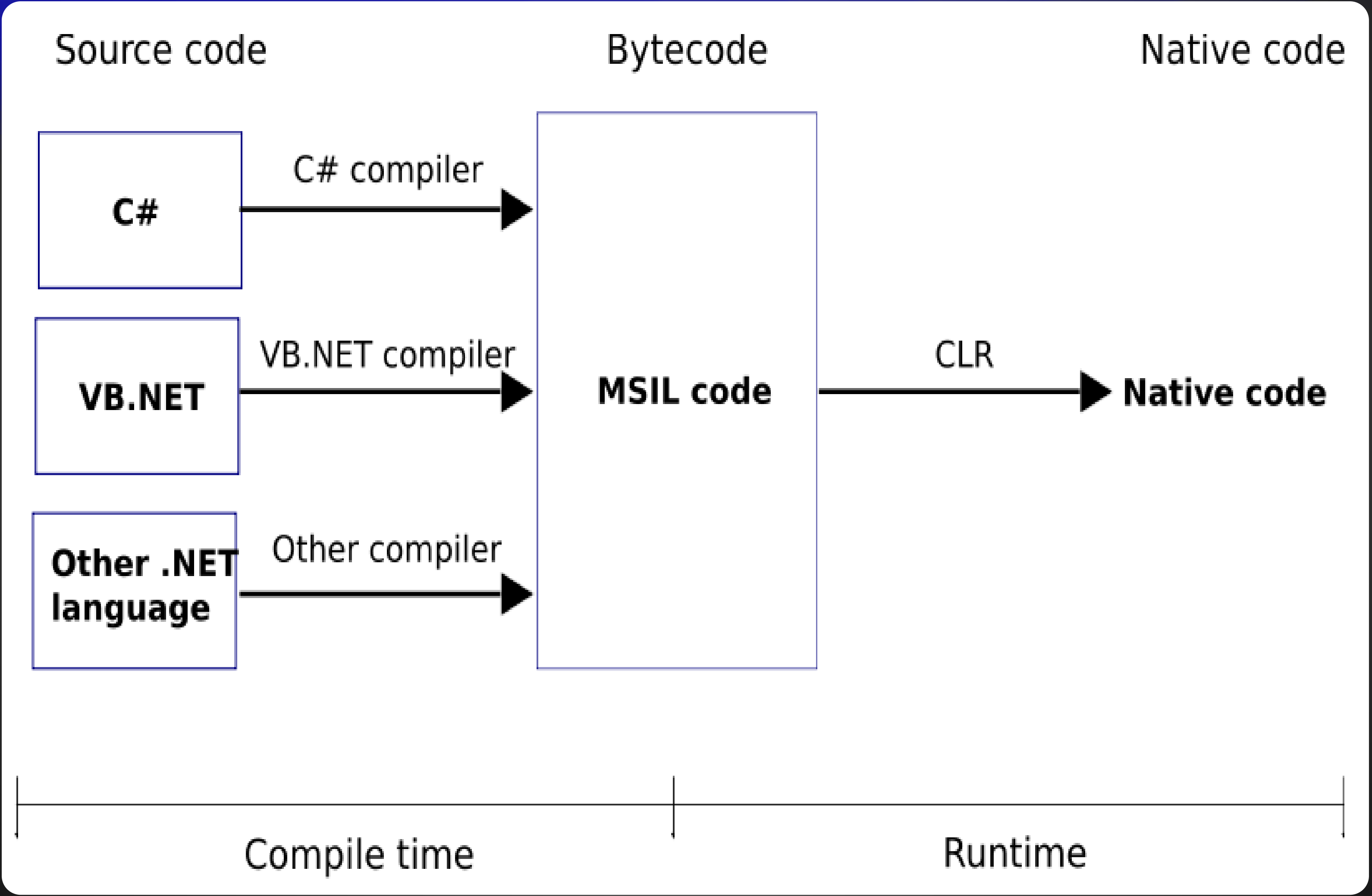
Common Language Infrastructure

- Common Language Infrastructure (CLI)
  - Open specification developed by Microsoft (ECMA – 335)
  - Multiple high-level languages run on different platforms without changes in the source code or pre-compilation
  - Standardized part of CLR
  - .NET Framework is CLI implementation for Windows
  - Mono is CLI implementation for Linux

Common Language Infrastructure (2)

