NAV TECH DAYS 2014

mibuso.com

.NET INTEROPERABILITY IN NAV 2013 R2 FOR MERE MORTALS

Vjekoslav Babić
(Fortempo)

ABOUT ME

Vjekoslav Babić
consultant, trainer, blogger, author

Twitter: @vjekob
Mibuso: Vjeko
Blog: vjeko.com

Author of many How Do I... videos for MSDN and PartnerSource for NAV 2013, NAV 2013 R2, and NAV 2015

Co-author of "Implementing Microsoft Dynamics NAV 2009" book

AGENDA

Many practical *"how do I do this or that"* in .NET Interoperability

- Arrays
- Strings
- Regular expressions
- File system
- Surviving different data formats
- Converting between types
- Handling Base64
- Using Visual Studio and C# like a pro

System.Array

"A mere mortal C/AL developer understands .NET arrays."

(ancient Chinese proverb)

WHY DO WE NEED .NET ARRAYS?

```
[string[] Split :=] Split(string[] separator, StringSplitOptions options)
```

```
[string[] MonthNames :=] MonthNames(string[] value)
```

```
static [string ToBase64String :=] ToBase64String(byte[] inArray)
```

```
[int LastIndexOfAny :=] LastIndexOfAny(char[] anyOf, int startIndex)
```

```
static [MemoryStream MemoryStream :=] MemoryStream(byte[] buffer)
```

```
[System.Reflection.MethodInfo GetMethod :=] GetMethod(string name, Type[] types, System.Reflection.ParameterModifier[] modifiers)
```

```
[int GetBytes :=] GetBytes(char[] chars, int charIndex, int charCount, byte[] bytes, int byteIndex)
```

THE “ARRAY” PATTERN

```
ArrayOf<MyType>s := ArrayOf<MyType>s.CreateInstance(
  GETDOTNETTYPE(<AnExpressionOfMyType>),
  <NumberOfElements>);

ArrayOfDateTimes := ArrayOfDateTimes.CreateInstance(
  GETDOTNETTYPE(0DT),
  5);

ArrayOfStrings := ArrayOfStrings.CreateInstance(
  GETDOTNETTYPE(''),
  3);
```

LIFE HACK #1

(Pssst! This is trade secrets, don't share this, you!)

Create a template in an easy accessible place (OneNote, for example):

- List DotNet variables of most commonly used types
- Create code templates
- Copy/Paste them when needed

System.String

A powerful class to handle text information.
Maps fully and directly to Text data type in C/AL.

System.String

A powerful class that handles text information

Maps fully and directly to Text data type in C/AL

DEMO

- Parsing a delimited string
- Finding an extension of a file name

REGULAR EXPRESSIONS

Regular expressions are to text what SQL language is to relational databases

Fast and efficient:

- Pattern matching
- Searching
- Replacing

DEMO

Detecting an incorrect e-mail address

Finding e-mail addresses in text

Replacing e-mail addresses in text

HANDLING FILES

System.IO.File and System.IO.FileInfo types

System.IO.Directory and System.IO.DirectoryInfo types

System.Path type

DEMO

Creating a directory structure

Appending text to a file

Checking if a filename contains an invalid character

HANDLING DATE AND NUMBER FORMATS

C/AL only understands date, time and number formats of the current language

.NET understands date and number formats of any language

System.DateTime, System.Decimal, and System.CultureInfo types allow you to:

- Show dates, time, and numbers in any date format supported by Windows
- Convert text in any date format supported by Windows into actual DateTime or Decimal value

DEMO

Reading date, time, and number information in different formats

STREAMS IN .NET

C/AL only knows of two types of streams: input stream, and output stream

There are many different types of streams in .NET, they are all bi-directional (both input and output)

.NET stream map directly and fully to C/AL streams, and variables can be used practically interchangeably

.NET streams can be independent of storage objects and can be manipulated programmatically

DEMO

Downloading a picture from internet and storing it in a BLOB field in the database

System.Convert

EVALUATE's big momma

Allows conversion between practically all built-in value types in .NET

Provides some very useful functionality to handle Base64 encoding

DEMO

Encoding and decoding to and from Base64

CREATING CUSTOM ASSEMBLIES

.NET code is easier to write in C# than in C/AL

C# has no limitations about accessing .NET, C/AL has aplenty

Native .NET code executes faster than any .NET interoperability code written in C/AL

DEMO

Creating and deploying a simple assembly

SOME BEST PRACTICES

- Deploy to Global Assembly Cache (GAC)
- Declare properties, not fields
- Make classes serializable
- Use interfaces if at all possible

GLOBAL ASSEMBLY CACHE (GAC)

Central, machine-wide repository of .NET assemblies

Allows referencing assemblies by knowing their fully qualified name

NAV shows the contents of the Global Assembly Cache in the .NET tab of the Assembly List form

Why should you use GAC?

- Simplifies deployment
- Speeds up development
- Forces you to sign assemblies with a strong name

DEMO

Deployment to Global Assembly Cache (GAC)

FIELDS VS. PROPERTIES

Fields are variables declared directly in a class

Properties look like fields, but they are not just variables; they are essentially facades to access data internally stored in the class

Properties are accessible in C/AL

Fields are not accessible in C/AL

Never declare fields! Properties can be auto-implemented, and it's easy to turn a field into an auto-implemented property.

DEMO

Fields

Properties

Auto-implemented properties

SERIALIZATION

Process that allows a class to be stored (e.g. on a disk, in a database…) and later created as an exact copy of the original

Serialization preserves state of the class

You rarely need to call serialization directly in C/AL, but you benefit if classes are serializable

When creating new classes it is easy to make a class serializable – it costs nothing and makes a massive difference

DEMO

Why non-serializable classes are not good

How to make a class serializable

Benefitting from serialization

INTERFACES

Interfaces enable the highest level of code flexibility by allowing a class to be represented (and handled) as a more general type

Interfaces open doors to various code-level design patterns that simplify coding and enhance extensibility

Interfaces make a lot of senses on classes that perform tasks

Creating an interface from a class is fully automated, it costs nothing, and provides better code structure

Using interfaces in C/AL is as simple as using classes

DEMO

Extracting an interface from a class

Using interfaces in C/AL

Any Questions?

THANK YOU

WHEN YOU ARE PASSIONATE ABOUT MICROSOFT DYNAMICS NAV | *www.navtechdays.com*

NAV TECH DAYS 2014

mibuso.com

LUNCH BREAK

see you back in **60** min.