How to use a method from a .NET assembly in MapBasic?

This document describes how to create a .Net assembly – an application extension – with C# with a method you can call from your MapBasic application.

The purpose of this example is to be very basic – you do not even have to install and use Microsoft Visual Studio. Instead you will be using the C# compiler which is available by default on your computer when you are running MapInfo Professional 9.5 or higher.

The business case:

- **convert a Date or a Date/Time value to a long date string, using Regional and Language Settings**
- **retrieve name of weekday or month from a Date or a Date/Time value, using Regional and Language Settings**

As an example three methods are created to convert a Date (or a Date/Time) value to its equivalent long date string representation or to retrieve the name of the weekday or month, which – of course – depend on the Regional and Language Settings.

For example:

You do have the Date value 20120426 (→ YYYYMMDD)

With the existing MapBasic function **FormatDate$()** you can format this date. The **FormatDate$()** function returns a date formatted in the short date style specified by the Control Panel.

If you use *English (United States)* this date will be formatted to 4/26/2012; with *Dutch (Netherlands)* you will get 26-4-2012 instead.

The MapBasic function **Weekday()** returns an integer from 1 to 7, indicating the weekday of a specified date. (Please note: 1 represents Sunday.) So, in the case of our example the return value would be 5.

The MapBasic function **Month()** returns the month component (1 - 12) of a date value So, in the case of our example the return value would be 4.

Unfortunately, MapBasic does not offer a function to return a date formatted in the long date style, or to retrieve the *name* of the weekday or month… That's why you are going to build these methods yourself.

For example to use a "proper" date on printed output, you will use this method to format the date to:

Thursday, April 26, 2012 - *English (United States)* or
donderdag 26 april 2012 - *Dutch (Netherlands)* or
Huebes, Abril 26, 2012 - *Filipino (Philippines)* or
xoves, 26 de abril de 2012 – *Galician*

Creating the .NET assembly

You will create a .NET assembly – a Dynamic Link Library (*.dll) – by creating a single C# source file `MBExtensions.cs` and compiling this file with `csc.exe`.

The C# compiler – csc.exe

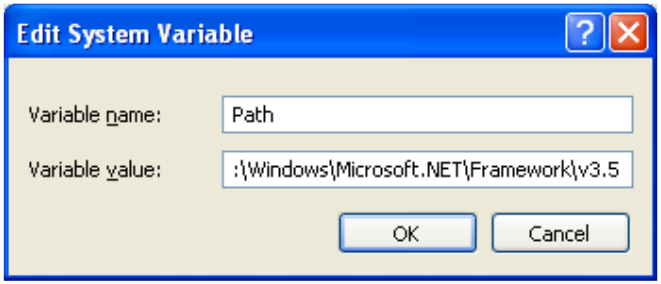
As mentioned before – the C# compiler `csc.exe` is available on your machine, because this compiler is part of the Microsoft .NET Framework 3.5, and this version of the Framework has been installed on your machine – if it was not already there – when you installed MapInfo Professional 9.5 or higher.

Typically, you will find the `csc.exe` here:
C:\WINDOWS\Microsoft.NET\Framework\v3.5\csc.exe

You can call the compiler from a Command Prompt (a "DOS box"). To get some more information about the compiler, you can type:

```
csc -help
```

Please note:
To be able to call the compiler from any location, you should add the location (folder) of the `csc.exe` to your Windows path.



The C# source file

Use your favorite text editor – Notepad++, isn't it? – to create the source file `MBExtensions.cs`.

For the content of the file – see listing 2 below.

This source file contains a method `RegionalLongDate` which will take an unformatted date as input and return a long date string and a method `RegionalMonth` and a method `RegionalWeekday` to return the name of the month and the week respectively. Please have a look at the code (and the comments) to see how it works.

Compile the source file

You can compile the file by typing one of the following commands in your Command Prompt:

```
csc /target:library MBExtensions.cs
or – shorter –
csc /t:library MBExtensions.cs
```

The resulting file – if everything went well – is:

```
MBExtensions.dll
```

Declaration

Before you can call the methods from the `MBExtensions.dll` you have to declare them in your MapBasic application, using the Declare Method statement:

```
Declare Method RegionalLongDate
    Class "MBExtensions.MBDateAndTime"
    Lib "MBExtensions.dll" (ByVal sDateString as
string) As String

Declare Method RegionalMonth
    Class "MBExtensions.MBDateAndTime"
    Lib "MBExtensions.dll" (ByVal sDateString as
string) As String

Declare Method RegionalWeekday
    Class "MBExtensions.MBDateAndTime"
    Lib "MBExtensions.dll" (ByVal sDateString as
string) As String
```

In listing 1 below you will find the code of a little MapBasic application to test your methods.

Please note:
You can populate your string either with a Date (i.e. YYYYMMDD) or a DateTime (i.e. YYYYMMDDHHMMSSFFF) value; in both cases the first eight characters will contain the Year, Month and Day information.