- CLI describes the following aspects:
  - The Common Type System (CTS)
  - Assemblies and metadata
  - Common Language Specification (CLS)

.NET Code Compilation and Execution



Common Type System (CTS)

- CTS defines the CLR supported types of data and the operations over them
- Ensures data level compatibility between different .NET languages
  - E.g. `string` in C# is the same like `String` in VB.NET and in J#
- Value types and reference types
- All types derive from `System.Object`

Common Language Specification (CLS)

- CLS is a system of rules and obligations, that all .NET languages must obey
  - Ensures compatibility and ease of interaction between .NET languages
- Example: CLS enforces all .NET languages to be object-oriented
- When using non-CLS-compliant programming techniques you lose compatibility with the other .NET languages

The .NET Languages

C#, VB.NET, C++, J#, etc.

.NET Languages

- .NET languages by Microsoft
  - C#, VB.NET, Managed C++, J#, F#, JScript
- .NET languages by third parties
  - Object Pascal, Perl, Python, COBOL, Haskell, Oberon, Scheme, Smalltalk…
- Different languages can be mixed in a single application
- Cross-language inheritance of types and exception handling

C# Language

- C# is mixture between C++, Java
  - Fully object-oriented by design
- Component-oriented programming model
  - Components, properties and events
  - No header files like C/C++
  - Suitable for GUI and Web applications
  - XML based documentation
- In C# all data types are objects
  - Example: `5.ToString()` is a valid call

C# Language – Example

- C# is standardized by ECMA and ISO
- Example of C# program:

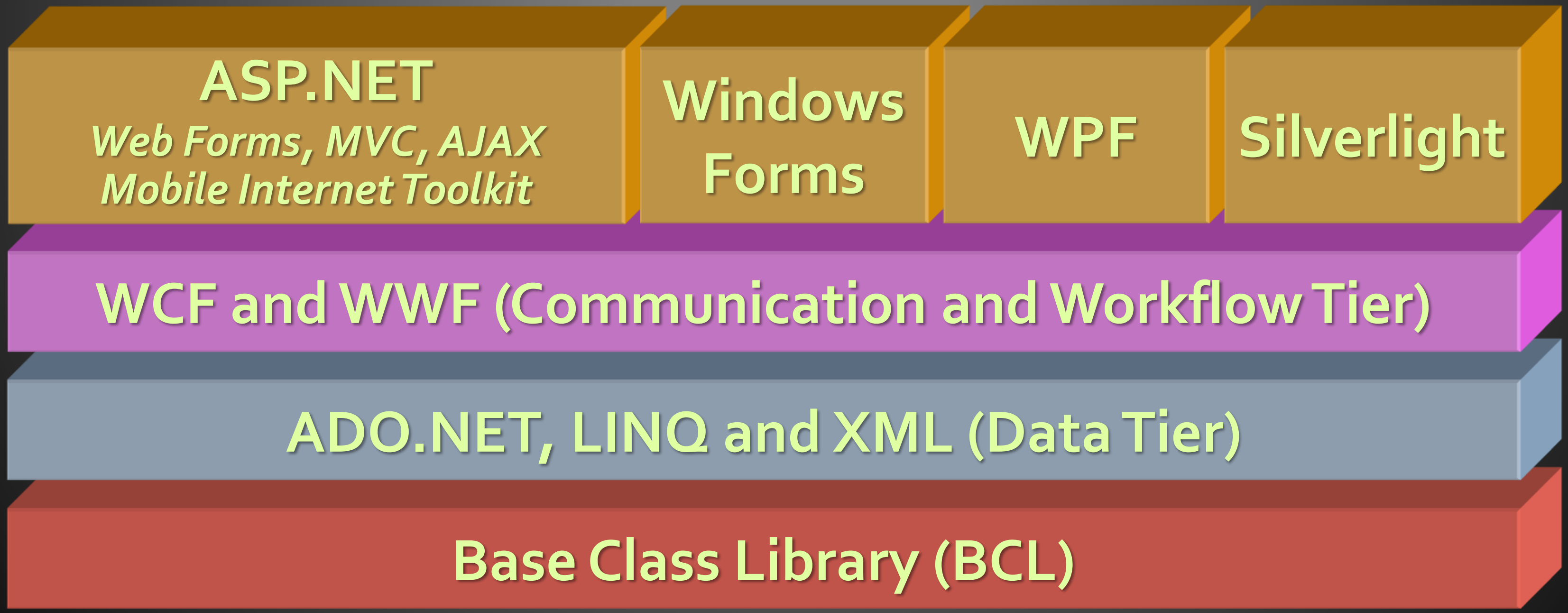
```
using System;

class NumbersFrom1to100
{
    static void Main()
    {
        for (int i=1; i<=100; i++)
        {
            Console.WriteLine(i);
        }
    }
}
```

