Multilingual Support

Displaying Unicode Characters Where Client and Server are Running the Same Language

ExcelWriter can be used to generate Excel spreadsheets which contain Unicode characters.

When the client and server are running the same language, to correctly display Unicode characters in an Excel spreadsheet, set the charset property in the script that will display the spreadsheet to either:

- UTF-8, to correctly display characters in any language
  OR
- A specific language charset - like GB2312 for Simplified Chinese - to correctly display characters in a single language

For a complete list of charset values, see Character Set Recognition.

Displaying Non-Latin Alphabet Strings Where

Client and Server are not Running the Same Language

The ExcelWriter methods AnsiToUnicode and UnicodeToAnsi allow you to correctly display non-Latin alphabet strings, both when getting a value from a spreadsheet and displaying it in the browser, and when inserting a string from the client in a spreadsheet generated on the server. AnsiToUnicode and UnicodeToAnsi are methods of both ExcelApplication and ExcelTemplate.

AnsiToUnicode

If a client submits a non-Latin alphabet string to a server running a language different from the language of the client's HTML page, to display the string correctly in an Excel spreadsheet the string must be converted to Unicode. ExcelWriter's AnsiToUnicode method takes a non-Latin alphabet string and its language's code page and returns a Unicode string:

```
UnicodeString = AnsiToUnicode(AnsiString, CodePage)
```

To correctly display a non-Latin alphabet string in your ExcelWriter generated spreadsheet where the server's language and the string language are different:

1. Set the client-side HTML script's charset property to the string language's charset. For example, to set the charset to Hebrew, use:

   ```
   <META HTTP-EQUIV="content-type" CONTENT="text/html;charset=windows-1255">
   ```

2. Use AnsiToUnicode to convert the string to Unicode. The following example gets a Hebrew string from an HTML form, converts the string to Unicode, and assign the string to cell E7:

   ```
   Set xlw = Server.CreateObject("Softartisans.ExcelWriter")
   HebrewString = xlw.ansitounicode(Request.Form("FirstName"), 1255)
   ws.cells("E7").value = HebrewString
   %>
   ```

Note: If your ExcelWriter script will receive strings from the client in only one language, use that language's code page and charset. However, to correctly display values in any language, use the UTF-8 charset and 65001 code page.

For a complete list of charset values, see, Character Set Recognition.

UnicodeToAnsi

To get a non-Latin alphabet string from a spreadsheet, and display the string correctly in the browser from a server running a language different from the language of the client's HTML page, you must:

1. Set the ExcelWriter script's **charset** property to the string language's charset.

2. Use the UnicodeToAnsi method to convert the string to an Ansi encoded string that can be displayed correctly to the user.

UnicodeToAnsi takes a Unicode string and the string language's code page and returns an Ansi string:

```
AnsiString = UnicodeToAnsi(UnicodeString, CodePage)
```

In the following example, Cell E7 contains a Hebrew string represented in Unicode. The script converts the string from Unicode to Ansi, and displays the converted string in the browser. The Hebrew characters will be displayed correctly.

```
<META HTTP-EQUIV="content-type" CONTENT="text/html;charset=windows-1255">
<%
Set xlw = Server.CreateObject("Softartisans.ExcelWriter")...
response.write xlw.unicodetoansi(order.cells("E7").value, 1255)
```

Note: If your ExcelWriter script will get and display strings only in one language, use that language's code page and charset. However, to correctly display values in any language, use the UTF-8 charset and 65001 code page.

For a complete list of charset values, see, Character Set Recognition.