Introduction to .NET Framework

Authored by : Sushant Banerjee
Email : sushantba@cybage.com

Presented by : Sushant Banerjee
Extn. 7210

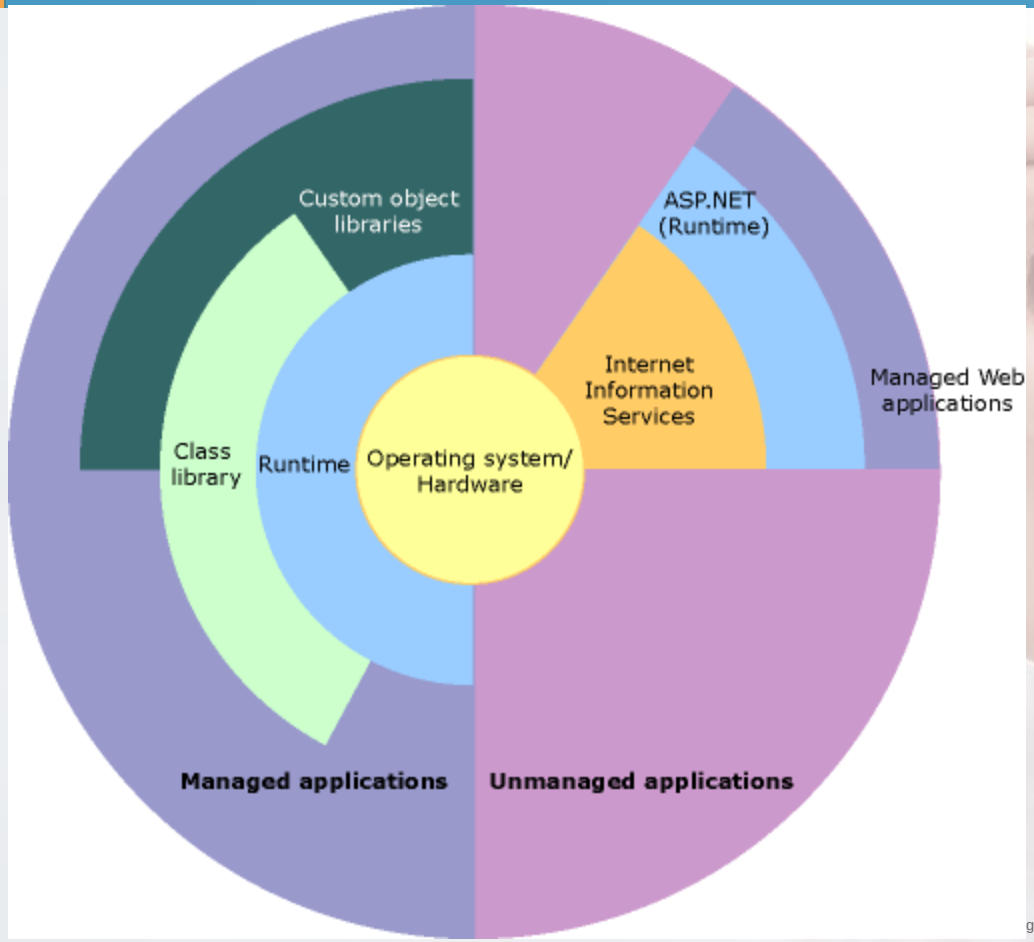
Agenda

- Introduction to .NET Framework
- CLR
- CTS
- CLS
- Assembly
- Garbage Collection
- .NET Framework Class Library

What is .NET Framework

- The .NET Framework is a technology using which you can build and run almost all types of applications.
- The .NET Framework consists of
  - Common Language Runtime (CLR)
  - .NET Framework Class Library

