

Introduction to .Net Framework

Unit: 1



Venkat Shiva Reddy
Technical Consultant and Trainer

Agenda

- **Introduction to .Net Framework**
- **Components of .Net Framework**
- **Compilation and Execution Model**
- **The C# Language Essentials**
- **OO Programming**
- **LINQ**

A Rich History

2000

XML
Web Services

1981
PC



MS-DOS
BASIC

1990
GUI



Windows
Visual BASIC

1995
Internet



Microsoft®
.net

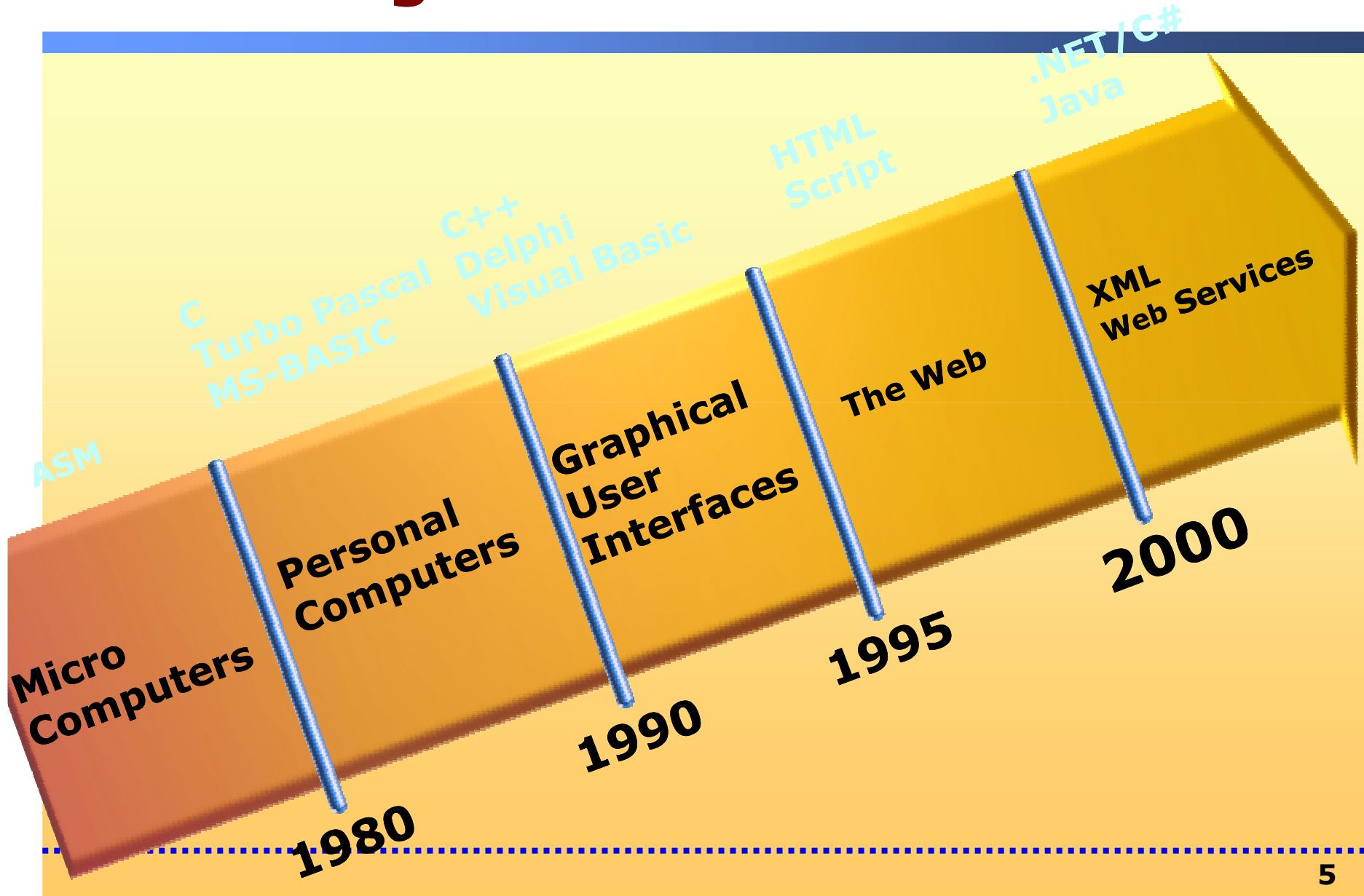
Visual
Studio
.NET

IE, IIS
Visual Studio

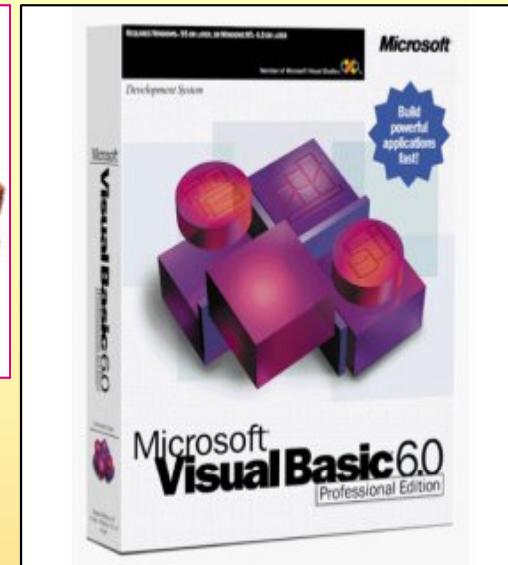
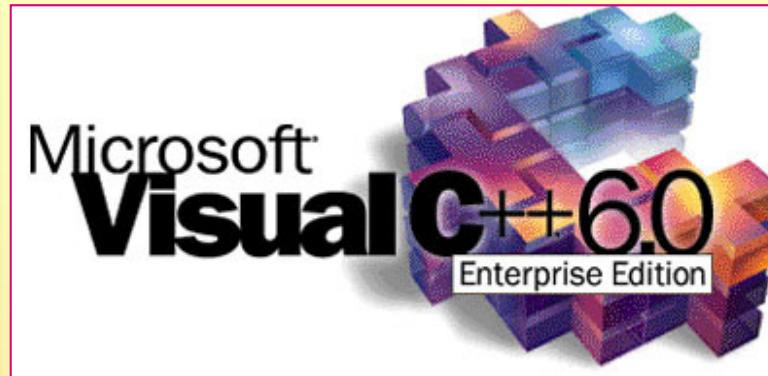
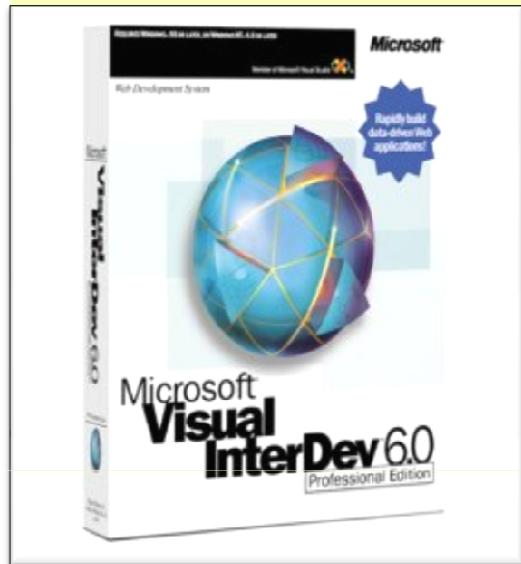
Before .NET

- **C/Win 32 API**
 - Unsafe (C) and complex (Win32)
- **C++ and MFC**
 - Better but still difficult
- **Visual Basic 6.0**
 - Nice, but not completely Object Oriented
- **COM**
 - Binary Compatibility, good model, still complex
 - Versioning problems
- **Java/J2EE**
 - Great, but not from Microsoft
 - GUI integration not optimal

Increasing Level of Abstraction

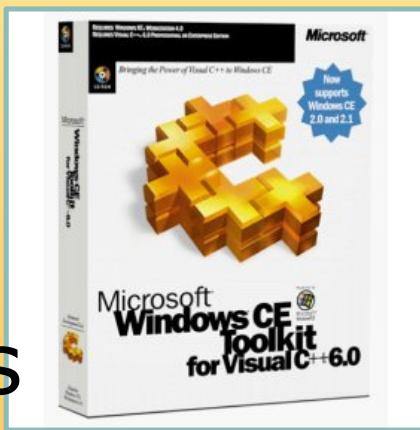


Why .NET?

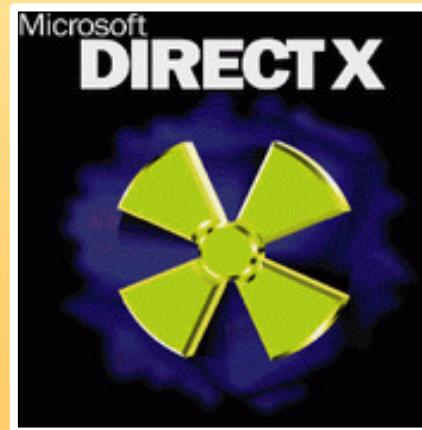


- Low Level
- Advance UI

Web



Devices



RAD

Advance
Graphics

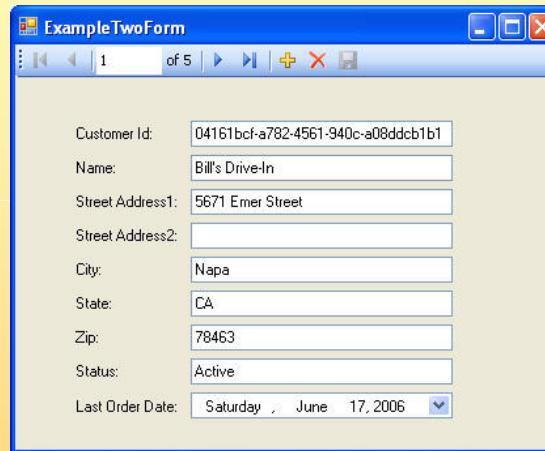
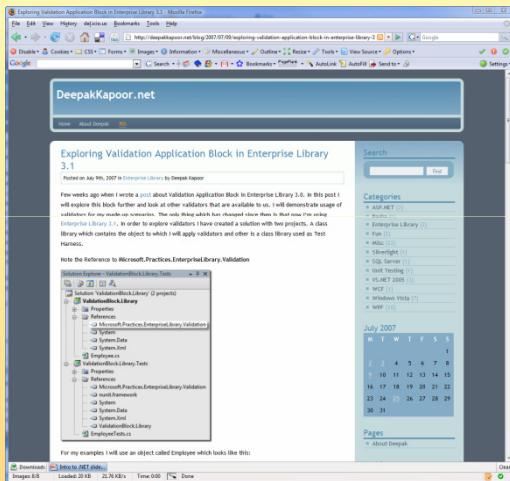
Why .NET?



- **Distinct Programming Models**
- **Learning Curve**
 - Rich Client
 - Thin Client
- **Different Job Different Language**



■ Consistent Programming Model (BCL)



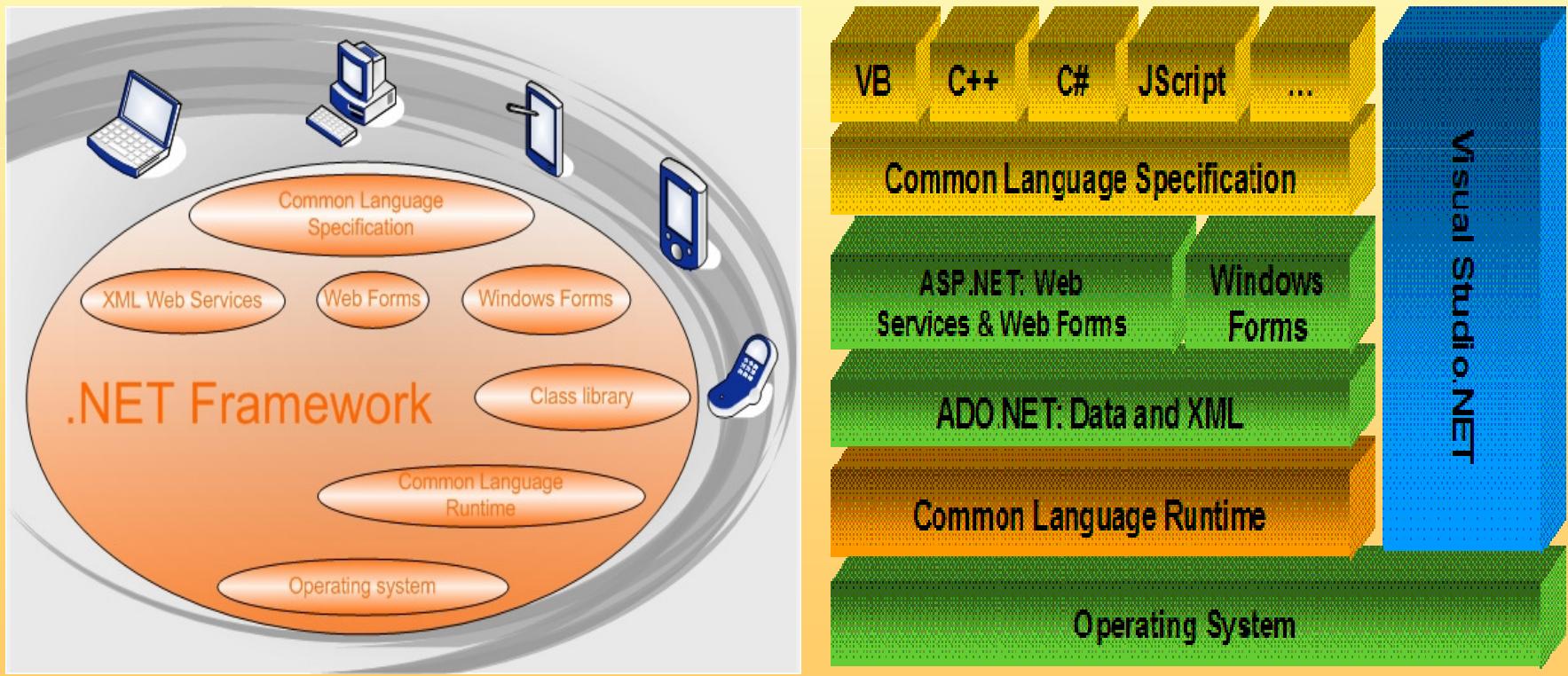
Web App's

Desktop App's

Mobile App's

What is .NET ?

