

---

**The unicodfonttable package\***

Frank Mittelbach

**Abstract**

A package for typesetting font tables for larger fonts, e.g., TrueType or OpenType Unicode fonts. To produce a one-off table, a standalone version is available as well.

**Contents**

1	Introduction	901
2	The user interface	902
	2.1 Keys and their values . . . . .	903
	2.2 A standalone interactive version . . . . .	906
3	Notes on the table data	906
4	Examples	907
	4.1 Computer Modern Sans — 7-bit font . . . . .	907
	4.2 T <sub>E</sub> X Gyre Heros — 8-bit font . . . . .	908
	4.3 Latin Modern Math — 8-bit fonts . . . . .	908
	4.4 Latin Modern Math compared to New Computer Modern Math . .	909
	4.5 Garamond Libre’s Byzantine Musical Symbols . . . . .	918
5	The package implementation	919
	5.1 User interface commands . . . . .	920
	5.2 The overall table layout . . . . .	922
	5.3 The producing the table content . . . . .	926
	5.4 Handling a single row . . . . .	930
	5.5 Initialisation at the start of the table . . . . .	933
	5.6 Handling block titles . . . . .	933
6	The standalone <code>unicodfont.tex</code> file	939
7	A samples file	940

**1 Introduction**

When I started to write a new chapter for the third edition of *The L<sup>A</sup>T<sub>E</sub>X Companion* on modern fonts available for different L<sup>A</sup>T<sub>E</sub>X engines, I was a bit surprised that I couldn’t find a way to easily typeset tables showing the glyphs available in TrueType or OpenType fonts. The `nfssfont` package available with L<sup>A</sup>T<sub>E</sub>X only supports fonts from the 8-bit world, but modern fonts that can be used with X<sub>Y</sub>L<sup>A</sup>T<sub>E</sub>X or Lua<sub>T</sub>E<sub>X</sub> can contain thousands of glyphs and having a method to display what is available in them was important for me.

I therefore set out to write my own little package and what started as an afternoon exercise ended up being this package, offering plenty of bells and whistles for typesetting such font tables.

As there can be many glyphs in such fonts a tabular representation of them might run for several pages, so the package internally uses the `longtable` package to handle that. In most cases the glyphs inside the fonts are indexed by their Unicode numbers so it is natural to display them sorted by their position in the Unicode character set.

---

\* This is version 1.0h of the package, dated 2023/05/10; the license is LPPL.

Unicode is organized in named blocks such as “Basic Latin”, “Latin-1 Supplement”, etc., typically consisting of 265 characters each.<sup>1</sup> It is therefore helpful to use these block names as subtitles within the table, to more easily find the information one is looking for.

A common way to represent the number of a single Unicode character is U+ followed by four (or more) hexadecimal digits. For example, U+0041 represents the letter “A” and U+20AC the Euro currency symbol “€”. We use this convention by showing a Unicode range of sixteen characters at the left of each table row, e.g., U+0040 – 004F, followed by the sixteen glyphs in the range. Thus that particular table row from the “Basic Latin” block would show something like

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+0040 – 004F	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O

If a Unicode character has no glyph representation in a given font then this is indicated by a special symbol (by default a colored hyphen). By default some color is used, but we’ve grayscaled the output for *TUGboat*.

In order to easily locate any Unicode character the table shows by default sixteen hex digits as a column heading. For example, to find Euro currency symbol (U+20AC) one first finds the right row, which is the range U+20A0 – 20AF, and then the C column in that row, and the glyph is there (or an indication that the font is missing that glyph; the line shows that for some of the other slots).

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+20A0 – 20AF	-	€	-	-	£	-	N	-	-	W	-	d	€	-	-	-

It can be useful to compare two fonts with each other by filling the table with glyphs from a secondary font if the primary font is missing them. For example, the next display shows two rows of Latin Modern Math (black glyphs) and instead of showing a missing glyph symbol in most slots, we use the glyphs from New Computer Modern Math, which has a much larger glyph set (normally red glyphs with gray background but again, grayscaled for *TUGboat*).

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+2A00 – 2A0F	⊙	⊕	⊗	⊔	⊕	∏	∏	M	W	×	Σ	\$	ffff	f	f	f
U+2A10 – 2A1F	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f	f

## 2 The user interface

The package offers one command to typeset a font table. The appearance of the table can be customized by specifying key/value pairs.

---

`\displayfonttable` `\displayfonttable * [(key/value-list)] {font-name} [font-features]`

The `{font-name}` is the font to be displayed. This and the `{font-features}` argument are passed to `fontspec`, thus they should follow the conventions of that package for specifying a font. The `{key/value-list}` offers customization possibilities discussed below. The `\displayfonttable*` is a variant of the command, intended for use with 8-bit legacy fonts. It presets some keys, but otherwise behaves identically. The preset values are:

`nostatistics, display-block=none, hex-digits=head, range-end=FF`

For details see the next section.

<sup>1</sup> Some blocks are smaller, while those containing the Asian ideographs are much larger.

---

**\fonttablesetup** `\fonttablesetup {(key/value-list)}`

Instead of or in addition to specifying key/values to `\displayfonttable` it is possible to set them up as defaults. Inside `\displayfonttable` the defaults are applied first, so one can still overwrite their values for an individual table.

---

**\fonttableglyphcount** `\fonttableglyphcount`

While typesetting a font table the package keeps track of the number of glyphs it finds in the font. After the table has finished, this value is available in `\fonttableglyphcount` and it is, for example, used when statistics are produced. At the start of the next table it is reset to zero.

## 2.1 Keys and their values

Several of the available keys are booleans accepting `true` or `false`. They usually exist in pairs so that one can specify the desired behavior without needing to provide a value, e.g., specifying `header` is equivalent to specifying `header=true` or `noheader=false`, etc. In the lists below the default settings are indicated by an underline.

---

**header**  
**noheader**  
**title-format**  
**title-format-cont**

The first set of keys is concerned with the overall look and feel of the generated table.

**header**, **noheader** These keys determine whether a header to the table is produced.

**title-format**, **title-format-cont** These keys define what is provided as a header title or continuation title if the table consists of several pages. They expect code as their value. This code can contain `#1` and `#2` to denote the `<font-name>` and `<font-features>` arguments, respectively.

By default a title using the `\caption` command is produced; on continuation titles, the `<font-features>` are not shown. This is typeset as a `longtable` header row, so you either need to use `\multicolumn` or a `\caption` command — otherwise everything ends up in the first column.

---

<code>display-block</code>	These keys handle the inner parts of the table.
<code>hex-digits</code>	
<code>hex-digits-font</code>	<b>display-block</b> The Unicode dataset is organized in named blocks that are typically 128 or 256 characters, though some are noticeably larger and a few are smaller. With the <code>display-block</code> key it is possible to specify if and how such blocks should be made visible. The following values are supported:
<code>hex-digits-row-format</code>	<b>titles</b> Above each display block that contains glyphs the Unicode title of the block is displayed.
<code>color</code>	<b>rules</b> Display blocks are indicated only by a <code>\midrule</code> .
	<b>none</b> Display blocks are not indicated at all.
	<b>hex-digits</b> To ease reading the table, rows of hex digits are added to it. Where or if this happens is controlled by this key. Allowed values for it are the following:
	<b>block</b> A row of hex digits is placed at the beginning of each Unicode block containing glyphs in the displayed font.
	<b>foot</b> A row is added to the foot of each table page.
	<b>head</b> A row is added to the top of each table page.
	<b>head+foot</b> A row is added to the top and the foot of each table page.
	<b>none</b> All hex digit rows are suppressed.
	<b>hex-digits-font</b> The font to use for the hex digits, by default <code>\ttfamily\scriptsize</code> .
	<b>hex-digits-row-format</b> This key defines the format for the hex digits shown on the left of each row. It accepts one argument hold the hex values for the row except for the last digit, e.g. <code>0A3</code> for the values from <code>0A30</code> to <code>0A3F</code> . The default formatting is <code>U+#10\,-\,#1F</code> and without further adjustments it is automatically set in <code>\footnotesize \ttfamily</code> and in the color specified by the <code>color</code> key.
	<b>color</b> This key determines the color for parts of the table (hex digits and Unicode ranges). It can be either <code>none</code> or a color specification as understood by the <code>\color</code> command. The default is <code>blue</code> .

---

<code>statistics</code>	The next set of keys allows altering the statistics that are produced.
<code>nostatistics</code>	
<code>statistics-font</code>	<b>statistics, nostatistics</b> These keys determine whether some statistics are listed at the end of the table.
<code>statistics-format</code>	<b>statistics-font</b> The font used to typeset the statistics; the default is <code>\normalfont\small</code> .
	<b>statistics-format</b> Code (text) to specify what should be typeset in the statistics. One can use <code>#1</code> for the <i>font-name</i> and <code>#2</code> for the glyph count. The material is typeset on a single line at the end of the table. If several lines are needed you need to use <code>\parbox</code> or a similar construct.

---

<code>glyph-width</code>	Another set of keys deals with customization on the glyph level.
<code>missing-glyph</code>	
<code>missing-glyph-font</code>	<b>glyph-width</b> All glyphs are typeset in a box with the same width, the default value is <code>6pt</code> which is suitable for most 10pt fonts and make the table fit comfortably into the text width of a typical document.
<code>missing-glyph-color</code>	<b>missing-glyph</b> If a slot in a row doesn't have a glyph in the font you may still want display something to indicate this state. By giving the key a value any arbitrary glyph or material can be typeset. The default is to typeset a - (hyphen) in a special color. Rows that contain no glyph whatsoever are not displayed at all. Instead a small vertical space is added to indicate the one or more rows are omitted.
	<b>missing-glyph-font</b> The font used for the missing glyphs (the default value is <code>\ttfamily\scriptsize</code> ).
	<b>missing-glyph-color</b> If not specified it uses the value specified with the <code>color</code> key. If you want a different color, e.g., <code>red</code> , you can use a color value or you can specify <code>none</code> to use no coloring.

---

<code>compare-with</code>	You can make comparisons between two fonts, which is useful, for example when dealing with incomplete math fonts and you need to see how well the symbols from one font blend with the supplementary symbols from another font.
<code>compare-color</code>	
<code>compare-bgcolor</code>	
<code>statistics-compare-format</code>	

---

	<b>compare-with</b> If given, the value is a <i>⟨comparison-font-name⟩</i> that is used to supply missing glyphs. This means that if the <i>⟨font-name⟩</i> to be displayed is missing a glyph in a slot, then the <i>⟨comparison-font-name⟩</i> is checked, and if that font has the glyph in question, it will be displayed instead of showing a missing glyph indicator.
	<b>compare-color, compare-bgcolor</b> To distinguish real glyphs from missing but substituted glyphs, they can be colored specially (default <code>red</code> ) and/or you can have their background colored (default is <code>black!10</code> , i.e., a light gray).
	<b>statistics-compare-format</b> Code (text) to specify what should be typeset in the statistics when comparing two fonts. One can use <code>#1</code> for the <i>⟨font-name⟩</i> and <code>#2</code> for its glyph count, <code>#3</code> is the name of the comparison font, <code>#4</code> its glyph count, <code>#5</code> for the number of glyphs missing in this font and <code>#6</code> the number of extra glyphs in it. This code is used instead of <code>statistics-format</code> when comparisons are made. The material is typeset on a single line at the end of the table. If several lines are needed you need to use <code>\parbox</code> or a similar construct.

---

**range-start** Finally there are two keys for restricting the display range.  
**range-end**

---

**range-start, range-end** The full Unicode set of characters is huge and checking every slot to see if the current font contains a glyph in the slot takes a long time. If you know that font contains only a certain subset then you can speed up the table generation considerably by limiting the search (and consequently the output generation). The **range-start** specifies where to start with the search (default **0000**) and **range-end** gives the last slot that is tested (default **FFFF**).

Thus, by default we restrict the display to slots below 10000, because text fonts seldom contain glyphs in the higher planes. But if you want to see everything of the font (as far as supported by this package) and are prepared to wait for the higher planes to be scanned, you can go up to a value of **FFFF**.

However, please note that the LuaTeX fontloader uses the “Supplementary Private Use Area-A”, which starts at **F0000**, as its own playground and places remapping into it, so by default you see random data instead of font data there. You either have to use the XeTeX engine or load the font with **Renderer=HarfBuzz** in LuaTeX.

These keys are also quite useful in combination with the previous **compare-with** key, to display only, for example, the Greek letters and see how glyphs from two fonts blend with each other.

## 2.2 A standalone interactive version

If you want to quickly display a single font, you can run **unicodfont.tex** through LuaTeX (or XeTeX). Similar to **nfssfont.tex** (which is for 8-bit fonts with pdfTeX) it asks you a few questions and then generates the font table for you. There are fewer configuration options available, but this workflow saves you writing a document to get a one-off table.

Most font tables need several runs due to the use of **longtable**, which has to find the right width for the columns across several pages. The **unicodfont** file therefore remembers your selection from the previous run and asks you if you want to reapply it to speed up the process.

## 3 Notes on the table data

If you look at some parts of a Unicode font table you see a number of slots that do not show a “missing glyph” sign, but nonetheless appear to be empty. For example:

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+0020–002F		!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
U+0030–003F	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
U+0040–004F	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
U+0050–005F	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
U+0060–006F	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
U+0070–007F	p	q	r	s	t	u	v	w	x	y	z	{		}	~	-
U+00A0–00AF		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	®	¯	°
U+00B0–00BF	°	±	²	³	´	µ	¶	·	,	¹	º	»	¼	½	¾	¿

The reason is that Unicode contains a lot of special spaces or otherwise invisible characters, e.g., **U+0020** is the normal space, **U+00A0** is a non-breaking space, **U+00AD** is a soft-hyphen (what L<sup>A</sup>T<sub>E</sub>X users would indicate with  $\-$ ), and so forth. Especially the row **U+2000–200F** in Table 6 looks strange as it appears to be totally empty, but in fact most of its slots contain spaces of different width.

### General Punctuation

U+2000 - 200F	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
U+2010 - 201F	-	-	-	-	-	-		=	'	'	,	-	“	”	”	-
U+2020 - 202F	†	‡	•	-	-	-	...	-	-	-	-	-	-	-	-	-
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Another somewhat surprising area is the “Mathematical Alphanumeric Symbols” block in math fonts, starting at U+1D400. There you see a number of missing characters, the first two being U+1D455 (math italic small h) and U+1D49D (math script B).

