

Using Resource Files to Internationalize Your Applications

by Garry English

Background

Resource files are utilized by computer applications to display culture and language specific data. The data contained in resource files can be of any non-executable source, for instance, strings, images, and binary data.

There are many benefits of utilizing resource files. With them you can:

- Modify the resource data without having to recompile your application
- Release new culture-specific resource files without having to recompile your application
- Eliminate the programming complexity for supporting multiple cultures
- Reduce the size of the main executable since the resources are stored in a separate file
- Reduce your application's memory usage since the resources aren't loaded into memory with the main executable

Resources in the .Net Framework

The .Net Framework uses a hub and spoke model for working with resource files. The hub is the main assembly that contains the application's default culture. The spokes are the culture specific satellite assemblies that contain the resources for the specific culture.

When a resource cannot be located in the spoke, the .Net runtime falls back to the hub.

Visual Studio .Net makes the process of creating resource satellite assemblies simple for the developer. It has a built in resource editor that allows the developer to modify resource files.

When compiling an application within Visual Studio that contains resources files, Visual Studio will automatically compile the resource file into a satellite assembly.

For more technical detail we have provided reference links at the end of this article for more information.

Challenges

Even though resource files have simplified application internationalization enormously, they have also introduced new challenges.

The main challenge with working with resource files is having them translated. In a preproduction scenario, the developer would send the resources files to a translator. Translation can require a significant amount time, and during this time, the development version of the resource files will most likely be modified. When the translator delivers the translated version of the resource files, the developer would have to manually merge the resource files together. This is a costly and time consuming process.

In a postproduction scenario, often the client would like to modify the application's resources. Perhaps they would like to rephrase a sentence, change an image, or even introduce an entire new culture. This is all possible with resource files, however most clients do not have the means to perform these tasks. The client then needs to decide whether they will purchase development tools such as Visual Studio .Net or if they should contract a developer to handle the task. Either way this can be an expensive process.

Old Solution

In the past, to address these challenges, we would store the resources in a custom Microsoft Excel spreadsheet. The Excel spreadsheet contained VBScript code that would handle the merge challenge. Also by using Excel, it allowed our client to view and modify the resources without having to purchase a resource file editor.

On the development side, we created a custom tool that would convert our Excel files into resource files. These resource files were then imported into the project and compiled into satellite assemblies.

Using Resource Files to Internationalize Your Applications continued

Although this solution helped improve the resource file process, there were still some drawbacks. Storing the resources in Excel exposed additional steps in the development environment, which decreased productivity. Also, the client still didn't have the means to modify resources themselves once the application was published.

New Solution – The Resource Editor

The Resource Editor is a comprehensive Windows application, developed by Agora that allows a user to modify any type of resource file. It also consists of several features that solve all of the above noted challenges.

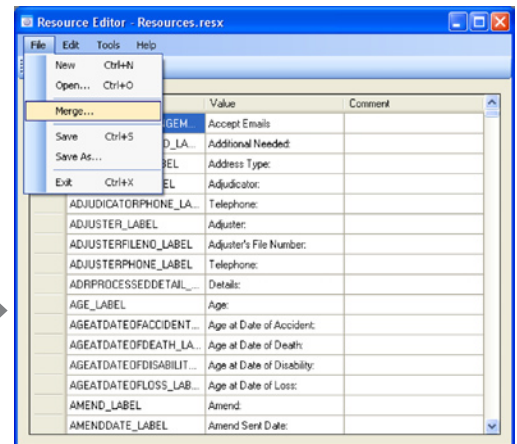
To address the merge challenge, we built a merge feature into the Resource Editor. Simply open a resource file by selecting File > Open. Then to merge another resource file into the opened resource file select File > Merge.

The Resource Editor was primarily designed for the client. The Resource Editor allows the client to create new resource files, update existing resource files and it also generates satellite assemblies that can be deployed to their published application.

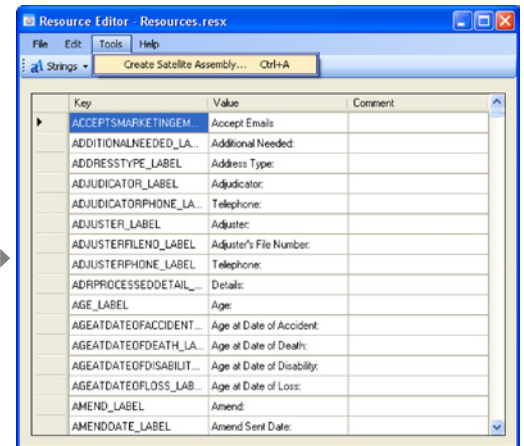
To create a satellite assembly from a resource file, simply select Tools > Create Satellite Assembly

The client no longer needs to purchase expensive development tools or contract a developer for maintaining their application's resources. It can all be done using the Resource Editor.

The Resource Editor also dramatically increases the developer's productivity, because there is no need for any additional tools or steps for working with resources in the development environment. The developers can continue to use Visual Studio .Net for managing the application's resources.



Merge Function in Resource Editor



Satellite Assembly Creation in Resource Editor

Agora Consulting Partners
36 Toronto Street, Suite 960
Toronto, ON M5C 2C5

www.agorainc.com
sales@agorainc.com
416-304-1338

For more information, please visit our
website at www.agorainc.com/development

Garry English is a Senior Software Developer.
He can be reached at genglish@agorainc.com



Agora
Enabling your business
through technology