- **.NET is a set of Microsoft software technologies**
 - for connecting information, people, systems, and devices.



What is a Framework?

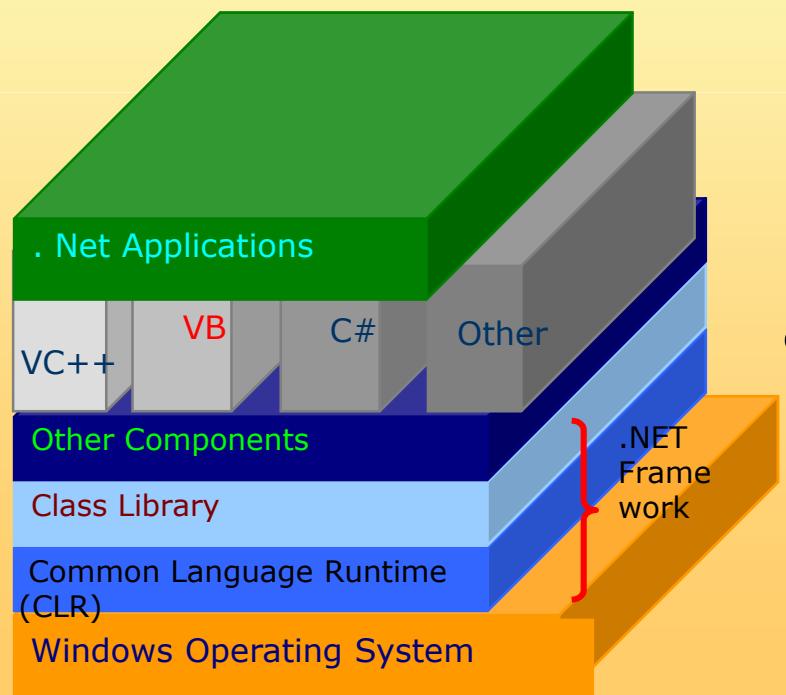
- A Framework is a “defined support structure” in which another software project can be organized and developed.
- Typically, a framework may include
 - support programs,
 - code libraries
 - and a language amongst other software to help develop and glue together the different components of your project.

What is .NET Framework?

- The .NET framework created by Microsoft is a software development platform focused on rapid application development, platform independence and network transparency.
- .NET is Microsoft's strategic initiative for server and desktop development for the next decade.
- According to Microsoft, .NET includes many technologies that are designed to facilitate rapid development of Internet and intranet applications.

What is .NET ?

- **.NET is a set of Microsoft software technologies for connecting information, people, systems, and devices.**



- **.NET Framework** is a common layer between .NET applications and Windows OS.
- **.NET Applications** are VB / VC++ / C# programs written on .NET Framework.

.Net Framework Design Goals

- **Simplify development**
 - More power, less plumbing
- **Unify programming models**
 - Across all languages and application types
- **Utilize web standards and best practices**
 - Rich XML, standard protocols, stateless
- **Easier to deploy, run, & maintain**
 - For components, versioning, availability

.NET Framework Components

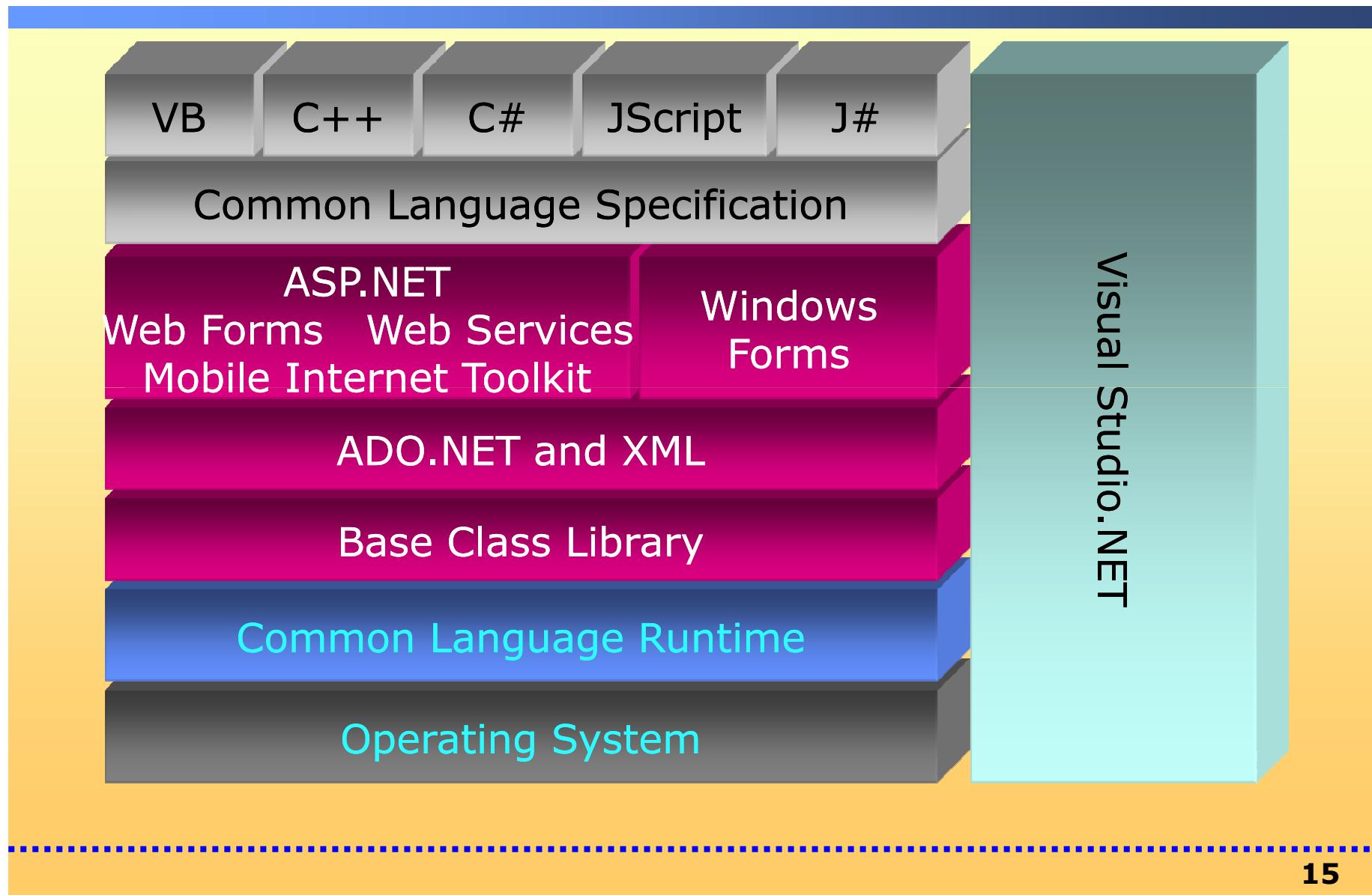
■ Common Language Runtime (CLR)

- Common type system for all languages
- Rich runtime environment

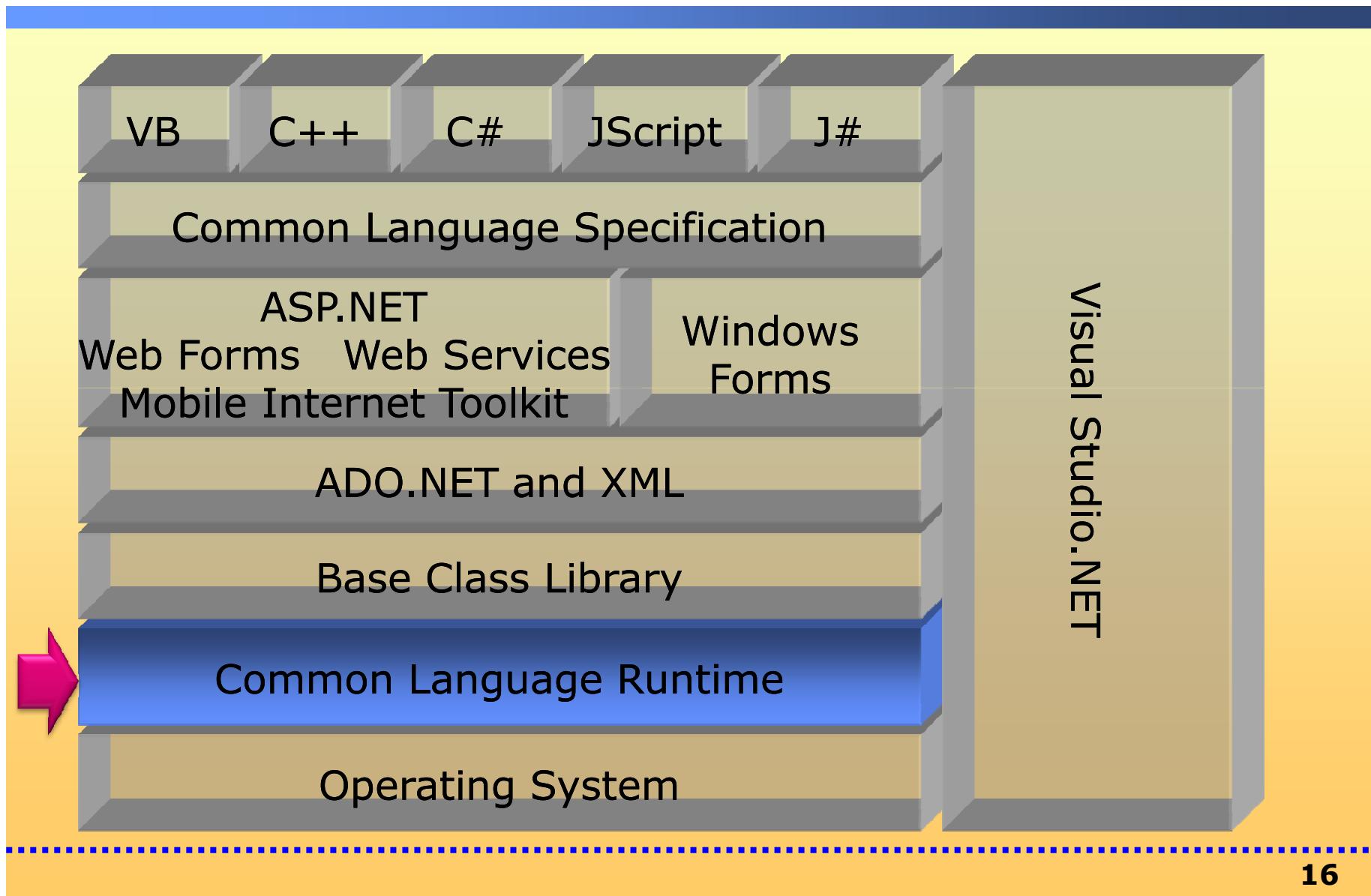
■ Rich class libraries (.NET Framework)

- Base class libraries, ADO.NET and XML
- Windows Forms for rich, Win32 applications
- Rich, interactive pages
- Powerful web services

