#### rFonts (Run Fonts)

This element specifies the [fonts](fonts.docx) which shall be used to display the text contents of this run. Within a single run, there may be up to four types of content present which shall each be allowed to use a unique font:

* ASCII
* High ANSI
* Complex Script
* East Asian

The use of each of these [fonts](fonts.docx) shall be determined by the Unicode character values of the run content, unless manually overridden via use of the [cs](cs.docx) element (§**Error! Reference source not found.**).

If this element is not present, the default value is to leave the formatting applied at previous level in the style hierarchy. If this element is never applied in the style hierarchy, then the text shall be displayed in any default font which supports each [type](type.docx) of content.

[Example: Consider a single text run with both Arabic and English text, as follows:

English العربية

This content may be expressed in a single WordprocessingML run:

<w:[r](r.docx)>  
 <w:[t](t.docx)>English العربية</w:[t](t.docx)>  
</w:[r](r.docx)>

Although it is in the same run, the contents are in different font faces by specifying a different font for ASCII and CS characters in the run:

<w:[r](r.docx)>  
 <w:[rPr](rPr.docx)>  
 <w:rFonts w:ascii="Courier New" w:[cs](cs.docx)="Times New Roman" />  
 </w:[rPr](rPr.docx)>  
 <w:[t](t.docx)>English العربية</w:[t](t.docx)>  
</w:[r](r.docx)>

This text run shall therefore use the Courier New font for all characters in the ASCII range, and shall use the Times New Roman font for all characters in the Complex Script range. end example]

|  |
| --- |
| Parent Elements |
| [rPr](rPr.docx) (§**Error! Reference source not found.**); [rPr](rPr.docx) (§**Error! Reference source not found.**); [rPr](rPr.docx) (§**Error! Reference source not found.**); [rPr](rPr.docx) (§**Error! Reference source not found.**); [rPr](rPr.docx) (§**Error! Reference source not found.**); [rPr](rPr.docx) (§**Error! Reference source not found.**); [rPr](rPr.docx) (§**Error! Reference source not found.**); [rPr](rPr.docx) (§**Error! Reference source not found.**); [rPr](rPr.docx) (§**Error! Reference source not found.**); [rPr](rPr.docx) (§**Error! Reference source not found.**) |