### Mathematical Alphanumeric Symbols

U+1D400 - 1D40F	<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>	<b>L</b>	<b>M</b>	<b>N</b>	<b>O</b>	<b>P</b>
U+1D410 - 1D41F	<b>Q</b>	<b>R</b>	<b>S</b>	<b>T</b>	<b>U</b>	<b>V</b>	<b>W</b>	<b>X</b>	<b>Y</b>	<b>Z</b>	<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>	<b>e</b>	<b>f</b>
U+1D420 - 1D42F	<b>g</b>	<b>h</b>	<b>i</b>	<b>j</b>	<b>k</b>	<b>l</b>	<b>m</b>	<b>n</b>	<b>o</b>	<b>p</b>	<b>q</b>	<b>r</b>	<b>s</b>	<b>t</b>	<b>u</b>	<b>v</b>
U+1D430 - 1D43F	<b>w</b>	<b>x</b>	<b>y</b>	<b>z</b>	<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>	<i>F</i>	<i>G</i>	<i>H</i>	<i>I</i>	<i>J</i>	<i>K</i>	<i>L</i>
U+1D440 - 1D44F	<i>M</i>	<i>N</i>	<i>O</i>	<i>P</i>	<i>Q</i>	<i>R</i>	<i>S</i>	<i>T</i>	<i>U</i>	<i>V</i>	<i>W</i>	<i>X</i>	<i>Y</i>	<i>Z</i>	<i>a</i>	<i>b</i>
U+1D450 - 1D45F	<i>c</i>	<i>d</i>	<i>e</i>	<i>f</i>	<i>g</i>	-	<i>i</i>	<i>j</i>	<i>k</i>	<i>l</i>	<i>m</i>	<i>n</i>	<i>o</i>	<i>p</i>	<i>q</i>	<i>r</i>
U+1D460 - 1D46F	<i>s</i>	<i>t</i>	<i>u</i>	<i>v</i>	<i>w</i>	<i>x</i>	<i>y</i>	<i>z</i>	<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>	<i>F</i>	<i>G</i>	<i>H</i>
U+1D470 - 1D47F	<i>I</i>	<i>J</i>	<i>K</i>	<i>L</i>	<i>M</i>	<i>N</i>	<i>O</i>	<i>P</i>	<i>Q</i>	<i>R</i>	<i>S</i>	<i>T</i>	<i>U</i>	<i>V</i>	<i>W</i>	<i>X</i>
U+1D480 - 1D48F	<i>Y</i>	<i>Z</i>	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>	<i>f</i>	<i>g</i>	<i>h</i>	<i>i</i>	<i>j</i>	<i>k</i>	<i>l</i>	<i>m</i>	<i>n</i>
U+1D490 - 1D49F	<i>o</i>	<i>p</i>	<i>q</i>	<i>r</i>	<i>s</i>	<i>t</i>	<i>u</i>	<i>v</i>	<i>w</i>	<i>x</i>	<i>y</i>	<i>z</i>	<i>A</i>	-	<i>C</i>	<i>D</i>
U+1D4A0 - 1D4AF	-	-	<i>G</i>	-	-	<i>J</i>	<i>K</i>	-	-	<i>N</i>	<i>O</i>	<i>P</i>	<i>Q</i>	-	<i>S</i>	<i>T</i>
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

In this case the reason is *not* that the font fails to implement the characters, but that these characters have already been defined in earlier revisions of the Unicode standard in the lower Unicode plane. For example, the “h” is the Planck constant U+210E and U+212C is the script capital B, etc. The Unicode Consortium decided not to encode the *same* character twice, hence the apparent holes.

## 4 Examples

In this section we show the results of a few calls to `\displayfonttable`. The tables are a bit easier to navigate if they use color in some places, but for *TUGboat* this is not practical, so we use black and gray.

Please note that this documentation was produced with Lua $\TeX$ . If you reuse the examples with X $\TeX$ , you may have to specify the font names differently (i.e., following to the `fontspec` documentation for this engine).

### 4.1 Computer Modern Sans — 7-bit font

Our first example is the original Computer Modern Sans, with character codes  $\leq 127$ . Command used:

```
\displayfonttable*[color=none, range-end=7F]{cmss10}
```

Table 1: cmss10

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+0000 - 000F	Γ	Δ	Θ	Λ	Ξ	Π	Σ	Υ	Φ	Ψ	Ω	ff	fi	fl	ffi	ffl
U+0010 - 001F	ı	ı	`	'	˘	˙	-	°	,	β	æ	œ	ø	Æ	Œ	Ø
U+0020 - 002F	-	!	”	#	\$	%	&	'	(	)	*	+	,	-	.	/

Table 1: cmss10 cont.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+0030-003F	0	1	2	3	4	5	6	7	8	9	:	;	i	=	¿	?
U+0040-004F	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
U+0050-005F	P	Q	R	S	T	U	V	W	X	Y	Z	[	“	]	^	·
U+0060-006F	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
U+0070-007F	p	q	r	s	t	u	v	w	x	y	z	-	—	”	~	..

### 4.2 T<sub>E</sub>X Gyre Heros — 8-bit font

This example shows the T<sub>E</sub>X Gyre Heros 8-bit font, in the T1 encoding, with character codes ≤ 255. We used `hex-digits-row-format` to shorten the row titles on the left:

```
\displayfonttable*[color=none,hex-digits-row-format=U+#1]{ec-qhvr}
```

Table 2: ec-qhvr

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+000	`	´	^	~	¨	˘	˙	˚	¸	ˆ	·	ˆ	ˆ	ˆ	ˆ	ˆ
U+001	“	”	„	«	»	—	—	o	l	j	ff	fi	fl	ffi	ffl	
U+002	ı	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
U+003	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
U+004	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
U+005	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
U+006	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
U+007	p	q	r	s	t	u	v	w	x	y	z	{		}	~	-
U+008	Ă	Ą	Ć	Č	Ď	Ě	Ę	Ğ	Ĺ	Ł	ł	Ń	Ň	Ń	Ŏ	Ŕ
U+009	Ř	Ś	Ŝ	Ş	Ť	Ŧ	Ũ	Û	Ÿ	Ž	ž	Ž	ı	ı	đ	š
U+00A	ă	ą	ć	č	ď	ě	ę	ğ	ĺ	ł	ł	ń	ň	ň	ŋ	ő
U+00B	ř	ś	ŝ	ş	ť	ŧ	ű	ű	ÿ	ž	ž	ž	ı	ı	ı	£
U+00C	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
U+00D	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß	
U+00E	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
U+00F	ð	ñ	ò	ó	ô	õ	ö	ø	ù	ú	û	ü	ý	þ	ß	

### 4.3 Latin Modern Math — 8-bit fonts

The traditional Latin Modern Math Italic, Symbol and Extension fonts. The symbol font (`lmsy10`) has two characters added to the Computer Modern symbol repertoire, seen in the last row of the table. Commands used:

```
\displayfonttable*[color=none]{lmmi10}
\displayfonttable*[color=none]{lmsy10}
\displayfonttable*[color=none]{lmex10}
```

Table 3: lmmi10

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+0000-000F	Γ	Δ	Θ	Λ	Ξ	Π	Σ	Υ	Φ	Ψ	Ω	α	β	γ	δ	ε
U+0010-001F	ζ	η	θ	ι	κ	λ	μ	ν	ξ	π	ρ	σ	τ	υ	φ	χ
U+0020-002F	ψ	ω	ε	ϑ	ϖ	ϗ	ς	φ	←	↖	↗	→	↘	↙	↞	↠
U+0030-003F	0	1	2	3	4	5	6	7	8	9	.	,	<	/	>	*
U+0040-004F	∂	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
U+0050-005F	P	Q	R	S	T	U	V	W	X	Y	Z	b	⋮	#	⌋	⌌
U+0060-006F	ℓ	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o



**Table 3:** *lmmi10 cont.*

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+0070-007F	<i>p</i>	<i>q</i>	<i>r</i>	<i>s</i>	<i>t</i>	<i>u</i>	<i>v</i>	<i>w</i>	<i>x</i>	<i>y</i>	<i>z</i>	<i>ı</i>	<i>j</i>	ø	→	←

**Table 4:** *lmsy10*

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+0000-000F	−	·	×	*	÷	◇	±	∓	⊕	⊖	⊗	⊘	⊙	○	◦	●
U+0010-001F	≠	≡	⊆	⊇	≤	≥	≲	≳	≈	≈	⊂	⊃	⊂	⊃	↖	↗
U+0020-002F	←	→	↑	↓	↔	↗	↘	≈	⇐	⇒	↑	↓	↔	↖	↗	∞
U+0030-003F	/	∞	∈	∃	△	▽	/	ı	∇	∃	¬	∅	℔	℔	⊥	⊥
U+0040-004F	℔	<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	<i>E</i>	<i>F</i>	<i>G</i>	<i>H</i>	<i>I</i>	<i>J</i>	<i>K</i>	<i>L</i>	<i>M</i>	<i>N</i>	∅
U+0050-005F	<i>P</i>	<i>Q</i>	<i>R</i>	<i>S</i>	<i>T</i>	<i>U</i>	<i>V</i>	<i>W</i>	<i>X</i>	<i>Y</i>	<i>Z</i>	∩	∩	∩	∩	∩
U+0060-006F	⊢	⊣	⊥	⊥	⊥	⊥	{	}	<	>			↕	↕	↖	↗
U+0070-007F	√	∏	∇	∫	∫	∫	∫	∫	∫	†	‡	♣	♣	♣	♣	♣
U+00A0-00AF	-	-	-	-	-	-	-	-	-	-	-	-	≤	≥	-	-

**Table 5:** *lmex10*

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+0000-000F	( )	[ ]	[ ]	[ ]	[ ]	[ ]	{ }	< >			/	\				
U+0010-001F	( )	( )	[ ]	[ ]	[ ]	[ ]	{ }	< >	/	\						
U+0020-002F	( )	[ ]	[ ]	[ ]	[ ]	[ ]	{ }	< >	/	\	/	\				
U+0030-003F	( )	[ ]	[ ]	[ ]	'	'	'	'	'	'	'	'	'	'	'	'
U+0040-004F	( )	'	'	< >	∩	∩	∫	∫	⊙	⊙	⊕	⊕	⊗	⊗		
U+0050-005F	Σ	Π	∫	∩	∩	∩	∩	∩	∩	∩	∫	∩	∩	∩	∩	∩
U+0060-006F	∏	∏	∩	∩	∩	∩	∩	∩	[ ]	[ ]	[ ]	[ ]	{ }	{ }		
U+0070-007F	√	√	√	√	√		∫	∫	↑	↓	∩	∩	∩	∩	∩	∩

#### 4.4 Latin Modern Math compared to New Computer Modern Math

This example shows the extra symbols available in New Computer Modern Math in comparison to Latin Modern Math as the base font. We use the following setup (including settings for the grayscale *TUGboat* output, as an example of color overrides):

```
\displayfonttable[hex-digits=head+foot, range-end=1FFFF,
compare-with=New Computer Modern Math,
title-format=\caption{Latin Modern Math compared to
New Computer Modern Math},
title-format-cont=\caption{LM Math vs.\ NewCM Math,
\emph{cont.}},
```

```
compare-color=black, compare-bgcolor=black!5,
missing-glyph-color=black!50, color=black!75]
{Latin Modern Math}
```

That is, glyphs only in NewCM are shown with a light gray background.

We also extended the range to cover U+10000 to U+1FFFF in order to include the Unicode Math alphabets.

**Table 6:** Latin Modern Math compared to New Computer Modern Math

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
<b>Basic Latin</b>																
U+0020-002F		!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
U+0030-003F	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
U+0040-004F	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
U+0050-005F	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
U+0060-006F	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
U+0070-007F	p	q	r	s	t	u	v	w	x	y	z	{		}	~	<span style="border: 1px solid black; padding: 0 2px;">DEL</span>
<b>Latin-1 Supplement</b>																
U+00A0-00AF		ı	ċ	£	¤	¥		§	¨	©	ª	«	¬	•	®	¯
U+00B0-00BF	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
U+00C0-00CF	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
U+00D0-00DF	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
U+00E0-00EF	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
U+00F0-00FF	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ
<b>Latin Extended-A</b>																
U+0100-010F	Ā	ā	Ă	ă	Ą	ą	Ć	ć	Ĉ	ĉ	Č	č	Ď	ď		
U+0110-011F	Đ	đ	Ē	ē	Ĕ	ĕ	È	è	Ě	ě	Ĝ	ĝ	Ğ	ğ		
U+0120-012F	Ġ	ġ	Ģ	ģ	Ĥ	ĥ	Ħ	ħ	Ĩ	ĩ	Ī	ī	Ĭ	ĭ	Į	į
U+0130-013F	Ĳ	ı	IJ	ij	Ĵ	ĵ	Ķ	ķ	ĸ	Ĺ	ĺ	Ł	ł	Ł	ł	Ł
U+0140-014F	Ĺ	ł	Ł	ł	Ń	ń	Ņ	ņ	Ň	ň	ŉ	Đ	đ	Ō	ō	Ŏ
U+0150-015F	Ŏ	ŏ	Œ	œ	Ŕ	ŕ	Ŗ	ŗ	Ř	ř	Ś	ś	Ŝ	ŝ	Ş	ş
U+0160-016F	Š	š	Ţ	ţ	Ť	ť	Ŧ	ŧ	Ũ	ũ	Ū	ū	Ŭ	ŭ	Ů	ů
U+0170-017F	Ů	ů	Ū	ū	Ŵ	ŵ	Ŷ	ŷ	Ÿ	Ž	ž	Ž	ž	Ž	ž	f
<b>Latin Extended-B</b>																
U+0180-018F	ƀ	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
U+01A0-01AF	Œ	œ	-	-	-	-	-	-	-	-	-	-	-	-	-	Ū
U+01B0-01BF	Ƴ	-	-	-	-	Ʒ	-	-	-	-	-	-	-	-	-	-
U+0210-021F	-	-	-	-	-	-	-	-	Ş	ş	Ţ	ţ	-	-	-	-
U+0230-023F	-	-	-	-	-	-	-	Ƶ	-	-	-	-	-	-	-	-
<b>Spacing Modifier Letters</b>																
U+02C0-02CF	-	-	-	-	-	-	^	˘	-	-	-	-	-	-	-	-
U+02D0-02DF	-	-	-	-	-	-	-	˙	˚	˛	˜	˝	-	-	-	-
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

