

Introduction to .Net Framework

Before .NET

- Windows GUI development: Win32 API, MFC, Visual Basic, COM
- Web development: ASP
- Java – “Write once, run anywhere.”
- Embrace and extend: Visual J++

C/Win32 AP

- Traditional software development for the Windows.
 - C developers are forced to contend with complex **memory management** and **pointer** arithmetic.
 - It lacks the benefits provided by the object-oriented approach
- When you combine the **thousands of global functions** and data types defined by the Win32 API to an already formidable language, it is little wonder that there are so many **buggy** applications floating around today.

C++/MFC

- C++ is an object-oriented *layer* on top of C.
- **Microsoft Foundation Classes** (MFC) provides a set of C++ classes that facilitate the construction of Win32 applications.
- Regardless of the helpful MFC, programming for Windows using C++ remains a difficult and error-prone experience

Visual Basic 6.0 Programmer

- Ability to build complex user interfaces, code libraries, and data access logic with minimal fuss and bother.
 - VB6 **hides** the complexities of the raw Win32 API from view using integrated code wizards, intrinsic data types, classes, and VB-specific functions.
- Not fully object-oriented
 - No “is-a” relationships between types (i.e., no classical inheritance)

Java/J2EE

- Object oriented with syntactic roots in C++.
 - Java **cleans up** many unsavory syntactical aspects of C++.
 - Java provides programmers with a large number of predefined **packages** that contain various type definitions.
- Limited ability to access non-Java APIs.
- Little support for true **cross-language** integration.
 - Not appropriate for many **graphically** or **numerically** intensive applications.
 - A better approach for such programs would be to use a language such as C++ where appropriate.

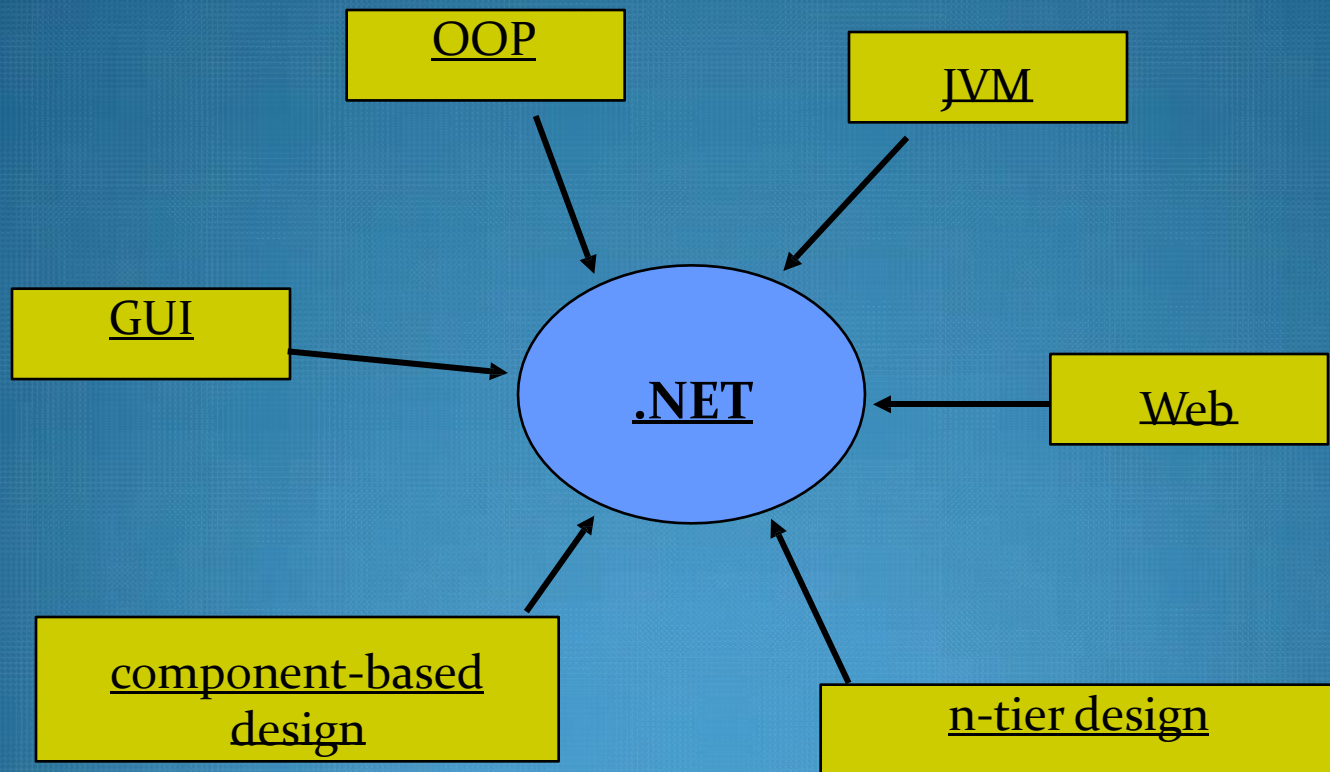
COM

- Microsoft's previous application development framework.
 - *reusable binary code*.
 - C++ programmers can build COM classes that can be used by VB6. Delphi programmers can use COM classes built using C.
- COM's language independence is limited.
 - COM has no support for *classical inheritance*).
- COM is extremely *complex* under the hood.
 - The *Active Template Library* (ATL) provides a set of C++ classes, templates, and macros to *ease* the creation of COM types.

Windows DNA

- Microsoft has been adding more Internet-aware features into its family of operating systems and products.
 - COM-based Windows **Distributed interNet Applications Architecture** (DNA) is quite complex.
 - Due to the simple fact that Windows DNA requires the use of **numerous technologies and languages** (ASP, HTML, XML, JavaScript, VBScript, COM(+), and data access API like ADO).

.Net



.Net provides

- Integrated environment
- Internet, Desktop , Mobile devices
- consistent object-oriented
- To provide a **portable** environment
- A managed environment

What Is .NET

- .NET is a framework
- New programming methodology
- .NET is platform independent / cross platform
- .NET is language-insensitive

Cont..