|  |  |
| --- | --- |
| Attributes | Description |
| ascii (ASCII Font) | Specifies a font which shall be used to [format](format.docx) all characters in the ASCII range (0 - 127) within the parent run.  If the asciiTheme attribute is also specified \*in the run properties\*., then this attribute shall be ignored and that value shall be used instead.  If this attribute is not present, the default value is to leave the formatting applied at previous level in the style hierarchy. If this attribute is never applied in the style hierarchy, then the text shall be displayed in any default font which supports ASCII content.  [Example: Consider a run of ASCII text which shall be displayed using the Courier New font. This requirement would be specified as follows in the resulting WordprocessingML:  <w:[rPr](rPr.docx)>  <w:rFonts w:ascii=”Courier New” />  </w:[rPr](rPr.docx)>  The ascii attribute specifies that the run shall use the Courier New font for all text in the ASCII range. end example]  The possible values for this attribute are defined by the [ST\_String](ST_String.docx) simple [type](type.docx) (§**Error! Reference source not found.**). |
| asciiTheme (ASCII Theme Font) | Specifies a theme font which shall be used to [format](format.docx) all characters in the ASCII range (0 - 127) within the parent run. This theme font is a reference to one of the predefined theme [fonts](fonts.docx), located in the document's Theme part,which allows for font information to be set centrally in the document.  If the ascii attribute is also specified, then that attribute shall be ignored and this value shall be used instead.  If this attribute is not present, the default value is to leave the formatting applied at previous level in the style hierarchy. If this attribute is never applied in the style hierarchy, then the text shall be displayed in the font specified by the ascii attribute.  [Example: Consider a run of ASCII text which shall be displayed using the majorASCII theme font. This requirement would be specified as follows in the resulting WordprocessingML:  <w:[rPr](rPr.docx)>  <w:rFonts w:asciiTheme=”majorAscii” />  </w:[rPr](rPr.docx)>  The ascii attribute specifies that the run shall use the majorAscii theme font as defined in the document's themes part for all text in the ASCII range. end example]  The possible values for this attribute are defined by the [ST\_Theme](ST_Theme.docx) simple [type](type.docx) (§**Error! Reference source not found.**). |
| [cs](cs.docx) (Complex Script Font) | Specifies a font which shall be used to [format](format.docx) all characters in a complex script Unicode range within the parent run.  If the csTheme attribute is also specified, then this attribute shall be ignored and that value shall be used instead.  If this attribute is not present, the default value is to leave the formatting applied at previous level in the style hierarchy. If this attribute is never applied in the style hierarchy, then the text shall be displayed in any default font which supports complex script content.  [Example: Consider a run of Arabic text which shall be displayed using the Arial Unicode MS font. This requirement would be specified as follows in the resulting WordprocessingML:  <w:[rPr](rPr.docx)>  <w:rFonts w:[cs](cs.docx)=”Arial Unicode MS” />  </w:[rPr](rPr.docx)>  The [cs](cs.docx) attribute specifies that the run shall use the Arial Unicode MS font for all text in a complex script range. end example]  The possible values for this attribute are defined by the [ST\_String](ST_String.docx) simple [type](type.docx) (§**Error! Reference source not found.**). |
| cstheme (Complex Script Theme Font) | Specifies a theme font which shall be used to [format](format.docx) all characters in a complex script Unicode range within the parent run. This theme font is a reference to one of the predefined theme [fonts](fonts.docx), located in the document's Theme part,which allows for font information to be set centrally in the document.  If the [cs](cs.docx) attribute is also specified, then that attribute shall be ignored and this value shall be used instead.  If this attribute is not present, the default value is to leave the formatting applied at previous level in the style hierarchy. If this attribute is never applied in the style hierarchy, then the text shall be displayed in the font specified by the [cs](cs.docx) attribute.  [Example: Consider a run of Arabic text which shall be displayed using the majorBidi theme font. This requirement would be specified as follows in the resulting WordprocessingML:  <w:[rPr](rPr.docx)>  <w:rFonts w:csTheme=”majorBidi” />  </w:[rPr](rPr.docx)>  The csTheme attribute specifies that the run shall use the majorBidi theme font as defined in the document's themes part for all text in a complex script range. end example]  The possible values for this attribute are defined by the [ST\_Theme](ST_Theme.docx) simple [type](type.docx) (§**Error! Reference source not found.**). |
| eastAsia (East Asian Font) | Specifies a font which shall be used to [format](format.docx) all characters in an East Asian Unicode range within the parent run.  If the eastAsiaTheme attribute is also specified, then this attribute shall be ignored and that value shall be used instead.  If this attribute is not present, the default value is to leave the formatting applied at previous level in the style hierarchy. If this attribute is never applied in the style hierarchy, then the text shall be displayed in any default font which supports East Asian content.  [Example: Consider a run of Japanese text which shall be displayed using the MS Mincho font. This requirement would be specified as follows in the resulting WordprocessingML:  <w:[rPr](rPr.docx)>  <w:rFonts w:eastAsia=”MS Mincho” /> </w:[rPr](rPr.docx)>  The eastAsia attribute specifies that the run shall use the MS Mincho font for all text in an East Asian range. end example]  The possible values for this attribute are defined by the [ST\_String](ST_String.docx) simple [type](type.docx) (§**Error! Reference source not found.**). |
| eastAsiaTheme (East Asian Theme Font) | Specifies a theme font which shall be used to [format](format.docx) all characters in an East Asian Unicode range within the parent run. This theme font is a reference to one of the predefined theme [fonts](fonts.docx), located in the document's Theme part,which allows for font information to be set centrally in the document.  If the eastAsia attribute is also specified, then that attribute shall be ignored and this value shall be used instead.  If this attribute is not present, the default value is to leave the formatting applied at previous level in the style hierarchy. If this attribute is never applied in the style hierarchy, then the text shall be displayed in the font specified by the eastAsia attribute.  [Example: Consider a run of Japanese text which shall be displayed using the minorEastAsia theme font. This requirement would be specified as follows in the resulting WordprocessingML:  <w:[rPr](rPr.docx)>  <w:rFonts w:eastAsiaTheme=”minorEastAsia” />  </w:[rPr](rPr.docx)>  The eastAsiaTheme attribute specifies that the run shall use the minorEastAsia theme font as defined in the document's themes part for all text in an East Asian range. end example]  The possible values for this attribute are defined by the [ST\_Theme](ST_Theme.docx) simple [type](type.docx) (§**Error! Reference source not found.**). |
| hAnsi (High ANSI Font) | Specifies a font which shall be used to [format](format.docx) all characters in a Unicode range within the parent run which does not fall into one of the three categories defined above, which is called the high ANSI range in WordprocessingML.  If the hAnsiTheme attribute is also specified, then this attribute shall be ignored and that value shall be used instead.  If this attribute is not present, the default value is to leave the formatting applied at previous level in the style hierarchy. If this attribute is never applied in the style hierarchy, then the text shall be displayed in any default font which supports high ANSI content.  [Example: Consider a run of text which falls into a high ANSI range, and shall be displayed using the Bauhaus 93 font. This requirement would be specified as follows in the resulting WordprocessingML:  <w:[rPr](rPr.docx)>  <w:rFonts w:hAnsi=”Bauhaus 93” /> </w:[rPr](rPr.docx)>  The hAnsi attribute specifies that the run shall use the Bauhaus 93 font for all text in a high ANSI range. end example]  The possible values for this attribute are defined by the [ST\_String](ST_String.docx) simple [type](type.docx) (§**Error! Reference source not found.**). |
| hAnsiTheme (High ANSI Theme Font) | Specifies a theme font which shall be used to [format](format.docx) all characters in a Unicode range within the parent run which does not fall into one of the three categories defined above, which is called the high ANSI range in WordprocessingML. This theme font is a reference to one of the predefined theme [fonts](fonts.docx), located in the document's Theme part,which allows for font information to be set centrally in the document.  If the hAnsi attribute is also specified, then that attribute shall be ignored and this value shall be used instead.  If this attribute is not present, the default value is to leave the formatting applied at previous level in the style hierarchy. If this attribute is never applied in the style hierarchy, then the text shall be displayed in the font specified by the hAnsi attribute.  [Example: Consider a run of text which falls into a high ANSI range, and shall be displayed using the minorHAnsi theme font. This requirement would be specified as follows in the resulting WordprocessingML:  <w:[rPr](rPr.docx)>  <w:rFonts w:hAnsiTheme="minorHAnsi" />  </w:[rPr](rPr.docx)>  The hAnsiTheme attribute specifies that the run shall use the minorHAnsi theme font as defined in the document's themes part for all text in a high ANSI range. end example]  The possible values for this attribute are defined by the [ST\_Theme](ST_Theme.docx) simple [type](type.docx) (§**Error! Reference source not found.**). |
| hint (Font Content Type) | Specifies the font [type](type.docx) which shall be used to [format](format.docx) any ambiguous characters in the current run.  There are certain characters which are not explicitly stored in the document, and may be mapped into multiple categories of the four mentioned above. This attribute shall be used to arbitrate that conflict, and determine how ambiguities in this run shall be handled. [Note: This is primarily used to handle the formatting on the paragraph mark glyph, and other characters that are not stored as text in the WordprocessingML document. end note]  If this attribute is omitted, then this ambiguity may be resolved by any means available.  [Example: Consider the run representing the paragraph mark glyph, which is not stored as a physical character. Since this could therefore be formatted with any of the [fonts](fonts.docx) specified for the run, this ambiguity is resolved using the following WordprocessingML:  <w:[pPr](pPr.docx)>  <w:[rPr](rPr.docx)>  <w:rFonts w:hint="eastAsia" />  </w:[rPr](rPr.docx)> </w:[pPr](pPr.docx)>  The hint attribute specifies that the run shall use the eastAsia font (theme or not, whichever is in use for East Asian text) as defined for this range. end example]  The possible values for this attribute are defined by the [ST\_Hint](ST_Hint.docx) simple [type](type.docx) (§**Error! Reference source not found.**). |