Framework, Languages, And Tools

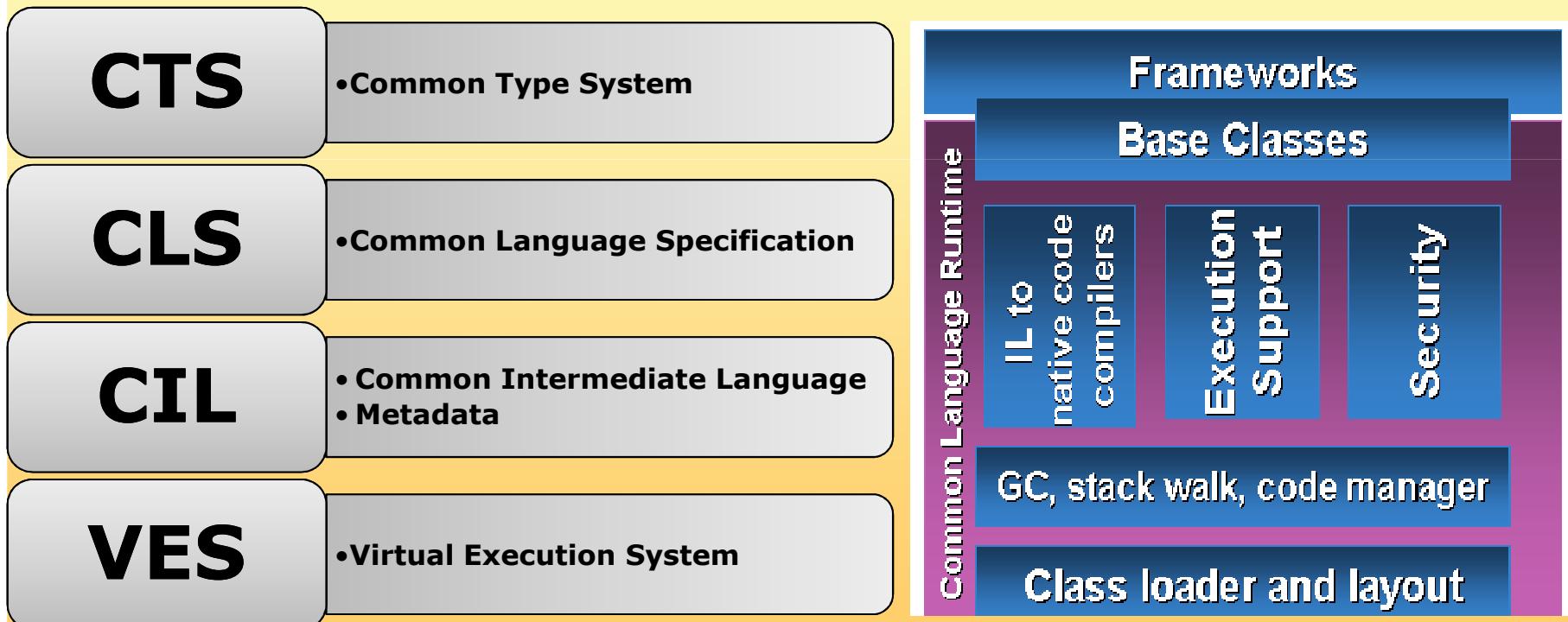


Common Language Runtime (CLR)

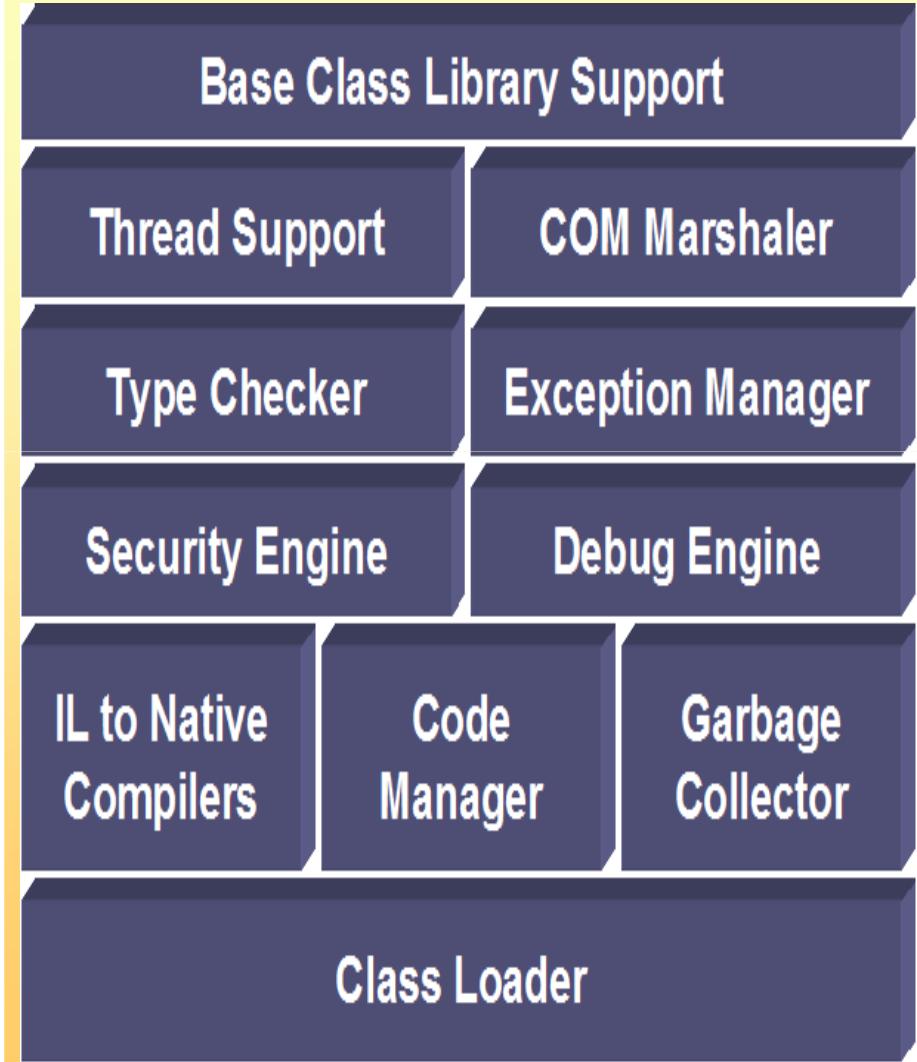


Common Language Runtime - CLR

- The Common Language Runtime is the execution engine for .NET Framework applications:

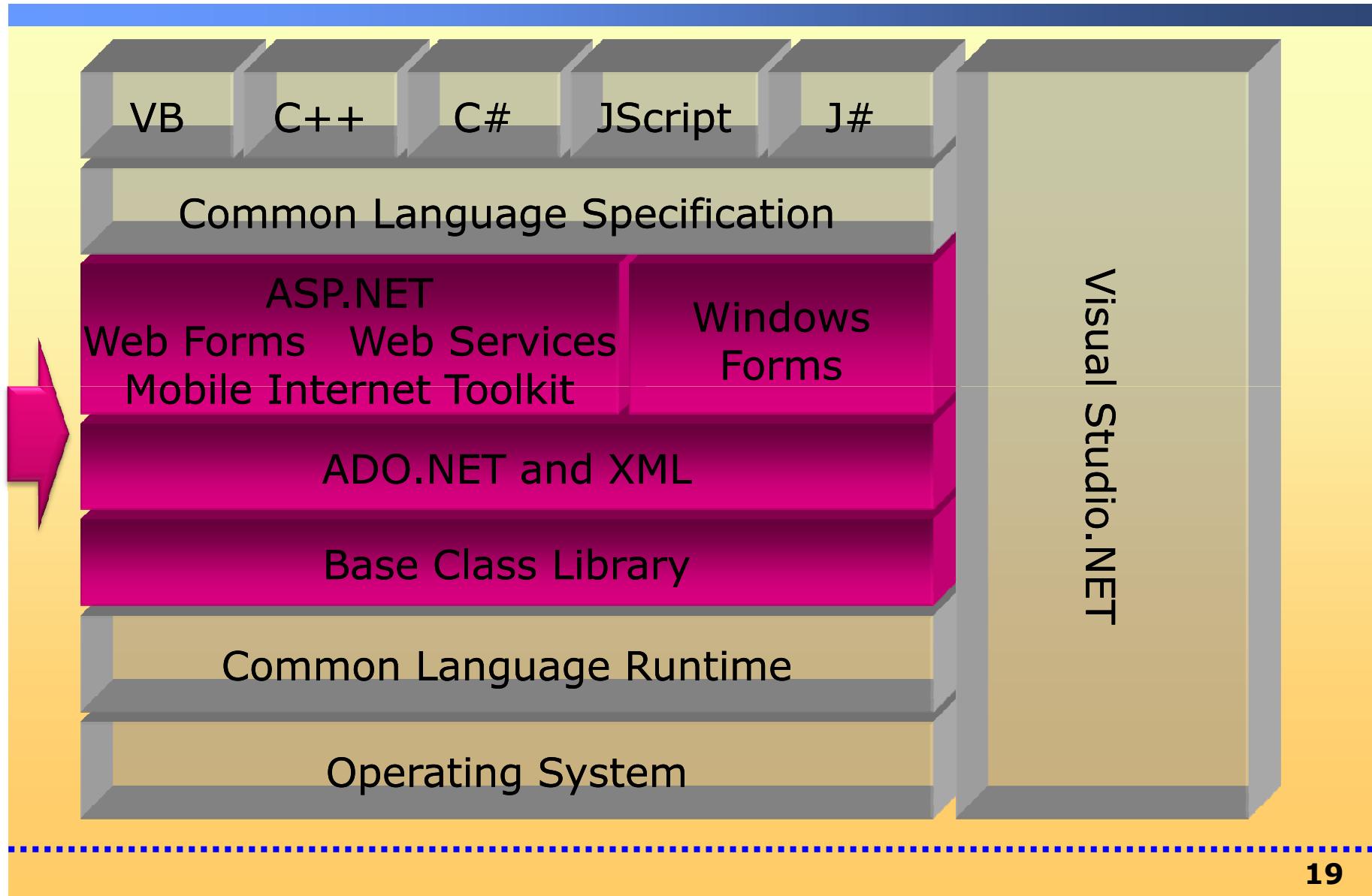


Common Language Runtime



- **Manages running code**
 - Verifies type safety
 - Provides garbage collection, error handling
 - Code access security for semi-trusted code
- **Provides common type system**
 - Value types (integer, float, user defined, etc)
 - Objects, Interfaces
- **Provides access to system resources**
 - Native API, COM interop, etc

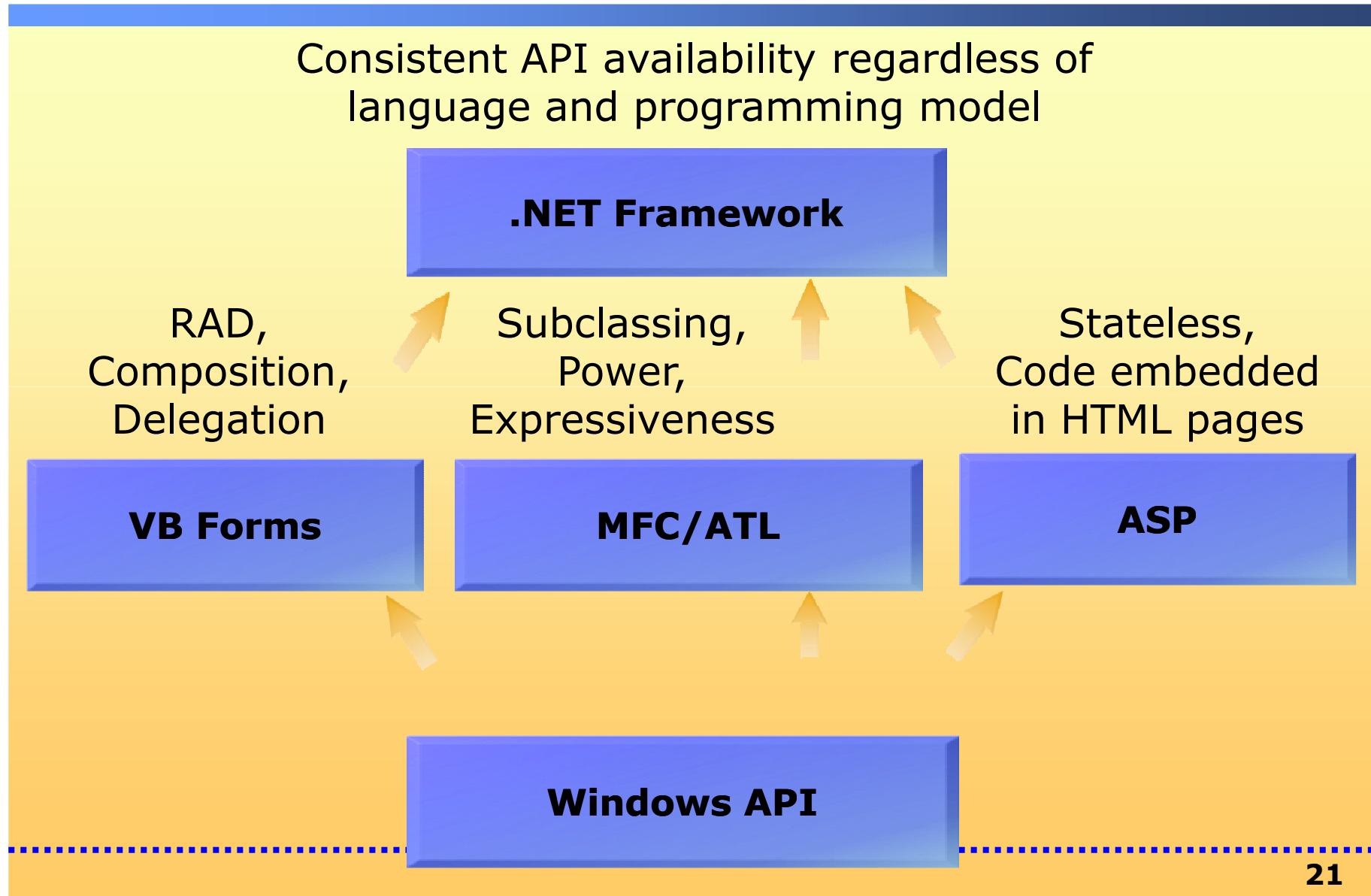
Base Class Library (BCL)



.NET Framework Libraries

- Single *consistent* set of object oriented class libraries to enable building distributed applications (**Unified Classes**)
- Built using classes arranged across logical hierarchical namespaces
 - For example: System.Data.SQL
- Work with all CLR languages
 - No more “VBRun” or “MFC” divide

Unify Programming Models



How Much Simpler?