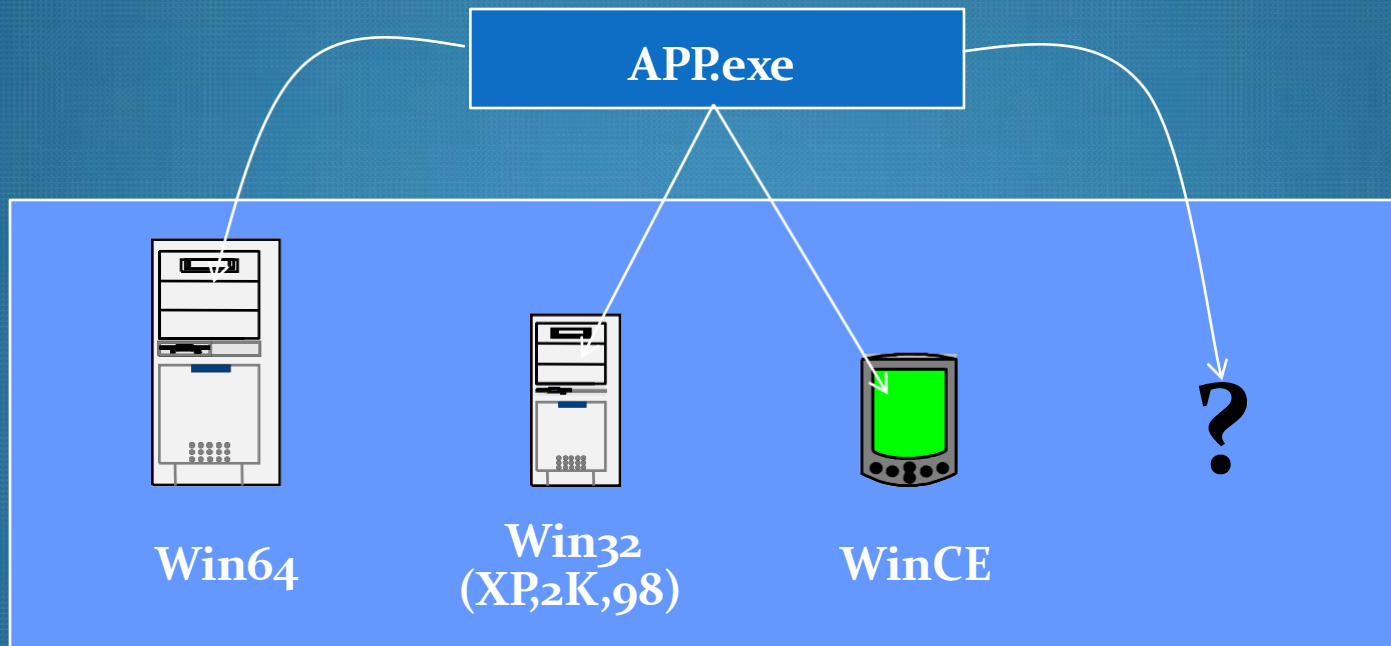
➤ NET Framework (pronounced dot net) is a software framework developed by Microsoft that runs primarily on Microsoft Windows.

■ It includes a large class library known as **Framework Class Library (FCL)** and provides language interoperability (each language can use code written in other languages) across several programming languages

Cont..

- **The Framework Class Library (FCL)** is a standard library and Microsoft's .NET Framework implementation of the Standard Libraries as defined in the Common Language Infrastructure
- System.Globalization, System.IO, System.Resources

.NET is cross-platform



Mono Project

Cross platform, open source .NET framework




Mono is a software platform designed to allow developers to easily create cross platform applications

Sponsored by [Xamarin](#), Mono is an open source implementation of Microsoft's .NET Framework based on the [ECMA](#) standards for [C#](#) and the [Common Language Runtime](#). A growing family of solutions and an active and enthusiastic contributing community is helping position Mono to become the leading choice for development of cross platform applications.

Get Mono

The latest Mono release is waiting for you!

 [Download](#)

Read the docs

We cover everything you need to know, from configuring Mono to how the internals are implemented.

Our documentation is open source too, so you can help us improve it.

 [Learn more](#)