# 2.1.87 Part 1 Section 17.3.2.26, rFonts (Run Fonts)

a. The standard does not limit the length of the ascii, hAnsi, cs, and eastAsia attributes.

Word restricts the value of these attributes to be less than 32 characters

b. The standard states that the use of each of the run fonts shall be determined by the Unicode character values of the run content, but does not explain precisely how this is done.

Word determines the use of each of these fonts from the Unicode character values in the following manner:

If the run has the **cs** element ("[[ISO/IEC-29500-1]](http://go.microsoft.com/fwlink/?LinkId=132464) §17.3.2.7; cs") or the **rtl** element ("[[ISO/IEC-29500-1]](http://go.microsoft.com/fwlink/?LinkId=132464) §17.3.2.30; rtl"), then the **cs** (or **cstheme** if defined) font is used, regardless of the Unicode character values of the run’s content.

If the **eastAsia** (or **eastAsiaTheme** if defined) attribute’s value is “Times New Roman” and the **ascii** (or **asciiTheme** if defined) and **hAnsi** (or**hAnsiTheme** if defined) attributes are equal, then the **ascii** (or **asciiTheme** if defined) font is used.

Otherwise, the following table is used. For all ranges not listed in the following table, the **hAnsi** (or **hAnsiTheme** if defined) font shall be used.