Overview of .NET Framework

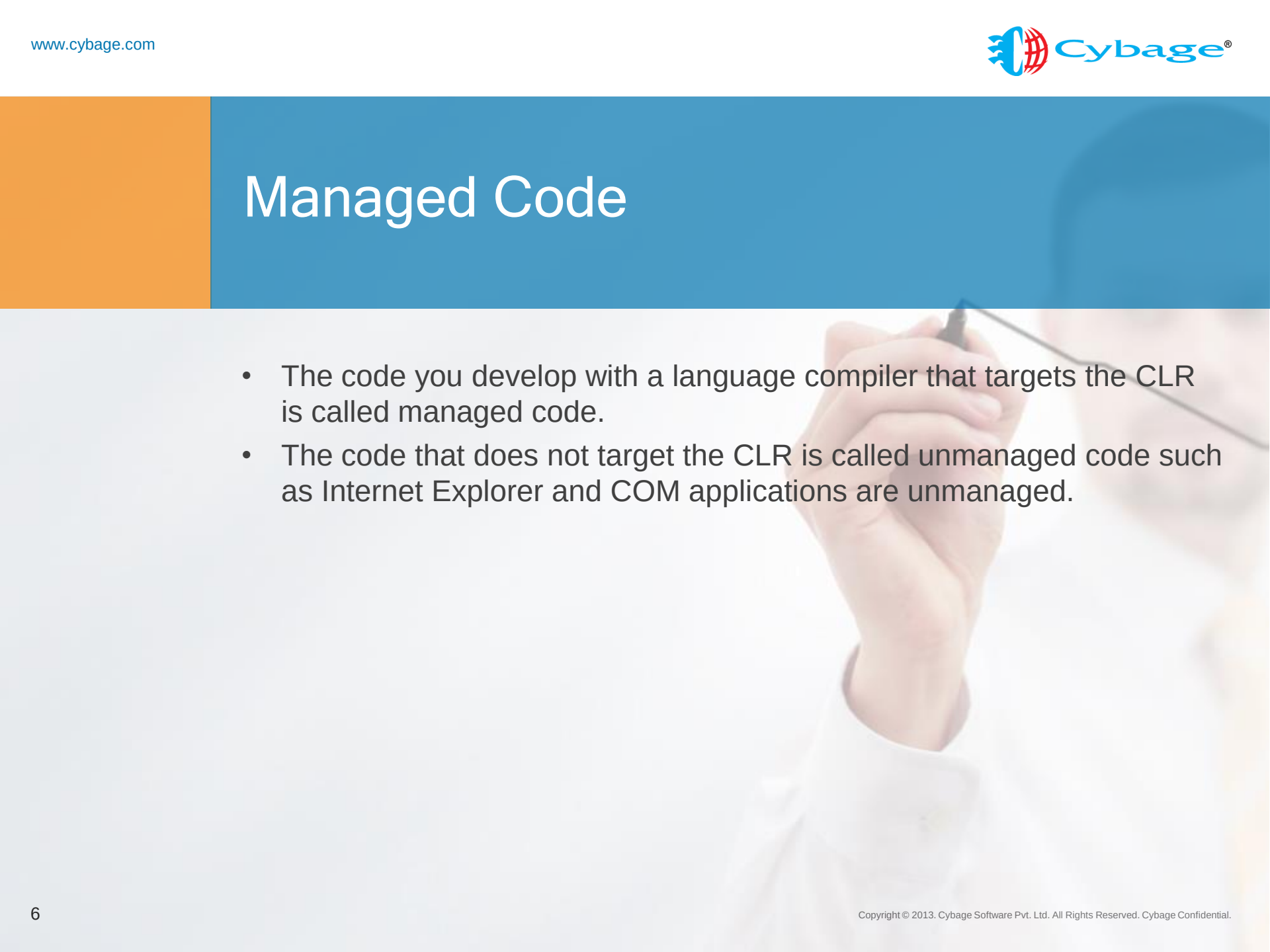


Features of .NET Framework

- Object Oriented Programming Environment
- Minimizes Software Deployment
- Minimizes Version Conflicts
- Promotes Safe Execution of Code
- Eliminates Performance Problems
- Consistent Development Experience
- Industry Standard

