Programming in C#
Class 1: Overview

Kevin.Li@GeorgianCollege.ca

https://www.linkedin.com/in/kevinca

Welcome to connect

Success Stories

https://www.linkedin.com/in/luluwheelan/

https://www.linkedin.com/in/hongjo-lim-a00562158/

https://www.linkedin.com/in/lynette-xie/

Outline

- 1. Syllabus, Lesson Outline, Text book
- 2. Student Survey
- 3. A real C# project introduction
- 4. .NET Framework overview
- 5. C# Overview
- 6. IDE
- 7. C# Program structure and Basic Syntax
- 8. Practice

Text book

- Murach's C# 2015

by Anne Boehm and Joel Murach
26 chapters, 884 pages, 378 illustrations
Published January 2016
ISBN 978-1-890774-94-3

https://www.murach.com/shop/murach-s-c-2015-detail

First thing

- Start to download/install Visual Studio

1. Lesson Outline

- 01.00.00 .NET Framework and Development Environment
- 02.00.00 Language/Grammar
- 03.00.00 Data Types and Declarations
- 04.00.00 Error Handling
- 05.00.00 Methods and Events
- 06.00.00 Class Modules and Objects
- 07.00.00 Flat Data Files
- 08.00.00 Database Connections

1. Lesson Outline

- 09.00.00 .NET Language-Integrated Query (LINQ)
- 10.00.00 Data Interchange Formats
- 11.00.00 Data Manipulation
- 12.00.00 Visual Form Design
- 13.00.00 Controls and Properties
- Final Project
- Mid-Term Test

Microsoft .net

My current C# project

- System Integration

.NET Enterprise Vision

Users
Any device,
Any place,
Any time

Internet

XML Web Services
Integrate business applications and processes

Authentication

Scheduling

Notification

Back Office
Heterogeneous application and server infrastructure

ERP & Billing

Customer Service

Sales

.NET Enterprise Vision

Users
Any device,
Any place,
Any time

Internet

XML Web Services /Web API

Authentication

Scheduling

Notification

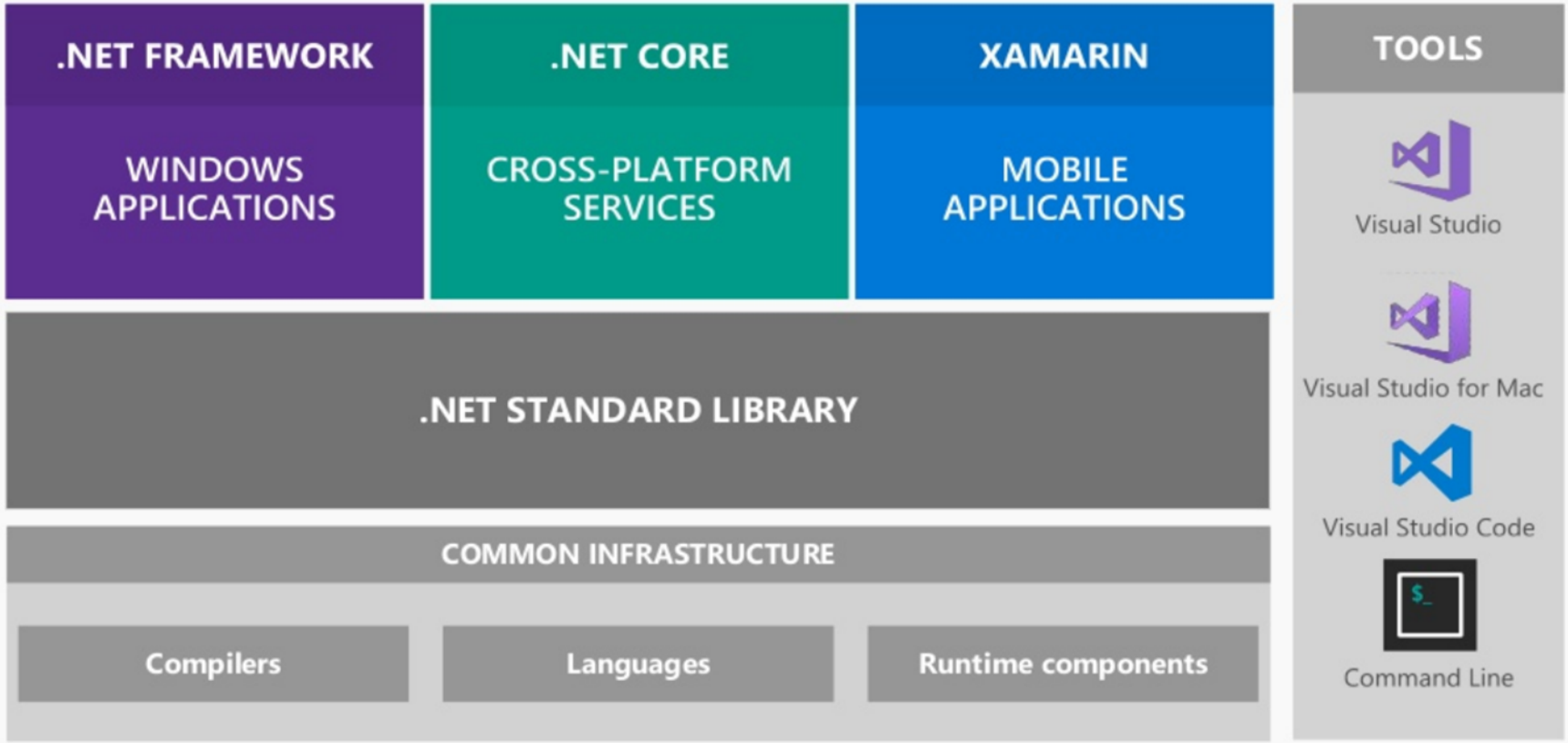
Back Office
Heterogeneous application and server infrastructure

ERP & Billing

Customer Service

Sales

.NET – A unified platform



So what is .NET?