|  |  |  |
| --- | --- | --- |
| Unicode Block | Range | Classification |
| **Basic Latin** | 0000 – 007F | **ascii** (or **asciiTheme** if defined) |
| **Latin-1 Supplement** | 00A0 – 00FF | **hAnsi** (or **hAnsiTheme** if defined), with the following exceptions:  If **hint** is eastAsia, the following characters use **eastAsia** (or **eastAsiaTheme** if defined): A1, A4, A7 – A8, AA, AD, AF, B0 – B4, B6 – BA, BC – BF, D7, F7  If **hint** is eastAsia and the language of the run is either Chinese Traditional or Chinese Simplified, the following characters use **eastAsia** (or **eastAsiaTheme** if defined): E0 – E1, E8 – EA, EC – ED, F2 – F3, F9 – FA, FC |
| **Latin Extended-A** | 0100 – 017F | **hAnsi** (or **hAnsiTheme** if defined), with the following exception:  If **hint** is eastAsia, and the language of the run is either Chinese Traditional or Chinese Simplified, or the character set of the **eastAsia** (or **eastAsiaTheme** if defined) font is Chinese5 or GB2312 then **eastAsia** (or **eastAsiaTheme**if defined) font is used. |
| **Latin Extended-B** | 0180 – 024F | **hAnsi** (or **hAnsiTheme** if defined), with the following exception:  If **hint** is eastAsia, and the language of the run is either Chinese Traditional or Chinese Simplified, or the character set of the **eastAsia** (or **eastAsiaTheme** if defined) font is Chinese5 or GB2312 then **eastAsia** (or **eastAsiaTheme**if defined) font is used. |
| **IPA Extensions** | 0250 – 02AF | **hAnsi** (or **hAnsiTheme** if defined), with the following exception:  If **hint** is eastAsia, and the language of the run is either Chinese Traditional or Chinese Simplified, or the character set of the **eastAsia** (or **eastAsiaTheme** if defined) font is Chinese5 or GB2312 then **eastAsia** (or **eastAsiaTheme**if defined) font is used. |
| **Spacing Modifier Letters** | 02B0 – 02FF | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Combining Diacritical Marks** | 0300 – 036F | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Greek** | 0370 – 03CF | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Cyrillic** | 0400 – 04FF | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Hebrew** | 0590 – 05FF | **ascii** (or **asciiTheme** if defined) |
| **Arabic** | 0600 – 06FF | **ascii** (or **asciiTheme** if defined) |
| **Syriac** | 0700 – 074F | **ascii** (or **asciiTheme** if defined) |
| **Arabic Supplement** | 0750 – 077F | **ascii** (or **asciiTheme** if defined) |
| **Thaana** | 0780 – 07BF | **ascii** (or **asciiTheme** if defined) |
| **Hangul Jamo** | 1100 – 11FF | **eastAsia** (or **eastAsiaTheme** if defined) |
| **Latin Extended Additional** | 1E00 – 1EFF | **hAnsi** (or **hAnsiTheme** if defined), with the following exception:  If the **hint** is eastAsia and the language of the run is Chinese Traditional or Chinese Simplified then **eastAsia** (or**eastAsiaTheme** if defined) is used. |
| **General Punctuation** | 2000 – 206F | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Superscripts and Subscripts** | 2070 – 209F | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Currency Symbols** | 20A0 – 20CF | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Combining Diacritical Marks for Symbols** | 20D0 – 20FF | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Letter-like Symbols** | 2100 – 214F | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Number Forms** | 2150 – 218F | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Arrows** | 2190 – 21FF | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Mathematical Operators** | 2200 – 22FF | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Miscellaneous Technical** | 2300 – 23FF | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Control Pictures** | 2400 – 243F | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Optical Character Recognition** | 2440 – 245F | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Enclosed Alphanumerics** | 2460 – 24FF | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Box Drawing** | 2500 – 257F | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Block Elements** | 2580 – 259F | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Geometric Shapes** | 25A0 – 25FF | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Miscellaneous Symbols** | 2600 – 26FF | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Dingbats** | 2700 – 27BF | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **CJK Radicals Supplement** | 2E80 – 2EFF | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **Kangxi Radicals** | 2F00 – 2FDF | **eastAsia** (or **eastAsiaTheme** if defined) |
| **Ideographic Description Characters** | 2FF0 – 2FFF | **eastAsia** (or **eastAsiaTheme** if defined) |
| **CJK Symbols and Punctuation** | 3000 – 303F | **eastAsia** (or **eastAsiaTheme** if defined) |
| **Hiragana** | 3040 – 309F | **eastAsia** (or **eastAsiaTheme** if defined) |
| **Katakana** | 30A0 – 30FF | **eastAsia** (or **eastAsiaTheme** if defined) |
| **Bopomofo** | 3100 – 312F | **eastAsia** (or **eastAsiaTheme** if defined) |
| **Hangul Compatibility Jamo** | 3130 – 318F | **eastAsia** (or **eastAsiaTheme** if defined) |
| **Kanbun** | 3190 – 319F | **eastAsia** (or **eastAsiaTheme** if defined) |
| **Enclosed CJK Letters and Months** | 3200 – 32FF | **eastAsia** (or **eastAsiaTheme** if defined) |
| **CJK Compatibility** | 3300 – 33FF | **eastAsia** (or **eastAsiaTheme** if defined) |
| **CJK Unified Ideographs Extension A** | 3400 – 4DBF | **eastAsia** (or **eastAsiaTheme** if defined) |
| **CJK Unified Ideographs** | 4E00 – 9FAF | **eastAsia** (or **eastAsiaTheme** if defined) |
| **Yi Syllables** | A000 – A48F | **eastAsia** (or **eastAsiaTheme** if defined) |
| **Yi Radicals** | A490 – A4CF | **eastAsia** (or **eastAsiaTheme** if defined) |
| **Hangul Syllables** | AC00 – D7AF | **eastAsia** (or **eastAsiaTheme** if defined) |
| **High Surrogates** | D800 – DB7F | **eastAsia** (or **eastAsiaTheme** if defined) |
| **High Private Use Surrogates** | DB80 – DBFF | **eastAsia** (or **eastAsiaTheme** if defined) |
| **Low Surrogates** | DC00 – DFFF | **eastAsia** (or **eastAsiaTheme** if defined) |
| **Private Use Area** | E000 – F8FF | If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used. |
| **CJK Compatibility Ideographs** | F900 – FAFF | **eastAsia** (or **eastAsiaTheme** if defined) |
| **Alphabetic Presentation Forms** | FB00 – FB4F | **hAnsi** (or **hAnsiTheme** if defined), with the following exceptions:  If the **hint** is eastAsia then **eastAsia** (or **eastAsiaTheme** if defined) is used for characters in the range FB00 – FB1C.  For the range FB1D – FB4F, **ascii** (or **asciiTheme** if defined) is used. |
| **Arabic Presentation Forms-A** | FB50 – FDFF | **ascii** (or **asciiTheme** if defined) |
| **CJK Compatibility Forms** | FE30 – FE4F | **eastAsia** (or **eastAsiaTheme** if defined) |
| **Small Form Variants** | FE50 – FE6F | **eastAsia** (or **eastAsiaTheme** if defined) |
| **Arabic Presentation Forms-B** | FE70 – FEFE | **ascii** (or **asciiTheme** if defined) |
| **Halfwidth and Fullwidth Forms** | FF00 – FFEF | **eastAsia** (or **eastAsiaTheme** if defined) |

c. The standard states that the default fonts for the ascii, hAnsi, cs, and eastAsia attributes are application-defined.

Word uses a default font of Times New Roman for all of these attributes.

d. The standard states that all characters that are determined to be East Asian should use the value of eastAsia element as the font.

If the value of eastAsia attibute is "Times New Roman", and the value of the ascii (or asciiTheme) and the hAnsi (or hAnsiTheme) attributes are equal, then the value of the ascii (or asciiTheme, if defined) attribute is used as the font.

