MSIL & CLI

Kun-Yuan Shieh

Outline

- **.NET?**
- **.NET framework**
- **Inside the CLR execution engine**
- **Introduction to MSIL**
  - Part I
  - Part II
- **Tools**

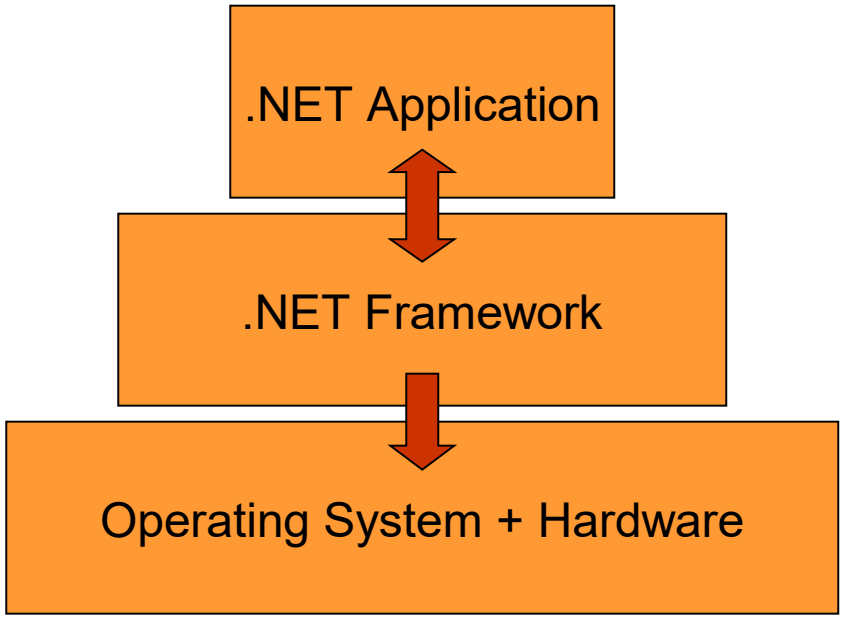

.Net?

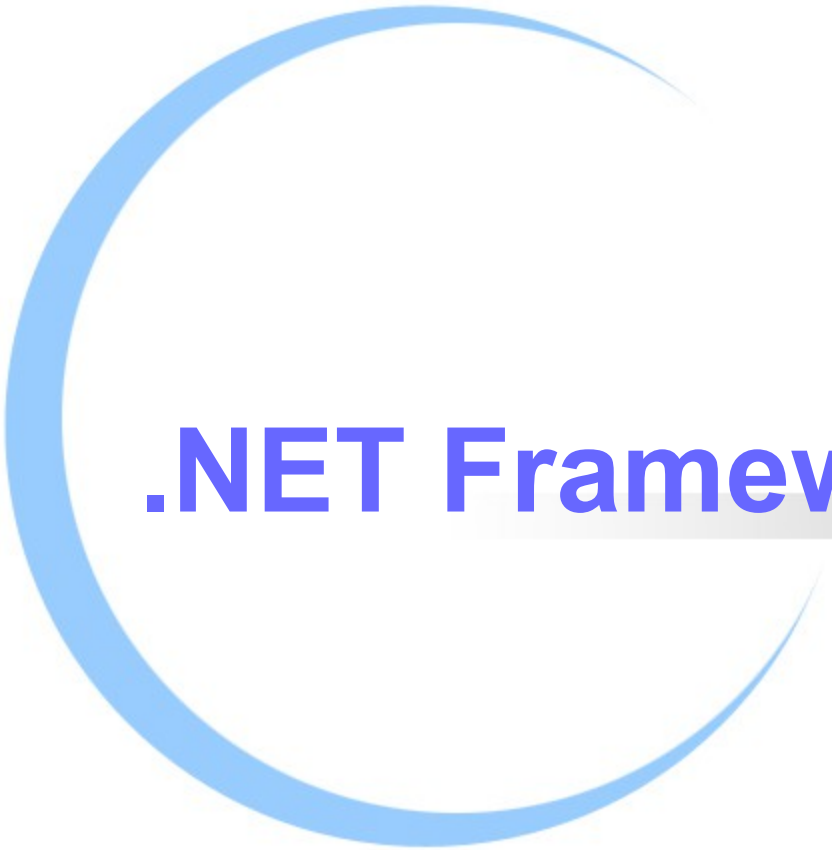
What is .NET?