- .NET is a platform that provides a standardized set of services.
  - It's just like Windows, except distributed over the Internet.
  - It exports a common interface so that it's programs can be run on any system that supports .NET.
- A specific software framework
  - Includes a common runtime

.NET Framework

- Programming model for .NET
- Platform for running .NET managed code in a virtual machine
- Provides a very good environment to develop networked applications and Web Services,

  Cross platform mobile apps.
- Provides programming API and unified language-independent development framework

The Core of .NET Framework: FCL & CLR

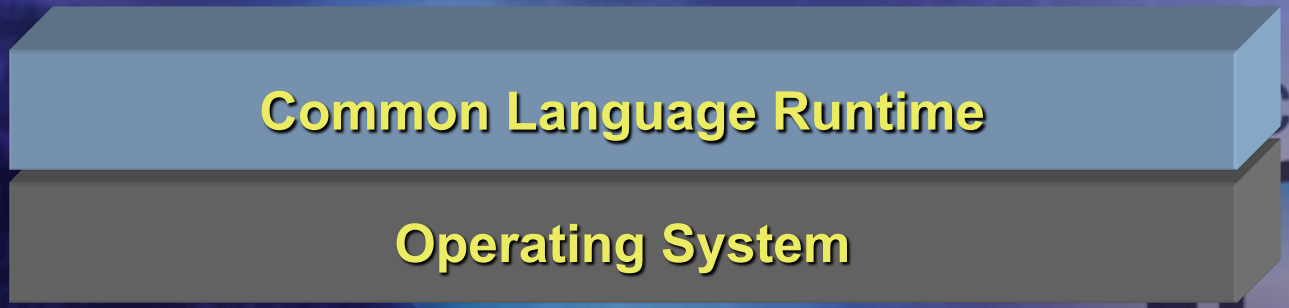
- Common Language Runtime
  - Garbage collection
  - Language integration (eg: C#, VB.NET)
  - Multiple versioning support (no more DLL hell!) (eg:3rd party dlls)
  - *Integrated security*
- Framework Class Library
  - Provides the core functionality:

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

.NET Framework

Common Language Runtime

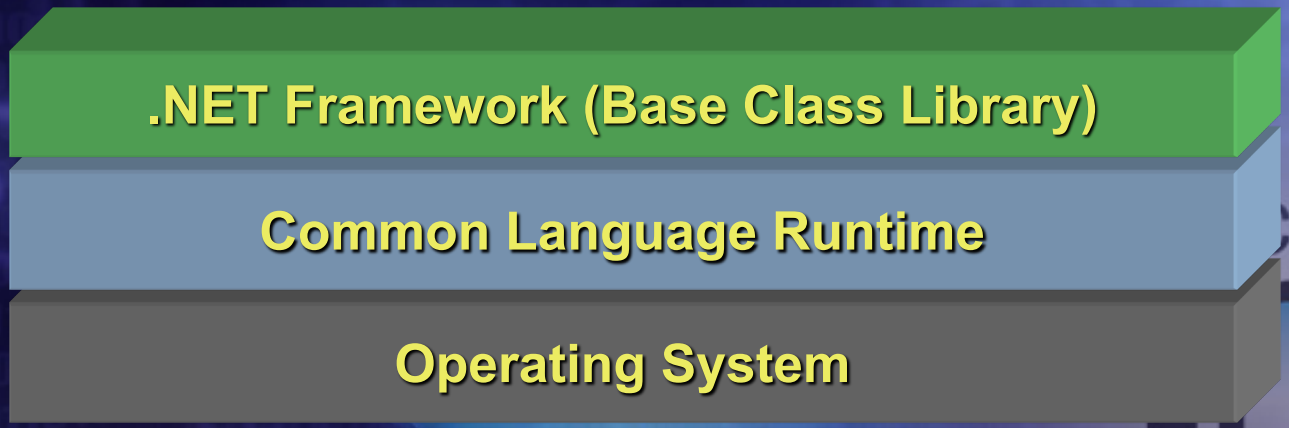
- CLR manages code execution at runtime
- Memory management, thread management, etc.