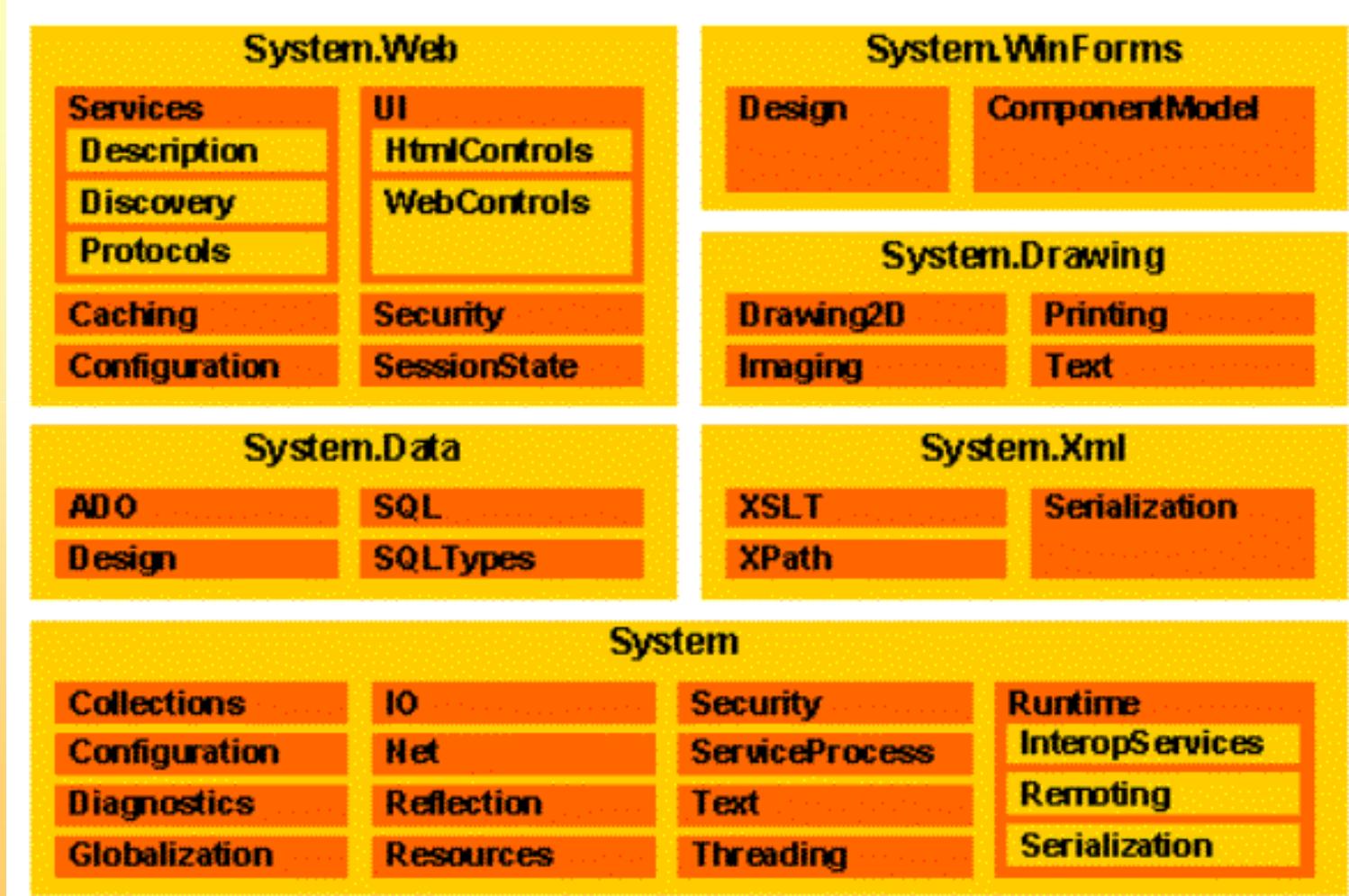
Windows API

```
HWND hwndMain = CreateWindowEx(
    0, "MainWClass", "Main Window",
    WS_OVERLAPPEDWINDOW | WS_HSCROLL | WS_VSCROLL,
    CW_USEDEFAULT, CW_USEDEFAULT,
    CW_USEDEFAULT, CW_USEDEFAULT,
    (HWND) NULL, (HMENU) NULL, hInstance, NULL);
ShowWindow(hwndMain, SW_SHOWDEFAULT);
UpdateWindow(hwndMain);
```

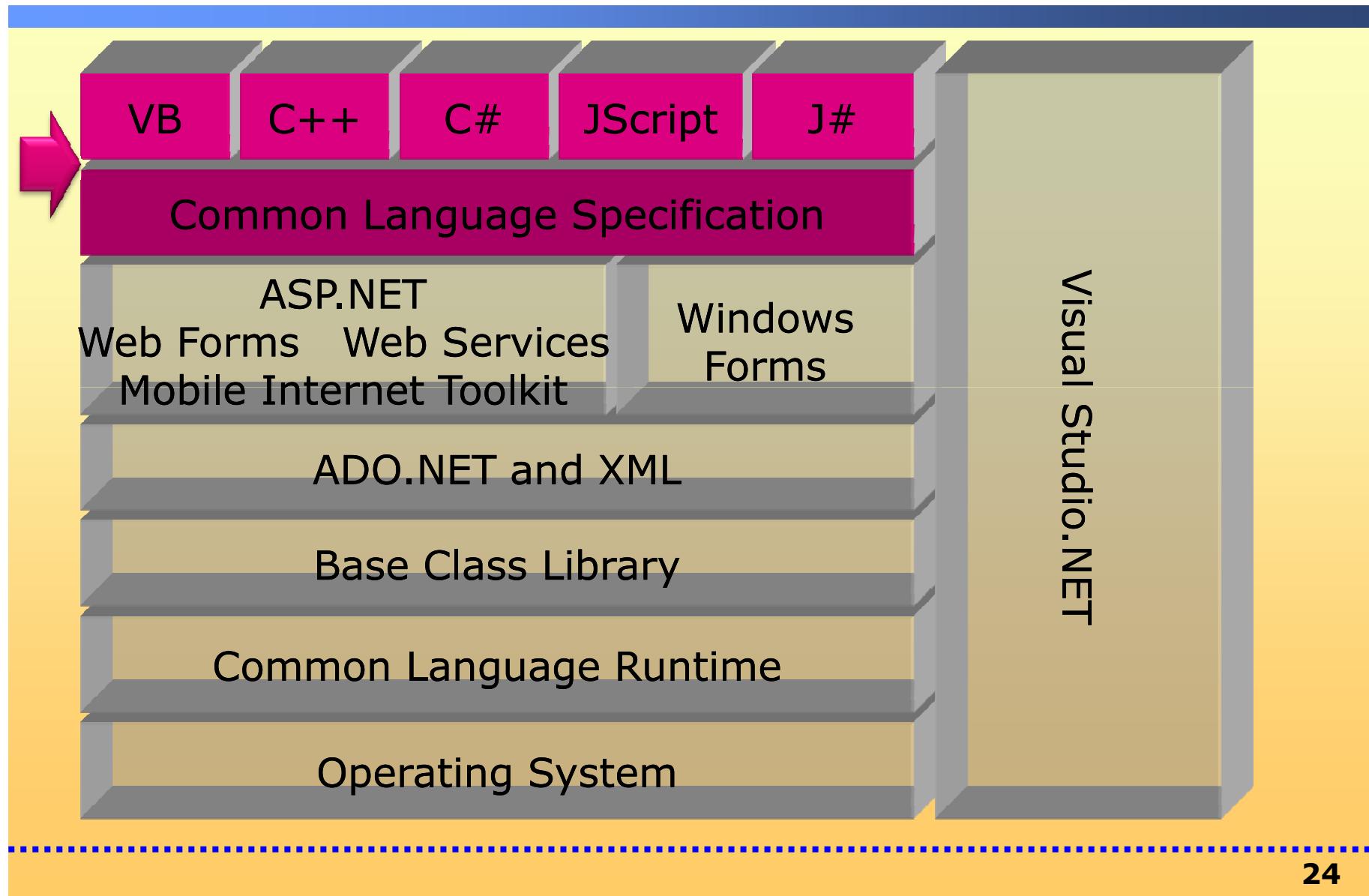
.NET Framework

```
Form form = new Form();
form.Text = "Main Window";
form.Show();
```

.NET Base Class Libraries



.NET Languages



.Net Languages

- **The .NET Platform is language neutral**
 - All .NET languages are first class players
 - You can leverage your existing skills
- **Common language specification**
 - Set of features guaranteed to be in all languages

.NET Languages from Microsoft

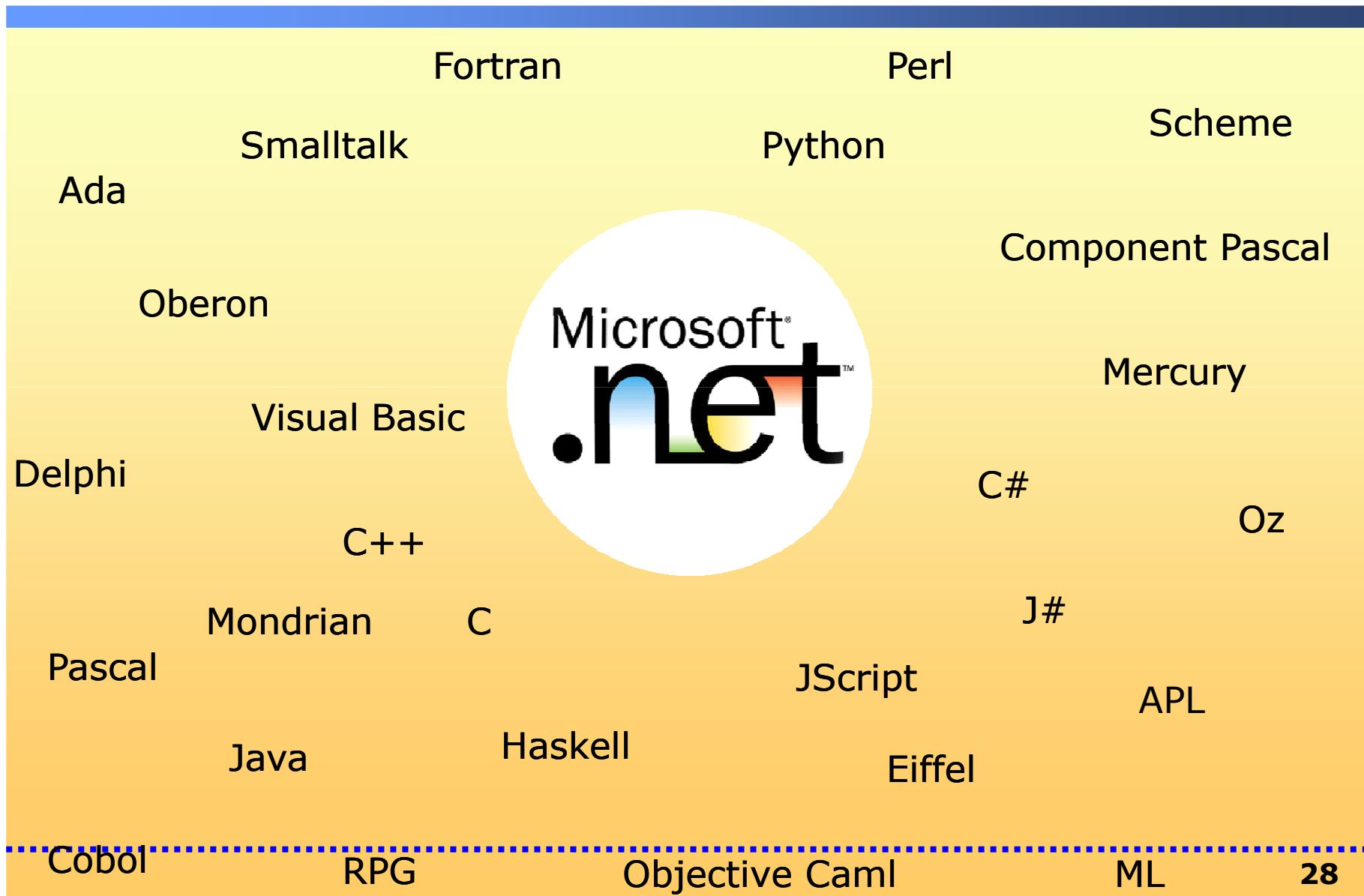
- **Microsoft: Visual Basic.NET**
- **Microsoft: C#**
- **Microsoft: F#**

