# INTERNATIONALIZATION FEATURES IN THE MICROSOFT .NET DEVELOPMENT PLATFORM AND WINDOWS 2000/XP

**Dr. William A. Newman, Texas A&M International University, wnewman@tamiu.edu**
**Mr. Syed S. Ghaznavi, Texas A&M International University, ghaznavi7070@yahoo.com**

## ABSTRACT

*Recent advancements in Information Technology and in particular, applications development, have made it possible to envision an online world where PCs, servers, smart devices and Internet-based services can collaborate seamlessly. Businesses need to be able to share data across global distances, integrate processes, and join forces to provide for customized and comprehensive business solutions. Information will need to be available wherever and whenever irrespective of the computing device, platform or application in use. With these developments comes the challenge of standardizing global applications across different platforms and worldwide cultures. Microsoft's new platform for building, deploying, operating and integrating international services is Visual Studio. NET. This paper describes the internationalization capabilities of the .NET platform under Windows 2000/XP and discusses its features for developing applications for a global audience.*

**Keywords:** Globalization, .NET Framework, Multi-Language Computing, Multi-Lingual Interface

## INTRODUCTION

An important topic in software development that has often been overlooked is internationalization. The Internet now allows for the worldwide distribution of software. This issue was overlooked in the early days of the PC since most users were also part-time programmers and most programming languages are based on a subset of English. At the time, the majority of users could deal with English-only applications even when it wasn't their native language, however, with the widespread use of PC's across the globe, users are becoming less tolerant of applications that don't communicate with them in their own native language. With the release of the .NET Framework, Microsoft has both simplified and enhanced the ability for developers to create global-ready applications.(1) The Microsoft .NET Framework not only makes it possible to build international applications, but many of the tools such as Visual Studio .NET make it quite easy. This paper looks at internationalization with .NET and discusses some features in the .NET development environment that can be used to make applications truly global.

## OVERVIEW OF WINDOWS 2000/XP INTERNATIONAL SUPPORT

Providing increased support for international and multilingual computing, Windows 2000 supports sixty scripts, hundreds of languages and 126 locales that will be discussed in detail in the following. Above all, the international feature set of Windows 2000 satisfies two key customer user requirements by:

- Enabling global communication, allowing users to edit content in any language and to work in any given language.
- Simplifying the deployment, maintenance and support of different language releases of the operating system worldwide, reducing the complexity and overall cost.

Windows XP extends this global support and brings for the very first time multilingual computing support into the client space. The following section provides a brief review of Windows 2000's international support and then Windows XP's improvements and discussion of the expanded feature list for use in a global solution.

## WINDOWS 2000 INTERNATIONAL FEATURES

### Single Worldwide Binary Code:

Windows 2000 and all its different versions use the same executable code to run the Win32 based applications; the only difference between these binaries from one language to another is in the translation of their resources. This allows users to:

- Enter text in any language using any version of Windows 2000. For example, a user can write Hebrew on a Japanese version of Windows 2000.
- Run any language version of Win32 applications on any language version of Windows 2000. For example, one can run an Arabic Win32 application on a Russian Windows 2000.

The single worldwide binary also benefits developers, who can:

- Build all language versions of their product on one system. Shipping one functional core-binary to all platforms and for all different language versions reduces the application distribution problem and the costs of development significantly by eliminating conditional compiling and the need to maintain separate source code branches. Providing support for shipping in all supported languages at the same time greatly simplifies application deployment.(2)

### Fully Unicode:

Multilingual computing is made possible through Windows 2000's native Unicode encoding, Unicode Transformation Format (UTF-16 little-endian) and its complete support for Unicode. There is no more ANSI/OEM code page dependency under Windows 2000 and support for new scripts (such as the Indic family, Armenian, and Georgian) is added through pure Unicode encoding and no ANSI or OEM code pages are defined for these scripts. Therefore, hundreds of languages are supported using this approach..(2)

### Complex scripts support:

Several scripts, due to their linguistic requirements, need special handling when it comes to the layout and display of text electronically. A few of these challenges are:

- Bi-directionality of certain scripts (Arabic, Hebrew) in which characters are usually written from right to left, with some portions of text written from left to right such as numbers and embedded Latin based characters.(9)
- Contextual shaping of characters (Arabic and Indic family of scripts). Characters change their shape and location within a rendering area column of the characters, e.g. an Arabic character can have four different shape forms.
- Layout of combining characters (Arabic, Indic, and Thai scripts) for a complete word due to arrangement of characters.
- Justification (Arabic).
- Word breaking and warping (Thai).(2)

"Uniscribe" the layout and shaping engine for these scripts is also known as the Unicode Script Processor. It is built into Windows 2000 and provides consistent support across its clients (Windows 2000, Microsoft Office 2000 and Office XP, Microsoft Internet Explorer 5.0 and beyond). Uniscribe is a collection of APIs that enables a text layout client to format complex scripts. Uniscribe supports the complex rules found in scripts such as Arabic, Indian, and Thai. Uniscribe also handles scripts written from right-to-left such as Arabic or Hebrew, and supports the mixing of scripts. For plain-text clients, Uniscribe provides a range of ScriptString functions that are similar to TextOut, with additional support for caret placement. The remainder of the Uniscribe interfaces provides finer control to clients.(3)

**Multilingual User Interface:**

The Windows 2000 Multi-Language Version (MUI) allows users to select the language of the User Interface (dialog boxes, menus, HTML help files etc). This allows users to:

- Switch the language of the UI without rebooting
- Set the UI language per user (based upon user account profiles on the system)
- Add/remove language resource modules

To support MUI, changes were made in the system's resource loader to load the right resource files for a given process based on the currently selected UI language. Users can install MUI resource files for all 24 languages in which Windows 2000 is localized.(2)

## WINDOWS XP INTERNATIONAL FEATURES & IMPROVEMENTS

Support for displaying regional and culturally correct formatted data (calendar, currency symbol, date & time, sort tables etc.) is provided by the Windows 2000 National Language Support (NLS) API's. While all versions of Windows XP (Personal, Professional, and Server) provide the same level of international support as Windows 2000. Windows XP's new international coverage includes:

- 135 locales – nine more than Win 2000 (A locale is a collection of language-related user preference information represented as a list of values. Each system has at least one installed locale and often has many locales from which the user can choose)
- Support for Farsi and Urdu
- Support for complex scripts such as Telugu, Syriaci and Old Hangul.

- East Asian languages enjoy better font support.
- More Regional Setting options.
- Additional setup and administration options"(2)

## DEVELOPING GLOBAL APPLICATIONS

Microsoft recommends a three-step process developing globalize applications consisting globalization, localizability, and localization. Globalization is writing the application's executable code. A truly global application is culture and language neutral with a user interface that supports localized user interfaces and regional data. The globalization process however, does not involve translating the user interface. Localizability verifies that an application can be localized due to the separation of code from local culture resources. Finally, localization customizes the user interface into the target culture.(5)

To identify the target culture/language, Windows 2000/XP uses a four part locale identifier (LCID) to identify a locale. In addition to specifying sorting orders for text strings, the LCID contains information about the language and any sub-language. The LCID is assigned at system install and is assigned to specific users and all created user threads. The LCID can be changed both by using the Control Panel/Regional Settings in the Control Panel or by an application. The LCID controls not only resource selection, language culture (like what should be the language of an error message), and items like currency symbols and date formatting. (6)

The advantages of global development consist of 1) a much larger user base for products, 2) the inherent ability to scale upward to new cultures, and 3) efficient resource use.(5)

## IMPLEMENTING A MULTILINGUAL USER INTERFACE

Writing only a single binary is the first step towards truly world-ready products and helps to reduce development and support costs. Implementing a multilingual user interface that allows users to switch between all supported languages is the next logical step towards achieving customer satisfaction worldwide and writing Unicode-aware code and creating satellite DLLs for the language resources makes these goals much easier to achieve. There are generally three methods used to implement Multilingual User Interface (MLI) binaries: (7)

1) Language dependent binary with built in resources
2) One core binary with one international resource DLL
3) One core binary with one resource DLL per target language

Figure 1 illustrates the file distribution of an international application using each of these techniques and targeting the English, German and Japanese languages.