.NET Framework

Base Class Library

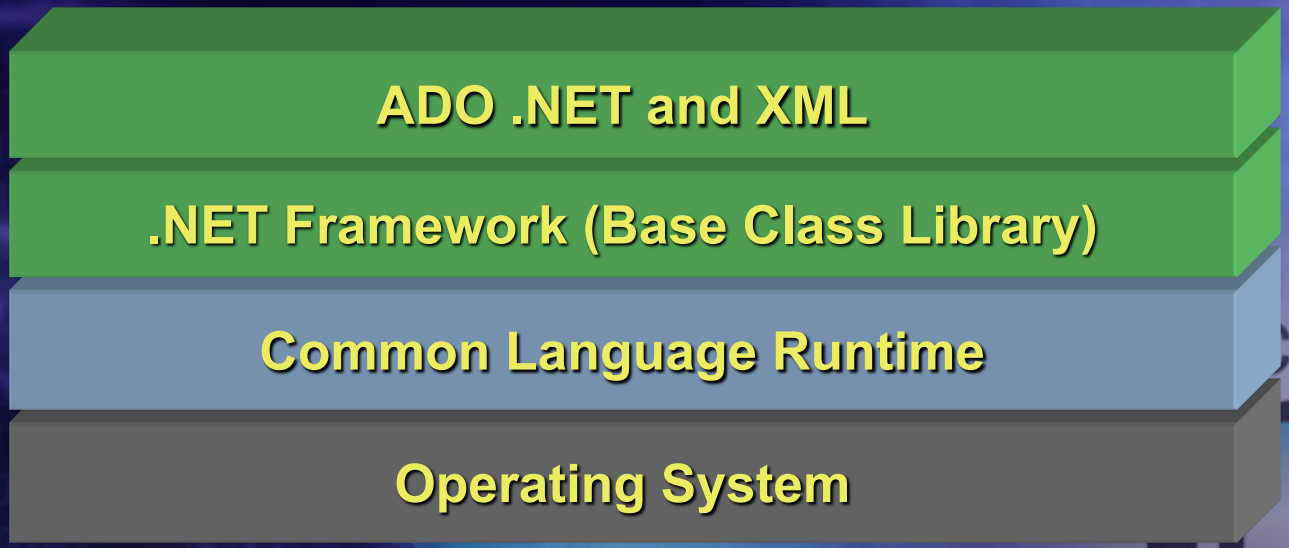
- Object-oriented collection of reusable types
- Collections, I/O, Strings, …



.NET Framework

Data Access Layer

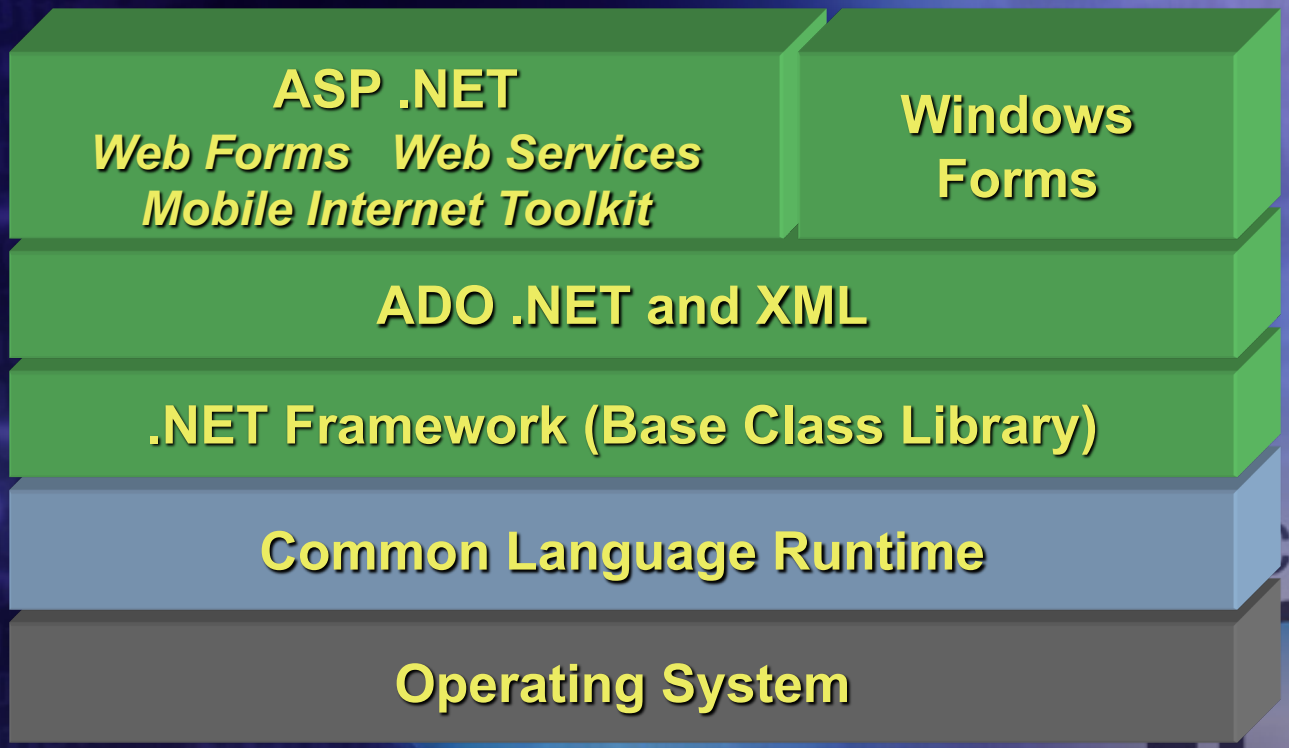
- Access relational databases
- Disconnected data model
- Work with XML