- **Microsoft: C++ Managed**
- **Microsoft: Jscript.Net**
- **Microsoft: J#**

.NET Languages from Others

- APL
- Fujitsu COBOL
- Micro Focus COBOL
- Eiffel
- Forth
- FORTRAN 95
- Haskell
- Mercury
- Mondrian
- Oberon
- Pascal
- Perl
- Python
- RPG
- S#
- Scheme
- Standard Meta Language

Multilingual Development



The .NET Platform is multi-language

■ Benefits:

- Make code modules reusable
- Complete cross-language integration
- Make API access the same for all languages
- Let developers use the right language for the task and share the code

■ All .NET languages are first-class players

- Performance roughly equal
- Can use all .NET features

Multi Language Support

```
Dim s as String  
s = "authors"  
Dim cmd As New SqlCommand("select * from " & s, sqlconn)  
cmd.ExecuteReader()
```

VB.NET

```
string s = "authors";  
SqlCommand cmd = new SqlCommand("select * from "+s, sqlconn);  
cmd.ExecuteReader();
```

C#

```
String *s = S"authors";  
SqlCommand cmd = new SqlCommand(String::Concat(S"select * from ", s), sqlco  
cmd.ExecuteReader();
```

C++

```
String s = "authors";  
SqlCommand cmd = new SqlCommand("select * from "+s, sqlconn);  
cmd.ExecuteReader();
```

J#

```
var s = "authors"  
var cmd = new SqlCommand("select * from " + s, sqlconn)  
cmd.ExecuteReader()
```

JScript

```
s = "authors"  
cmd =SqlCommand("select * from " + s, sqlconn)  
cmd.ExecuteReader()
```

Python

Multi Language Support

```
ENVIRONMENT DIVISION.  
CONFIGURATION SECTION.  
REPOSITORY.  
    CLASS SqlCommand AS "System.Data.SqlClient.SqlCommand"  
    CLASS SqlConnection AS "System.Data.SqlClient.SqlConnection".  
DATA DIVISION.  
WORKING-STORAGE SECTION.  
01 str PIC X(50).  
01 cmd-string PIC X(50).  
01 cmd OBJECT REFERENCE SqlCommand.  
01 sqlconn OBJECT REFERENCE SqlConnection.  
PROCEDURE DIVISION.  
    *> Establish the SQL connection here somewhere.  
MOVE "authors" TO str.  
STRING "select * from " DELIMITED BY SIZE,  
      str DELIMITED BY " " INTO cmd-string.  
INVOKE SqlCommand "NEW" USING BY VALUE cmd-string sqlconn RETURNING cmd.  
INVOKE cmd "ExecuteReader".
```

COBOL

```
assembly_external(name="System.Data.SqlClient.SqlCommand")  
sqlcmdcharacter*10 xsq1cmd  
Cmd x='authors'  
cmd = sq1cmd("select * from //x, sq1conn)  
call cmd.ExecuteReader()  
end
```

Fortran

Multi Language Support

```
|s| := 'authors'.
|cmd| := SqlCommand('select * from '+s, sqlconn).
cmd.ExecuteReader().
```

Smalltalk

```
(let* ( (s "authors")
        (cmd (new-SqlCommand (string-append "select * from " s) sqlconn)
        (execute-command cmd))
```

Scheme

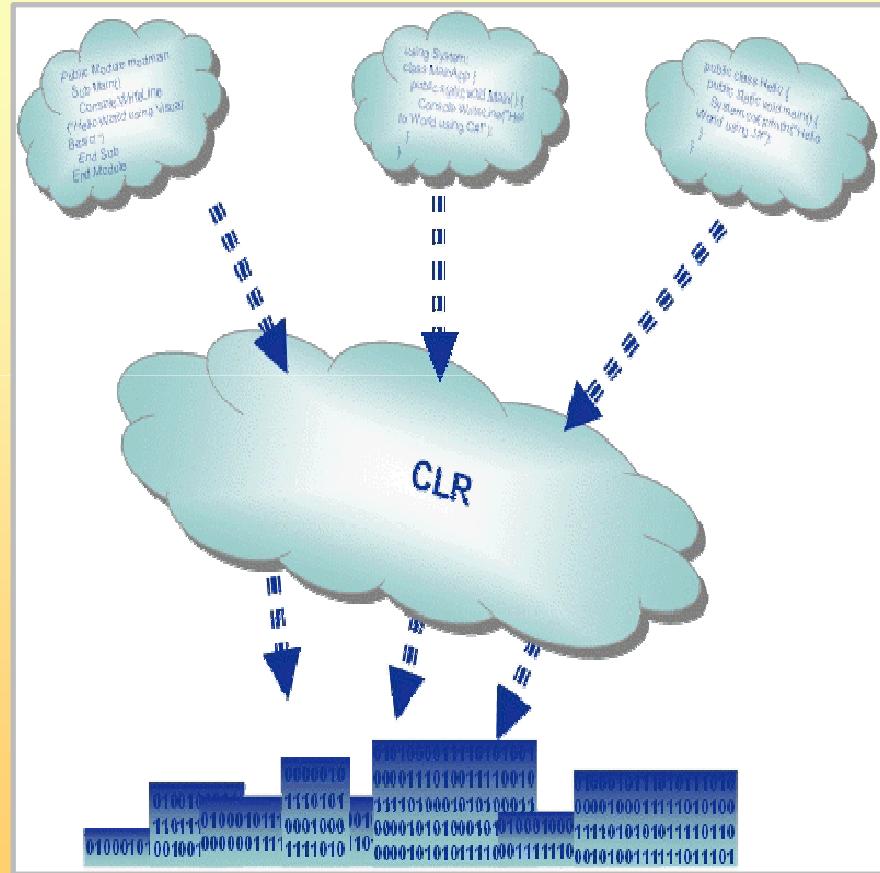
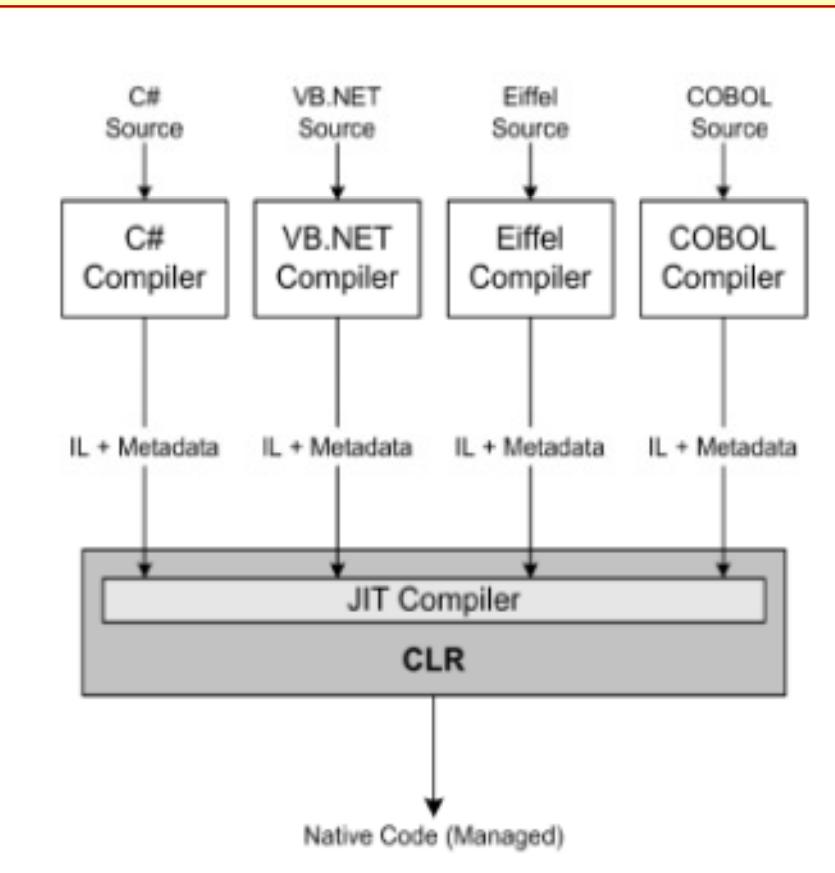
```
local
    s: STRING
    cmd: SQLCOMMAND
do
    s := "authors"
    create cmd("select * from " + s, sqlconn)
    cmd.ExecuteReader()
end
```

Eiffel

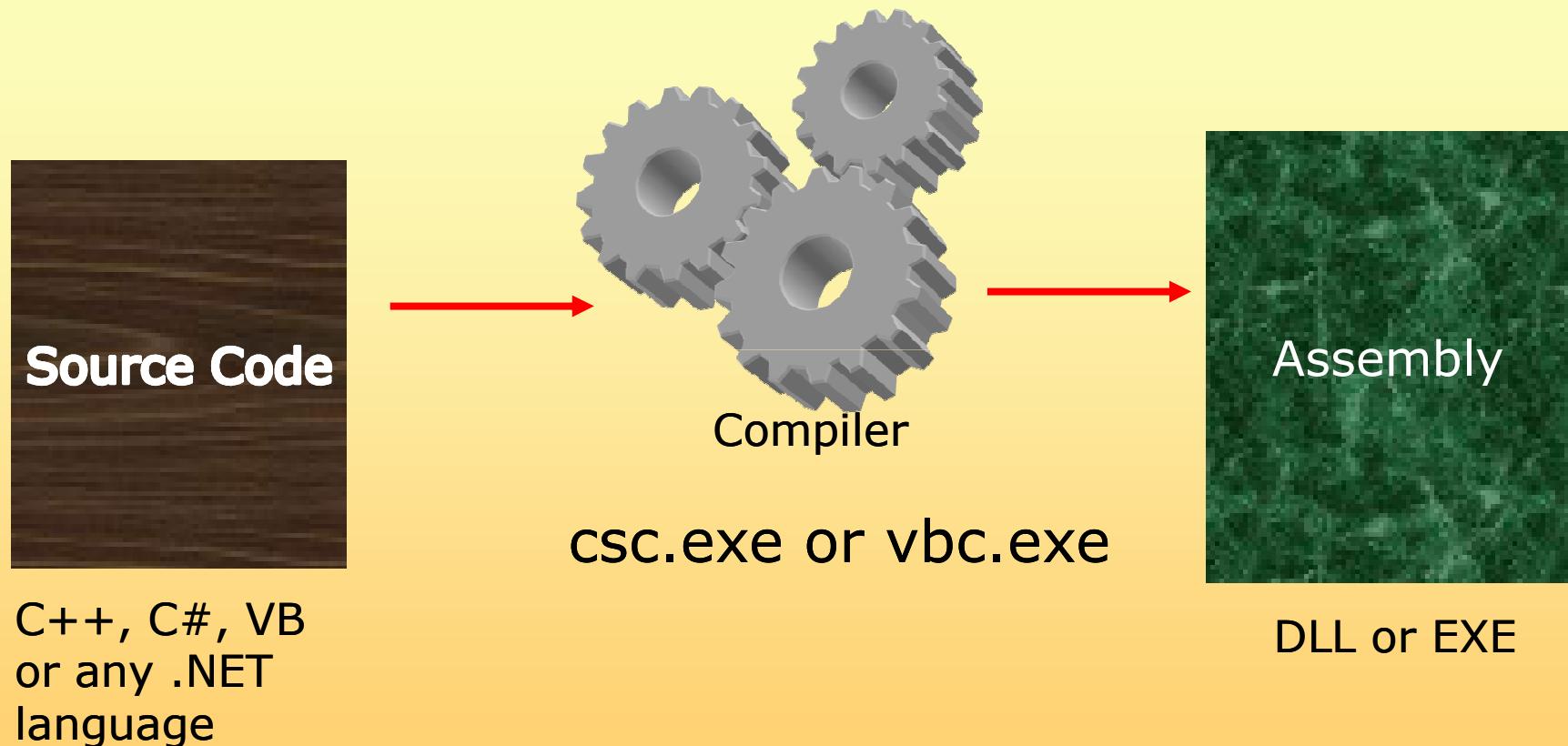
```
ExecuteReader = invoke System.Data.SqlClient.ExecuteReader();
SqlCommand = create System.Data.SqlClient.SqlCommand(String,\n            System.Data.SqlClient.SqlConnection);
query = sqlconn -> let{ s = "authors"; } in {
    cmd <- SqlCommand ("select * from "+s, sqlconn);
    cmd # ExecuteReader();
};
```