**Table 6:** LM Math vs. NewCM Math, *cont.*

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
<b>Combining Diacritical Marks</b>																
U+0300-030F	˘	˙	˚	ˇ	˘	˙	˚	˛	˜	˝	ˆ	˜	˘	˙	˚	˛
U+0310-031F	◌̂	◌̃	◌̄	◌̅	◌̆	◌̇	◌̈	◌̉	◌̊	◌̋	◌̌	◌̍	◌̎	◌̏	◌̐	◌̑
U+0320-032F	◌̒	◌̓	◌̔	◌̕	◌̖	◌̗	◌̘	◌̙	◌̚	◌̛	◌̜	◌̝	◌̞	◌̟	◌̠	◌̡
U+0330-033F	◌̢	◌̣	◌̤	◌̥	◌̦	◌̧	◌̨	◌̩	◌̪	◌̫	◌̬	◌̭	◌̮	◌̯	◌̰	◌̱
U+0340-034F	◌̲	◌̳	◌̴	◌̵	◌̶	◌̷	◌̸	◌̹	◌̺	◌̻	◌̼	◌̽	◌̾	◌̿	◌̀	◌́
<b>Greek and Coptic</b>																
U+0390-039F	Α	Β	Γ	Δ	Ε	Ζ	Η	Θ	Ι	Κ	Λ	Μ	Ν	Ξ	Ο	
U+03A0-03AF	Π	Ρ	Σ	Τ	Υ	Φ	Χ	Ψ	Ω							
U+03B0-03BF	α	β	γ	δ	ε	ζ	η	θ	ι	κ	λ	μ	ν	ξ	ο	
U+03C0-03CF	π	ρ	ς	σ	τ	υ	φ	χ	ψ	ω						
U+03D0-03DF	ϑ				ϕ	ϖ					Ϝ	ϝ				
U+03F0-03FF	Ϙ	ϙ			Ϡ	ϡ	Ϣ									
<b>Latin Extended Additional</b>																
U+1EA0-1EAF	À	á	Â	â	Ã	ã	Ä	ä	Å	å	Ă	ă	Ą	ą	Ȧ	ȧ
U+1EB0-1EBF	Ⱥ	Ȼ	ȼ	Ƚ	Ⱦ	ȿ	Ⱥ	Ȼ	ȼ	Ƚ	Ⱦ	ȿ	Ⱥ	Ȼ	ȼ	Ƚ
U+1EC0-1ECF	È	é	Ê	ê	Ë	ë	Ĕ	ĕ	Ė	ė	Į	į	İ	ı	Œ	œ
U+1ED0-1EDF	Ó	ó	Ô	ô	Õ	õ	Ö	ö	Û	ü	Ų	ų	Ŵ	ŵ	Ŷ	ŷ
U+1EE0-1EEF	Ÿ	ÿ	Ȫ	ȫ	Ȭ	ȭ	Ȯ	ȯ	Ȱ	ȱ	Ȳ	ȳ	ȴ	ȵ	ȶ	ȷ
U+1EF0-1EFF	ȸ	ȹ	Ⱥ	Ȼ	ȼ	Ƚ	Ⱦ	ȿ	Ⱥ	Ȼ	ȼ	Ƚ	Ⱦ	ȿ	Ⱥ	Ȼ
<b>General Punctuation</b>																
U+2000-200F																
U+2010-201F	-	—	—	—	—		=	‘	’	‚	“	”	„	”	”	”
U+2020-202F	†	‡	•	▶	◦	…	◦	-	-	-	-	-	-	-	-	-
U+2030-203F	‰	‰	′	″	‴	∖	″	‴	◀	◁	▷	※	!!	?	—	◌̣
U+2040-204F	◌̂	◌̃	◌̄	◌̅	◌̆	◌̇	◌̈	◌̉	◌̊	◌̋	◌̌	◌̍	◌̎	◌̏	◌̐	◌̑
U+2050-205F	◌̒	◌̓	◌̔	◌̕	◌̖	◌̗	◌̘	◌̙	◌̚	◌̛	◌̜	◌̝	◌̞	◌̟	◌̠	◌̡
U+2060-206F																
<b>Currency Symbols</b>																
U+20A0-20AF	€													€		
<b>Combining Diacritical Marks for Symbols</b>																
U+20D0-20DF	◌̑	◌̒			◌̑	◌̒	◌̑	◌̒	◌̑	◌̒	◌̑	◌̒	◌̑	◌̒	◌̑	◌̒
U+20E0-20EF	◌̑	◌̒		△	∖		◌̑	◌̒	◌̑	◌̒	◌̑	◌̒	◌̑	◌̒	◌̑	◌̒
U+20F0-20FF	*															
<b>Letterlike Symbols</b>																
U+2100-210F	‰	‰	©	°	℄	‰	‰	ε	∅	°	℄	ℋ	ℋ	ℋ	ℋ	ℋ
U+2110-211F	ℐ	ℐ	ℒ	ℓ	℄	ℕ	№	©	∅	ℙ	ℚ	ℛ	ℜ	ℝ	ℝ	ℝ
U+2120-212F	SM	TEL	TM	℥	ℤ	ℤ	Ω	∪	∩	∩	℔	℔	℔	℔	℔	℔
U+2130-213F	ℰ	ℱ	ℱ	ℳ	ℴ	ℵ	∩	∩	∩	∩	∩	∩	∩	∩	∩	∩
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Table 6: LM Math vs. NewCM Math, cont.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+2140 - 214F	$\Sigma$	$\mathcal{O}$	$\mathcal{T}$	$\mathcal{J}$	$\mathcal{A}$	$\mathcal{D}$	$d$	$e$	$i$	$j$	$\mathcal{E}$	$\mathcal{F}$	$\mathcal{G}$	$\mathcal{H}$	$\mathcal{I}$	$\mathcal{J}$

Arrows

U+2190 - 219F	$\leftarrow$	$\uparrow$	$\rightarrow$	$\downarrow$	$\leftrightarrow$	$\updownarrow$	$\nearrow$	$\searrow$	$\swarrow$	$\nwarrow$	$\Leftrightarrow$	$\Leftrightarrow$	$\curvearrowright$	$\curvearrowleft$	$\leftarrow$	$\uparrow$
U+21A0 - 21AF	$\Rightarrow$	$\Downarrow$	$\Leftarrow$	$\Uparrow$	$\Leftarrow$	$\Uparrow$	$\Downarrow$	$\Downarrow$	$\Leftrightarrow$	$\Leftrightarrow$	$\Leftrightarrow$	$\Leftrightarrow$	$\Leftrightarrow$	$\Leftrightarrow$	$\Leftrightarrow$	$\Downarrow$
U+21B0 - 21BF	$\curvearrowright$	$\curvearrowleft$	$\curvearrowright$	$\curvearrowleft$	$\curvearrowright$	$\curvearrowleft$	$\curvearrowright$	$\curvearrowleft$	$\curvearrowright$	$\curvearrowleft$	$\curvearrowright$	$\curvearrowleft$	$\curvearrowright$	$\curvearrowleft$	$\curvearrowright$	$\curvearrowleft$
U+21C0 - 21CF	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$
U+21D0 - 21DF	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$
U+21E0 - 21EF	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$	$\Leftarrow$
U+21F0 - 21FF	$\Rightarrow$	$\Rightarrow$	$\Rightarrow$	$\Rightarrow$	$\Rightarrow$	$\Rightarrow$	$\Rightarrow$	$\Rightarrow$	$\Rightarrow$	$\Rightarrow$	$\Rightarrow$	$\Rightarrow$	$\Rightarrow$	$\Rightarrow$	$\Rightarrow$	$\Rightarrow$

Mathematical Operators

U+2200 - 220F	$\forall$	$\complement$	$\partial$	$\exists$	$\nexists$	$\emptyset$	$\Delta$	$\nabla$	$\in$	$\notin$	$\in$	$\ni$	$\neq$	$\approx$	$\blacksquare$	$\prod$
U+2210 - 221F	$\prod$	$\Sigma$	$-$	$\mp$	$\div$	$/$	$\backslash$	$*$	$\circ$	$\bullet$	$\sqrt{\quad}$	$\sqrt[3]{\quad}$	$\sqrt[4]{\quad}$	$\propto$	$\infty$	$\perp$
U+2220 - 222F	$\angle$	$\sphericalangle$	$\sphericalangle$	$\sphericalangle$	$\sphericalangle$	$\parallel$	$\nparallel$	$\wedge$	$\vee$	$\cap$	$\cup$	$\int$	$\iint$	$\iiint$	$\$$	$\$$
U+2230 - 223F	$\text{\textcircled{#}}$	$\text{\textcircled{f}}$	$\text{\textcircled{g}}$	$\text{\textcircled{h}}$	$\text{\textcircled{i}}$	$\text{\textcircled{j}}$	$\text{\textcircled{k}}$	$\text{\textcircled{l}}$	$\text{\textcircled{m}}$	$\text{\textcircled{n}}$	$\text{\textcircled{o}}$	$\text{\textcircled{p}}$	$\text{\textcircled{q}}$	$\text{\textcircled{r}}$	$\text{\textcircled{s}}$	$\text{\textcircled{t}}$
U+2240 - 224F	$\sim$	$\approx$	$\approx$	$\approx$	$\approx$	$\approx$	$\approx$	$\approx$	$\approx$	$\approx$	$\approx$	$\approx$	$\approx$	$\approx$	$\approx$	$\approx$
U+2250 - 225F	$\doteq$	$\doteq$	$\doteq$	$\doteq$	$\doteq$	$\doteq$	$\doteq$	$\doteq$	$\doteq$	$\doteq$	$\doteq$	$\doteq$	$\doteq$	$\doteq$	$\doteq$	$\doteq$
U+2260 - 226F	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$
U+2270 - 227F	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$
U+2280 - 228F	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$
U+2290 - 229F	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$
U+22A0 - 22AF	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$
U+22B0 - 22BF	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$
U+22C0 - 22CF	$\wedge$	$\vee$	$\cap$	$\cup$	$\diamond$	$\cdot$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
U+22D0 - 22DF	$\in$	$\ni$	$\ni$	$\ni$	$\ni$	$\ni$	$\ni$	$\ni$	$\ni$	$\ni$	$\ni$	$\ni$	$\ni$	$\ni$	$\ni$	$\ni$
U+22E0 - 22EF	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$	$\neq$
U+22F0 - 22FF	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$	$\cdot$

Miscellaneous Technical

U+2300 - 230F	$\emptyset$	$\leftarrow$	$\triangleleft$	$\hat{\quad}$	$\vee$	$\wedge$	$\approx$	$\{$	$ $	$ $	$ $	$ $	$ $	$ $	$ $	$ $
U+2310 - 231F	$\ulcorner$	$\square$	$\frown$	$\smile$	$\nabla$	$\circ$	$\oplus$	$\#$	$\&$	$\ulcorner$	$\ulcorner$	$\ulcorner$	$\ulcorner$	$\ulcorner$	$\ulcorner$	$\ulcorner$
U+2320 - 232F	$f$	$J$	$\frown$	$\smile$	$\wedge$	$\approx$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\langle$	$\rangle$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$
U+2330 - 233F	$\curvearrowright$	$\curvearrowleft$	$\triangleleft$	$\triangleleft$	$\ulcorner$	$\vee$	$\sqcap$	$\sqcap$	$\sqcap$	$\sqcap$	$\sqcap$	$\sqcap$	$\sqcap$	$\sqcap$	$\sqcap$	$\sqcap$
U+2340 - 234F	$\sphericalangle$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$	$\square$
U+2350 - 235F	$\boxtimes$	$\top$	$\nabla$	$\nabla$	$\nabla$	$\nabla$	$\nabla$	$\nabla$	$\nabla$	$\nabla$	$\nabla$	$\nabla$	$\nabla$	$\nabla$	$\nabla$	$\nabla$
U+2360 - 236F	$\boxtimes$	$\dagger$	$\nabla$	$\star$	$\circ$	$\circ$	$\psi$	$\psi$	$\psi$	$\psi$	$\psi$	$\psi$	$\psi$	$\psi$	$\psi$	$\psi$
U+2370 - 237F	$\boxtimes$	$\forall$	$\mathcal{A}$	$\mathcal{L}$	$\rho$	$\omega$	$\alpha$	$\epsilon$	$\epsilon$	$\epsilon$	$\alpha$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$
U+2380 - 238F	$\boxtimes$	$\underline{a}$	$\underline{aa}$	$\geq a \leq$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$	$\boxtimes$
U+2390 - 239F	$\diamond$	$\diamond$	$\diamond$	$\diamond$	$\diamond$	$\diamond$	$\diamond$	$\diamond$	$\diamond$	$\diamond$	$\diamond$	$\diamond$	$\diamond$	$\diamond$	$\diamond$	$\diamond$
U+23A0 - 23AF	$\ulcorner$	$ $	$ $	$ $	$ $	$ $	$ $	$ $	$ $	$ $	$ $	$ $	$ $	$ $	$ $	$ $
U+23B0 - 23BF	$\ulcorner$	$\ulcorner$	$\ulcorner$	$\ulcorner$	$\ulcorner$	$\ulcorner$	$\ulcorner$	$\ulcorner$	$\ulcorner$	$\ulcorner$	$\ulcorner$	$\ulcorner$	$\ulcorner$	$\ulcorner$	$\ulcorner$	$\ulcorner$

**Table 6:** LM Math vs. NewCM Math, *cont.*

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+23C0 - 23CF																
U+23D0 - 23DF																
U+23E0 - 23EF									10							
U+23F0 - 23FF																
<b>Control Pictures</b>																
U+2420 - 242F	-	-	b	□	-	-	-	-	-	-	-	-	-	-	-	-
<b>Box Drawing</b>																
U+2500 - 250F	-	-			---	---	---	---	---	---	---	---	---	---	---	---
U+2510 - 251F	┌	┌	┌	┌	┌	┌	┌	┌	┌	┌	┌	┌	┌	┌	┌	┌
U+2520 - 252F	┐	┐	┐	┐	┐	┐	┐	┐	┐	┐	┐	┐	┐	┐	┐	┐
U+2530 - 253F	└	└	└	└	└	└	└	└	└	└	└	└	└	└	└	└
U+2540 - 254F	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
U+2550 - 255F	=		┌	┌	┌	┌	┌	┌	┌	┌	┌	┌	┌	┌	┌	┌
U+2560 - 256F	┆	┆	┆	┆	┆	┆	┆	┆	┆	┆	┆	┆	┆	┆	┆	┆
U+2570 - 257F	┌	/	\	X	-		-		-		-		-		-	
<b>Block Elements</b>																
U+2580 - 258F	■	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
U+2590 - 259F	■	▒	▒	▒	▒	▒	▒	▒	▒	▒	▒	▒	▒	▒	▒	▒
<b>Geometric Shapes</b>																
U+25A0 - 25AF	■	□	○	■	▧	▧	▧	▧	▧	▧	▧	▧	▧	▧	▧	▧
U+25B0 - 25BF	▴	▴	▴	▴	▴	▴	▴	▴	▴	▴	▴	▴	▴	▴	▴	▴
U+25C0 - 25CF	▾	▾	▾	▾	▾	▾	▾	▾	▾	▾	▾	▾	▾	▾	▾	▾
U+25D0 - 25DF	◐	◑	◒	◓	◔	◕	◖	◗	◘	◙	◚	◛	◜	◝	◞	◟
U+25E0 - 25EF	◠	◡	◢	◣	◤	◥	◦	◧	◨	◩	◪	◫	◬	◭	◮	◯
U+25F0 - 25FF	◰	◱	◲	◳	◴	◵	◶	◷	◸	◹	◺	◻	◼	◽	◾	◿
<b>Miscellaneous Shapes</b>																
U+2600 - 260F	✳	-	-	-	-	★	★	-	-	☉	-	-	-	-	-	-
U+2610 - 261F	-	-	☒	-	-	-	-	-	-	-	-	-	-	-	-	-
U+2620 - 262F	-	℘	-	-	-	-	-	-	-	-	-	-	-	-	-	-
U+2630 - 263F	-	-	-	-	-	-	-	-	-	☹	☺	☻	☼	☾	☽	-
U+2640 - 264F	♀	-	♂	-	-	-	-	-	-	-	-	-	-	-	-	-
U+2660 - 266F	♠	♥	♦	♣	♠	♥	♦	♣	-	♪	♪	♪	-	♭	♮	♯
U+2670 - 267F	-	-	-	-	-	-	-	-	-	-	-	-	-	-	⊗	-
U+2680 - 268F	◻	◻	◻	◻	◻	◻	◻	◻	◻	◻	◻	◻	◻	◻	◻	◻
U+26A0 - 26AF	-	-	-	-	-	♀	-	-	-	-	○	●	○	⊗	⊙	-
U+26B0 - 26BF	-	-	♀	-	-	-	-	-	-	-	-	-	-	-	-	-
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Table 6: LM Math vs. NewCM Math, *cont.*

