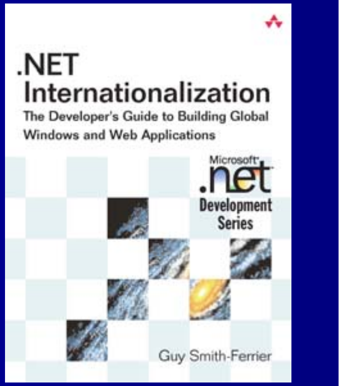
10 Things To Know Before Internationalizing An Application

Microsoft CERTIFIED Professional

Guy Smith-Ferrier
guy@guysmithferrier.com
Blog: http://www.guysmithferrier.com

About…

- Author of .NET Internationalization
  - Visit http://www.dotneti18n.com to download the complete source code



- The .NET Developer Network
  - http://www.dotnetdevnet.com
  - Free user group for .NET developers, architects and IT Pros based in Bristol

Agenda

- 1. The Schizophrenic User Interface
- 2. Localizing ASP.NET 2 Components
- 3. Windows Forms Layout
- 4. Strongly Typed Resources
- 5. Localizing Exception Messages
- 6. Extending The CultureInfo Class
- 7. Uses For Custom Cultures
- 8. Pseudo Translation
- 9. Machine Translation
- 10. Testing Internationalization

demo

1. The Schizophrenic User Interface

1. The Schizophrenic User Interface

Application Type	Class / Component	Localization Support Provided By
Windows Forms	ColorDialog	Operating System
Windows Forms	FolderBrowserDialog	Operating System
Windows Forms	FontDialog	Operating System
Windows Forms	MessageBox	Operating System
Windows Forms	OpenFileDialog	Operating System
Windows Forms	PageSetupDialog	Operating System
Windows Forms	PrintDialog	Operating System
Windows Forms	PrintPreviewDialog	.NET Framework Language Pack
Windows Forms	SaveFileDialog	Operating System
ASP.NET	Calendar	.NET Framework

1. The Schizophrenic User Interface

- 1. CurrentUICulture and CurrentCulture
  - Windows Forms
    - Don't change CurrentUICulture from the Operating System culture
    - Ensure the .NET Framework Language Pack corresponding to the CurrentUICulture is installed
  - ASP.NET
    - Set CurrentUICulture and CurrentCulture to the same culture
- 2. Write your own dialog controls that load resources using the CurrentUICulture
- 3. Write your own dialog controls that use the original controls but localize those controls according to the CurrentUICulture before the control is displayed

demo

2. Localizing ASP.NET 2 Components

demo

3. Windows Forms Layout

4. Strongly Typed Resources

- Non-localizable string resource:-

```
MessageBox.Show("Insufficient funds for the transfer");
```

- Localizable string resource:-

```
MessageBox.Show(resourceManager.GetString("InsufficientFunds"));
```

- Strongly typed localizable string resource:-

```
MessageBox.Show(Form1Resources.InsufficientFunds);
```

5. Localizing Exception Messages

- Non-localizable exception message:-

```
throw new ArgumentException(
    "Value does not fall within the expected range.");
```

- Localized exception messages:-

```
throw new ArgumentException();
throw new ArgumentException(null);
```

- The localized exception message is determined by:-
  - CultureInfo.CurrentUICulture
  - The presence of a corresponding .NET Framework Language Pack

5. Localizing Exception Messages

- What language should exception messages be in ?
  - The user's language ?
  - The developers' language ?
- Should you make a distinction between exception messages ?
  - Developer exceptions
    - ArgumentException
    - IndexOutOfRangeException
  - User exceptions
    - DirectoryNotFoundException
    - FileNotFoundException
    - PathTooLongException

5. Localizing Exception Messages Solutions

- 1. Do nothing
  - Exception messages will be localized
  - Ensure that the exception class name is included in the crash report so that developers can look up the exception message in their own language
- 2. Hard code exception messages in the developer's language
  - Can produce false positives in FxCop rules
- 3. Write an ExceptionFactory class to create exceptions using a fixed culture instead of CurrentUICulture

```
throw new ExceptionFactory.CreateException(typeof(IOException));
```

demo

6. Extending The CultureInfo Class

6. Extending The CultureInfo Class

- The CultureInfo, RegionInfo and TextInfo classes are ideal holders for cultural and regional information:-
  - postal code formats
  - address formats
  - preferred paper sizes
  - units of measurement (temperature, distance, volumes etc.)
  - region’s continent
  - IANA Top Level Domains used by the region
  - time zones that span the region
  - International Distance Direct Dialing code
  - demographics (population, literacy, religions etc.)
  - ToPlural, ToSingular, NumbersToWords