.NET Framework

ASP.NET & Windows Forms

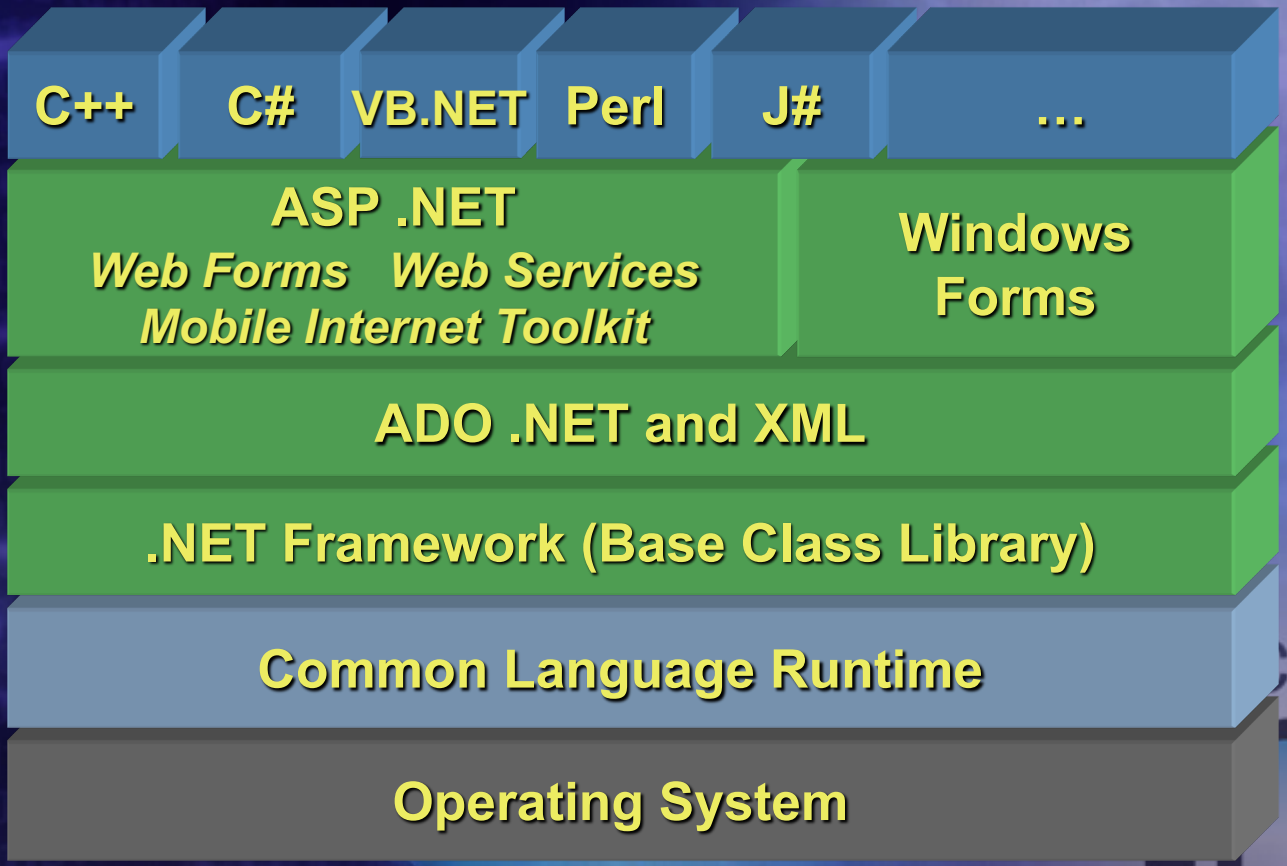
- Create application's front-end – Web-based user interface, Windows GUI, Web services, …