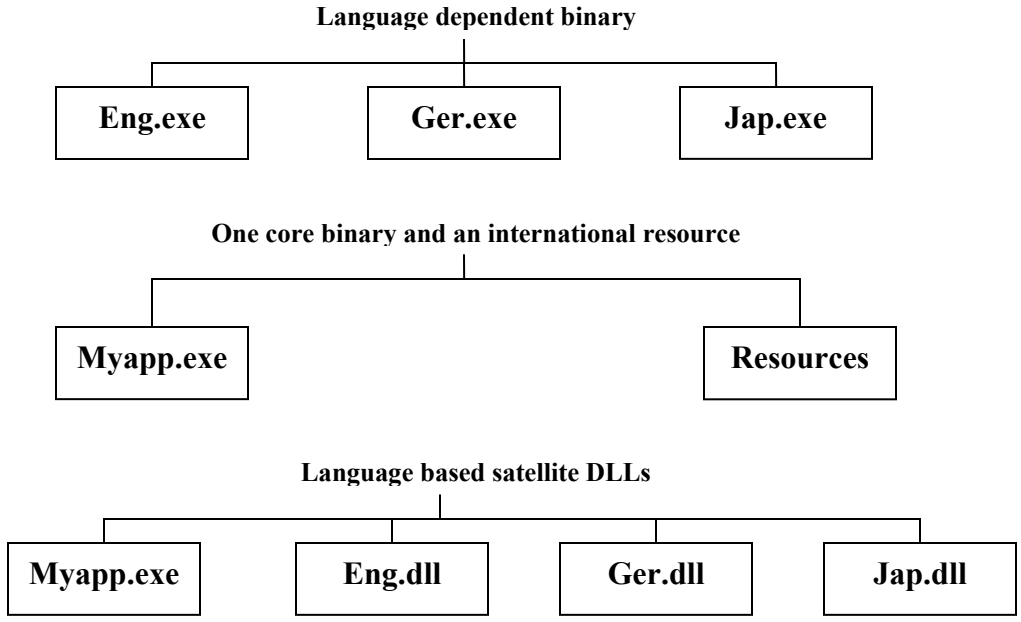


**Figure 1: File distribution comparison (Adapted from Global Software development)** (7)

**Method 1: Language dependent binary**
This traditional approach involves compiling one binary, containing both source code and resources. A separate binary is required for each of the languages targeted.
Advantages

- Implementation.

Disadvantages

- Separate source tree required for each language supported
- Resource changes require a complete binary compilation
- Different instances of the application required to switch UI language
- Excessive storage requirements, multiple copies of core binary

The disadvantages of language dependent binaries are such that this approach is now considered obsolete. The two approaches discussed below are considered better for UI switching applications.(7)

**Method 2: One resource DLL for all languages**
This method attempts to separate the resources from the source code by creating DLL's that are resource-only. Designers can use the same resource ID's identified under different language tags.
Advantages

- Implementation feasibility
- Allows UI switching without restart
- "No compile" resource update possible

Disadvantages

- Difficult to update with new languages
- Difficult to install a subset of UI languages

- Unsupported languages waste memory and disk space
- Cannot change thread locale on Windows 9x(7)

**Method 3: One resource DLL per target language (satellite DLL's)**
This method expands on method 2. With this method, a separate DLL is created per language rather than having a single DLL containing all languages. In the Windows 2000 Multi-Language Version, it is also possible to have different users with different UI languages selected. The satellite DLL's method is used in Office 2000, Internet Explorer 5.x up, and Windows 2000/XP.(7)
Advantages

- Allows user interface switching
- "No compile" resource update possible
- Complete control over the languages installed
- Easy to update with new languages
- Language-specific updates do not affect all languages
- No need to worry about user, system and thread locales
- Can be implemented for Windows 9x, Windows NT and Windows 2000

Disadvantages

- Need to maintain all language satellite DLLs synchronized(7)