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
<b>Dingbats</b>																
U+2710 - 271F	-	-	-	✓	-	-	-	-	-	-	-	-	-	-	-	-
U+2720 - 272F	✠	-	-	-	-	-	✦	-	-	-	⊕	-	-	-	-	-
U+2730 - 273F	-	-	-	-	-	-	✳	-	-	-	-	-	-	✳	-	-
U+2750 - 275F	-	-	-	-	-	-	-	-	-	-		-	-	-	-	-
U+2770 - 277F	-	-	(	)	-	-	-	-	-	-	-	-	-	-	-	-
U+2790 - 279F	-	-	-	-	-	-	-	-	-	-	-	→	-	-	-	-
U+27A0 - 27AF	-	➔	-	-	-	-	-	-	-	-	-	-	-	-	-	-
<b>Miscellaneous Mathematical Symbols-A</b>																
U+27C0 - 27CF	∟	△	⊥	⊙	⊖	∫	∫	∇	∞	∞	+	/	)	\	⊠	⊠
U+27D0 - 27DF	◇	△	∩	∪	∩	∩	∩	∩	∩	∩	∩	∩	∩	∩	∩	∩
U+27E0 - 27EF	◇	◇	◇	◇	□	□	[	]	<	>	⟨	⟩	(	)	(	)
<b>Supplemental Arrows-A</b>																
U+27F0 - 27FF	⇕	⇓	↻	↻	⊕	←	→	↔	⇐	⇒	↔	⇐	⇐	⇐	⇐	↪
<b>Supplemental Arrows-B</b>																
U+2900 - 290F	⇄	⇄	⇄	⇄	⇄	⇄	⇄	⇄	⇄	⇄	⇄	⇄	⇄	⇄	⇄	⇄
U+2910 - 291F	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗
U+2920 - 292F	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗
U+2930 - 293F	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗
U+2940 - 294F	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗
U+2950 - 295F	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗
U+2960 - 296F	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗
U+2970 - 297F	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗	↗
<b>Miscellaneous Mathematical Symbols-B</b>																
U+2980 - 298F		•	⋮	∫	∫	(	)	(	)	(	)	(	)	(	)	(
U+2990 - 299F	∫	<	>	<	>	<	>	<	>	<	>	<	>	<	>	<
U+29A0 - 29AF	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫
U+29B0 - 29BF	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫
U+29C0 - 29CF	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫
U+29D0 - 29DF	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫
U+29E0 - 29EF	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫
U+29F0 - 29FF	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫
<b>Supplemental Mathematical Operators</b>																
U+2A00 - 2A0F	⊙	⊕	⊗	⊙	⊕	∏	∏	∏	∏	∏	∏	∏	∏	∏	∏	∏
U+2A10 - 2A1F	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫
U+2A20 - 2A2F	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫	∫
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

**Table 6: LM Math vs. NewCM Math, cont.**

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+2A30 - 2A3F	×	×	×	⊗	⊗	⊗	⊗	⊗	⊗	⊗	⊗	⊗	⊗	⊗	⊗	⊗
U+2A40 - 2A4F	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂
U+2A50 - 2A5F	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂
U+2A60 - 2A6F	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂
U+2A70 - 2A7F	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂
U+2A80 - 2A8F	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂
U+2A90 - 2A9F	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂
U+2AA0 - 2AAF	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂
U+2AB0 - 2ABF	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂
U+2AC0 - 2ACF	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂
U+2AD0 - 2ADF	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂
U+2AE0 - 2AEF	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂
U+2AF0 - 2AFF	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂

**Miscellaneous Symbols and Arrows**

U+2B00 - 2B0F	↗	↘	↙	↚	↛	↜	↝	↞	↠	↡	↢	↣	↤	↥	↦	↧
U+2B10 - 2B1F	↩	↪	↫	↬	↭	↮	↯	↰	↱	↲	↳	↴	↵	↶	↷	↸
U+2B20 - 2B2F	↹	↺	↻	↼	↽	↾	↿	⇀	⇁	⇂	⇃	⇄	⇅	⇆	⇇	⇈
U+2B30 - 2B3F	⇉	⇊	⇋	⇌	⇍	⇎	⇏	⇐	⇑	⇒	⇓	⇔	⇕	⇖	⇗	⇘
U+2B40 - 2B4F	⇙	⇚	⇛	⇜	⇝	⇞	⇟	⇠	⇡	⇢	⇣	⇤	⇥	⇦	⇧	⇨
U+2B50 - 2B5F	⇩	⇪	⇫	⇬	⇭	⇮	⇯	⇰	⇱	⇲	⇳	⇴	⇵	⇶	⇷	⇸
U+2B60 - 2B6F	⇹	⇺	⇻	⇼	⇽	⇾	⇿	↩	↪	↫	↬	↭	↮	↯	↰	↱
U+2B70 - 2B7F	↲	↳	↴	↵	↶	↷	↸	↹	↺	↻	↼	↽	↾	↿	⇀	⇁
U+2B80 - 2B8F	⇀	⇁	⇂	⇃	⇄	⇅	⇆	⇇	⇈	⇉	⇊	⇋	⇌	⇍	⇎	⇏
U+2B90 - 2B9F	⇐	⇑	⇒	⇓	⇔	⇕	⇖	⇗	⇘	⇙	⇚	⇛	⇜	⇝	⇞	⇟
U+2BA0 - 2BAF	⇠	⇡	⇢	⇣	⇤	⇥	⇦	⇧	⇨	⇩	⇪	⇫	⇬	⇭	⇮	⇯
U+2BB0 - 2BBF	⇰	⇱	⇲	⇳	⇴	⇵	⇶	⇷	⇸	⇹	⇺	⇻	⇼	⇽	⇾	⇿
U+2BC0 - 2BCF	⬛	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜
U+2BD0 - 2BDF	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜
U+2BE0 - 2BEF	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜
U+2BF0 - 2BFF	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜	⬜

**Supplemental Punctuation**

U+2E10 - 2E1F	-	-	-	-	-	-	-	-	↓	-	-	-	-	-	-	-
---------------	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

**CJK Symbols and Punctuation**

U+3010 - 301F	-	-	〒	-	-	-	【	】	-	-	-	-	-	-	-	-
U+3030 - 303F	〜	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

**Private Use Area**

U+E000 - E00F	A	B	Γ	Δ	E	Z	H	Θ	I	K	Λ	M	N	Ξ	O	Π
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

**Table 6:** LM Math vs. NewCM Math, *cont.*

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+E010 - E01F	P	Σ	T	Υ	Φ	X	Ψ	Ω	α	β	γ	δ	ε	ζ	η	θ
U+E020 - E02F	ι	κ	λ	μ	ν	ξ	ο	π	ρ	ς	σ	τ	υ	φ	χ	ψ
U+E030 - E03F	ω	€	⚡	⚡	-	-	-	-	-	-	-	-	-	-	-	-
U+E040 - E04F	-	A	B	Γ	Δ	E	Z	H	Θ	I	K	Λ	M	N	Ξ	O
U+E050 - E05F	Π	P	Σ	T	Υ	Φ	X	Ψ	Ω	α	β	γ	δ	ε	ζ	η
U+E060 - E06F	θ	ι	κ	λ	μ	ν	ξ	ο	π	ρ	ς	σ	τ	υ	φ	χ
U+E070 - E07F	ψ	ω	€	-	-	-	-	-	-	-	-	-	-	-	-	-
U+E370 - E37F	-	-	-	-	-	-	§	§	-	-	-	-	-	-	-	-
U+E390 - E39F	-	-	-	-	-	§	-	§	§	§	⌘	⌘	-	-	-	-
U+E3D0 - E3DF	-	-	-	f	-	-	-	-	-	-	-	-	-	-	-	-
U+EA50 - EA5F	-	-	-	-	-	-	-	⌘	-	-	-	-	-	-	-	-

**Alphabetic Presentation Forms**

U+FB00 - FB0F	ff	fi	fl	ffi	ffl	-	-	-	-	-	-	-	-	-	-	-
---------------	----	----	----	-----	-----	---	---	---	---	---	---	---	---	---	---	---

**Arabic Presentation Forms-B**

U+FEF0 - FEFF	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
---------------	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

**Mathematical Alphanumeric Symbols**

U+1D400 - 1D40F	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
U+1D410 - 1D41F	Q	R	S	T	U	V	W	X	Y	Z	a	b	c	d	e	f
U+1D420 - 1D42F	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v
U+1D430 - 1D43F	w	x	y	z	A	B	C	D	E	F	G	H	I	J	K	L
U+1D440 - 1D44F	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	a	b
U+1D450 - 1D45F	c	d	e	f	g	-	i	j	k	l	m	n	o	p	q	r
U+1D460 - 1D46F	s	t	u	v	w	x	y	z	A	B	C	D	E	F	G	H
U+1D470 - 1D47F	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
U+1D480 - 1D48F	Y	Z	a	b	c	d	e	f	g	h	i	j	k	l	m	n
U+1D490 - 1D49F	o	p	q	r	s	t	u	v	w	x	y	z	A	-	C	D
U+1D4A0 - 1D4AF	-	-	ℒ	-	-	ℒ	ℒ	-	-	N	ℒ	ℒ	ℒ	-	S	T
U+1D4B0 - 1D4BF	U	V	W	X	Y	Z	a	b	c	d	-	f	-	h	i	j
U+1D4C0 - 1D4CF	k	l	m	n	-	p	q	r	s	t	u	v	w	x	y	z
U+1D4D0 - 1D4DF	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
U+1D4E0 - 1D4EF	Q	R	S	T	U	V	W	X	Y	Z	a	b	c	d	e	f
U+1D4F0 - 1D4FF	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v
U+1D500 - 1D50F	w	x	y	z	ℒ	ℒ	-	ℒ	ℒ	ℒ	-	-	ℒ	ℒ	ℒ	ℒ
U+1D510 - 1D51F	ℒ	ℒ	ℒ	ℒ	ℒ	-	ℒ	ℒ	ℒ	ℒ	ℒ	ℒ	ℒ	-	a	b
U+1D520 - 1D52F	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r
U+1D530 - 1D53F	s	t	u	v	w	x	y	z	A	B	-	D	E	F	G	-
U+1D540 - 1D54F	I	J	K	L	M	-	O	-	-	-	S	T	U	V	W	X
U+1D550 - 1D55F	Y	-	a	b	c	d	e	f	g	h	i	j	k	l	m	n
U+1D560 - 1D56F	o	p	q	r	s	t	u	v	w	x	y	z	ℒ	ℒ	ℒ	ℒ
U+1D570 - 1D57F	ℒ	ℒ	ℒ	ℒ	ℒ	ℒ	ℒ	ℒ	ℒ	ℒ	ℒ	ℒ	ℒ	ℒ	ℒ	ℒ
U+1D580 - 1D58F	ℒ	ℒ	ℒ	ℒ	ℒ	ℒ	a	b	c	d	e	f	g	h	i	j
U+1D590 - 1D59F	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F



Table 6: LM Math vs. NewCM Math, *cont.*

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+1D5A0-1D5AF	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
U+1D5B0-1D5BF	Q	R	S	T	U	V	W	X	Y	Z	a	b	c	d	e	f
U+1D5C0-1D5CF	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v
U+1D5D0-1D5DF	w	x	y	z	A	B	C	D	E	F	G	H	I	J	K	L
U+1D5E0-1D5EF	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	a	b
U+1D5F0-1D5FF	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r
U+1D600-1D60F	s	t	u	v	w	x	y	z	A	B	C	D	E	F	G	H
U+1D610-1D61F	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
U+1D620-1D62F	Y	Z	a	b	c	d	e	f	g	h	i	j	k	l	m	n
U+1D630-1D63F	o	p	q	r	s	t	u	v	w	x	y	z	A	B	C	D
U+1D640-1D64F	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
U+1D650-1D65F	U	V	W	X	Y	Z	a	b	c	d	e	f	g	h	i	j
U+1D660-1D66F	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
U+1D670-1D67F	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
U+1D680-1D68F	Q	R	S	T	U	V	W	X	Y	Z	a	b	c	d	e	f
U+1D690-1D69F	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v
U+1D6A0-1D6AF	w	x	y	z	ι	ϵ	-	-	A	B	Γ	Δ	E	Z	H	Θ
U+1D6B0-1D6BF	I	K	Λ	M	N	Ξ	O	Π	P	Θ	Σ	T	Υ	Φ	X	Ψ
U+1D6C0-1D6CF	Ω	∇	α	β	γ	δ	ε	ζ	η	θ	ι	κ	λ	μ	ν	ξ
U+1D6D0-1D6DF	ο	π	ρ	ς	σ	τ	υ	φ	χ	ψ	ω	δ	ε	θ	κ	φ
U+1D6E0-1D6EF	ϱ	Ϙ	A	B	Γ	Δ	E	Z	H	Θ	I	K	Λ	M	N	Ξ
U+1D6F0-1D6FF	O	Π	P	Θ	Σ	T	Υ	Φ	X	Ψ	Ω	∇	α	β	γ	δ
U+1D700-1D70F	ε	ζ	η	θ	ι	κ	λ	μ	ν	ξ	ο	π	ρ	ς	σ	τ
U+1D710-1D71F	υ	φ	χ	ψ	ω	δ	ε	θ	κ	φ	ϱ	Ϙ	A	B	Γ	Δ
U+1D720-1D72F	E	Z	H	Θ	I	K	Λ	M	N	Ξ	O	Π	P	Θ	Σ	T
U+1D730-1D73F	Υ	Φ	X	Ψ	Ω	∇	α	β	γ	δ	ε	ζ	η	θ	ι	κ
U+1D740-1D74F	λ	μ	ν	ξ	ο	π	ρ	ς	σ	τ	υ	φ	χ	ψ	ω	δ
U+1D750-1D75F	ε	θ	κ	φ	ϱ	Ϙ	A	B	Γ	Δ	E	Z	H	Θ	I	K
U+1D760-1D76F	Λ	M	N	Ξ	O	Π	P	Θ	Σ	T	Υ	Φ	X	Ψ	Ω	∇
U+1D770-1D77F	α	β	γ	δ	ε	ζ	η	θ	ι	κ	λ	μ	ν	ξ	ο	π
U+1D780-1D78F	ρ	ς	σ	τ	υ	φ	χ	ψ	ω	δ	ε	θ	κ	φ	ϱ	Ϙ
U+1D790-1D79F	A	B	Γ	Δ	E	Z	H	Θ	I	K	Λ	M	N	Ξ	O	Π
U+1D7A0-1D7AF	P	Θ	Σ	T	Υ	Φ	X	Ψ	Ω	∇	α	β	γ	δ	ε	ζ
U+1D7B0-1D7BF	η	θ	ι	κ	λ	μ	ν	ξ	ο	π	ρ	ς	σ	τ	υ	φ
U+1D7C0-1D7CF	χ	ψ	ω	δ	ε	θ	κ	φ	ϱ	Ϙ	F	f	-	-	0	1
U+1D7D0-1D7DF	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7
U+1D7E0-1D7EF	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3
U+1D7F0-1D7FF	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9