- **Software platform**
- **Language neutral**
- **Accessible over many devices and operating systems**
- **Component Based**
- **Microsoft's Vision of the future**

What is .NET?

- **.NET is a new framework for developing *web-based and windows-based applications* within the Microsoft environment.**





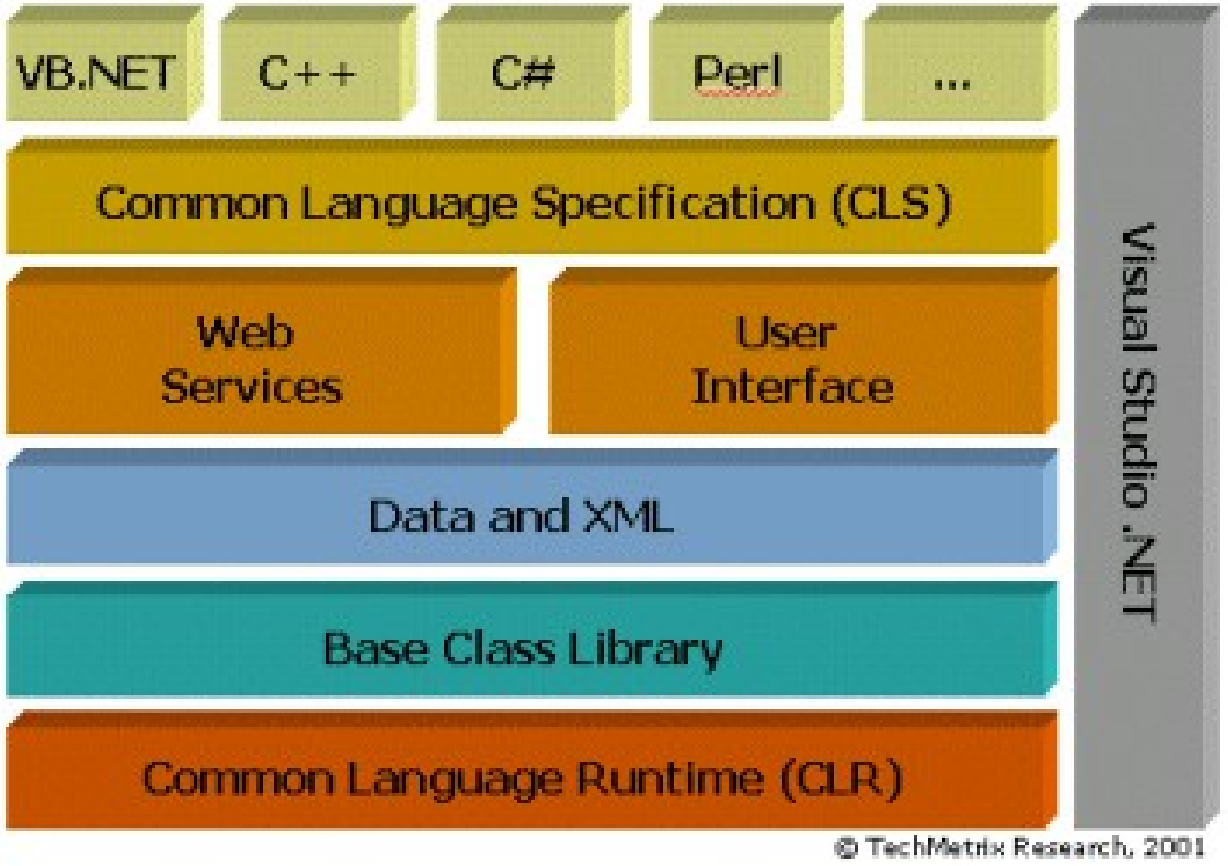
.NET Framework

What is the .NET framework

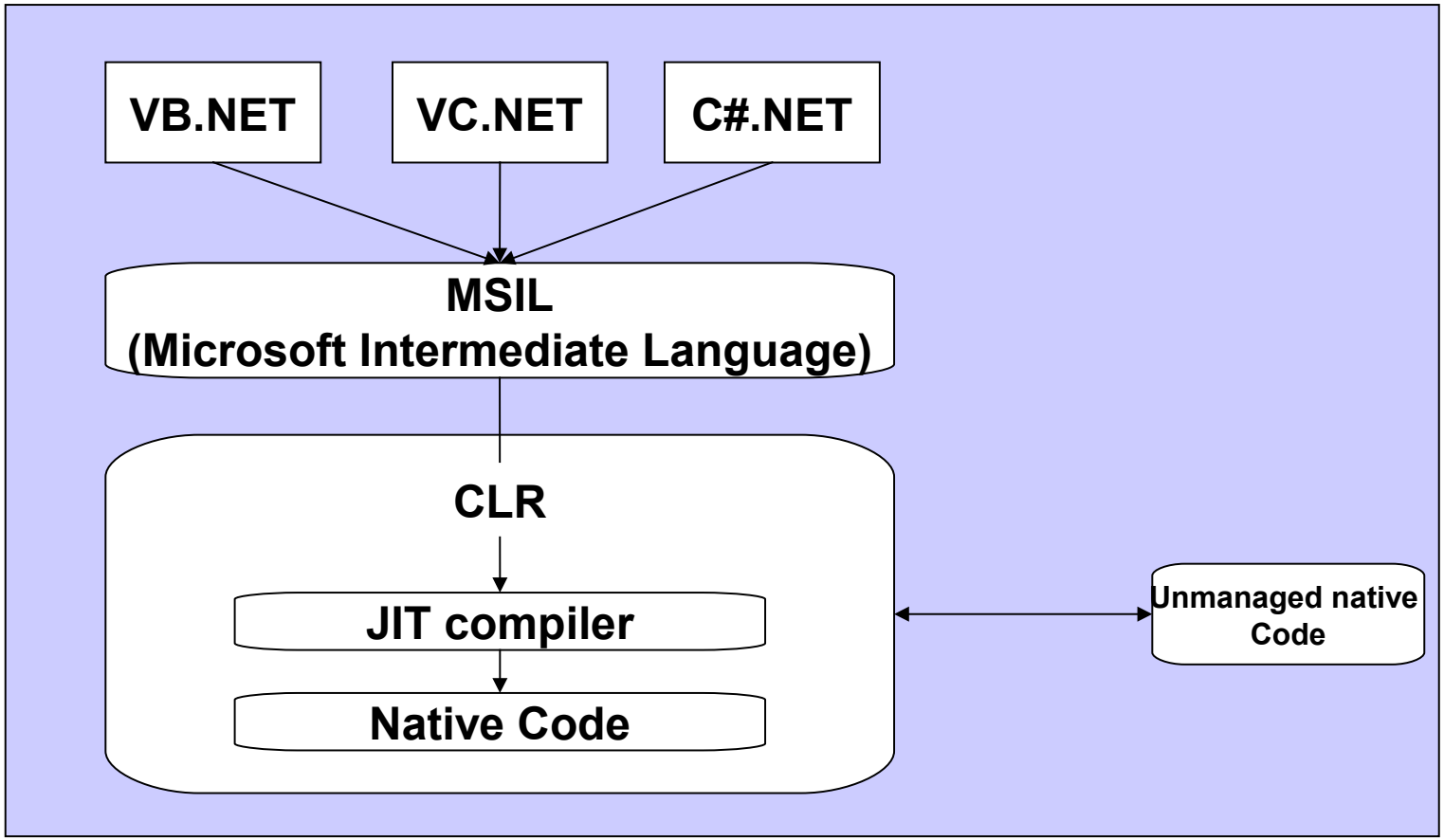
- **Infrastructure for the overall .NET Platform.**
- **Major Components :**
  - Common Language Runtime (CLR)
  - Base Class Library
  - Common Type System (CTS)
  - Common Language Specification (CLS)

.NET Framework structure

(http://www.dotnet101.com/articles/art014_dotnet.asp)



Compiling process in .NET



Compiling process in .NET

- **Compiled into the Intermediate Language (IL), Not directly compiled into machine code**
- **Metadata accompanies the IL, it describes the contents of the file (e.g.. parameters, methods…)**
- **The Manifest describes what other components the Intermediate Language (IL) executable needs**

Common Language Runtime (CLR)

- **CLR works like a virtual machine in executing all languages**
- **Checking and enforcing security restrictions on the running code**
- **Manages memory through an extremely efficient garbage collector**
- **Common Type System (CTS)**
- **Conversion from IL into code native to the platform being executed on**

Common Language Runtime (CLR)

- **All .NET languages must obey the rules and standards imposed by CLR. Examples:**
  - Object declaration, creation and use
  - Data types, language libraries
  - Error and exception handling
  - Interactive Development Environment (IDE)

Common Type System (CTS)

- **CTS is a rich type system built into the CLR**
  - Implements various types (int, double, etc)
  - And operations on those types
- **Strictly enforces type safety**
- **Ensures that classes are compatible with each other by describing them in a common way**
- **Enables types in one language to interoperate with types in another language**

Common Language Specification

- **CLS is a set of specifications that language and library designers need to follow**
  - This will ensure interoperability between languages
- **Specification that a language must conform to, to be accepted into the .NET framework**
- **The specification are detailed at http://msdn.microsoft.com/net/ecma/**

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.