## CAPABLITIES FOR ADAPTING TO A SPECIFIC CULTURE

The DateTime Structure provides methods such as the DateTime.ToString Method and DateTime.Parse Method that allow you to perform culture-sensitive operations on a DateTime. The DateTimeFormatInfo Class is used to format and display a DateTime based on culture. DateTimeFormatInfo defines how DateTime values are formatted and displayed, depending on the culture. For example, using the ShortDatePattern, the date February 1, 2001 is formatted as 2/1/2001 for the "en-US" culture and 01/02/2001 for the "en-GB" culture.(5) Methods and properties in the DateTime structure use the local time zone for calculations and comparisons. These methods provide overloads that allow converting the string representation of a date and time to a DateTime type. Format of a DateTime for a specific culture can also be changed. If time zone is not specified in the string that is passed to these methods, they return the parsed date and time without performing a time zone adjustment. The date and time are based on the system's time zone setting.

A globalized application should be able to display and use calendars based on the current culture. The .NET Framework provides the Calendar Class as well as the following Calendar implementations: GregorianCalendar, HebrewCalendar, HijriCalendar, JapaneseCalendar, JulianCalendar, KoreanCalendar, TaiwanCalendar, and ThaiBuddhistCalendar. The CultureInfo Class has a CultureInfo.Calendar Property that specifies a culture's default calendar. Some cultures support more than one calendar. The CultureInfo.OptionalCalendars Property specifies the optional calendars supported by a culture.The NumberFormatInfo Class defines how currency, decimal separators, and other numeric symbols are formatted and displayed based on culture. For example, the decimal number 10000.50 is formatted as 10,000.50 for the culture "en-US" and 10.000,50 for the culture "de-DE".

Finally, the CultureInfo and the RegionInfo classes include information on currency however; only one currency can be specified per culture. For example, since the euro is the

official currency of Belgium, Germany, Spain, France, Ireland, Italy, Luxembourg, Netherlands, Austria, Portugal, Finland, and Greece, the .NET Framework and Microsoft Windows XP use the euro as a default (5)

## CONCLUSION

Developing applications customized for global audience is imperative for today's business environment. The .NET Framework, Common Language Runtime, and Windows 2000/XP make the development of international applications an easy and smooth endeavor. Using Visual Studio.NET and WinRes, it's possible to create Windows Forms applications, which can present an adaptive user interface depending on the language settings of the target machine. In addition, the model of resources contained in satellite assemblies and the fallback mechanism built into the runtime allow for developers to build international-ready applications that can become global applications without changing code or deploying a completely new version of the application. These features in the .NET framework work in conjunction with the operating system's encoding features for different languages and regions.

## REFERENCES


1 http://www.microsoft.com/net/basics/net_today.asp, Letter by Bill Gates to Developers and IT Professionals, January 14, 2002

2 http://www.microsoft.com/globaldev/articles/winxpintl.asp, Global Software Development

3 Microsoft Development Network, 'Uniscribe Text formatting Engine', F. Avery Bishop, David C. Brown, David M. Meltzer http://www.microsoft.com/msj/defaultframe.asp?page=/msj/1198/multilang/multilang.htm&nav=/msj/1198/newnav.htm

4 http://msdn.microsoft.com/netframework/productinfo/overview/default.asp, .NET Framework explained

5 http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguide/html/cpcondevelopingworld-readyapplicationsoverview.asp, Developing World Ready Applications, ".NET Framework Developer's Guide

6 'Let your Applications Span the Globe with Windows Forms and Visual Studio.NET', Jason R. Bell, MSDN Magazine home, http://msdn.microsoft.com/msdnmag/issues/02/06/internat/default.aspx

7 http://www.microsoft.com/globaldev/articles/muiapp.asp, Global Software Development

8 Design a single UNICODE Application that http://www.microsoft.com/globaldev/articles/singleunicode.asp

9 http://www.sun.com/developers/gadc/technicalpublications/presentations/iuc14.pdf, Bidirectional text formatting Sun Microsystems

10 Microsoft Development Network, Locale Information http://msdn.microsoft.com/library/default.asp?url=/library/en-us/wceui/htm/_wcesdk_specifying_locales_with_nls.asp

11 'Opening the Gates to a new era', Application Development Trends Magazine, http://www.adtmag.com

12 http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpref/html/frlrfSystemGlobalization.asp, MSDN .Net Framework Class Library