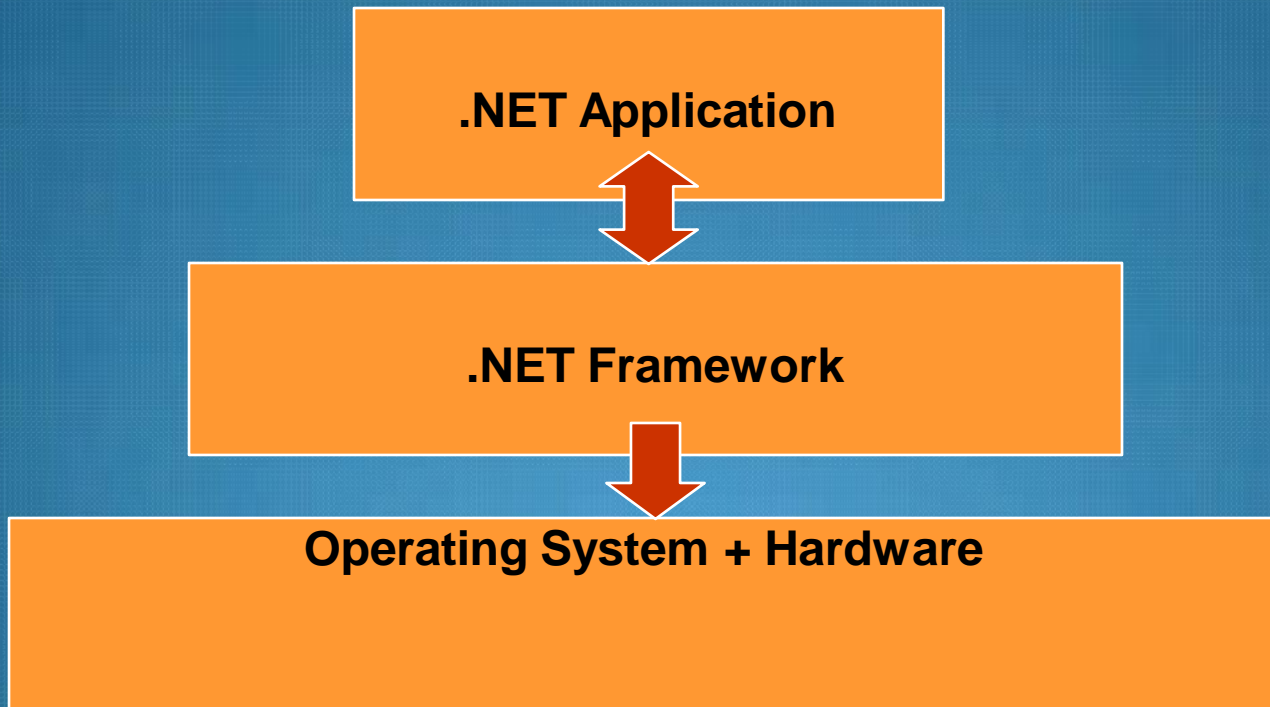
Community

As an open source project, we love getting contributions from the community.

File a bug report, add new code or chat with the developers.

 [Contribute to Mono](#)

Narrow view of .Net applications



.Net Framework

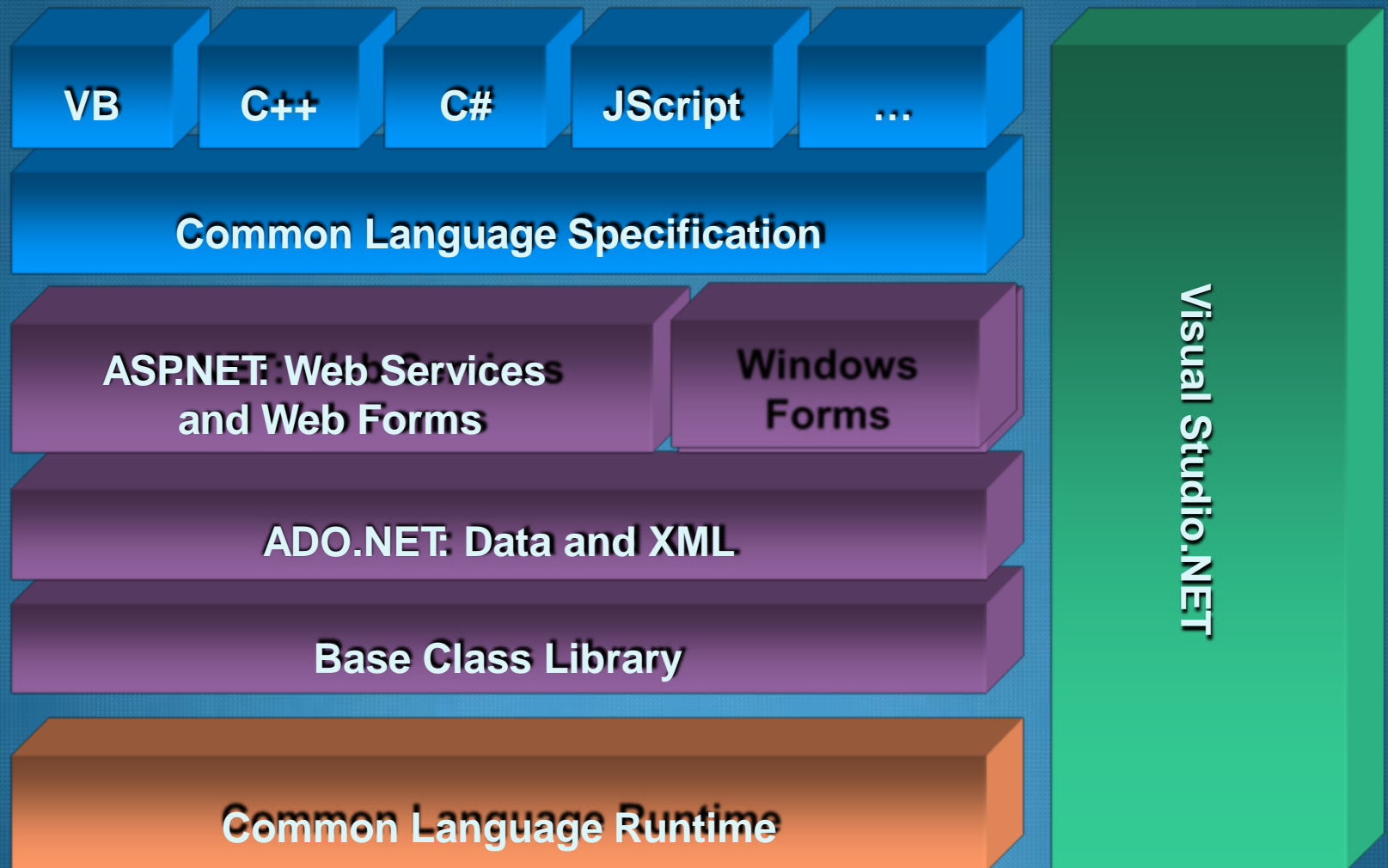
.Net Architecture

- .NET architecture is:
 - multi-language
 - cross-platform
 - based on the CLR, FCL, and JIT technology
 - .NET components are packaged as assemblies

.Net Architecture

Web services	UDDI, WSDL, Passport
Open interchange formats	XML & SOAP
Frameworks & libraries	ASP.NET, ADO.NET, Windows Forms, Remoting, Serialization...
Specific language compilers	C#, Visual Basic.Net, Managed C++, Cobol, Eiffel for .NET...
Language interoperability	Common Language Specification (CLS)
Development environment	Visual Studio.Net
Compilation, execution...	Common Language Runtime (CLR)
Underlying platform	Hardware, Operating system, database system

