crystal decisions ™

Crystal Enterprise

Using Crystal Enterprise in Multiple Language Environments

Overview

Crystal Enterprise and Crystal Reports provide powerful capabilities for presenting data in a varied number of languages. This whitepaper gives some guidance on the various techniques that can be used to create one report that is delivered in multiple languages. This whitepaper assumes the user has advanced report design skills. The recommendations given in this whitepaper have been tested with Crystal Report 8.5 and Crystal Enterprise 8.0.

## Contents

Report Design Techniques

When considering a report that must represent data in multiple languages, there are two key considerations that must be made:

1. Can the languages being displayed use the same font (like French and English) or do they require different fonts (such as English and Russian)?

2. Does the whole report (both data and labels) need to be presented in multiple languages, or can the labels be presented in one language?

Using Multiple Fonts in a Report

If a report contains languages that require different fonts, special formatting must be added to the report. In fact, conditional formatting must be applied to every field that requires font changes. In order to apply this formatting, right-click on the field that needs to be modified, choose the "Format Field" option, and then click on the "Font" tab. Figure 1 shows what should be displayed.

Figure 1
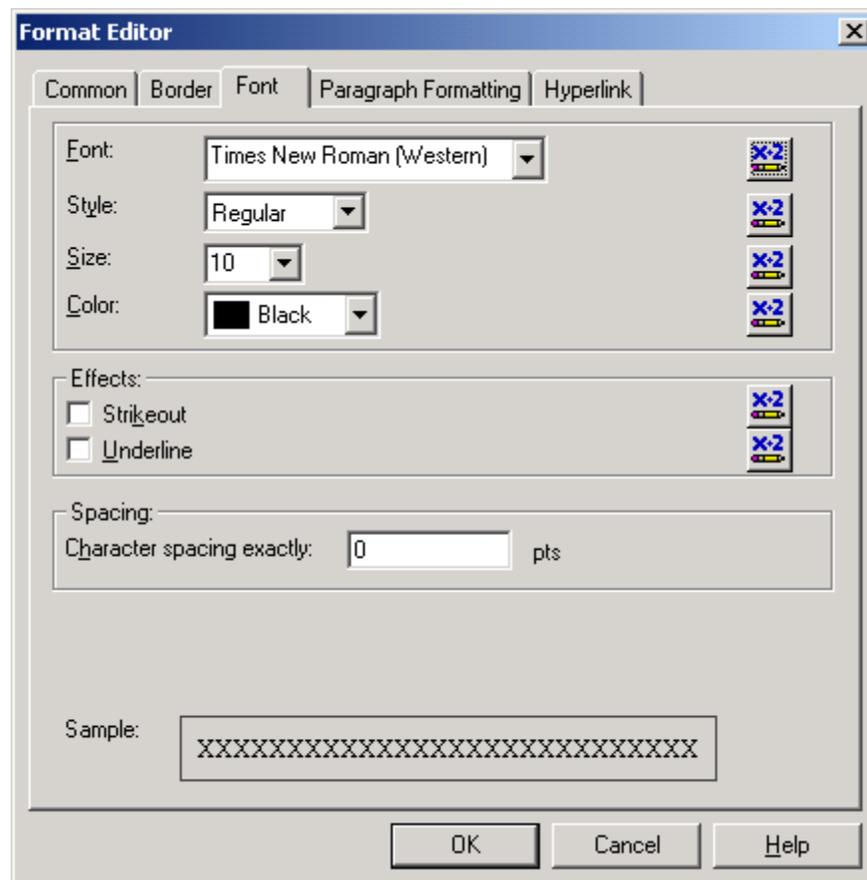
At this point, click on the conditional formatting button. This will bring up the Formula Editor. Within the Formula Editor, create a formula to change the font. Usually, this is most easily done

ce8_multi_language.pdf

through the use of a parameter field. Depending on the value of the parameter field, the appropriate font can be used. Figure 2 shows an example of this.

NOTE	Each field must have the conditional formatting applied to it manually. There is no way to apply the conditional formatting globally to all fields.