<http://social.msdn.microsoft.com/Forums/en-US/ef7b4b55-9d38-4cd8-89f0-38389419c672/determining-font-script-from-majorminorfont-and-lang-and-text-in-drawingml>

# Determining font script from major/minorFont and lang and text in DrawingML [http://i1.social.s-msft.com/globalresources/Images/trans.gif?cver=0%0d%0a](http://social.msdn.microsoft.com/Forums/en-US/os_openXML-ecma/thread/ef7b4b55-9d38-4cd8-89f0-38389419c672?outputAs=rss)

I can't quite figure out how to do something and need a fairly urgent reply. I don't think this is as much about implementation as it is lack of documentation. I need to determine what the mapping is for a font's script code/name (such as Jpan or Thai) below and the language tag (such as ja-JP or th-Th).

I have the following in my theme1.xml:

      <a:minorFont>

        <a:latin typeface="Segoe UI"/>

        <a:ea typeface=""/>

        <a:cs typeface=""/>

        <a:font script="Jpan" typeface="ＭＳ Ｐゴシック"/>

        <a:font script="Hang" typeface="맑은 고딕"/>

:

        <a:font script="Uigh" typeface="Microsoft Uighur"/>

        <a:font script="Geor" typeface="Sylfaen"/>

      </a:minorFont>

I have a run in simplified Chinese, as below which displays in the 宋体 (called "SimSun" in English):

            <a:r>

              <a:rPr lang="zh-CN" smtClean="0">

                <a:solidFill>

                  <a:schemeClr val="bg1"/>

                </a:solidFill>

              </a:rPr>

              <a:t>你好</a:t>

            </a:r>

            <a:r>

It displays correctly.

However, if I change the lang="zh-CN"  to lang="en-US", it displays it in my  <a:latintypeface="Segoe UI"/> font.

If I use Chinese with an English language attribute of lang="en-US", it still displays in the SimSun font.

            <a:r>

              <a:rPr lang="en-US" dirty="0" smtClean="0">

                <a:solidFill>

                  <a:schemeClr val="bg1"/>

                </a:solidFill>

              </a:rPr>

              <a:t>你好</a:t>

            </a:r>

So again, the question is how do lang ids/tags in runs map to script codes in minorFont/majorFont font elements in DrawingML. Also, is there something else in determining what font displays?

I'm well aware of RFC 4646/BCP 47 for the lang attribute. The script codes ("Hans, Ethi, etc.") appear to be from [ISO 15924](http://en.wikipedia.org/wiki/ISO_15924), but that is not explicitly stated - nor is any kind of mapping from RFC 4646 (or 4647) defined in any of the documentation.  20.1.4.1.16 of ISOIEC-29500 states under the script attribute: "Specifies the script, or language, in which the typeface is supposed to be used. The possible values for this attribute are defined by the W3C XML Schema string datatype." This just tells us it could be any string at all, like "gingersnap" - not that the actual values listed in the attribute are semantic and have a mapping.

It appears that WG4 has also noticed this is a problem and has started discussing it (see .doc on<http://mailman.vse.cz/pipermail/sc34wg4/2011-March/002204.html>) - but the focus appears to be on WordprocessingML, not DrawingML.

[Todd Main](http://social.msdn.microsoft.com/profile/todd%20main/?ws=usercard-mini)

165 Points

Thanks Tom.

I've done some research on what PowerPoint is implementing and wanted to share my results with you.

**Research Scenario:**

* 1. I created a .PPTM in PowerPoint 2010 (x64/SP1) and a VBA script which inserted a single textbox for every available MsoLanguageID (for example msoLanguageIDEdo, msoLanguageIDChineseHongKongSAR, etc.). The ".Text" string of the ".TextFrame.TextRange" was the string value of the MsoLanguageID (so inside the textbox would be the text "msoLanguageIDEdo", for example). I then ran a VBA script to find what font the TextBox was displaying in.
  2. After this, I looped through the .PPTX in Visual Studio and grabbed the "lang" value (and "alt-lang", if exists) of the TextBox.
  3. Then I compared the display fonts of the TextBoxes from <minorFont>.<font script="X"> values of my theme1.xml.
  4. From there, I manually compared all lang values (IETF language tags) against the IANA sub-tag registry on <http://www.iana.org/assignments/language-subtag-registry>. I believe this list is the same as ISO 15924.

I used a slightly different <minorFont> set this time, here it is:

      <a:minorFont>

        <a:latin typeface="Calibri"/>

        <a:ea typeface=""/>

        <a:cs typeface=""/>

        <a:font script="Jpan" typeface="ＭＳ Ｐゴシック"/>

        <a:font script="Hang" typeface="맑은 고딕"/>

        <a:font script="Hans" typeface="宋体"/>

        <a:font script="Hant" typeface="新細明體"/>

        <a:font script="Arab" typeface="Times New Roman"/> <!--used = yes-->

        <a:font script="Hebr" typeface="Arial"/> <!--used = yes-->

        <a:font script="Thai" typeface="Cordia New"/> <!--used = yes-->

        <a:font script="Ethi" typeface="Nyala"/>  <!--used = yes-->

        <a:font script="Beng" typeface="Vrinda"/> <!--used = yes-->

        <a:font script="Gujr" typeface="Shruti"/> <!--used = yes-->

        <a:font script="Khmr" typeface="DaunPenh"/> <!--used = yes-->

        <a:font script="Knda" typeface="Tunga"/> <!--used = yes-->

        <a:font script="Guru" typeface="Raavi"/> <!--used = yes-->

        <a:font script="Cans" typeface="Euphemia"/>  <!--used = yes-->

        <a:font script="Cher" typeface="Plantagenet Cherokee"/> <!--used = yes-->

        <a:font script="Yiii" typeface="Microsoft Yi Baiti"/>  <!--used in PowerPoint = NO-->