Mondrian

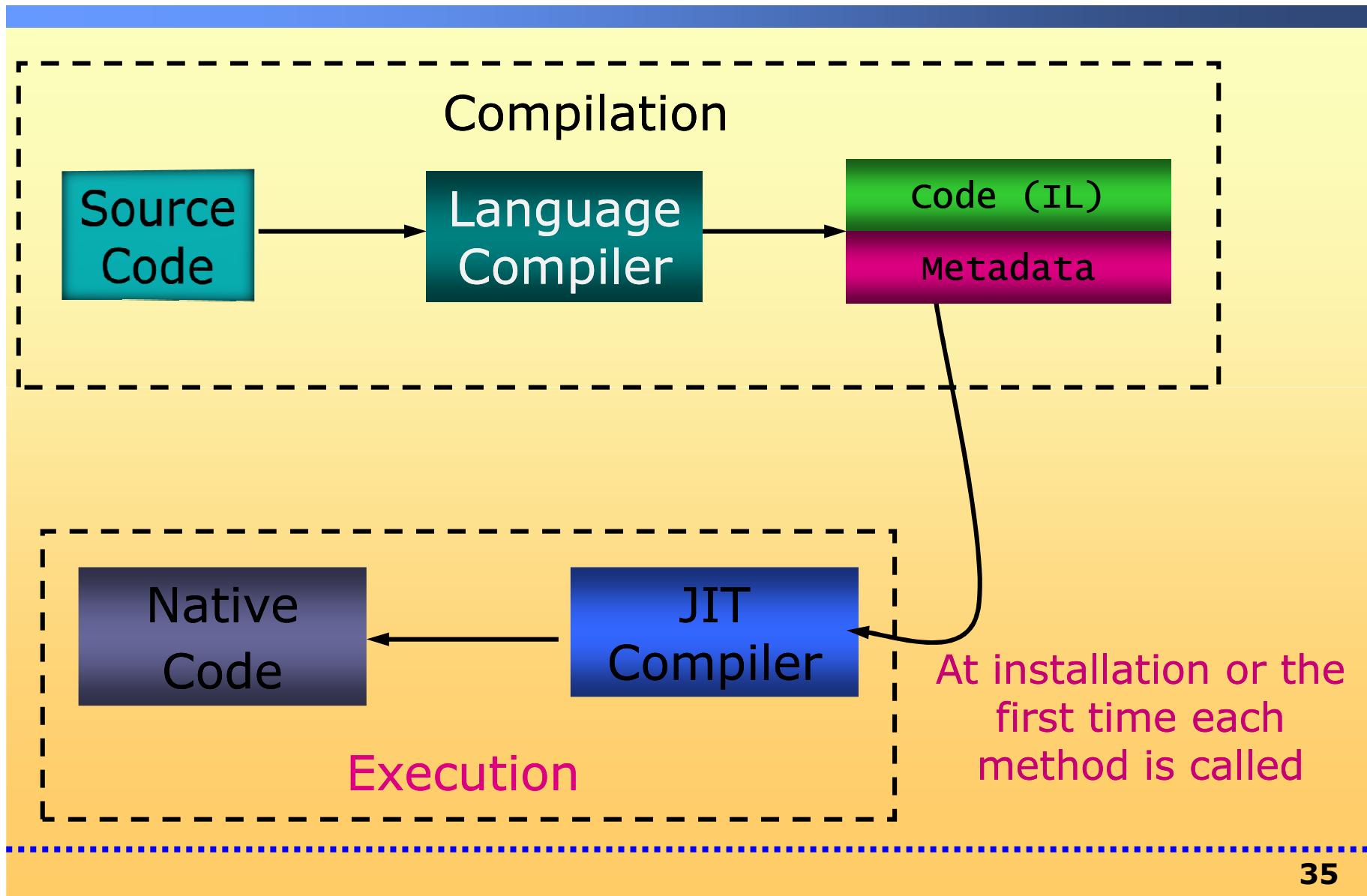
All Languages are compiled to IL code



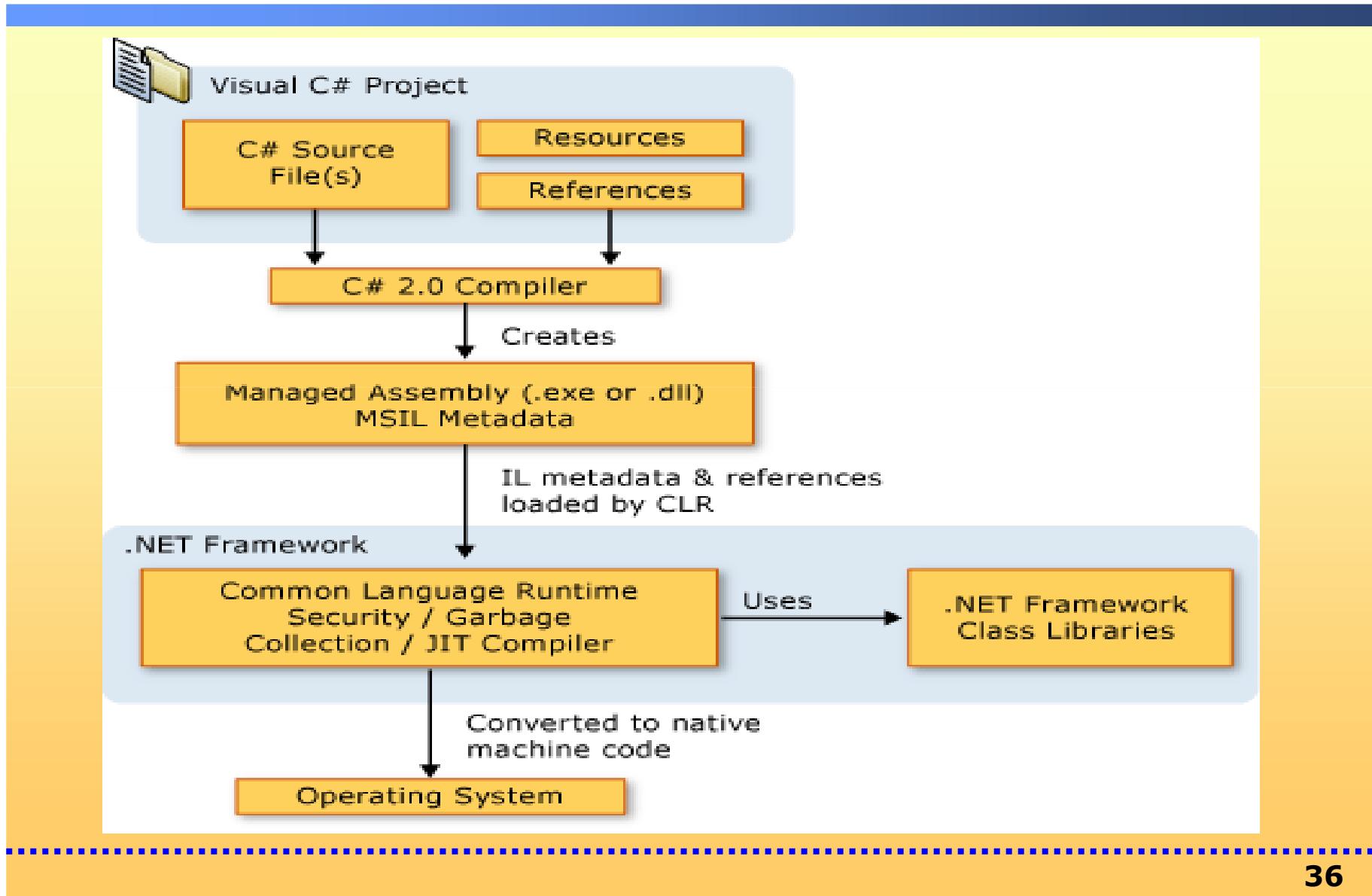
.Net Assemblies: *Compilation*



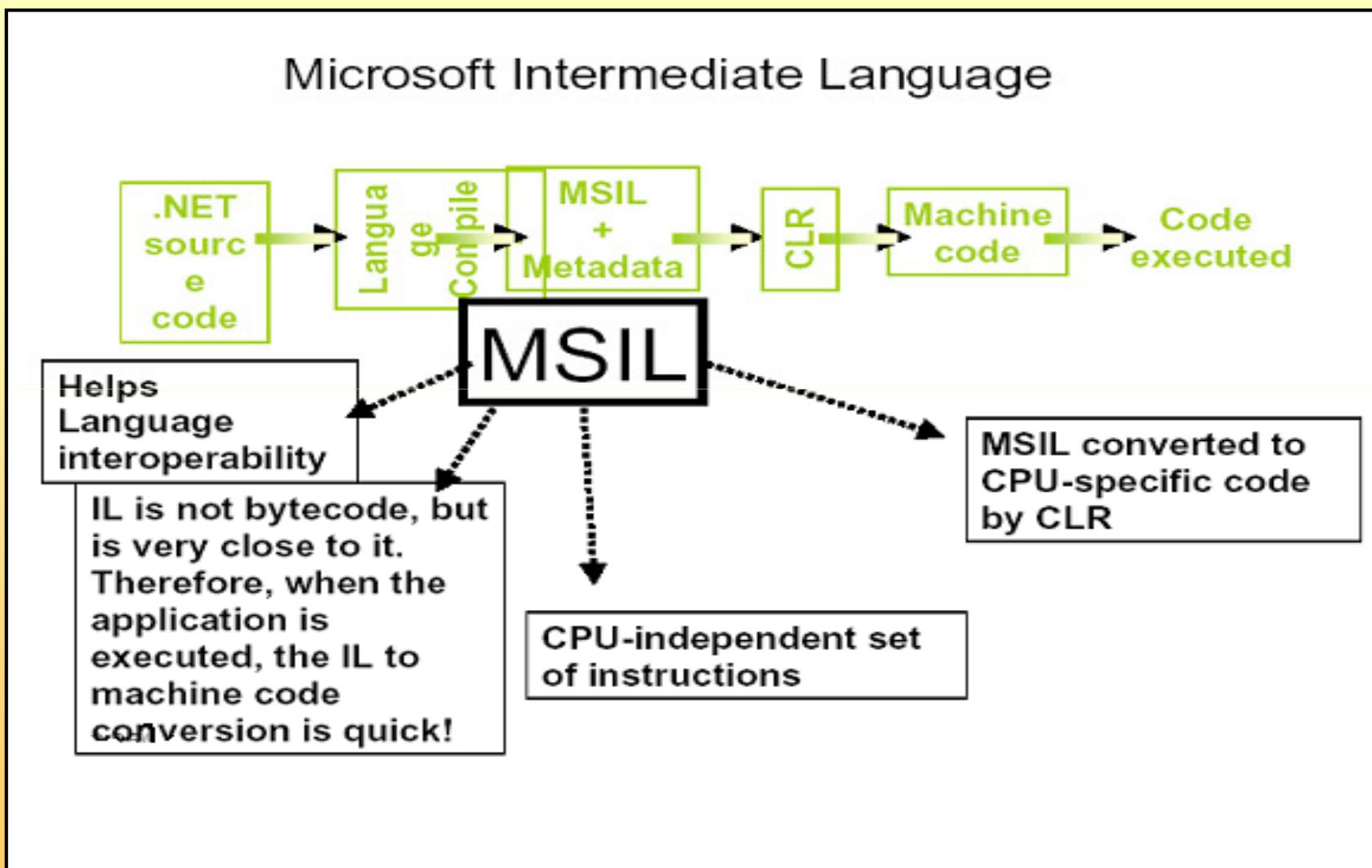
Compilation And Execution



.Net Execution Model



MSIL



Metadata and IL Code

Example: Hello World!