.Net Technical Architecture



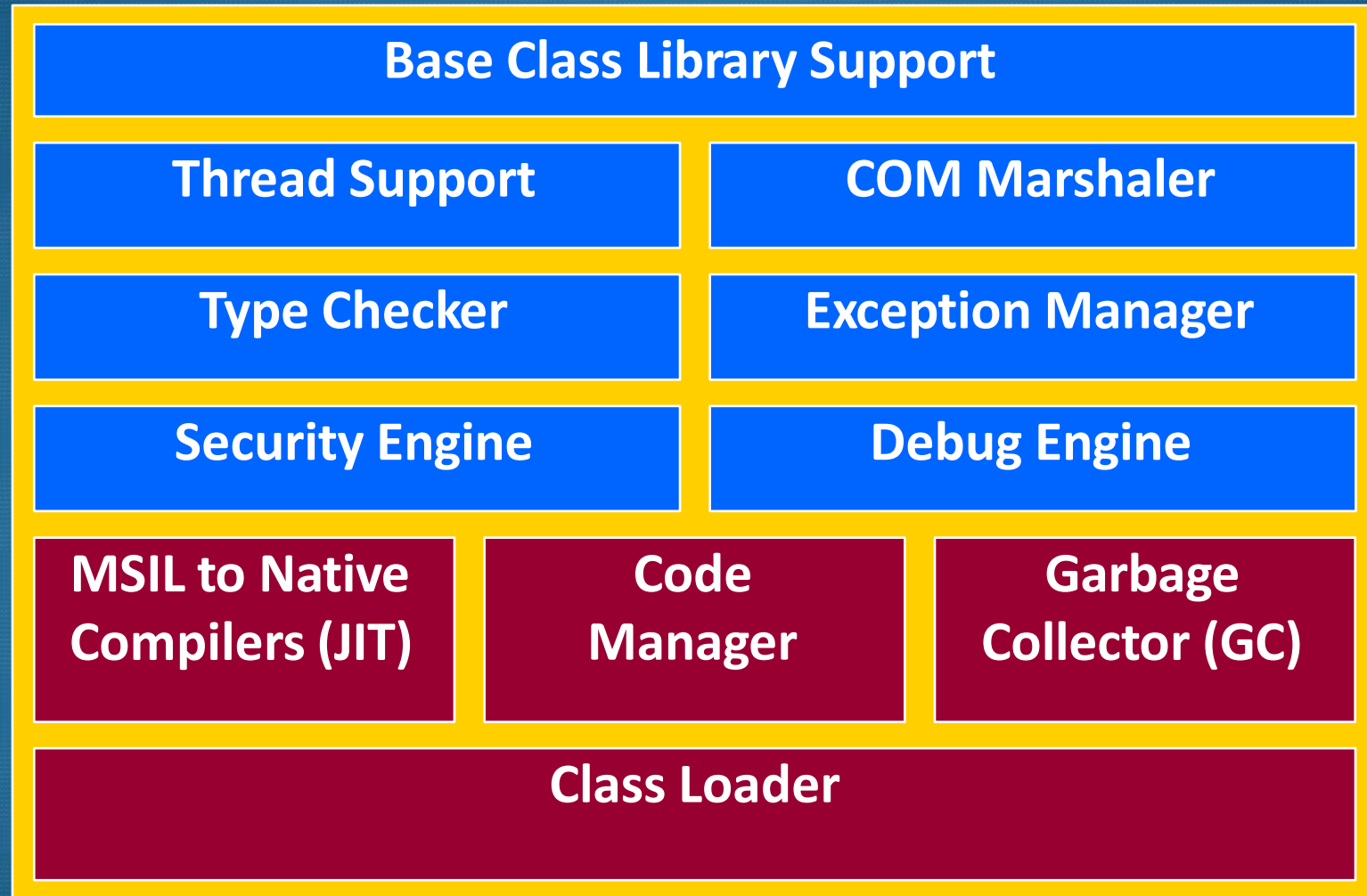
Common Language Runtime

- A common runtime for all .NET languages
 - Common type system
 - Common metadata
 - Intermediate Language (IL) to native code compilers
 - Memory allocation and garbage collection
 - Code execution and security
- Over 15 languages supported today
 - C#, VB, Jscript, Visual C++ from Microsoft
 - Perl, Python, Smalltalk, Cobol, Haskell, Mercury, Eiffel, Oberon, Oz, Pascal, APL, CAML, Scheme, etc.