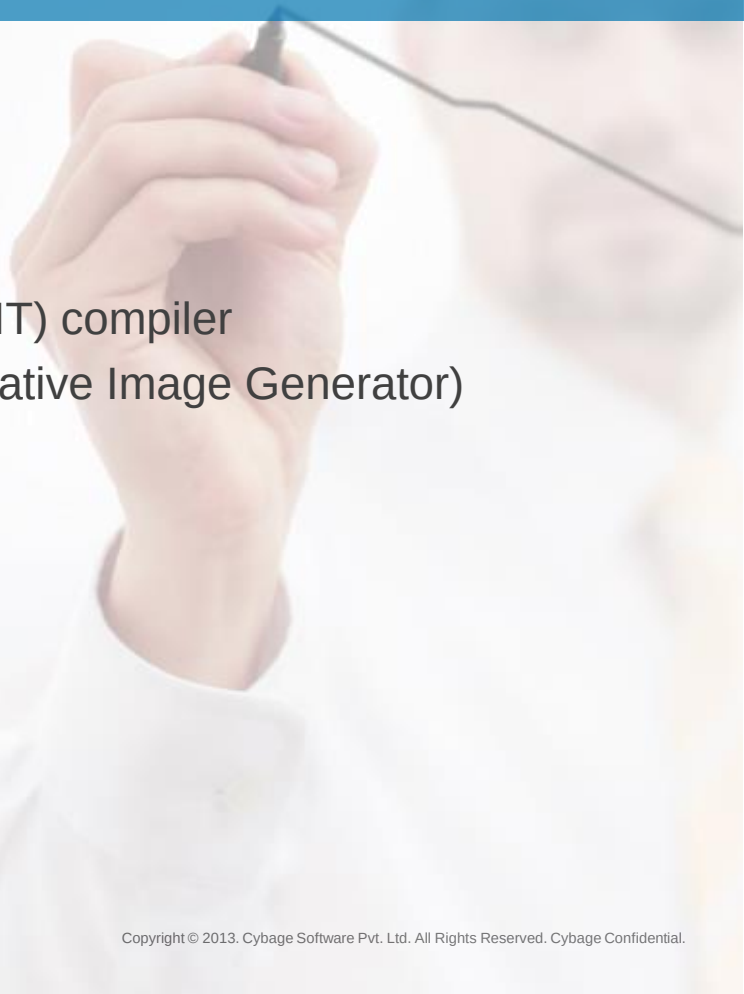
Managed Code

- The code you develop with a language compiler that targets the CLR is called managed code.
- The code that does not target the CLR is called unmanaged code such as Internet Explorer and COM applications are unmanaged.



Managed Execution Process

- Choosing a compiler
- Compiling your code to MSIL
- Compiling MSIL to Native Code
  - A .NET Framework Just-In-Time (JIT) compiler
  - The .NET Framework Ngen.exe (Native Image Generator)
- Running Code



Native Image Generator

- The Ngen.exe converts MSIL assemblies to native code much like the JIT compiler does.
- However the operations of Ngen.exe differs from that of the JIT compiler in three ways :
  - It converts MSIL to native code before running the application instead of while the application is running.
  - It compiles an entire assembly at a time, instead of one method at a time.
  - It persists the generated code in the native image cache as a file on disk.

Common Language Runtime

- CLR or Common Language Runtime is also known as runtime.
- The CLR is the foundation of the .NET Framework.
- The CLR is responsible for providing core services such as
  - Memory Management
  - Thread Management
  - Code Compilation and Management
  - Remoting
  - Enforcing Type Safety
  - Code Security
  - Structured Exception Handling
  - Cross-language Integration

CTS

- Common Type System (CTS) defines how types are declared, used and managed in the Common Language Runtime.
- CTS ensures Cross Language Integration.
- CTS defines a set of rules that languages must follow, which ensures that objects of different languages can interact with each other.
- Common Type System implements Strong Typing.
- The various Microsoft and third party language compilers generate managed code that conforms to CTS.
- This means that the managed code can consume other managed types and instances while strictly enforcing type safety.

CTS (contd…)

- CTS provides library that contains the primitive data types such as Boolean, Byte, char, Int32 and Int64.
- The Common Type System in the .NET Framework supports the following types:
  - Classes
  - Structures
  - Enumerations
  - Interfaces
  - Delegates

CLS

- Common Language Specification (CLS) is a set of basic language features.
- All the rules that apply to the CTS apply to the CLS as well.
- CLS ensures language interoperability by defining a set of features available in a wide variety of languages.
- If your code uses CLS features, it is guaranteed to be accessible from any programming language that supports CLS.