Some CLI implementation

- **CLR – Microsoft's commercial offering**
- **SSCLI (code-named "Rotor") – Microsoft's Shared Source CLI**
- **Mono - open source project initiative sponsored by Ximian (now a part of Novell)**
- **Portable .NET – by Southern Storm Software, Pty Ltd, an Australian company**
- **OCL – portions of the implementation of the CLI by Intel**

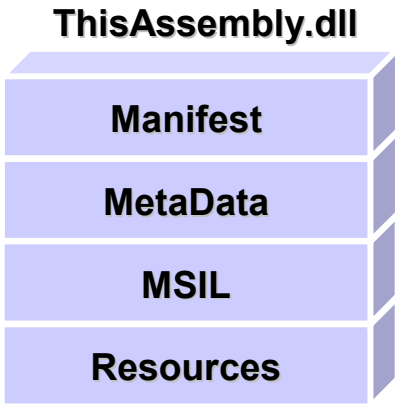
Inside the CLR Execution engine

Assemblies

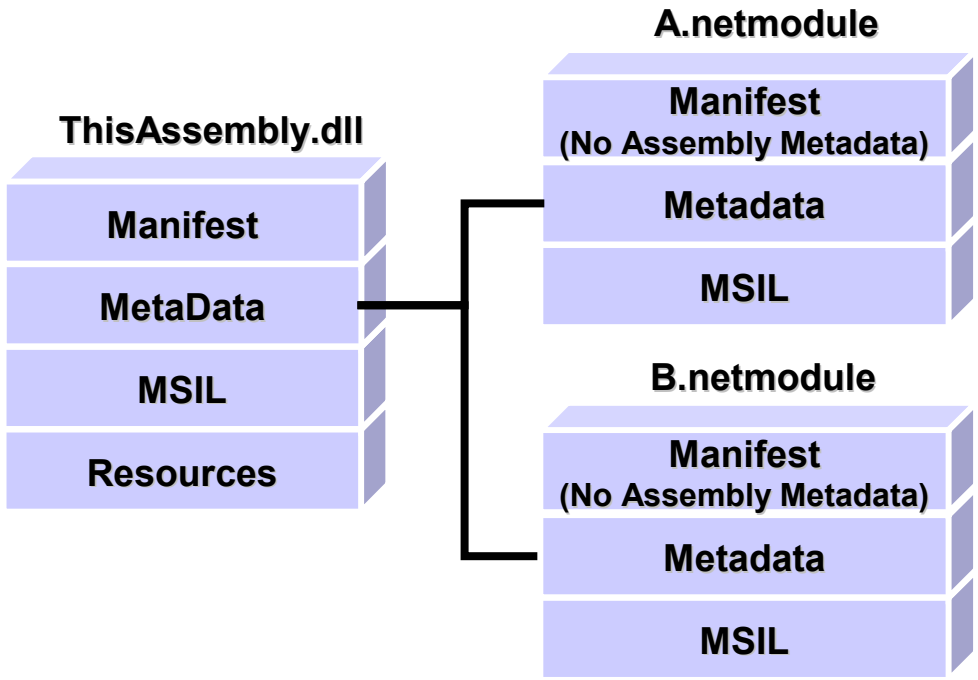
- **Assemblies are the smallest unit of code distribution, deployment and versioning**
- **Individual components are packaged into units called *assemblies***
- **Can be *dynamically loaded into the execution engine on demand* either from local disk, across network, or even created on-the-fly under program control**

Single file and multi file assembly

Single File Assembly



Multi File Assembly



Assembly characteristics

- **Self-describing**
  - To enable data-driven execution
- **Platform-independent**
- **Bounded by name**
  - Locate assemblies by querying four-part tuple that consists of a human-friendly name, an international culture, a multipart version number, and a public key token.
- **Assembly loading is sensitive to version and policy**
  - Assemblies are loaded using tunable biding rules, which aloow programmers and administrators to contribute policy to assembly-loading behavior.
- **Validated**
  - Each time an assembly is loaded, it is subjected to a series of checks to ensure the assembly's integrity.

