



Extension Methods Will Save The World!



Guy Smith-Ferrier
guy@guysmithferrier.com

Blog: <http://www.guysmithferrier.com>

What's This About ?

- Extension Methods are methods that extend existing classes
- They are used by LINQ
- You can use Extension Methods to extend any class

The Problem

- The RegionInfo class holds information about a geographic region:-
 - The name of the region
 - The currency symbol
 - The name of the currency
- It does not hold:-
 - Country Dialling code
 - Postal code formats
 - Units of measurement (distance, temperature etc.)

The Problem (continued)

```
using System;
using System.Globalization;
using Globalization;

namespace ExtensionMethodsExample
{
    class Program
    {
        static void Main(string[] args)
        {
            RegionInfo regionInfo = new RegionInfo("US");
            Console.WriteLine(regionInfo.GetDiallingCode());
        }
    }
}
```

The Problem (continued)

```
using System;
using System.Globalization;
using Globalization;

namespace ExtensionMethodsExample
{
    class Program
    {
        static void Main(string[] args)
        {
            RegionInfo regionInfo = new RegionInfo("US");
            Console.WriteLine(regionInfo.GetDiallingCode());
        }
    }
}
```

The Solution

```
using System.Globalization;
using System.Runtime.CompilerServices;

namespace Globalization
{
    static class RegionInfoExtensions
    {
        public static string GetDiallingCode(this RegionInfo regionInfo)
        {
            if (regionInfo.Name == "GB")
                return "44";
            else if (regionInfo.Name == "US")
                return "1";
            else
                return null;
        }
    }
}
```

The Solution

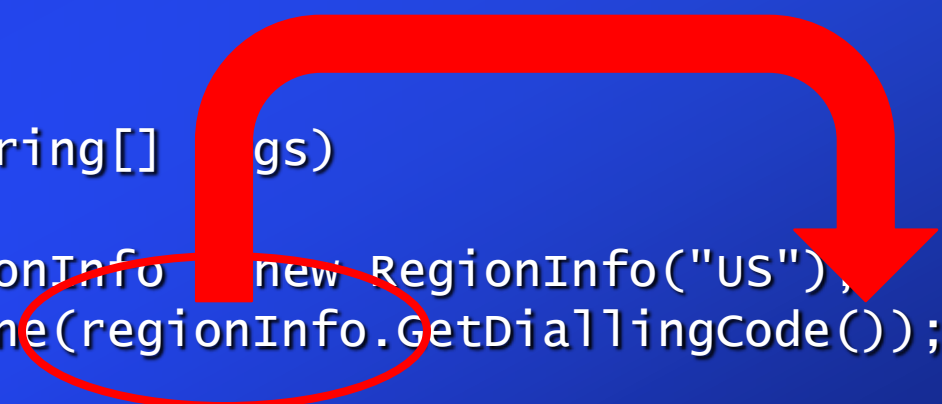
```
using System.Globalization;
using System.Runtime.CompilerServices;

namespace Globalization
{
    static class RegionInfoExtensions
    {
        public static string GetDiallingCode(this RegionInfo regionInfo)
        {
            if (regionInfo.Name == "GB")
                return "44";
            else if (regionInfo.Name == "US")
                return "1";
            else
                return null;
        }
    }
}
```