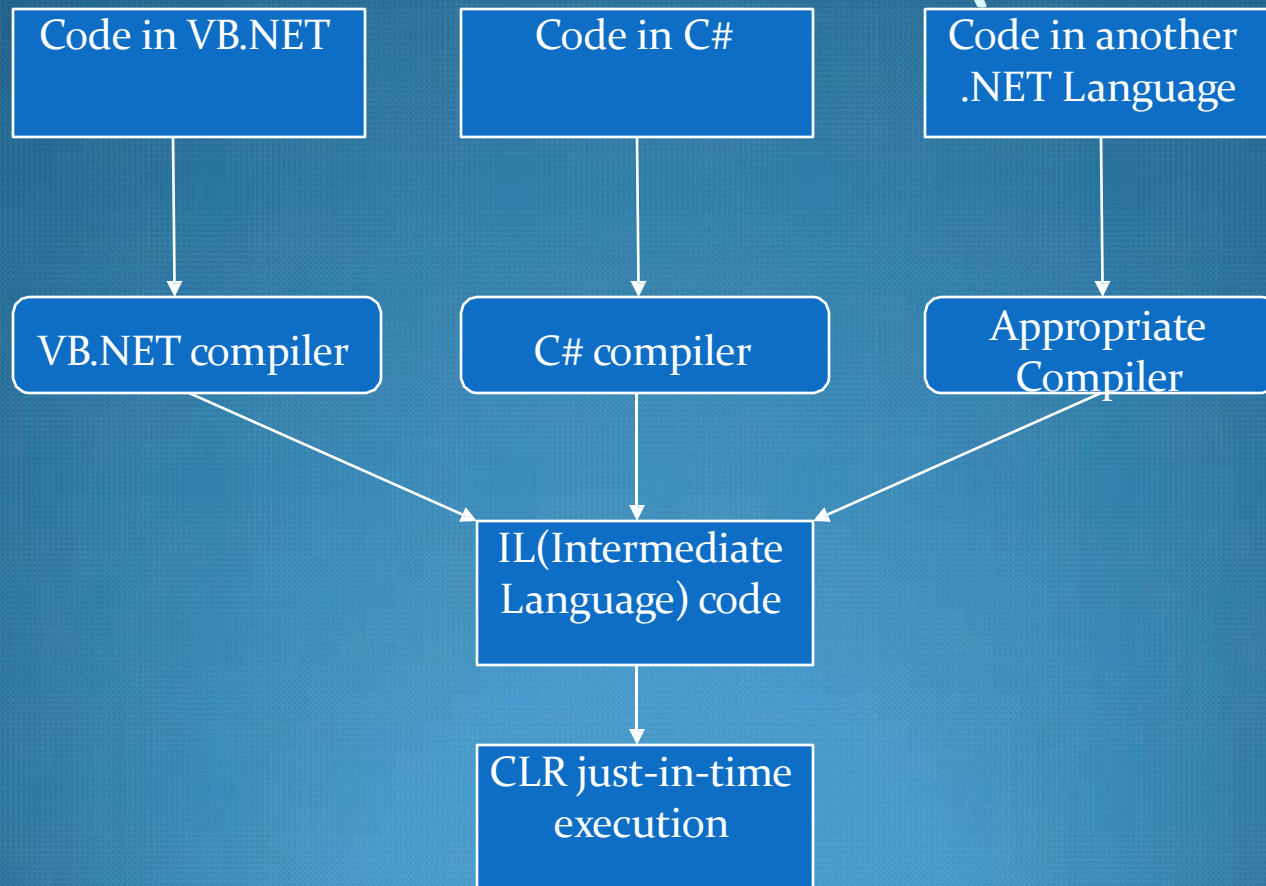


Next class

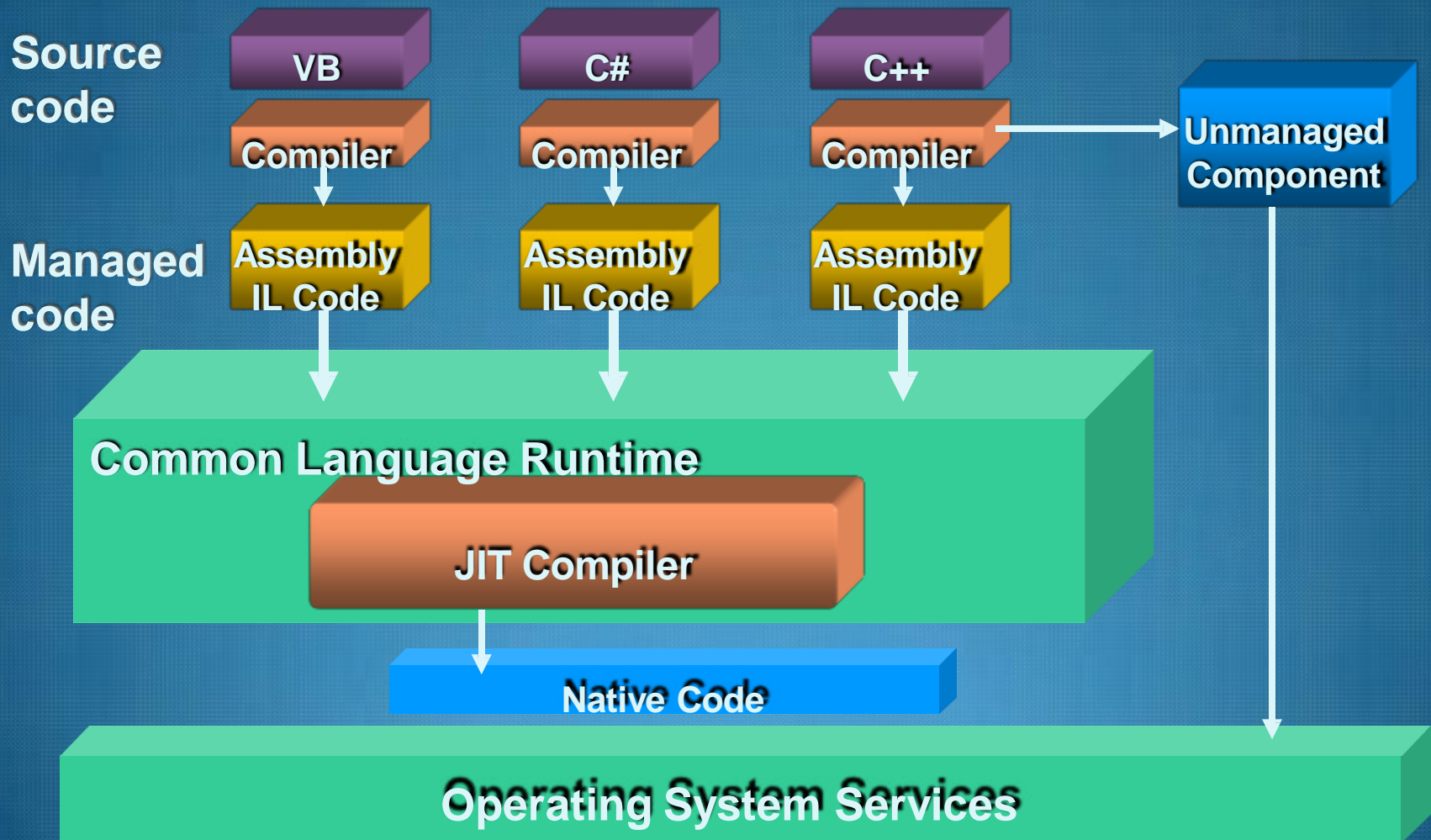
The CLR Architecture



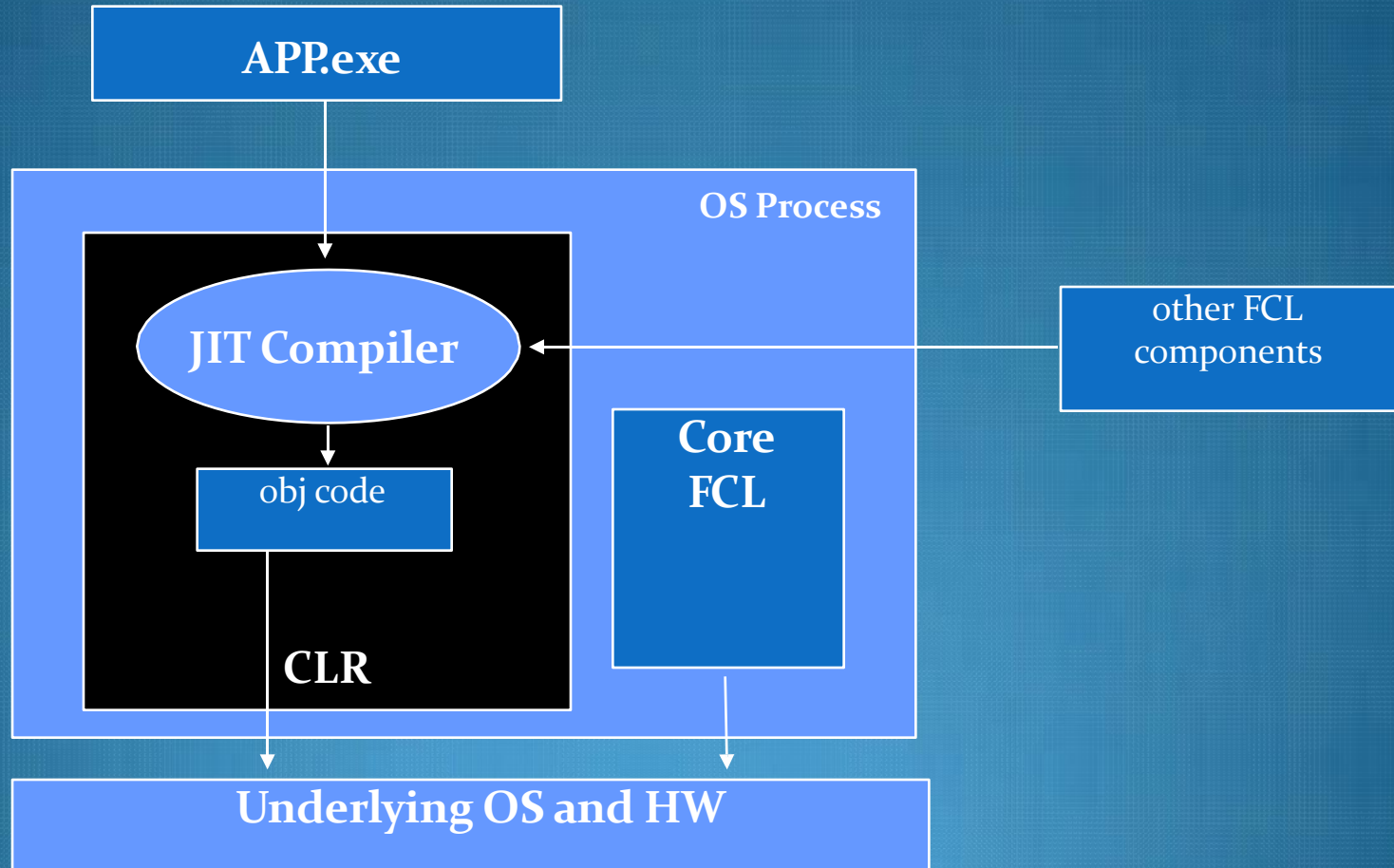
CLR Execution Model (Narrow)



CLR Execution Model



CLR based execution



Common Language Runtime

- Execution Engine
 - Compiles Microsoft Intermediate Language (MSIL) into native code
 - Handles garbage collection
 - Handles exceptions
 - Enforces code access security
 - Handles verification
 - Managed v. Unmanaged

Implications of CLR execution model

1. Clients need CLR & FCL to run .NET apps
 - available via *Redistributable .NET Framework*
2. Design trade-off...
 - + managed execution (memory protection, verifiable code, etc.)
 - + portability:
 - slower execution?

CIL and JIT

- When you compile code that uses the .NET Framework library, you don't immediately create operating-system-specific native code.
- Instead, you compile your code into ***Common Intermediate Language (CIL) code***. This code isn't specific to any operating system (OS) and isn't specific to C#.

- Obviously, more work is necessary to execute an application. That is the job of a ***just-in-time (JIT)*** compiler, which compiles CIL into native code that is specific to the OS and machine architecture being targeted.
- Only at this point can the OS execute the application. The *just-in-time part of the name* reflects the fact that CIL code is compiled only when it is needed.

➤ 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.

Cont..

In addition to containing CIL, assemblies also include **meta information** (that is, information about the information contained in the assembly, also known as metadata) and **optional resources** (additional data used by the CIL, such as manifest ,sound files and pictures).

CLR and JIT compiling.

- Indirect execution of .Net applications.
- All .NET languages compile to the same CIL.
- The CLR transforms the CIL to assembly instructions for a particular hardware architecture.
 - This is termed jit'ing or Just-in-time compiling.
 - Some initial performance cost, but the jitted code is cached for further execution.
 - The CLR can target the specific architecture in which the code is executing, so some performance gains are possible.