Self-describing

- **Modules:**
  - Blueprint for types in the form of metadata and CIL
  - Single file containing the structure and behavior for some or all he types and/or resources found in the assembly
- **An assembly always contains at least one module but has the capacity to include more**
- **Assemblies themselves have metadata that describe their structure: *manifest***

Manifest

- **Compound name for the assembly**
- **Describe the public types that the assembly exports**
- **Describe types that the assembly will import from other assemblies**

```
.assembly extern mscorlib
{
  .ver 0:0:0:0
}
.assembly HelloWorld
{
  .ver 0:0:0:0
}
.module HelloWorld.exe
// MVID: {D662624F-D333-48B7-
    ACB2-E06B4F75DC3D}
.imagebase 0x00400000
.subsystem 0x00000003
.file alignment 512
.corflags 0x00000001
// Image base: 0x06d20000
```

Take a look at metadata

```
ScopeName : HelloWorld.exe
MVID      : {D662624F-D333-48B7-ACB2-E06B4F75DC3D}
=============================================================
Global functions
-------------------------------------------------------

Global fields
-------------------------------------------------------

Global MemberRefs
-------------------------------------------------------

TypeDef #1
-------------------------------------------------------
        TypDefName: Hello.World.Hello  (02000002)
        Flags     : [Public] [AutoLayout] [Class] [AnsiClass]  (00000001)
        Extends   : 01000001 [TypeRef] System.Object
        Method #1 [ENTRYPOINT]
        -------------------------------------------------------
                MethodName: hello (06000001)
                Flags     : [Public] [Static] [ReuseSlot]  (00000016)
                RVA       : 0x00002050
                ImplFlags : [IL] [Managed]  (00000000)
                CallCnvntn: [DEFAULT]
                ReturnType: Void
                No arguments.
```

Tools

Tools

- **Download & install from Microsoft®**
  - .NET Framework Redistributable Package version 1.1
  - .NET Framework SDK Version 1.1
- **Tools**
  - C# compiler: Csc.exe
  - IL assembler: ilasm.exe
  - IL diassembler: ildasm.exe

Path

- **csc.exe**

  c:\WIDOWS\Microsoft.NET\Framework\v1.1.4322

  (for MS. CLR)

- **ilasm.exe**

  C:\WINDOWS\Microsoft.NET\Framework\v1.1.4322

  (for MS. CLR)

- **ildasm.exe**

  C:\Program Files\Microsoft.NET\SDK\v1.1\Bin

  (for MS. CLR)

If you have
Microsoft Visual Studio .NET 2003

- **csc.exe**

  c:\WIDOWS\Microsoft.NET\Framework\v1.1.4322

  (for MS. CLR)