        <a:font script="Tibt" typeface="Microsoft Himalaya"/> <!--used = yes-->

        <a:font script="Thaa" typeface="MV Boli"/> <!--used = yes-->

        <a:font script="Deva" typeface="Mangal"/> <!--used = yes-->

        <a:font script="Telu" typeface="Gautami"/> <!--used = yes-->

        <a:font script="Taml" typeface="Latha"/> <!--used = yes-->

        <a:font script="Syrc" typeface="Estrangelo Edessa"/> <!--used = yes-->

        <a:font script="Orya" typeface="Kalinga"/> <!--used = yes-->

        <a:font script="Mlym" typeface="Kartika"/> <!--used = yes-->

        <a:font script="Laoo" typeface="DokChampa"/> <!--used = yes-->

        <a:font script="Sinh" typeface="Iskoola Pota"/> <!--used = yes-->

        <a:font script="Mong" typeface="Mongolian Baiti"/>  <!--used in PowerPoint  = NO-->

        <a:font script="Viet" typeface="Candara"/>  <!--used = yes-->

        <a:font script="Uigh" typeface="Microsoft Uighur"/>   <!--used in PowerPoint  = NO-->

      </a:minorFont>

**Conclusion:**

The results are in an Excel file on <http://dl.dropbox.com/u/16442383/font%20list.xlsx> for you to view. The conclusion is that "almost everything" maps to IANA sub tags.

* 1. Not everything though as there are non-IETF tags used in PowerPoint (or at least, non-consistent). Sometimes a three-letter tag is used. Sometimes a two-letter tag + a four-letter script are used.
  2. In one case, I was expecting the "Ethi" script to be used based off the language tag, but it wasn't.
  3. Not all scripts are used by (or available to) the PowerPoint client - in particular "Uigh", "Mong" and "Yiii".
  4. It appears that the display mechanism generally uses IANA script groupings, unless it is is an East Asian font, in which case it is a combination of the the lang attribute and Unicode.

My conclusion is that I believe I can figure out about 80-90% of what is going on. However, it would be great to have confirmation from you on the exact font selection algorithm and mapping.

----------

Thank you Tom for the reply! Note that I'm **not**asking about Word here, I'm asking about PowerPoint. Word appears to have richer features in determining which script to use, it's DrawingML that is my concern.

 Okatu,

Here's an overview of the algorithm from the PowerPoint team which may help a little:

Algorithm:  
   a. Call MSO APIs to convert a LCID to a script tag. It uses language or alt language (if the former is missing). And it requires a language and slot match; otherwise it returns the default typeface.  
   b. Use the script tag to search the major (or minor) font collection. If found, then OK.  
   c. If not found, we return the default typeface, depending on whether it’s latin, or ea, or cs.  
For each run, we run three times: latin, ea, and cs. That said, for each run, we have three typefaces at runtime. But in the Ribbon font gallery, OArt text appears the theme name – e.g. a piece of English text with “Mlym” (Malayalam) appears in Ribbon “Kartika (Body)” in font gallery, but actually it renders in the slide with Calibri.

Regarding the three mismatch cases:  
1. Why doesn't “Mong” work? Most likely, its LCID (0x450) is converted to script “Cyrl”.

2. Why doesn't “Yiii” work? We can’t reproduce this. Can you provide a .pptx sample document to help us reproduce the problem?

3. Why doesn't “Uigh” work? Do you mean VBA doesn’t support it? If so, you are right. But you can change the Xml using “UG-CN”.

Best regards,  
Tom Jebo  
Escalation Engineer  
Microsoft Open Specifications

Thanks Tom.

* 1. **Regarding point A:**How are the LCIDs mapped to script tags? Does it use IANA scripts or...? I can't imagine this would be propriatary information as that would defeat the whole of purpose of having an open format for DrawingML and any type of globalization. That said, it doesn't appear to be documented anywhere. As demonstrated in my follow up post, I can sort of figure things out, but need the standards to document what the standard is.

* 1. **Regarding point C:** Is it that there is only one default typeface (latin, ea or cs) at a type in <rPr> ? Is the default script found based on unicode of the characters in the run text or something else?

* 1. **Regarding the sentance that begins with "For each run":** I'm not sure if I understand what you're saying. Can you elaborate?

* 1. **Regarding your questions 1, 2 and 3:** those are eamples of where VBA doesn't create this using any of the MSOLanguageIDs - i.e. if you create a textbox in all languages that PowerPoint supports automated input from, these 3 never appear (using PowerPoint 2010 x64).

------------------

Okatu,

Here are the answers I mentioned:

>> Regarding point A: How are the LCIDs mapped to script tags? Does it use IANA scripts or...? I can't imagine this would be propriatary information as that would defeat the whole of purpose of having an open format for DrawingML and any type of globalization. That said, it doesn't appear to be documented anywhere. As demonstrated in my follow up post, I can sort of figure things out, but need the standards to document what the standard is.

Please send an email to dochelp at microsoft.com and I will discuss this with you further.

