Tips for Developers: Writing Code for Localization

1.0 Introduction

Software localization is the process of adapting software for a specific geographical region. The localization process includes translating the software interface and messaging into the appropriate language and adapting formats for date, time, currency, measurements, etc. There may be additional changes made to accommodate the culture of the region such as changes to colors, logos, or pictures.

This paper provides a few simple tips on how to develop software for localization. Most localization issues can be avoided by designing and writing software with localization in mind. Considering localization up front can help increase the efficiency and reduce the cost of localization. Furthermore, if developers use the localization guidelines provided in this paper they can improve the quality, accuracy, and usability of the product after it has been localized. This paper is not intended to be a comprehensive list of good practices to ensure that an application is ready for localization, but rather it should serve as a starting point for developers learning about localization.

2.0 String Length

Developers should keep in mind that when an application is localized the strings in the user interface will be translated into the chosen language. Problems occur in the user interface when the translated strings are significantly longer lengths than the original strings. Restrictions on strings lengths can potentially break a build. For example, if a string goes into a boot sector of a fixed byte size and the assembly code grows, there is no space for a longer string, and the application could crash. If the application cannot accommodate a longer string, the user-interface text strings and features such as control buttons may need to be redesigned adding time and cost to the project.[1]

The developer can avoid user interface issues by either dynamically allocating or limiting the length of text strings. Developers can limit the maximum length of all text strings to no more than 70 percent of the space available. This ensures that all text strings can expand up to 30 percent after translation. Push buttons can be designed to change size dynamically to accommodate any text size increases that may occur in translation, avoiding further software changes.[2]

3.0 Variables

Developers should use unique variable names for each variable. Word order varies from language to language. If the same variable is used more than once, the translated result may be worded incorrectly. See Table 3.0 for examples of code lines written first with duplicate variables and then with unique variables. [1]

Duplicate Variable Names	Unique Variable Names
Set created on %s of %s	Set created on %1 at %2
Printing %s of %s on %s	Printing %2 of %4 on %5

Table 3.0 Duplicate and Unique Variable Names

4.0 Prepositions

Developers should avoid strings that use prepositions with variables. Strings that contain a preposition with a variable can be difficult to localize because, in some languages, different prepositions are used in different contexts. Since prepositions describe space and time, the developer should use a word that specifies what the variable describes. For example if data is collected on a specific date and at a specific time, the words "date" and "time" should be used instead of the prepositions “on” and “at.’”[1]

5.0 Sentences

Developers should put all of the text for a complete sentence in a single string. When a sentence is broken up into several strings, the strings do not necessarily appear consecutively in the localizer's string table. It is time consuming to piece strings together to form a correct sentence. Because sentence structures differ from language to language, there will not be a one-to-one match across strings. Erroneous automated translations may result. An example of such a translation is shown in Example 5.0.

The original sentence written in English:

> "When this box is checked, Windows NT does not" "automatically display the user name of the last person" "to log on."

The same sentence translated into German and then translated back into English:

> "When this control box checked is, plays" "Windows NT not automatically the name of the user, who him/herself last in the dialog box" "Authentication logged has."

Example 5.0 Sentence Divided Across Three Strings

6.0 Conclusion

Localization can be a costly and time-consuming part of a software release cycle. Many common issues can be avoided if developers design software using the tips given in this paper.

References

1. "Localization Guidelines for Your Code. " *Microsoft Office Dev Center*. Microsoft, n.d., Web. 09 <Aug. 2013, http://msdn.microsoft.com/en-<us/library/office/aa163852(v=office.10).aspx>

2. “Understanding Translation.”*Tools Development Guidelines for Application Design Guide*

Release 8.98 Update 4 Part Number E14698-02. JD Edwards EnterpriseOne.n.d., Web. 09

Aug.2013,<http://docs.oracle.com/cd/E17984_01/doc.898/e14698/translation_issues.htm#g8d6ab57f7cedeaac_ef90c_10a77c8e3f7__7883>