Multilingual Support of Web Applications using Server Side Java

Surekha Sastry
Indic- Sarai Fellow
Servelots Infotech
surekha@servelots.com

K Srinivasa Raghavan
Indic-Sarai Felllow
Servelots Infotech
raghavan@servelots.com

Multi-lingual support

Whether it be e-mail or a project like PANTOTO, internationalization [i18n] allows the application to work in a specific local language. Multi-lingual applications however, go a step further and enable the user to switch locales (language) dynamically at run-time.

Issues

The issues involved in providing multilingual support for web applications are

1. *Data Storage-How is the data stored at the back-end (ISCII or Unicode)?*

    - Data can be stored at the back-end either in ISCII or Unicode encoding schemes. Both these encoding schemes have their own merits and demerits.
    - The common code representation for all Indian Languages and the disk space for storing each character is less in ISCII encoding scheme.
    - Unicode uses two bytes to store a single character.
    - With regard to collation, ISCII does not sort all Indian Language characters properly, whereas Unicode has a different collation table for each Indian Language and thus sorting is not a problem in this encoding scheme.

    We need to analyze which encoding scheme would be most appropriate with PANTOTO[1] which is our test bed, and we will start this analysis storing data in ISCII.

2. *Display issues: How is data presented to the user?*

    - Since most commonly used browsers are Unicode enabled and all the Indian Language scripts can be represented in Unicode, the data can be presented to the user in Unicode[2] at the display end.
    - For displaying, we can make use of an Open Type Font.
    - Mozilla and Netscape do not have proper *Rendering*[3] mechanism to display Indian scripts. For the time being we have to tolerate this rendering. But Internet Explorer has a rendering engine called ***Uniscribe***[4] which does this properly when an Open Type Font is used for display.
    - If we store data in ISCII in the back-end, then there is a need for conversion between ISCII and Unicode while the data travels between the front-end and the back-end.
    - We have developed a tool[5] that can be used for this conversion. The current version of the converter has conversion support for Indian Languages based

[1] http://www.Pantoto.com
[2] If we store data at the back-end in ISCII, then we need to convert data between ISCII and Unicode.
[3] Rendering is the process of correctly placing the font glyphs so that it will not violate the display rules of that particular Indian Language script. In other words, rendering is the process of ordering the font character glyphs so that they stick to the display order rules of that particular Indian Language script.
[4] http://www.microsoft.com
[5] Conversion tool is available at http://www.sarovar.org/projects/codeconverters

on “Devanagari Script” and the language “Telugu”. We require adding conversion routines for other Indian Languages.

3. *Input Mechanism: How to provide support to key in data into the web application in Indian languages?*

    - There are many Input Method Editors[6] (IME) available, developed in Java, and can be used if an applet is embedded into a web page, provided the IME is available at the client end along with Java Runtime Environment.
    - We propose to develop an IME that works in most of the popular browsers and across platforms.

Through the course of our project we write a technical document along with some APIs, that will address the above three issues. Based on our experience and research we believe that this line of work will contribute to the existing body of knowledge on localization and multilingual support.

[6] An Input Method Editor (IME) is an input filter that maps the keystrokes of the qwerty keyboard to the required characters or symbols.