Figure 2
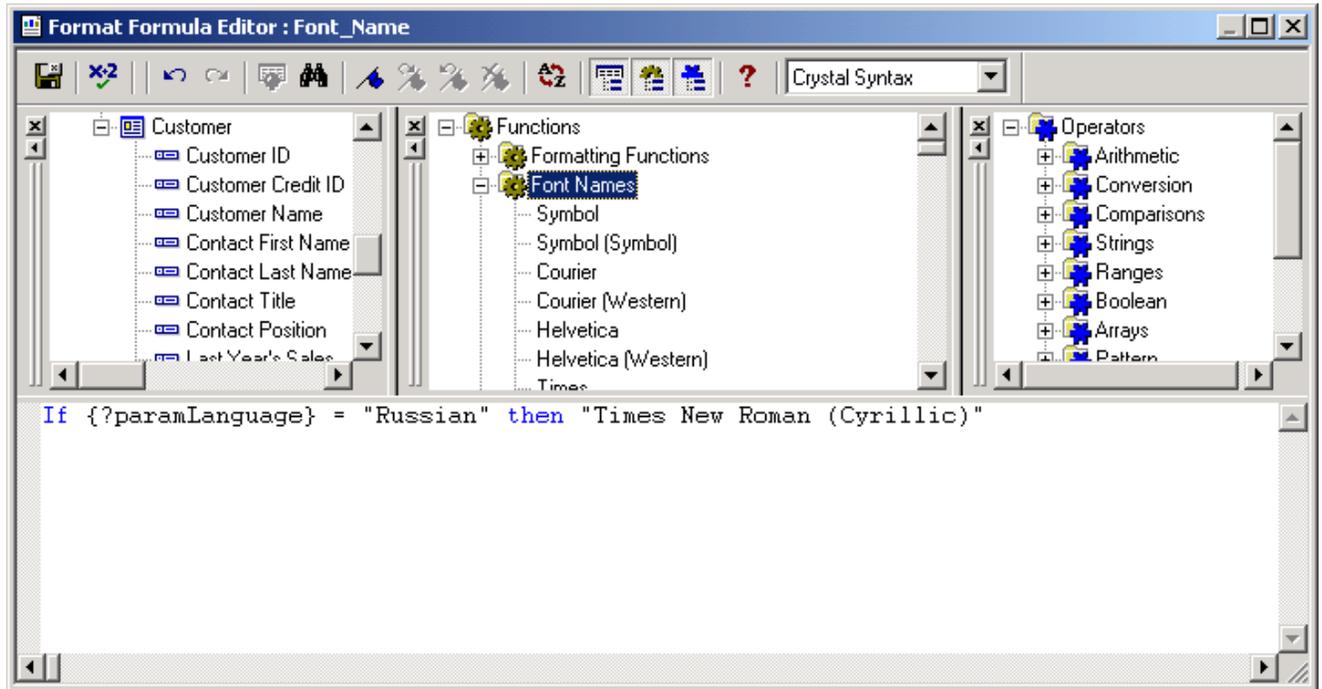
Using Multiple Language Labels in a Report

In reports where it is necessary to have labels in the same language as the data, some methods must be devised for obtaining the different label data. Here are a few methods, but there could be other variations of these methods.

Multiple Text Objects

The simplest way to support multi-language labels is to have a separate text object for each label. The labels can be placed on top of one another, and then conditionally suppressed to display only the appropriate label. This technique works well for simple reports, with only 2 or 3 languages and a small number of labels.

Shared Variables in Subreports

For more complex reports with numerous languages and a large number of labels, it is advantageous to store the label information in a database and then retrieve this information into the report using a subreport. This subreport can be placed in the Report Header, and then using shared variables, the values can be passed to the main report as labels.

User Function Library

A user function library (UFL) is useful for situations where a large number of reports need to be designed that contain multi-lingual data. A UFL has the advantage of being very easy to implement in a report, since it is used in the exact same manner as built-in functions. Since, at its core, it is a DLL, varying degrees of business logic can be coded into the function to shield users from additional complexity. The downside to using a UFL is that it requires a considerable amount of work to plan, code, test, and implement.

Crystal Enterprise Server Considerations

Once a report has been tested and works properly in the Report Designer, the report must be published into the Enterprise framework so that it can be run in an on-demand fashion (by the Page Server) or in a scheduled fashion (by the Job Server). Thus, the Page Server and Job Server require that the operating system support the language that is stored in the report.