The Problem (continued)

```
using System;
using System.Globalization;
using Globalization;

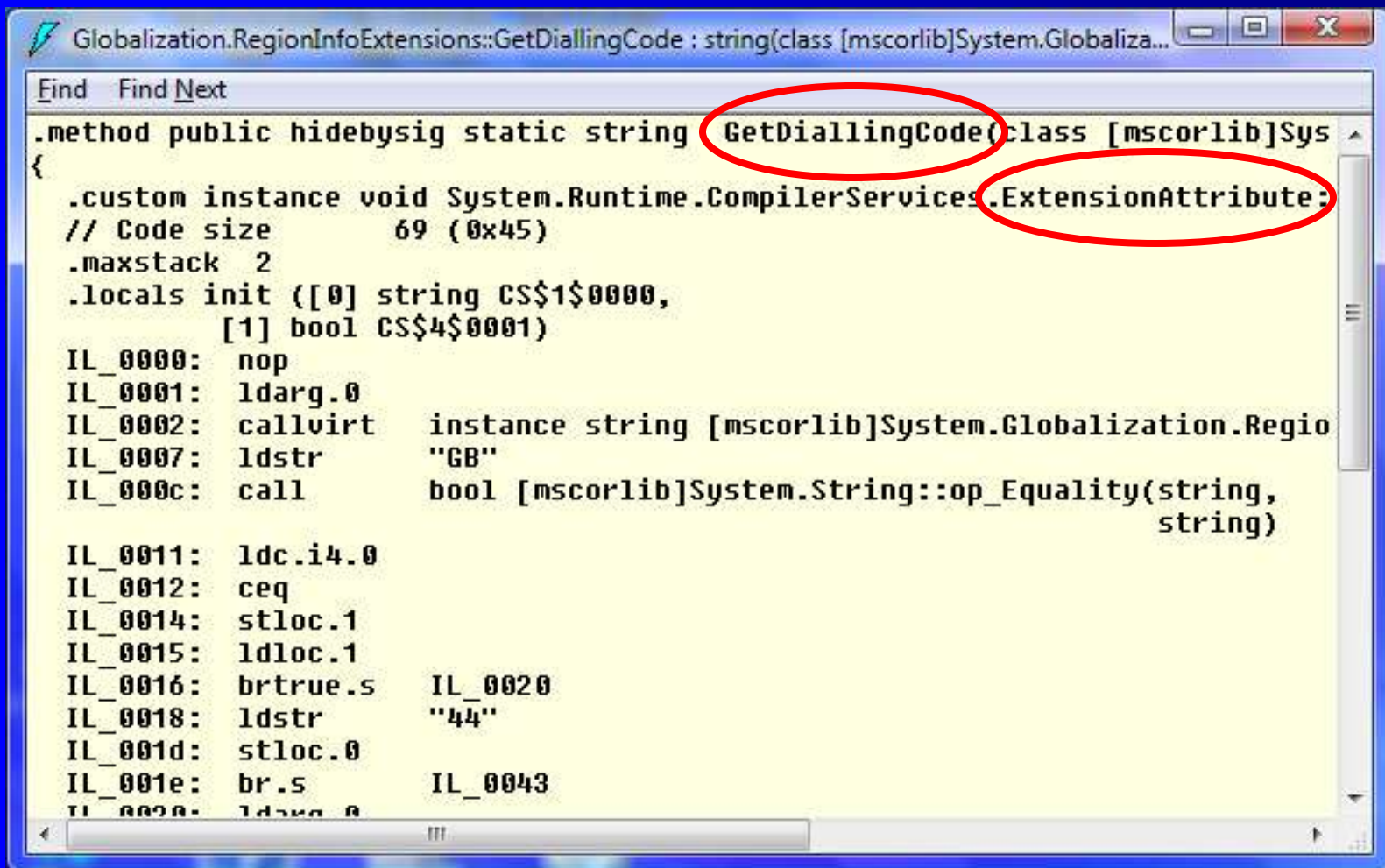
namespace ExtensionMethodsExample
{
    class Program
    {
        static void Main(string[] args)
        {
            RegionInfo regionInfo = new RegionInfo("US");
            Console.WriteLine(regionInfo.GetDiallingCode());
        }
    }
}
```



Extension Methods And The .NET Framework 3.5

- Extension Methods are:-
 - public
 - static
 - receive a parameter of the type being extended
 - decorated with an `ExtensionAttribute`
 - Added by the compiler when the "this" keyword is used

RegionInfoExtensions.GetDiallingCode



```
Globalization.RegionInfoExtensions::GetDiallingCode : string(class [mscorlib]System.Globaliza...
Find Find Next
.method public hidebysig static string GetDiallingCode(class [mscorlib]Sys
{
    .custom instance void System.Runtime.CompilerServices.ExtensionAttribute:
    // Code size          69 (0x45)
    .maxstack 2
    .locals init ([0] string CS$1$0000,
                 [1] bool CS$4$0001)
    IL_0000:  nop
    IL_0001:  ldarg.0
    IL_0002:  callvirt instance string [mscorlib]System.Globalization.Regio
    IL_0007:  ldstr    "GB"
    IL_000c:  call    bool [mscorlib]System.String::op_Equality(string,
                                                    string)

    IL_0011:  ldc.i4.0
    IL_0012:  ceq
    IL_0014:  stloc.1
    IL_0015:  ldloc.1
    IL_0016:  brtrue.s IL_0020
    IL_0018:  ldstr    "44"
    IL_001d:  stloc.0
    IL_001e:  br.s    IL_0043
    IL_0020:  ldarg.0
```

Extension Methods And The .NET Framework 2.0

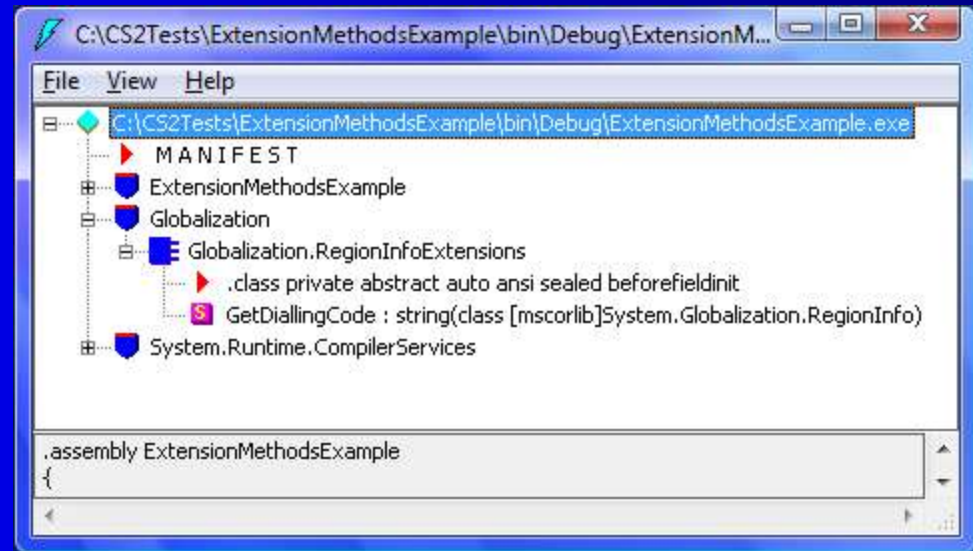
```
namespace System.Runtime.CompilerServices
{
    public class ExtensionAttribute: Attribute
    {
    }
}
```