Advantages of CLR

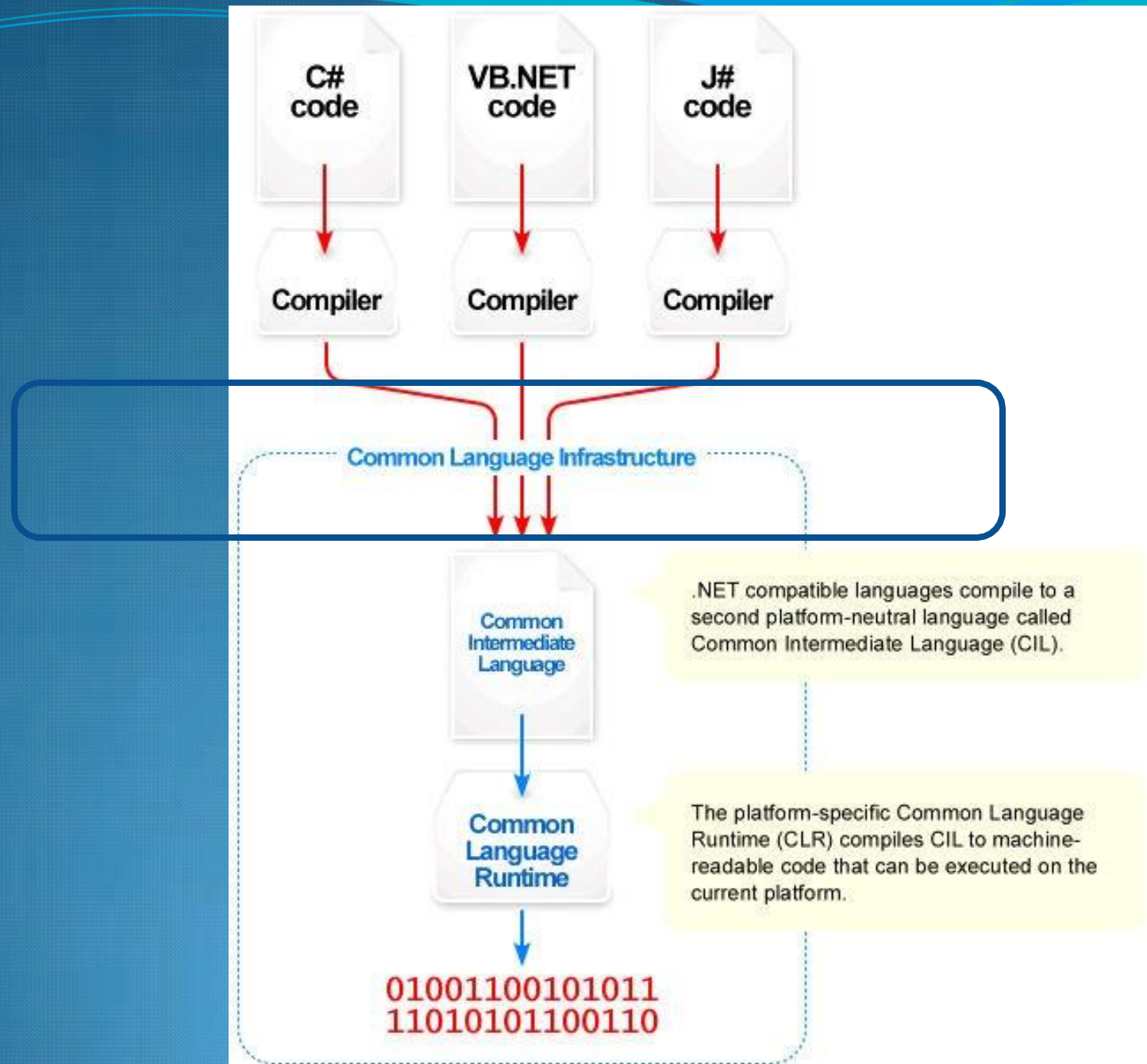
- Support for developer services (debugging)
- Interoperation between managed code and unmanaged code (COM, DLLs).
- Managed code environment
- Improved memory handling
- Improved Garbage collection

Advantages of CLR

- JIT allows code to run in a protected environment as managed code.
- JIT allows the IL code to be hardware independent.
- CLR also allows for enforcement of code access security.
- Verification of type safety.
- Access to Metadata (enhanced Type Information)