There are two considerations that need to be made with regards to what the operating system provides:

1. The operating system needs the code page(s) of the language (see Appendix A for more discussion of code pages). There are separate code pages for different writing systems, such as Western European and Cyrillic.

2. The operating system needs a font to display the characters supported by this code page.

In practice, both code pages and fonts can be installed at the same time. It depends on the language and the operating system. Since the Crystal Enterprise server components are currently restricted to Windows NT-based platforms, those are the ones listed here. Windows 2000 is recommended as it has better multi-language support than Windows NT 4.

Supporting Multi-Language Development

If reports are to be designed on an international O/S and the servers run on an English O/S, there are special considerations that need to be accounted for. The English O/S servers must, of course, have the appropriate language installed.

However, depending on the languages used, not all data will be rendered correctly. For example, if using a double-byte language like Japanese, the following will not display correctly when trying to render Kanji characters:

3. Strings in Formulas

4. Chart titles

5. Group tree values

6. Drill-down tabs

This is because these strings use the system font to render their text. Therefore, unless the system font is in the appropriate language, Crystal Enterprise will not render these strings correctly.

As well, most export formats (from the Viewers) do not support exporting double-byte characters on an English O/S. Exporting to RTF works correctly, but exporting to Microsoft Word, Excel, and PDF will not render any double-byte characters correctly.

Debugging Techniques

If, after installing the necessary fonts on to the various servers, Crystal Enterprise does not render the report correctly, install Crystal Reports on to the problematic servers. Open up the problematic report and refresh it. This usually indicates where the problems are.

Crystal Enterprise Web Front-end Considerations

To support multiple languages from a Crystal Enterprise web front-end, there are two options.

Multiple Sites

Every Crystal Enterprise web desktop (located in \Program Files\Seagate Software\Web Content) comes with all strings set as constants in the beginning of the CSP file. Modify these strings to produce the appropriate output language. To handle multiple languages, duplicate the files into different subdirectories, having one language for each subdirectory

CSP

Since CSP is handled at the server, some logic can be coded into the CSP to handle the multiple languages. For example, a cookie could be written onto the user's machine that indicated their desired language. Then, based on the value of this cookie, the CSP would set the strings to the appropriate language.

Appendix A – Code Pages

Typically, when storing text, programs use a single byte to represent a character. This byte is then interpreted as a code point, a value between 0 and 255. For example, a byte with the value 66 represents the capital letter *B* in English. Together, all of these characters form a code page. A single-byte code page contains a maximum of 256 bytes; because each character in the code page is represented by a single byte, a code page can contain as many as 256 characters. One code page with its limit of 256 characters cannot accommodate all languages because some languages use far more than 256 characters. Therefore, different scripts use separate code pages. There is one code page for Greek, another for Cyrillic, and so on.

For example, the code point 230 can have several different values. It might be the Greek lowercase zeta (ζ), the Cyrillic lowercase zhe (ж), or the Western European diphthong (æ). All three characters have the same code point (230), but based on the code page (1253, 1251, and 1252, respectively), the value of the text will be different.

Obviously, single-byte code pages cannot accommodate Asian languages, which commonly use more than 5,000 characters. Double-byte or multi-byte code pages were developed to support these languages. Crystal Reports and Crystal Enterprise use multi-byte code pages, or Multi-Byte Character Strings (MBCS) to encode their text.

As opposed to MBCS, Unicode was developed to create a universal character set that can accommodate all known scripts. Unicode uses a unique, two-byte encoding for every character (currently more than 40,000); so in contrast to code pages, every character has its own unique code point. For example, the Unicode code point of lowercase zeta (ζ) is the hexadecimal value 03B6, lowercase zhe (ж) is 0436, and the diphthong (æ) is 00E6.

Currently, Crystal Reports is not fully Unicode-enabled.

Contacting Crystal Decisions for Technical Support

We recommend that you refer to the product documentation and that you visit our Technical Support web site for more resources.

Self-serve Support:

http://support.crystaldecisions.com/

Email Support:

http://support.crystaldecisions.com/support/answers.asp

Telephone Support:

http://www.crystaldecisions.com/contact/support.asp

****