CLS-Compliant Code

- When your code follows all CLS rules and restrictions, it is called CLS-compliant code.
- CLS-Compliant development tools include compilers, such as C# compiler or VB.NET compiler.
- You can mark your assembly or part of code using the attribute name "**CLSCompliantAttribute**" which can be set to either **true** or **false**.
- When your code is not CLS-compliant the compiler generates warning.

Writing CLS-compliant Code

```
//Mark the assembly as CLS-compliant
[assembly: CLSCompliant(true)]
namespace MyNamespace
{
    //Mark the class as CLS-compliant
    [CLSCompliant(true)]
    public class Program
    {
```

Writing CLS-compliant Code

```
[CLSCompliant(true)]
public void MyMethod(uint x)
{
  //generates compile time warning since
  //uint is not a CLS-compliant
  //set to false to mark the method is not CLS-
  compliant
  //to remove compile time warning
}
```

Assembly

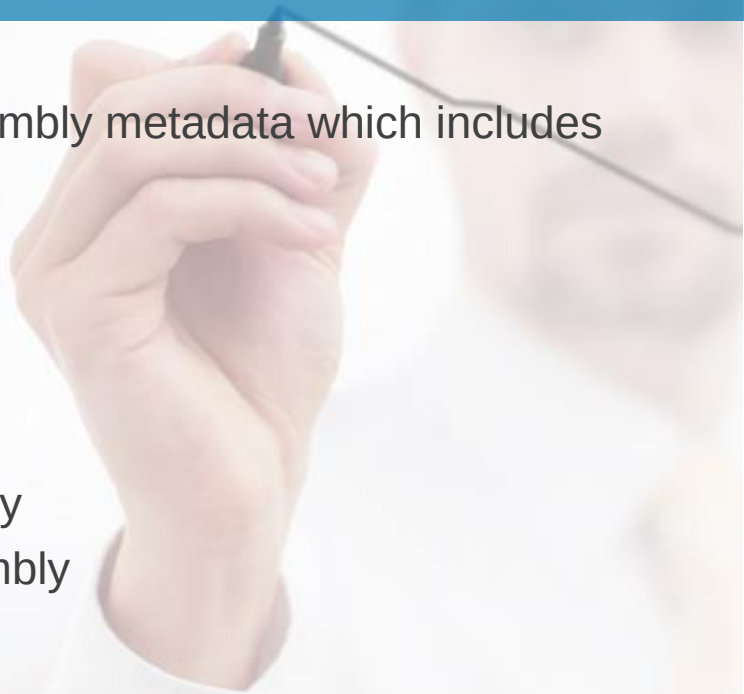
- Assembly Manifest
- Type Metadata
- MSIL Code
- Resources

Portable Executable

- A Portable Executable (PE) file contains
  - Metadata
  - MSIL Code

Assembly Manifest

- The Assembly Manifest contains assembly metadata which includes following information
  - Assembly Name
  - Version Number
  - Culture Information
  - Strong Name Information
  - List of all the files in the assembly
  - Information on referenced assembly



Metadata

- The .NET Framework enables language compilers to generate metadata and insert into PE file along with MSIL code.
- This metadata describes every type and member defined in your code in a language neutral manner.
- When your code executed, runtime loads metadata into memory and references it to discover information about your code's classes, members, inheritance and so on.
- Metadata enables language interoperability.

Automatic Memory Management

- Garbage Collector
  - Memory Allocation
  - Releasing Memory
- Generations of Memory
  - Generation 0
  - Generation 1
  - Generation 2
- Strong Reference
- Weak Reference

.NET Framework Class Library

- The .NET Framework class library is also called Class Library.
- The class library is wide-ranging, object oriented collection of reusable types.
- The class library helps you developing wide variety of application ranging from Console, Windows to Web application.
- The System namespace is root namespace for fundamental type in .NET framework
- The System namespace contains over 100 classes that deal with core runtime concepts and exception handling.

Namespaces in Class Library

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading;
using System.IO;
using System.Data;
using System.Globalization;
```

Bibliography, Important Links

- C# 5.0 in a Nutshell – Published by O'Reilly
- www.msdn.com – Library
- http://en.wikipedia.org

Any Questions?

Thank you!