.NET Framework

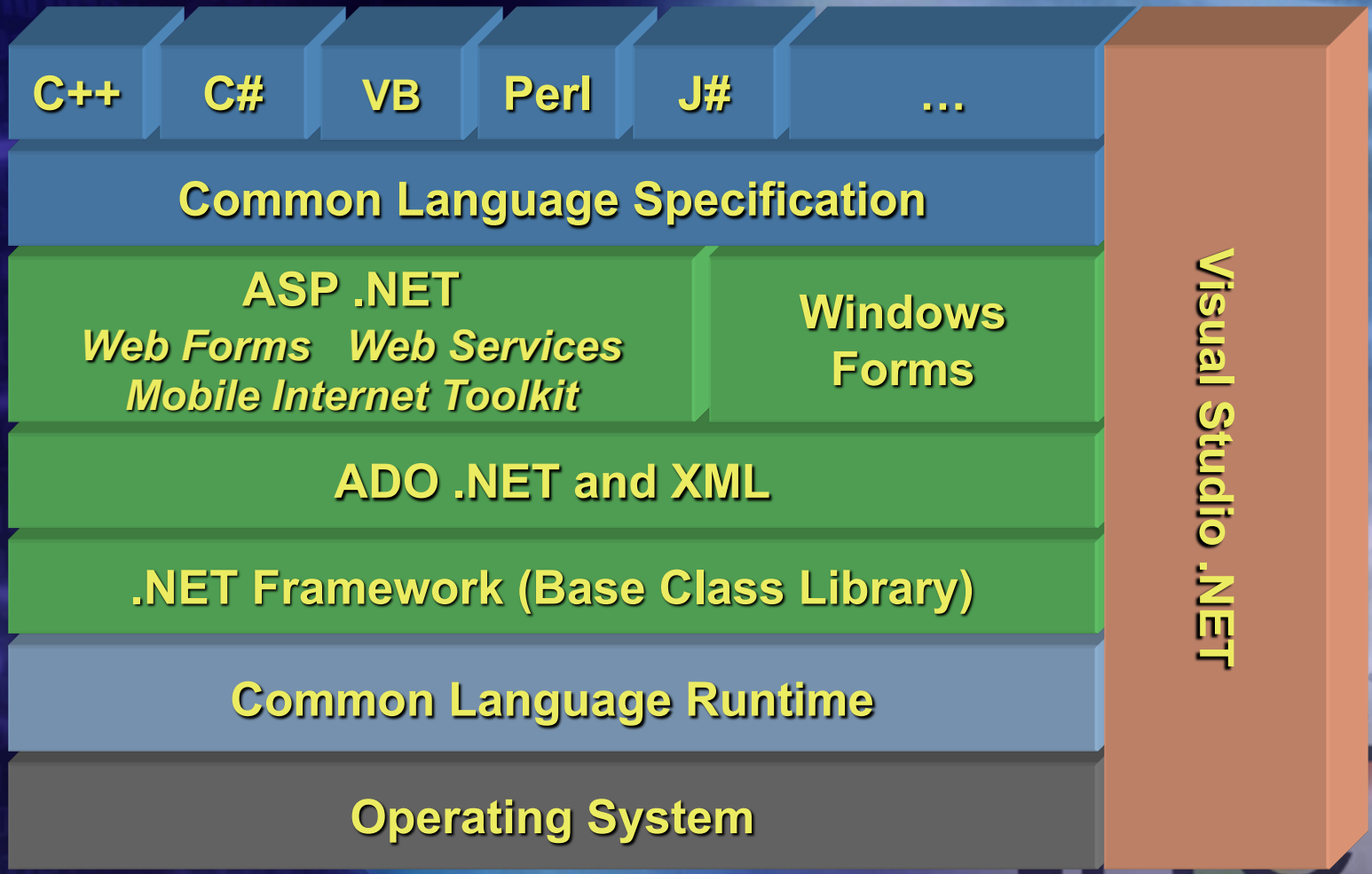
Programming Languages

- Use your favorite language



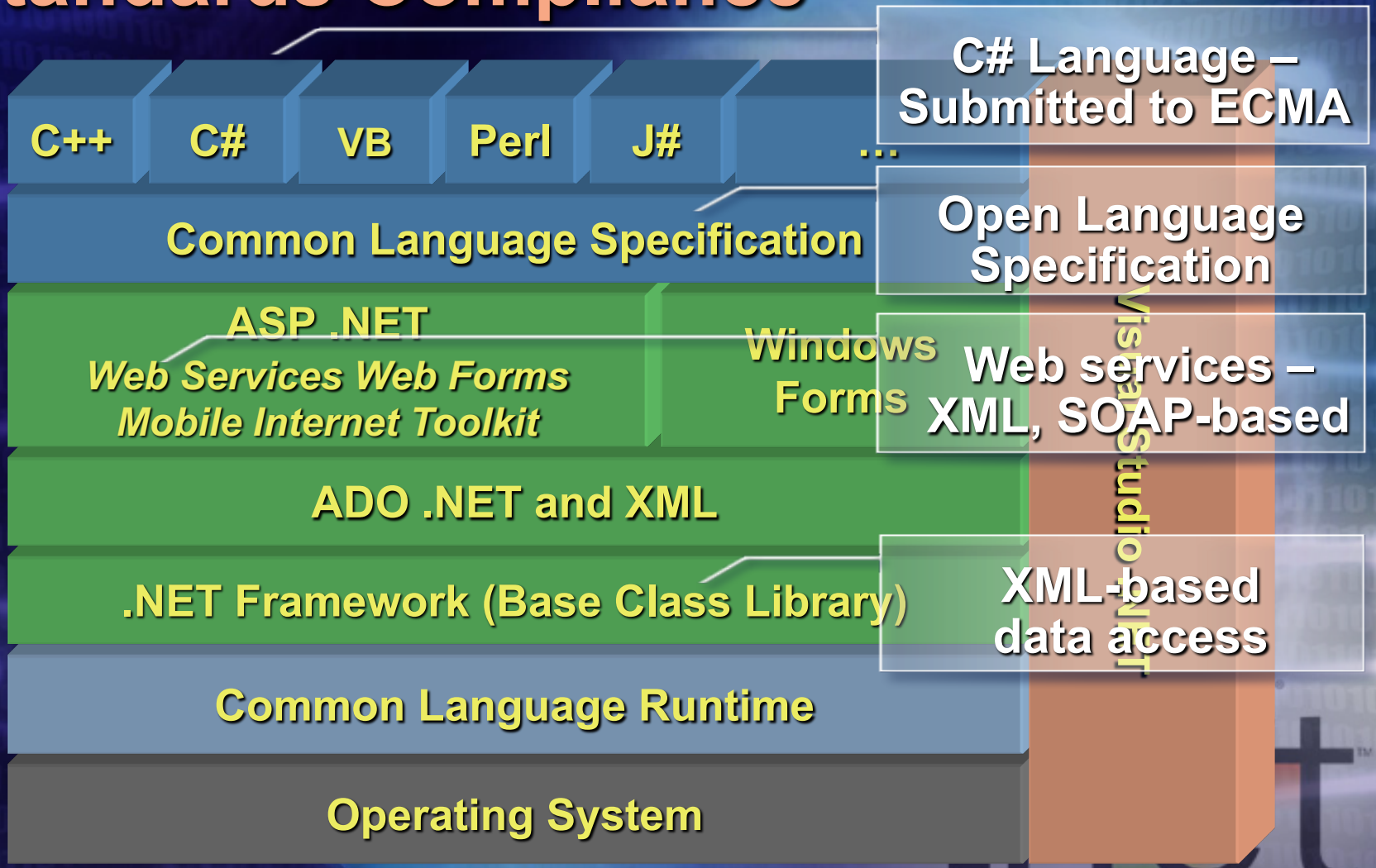
.NET Framework

Visual Studio .NET



.NET Framework

Standards Compliance



Common Language Runtime

- Manages running code – like a virtual machine
  - Threading
  - Memory management
  - *No interpreter*: JIT-compiler produces native code – during the program installation or at run time (https://en.wikipedia.org/wiki/Dynamic_Language_Runtime)
- *Fine-grained evidence-based security*
  - *Code access security*
    - *Code can be verified to guarantee type safety*
    - *No unsafe casts, no un-initialized variables and no out-of-bounds array indexing*

Managed Code

- Code that targets the CLR is referred to as managed code
- All managed code has the features of the CLR
  - Object-oriented
  - Type-safe
  - Cross-language integration
  - Cross language exception handling
  - Multiple version support
- Managed code is represented in special Intermediate Language (IL)

Automatic Memory Management

- The CLR manages memory for managed code
  - All allocations of objects and buffers made from a *Managed Heap*
  - Unused objects and buffers are cleaned up automatically through *Garbage Collection*
- Some of the worst bugs in software development are not possible with managed code
  - Leaked memory or objects
  - *References to freed or non-existent objects*
  - Reading of uninitialized variables
- Pointerless environment

Multiple Language Support

- CIL (MSIL or IL) – Intermediate Language
  - It is low-level (machine) language, like Assembler, but is Object-oriented