- **ilasm.exe**

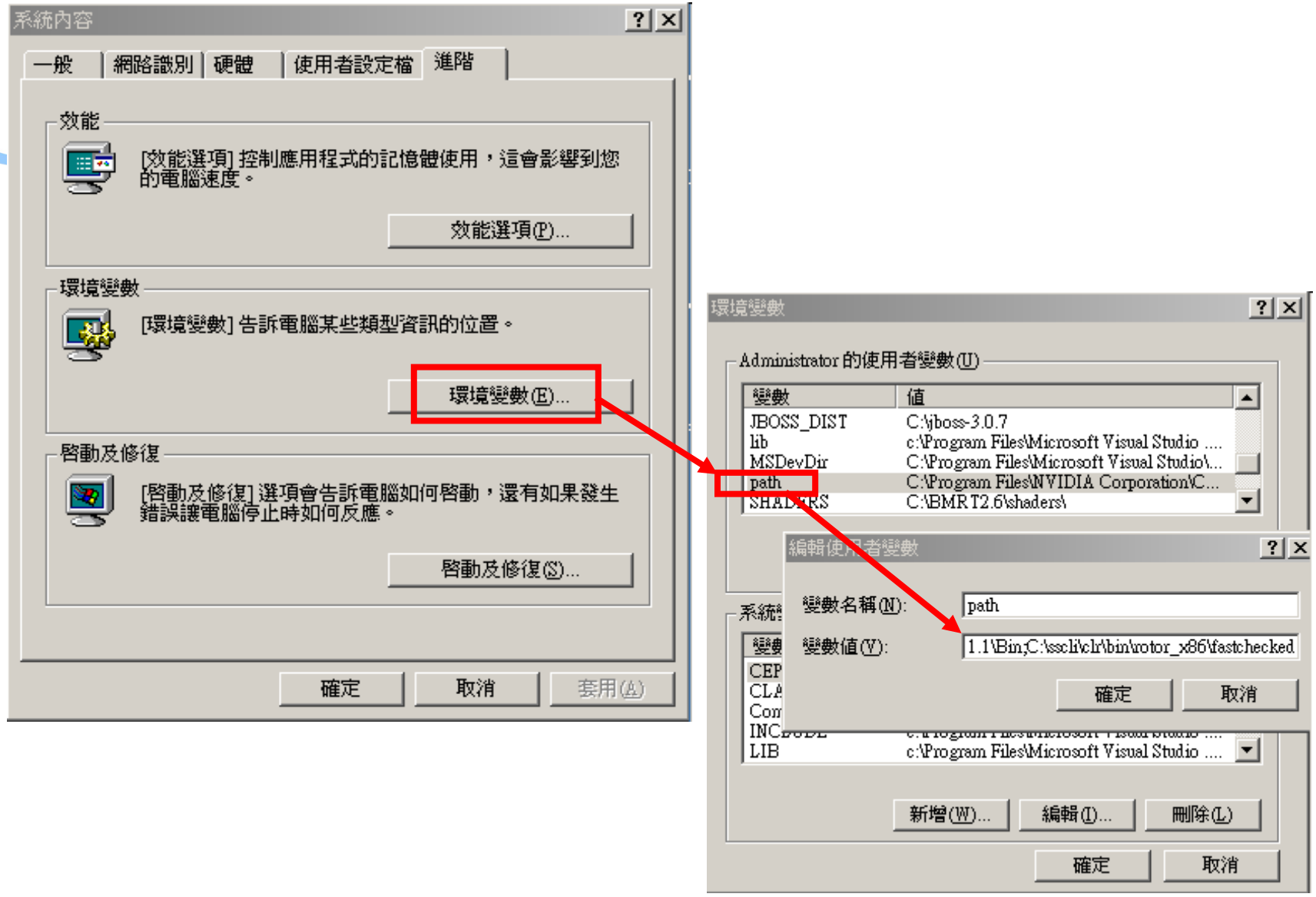
  C:\WINDOWS\Microsoft.NET\Framework\v1.1.4322

  (for MS. CLR)