Arabic Mathematical Alphabetic Symbols

U+1EE00-1EE0F	ا	ب	ج	د	هـ	و	ز	ح	ط	ي	ك	ل	م	ن	س	ع
U+1EE10-1EE1F	ف	ص	ق	ر	ش	ت	ث	خ	ذ	ض	ظ	غ	ب	ن	ف	و
U+1EE20-1EE2F	و	ب	ج	-	هـ	-	-	ح	-	ي	ك	ل	م	ن	س	ع
U+1EE30-1EE3F	ف	ص	ق	-	ش	ت	ث	خ	-	ض	-	غ	-	-	-	-
U+1EE40-1EE4F	-	-	ج	-	-	-	-	ح	-	ي	-	ل	-	ن	س	ع
U+1EE50-1EE5F	-	ص	ق	-	ش	-	-	خ	-	ض	-	غ	-	ن	-	و
U+1EE60-1EE6F	-	با	جا	-	ها	-	-	حا	طا	يا	كا	-	ما	نا	سا	عا
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

**Table 6:** LM Math vs. NewCM Math, *cont.*

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
U+1EE70-1EE7F	فا	قا	قا	-	شا	تا	ثا	خا	-	ضا	ظا	غا	نا	-	فا	ه
U+1EE80-1EE8F	ه	به	ج	ه	هه	و	نه	ح	طه	يجه	-	له	م	نه	س	ع
U+1EE90-1EE9F	فه	صه	قه	ره	شه	ته	شه	خ	فه	ظه	غه	-	-	-	-	-
U+1EEA0-1EEAF	-	ب	ج	د	-	و	ز	ح	ط	ي	-	ل	م	ن	س	ع
U+1EEB0-1EEBF	ف	ص	ق	ر	ش	ت	ث	خ	ذ	ض	غ	-	-	-	-	-
U+1EEF0-1EEFF	حد	مح	-	-	-	-	-	-	-	-	-	-	-	-	-	-

**Transport and Map Symbols**

U+1F6D0-1F6DF	-	•	-	-	-	-	-	-	-	-	-	-	-	-	-	-
---------------	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

**Geometric Shapes Extended**

U+1F780-1F78F	◀	▲	▶	▼	•	○	○	○	○	○	⊙	⊙	▪	▪	◻	◻
U+1F790-1F79F	◻	◻	◻	◻	◻	◻	◻	◻	◻	◻	◻	◻	◻	◻	◻	◻
U+1F7A0-1F7AF	◊	+	+	+	+	+	+	+	×	×	×	×	×	×	×	×
U+1F7B0-1F7BF	★	★	★	★	★	*	*	*	*	*	*	*	*	*	*	*
U+1F7C0-1F7CF	^	^	^	^	+	+	+	+	+	*	*	*	*	*	*	*
U+1F7D0-1F7DF	*	*	*	*	*	⊗	⊗	⊗	⊗	-	-	-	-	-	-	-
U+1F7E0-1F7EF	●	●	●	●	●	■	■	■	■	■	■	■	■	■	■	■
U+1F7F0-1F7FF	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	←

**Supplemental Arrows-C**

U+1F800-1F80F	←	↑	→	↓	↔	↕	↔	↕	↔	↕	-	-	-	-	-	-
U+1F810-1F81F	←	↑	→	↓	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕
U+1F820-1F82F	←	↑	→	↓	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕
U+1F830-1F83F	←	↑	→	↓	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕
U+1F840-1F84F	↔	↕	↔	↕	↔	↕	↔	↕	-	-	-	-	-	-	-	-
U+1F850-1F85F	←	↑	→	↓	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕
U+1F860-1F86F	←	↑	→	↓	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕
U+1F870-1F87F	←	↑	→	↓	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕
U+1F880-1F88F	←	↑	→	↓	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕
U+1F890-1F89F	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕
U+1F8A0-1F8AF	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕	↔	↕
U+1F8B0-1F8BF	↔	↕	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Total number of glyphs shown from Latin Modern Math: 2045  
 Comparison font New Computer Modern Math has 0 missing and 1963 extra glyphs

**4.5 Garamond Libre’s Byzantine Musical Symbols**

As a final example we exhibit the Byzantine Musical Symbols as provided by Garamond Libre. Command used:

```
\displayfonttable[range-start=1D000, range-end=1D0FF,
                  hex-digits=block,
                  missing-glyph-color=black!50, color=black!75,
```

statistics-format=Total number of glyphs in  
 this block of #1 is #2  
 {Garamond Libre}

Note that we have altered the text produced by the statistics, because the default is somewhat misleading if only a portion of the font is displayed. This produces the following table:

Table 7: Garamond Libre

Byzantine Musical Symbols																
U+1D000-1D00F	†	‡	ˆ	↗	↘	↙	↘	↙	↘	↙	↘	↙	↘	↙	↘	
U+1D010-1D01F	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	
U+1D020-1D02F	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	
U+1D030-1D03F	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	
U+1D040-1D04F	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	
U+1D050-1D05F	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	
U+1D060-1D06F	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	
U+1D070-1D07F	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	
U+1D080-1D08F	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	
U+1D090-1D09F	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	
U+1D0A0-1D0AF	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	
U+1D0B0-1D0BF	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	
U+1D0C0-1D0CF	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	
U+1D0D0-1D0DF	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	
U+1D0E0-1D0EF	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	
U+1D0F0-1D0FF	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Total number of glyphs in this block of Garamond Libre is 246

### 5 The package implementation

```
1 <*package>
```

By default the package uses coloring to improve the table appearance and therefore requires a color package.

```
2 \RequirePackage{xcolor}
```

```
3 <@@=fmuft>
```

We need the package xparse for specifying the document-level interface commands and l3keys2e to use the expl3 key value methods within L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>. These packages automatically require expl3 so there is no need to load that explicitly. Actually, expl3, l3keys2e and the xparse functionality is now all part of the L<sup>A</sup>T<sub>E</sub>X kernel so the next line is actually not needed at all with a current L<sup>A</sup>T<sub>E</sub>X kernel, but in order to support older installations we keep it for now.

```
4 \RequirePackage{xparse,l3keys2e}
```

Here we introduce the package and specify its version number:

```
5 \ProvidesExplPackage{unicodfonttable}
6     {\unicodfonttabledate}
7     {\unicodfonttableversion}
8     {Producing font tables for Unicode and other fonts}
```

## 5.1 User interface commands

Throughout the implementation we will define a number of keys (and their allowed values). We introduce them at the point where they are used, so they are sprinkled throughout the code.<sup>2</sup>

`\fonttablesetup` To set up user defaults for the keys we provide a standard interface. The command `\unicodefonttablesetup` expects a key/value list and can be called as often as necessary.

```

9 \NewDocumentCommand \fonttablesetup { m }
10 { \keys_set:nn {__fmuft} {#1} \ignorespaces }

```

*(End definition for \fonttablesetup. This function is documented on page 903.)*

`\displayfonttable` The document-level command for generating a font table.

```

11 \NewDocumentCommand\displayfonttable {s O{ } m o}{%
12 \IfBooleanTF #1
13 {

```

For the starred form we preset a number of keys with values suitable when displaying 8-bit legacy fonts. With such fonts Unicode block headers make little sense (as the fonts do not conform to the Unicode layout and since they have at most 265 glyphs). It is therefore also unnecessary to loop over the whole Unicode range of the first plane. If necessary all of them can still be overwritten in the optional argument.

```

14     \__fmuft_display_fonttable:nnn
15     {nostatistics,display-block=none,hex-digits=head,range-end=FF,#2}
16     {#3}{#4}
17   }
18   {
19     \__fmuft_display_fonttable:nnn {#2}{#3}{#4}
20   }
21 }

```

*(End definition for \displayfonttable. This function is documented on page 902.)*

`\__fmuft_display_fonttable:nnn` This command is the main workhorse of the package. It produces a `longtable` containing all font glyphs with 16 glyphs per row. The first optional argument is used to configure the table through key/value pairs, the mandatory argument is the font name to display (in `fontspec` conventions) and the final optional argument is the font feature list if any. If the latter is not provided it will get a special value (`--NoValue--`) assigned by `xparse`, which is something that can be tested for.

```

22 \cs_new:Npn \__fmuft_display_fonttable:nnn #1#2#3 {
23   \group_begin:

```

First initialize the font that should be displayed (perhaps with a feature list) and then update the key/value list using `#1`.

```

24     \fontspec{#2}[#3]
25     \keys_set:nn{__fmuft}{#1}

```

If the Lua<sub>TeX</sub> engine is used without HarfBuzz and the display range includes code points above U+EFFFD the output shows remappings and not what is in the font, so we issue a warning.

```

26     \bool_lazy_and:nnT
27     { \sys_if_engine luatex_p: }

```

<sup>2</sup> This fits with the way this package was developed. I first implemented a single rigid table layout without configuration possibilities and then thought about which parts I wanted to have flexible. I then replaced the rigid code with code that is affected by setting key/value pairs.

```

28     { \int_compare_p:nNn { "EFFF" } < { "\l_fmft_range_end_tl } }
29     {
30       \directlua{token.put_next(token.create(font.getfont(font.current()).hb~
31         and~ 'use_none:n'~ or~ 'use:n'))}
32       { \msg_warning:nn {unicodefonttable}{noharfbuzz} }
33     }

```

If the user has asked for a comparison to some other font we need to set this up:

```

34   \tl_if_empty:NTF \l_fmft_compare_with_tl
35   { \tl_clear:N \l_fmft_compare_font_tl }
36   {
37     \setfontface \l_fmft_compare_font_tl {\l_fmft_compare_with_tl}[]
38     \cs_set_eq:NN \_fmft_handle_missing_glyph:n
39     \_fmft_handle_missing_glyph_compare:n
40   }

```

Typesetting the font tables in twocolumn mode makes little sense due to their width, and if `longtable` is used it will complain. However there is one case where it should work: in a page-wide float. To make this happen we claim that we are not in twocolumn mode if the display is inside a vertical box.

```

41   \if_mode_vertical: \if_mode_inner: \@twocolumnfalse \fi: \fi:

```

Then we start the table with 17 columns. We use `longtable` if we produce a caption and `longtable*` if not (so that the table number is not increased, which would look odd if you have other tables in your document).

```

42   \begin{longtable\bool_if:NF\l_fmft_display_header_bool{*}}
43     {@{}r@{\quad}*{16}{c}@{}}

```

Special headers and footers are set up first:

```

44     \_fmft_setup_header_footer:nn{#2}{#3}

```

Then we produce all table rows with the glyphs.

```

45     \_fmft_produce_table_rows:

```

At the very end we may typeset some statistics. This can't be done in the table footer, because the data is dynamic (e.g., number of glyphs processed) and the table footers are static and do not change based on the table content.

```

46     \_fmft_handle_table_ending:n {#2}
47     \end{longtable\bool_if:NF\l_fmft_display_header_bool{*}}
48   \group_end:
49 }

```

(End definition for `\_fmft_display_fonttable:nnn`.)

```

50 \msg_new:nnn {unicodefonttable}{noharfbuzz}
51   { You~ asked~ for~ displaying~ glyphs~ with~ code \iow_newline:
52     points~ above~ U+EFFF~ \msg_line_context: ,~ i.e.,~ from~ the~
53     'Supplementary~ Private~ Use~ Area-A'\iow_newline:
54     without~ specifying~ '[Renderer=Harfbuzz]~'~ when~
55     loading~ the~ font.
56     \iow_newline:\iow_newline:
57     With~ LuaLaTeX,~ this~ Unicode~ region~ is~ used~
58     for~ remappings~ (if~ the~ HarfBuzz~ engine~ is~ not~ used).~
59     Thus,~ the~ results~ shown~ do~ not~ reflect~ what~
60     is~ in~ the~ font!
61   }

```

```

\fonttableglyphcount
  \g__fmft_glyph_int
\g__fmft_glyph_only_B_int
\g__fmft_glyph_also_B_int

```

While generating the font table we count the number of glyphs we see (and typeset). The total is available in the command `\fonttableglyphcount` after the table got finished and will be reset to zero when the next table starts.

```

62 \DeclareDocumentCommand \fonttableglyphcount {}
63     { \int_use:N \g__fmuft_glyph_int }
64 \int_new:N \g__fmuft_glyph_int

```

When comparing fonts we also record data for the second font: the number of glyphs in both and the number of glyphs only in the second one.

```

65 \int_new:N \g__fmuft_glyph_only_B_int
66 \int_new:N \g__fmuft_glyph_also_B_int

```

(End definition for `\fonttableglyphcount` and others. This function is documented on page 903.)

## 5.2 The overall table layout

`\__fmuft_setup_header_footer:nn` Setting up header and footer lines of the table. This macro receives the *font name* and the *font features* specified by the user as its arguments.

```

67 \cs_new:Npn \__fmuft_setup_header_footer:nn #1#2{

```

On the first page of the table the header may show a caption or some other sort of title based on the value of `\l__fmuft_display_header_bool`. The formatting is handled by `\__fmuft_format_table_title:nn` which can be customized through the key `title-format`.

```

68     \bool_if:NT \l__fmuft_display_header_bool
69     { \__fmuft_format_table_title:nn{#1}{#2} \__fmuft_debug_nl:n{T}\*[6pt] }

```

We may also want to display a line of hex digits. This is controlled through the key `hex-digits` that accepts different values: `head`, `foot`, `head+foo`, `block` (after a block title) or `none`.

```

70     \bool_if:NT \l__fmuft_header_hex_digits_bool
71     { \__fmuft_display_row_of_hex_digits: \__fmuft_debug_nl:n{H}\*      }
72 \endfirsthead

```

Headers for later table pages have a continuation title and maybe a row of hex digits.

```

73     \bool_if:NT \l__fmuft_display_header_bool
74     { \__fmuft_format_table_cont:nn{#1}{#2} \__fmuft_debug_nl:n{T}\*[6pt] }
75     \bool_if:NT \l__fmuft_header_hex_digits_bool
76     { \__fmuft_display_row_of_hex_digits: \__fmuft_debug_nl:n{H}\*      }
77 \endhead

```

Footers of the table are either empty or show a row of hex digits.

```

78     \bool_if:NT \l__fmuft_footer_hex_digits_bool
79     { \__fmuft_display_row_of_hex_digits: \__fmuft_debug_nl:n{H}\*      }
80 \endfoot

```

The footer of the last page of the table will always be empty. Any special row, such as a row of hex digits, will be provided in the table body. The reason is that we may want to display statistics at the very end of the table and those can't be placed into a static footer.

```

81 \endlastfoot
82 }

```

(End definition for `\__fmuft_setup_header_footer:nn`.)