- CTS is a rich type system built into the CLR
  - Implements various types (int, float, string, …)
  - And operations on those types
- CLS is a set of specifications that all languages and libraries need to follow
  - This will ensure interoperability between languages

Intermediate Language

- .NET languages are compiled to an Intermediate Language (IL)
- Common Intermediate Language is also known as MSIL
- CLR compiles IL in just-in-time (JIT) manner – each function is compiled just before execution
- The JIT code stays in memory for subsequent calls
- Recompilations of assemblies are also possible

Example of MSIL Code

```
.method private hidebysig static void  Main()
  cil managed
{
  .entrypoint
  // Code size       11 (0xb)
  .maxstack  8
  IL_0000:  ldstr      "Hello, world!"
  IL_0005:  call       void
  [mscorlib]System.Console::WriteLine(string)
  IL_000a:  ret
} // end of method HelloWorld::Main
```

Common Type System (CTS)

- All .NET languages have the same primitive data types. An *int* in C# is the same as an *int* in VB.NET
- When communicating between modules written in any .NET language, the types are guaranteed to be compatible on the binary level
- Types can be:
  - Value types – passed by value, stored in the stack
  - Reference types – passed by reference, stored in the heap
- Strings are a primitive data type now

C# Language

- Object-oriented
  - Properties, Methods, Events
  - Attributes
  - All in one place, no header files etc.
  - Can be embedded in ASP pages
- Everything really is an object
  - Primitive types aren't magic
  - Unified type system == Deep simplicity
  - Improved extensibility and reusability

