An Introduction to the .NET Framework

Kristian Guillaumier,

Adapted from Applied .NET Framework
Programming by Jeffrey Richter
(Microsoft Press)

Introduction

- Development tools:
  - Languages and Compilers.
  - Development environments (Visual Studio)
  - C# vs Visual C#/
- Class Library.

The CLR

- Common Language Runtime.
- Runtime used to execute all .NET code.
- Instead of compiling to native code, a CLR-compliant language compiles to IL (intermediate language).

Managed Modules

- PE (Portable Executable) Header:
  - Native Windows executable header contains details such as whether the application is a console or window based application or a DLL.
- CLR header:
  - Header interpreted by the CLR.
  - Version of CLR required, entry point (Main()), location of resources, …
- Metadata:
  - Tables of types and members created and referenced.
- IL Code:
  - The output of the .NET compiler. The CLR later translates this to native code.

Calling a Method for the First Time

- Consider the Console.WriteLine method being called in the Main() method.
- Interally the CLR has an undocumented method called JITCompiler:
    - Look up the assembly that contains the WriteLine method (Console).
    - Search the metadata of the assembly to locate the WriteLine method.
    - Get the IL of the WriteLine method.
    - Compile the IL into memory.
    - Set the ‘execution pointer’ to that memory block.
    - Jump to the execution pointer.

Calling a Method for the Second Time

- The JIT compiler ‘caches’ the native CPU instructions in dymamic memory.
- The second time a method is called, the compiled native code block for the method is detected and executed.
- A performance hit (due to re-compilation) is incurred only once.

IL and Verification

- IL is stack based.
- Idea (not actual IL):

```
load 3
add 5
=
8
```

- S
- When compiling IL into Native code, the CLR verifies the code. E.g.:
  - Checks that no memory is read if it wasn’t written to.
  - Methods are called with the right type and number of parameters.
  - …
- Verification has the advantage of guaranteeing that an application running on the CLR does not effect another application on the CLR.
- This means that you can have multiple ‘managed’ applications running inside the SAME windows process without them interfering with each other (these managed applications are called AppDomains).

Framework Class Libraries

- XML Web Services.
- Web Forms (web user interfaces).
- Windows Forms.
- Console Applications.
- Windows Services.
- Component development.

Common Namespaces

- System: basic types.
- System.Collections: stacks, array lists, queues, …
- System.Drawing: 2D graphics (GDI).
- System.IO: Streams, files, directories.
- System.Net: Network programming.
- System.Text: Handling diffrerent text encodings (e.g. ASCII and Unicode).
- System.XML.
- System.Windows.Forms.

The Common Type System (CTS)

- The CTS specification states that a type can have zero or more members:
  - Fields (state of an object).
  - Methods.
  - Properties (one or two methods actually).
  - Events.
- Access modifiers:
  - Private, Family (protected), Assembly (internal), Family and Assemble, Family or Assembly, public.
- Inheritance.
- CTS only supports single inheritance.

Further Reading

- Common Language Specification (CLS).
- Interoperability with unmanaged code (unsafe, DLLImport).