Common Language Infrastructure

CLI



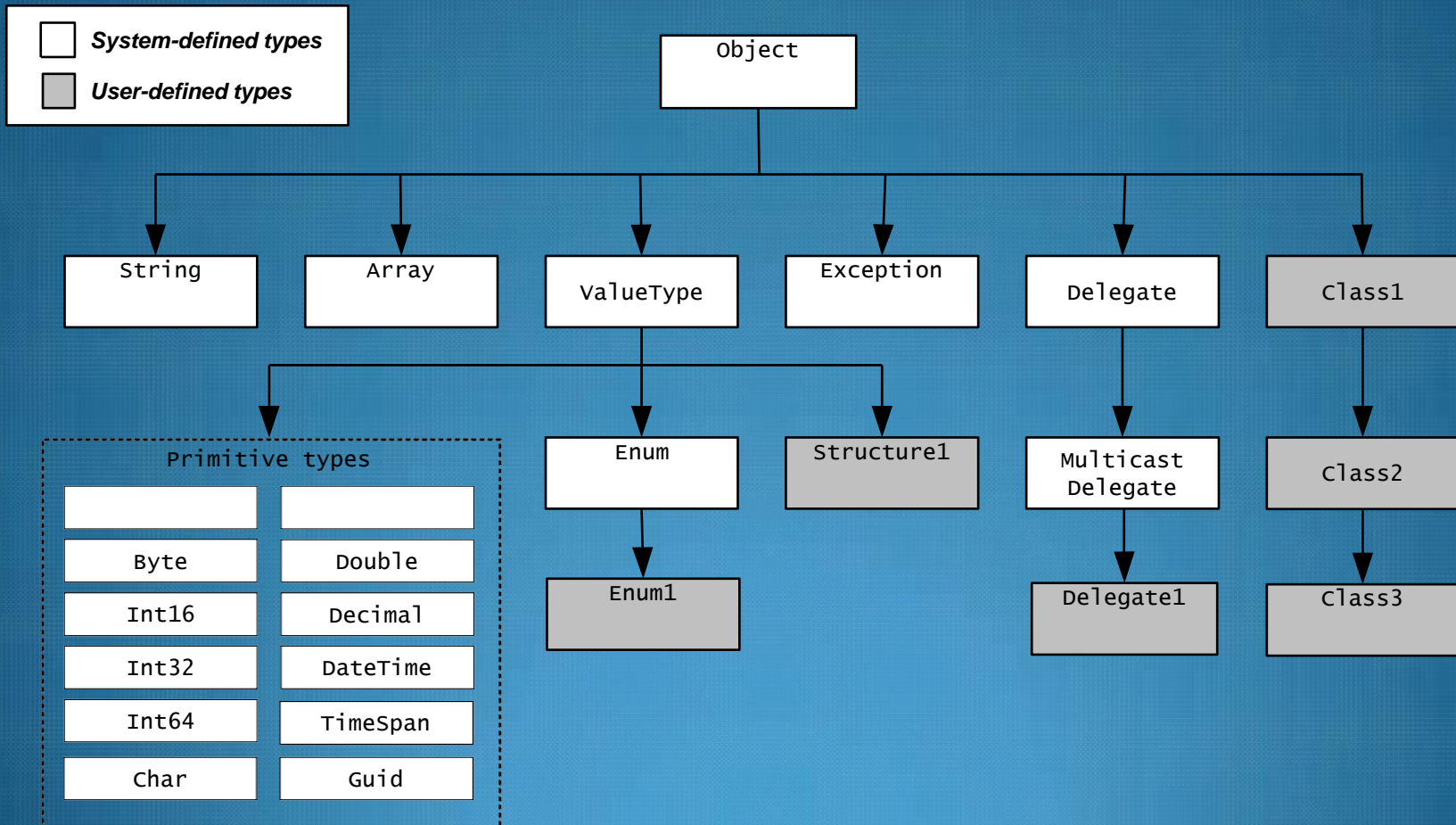
Common Language Infrastructure

- CLI allows for cross-language development.
- Four components:
 - Common Type System (CTS)
 - Meta-data in a language agnostic fashion.
 - Common Language Specification – behaviors that all languages need to follow
 - A Virtual Execution System (VES).

Common Type System (CTS)

- A specification for *how* types are *defined* and *how they behave*.
 - no syntax specified
- A type can contain zero or more members:
 - Field
 - Method
 - Property
 - Event

Common Type System (CTS)



CTS Data Types

CTS Data Type	VB .NET Keyword	C# Keyword	Managed Extensions for C++ Keyword
System.Byte	Byte	byte	unsigned char
System.SByte	SByte	sbyte	signed char
System.Int16	Short	short	short
System.Int32	Integer	int	int or long
System.Int64	Long	long	__int64
System.UInt16	UShort	ushort	unsigned short
System.UInt32	UInteger	uint	unsigned int or unsigned long
System.UInt64	ULong	ulong	unsigned __int64
System.Single	Single	float	Float
System.Double	Double	double	Double
System.Object	Object	object	Object^
System.Char	Char	char	wchar_t
System.String	String	string	String^
System.Decimal	Decimal	decimal	Decimal
System.Boolean	Boolean	bool	Bool

Common Data Types

CLR provides a set of primitive types that all languages must support. The data types include:

- Integer—three types 16/32/64 bits
- Float—two types: 32/64 bits
- Boolean and Character
- Date/time and Time span
- The primitive types can be collected into
 - Arrays
 - Structures
 - Combination of the two

Common Language Specification (CLS)

- Not all languages support all CTS types and features
 - C# is case sensitive, VB.NET is not
 - C# supports pointer types (in unsafe mode), VB.NET does not
 - C# supports operator overloading, VB.NET does not
- CLS was drafted to promote language interoperability
 - vast majority of classes within FCL are CLS-compliant

Comparison to Java



Source code

Byte code



Source code

CIL

Base Class Library @ FCL

Unified Classes

Web Classes (ASP.NET)

Controls, Caching, Security, Session, Configuration etc

Data (ADO.NET)

ADO, SQL, Types etc

Windows Forms

Design, Cmpnt Model etc

XML Classes

XSLT, Path, Serialization
etc

Drawing Classes

Drawing, Imaging, Text, etc

System Classes

Collections, Diagnostics, Globalization, IO,
Security, Threading Serialization, Reflection,
Messaging etc

Base Class Library

- Similar to Java's System namespace.
- Used by all .NET applications
- Has classes for IO, threading, database, text, graphics, console, sockets/web/mail, security, cryptography, COM, run-time type discovery/invocation, assembly generation

Example

Namespace	What for?
System.Text	ASCII, Unicode, etc. encoding
System.Text.RegularExpressions	Regular expressions handling
System.Threading	Thread support
System.Timers	Raising time-controlled events
System.Web.UI.WebControls	Graphical Web controls
System.Windows.Forms	Graphical controls: <i>Button...</i>

Intermediate Language (IL)

- .NET languages are not compiled to machine code. They are compiled to an Intermediate Language (IL).
- CLR accepts the IL code and recompiles it to machine code. The recompilation is just-in-time (JIT) meaning it is done as soon as a function or subroutine is called.
- The JIT code stays in memory for subsequent calls. In cases where there is not enough memory it is discarded thus making JIT process interpretive.

What is C#

- C# is an elegant and type-safe object-oriented language that enables developers to build a variety of secure and robust applications that run on the .NET Framework.
- It is one of the languages you can use to create applications that will run in the .NET CLR
- It is an evolution of the C and C++ languages and has been created by Microsoft specifically to work with the .NET platform.

Applications You Can Write with C#

- Windows applications
- Web applications: ASP.NET, ASP.NET MVC
- Web services :- cloud service
- Service:- WCF(windows communication Foundation):- is a framework for building service- oriented application.

Project Topics

- Customer Service Management
- Human Resource Management
- Prosperity Administration
- Payroll Management System

Proposal Template

- Title
- Introduction
- Objective
- Scope and limitation
- Functional Requirements

Due date: Oct 25, 2017.



Queries?

Thank you !!!