Framework Class Library (FCL)

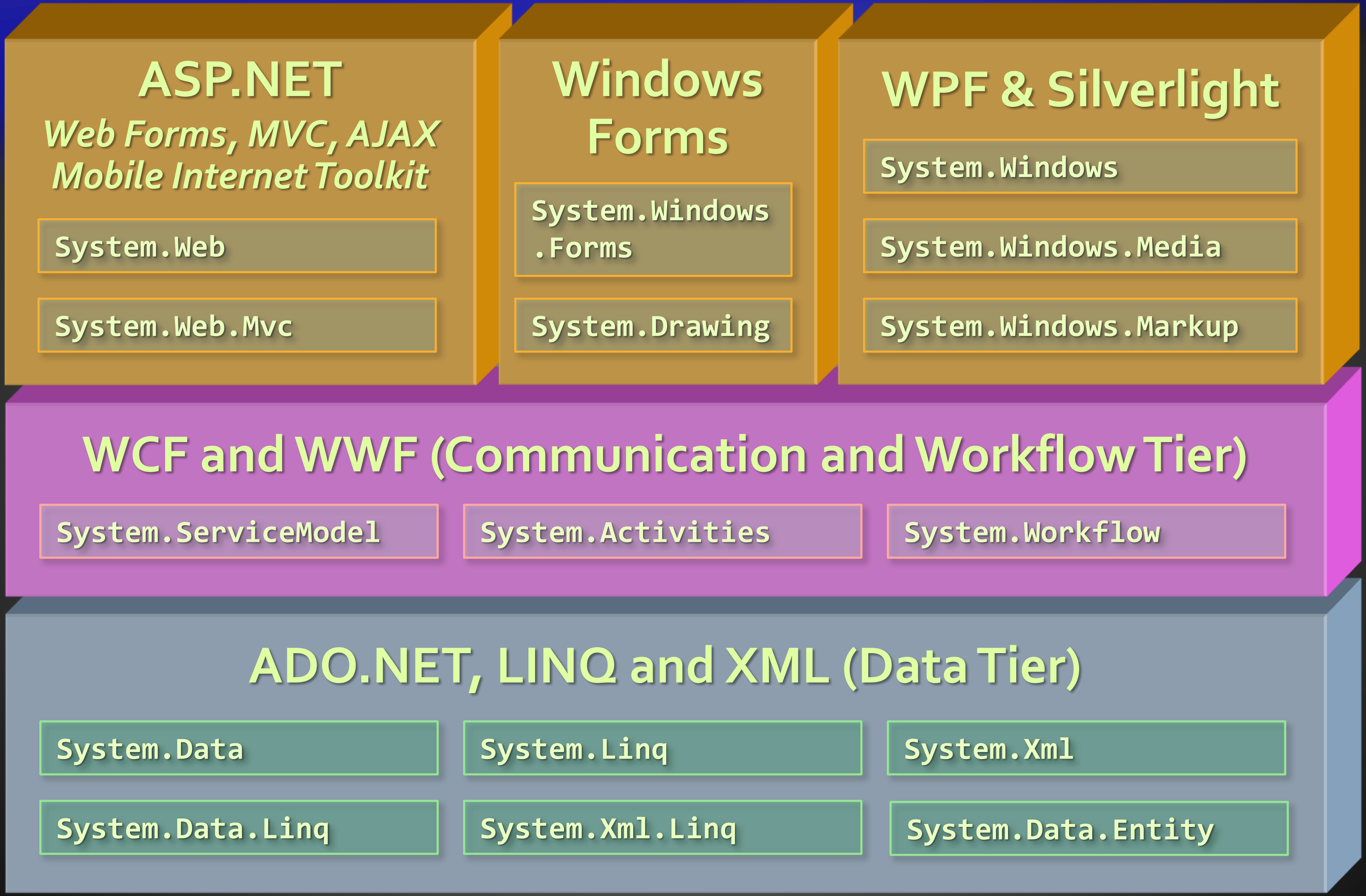
Standard Out-of-the-box .NET APIs

Framework Class Library (FCL)

