Helping You Understand Internationalization

I18N

INTRODUCTION

This paper is intended for program managers, project managers, engineering leads, release engineers or anyone, who will be involved in a product or service adaptation for its global deployment. Internationalization is a prerequisite for successful localization and this paper describes the basic dos and don'ts of this globalization discipline.

KEYWORDS: *Internationalization, I18N*

First: What does I18N Stand for?

It is an industry recognized abbreviation for the term internationalization. There are 18 letters between the letters I and N.

Why Internationalization?

The goal of internationalization is to develop a program's code in such a way so that it can function properly on non-native operating systems and so that it can be successfully localized for a particular user community without having to rewrite the executable code.

Single Executable

According to the findings of the ELTIS Project[1], "software maintenance is clearly the most expensive and laborious phase of system development. Often in case of successful software it causes 50-75% of the costs of system development to the producing organization". So this means that

[1] ELTIS is a research project carried out by Information Technology Research Institute of University of Jyväskylä with two main goals: 1) development of the methods to evaluate the profitability of extending the life time of information systems, and 2) development of a framework for organizing modernization of legacy systems. http://www.titu.jyu.fi/eltis/index.en.html Source of the quote no longer available on the Internet.

if an organization spends $1 million on the initial deployment of a software product, they should have additional $1 - $3 million in their budget for the software maintenance! Now, if an organization has localized this software into, let's say nine languages and they have not kept a single executable code, this means that they will have to maintain nine different versions. There is no need to estimate the cost; point well taken → single executable is the way to go!

There are clear advantages to developing international software while keeping a single executable code. These are some of the most obvious:

- One code base to maintain, thus lower maintenance cost;
- Identical user experience around the world;
- Simplified localization effort;
- Decreased overall testing effort;
- One upgrade path for all languages.

You may ask, since it is so obvious, why doesn't everybody do it? Well, there are some challenges as well:

- To maintain the identical code base, localized versions sometimes ship with known defects, even if they are easy to fix;
- Legacy code must often be rewritten, which can be difficult and expensive;
- Legacy native and localized versions must be merged which can present engineering, testing and organizational challenges.

This is to say that single executable cannot be taken as one size fits all approach. Sometimes, where legacy software exists, rewriting the code may not be the right business decision.

Maintaining single executable code may require an up-front investment but will present significant savings in the long run.

Features of Properly Internationalized Software

You have decided to enter the global market and you are ready to internationalize your software. The following gives a very brief overview of the most basic internationalization tasks:

STRING EXTERNALIZATION

In the ideal localization process, the strings (text that appears in the user interface) are translated and then plugged back into the software. Upon compilation, the software runs properly and text appears in the target language.

In order to achieve this, strings must be externalized from the code and placed in the so-called string tables. No string should stay hard-coded, as it will be likely omitted during the localization process. At the same time, if translators go directly into the code to translate strings, they may erroneously translate pieces of the executable code, which will cause the program to malfunction.

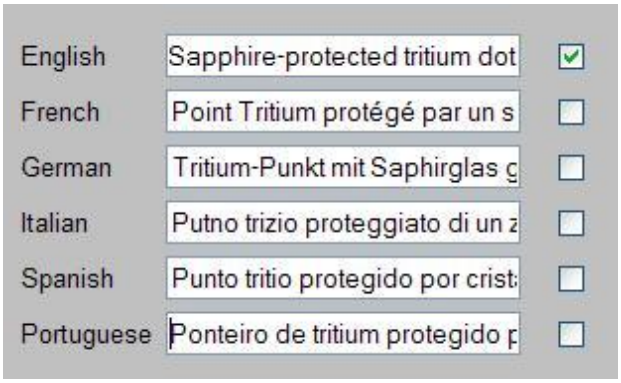
Hardcoded strings complicate localization and present risks of defects leading to potential localization schedule slippage and cost increase.

STRING EXPANSION

Some languages use shorter words and simpler sentences, while others use long words and are rather complex. This means that when you transalte from English into German, for example, it is likely that the German translation will be longer than the English. The following is an example of an English string translated into French, German, Italian, Spanish, and Portuguese respectively:

Sapphire-protected tritium dot
Point Tritium protégé par un saphir
Tritium-Punkt mit Saphirglas geschützt
Putno trizio proteggiato di un zaffiro
Punto tritio protegido por cristal zafiro
Ponteiro de tritium protegido por uma safira

As you can see, all of the translations are longer than the English. Therefore, when developing an application, the developers must make sure to leave extra space (at least 30%) for dialog expansion. Otherwise, the strings will be truncated as in the following example.



STRING CONCATENATION

Another interesting thing about languages is that the syntax (the word order) varies. So, for example, if you want say "All credit institutions must observe the credit laws from 1961." watch what happens to the word order when this sentence is translated into German.

All credit institutions must observe the credit laws from 1961.
Alle Kreditinstitute müssen sich an das Kreditwesengesetz von 1961 halten.

Hard for the eyes? Indeed. The words assume different positions in the sentence, depending on the language. In this German example, the verb *observe* is translated as *"hold onto"* and one part of it comes last in the sentence. Also, an additional preposition *"an"* is added to the sentence.

For this reason, concatenating strings, where the word order is hard coded is an issue for localization.

CALENDARS AND DATE/TIME FORMATS

Most countries use the Gregorian calendar. However, some cultures use calendars that have different starting points or that are synchronized with the moon rather than with the earth's rotation around the sun.

Date and time are also displayed differently when you move from a culture to culture. For example, September 3rd, 2003 would be displayed:

In the USA as	*9/3/2003*
In Germany as	*3.9.2003*
In Korea as	*2003/09/03*

Naturally, the application must be designed so that it can handle these differences.

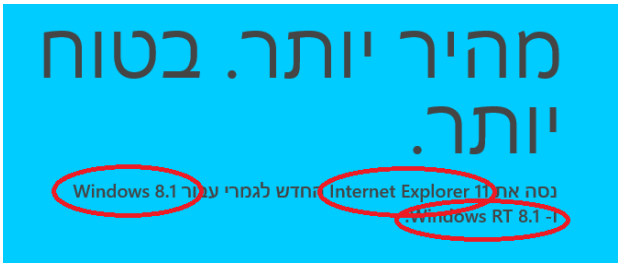
CURRENCY FORMATS

Not only that application must be able to correctly display various currency signs ($, £, €, and ¥, for example) but it must also be smart enough to display the numerical values properly. It must consider the various uses of thousands separators, decimal separators, the number of decimal points etc… The following are just a few usage examples:

France: *1 234 567,89 €*
Germany: *1.234.567,89 €*
Greece: *€ 1.234.567,89*
Japan: *¥1,234,567*
UK: *£1,234,567.89*
USA: *$1,234,567.89*

BIDIRECTIONAL SUPPORT

In the Western world, people are used to a writing system that goes from left to right. However, some languages, such as Hebrew or Arabic are written from right to left. To make things a bit more complicated, even in languages such as Hebrew, some words (especially those of a foreign origin) or numbers are written from left to right. So the application must be able to support this bi-directionality. The following is an example of a Hebrew right to left writing with some words written from left to right (circled).



DOUBLE-BYTE CHARACTER SUPPORT

If you are developing an application that may be eventually localized into languages such as Chinese and Japanese, you must implement double-byte support. Most of the languages that use pictorial characters that represent words require more than one byte to display.

Conclusion

These were just a few but the most fundamental concepts of internationalization. If you have internationalization questions, if you need advice about internationalization support or if you need an audit of your existing software application do not hesitate to contact us. EzGlobe's team will be happy to help.

