How do I use text that has been translated into other languages in my screens and messages?

Question:

How can I display Chinese, Japanese or other double-byte characters in my user interface?
How do I specify Unicode characters in a language resource file?
What file format do I need to get from the translator?

What word processors or editors do I need to use?

Answer:

The isCOBOL APS language resource feature provides an easy way to localize your application to multiple locales and to configure the text for your user interface at runtime from among different languages.

During the initial program initialization and optionally before calling a subprogram, the isCOBOL Runtime Framework loads a specified language resource file. This file is a Java properties file which contains any number of name=value pairs. You specify where in your program you want these resource values to be used with a letter R followed by the quoted name of the resource. For example,

DISPLAY R"Greeting".

will use a resource named Greeting that has been loaded from the language resource file. If your resource file contains the following:

Greeting=Hello World

Then the program will display:

Hello World

A resource name can be used anywhere in your source code where it is valid to use a string literal.

Follow these steps to create a multilanguage version of your program:

1. Search for string literals in your program source code

2. Copy each string literal to a text file and remove the quotes
3. Add a resource name followed by an equal sign to the beginning of the line in the text file
4. Replace the original string literal in the source code with the letter R (or lowercase r) followed by the quoted resource name

For example, you could replace "Please enter a valid credit card number" with r"ValidCCNumberMsg" and then add the following line to the resource file:

ValidCCNumberMsg=Please enter a valid credit card number

To assist the translator you can add a comment line before or after each text resource value and describe the usage and/or context. For example,

# Program: CCVALIDATE
# Context: Message displayed on line 24 when user has entered an invalid credit card number
ValidCCNumberMsg=Please enter a valid credit card number

The name of the language resource file is constructed from the values of the following framework properties:

iscobol.resource.file
iscobol.resource.country
iscobol.resource.language
iscobol.resource.variant

as follows (square brackets enclose optional elements):

file[_language[_country[variant]]].properties

For example, if iscobol.resource.file=RES and iscobol.resource.language=en then the isCOBOL Runtime Framework will attempt to load a file named RES_en.properties.

The resource file must be located in a directory or jar file listed in the class path.

The above framework properties can be set programmatically using SET ENVIRONMENT leaving off the "iscobol." prefix.

An example program which demonstrates this feature is provided in the isCOBOL/sample/multilanguage directory.

While reading property files Java uses the ISO 8859-1 encoding, so the resource files must be written with this encoding. To insert characters that cannot be represented by the ISO encoding, Unicode escape sequences must be used. The escape sequence is composed by \u followed by the utf-16 representation of the character (the same you would use with nx syntax in the cobol source).

You can use the native2ascii utility, which is included with the JDK, to convert localized text files into ISO 8859-1 compliant Unicode escape sequences. The documentation for native2ascii is at:

http://java.sun.com/javase/6/docs/technotes/tools/windows/native2ascii.html

To translate your resource file into another language, you can use any word processor or editor that can accomplish the job. Save the file as a text file using whatever encoding is convenient. Then run native2ascii to convert the text file to a Java properties file in the correct format (i.e. ISO 8859-1). The native2ascii utility supports all of the commonly used encodings.

The supported encodings are listed at:

http://java.sun.com/javase/6/docs/technotes/guides/intl/encoding.doc.html

You might find Google Translate to be a useful tool during development and testing:

http://translate.google.com

For example, at http://translate.google.com/translate_t?sl=en&tl=zh-CN, type the following into the entry-field:

Chinese
Username
Type username
Password
Type password
Authentication
Invalid username/password, login failed

Then press the Translate button
Copy the Chinese values from the Translation field into Wordpad
Save the Wordpad document as "Unicode Text Document" named chinese.txt
Execute the following command from the directory containing chinese.txt:

native2ascii -encoding utf16 chinese.txt

This produces the following output:

\u4e2d\u56fd
\u7528\u6237\u540d
\u8f93\u5165\u7528\u6237\u540d
\u5bc6\u7801
\u8f93\u5165\u5bc6\u7801
\u9a8c\u8bc1
\u65e0\u6548\u7684\u7528\u6237\u540d/\u5bc6\u7801\uff0c\u767b\u5f55\u5931\u8d25

Copy these entries into a resource property file as follows:

chinese=\u4e2d\u56fd
usr=\u7528\u6237\u540d:
usrhint=\u8f93\u5165\u7528\u6237\u540d
pwd=\u5bc6\u7801:
pwdhint=\u8f93\u5165\u5bc6\u7801
aut=\u9a8c\u8bc1
wrong=\u65e0\u6548\u7684\u7528\u6237\u540d/\u5bc6\u7801\uff0c\u767b\u5f55\u5931\u8d25

Alternatively, you can copy the text into Microsoft Word. Then save the Word document as a Plain text file. Then in the file conversion dialog box, select "Other encoding", select "Unicode (Big-Endian)" and press OK.

As mentioned earlier, you can use other encodings. For example, if you have a text file in Big5 encoding (i.e. Traditional Chinese) then you can convert it using "native2ascii -encoding Big5"