- Framework Class Library is the standard .NET Framework library of out-of-the-box reusable classes and components (APIs)


ASP.NET
Web Forms, MVC, AJAX
Mobile Internet Toolkit
Windows Forms
WPF
Silverlight
WCF and WWF (Communication and Workflow Tier)
ADO.NET, LINQ and XML (Data Tier)
Base Class Library (BCL)

FCL Namespaces



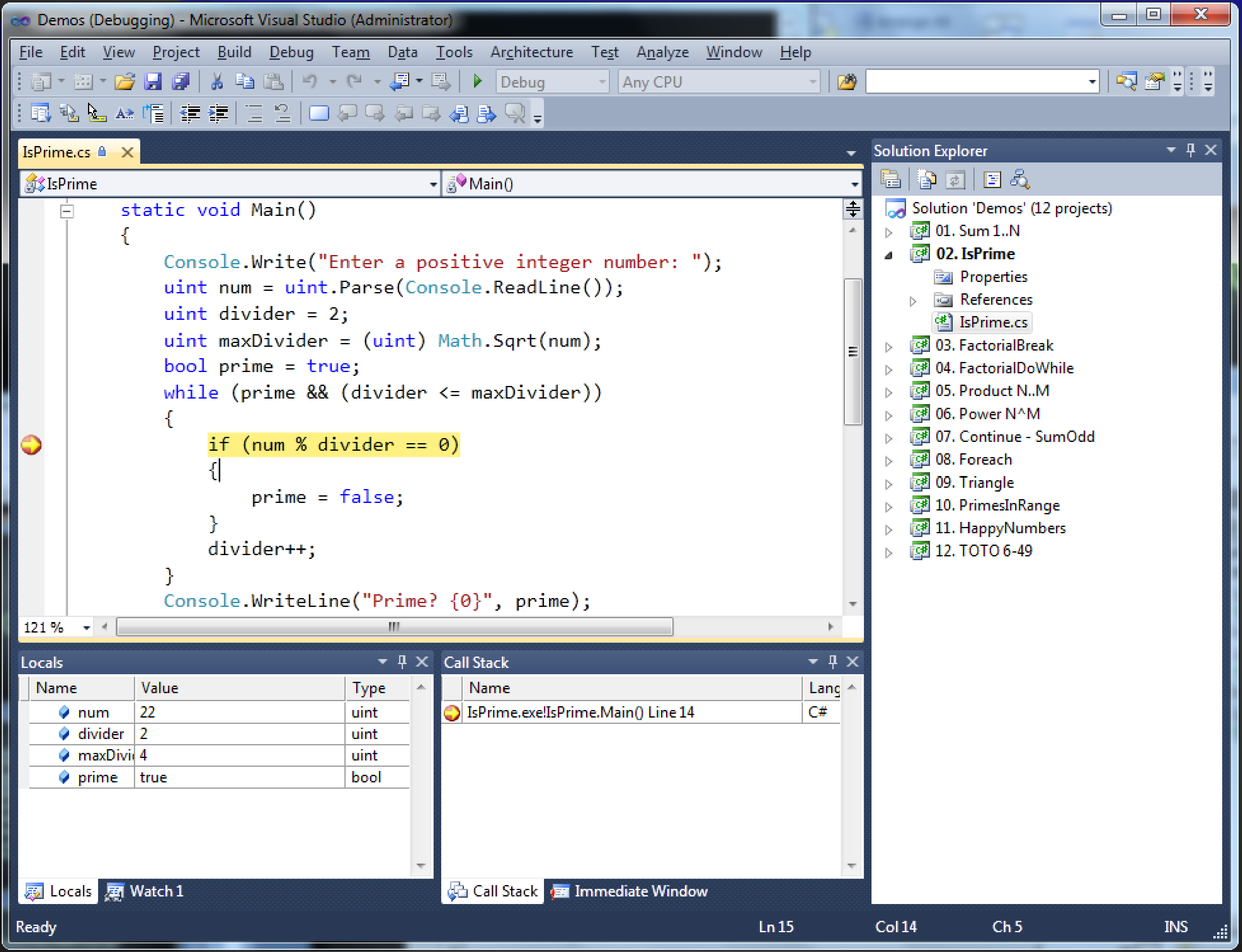
Visual Studio

- Visual Studio is powerful Integrated Development Environment (IDE) for .NET Developers
  - Create, edit, compile and run .NET applications
  - Different languages – C#, C++, VB.NET, J#, …
  - Flexible code editor
  - Powerful debugger
  - Integrated with SQL Server and IIS
  - Strong support of Web services, WCF and WWF

