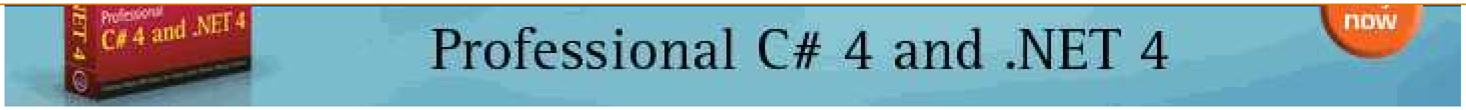
Best practice to make a multi language application in C#/WinForms?



I've been looking into making applications suitable for multiple languages in C# since I need to work on a small project where this is the case. I have found basically two ways to do this:

Set a form's Localizable property to true, set the Language property, fill all the labels and such, and you're 'done'. The major drawback I see in this is: how to make other stuff which is not part of a form ready for multiple languages (e.g. pop-up windows, log files or windows, etc).

Create a resource file, for example 'Lang.en-us.resx' and one for every language, for example 'Lang.nl-nl.resx' and fill it up with Strings. The IDE seems to generate a class for me automatically, so in the code I can just use Lang.SomeText. The biggest drawback I see in this is: for every form I need to set all the labels and other captions myself in the code (and it doesn't seem data binding works with these resources).

I'm sure, however, that there are other methods to do this as well.

So, what is the best practice? What's the easiest for small applications (a few forms, database connection etc) and what scales best for larger applications?

c# winforms localization internationalization

edited **Oct 11 '08 at 16:29** asked **Sep 23 '08 at 7:26**
Mihai Limbasan pbean
5,921 17 36

5 Answers

I have always used resource files for multi-language applications.
The are many articles on the web explaining how to use them.

I have used two different ways:

- A resource file per form
- A global resource file

The global resource file allows you to centralise all the labels (images etc.) in one file (per language), but it means manually setting the labels in the form load. This file can also be used for error messages etc.

A question of taste...

One last point, I write programs in English and French, I use "en" and "fr" and not "en-US" and "fr-FR". Do not complicate things, the different dilelects of English (American, English, Australian etc) have few enough differences to use only one (the same goes for French).

answered **Sep 23 '08 at 7:49**
ThatBloke
266 2 11

1 You have to worry about dialects sometimes. For instance the Chinese characters are completely different in Traditional Chinese (Taiwan) versus Simplified Chinese (mainland China). – MarkJ May 6 '09 at 22:51

good start:

.NET Localization, Part 1: Resource Managers

.NET Localization, Part 2: Creating Satellite Assemblies

.NET Localization, Part 3: Localizing Text

.NET Localization, Part 4: Localizing Units

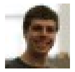
answered **Sep 23 '08 at 7:35**

Ricky AH
1,258 2 13

I recently wrote a program with both German and English language support. I was surprised to find out that if I simply named my english resources LanguageResources.resx and my German resources LanguageResources.de.resx, it automatically selected the correct language. The ResXFileCodeGenerator took care of it all for me.

Note that the fields in the two files were the same and any not yet entered German fields would show up in the application as English as the most non specific file language wise is the default file. When looking for a string it goes from most specific (ex .de-DE.resx) to most specific (ex. .resx).

To get at your strings use the ResourceManager.GetString or ResourceManager.GetObject calls. The application should give you the ResourceManager for free.

edited **Sep 23 '08 at 13:33** answered **Sep 23 '08 at 13:28**

Rick Minerich
1,642 2 12

See following for step by step implementation of localization for windows app:
http://urenjoy.blogspot.com/2008/11/windows-form-localization-in-net.html

answered **Nov 16 '09 at 9:05**

At the moment I am using this simple approach (it is not thread-safe, but it shows the basic idea). To get "Hello" in the current language, call Translations.Hello(). Other terms can be added to the class in the same way.

```
public class Translations
{
    private static string _es;
    private static string _en;

    public enum Languages
    {
        English,
        Español
    }

    public static Languages Language = Languages.English;

    public static string Hello()
    {
        _en = "Hello"; _es = "Hola";
        return Text();
    }

    private static string Text()
    {
        if (Language == Languages.English && !String.IsNullOrEmpty(_en))
            return _en;

        if (Language == Languages.Español && !String.IsNullOrEmpty(_es))
            return _es;

        return "No translation";
    }
}
```

1

Not the answer you're looking for? Browse other questions tagged c# winforms localization internationalization or ask your own question.