■ **Metadata:**

- data about data; description of assembly, description of types, attributes
- objects can communicate with each other
- Stored in binary format, CLR loads metadata into memory for reference

■ **IL code:**

- intermediate language, CPU independent sets of instructions

■ **Assembly:**

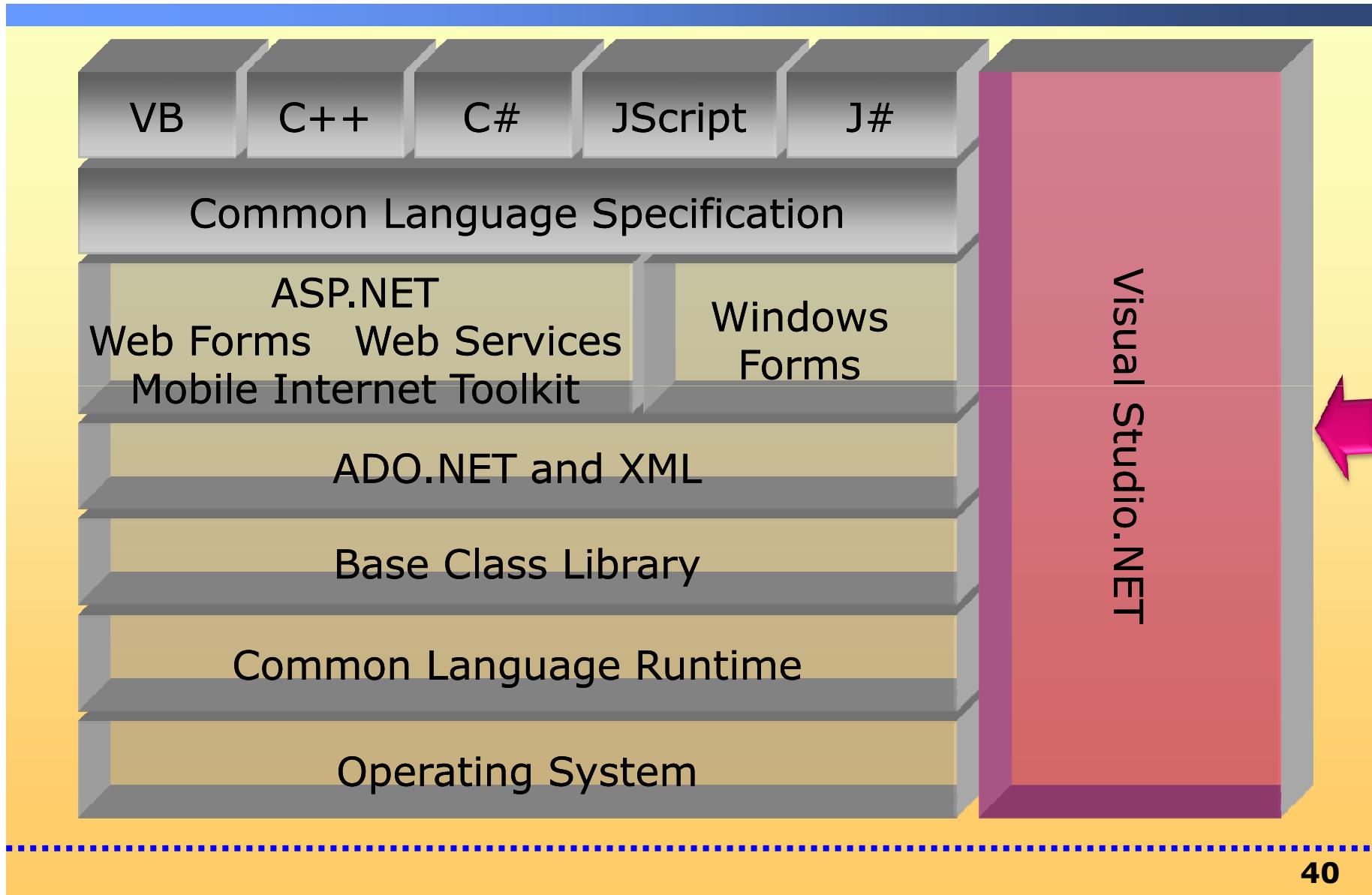
- a basic unit of versioning, deployment, security management, execute, reuse and sharing.

■ **Manifest:** Assembly Metadata

Portable Executable (PE)



Visual Studio.NET



Visual Studio .NET

- **Comprehensive best-of-breed tool for building connected applications based on .NET for Windows and the Web**
- **All languages under one roof**
- **All application types under one roof**
- **Consistent programming model for all application types**
- **Enterprise Lifecycle Support (VSS, Visio, Enterprise Templates)**

Developer Productivity Features

Increased IDE Performance

Startup time reduced

Improved IntelliSense®

Dynamic help faster

Object browser faster

Code editor drop-down menus faster

Enhanced "Add Web Reference" dialog

Code editor enhancements

New components

Debugging enhancements

Designer enhancements

Schema designer zoom

Mobile Web form designer

Integrated community search

Run multiple version of Visual Studio side-by-side:

Visual Studio 6.0

Visual Studio .NET 2002

Visual Studio .NET 2003

Upgrade from Visual Studio .NET 2002 to Visual Studio

Only the project files are updated

Doesn't change *any* of your source files

Visual Basic .NET

Fully object-oriented

Free threading, structured exception handling

Host VB6.0 controls in WinForms applications

Improved IntelliSense and debugger

Variable declaration in loops

Bitshift operators

Mobile applications

Microsoft® Visual C++® .NET

Managed extensions for C++

ISO C++ conformance

Windows Forms designer

Enhanced floating point performance

optimizations

P4 targeting

Microsoft® Visual C# ® .NET

Component-oriented, type-safe

Improved IntelliSense

Custom build steps

Code insertion for common tasks

Mobile applications Microsoft® Visual J#™ .NET

Java-language syntax

Full integration with Visual Studio .NET

Easy way to move Java-language skills,

Full access to the .NET Framework

Visual Studio.NET

■ Integrated Development Environment

- Visual Basic.NET
 - Many language enhancements
 - Inheritance, Overloading, Free Threading
- Visual C++
 - Integration with .NET Framework with managed extensions (classes)
- C#
 - New development language
 - Based on C/C++ with Garbage Collection/Memory Management

Developer Productivity

Productivity

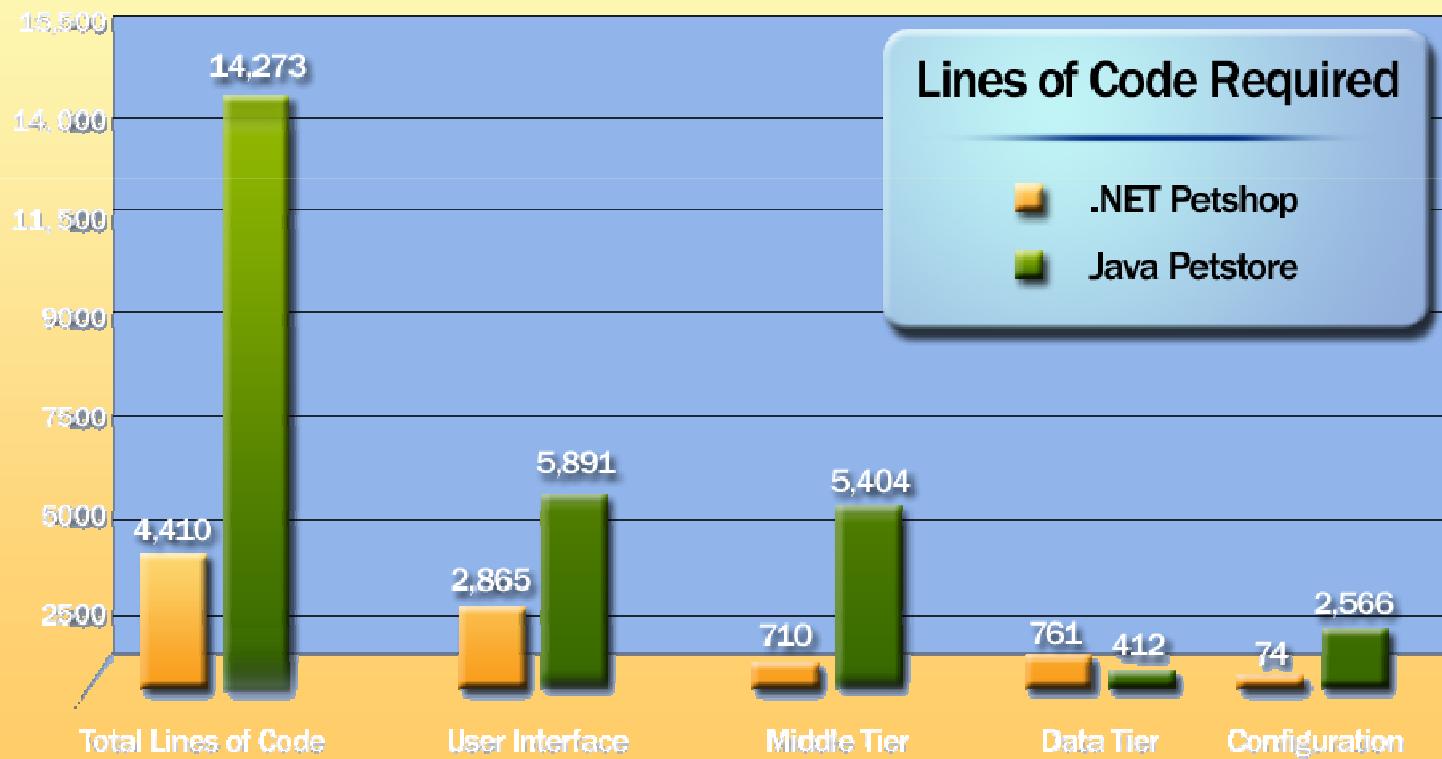
- One third the code

Performance

- 28 times faster

Scalability

- 6 times more users



ADO.NET

- **ADO.NET is a set of software components that can be used by applications to access data and data services.**
- **Its included as a port of .Net Framework.**
- **ADO.NET can access data from relational and non-relational sources.**
- **ADO.NET is considered as an evolution of ADO technology (ActiveX Data Objects)**
 - ADO.NET is extensively changed to support the new technology.

Architecture

- **ADO.NET Consists of two major parts**

- Data Provider
 - Provide access to the data sources
 - Each Datasource has its own set of provider objects
 - They share common set of utility classes
 - Connection
 - Command
 - Parameter
 - DataAdapter
 - DataReader
- DataSets
 - DataSet is group of classes describing a simple in-memory relational database
 - DataTable
 - DataView
 - DataColumn
 - DataRow
 - DataRelation
 - DataConstraint

ASP.NET

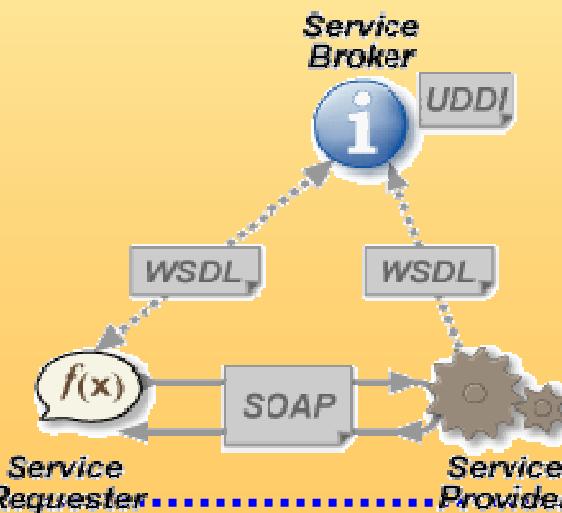
- **ASP.NET is a web application framework developed and marketed by Microsoft to allow programmers to build**
 - Dynamic web sites
 - Web applications
 - Web services.
- **It was first released in January 2002 with version 1.0 of the .NET Framework and is the successor to Microsoft's Active Server Pages (ASP) technology.**
- **ASP.NET is built on the Common Language Runtime (CLR), allowing programmers to write ASP.NET code using any supported .NET language.**

AJAX With ASP.NET

- **ASP.NET AJAX is a set of extensions to ASP.NET developed by Microsoft for implementing Ajax functionality.**
- **AJAX for ASP.NET was originally code named as ATLAS**
- **AJAX stands for Asynchronous JavaScript and XML.**
- **It's a group of interrelated web development techniques used on the client side to create interactive or rich web applications**
- **With AJAX, the web application can retrieve the data from the server without refreshing the page**

WebService

- **A Web Service is defined by the W3C as**
 - A software system designed to support interoperable machine-to-machine interaction over a network
 - It has an interface described in WSDL format.
 - Other systems interact with the web service using SOAP
 - These SOAP messages are conveyed over HTTP with XML Serialization
- **Web Service is also called as Internet API**



The .NET Framework Stack

.NET 3.5 Framework

AJAX

LINQ

PEER TO PEER
NETWORKING

WCF + WF
INTEGRATION

.NET 3.0 Framework

Workflow
Foundation (WF)

Communication
Foundation (WCF)

Presentation
Foundation (WPF)