Visual Studio (2)

- Visual programming
  - Component-oriented, event based
- Managed and unmanaged code
- Helpful wizards and editors
  - Windows Forms Designer
  - WCF / Silverlight Designer
  - ASP.NET Web Forms Designer
  - ADO.NET / LINQ-to-SQL / XML Data Designer
- Many third party extensions

Visual Studio IDE



Demos (Debugging) - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Team Data Tools Architecture Test Analyze Window Help
Debug
Any CPU
IsPrime.cs
IsPrime
Main()
121 %
Solution Explorer
Locals
Watch 1
Call Stack
Immediate Window
Ready
Ln 15
Col 14
Ch 5
INS

```
static void Main()
{
    Console.Write("Enter a positive integer number: ");
    uint num = uint.Parse(Console.ReadLine());
    uint divider = 2;
    uint maxDivider = (uint) Math.Sqrt(num);
    bool prime = true;
    while (prime && (divider <= maxDivider))
    {
        if (num % divider == 0)
        {
            prime = false;
        }
        divider++;
    }
    Console.WriteLine("Prime? {0}", prime);
```

Solution 'Demos' (12 projects)
- 01. Sum 1..N
- 02. IsPrime
  - Properties
  - References
  - IsPrime.cs
- 03. FactorialBreak
- 04. FactorialDoWhile
- 05. Product N..M
- 06. Power N^M
- 07. Continue - SumOdd
- 08. Foreach
- 09. Triangle
- 10. PrimesInRange
- 11. HappyNumbers
- 12. TOTO 6-49