`\l__fmuft_header_hex_digits_bool` Here are the booleans we use in the code.

```

\l__fmuft_footer_hex_digits_bool 83 \bool_new:N \l__fmuft_header_hex_digits_bool
\l__fmuft_blockwise_hex_digits_bool 84 \bool_new:N \l__fmuft_footer_hex_digits_bool
85 \bool_new:N \l__fmuft_blockwise_hex_digits_bool

```

(End definition for `\l__fmuft_header_hex_digits_bool`, `\l__fmuft_footer_hex_digits_bool`, and `\l__fmuft_blockwise_hex_digits_bool`.)

```

\__fmuft_display_row_of_hex_digits: Producing a row of hex digits is simple.
\__fmuft_format_hex_digit:n
86 \cs_new:Npn \__fmuft_display_row_of_hex_digits: {
87   & \__fmuft_format_hex_digit:n{0}   & \__fmuft_format_hex_digit:n{1}
88   & \__fmuft_format_hex_digit:n{2}   & \__fmuft_format_hex_digit:n{3}
89   & \__fmuft_format_hex_digit:n{4}   & \__fmuft_format_hex_digit:n{5}
90   & \__fmuft_format_hex_digit:n{6}   & \__fmuft_format_hex_digit:n{7}
91   & \__fmuft_format_hex_digit:n{8}   & \__fmuft_format_hex_digit:n{9}
92   & \__fmuft_format_hex_digit:n{A}   & \__fmuft_format_hex_digit:n{B}
93   & \__fmuft_format_hex_digit:n{C}   & \__fmuft_format_hex_digit:n{D}
94   & \__fmuft_format_hex_digit:n{E}   & \__fmuft_format_hex_digit:n{F} }

```

Each digit is typeset in typewriter and in script size. We offer font and color customizations. Note that it is important to set an explicit family. Otherwise the hex digits are formatted using the current table font (which may or may not work at all).

```

95 \cs_new:Npn \__fmuft_format_hex_digit:n #1 {
96   \l__fmuft_hex_digits_font_tl \l__fmuft_color_tl #1 }

```

*(End definition for \\_\_fmuft\_display\_row\_of\_hex\_digits: and \\_\_fmuft\_format\_hex\_digit:n.)*

`\l__fmuft_color_tl` The token list to hold definition if set up.

```

97 \tl_new:N \l__fmuft_color_tl

```

*(End definition for \l\_\_fmuft\_color\_tl.)*

**Key setup (overall table)** Here are the definitions for the keys used in the code above:

```

98 \keys_define:nn {__fmuft} {

```

The `header` key is a boolean that determines if a header title should be produced (default)

```

99   ,header .bool_set:N = \l__fmuft_display_header_bool
100   ,header .default:n = true
101   ,header .initial:n = true

```

To ease the setup we also support the key `noheader` which is a short form for `header=false`.

```

102   ,noheader .bool_set_inverse:N = \l__fmuft_display_header_bool
103   ,noheader .default:n = true

```

The default for the `title-format` key is to produce a `\caption` listing the font name and any features (if given). Note the `\IfValueTF` command (provided by `xparse`) that checks if the second argument got any value or has the special `--NoValue--` value.

```

104   ,title-format .cs_set:Np = \__fmuft_format_table_title:nn #1#2
105   ,title-format .initial:n =
106   \IfValueTF{#2} { \caption{ #1~ (features:~ \texttt{\small#2}) } }
107   { \caption{ #1 } }

```

The default continuation title ignores the given features, so the formatting is somewhat simpler. It uses `\caption[]{...}` to make a caption that doesn't alter the table number.

```

108   ,title-format-cont .cs_set:Np = \__fmuft_format_table_cont:nn #1#2
109   ,title-format-cont .initial:n = \caption[]{#1~ \emph{cont.}}

```

The key `hex-digits` is implemented as a choice, where each allowed value sets different booleans that are then used in the code.

```

110   ,hex-digits .choice:
111   ,hex-digits / block .code:n =
112   \bool_set_true:N \l__fmuft_blockwise_hex_digits_bool
113   \bool_set_false:N \l__fmuft_header_hex_digits_bool

```

```

114     \bool_set_false:N \l_fmufn_footer_hex_digits_bool
115 ,hex-digits / foot .code:n =
116     \bool_set_true:N \l_fmufn_footer_hex_digits_bool
117     \bool_set_false:N \l_fmufn_header_hex_digits_bool
118     \bool_set_false:N \l_fmufn_blockwise_hex_digits_bool
119 ,hex-digits / head .code:n =
120     \bool_set_true:N \l_fmufn_header_hex_digits_bool
121     \bool_set_false:N \l_fmufn_footer_hex_digits_bool
122     \bool_set_false:N \l_fmufn_blockwise_hex_digits_bool
123 ,hex-digits / head+foot .code:n =
124     \bool_set_true:N \l_fmufn_header_hex_digits_bool
125     \bool_set_true:N \l_fmufn_footer_hex_digits_bool
126     \bool_set_false:N \l_fmufn_blockwise_hex_digits_bool
127 ,hex-digits / none .code:n =
128     \bool_set_false:N \l_fmufn_header_hex_digits_bool
129     \bool_set_false:N \l_fmufn_footer_hex_digits_bool
130     \bool_set_false:N \l_fmufn_blockwise_hex_digits_bool
131 ,hex-digits .initial:n = head

```

The font for hex digits are set with hex-digits-font.

```

132 ,hex-digits-font .tl_set:N = \l_fmufn_hex_digits_font_tl
133 ,hex-digits-font .initial:n = \ttfamily \scriptsize

```

Customizing the row header (on the left) can be done with this key. Defaults for font, fontsize, and color is set on the outside, but can, of course, be overwritten inside if that is desired.

```

134 ,hex-digits-row-format .cs_set:Np = \_fmufn_format_row_hex_digits:n #1
135 ,hex-digits-row-format .initial:n = U+#1 0 \, - \, #1 F

```

The color key is used in most places that get colored; some have their own key but default to the main color.

```

136 ,color .choice:
137 ,color / none .code:n = \tl_clear:N \l_fmufn_color_tl
138 ,color / unknown .code:n = \tl_set:Nn \l_fmufn_color_tl { \color {#1} }
139 ,color .initial:n = blue
140 }

```

`\_fmufn_handle_table_ending:n` At the end of the table we may want to display a final row of hex digits and perhaps some statistics, i.e., the number of typeset glyphs.

```

141 \cs_new:Npn \_fmufn_handle_table_ending:n #1 {
142   \_fmufn_debug_nl:n{H} \*
143   \bool_if:NT \l_fmufn_footer_hex_digits_bool
144     { \_fmufn_display_row_of_hex_digits: \_fmufn_debug_nl:n{H} \* }
145   \bool_if:NT \l_fmufn_display_statistics_bool
146     { \*[2pt]
147       \multicolumn{17}{l}{ \l_fmufn_stats_font_tl

```

If we do font comparison, we use a different command for displaying statistics and pass more data to it.

```

148   \tl_if_empty:NTF \l_fmufn_compare_with_tl
149     {
150       \_fmufn_format_stats:nn{#1}{\fonttableglyphcount}
151     }
152     {
153       \_fmufn_format_compare_stats:nnnnn{#1}{\fonttableglyphcount}
154       { \l_fmufn_compare_with_tl }

```



The extra arguments are total glyph number in second font, glyphs missing in second font and glyphs only in second font.

```

155             { \int_eval:n { \int_use:N\g_fmft_glyph_also_B_int +
156                   \int_use:N\g_fmft_glyph_only_B_int }
157             }
158             { \int_eval:n { \fonttableglyphcount -
159                   \int_use:N\g_fmft_glyph_also_B_int }
160             }
161             { \int_use:N\g_fmft_glyph_only_B_int }
162         }

```

We don't know exactly how wide the table is (and nor does the user) but one may need to use `\parbox` when formatting the statistic line(s). So we back up a bit (rather random) which allows us to use `\parbox{\linewidth}{...}` in the key without thinking too much about it.

```

163         \hspace*{-3cm}
164     }
165 }
166 }

```

**Key setup (for statistics)** Here are the keys used above. By default we produce statistics.

```

167 \keys_define:nn {__fmft} {
168   ,statistics .bool_set:N = \l__fmft_display_statistics_bool
169   ,statistics .default:n = true
170   ,statistics .initial:n = true

```

the key `nostatistics` is just short for `statistics=false`:

```

171   ,nostatistics .bool_set_inverse:N = \l__fmft_display_statistics_bool
172   ,nostatistics .default:n = true

```

The default font we use is `\normalfont`. Again we need to supply a family to avoid getting the font used in the table body.

```

173   ,statistics-font .tl_set:N = \l__fmft_stats_font_tl
174   ,statistics-font .initial:n = \normalfont\small

```

And here we have the default text. There is only space for a single line. If more text is needed one needs to provide some explicit `\parbox`.

```

175   ,statistics-format .cs_set:Np = \__fmft_format_stats:nn #1#2
176   ,statistics-format .initial:n = Total~ number~ of~ glyphs~ shown~ from~ #1:~#2
177 }

```

*(End definition for `\__fmft_handle_table_ending:n`.)*

`\__fmft_debug_n1:n` While developing the code I had a bit of trouble getting the line endings correct, so I added a little macro that made them visible (displaying its argument in the table margin when the key `debug` is used. By default it does nothing.

```

178 \cs_new:Npn \__fmft_debug_n1:n #1 {}

```

**Key setup (debugging)** This key is really internal and is therefore not documented above (and its behavior may changes over time).

```

179 \keys_define:nn {__fmft} {
180   debug .code:n = \cs_set:Npn \__fmft_debug_n1:n ##1
181             {\rlap{\normalfont\scriptsize \quad ##1}} }

```

*(End definition for `\__fmft_debug_n1:n`.)*

### 5.3 The producing the table content

The body of the table consists of rows with sixteen glyphs each and to produce it we loop through all possible Unicode points starting at U+0000 and ending with U+FFFF. This is implemented with a four-level nested loop that runs through the values 0, 1, ..., F with the current hex value in each of the four positions stored in some variable.

```
\g__fmuft_hex_H_tl \g__fmuft_hex_H_tl is a bit special because, it is initially not zero, but empty, so
\g__fmuft_hex_A_tl that slots in the lower plane are denoted by 4 hex digits. We really only need three
\g__fmuft_hex_B_tl further variables, as the value in the innermost loop can used directly.
\g__fmuft_hex_C_tl
182 \tl_new:N \g__fmuft_hex_H_tl % higher plane
183 \tl_new:N \g__fmuft_hex_A_tl
184 \tl_new:N \g__fmuft_hex_B_tl
185 \tl_new:N \g__fmuft_hex_C_tl
```

(End definition for `\g__fmuft_hex_H_tl` and others.)

```
\c__fmuft_hex_digits_clist Here is the sequence we loop through on each level, except the one for the outer level.
186 \clist_const:Nn\c__fmuft_hex_digits_clist{0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F}
(End definition for \c__fmuft_hex_digits_clist.)
```

```
\__fmuft_produce_table_rows: The overall code layout is then fairly simply:
  \__fmuft_handle_hex_H:n 187 \cs_new:Npn \__fmuft_produce_table_rows: {
  \__fmuft_handle_hex_A:n First to some general initialization
  \__fmuft_handle_hex_B:n
  \__fmuft_handle_hex_C:n 188 \__fmuft_initialize_table_rows:
  \__fmuft_handle_hex_D:n
and then loop we start the loop. The outer level is a bit special as currently Unicode
has only slots allocated in plane 0, 1, 2 and E (well, and F, but that is a private area)
so we loop only over those and instead of 0 we use an empty value. Not covered is the
whole of plane 16 which too is now a private area.
```

```
189 \clist_map_function:nN { { } , 1, 2, E, F } \__fmuft_handle_hex_H:n }
```

Most fonts do not have glyphs in the higher planes, which is why by default we don't loop using a nonempty `\__fmuft_handle_hex_H:n`. But if the user wants to scan and display the higher slots they can by setting `range-end` appropriately.

So after setting `\__fmuft_handle_hex_H:n` we loop over `\c__fmuft_hex_digits_clist` for the next hex digit (which we call “A”).

```
190 \cs_new:Npn \__fmuft_handle_hex_H:n #1 { \tl_gset:Nn\g__fmuft_hex_H_tl{#1}
191 \clist_map_function:NN \c__fmuft_hex_digits_clist \__fmuft_handle_hex_A:n }
```

Handling “A” means storing its value for later use and then start a loop for setting the second (or third on higher planes) hex digits:

```
192 \cs_new:Npn \__fmuft_handle_hex_A:n #1 { \tl_gset:Nn\g__fmuft_hex_A_tl{#1}
193 \clist_map_function:NN \c__fmuft_hex_digits_clist \__fmuft_handle_hex_B:n }
```

Same game for “B” and “C”<sup>3</sup>:

```
194 \cs_new:Npn \__fmuft_handle_hex_B:n #1 { \tl_gset:Nn\g__fmuft_hex_B_tl{#1}
195 \clist_map_function:NN \c__fmuft_hex_digits_clist \__fmuft_handle_hex_C:n }
196 \cs_new:Npn \__fmuft_handle_hex_C:n #1 { \tl_gset:Nn\g__fmuft_hex_C_tl{#1}
197 \clist_map_function:NN \c__fmuft_hex_digits_clist \__fmuft_handle_hex_D:n }
```

<sup>3</sup> Actually this is a white lie. In reality we do a lot of extra stuff when handling “C” so later one we give a second definition for `\__fmuft_handle_hex_C:n` but for understanding the overall picture the simpler one shown here is better.

In the innermost loop we now have the full Unicode number available, so there we have to decide what to do with it. This is done by `\__fmuft_handle_hex_D:n` that receives the full number, e.g., 1A7C or 1AD00, as its argument.

```

198 \cs_new:Npn \__fmuft_handle_hex_D:n #1 {
199   \__fmuft_handle_slot:x
200     { " \g__fmuft_hex_H_tl \g__fmuft_hex_A_tl
201       \g__fmuft_hex_B_tl \g__fmuft_hex_C_tl #1 }
202 }

```

*(End definition for \\_\_fmuft\_produce\_table\_rows: and others.)*

`\g__fmuft_row_tl` We first collect the glyphs for a whole row before deciding to typeset it, because if the row is entirely empty we want to omit it. The data for the row is collected slot by slot and the typesetting information (the glyph or the indication for a missing glyph) is appended to `\g__fmuft_row_tl`.

```

203 \tl_new:N \g__fmuft_row_tl

```

*(End definition for \g\_\_fmuft\_row\_tl.)*

`\__fmuft_handle_slot:n` If the current slot number under inspection contains a glyph in our font we want to typeset it. But we don't do this immediately, instead we build up the whole row and typeset it later. We therefore append a `&` and the glyph (including the necessary formatting) to the token list `\g__fmuft_row_tl`.