But...

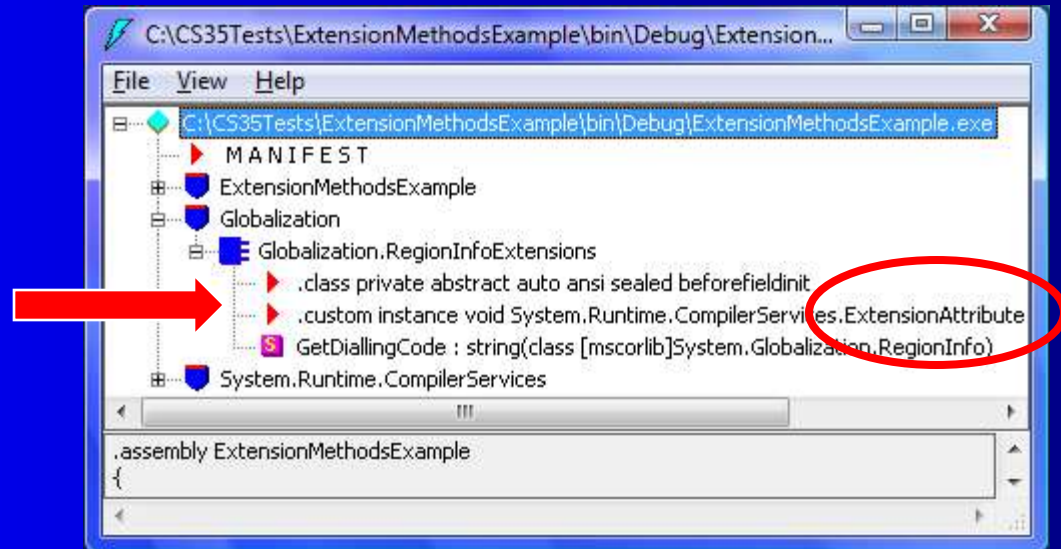
- This solution works with the .NET Framework 2.0 Runtime
- But...
 - It cannot be compiled using the .NET Framework 2.0 compiler

ILDASM And Extension Methods

- .NET Framework 2.0 Compiler



- .NET Framework 3.5 Compiler



The Solution

- It has to be compiled using the .NET Framework 3.5 compiler
 - The .NET Framework 3.5 compiler is multi-targeting
- So you cannot build using Visual Studio 2005
 - Use Visual Studio 2008
 - Or install the .NET Framework 3.5 and build outside of Visual Studio 2005

ASP.NET 2 Web Sites

- ASP.NET 2 Web Sites are compiled at runtime
 - They are compiled using the .NET Framework 2.0 compiler
 - Extension Methods will not compile
- Upgrade to the .NET Framework 3.5

Give Me Some Ideas!

- `String.ToPlural()`, `String.ToSingular()`
- `String.NumberToWords()`
- `RegionInfo.GetAddressFormat()`
- `RegionInfo.GetTopLevelDomains()`
- `RegionInfo.GetTimeZones()`

Give Me Some More Ideas!

- Any class in a file ending in "Helper.cs", "Utils.cs", "Extensions.cs"
 - Environment.GetFrameworkPath()
 - Environment.GetLanguagePacksInstalled()
 - CultureAndRegionInfoBuilder.Export()
 - CultureInfo.IsValidCultureName()
 - StronglyTypedResourceBuilder.GenerateClass()
 - RegionInfo.GetRegions()

Slide 18

- There are three situations where Extension Methods are essential:-
 - 1. The class you want to inherit from is sealed
 - e.g. .NET Framework classes

Slide 19

- There are three situations where Extension Methods are essential:-
 - 1. The class you want to inherit from is sealed
 - e.g. .NET Framework classes
 - 2. You don't own the source code to the class you want to extend
 - e.g. .NET Framework classes

Extension Methods Will Save The World!

- There are three situations where Extension Methods are essential:-
 - 1. The class you want to inherit from is sealed
 - e.g. .NET Framework classes
 - 2. You don't own the source code to the class you want to extend
 - e.g. .NET Framework classes
 - 3. You don't own the source code that creates objects of the class you want to extend
 - e.g. .NET Framework classes