Locals

Name	Value	Type
num	22	uint
divider	2	uint
maxDivi	4	uint
prime	true	bool

Call Stack

Name	Lang
IsPrime.exe!IsPrime.Main() Line 14	C#

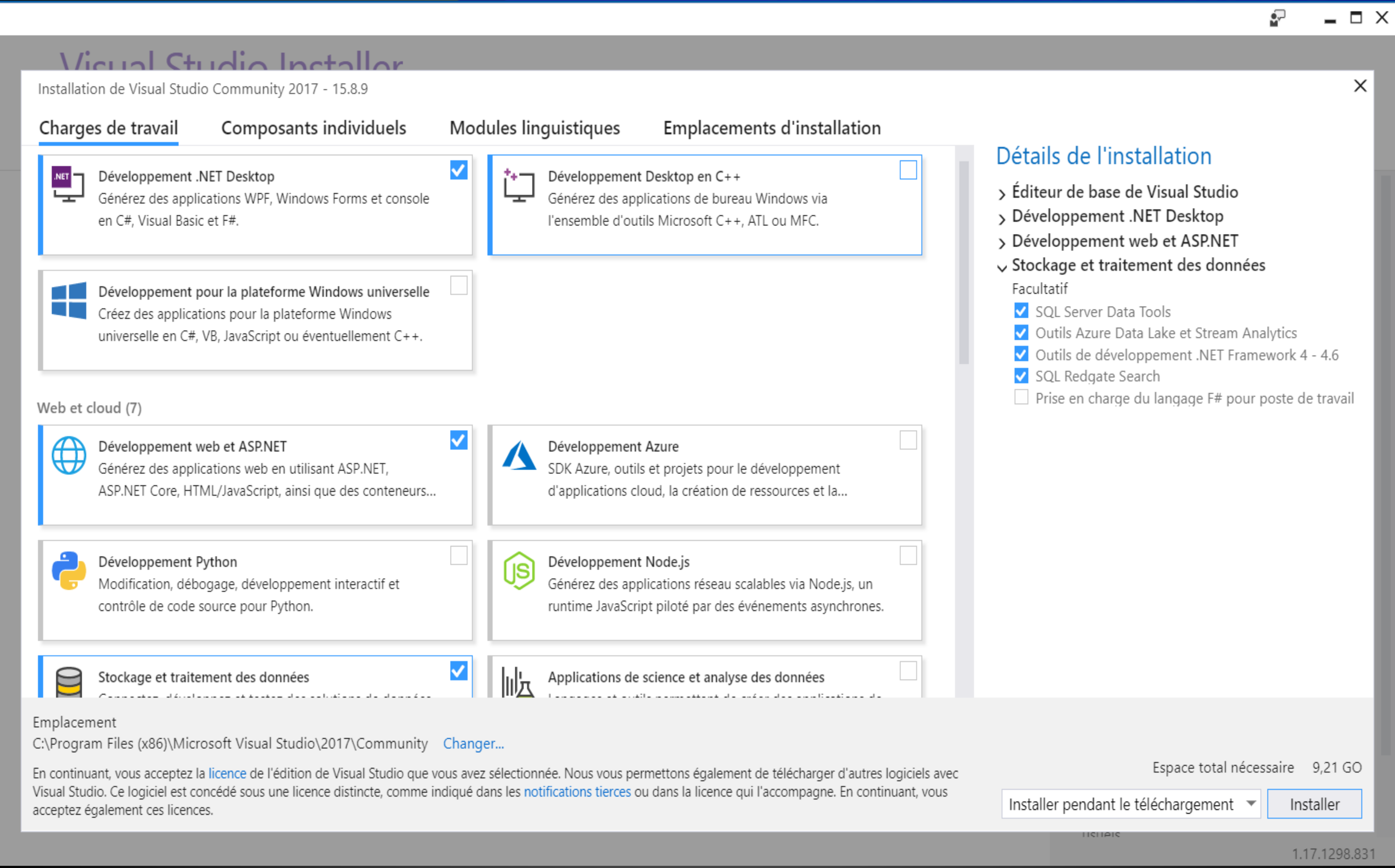
Installation de VS

- Microsoft Visual Studio Community 2017

- Téléchargeable via l'url

https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=Community

Installation de VS



Installation de Visual Studio Community 2017 - 15.8.9

Charges de travail | Composants individuels | Modules linguistiques | Emplacements d'installation

- [✓] **Développement .NET Desktop** — Générez des applications WPF, Windows Forms et console en C#, Visual Basic et F#.
- [ ] **Développement Desktop en C++** — Générez des applications de bureau Windows via l'ensemble d'outils Microsoft C++, ATL ou MFC.
- [ ] **Développement pour la plateforme Windows universelle** — Créez des applications pour la plateforme Windows universelle en C#, VB, JavaScript ou éventuellement C++.

Web et cloud (7)

- [✓] **Développement web et ASP.NET** — Générez des applications web en utilisant ASP.NET, ASP.NET Core, HTML/JavaScript, ainsi que des conteneurs...
- [ ] **Développement Azure** — SDK Azure, outils et projets pour le développement d'applications cloud, la création de ressources et la...
- [ ] **Développement Python** — Modification, débogage, développement interactif et contrôle de code source pour Python.
- [ ] **Développement Node.js** — Générez des applications réseau scalables via Node.js, un runtime JavaScript piloté par des événements asynchrones.
- [✓] **Stockage et traitement des données**
- [ ] **Applications de science et analyse des données**

Détails de l'installation

- Éditeur de base de Visual Studio
- Développement .NET Desktop
- Développement web et ASP.NET
- Stockage et traitement des données
  - Facultatif
  - [✓] SQL Server Data Tools
  - [✓] Outils Azure Data Lake et Stream Analytics
  - [✓] Outils de développement .NET Framework 4 - 4.6
  - [✓] SQL Redgate Search
  - [ ] Prise en charge du langage F# pour poste de travail

Emplacement
C:\Program Files (x86)\Microsoft Visual Studio\2017\Community Changer...

En continuant, vous acceptez la licence de l'édition de Visual Studio que vous avez sélectionnée. Nous vous permettons également de télécharger d'autres logiciels avec Visual Studio. Ce logiciel est concédé sous une licence distincte, comme indiqué dans les notifications tierces ou dans la licence qui l'accompagne. En continuant, vous acceptez également ces licences.

Espace total nécessaire 9,21 GO

Installer pendant le téléchargement | Installer

1.17.1298.831

.NET Framework Overview

Questions?