`\__fmuft_handle_slot:x`

```

204 \cs_new:Npn \__fmuft_handle_slot:n #1 {
205   \__fmuft_if_uchar_exists:nTF { #1 }
206     { \tl_gput_right:Nn \g__fmuft_row_tl
207       { & \__fmuft_format_glyph:n { \symbol{#1} } } }

```

We then increment the overall glyph count and record that we have seen at least one glyph in the current row. There is not much point in displaying rows that are completely empty; indeed, we'd end up with extremely large tables which are mostly empty.

```

208   \int_gincr:N \g__fmuft_glyph_int
209   \bool_gset_true:N \g__fmuft_glyph_seen_bool

```

If we do font comparison we also check if the glyph is in the second font and if so record that fact.

```

210   \tl_if_empty:NF \l__fmuft_compare_font_tl
211     {
212       \group_begin:
213         \l__fmuft_compare_font_tl
214         \__fmuft_if_uchar_exists:nTF { #1 }
215           { \int_gincr:N \g__fmuft_glyph_also_B_int }
216       \group_end:
217     }
218 }

```

If the current slot has no glyph in the font we also add a `&` followed by something that indicates the glyph is missing. If we do font comparison, it may show the glyph from the other font (if it exists there) in some special way to indicate which glyph should be in this slot.

```

219   { \__fmuft_handle_missing_glyph:n {#1} }
220 }
221 \cs_generate_variant:Nn \__fmuft_handle_slot:n {x}

```

*(End definition for \\_\_fmuft\_handle\_slot:n.)*

`\_fmuft_handle_missing_glyph:n` In the standard case we typeset a special symbol to indicate that the glyph is missing.  
`\_fmuft_handle_missing_glyph_std:n` For this case we provide some customization through keys: `\l_fmufmt_missing_g-`  
`\_fmuft_handle_missing_glyph_compare:n` `lyph_tl` holds the symbol for a missing glyph (default: a hyphen). It is typeset in a specific color and we allow for setting it in a special font. The actual symbol number in `#1` is not needed in this scenario.

```

222 \cs_new:Npn \_fmuft_handle_missing_glyph_std:n #1 {
223   \tl_gput_right:Nn \g_fmufmt_row_tl
224     { &
225       \_fmuft_format_glyph:n {
226         % \colorbox{black!30} % <--- provide interface
227           {\l_fmufmt_missing_glyph_color_tl
228             \l_fmufmt_missing_glyph_font_tl
229             \l_fmufmt_missing_glyph_tl }
230       }
231     }
232 }

```

**Key setup (missing glyphs)** Here are the keys for customizing the missing glyph representation.

```

233 \keys_define:nn {_fmufmt} {
234   missing-glyph-color .choice:
235     ,missing-glyph-color / none .code:n =
236       \tl_clear:N \l_fmufmt_missing_glyph_color_tl
237     ,missing-glyph-color / unknown .code:n =
238       \tl_set:Nn \l_fmufmt_missing_glyph_color_tl { \color {#1} }
239   %
240     ,missing-glyph-font .tl_set:N = \l_fmufmt_missing_glyph_font_tl
241     ,missing-glyph-font .initial:n = \ttfamily \scriptsize
242     ,missing-glyph .tl_set:N = \l_fmufmt_missing_glyph_tl
243     ,missing-glyph .initial:n = - }

```

The default definition for the color is to use the same as the one specified by the `color` key. We therefore define the default outside of the `l3keys` method.

```

244 \tl_new:N \l_fmufmt_missing_glyph_color_tl
245 \tl_set:Nn \l_fmufmt_missing_glyph_color_tl {\l_fmufmt_color_tl}

```

This is the version that handles a missing glyph by checking the `compare-with font` to see if that font contains the glyph. If yes, the substitute glyph will be typeset, otherwise the missing glyph symbol is shown by calling `\_fmuft_handle_missing_glyph_std:n`.

```

246 \cs_new:Npn \_fmuft_handle_missing_glyph_compare:n #1 {
247   \group_begin:

```

Locally switch to the other font, then check for the glyph:

```

248     \l_fmufmt_compare_font_tl
249     \_fmuft_if_uchar_exists:nTF { #1 }
250     {

```

If available, format it (together with the `&`) but use a special color and perhaps a background color.

```

251         \tl_gput_right:Nn \g_fmufmt_row_tl
252         { &
253           \_fmuft_format_glyph:n
254             { \l_fmufmt_compare_bgcolor_tl { \l_fmufmt_compare_color_tl
255               \l_fmufmt_compare_font_tl
256               \symbol {#1} }
257           }
258         }

```

Having seen a glyph only in the second font we record this fact.

```
259 \int_gincr:N \g_fmuf_t_glyph_only_B_int
```

Also tell the algorithm that we have seen a glyph to typeset. If we don't do this then a row consisting of only substitute glyphs is not typeset. However, we don't update the glyph count, because this is not a glyph from the main font we display.

```
260 \bool_gset_true:N \g_fmuf_t_glyph_seen_bool
261 }
```

If the alternate font doesn't have the glyph either we typeset the missing glyph symbol.

```
262 { \_fmuf_t_handle_missing_glyph_std:n {} }
263 \group_end:
264 }
```

**Key setup (comparison)** In order to display glyphs from a secondary font we need a secondary color for the glyph itself and possibly some background color.

```
265 \tl_new:N \l_fmuf_t_compare_with_tl
266 \tl_new:N \l_fmuf_t_compare_color_tl
267 \tl_new:N \l_fmuf_t_compare_bgcolor_tl
268 \keys_define:nn {_fmuf_t}
269 {
270 ,compare-with .tl_set:N = \l_fmuf_t_compare_with_tl
271 ,compare-with .initial:n =
272 ,compare-color .choice:
273 ,compare-color / none .code:n
274 = \tl_clear:N \l_fmuf_t_compare_color_tl
275 ,compare-color / unknown .code:n
276 = \tl_set:Nn \l_fmuf_t_compare_color_tl { \color {#1} }
277 ,compare-color .initial:n = red
278 ,compare-bgcolor .choice:
279 ,compare-bgcolor / none .code:n
280 = \tl_clear:N \l_fmuf_t_compare_bgcolor_tl
281 ,compare-bgcolor / unknown .code:n
282 = \tl_set:Nn \l_fmuf_t_compare_bgcolor_tl { \colorbox {#1} }
283 ,compare-bgcolor .initial:n = black!10
```

If we run a comparison we show different statistics that have their own key.

```
284 ,statistics-compare-format .cs_set:Np
285 = \_fmuf_t_format_compare_stats:nnnnn #1#2#3#4#5#6
286 ,statistics-compare-format .initial:n
287 = \parbox{\linewidth}{
288 Total~ number~ of~ glyphs~ shown~ from~ \texttt{#1}:~#2\
289 Comparison~ font~ \texttt{#3}~ has~ #5~ missing~ and~ #6~
290 extra~ glyphs}
291 }
```

By default, i.e., if no font for comparison has been specified, we handle missing glyphs by displaying a missing glyph symbol.

```
292 \cs_new_eq:NN \_fmuf_t_handle_missing_glyph:n
293 \_fmuf_t_handle_missing_glyph_std:n
```

(End definition for `\_fmuf_t_handle_missing_glyph:n`, `\_fmuf_t_handle_missing_glyph_std:n`, and `\_fmuf_t_handle_missing_glyph_compare:n`.)

`\_fmuf_t_format_glyph:n` Every glyph is typeset in a box of equal width with the glyph centered and if necessary protruding on both sides.

```
294 \cs_new:Npn \_fmuf_t_format_glyph:n #1 {
295 \hbox_to_wd:nn {\l_fmuf_t_glyph_box_dim} { \hss #1 \hss } }
```

**Key setup (glyph typesetting)** The key to customize the width. The 6pt are fine for most cases.

```

296 \dim_new:N\l__fmuft_glyph_box_dim
297 \keys_define:nn {__fmuft} {
298   glyph-width .dim_set:N = \l__fmuft_glyph_box_dim
299   ,glyph-width .initial:n = 6pt
300 }

```

*(End definition for \\_\_fmuft\_format\_glyph:n.)*

`\__fmuft_if_uchar_exists:n` For testing whether or not a slot position contains a glyph we need to resort to low-level methods, because so far `expl3` doesn't offer an interface.

```

301 \prg_set_conditional:Npnn \__fmuft_if_uchar_exists:n #1 { TF , T }
302 { \tex_iffontchar:D \tex_font:D #1 \scan_stop:
303   \prg_return_true:
304   \else:
305   \prg_return_false:
306   \fi:
307 }

```

*(End definition for \\_\_fmuft\_if\_uchar\_exists:n.)*

#### 5.4 Handling a single row

`\__fmuft_handle_hex_C:n` As promised here is the read definition for `\__fmuft_handle_hex_C:n` in all its glory.

```

308 \cs_set:Npn \__fmuft_handle_hex_C:n #1 {

```

We are now at the start of a new row (but with the last row not yet typeset) and this last row may need a Unicode block heading before it. This is the reason why we have to delay the typesetting, because in case the line doesn't contain any glyphs we want to typeset neither and that is only known after all slots in the row have been processed.

```

309   \__fmuft_maybe_typeset_a_row_and_display_a_block_title:

```

We then store away the value for the third hex digit (denoted as C) in order to start with the next row.

```

310   \tl_gset:Nn\g__fmuft_hex_C_tl{#1}

```

Being at the start of a new row we might be at the start of a new Unicode block. If so we have to update the block title to add in front of the row when we typeset it (or in front of one of the next rows if the first rows in the is block have no glyphs). If we are still in the same block no update happens.

```

311   \__fmuft_update_block_title:n { \g__fmuft_hex_H_tl
312                                   \g__fmuft_hex_A_tl
313                                   \g__fmuft_hex_B_tl
314                                   \g__fmuft_hex_C_tl }

```

We now check if this row is within the requested range, i.e., greater than or equal to `\l__fmuft_range_start_tl` and not greater than `\l__fmuft_range_end_tl`.

```

315   \int_compare:nNnF
316     { " \g__fmuft_hex_H_tl \g__fmuft_hex_A_tl
317       \g__fmuft_hex_B_tl \g__fmuft_hex_C_tl 0 }
318   < { "\l__fmuft_range_start_tl }
319   {
320     \int_compare:nNnTF
321       { " \g__fmuft_hex_H_tl \g__fmuft_hex_A_tl
322         \g__fmuft_hex_B_tl \g__fmuft_hex_C_tl 0 }
323     > { "\l__fmuft_range_end_tl }

```

If we are past the `end-range` we break out the clist mapping, to avoid unnecessary repetition. This should be propagated back to the outer clists as well (not done).

```
324     { \clist_map_break: }
```

If we are within range we process the slots in the row by first initializing `\g__fmuft_row_tl` with the row title (the info on the left) and then loop through all slots the row to append glyphs (or missing glyphs) to `\g__fmuft_row_tl` to build up everything we need to finally typeset it.

```
325     {
326       \tl_gset:Nx \g__fmuft_row_tl
327       {
328         \exp_not:N \__fmuft_format_row_title:n
329         { \g__fmuft_hex_H_tl \g__fmuft_hex_A_tl
330           \g__fmuft_hex_B_tl \g__fmuft_hex_C_tl }
331       }
332       \clist_map_function:NN \c__fmuft_hex_digits_clist
333         \__fmuft_handle_hex_D:n
334     }
335 }
336 }
```

*(End definition for `\__fmuft_handle_hex_C:n`.)*

`\__fmuft_format_row_title:n` The function to format the row title on the left, as used above.

```
337 \cs_new:Npn \__fmuft_format_row_title:n #1 {
338   \texttt { \footnotesize \l__fmuft_color_tl \__fmuft_format_row_hex_digits:n {#1} }
339 }
```

*(End definition for `\__fmuft_format_row_title:n`.)*

**Key setup (ranges)** For the range we have two keys, its start and the end. By default the whole range from 0 to FFFF is processed.

```
340 \tl_new:N \l__fmuft_range_start_tl
341 \tl_new:N \l__fmuft_range_end_tl
342 \keys_define:nn {__fmuft}
343   {
344     ,range-start .tl_set:N = \l__fmuft_range_start_tl
345     ,range-start .initial:n = 0000
346     ,range-end   .tl_set:N = \l__fmuft_range_end_tl
347     ,range-end   .initial:n = FFFF
348   }
```

`\__fmuft_maybe_typeset_a_row_and_display_a_block_title:` The function handles the just-finished row and, if the row does not consist only of missing glyphs, typesets it. If necessary it also typesets a Unicode block name first.

```
349 \cs_new:Npn \__fmuft_maybe_typeset_a_row_and_display_a_block_title: {
```

We first check if the row had any real glyphs.

```
350   \bool_if:NTF \g__fmuft_glyph_seen_bool
351   {
```

If the row needs typesetting the fun part starts. We first look at the content of `\g__fmuft_block_title_tl`.

```
352     \tl_if_empty:NTF \g__fmuft_block_title_tl
353     {
```

It is empty we are in the middle of a block and we can ignore the Unicode title. However, we have to see if the previous row (or several) was missing (i.e., contained no

glyphs). In that case we leave a little extra space, otherwise we just finish the previous row

```

354         \bool_if:NTF \g__fmuft_row_missing_bool
355         { \__fmuft_debug_nl:n{A}\[6pt] }
356         { \__fmuft_debug_nl:n{B}\[4pt] }
357     }
358     {

```

Otherwise we first have to typeset the Unicode block title (or whatever should happen instead).

```

359         \typeout{ Processing~ \tl_use:N \g__fmuft_block_title_tl }
360         \bool_if:NTF \l__fmuft_display_block_bool
361         {

```

If we are to typeset the title the action depends a bit on whether we are at the very first row or typesetting a later block.

```

362         \bool_if:NTF \g__fmuft_first_row_bool
363         {
364             \bool_gset_false:N \g__fmuft_first_row_bool
365             \__fmuft_debug_nl:n{C}\[-4pt]
366         }
367         {
368             \__fmuft_debug_nl:n{D}\[8pt]
369             \noalign{\vskip 1pt plus 1pt} % space above block: customizable?
370         }
371 % \noalign{\smallskip} % space above block: customizable?
372 \multicolumn{17}{c}{\normalfont \bfseries
373                 \tl_use:N \g__fmuft_block_title_tl}

```

After the block title is typeset we may want to add a row of hex digits as well if that was requested, otherwise we only leave a bit of extra space.

```

374         \bool_if:NTF \l__fmuft_blockwise_hex_digits_bool
375         { \__fmuft_debug_nl:n{E}\[*]
376           \__fmuft_display_row_of_hex_digits:
377           \__fmuft_debug_nl:n{H}\[*[2pt]
378         }
379         { \__fmuft_debug_nl:n{F}\[*[2pt] }
380     }
381     {

```

If the Unicode block title is not typeset we may still have to do something special and again it differs if we at the very beginning of the table (because there we do nothing except changing the state of `\g__fmuft_first_row_bool`).

```

382         \bool_if:NTF \g__fmuft_first_row_bool
383         { \bool_gset_false:N \g__fmuft_first_row_bool }
384         {
385             \__fmuft_debug_nl:n{G~ (new~ block)}
386             \l__fmuft_display_block_action_tl
387         }
388     }

```

Once we are past the block title we clear it, so that it is not retypeset before the next row.

```

389         \tl_gclear:N \g__fmuft_block_title_tl
390     }

```

The final action is to typeset the row and reset the booleans (in case they were true; if they are false already then we do this unnecessarily, but that is probably faster than testing first).



```

391     \bool_gset_false:N \g__fmuft_glyph_seen_bool
392     \bool_gset_false:N \g__fmuft_row_missing_bool
393     \tl_use:N \g__fmuft_row_tl
394   }

```

Current row had no glyphs; remember that fact, and that is all we have to do in that case.

```

395   {
396     \bool_gset_true:N \g__fmuft_row_missing_bool
397   }
398 }