>> Regarding point C: Is it that there is only one default typeface (latin, ea or cs) at a type in <rPr> ? Is the default script found based on unicode of the characters in the run text or something else?

For each theme, there is a default for each of them:

      <a:latin typeface="Calibri"/>  
      <a:ea typeface=""/>  
        <a:cs typeface=""/>

There is no need to calculate the default script. We figure out whether it’s latin or ea or cs from characters, then we know which one from above to use.

>> Regarding the sentance that begins with "For each run": I'm not sure if I understand what you're saying. Can you elaborate?

This is PowerPoint runtime behavior. E.g. for each <a:r>

      <a:r>  
         <a:rPr lang="ii-CN" altLang="ii-CN" dirty="0" smtClean="0"/>  
         <a:t>ZmsoLanguageIDYi</a:t>  
         Latin-typeface = the default latin typeface // Use this for layout / glyph at runtime  
         CS-typeface = the default cs typeface // Use this for layout / glyph at runtime  
      </a:r>

>> Regarding your questions 1, 2 and 3: those are eamples of where VBA doesn't create this using any of the MSOLanguageIDs - i.e. if you create a textbox in all languages that PowerPoint supports automated input from, these 3 never appear (using PowerPoint 2010 x64).

For “Mong” and “Yiii”, they work in PowerPoint 2010, and they are msoLanguageIDMongolian and msoLanguageIDYi. But for “Uigh”, you are right – there is no MsoLanguageID for this one. You can still use 1152 (0x480) as follows:

      activepresentation.Slides(1).Shapes(1).TextFrame.TextRange.LanguageID=1152  
   
Best regards,  
Tom Jebo  
Escalation Engineer   
Microsoft Open Specifications

 ---------

**>>There is no need to calculate the default script. We figure out whether it’s latin or ea or cs from characters, then we know which one from above to use.**

I guess this is the question. How do you figure out weither it's latin/ea/cs from the characers? Is it just looking for a particular set of codepages based on all the text, some of the text, or something else? Is it mulitple codepages?

**>>For “Mong” and “Yiii”, they work in PowerPoint 2010, and they are msoLanguageIDMongolian and msoLanguageIDYi. But for “Uigh”, you are right – there is no MsoLanguageID for this one. You can still use 1152 (0x480)**

Confirmed for Uigher, but for Mongolian and Yi, the display font remains Calibri. For example:

Sub CreateLangEntryBox()  
    Dim p As Presentation: Set p = ActivePresentation  
    Dim S As slide: Set S = p.Slides(1)  
    Dim sh As Shape: Set sh = S.Shapes.AddShape(msoShapeRectangle, 200, 200, 200, 50)  
    sh.Name = "Mongolian"  
    With sh.TextFrame.TextRange  
        .Text = "Mongolian"  
        .LanguageID = msoLanguageIDMongolian  
    End With  
End Sub

Maybe this is like Chinese where in order for the font to be displayed, the text in the shape needs to be written in the script? I'm not sure how I would enter either of those scripts in in VBA, but if you can confirm that this is like Chinese where it only displays in Calibri if \*not\* in expected script, that would be fine. Also, I believe this may be related to your **>>This is PowerPoint runtime behavior**. Is that correct?

Okatu,

Thanks for your patience.  Here are answers to your follow-up questions:

>>I guess this is the question. How do you figure out weither it's latin/ea/cs from the characers? Is it just looking for a particular set of codepages based on all the text, some of the text, or something else? Is it mulitple codepages?

We use UNICODE sub ranges + some Windows APIs to decide this. Some typical examples as follows:  
a. South Asia + Bidi -> CS.  
b. Surrogates are classified to FE.  
c. FE check also uses a Windows API GetStringTypeExW.

That said, we use UNICODE rather than depending on codepages / locales when dealing with text layout and glyph things.

>>Confirmed for Uigher, but for Mongolian and Yi, the display font remains Calibri. For example:  
   
Sub CreateLangEntryBox()  
     Dim p As Presentation: Set p = ActivePresentation  
     Dim S As slide: Set S = p.Slides(1)  
     Dim sh As Shape: Set sh = S.Shapes.AddShape(msoShapeRectangle, 200, 200, 200, 50)  
     sh.Name = "Mongolian"  
     With sh.TextFrame.TextRange  
         .Text = "Mongolian"  
         .LanguageID = msoLanguageIDMongolian  
     End With  
 End Sub  
   
Probably you want to set the language first, and then the text, something like:

activepresentation.Slides(1).Shapes(2).TextFrame.TextRange.LanguageID=msoLanguageIDYi  
activepresentation.Slides(1).Shapes(2).TextFrame.TextRange.Text="am I yiii"

Please notice that Mongolian corresponds to “Cyrl”.

Best regards,  
Tom Jebo  
Escalation Engineer   
Microsoft Open Specifications

------------------------

Greetings.  I would like to continue this discussion.  I have similar questions as Okatu, but I am not yet satisfied with the answers given.  In particular, I'd like to further discuss the algorithm Tom Jebo outlined above and repeated below:

------------------------------------------------------------------

Algorithm:  
   a. Call MSO APIs to convert a LCID to a script tag. It uses language or alt language (if the former is missing). And it requires a language and slot match; otherwise it returns the default typeface.  
   b. Use the script tag to search the major (or minor) font collection. If found, then OK.  
   c. If not found, we return the default typeface, depending on whether it’s latin, or ea, or cs.  
For each run, we run three times: latin, ea, and cs. That said, for each run, we have three typefaces at runtime. But in the Ribbon font gallery, OArt text appears the theme name – e.g. a piece of English text with “Mlym” (Malayalam) appears in Ribbon “Kartika (Body)” in font gallery, but actually it renders in the slide with Calibri.