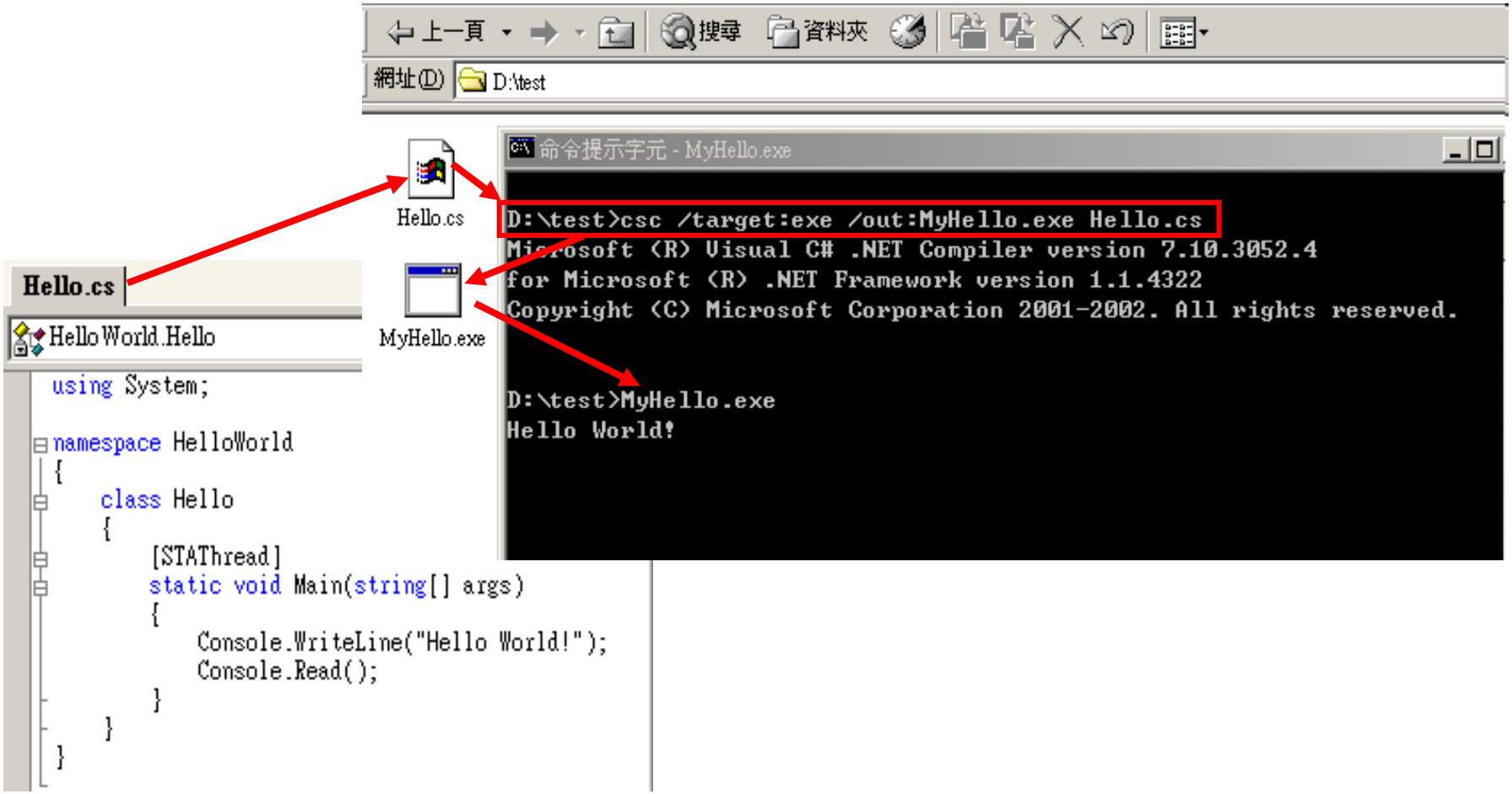
- **ildasm.exe**

  C:\Program Files\Microsoft Visual Studio .NET 2003\SDK\v1.1\Bin

  (for MS. CLR)


系統內容
一般
網路識別
硬體
使用者設定檔
進階
效能
[效能選項] 控制應用程式的記憶體使用，這會影響到您的電腦速度。
效能選項(P)...
環境變數
[環境變數] 告訴電腦某些類型資訊的位置。
環境變數(E)...
啟動及修復
[啟動及修復] 選項會告訴電腦如何啟動，還有如果發生錯誤讓電腦停止時如何反應。
啟動及修復(S)...
確定
取消
套用(A)
環境變數
Administrator 的使用者變數(U)
變數
值
JBOSS_DIST
C:\jboss-3.0.7
lib
c:\Program Files\Microsoft Visual Studio ....
MSDevDir
C:\Program Files\Microsoft Visual Studio\...
path
C:\Program Files\NVIDIA Corporation\C...
SHADERS
C:\BMRT2.6\shaders\
編輯使用者變數
變數名稱(N):
path
變數值(V):
1.1\Bin;C:\sscli\clr\bin\rotor_x86\fastchecked
確定
取消
LIB
c:\Program Files\Microsoft Visual Studio ....
新增(W)...
編輯(I)...
刪除(L)
確定
取消

CSC


上一頁
搜尋
資料夾
網址(D)
D:\test
Hello.cs
MyHello.exe
命令提示字元 - MyHello.exe
D:\test>csc /target:exe /out:MyHello.exe Hello.cs
Microsoft (R) Visual C# .NET Compiler version 7.10.3052.4
for Microsoft (R) .NET Framework version 1.1.4322
Copyright (C) Microsoft Corporation 2001-2002. All rights reserved.
D:\test>MyHello.exe
Hello World!
Hello.cs
HelloWorld.Hello
using System;
namespace HelloWorld
{
class Hello
{
[STAThread]
static void Main(string[] args)
{
Console.WriteLine("Hello World!");
Console.Read();
}
}
}

References

- **Standard ECMA-334(c#)**
  - http://www.ecma-international.org/publications/star