C# Language Internals – Part 1

Essential Techniques

Bart J.F. De Smet
bartde@outlook.com

pluralsight
hardcore dev and IT training

Why Behind the Scenes Matters

- **Satisfying our curiosity**
  - A good property for developers ☺
  - Inspiration for own work

- **Becoming a better coder**
  - Understand caveats of language features
  - Little-known features and tidbits

- **Debugging insights**
  - Being able to dig deeper (C#, IL, assembler)
  - Learn interactions between C#, the CLR, etc.

- **Performance engineering**
  - Cost of language features
  - Avoiding costly mistakes

No Secrets! Introducing Tools

- **The C# compilation pipeline**
  - C# “batch compiler” csc.exe
  - Input:
    - Code files (.cs)
    - References to assemblies (.exe, .dll)
    - Compiler flags (/target, /optimize, /debug)
  - Output:
    - Assemblies (.exe, .dll)
    - .NET modules (.netmodule)
    - Windows Runtime modules (.winmd)

- **Developer productivity tools**
  - Visual Studio (.csproj)
    - MSBuild

Hello World Example

```
using System;

class Program
{
    static void Main()
    {
        Console.WriteLine("Hello, world!");
    }
}
```

```
C:\Demo> csc.exe /nologo hello.cs
C:\Demo> dir /b
hello.cs
hello.exe
C:\Demo> hello.exe
Hello, world!
```

No Secrets! Introducing Tools

- **Inspecting Intermediate Language (IL)**
  - Virtual machine language of the CLR
  - Target language of C#, Visual Basic, etc.
  - JIT compiled into native code

- **ILDASM**
  - Ships with Visual Studio and the .NET Framework SDK
  - IL roundtripping with ILASM (textual language for IL)

- **.NET Reflector**
  - Built by Red Gate
  - Decompiler of IL into C#, Visual Basic, etc.

- **ILSpy**
  - Open source alternative to .NET Reflector

Hello World Example

```
C:\Demo> ildasm.exe hello.exe
```

```
.method private hidebysig static void  Main() cil managed
{
  .entrypoint
  // Code size
  .maxstack  8
  IL_0000: nop
  IL_0001: ldstr  "Hello, world!"
  IL_0006: call   void [mscorlib]System.Console::WriteLine(string)
  IL_000b: nop
  IL_000c: ret
} // end of method Program::Main
```

Non-optimized build

No Secrets! Introducing Tools

- **Inspecting runtime state**
  - State of your program (C#)
  - Underlying data structures of the runtime (CLR)

- **Visual Studio debugger**
  - Stepping through source, setting breakpoints
  - Controlling exception behavior
  - Expression evaluator (Watch, Immediate)

- **Native code debugger**
  - Debugging Tools for Windows download
  - WinDbg.exe and cdb.exe
  - SOS, PSSCOR4, SOSEX, CLRMD debugger extensions

Hello World Example

```
C:\Demo> cdb.exe hello.exe
```

```
0:000> sxe ld clrjit; g
(44c.1934): Unknown exception - code 04242420 (first chance)
ModLoad: C:\Windows\Microsoft.NET\Framework64\v4.0.30319\clrjit.dll
0:000> .loadby sos clr
0:000> !bpmd hello.exe Program.Main
Found 1 methods in module 00007ff7dbc52fb0...
MethodDesc = 00007ff7dbc53fe8
Adding pending breakpoints...
0:000> g
(44c.1934): CLR notification exception - code e0444143 (first chance)
JITTED hello!Program.Main()
Setting breakpoint: bp 00007FF7DBD70090 [Program.Main()]
Breakpoint 1 hit
0:000>
```

Summary

- **Understand internals**
  - Become a better engineer
  - Engineering for performance
  - Invaluable when debugging

- **C# has no secrets**
  - IL code inspection
  - Debugger techniques

- **Familiarize yourself with tools**
  - ILDASM
  - ILSpy
  - WinDbg & SOS