The output in the Message Window will be – depending on your regional settings and the day you run the app – similar to the one below:

Current month is: Abril

Today:
20120426
Huebes
Huebes, Abril 26, 2012

Tomorrow:
20120427
Biernes
Biernes, Abril 27, 2012

Yesterday:
20120425131111000
Mierkoles
Mierkoles, Abril 25, 2012

Listing 1: CallRegionalLongDate.mb

```
Declare Sub Main

''''''''' .NET Assembly MBExtensions.dll - Class "MBExtensions.MBDateAndTime"  ''''''
' Allows you to retreive information from a MapBasic Date (i.e. YYYYMMDD) or
' DateTime (i.e. YYYYMMDDHHMMSSFFF) string

' Convert the string to it's equivalent long date string representation
```

```
' (depends on Control Panel > Regional and Language Options)
Declare Method RegionalLongDate
    Class "MBExtensions.MBDateAndTime"
    Lib "MBExtensions.dll" (ByVal sDateString as string) As String

' Extract from the string the name of the month in the regional language
' (depends on Control Panel > Regional and Language Options)
Declare Method RegionalMonth
    Class "MBExtensions.MBDateAndTime"
    Lib "MBExtensions.dll" (ByVal sDateString as string) As String

' Extract from the string the name of the weekday in the regional language
' (depends on Control Panel > Regional and Language Options)
Declare Method RegionalWeekday
    Class "MBExtensions.MBDateAndTime"
    Lib "MBExtensions.dll" (ByVal sDateString as string) As String
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Global sDateString, sLongDateMB, sMonth, sWeekday as String

Sub Main

sDateString = CurDate()
sLongDateMB = RegionalLongDate(sDateString)
sMonth = RegionalMonth(sDateString)
sWeekday = RegionalWeekday(sDateString)

Print "Current month is: " + sMonth
Print "----"
Print "Today:"
Print sDateString
Print sWeekday
Print sLongDateMB
```

```
sDateString = CurDate() + 1
sLongDateMB = RegionalLongDate(sDateString)
sWeekday = RegionalWeekday(sDateString)
Print "----"
Print "Tomorrow:"
Print sDateString
Print sWeekday
Print sLongDateMB

sDateString = CurDateTime() - 1
sLongDateMB = RegionalLongDate(sDateString)
sWeekday = RegionalWeekday(sDateString)
Print "----"
Print "Yesterday:"
Print sDateString
Print sWeekday
Print sLongDateMB

End Sub Main
```

Listing 2: MBExtensions.cs

```
// Namespace Declaration
using System;

// The MapBasicExtensions Namespace
namespace MBExtensions
{
   // The MBDateAndTime class
   class MBDateAndTime
   {
      public static string RegionalLongDate(string sDateString)
      {
        string mbDateString;
        DateTime stringToDate;
```

```
  stringToDate = ConvertMBDate(sDateString);

  /* Convert the DateTime object stringToDate to
  it's equivalent long date string representation
  (depends on the Regional and Language Options) */
  mbDateString = (stringToDate.ToLongDateString());

  return mbDateString;
}

public static string RegionalMonth(string sDateString)
{
  string mbMonthString;
  DateTime stringToDate;

  stringToDate = ConvertMBDate(sDateString);

  /* Extract from this date the name of the month
  in the regional language (depends on the Regional
  and Language Options) */
  mbMonthString = stringToDate.ToString("MMMM");

  return mbMonthString;
}

public static string RegionalWeekday(string sDateString)
{
  string mbWeekdayString;
  DateTime stringToDate;

  stringToDate = ConvertMBDate(sDateString);

  /* Extract from this date the name of the weekday
  in the regional language (depends on the Regional
  and Language Options) */
  mbWeekdayString = stringToDate.ToString("dddd");
```

```
        return mbWeekdayString;
    }

    public static DateTime ConvertMBDate(string sDateString)
    /* The input string sDateString which comes from your
    MapBasic app will be either a Date (i.e. YYYYMMDD) or
    a DateTime (i.e. YYYYMMDDHHMMSSFFF) */
    {
        int mbYear;
        int mbMonth;
        int mbDay;
        DateTime stringToDate;

        /* Take the Year component from the input string
        (i.e. YYYY) and convert it to an integer */
        mbYear = Convert.ToInt32(sDateString.Substring(0, 4));

        /* Take the Month component from the input string
        (i.e. MM) and convert it to an integer */
        mbMonth = Convert.ToInt32(sDateString.Substring(4, 2));

        /* Take the Day component from the input string
        (i.e. DD) and convert it to an integer */
        mbDay = Convert.ToInt32(sDateString.Substring(6, 2));

        // Assign a value to the DateTime object stringToDate
        stringToDate = new DateTime(mbYear, mbMonth, mbDay);

        return stringToDate;
    }
  }
}
```