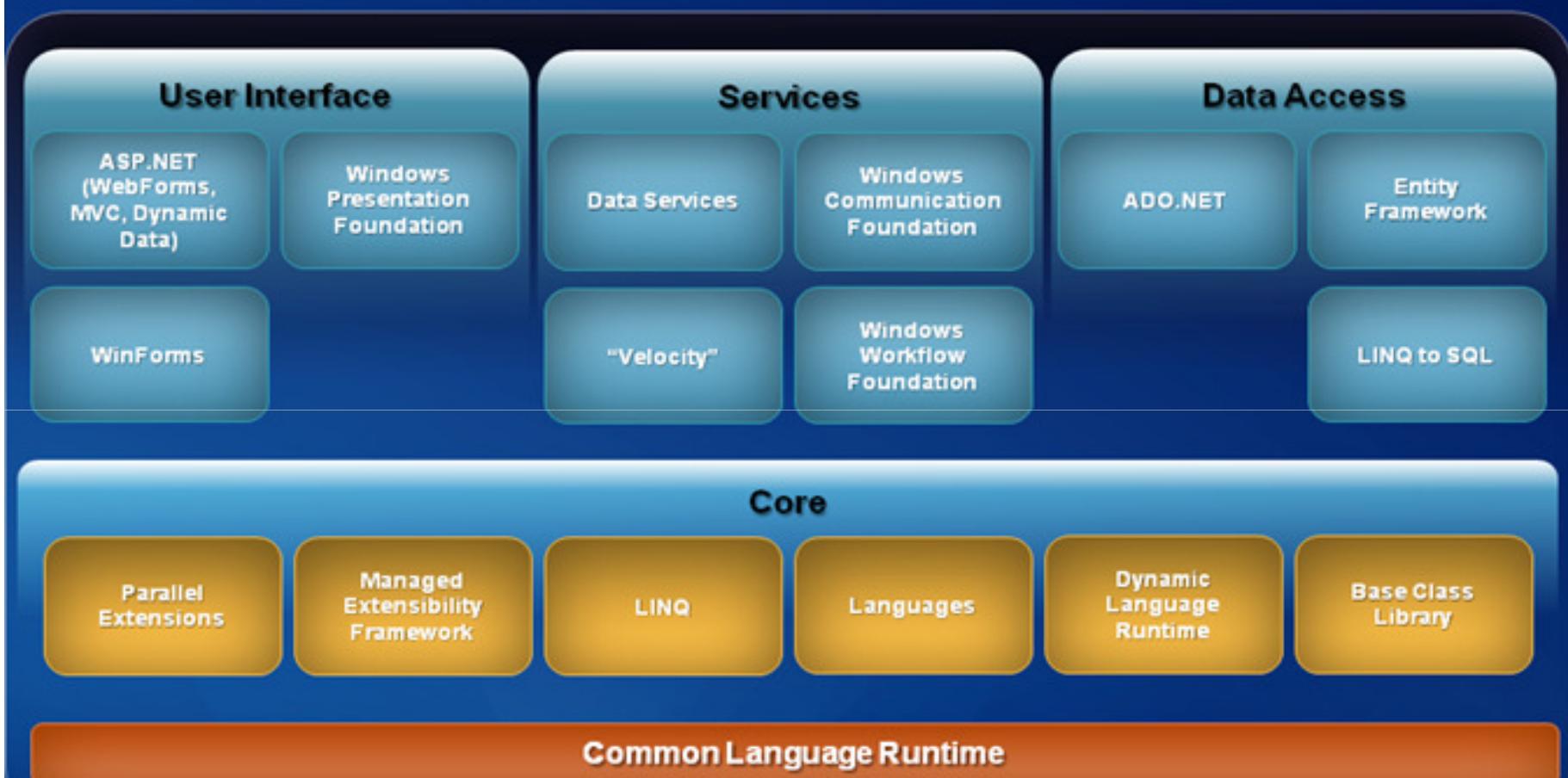
Windows
CardSpace

.NET 2.0 Framework

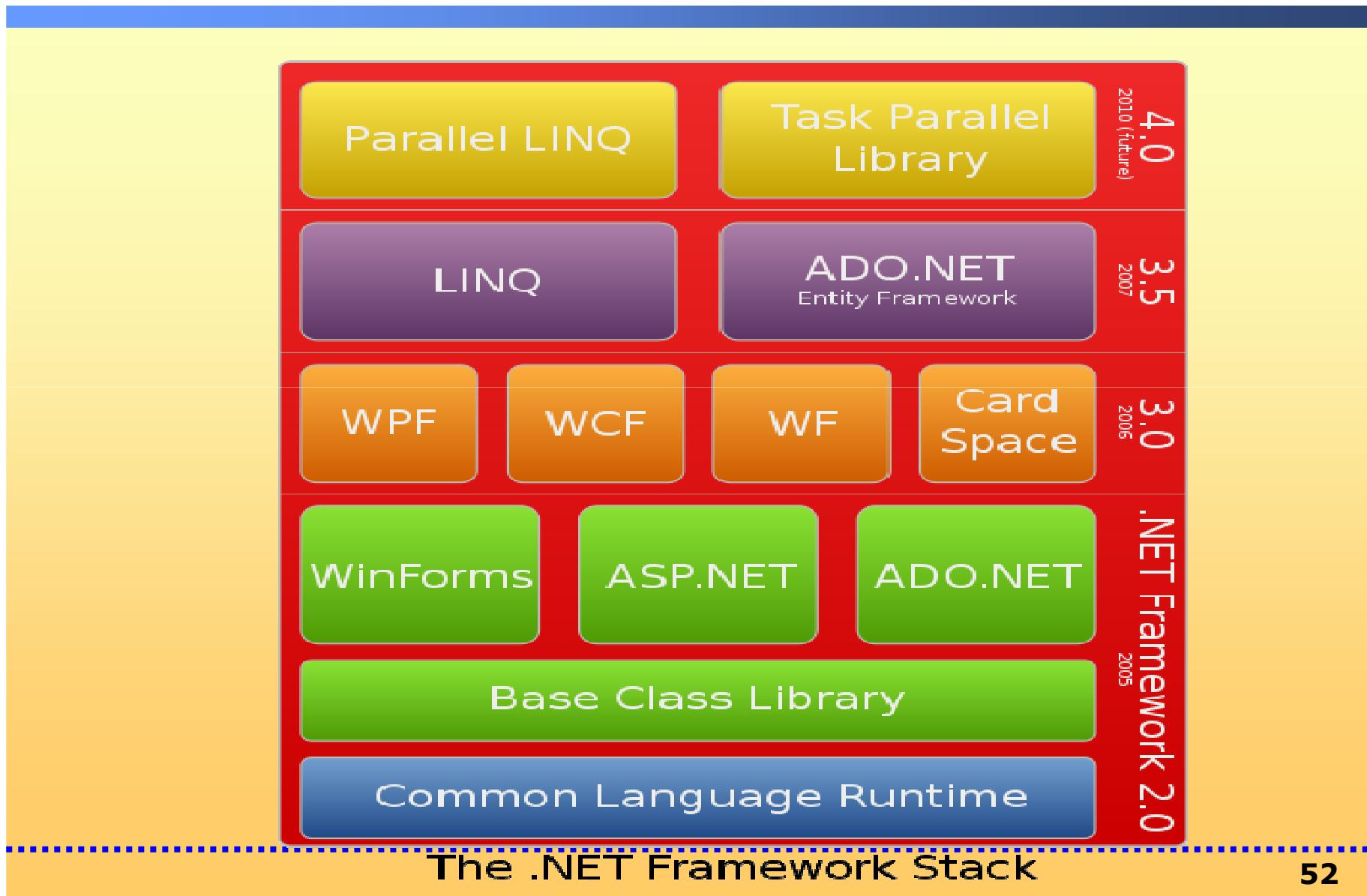
.NET Runtime

Base Class
Libraries (BCLs)

.NET Framework 4.0

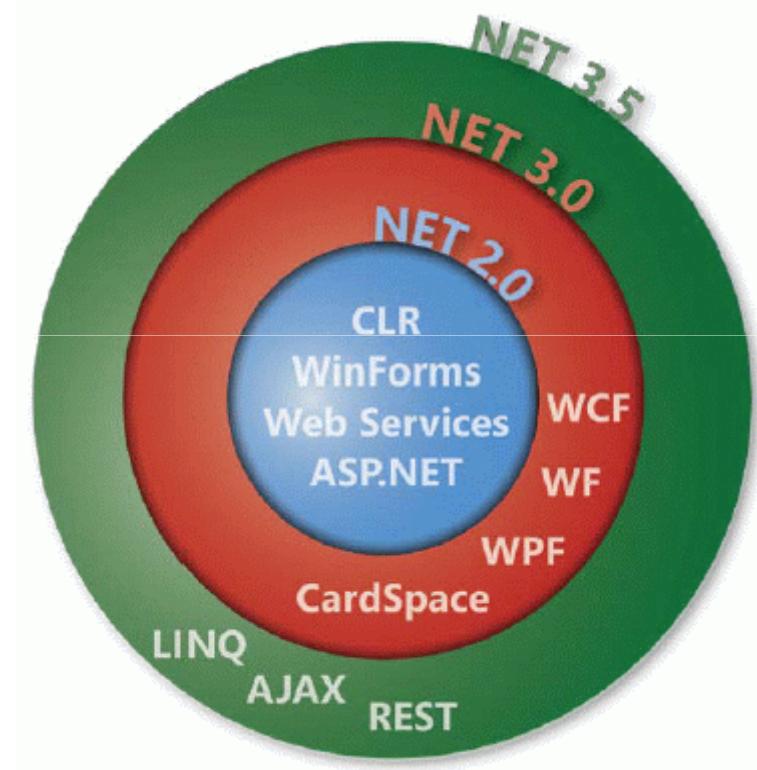


The .Net Framework Tech Stack



.Net Framework History

- **.NET Framework 1.0 - January 2002**
 - Initial release
- **.NET Framework 1.1 - April 2003**
 - Mobile ASP.NET Support
 - Security Enhancements
 - ODBC/Oracle Support
 - .NET Compact Framework
 - IPv6 Support
 - Various API Changes
- **.NET Framework 2.0 - November 2005**
 - Generics!
 - Hosting API
 - 64-bit Support
 - ASP.NET enhancements(web controls, personalisation)
 - Data Controls with Declarative Data Binding
 - .NET Micro Framework
 - Various API Changes
- **.NET Framework 3.0 - November 2006**
 - Windows Presentation Framework
 - Windows Communication Framework
 - Windows Workflow Foundation
 - Windows CardSpace
- **.NET Framework 3.5 – November 2007**
 - ASP.NET AJAX
 - LINQ
 - Various API Changes
- **.NET Framework 4.0 – April 2010**
 - MEF, TPL, PLINQ, DLR, MVC, ADO.NET EF



Additive versions of the .NET Framework

Summary

■ The .NET Framework

- Dramatically simplifies development and deployment
- Unifies programming models
- Provides robust and secure execution environment
- Supports multiple programming languages

Introduction to C#



Need for C#

- **Existing languages are powerful. Why do we need another language?**
- **Important features are spread out over multiple languages**
 - Example: must choose between pointers (C++) or garbage collection (Java)?
- **Old languages + new features = poor syntax**
 - Garbage collection in C++?
 - Event-driven GUIs in Java?

Goals of C#

- **Give developers a single language with**
 - A full suite of powerful features
 - A consistent and simple syntax
- **Increase developer productivity!**
 - Type safety
 - Garbage collection
 - Exceptions
 - Leverage existing skills

Goals of C# cont.

- **Support component-oriented programming**
 - First class support for component concepts such as properties, events, attributes
- **Provide unified and extensible type system**
 - Where everything can be treated as an object
- **Build foundation for future innovation**
 - Concise language specification
 - Standardization

Architectural History of C#

■ Anders Hejlsberg.

- C#'s principal designer and lead architect at Microsoft.
- He designed
 - Visual J++
 - Borland Delphi
 - Turbo Pascal
 - .NET CLR (Core)
 - C# Language



Anders Hejlsberg,
Microsoft Technical
Fellow and Chief
Architect, C#

Design of C#

- **Derived from the features and syntaxes of other languages**
 - The safety of **Java**
 - The ease of **Visual Basic**
 - The power of **C++**
- **Uses the .NET Framework**
- **Plus several unique features**

The safety of Java

- 100% object oriented
- Automatic garbage collection
- Array bounds checking at runtime
- Strict type checking
- Structured exception handling

The ease of Visual Basic

- First class support for properties
- First class support for events
- For-each loops

The power of C++

- **Enumerations**
- **Operator overloading**
 - Mathematical, Indexing, and Casting
- **Function pointers**
 - Called “delegates”
 - Type safe
- **Structs**
- **Option to pass parameters by reference or by value**
- **Can disable type-safety, garbage collection, and bounds checking**
- **Can directly manipulate memory with pointers (unsafe)**

C# Program Structure

- **Namespaces**
 - Contain types and other namespaces
- **Type declarations**
 - Classes, structs, interfaces, enums, and delegates
- **Members**
 - Constants, fields, methods, properties, indexers, events, operators, constructors, destructors
- **Organization**
 - No header files, code written “in-line”
 - No declaration order dependence

General Structure of C# Program

```
1 // A skeleton of a C# program
2 using System;
3 namespace YourNamespace
4 {
5     class YourClass...
6     struct YourStruct...
7     interface IYourInterface...
8     delegate int YourDelegate();
9     enum YourEnum...
10    namespace YourNestedNamespace...
11    class YourMainClass...
12 }
```