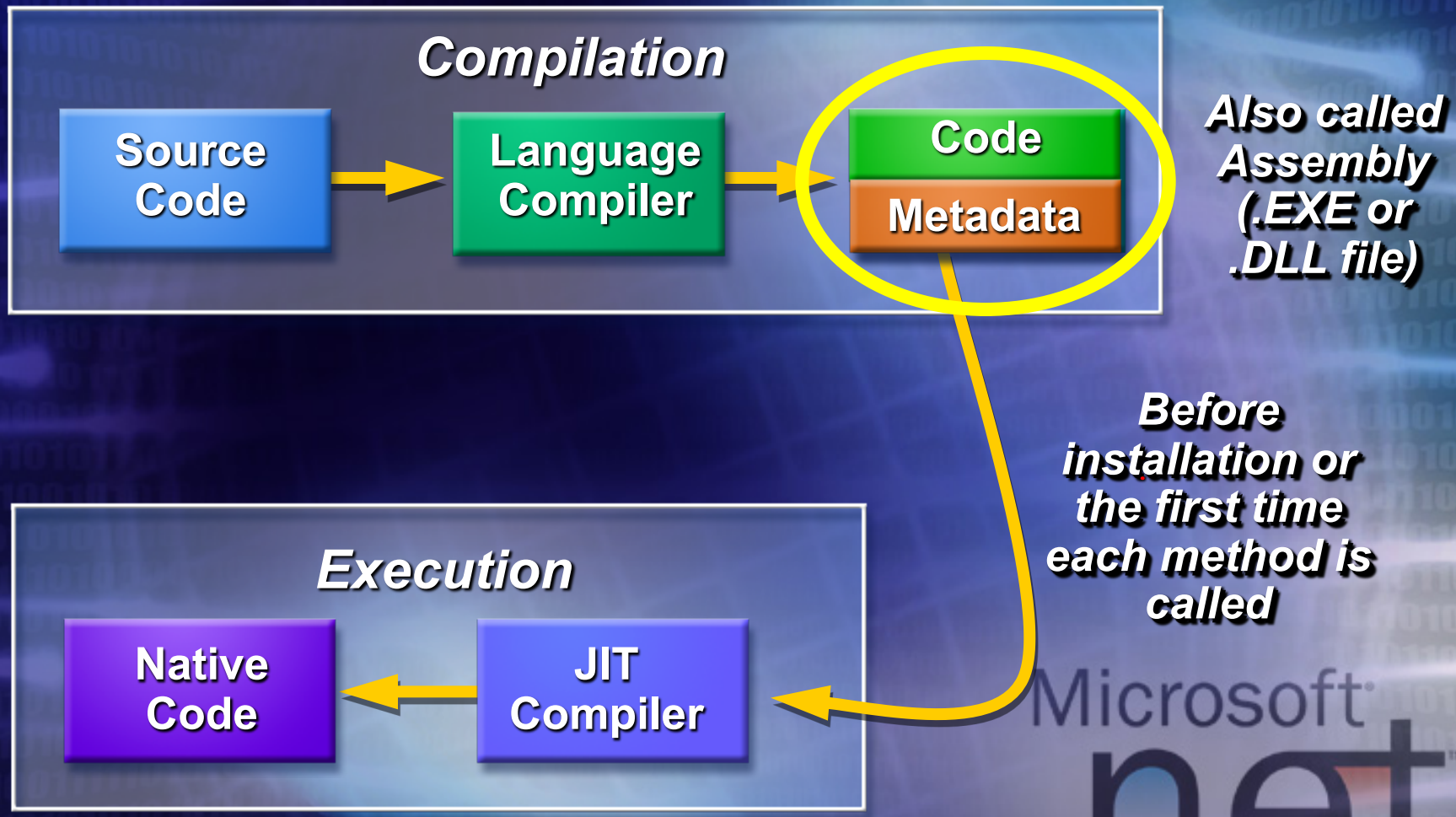
C# Language – Example

```
using System;

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

Microsoft .net

Code Compilation and Execution

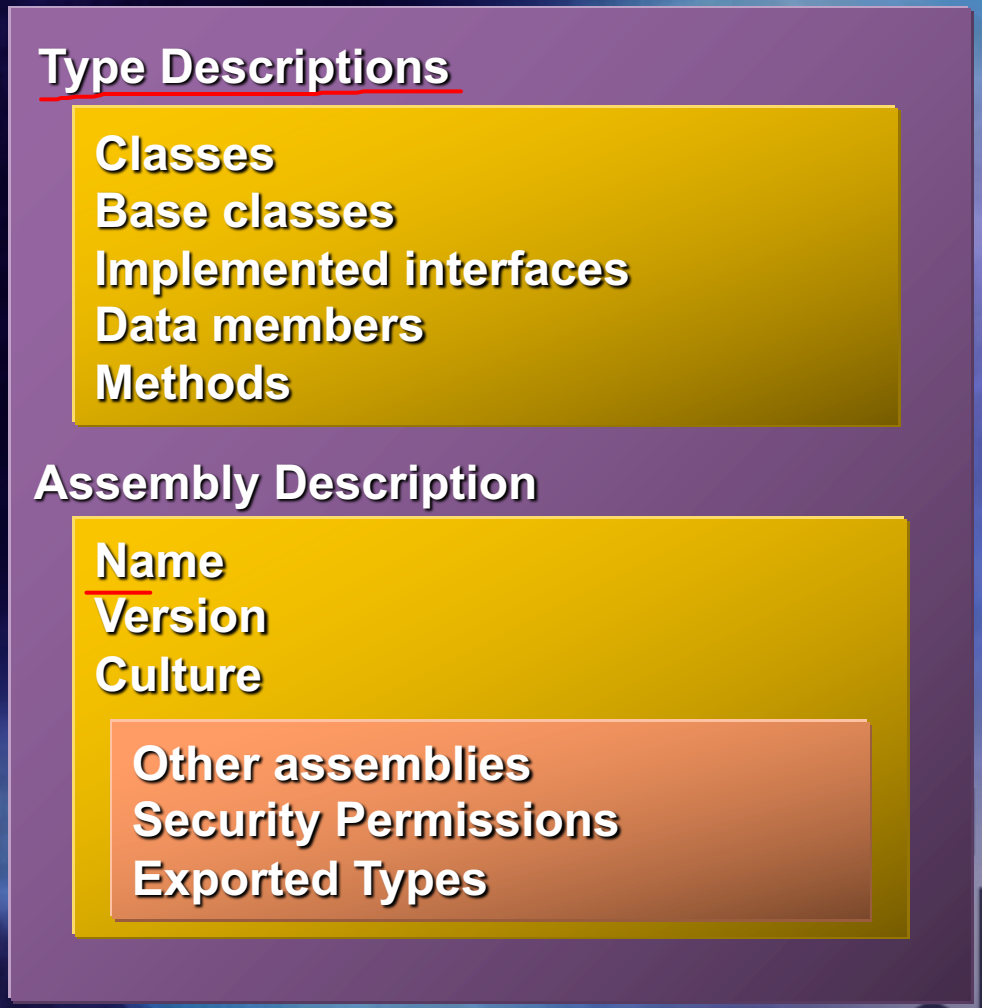


Microsoft .net

Assemblies

- DLL or EXE file
- Smallest deployable unit in the CLR
- Have unique version number
- No version conflicts (known as DLL hell)
- Contains IL code to be executed
- Security boundary – permissions are granted at the assembly level
- Type boundary – all types include the assembly name they are a part of
- **Self-describing manifest – metadata that describes the types in the assembly** (https://docs.microsoft.com/en-us/dotnet/standard/assembly/manifest:https:

Metadata in Assembly



Applications

- One or more assemblies
- Assemblies conflict resolution
  - Using metadata
    - Local (preferred)
    - Global Assembly Cache (GAC)
- Different applications may use different versions of an assembly
  - Easier software updates
  - Easier software removal

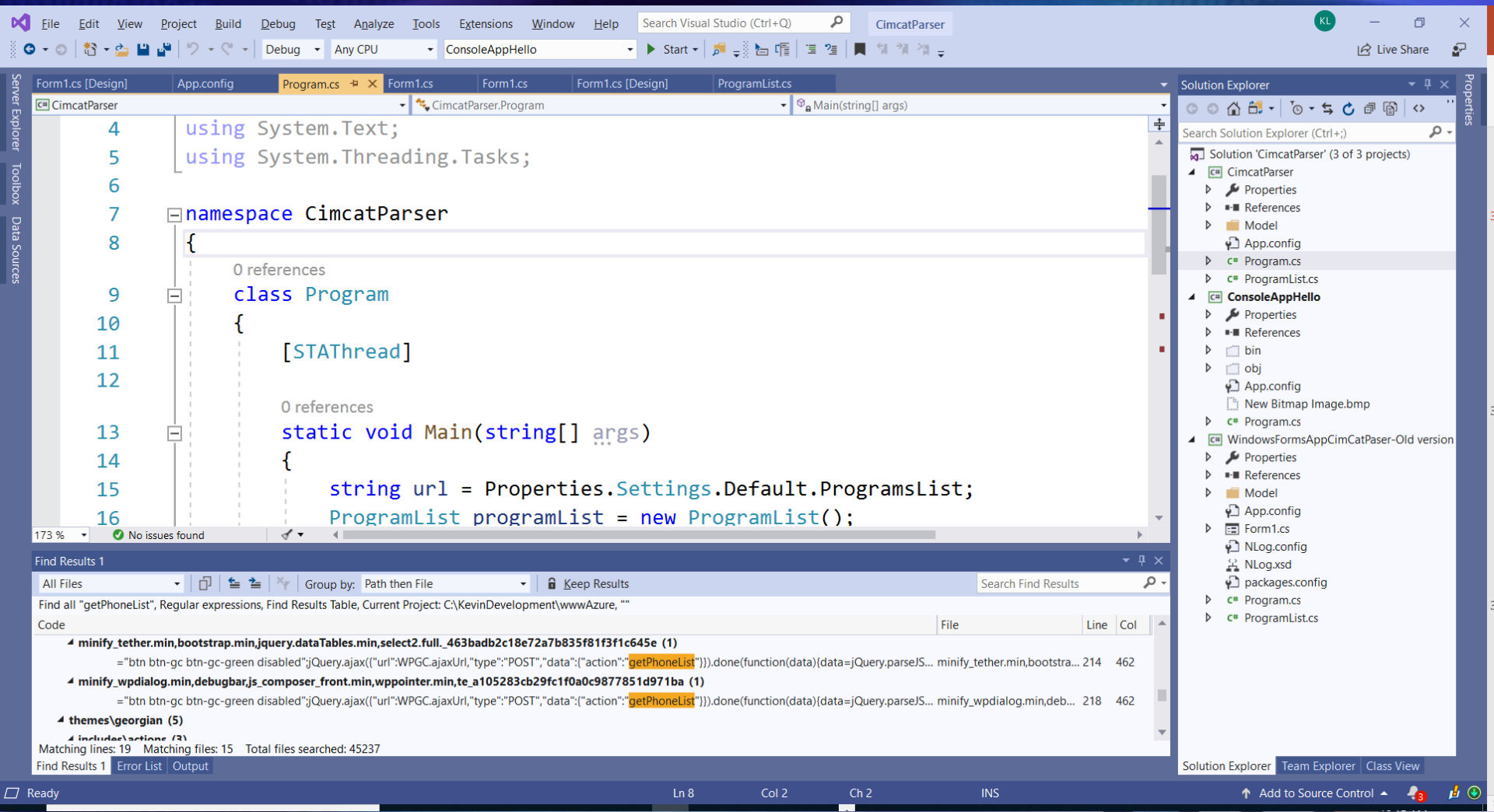
Visual Studio .NET

- 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
  - Project templates can be installed
- VS.NET is not part of the .NET Framework
  - Not necessary to build or run managed code
  - Because 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

Visual Studio .NET



Sample codes

- Program structure
- Basic syntax
- Project folders and files

Summary

- .NET Framework is a code execution platform – the environment which .NET programs run
- .NET Framework consists of two primary parts: Common Language Runtime and .NET Class Libraries
- The CLS (Common Language Specification) allows different languages to interact seamlessly.
- The CTS (Common Type System) allows all languages to share base data types.

Summary (2)

- .NET languages are compiled to MSIL by their respective compilers
- MSIL code is compiled to machine code by the JIT compiler
- All .NET languages have equal access to the FCL (Framework Class Library) which is a rich set of classes for developing software

Microsoft .net

Assignment Requirements

- Try to research and understand. Do not copy long standard texts.
- Answer briefly in your own words so you can remember and talk about it when people ask (in interviews).

Assignment 1

- ***What is CLR?**
- *What is CTS?
- What is CLS?
- *What is MSIL?
- What is JIT?
- ***What is GAC?**
- **What is managed code?**
- **What is a garbage collector? Who manage it?**