------------------------------------------------------------------

As I understand things, the example above contradicts the stated algorithm.  For the example, I understand that there is a run with a lang tag that specifies the language to be Malayalam.  According to the algorithm, we first convert the locale ID to the script tag Mlym.  We then use this script tag to locate the font Kartika.  As this was correctly found, then the algorithm states this is the font that should be used to render the text.  However, you state that it is actually rendered with Calibri.

How, then, did PowerPoint decide to use Calibri?

I have a couple other follow-up questions.  Somewhere in this thread it was stated that PowerPoint uses UNICODE sub ranges + some Windows APIs to decide whether a character falls in the latin, east asian, or complex script buckets.  I would like to know two things:

1)  What are the exact unicode ranges PowerPoint uses to define each bucket?

2)  Supposing that the Windows platform is not available, then a Window API call is not possible.  How, then, should I replace this missing functionality.  What does the Window API call do so that I can reproduce the logic in a non-Windows environment?

Hi BoulderPika,

Please check out the post I just made:

<http://social.msdn.microsoft.com/Forums/en-US/os_openXML-ecma/thread/1bf1f185-ee49-4314-94e7-f4e1563b5c00>

The table should give you what you need to translate unicode ranges to latin, cs and ea.

---------------

Tom Jebo -

I am still having difficulties understanding PowerPoint's algorithm for determining what font to use for a given text run.  I have created a simple test case and hand-edited the XML to further reduce complexity.  You can find that test case here:

http://home.comcast.net/~bellert/symbol1.pptx

Of particular interest is the following XML from theme1.xml:

------------------------------------------------------------------------------------------------------------------------

...

      <a:fontScheme name="Office">  
        <a:majorFont>  
          <a:latin typeface="Calibri"/>  
          <a:ea typeface=""/>  
          <a:cs typeface=""/>  
          <a:font script="Jpan" typeface="ＭＳ Ｐゴシック"/>  
...

------------------------------------------------------------------------------------------------------------------------

and the following from slide1.xml:

------------------------------------------------------------------------------------------------------------------------

...

             <a:r>  
                <a:rPr lang="en-US" sz="7200" dirty="0" smtClean="0">  
                  <a:latin typeface="+mj-lt"/>  
                  <a:ea typeface="+mj-ea"/>  
                  <a:cs typeface="+mj-cs"/>  
                </a:rPr>  
                <a:t>⓫</a:t>  
              </a:r>  
...

------------------------------------------------------------------------------------------------------------------------

The text within the a:t tag has Unicode value 0x24EB.  Using the table you referenced in the link above, I determine that this should use the East Asian font (character range 0x2460 - 0x266F).  Looking at my run properties for the text, I see that the ea typeface is "+mj-ea".  I interpret this to use the major East Asian font from the theme file.  However, retrieving the major East Asian font from the theme file gives me the empty string.  I'm perplexed.  I've greatly simplified the test case and am reasonably certain I have removed all other places where a typeface could have been set.

I'm referencing ECMA-376 Third Edition from June 2011.  This specification (20.1.4.1.24) does not indicate the use of any default fonts for the children of the majorFont tag and nor does the specification (21.1.2.3.3) indicate that there should be any default values for the ea tag.

Are you certain that the character range 0x2460 - 0x266F calls for the East Asian font?  For this test case, I simply don't see an East Asian font defined anywhere.

If it matters, I'm using PowerPoint 14.0.4760.1000 (32-bit).

BoulderPika,  
   
Yes, from what you’ve described, your interpretation seems correct and the specification is itself correct (aside from the defect report information that is pending, relevant parts of which I've posted here already).  And PowerPoint is following the specification wrt your font question.  We have discussed your scenario questions about selecting font when the theme typeface (i.e. <a:ea typeface=""/>) is blank, here is the result of our discussion:  
   
You correctly traced back to the a:fontScheme element and its children.  In this case, the file (typeface attribute of the a:ea element) has not requested the use of a particular font face for that type of text and the implementation is free to select an appropriate font for its context, given its particular needs and constraints.  
   
This gets into the realm of font substitution, fallback fonts and font linking.  This area is rather complicated and is a typical task for application developers dealing with text.  It arises even when font faces are specified, as the font requested is not always available in a given environment (e.g., which fonts are installed by default on various OSes, use of user-installed fonts).  Industry-wide, there is no common way to handle font selection, partly because applications sometimes have different needs for how detailed they need to be in choosing a font.  Often, developers will let the OS figure it out, but applications with more specific needs, such as Office, often put a lot of work into logic for finding the most appropriate font.  
   
So, this is up to the developer to decide which font to select based on the type of text, fonts available on the system, and any particular design goals of the application.  An application might decide to use another major font, or the corresponding minor font if it defined, or fall back to a “hard-coded” font appropriate for that font type (e.g., East Asian), or do further analysis of the text run and the available fonts on the system to find a more appropriate fallback.  There is a globalization article that discusses these topics on MSDN that you may find helpful in getting started determining how your application wants to handle fonts: <http://msdn.microsoft.com/en-us/goglobal/bb688134>.

Best regards,  
Tom Jebo  
Escalation Engineer   
Microsoft Open Specifications