6. Extending The CultureInfo Class (Using Extension Methods)

```
// compile using the LINQ Preview C# compiler
static class CultureInfoExtensions
{
  [Extension]
  public static string GetPostalcode(this CultureInfo cultureInfo)
  {
    CheckNeutral(cultureInfo);
    return new PostalCode(cultureInfo.Name);
  }
}


// use the new method like this:-
CultureInfo cultureInfo = new CultureInfo("en-GB");
maskedTextBox1.Mask = cultureInfo.GetPostalcode().Mask;
```

7. Uses For Custom Cultures Modifying Existing Cultures

```
CultureAndRegionInfoBuilder builder =
    new CultureAndRegionInfoBuilder("en-GB",
    CultureAndRegionModifiers.Replacement);

builder.GregorianDateTimeFormat.ShortTimePattern = "HH:mm tt";

builder.Register();
```

7. Uses For Custom Cultures Unknown Language And/Or Region

Culture Name	Culture EnglishName	Approx. Number Of Users Of This Language In This Region
bn-BD	Bengali (Bangladesh)	125,000,000
eo	Esperanto	2,000,000
fj-FJ	Fijian (Fiji)	364,000
gd-GB	Gaelic (United Kingdom)	88,892
tlh-KX	Klingonese (Klingon) ("tlh" is the ISO code assigned to "tlhIngan Hol", the name for the Klingon language)	431,892,000,000
la	Latin	?
tl-PH	Tagalog (Philippines)	14,000,000

7. Uses For Custom Cultures Known Language Not Known In Region

Culture Name	Culture EnglishName	Approx. Number Of Users Of This Language In This Region
es-US	Spanish (United States)	22,400,000
hi-GB	Hindi (United Kingdom)	1,300,000
pa-CA	Punjabi (Canada)	300,000
zh-CA	Chinese (Canada)	870,000
zh-US	Chinese (United States)	2,000,000

7. Uses For Custom Cultures

- Uses include:-
  - Replacement cultures (e.g. to change inappropriate culture information)
  - Supplementary cultures (e.g. Bengali (Bangladesh))
  - Pseudo translation cultures
  - Commercial dialects (e.g. English (United Kingdom) (Megadodo Publications))
  - New combinations of existing languages and regions (e.g. Spanish (United States))
  - Cultures which are exported from newer operating systems (e.g. Welsh (United Kingdom))

7. Uses For Custom Cultures Resources

- MSDN Webcast: Creating Custom Cultures
  - Download or view online at http://msdn.microsoft.com
- "Custom Cultures" sample chapter
  - Download from http://www.dotneti18n.com

demo

8. Pseudo Translation

9. Machine Translation
Examples Of Machine Translators

- (Check the usage conditions before using)

Name	Address	Language Pairs
AltaVista	www.altavista.com	36
Bowne Global	itranslator.bowneglobal.com	18
Dictionary.com	www.dictionary.com	24
FreeTranslation	www.freetranslation.com	16
Google	www.google.com	18
Online Translator	www.online-translator.com	16
Socrates	www.rustran.com	2
Systran	www.systranbox.com	42
WorldLingo	www.worldlingo.com	13 languages

demo

9. Machine Translation

10. Testing Internationalization

- Guy's Golden Rules Of Development #17
  - Anything a developer has to remember to do they will eventually forget to do

10. Testing Internationalization Examples Of New FxCop Rules

- Control characters embedded in resource string
- Form.Language must be (Default)
- Form.Localizable must be true
- CultureInfo must not be constructed from LCID
- DateTime.ToString() should not use a culture specific format
- Dialog culture dictated by operating system
- Do not use literal strings
- Resource string missing from fallback assembly
- Thread not provided by ThreadFactory

Resources

- .NET Internationalization WebSite
  - http://www.dotneti18n.com
- Blog
  - http://www.guysmithferrier.com

Summary

- 1. The Schizophrenic User Interface
- 2. Localizing ASP.NET 2 Components
- 3. Windows Forms Layout
- 4. Strongly Typed Resources
- 5. Localizing Exception Messages
- 6. Extending The CultureInfo Class
- 7. Uses For Custom Cultures
- 8. Pseudo Translation
- 9. Machine Translation
- 10. Testing Internationalization