```

*(End definition for \\_\_fmuft\_maybe\_typeset\_a\_row\_and\_display\_a\_block\_title:.)*

## 5.5 Initialisation at the start of the table

`\g__fmuft_first_row_bool`  
`\g__fmuft_glyph_seen_bool`  
`\g__fmuft_row_missing_bool`

Declare the three booleans used in the code below. They will tell us answers to the following questions:

- Are we processing the first row?
- Have we seen any glyph so far (in the current row)?
- Did we have one or more missing rows recently?

```

399 \bool_new:N \g__fmuft_first_row_bool
400 \bool_new:N \g__fmuft_glyph_seen_bool
401 \bool_new:N \g__fmuft_row_missing_bool

```

*(End definition for \g\_\_fmuft\_first\_row\_bool, \g\_\_fmuft\_glyph\_seen\_bool, and \g\_\_fmuft\_row\_missing\_bool.)*

`\__fmuft_initialize_table_rows:` At the start of a table we are processing the first row and so we (obviously) haven't seen a glyph yet and there wasn't a missing row recently.

```

402 \cs_new:Npn \__fmuft_initialize_table_rows: {
403   \bool_gset_true:N \g__fmuft_first_row_bool
404   \bool_gset_false:N \g__fmuft_glyph_seen_bool
405   \bool_gset_false:N \g__fmuft_row_missing_bool

```

And clearly the glyph count for the font(s) is zero.

```

406   \int_gzero:N \g__fmuft_glyph_int
407   \int_gzero:N \g__fmuft_glyph_only_B_int
408   \int_gzero:N \g__fmuft_glyph_also_B_int
409 }

```

*(End definition for \\_\_fmuft\_initialize\_table\_rows:.)*

## 5.6 Handling block titles

`g__fmuft_block_title_tl` We keep the current block title in this token list.

```

410 \tl_new:N \g__fmuft_block_title_tl

```

*(End definition for g\_\_fmuft\_block\_title\_tl.)*

`\__fmuft_update_block_title:n` A block title is updated when the hex digits A,B,C have a certain value, so this is nothing more than a huge case switch.

```

411 \cs_new:Npn \__fmuft_update_block_title:n #1 {
412   \tl_gset:Nx \g__fmuft_block_title_tl {
413     \int_case:nnF{ "#1 }
414     {
415       { "000 }{ Basic~ Latin }
416       { "008 }{ Latin-1~ Supplement }

```

```

417 { "010 }{ Latin~ Extended-A }
418 { "018 }{ Latin~ Extended-B }
419 { "025 }{ IPA~ Extensions }
420 { "02B }{ Spacing~ Modifier~ Letters }
421 { "030 }{ Combining~ Diacritical~ Marks }
422 { "037 }{ Greek~ and~ Coptic }
423 { "040 }{ Cyrillic }
424 { "053 }{ Armenian }
425 { "059 }{ Hebrew }
426 { "060 }{ Arabic }
427 { "070 }{ Syriac }
428 { "075 }{ Arabic~ Supplement }
429 { "078 }{ Thaana }
430 { "07C }{ NKo }
431 { "090 }{ Devanagari }
432 { "098 }{ Bengali }
433 { "0A0 }{ Gurmukhi }
434 { "0A8 }{ Gujarati }
435 { "0B0 }{ Oriya }
436 { "0B8 }{ Tamil }
437 { "0C0 }{ Telugu }
438 { "0C8 }{ Kannada }
439 { "0D0 }{ Malayalam }
440 { "0D8 }{ Sinhala }
441 { "0E0 }{ Thai }
442 { "0E8 }{ Lao }
443 { "0F0 }{ Tibetan }
444 { "100 }{ Myanmar }
445 { "10A }{ Georgian }
446 { "110 }{ Hangul~ Jamo }
447 { "120 }{ Ethiopic }
448 { "138 }{ Ethiopic~ Supplement }
449 { "13A }{ Cherokee }
450 { "140 }{ Unified~ Canadian~ Aboriginal~ Syllabics }
451 { "168 }{ Ogham }
452 { "16A }{ Runic }
453 { "170 }{ Tagalog }
454 { "172 }{ Hanunoo }
455 { "174 }{ Buhid }
456 { "176 }{ Tagbanwa }
457 { "178 }{ Khmer }
458 { "180 }{ Mongolian }
459 { "190 }{ Limbu }
460 { "195 }{ Tai~ Le }
461 { "198 }{ New~ Tai~ Le }
462 { "19E }{ Khmer~ Symbols }
463 { "1A0 }{ Buginese }
464 { "1B0 }{ Balinese }
465 { "1D0 }{ Phonetic~ Extensions }
466 { "1D8 }{ Phonetic~ Extensions~ Supplement }
467 { "1DC }{ Combining~ Diacritical~ Marks~ Supplement }
468 { "1E0 }{ Latin~ Extended~ Additional }
469 { "1F0 }{ Greek~ Extended }
470 { "200 }{ General~ Punctuation }
471 { "207 }{ Superscripts~ and~ Subscripts }
472 { "20A }{ Currency~ Symbols }
473 { "20D }{ Combining~ Diacritical~ Marks~ for~ Symbols }
474 { "210 }{ Letterlike~ Symbols }
475 { "215 }{ Number~ Forms }

```

```

476 { "219 }{ Arrows }
477 { "220 }{ Mathematical~ Operators }
478 { "230 }{ Miscellaneous~ Technical }
479 { "240 }{ Control~ Pictures }
480 { "244 }{ Optical~ Character~ Recognition }
481 { "246 }{ Enclosed~ Alphanumerics }
482 { "250 }{ Box~ Drawing }
483 { "258 }{ Block~ Elements }
484 { "25A }{ Geometric~ Shapes }
485 { "260 }{ Miscellaneous~ Shapes }
486 { "270 }{ Dingbats }
487 { "27C }{ Miscellaneous~ Mathematical~ Symbols-A }
488 { "27F }{ Supplemental~ Arrows-A }
489 { "280 }{ Braille~ Patterns }
490 { "290 }{ Supplemental~ Arrows-B }
491 { "298 }{ Miscellaneous~ Mathematical~ Symbols-B }
492 { "2A0 }{ Supplemental~ Mathematical~ Operators }
493 { "2B0 }{ Miscellaneous~ Symbols~ and~ Arrows }
494 { "2C0 }{ Glagolitic }
495 { "2C6 }{ Latin~ Extended-C }
496 { "2C8 }{ Coptic }
497 { "2D0 }{ Georgian~ Supplement }
498 { "2D3 }{ Tifinagh }
499 { "2D8 }{ Ethiopic~ Extended }
500 { "2E0 }{ Supplemental~ Punctuation }
501 { "2E8 }{ CJK~ Radicals~ Supplement }
502 { "2F0 }{ Kangxi~ Radicals }
503 { "2FF }{ Ideographic~ Description~ Characters }
504 { "300 }{ CJK~ Symbols~ and~ Punctuation }
505 { "304 }{ Hiragana }
506 { "30A }{ Katakana }
507 { "310 }{ Bopomofo }
508 { "313 }{ Hangul~ Compatibility~ Jamo }
509 { "319 }{ Kanbun }
510 { "31A }{ Bopomofo~ Extended }
511 { "31C }{ CJK~ Strokes }
512 { "31F }{ Katakana~ Phonetic~ Extensions }
513 { "320 }{ Enclosed~ CJK~ Letters~ and~ Months }
514 { "330 }{ CJK~ Compatibility }
515 { "4DC }{ Yijing~ Hexagram~ Symbols }
516 { "A00 }{ Yi~ Syllables }
517 { "A49 }{ Yi~ Radicals }
518 { "A70 }{ Modifier~ Tone~ Letters }
519 { "A72 }{ Latin~ Extended-D }
520 { "A80 }{ Syloti~ Nagri }
521 { "A84 }{ Phags-pa }
522 { "A88 }{ Saurashtra }
523 { "A8E }{ Devanagari Extended }
524 { "A90 }{ Kayah Li }
525 { "A93 }{ Rejang }
526 { "A96 }{ Hangul Jamo Extended-A }
527 { "A98 }{ Javanese }
528 { "A9E }{ Myanmar Extended-B }
529 { "AA0 }{ Cham }
530 { "AA6 }{ Myanmar Extended-A }
531 { "AA8 }{ Tai Viet }
532 { "AAE }{ Meetei Mayek Extensions }
533 { "AB0 }{ Ethiopic Extended-A }
534 { "AB3 }{ Latin Extended-E }

```

```

535     { "AB7 }{ Cherokee Supplement }
536     { "ABC }{ Meetei Mayek }
537     { "ACO }{ Hangul Syllables }
538     { "D7B }{ Hangul Jamo Extended-B }
539     { "D80 }{ High Surrogates }
540     { "DB8 }{ High Private Use Surrogates }
541     { "DC0 }{ Low Surrogates }
542     { "E00 }{ Private~ Use~ Area }
543     { "F90 }{ CJK~ Compatibility~ Ideographs }
544     { "FB0 }{ Alphabetic~ Presentation~ Forms }
545     { "FB5 }{ Arabic~ Presentation~ Forms-A }
546     { "FE0 }{ Variation~ Selectors }
547     { "FE1 }{ Vertical~ Forms }
548     { "FE2 }{ Combining~ Half~ Marks }
549     { "FE3 }{ CJK~ Compatibility~ Forms }
550     { "FE5 }{ Small~ Form~ Variants }
551     { "FE7 }{ Arabic~ Presentation~ Forms-B }
552     { "FF0 }{ Halfwidth~ and Fullwidth~ Forms }
553     { "FFF }{ Specials~ ... }
554 %% ... Plane 1 ...
555     { "1000 }{ Linear~ B~ Syllabary }
556     { "1008 }{ Linear~ B~ Ideograms }
557     { "1010 }{ Aegean~ Numbers }
558     { "1014 }{ Ancient~ Greek~ Numbers }
559     { "1019 }{ Ancient~ Symbols }
560     { "101D }{ Phaistos~ Disc }
561     { "1028 }{ Lycian }
562     { "102A }{ Carian }
563     { "102E }{ Coptic~ Epact~ Numbers }
564     { "1030 }{ Old~ Italic }
565     { "1033 }{ Gothic }
566     { "1035 }{ Old~ Permic }
567     { "1038 }{ Ugaritic }
568     { "103A }{ Old~ Persian }
569     { "1040 }{ Deseret }
570     { "1045 }{ Shavian }
571     { "1048 }{ Osmanya }
572     { "104B }{ Osage }
573     { "1050 }{ Elbasan }
574     { "1053 }{ Caucasian~ Albanian }
575     { "1060 }{ Linear~ A }
576     { "1080 }{ Cypriot~ Syllabary }
577     { "1084 }{ Imperial~ Aramaic }
578     { "1086 }{ Palmyrene }
579     { "1088 }{ Nabataean }
580     { "108E }{ Hatran }
581     { "1090 }{ Phoenician }
582     { "1092 }{ Lydian }
583     { "1098 }{ Meroitic~ Hieroglyphs }
584     { "109A }{ Meroitic~ Cursive }
585     { "10A0 }{ Kharoshthi }
586     { "10A6 }{ Old~ South~ Arabian }
587     { "10A8 }{ Old~ North~ Arabian }
588     { "10AC }{ Manichaean }
589     { "10B0 }{ Avestan }
590     { "10B4 }{ Inscriptional~ Parthian }
591     { "10B6 }{ Inscriptional~ Pahlavi }
592     { "10B8 }{ Psalter~ Pahlavi }
593     { "10C0 }{ Old~ Turkic }

```

```

594 { "10C8 }{ Old~ Hungarian }
595 { "10E6 }{ Rumi~ Numeral~ Symbols }
596 { "1100 }{ Brahmi }
597 { "1108 }{ Kaithi }
598 { "110D }{ Sora~ Sompeng }
599 { "1110 }{ Chakma }
600 { "1115 }{ Mahajani }
601 { "1118 }{ Sharada }
602 { "111E }{ Sinhala~ Archaic~ Numbers }
603 { "1120 }{ Khojki }
604 { "1128 }{ Multani }
605 { "112B }{ Khudawadi }
606 { "1130 }{ Grantha }
607 { "1140 }{ Newa }
608 { "1148 }{ Tirhuta }
609 { "1158 }{ Siddham }
610 { "1160 }{ Modi }
611 { "1166 }{ Mongolian~ Supplement }
612 { "1168 }{ Takri }
613 { "1170 }{ Ahom }
614 { "118A }{ Warang~ Citi }
615 { "11A0 }{ Zanabazar~ Square }
616 { "11A5 }{ Soyombo }
617 { "11AC }{ Pau~ Cin~ Hau }
618 { "11C0 }{ Bhaiksuki }
619 { "11C7 }{ Marchen }
620 { "11D0 }{ Masaram~ Gondi }
621 { "1200 }{ Cuneiform }
622 { "1240 }{ Cuneiform~ Numbers~ and~ Punctuation }
623 { "1248 }{ Early~ Dynastic~ Cuneiform }
624 { "1300 }{ Egyptian~ Hieroglyphs }
625 { "1440 }{ Anatolian~ Hieroglyphs }
626 { "1680 }{ Bamum~ Supplement }
627 { "16A4 }{ Mro }
628 { "16AD }{ Bassa~ Vah }
629 { "16B0 }{ Pahawh~ Hmong }
630 { "16F0 }{ Miao }
631 { "16FE }{ Ideographic~ Symbols~ and~ Punctuation }
632 { "1700 }{ Tangut }
633 { "1880 }{ Tangut~ Components }
634 { "1B00 }{ Kana~ Supplement }
635 { "1B10 }{ Kana~ Extended-A }
636 { "1B17 }{ Nushu }
637 { "1BC0 }{ Duployan }
638 { "1BCA }{ Shorthand~ Format~ Controls }
639 { "1D00 }{ Byzantine~ Musical~ Symbols }
640 { "1D10 }{ Musical~ Symbols }
641 { "1D20 }{ Ancient~ Greek~ Musical~ Notation }
642 { "1D30 }{ Tai~ Xuan~ Jing~ Symbols }
643 { "1D36 }{ Counting~ Rod~ Numerals }
644 { "1D40 }{ Mathematical~ Alphanumeric~ Symbols }
645 { "1D80 }{ Sutton~ SignWriting }
646 { "1E00 }{ Glagolitic~ Supplement }
647 { "1E80 }{ Mende~ Kikakui }
648 { "1E90 }{ Adlam }
649 { "1EE0 }{ Arabic~ Mathematical~ Alphabetic~ Symbols }
650 { "1F00 }{ Mahjong~ Tiles }
651 { "1F03 }{ Domino~ Tiles }
652 { "1FOA }{ Playing~ Cards }

```

```

653 { "1F10 }{ Enclosed~ Alphanumeric~ Supplement }
654 { "1F20 }{ Enclosed~ Ideographic~ Supplement }
655 { "1F30 }{ Miscellaneous~ Symbols~ and~ Pictographs }
656 { "1F60 }{ Emoticons }
657 { "1F65 }{ Ornamental~ Dingbats }
658 { "1F68 }{ Transport~ and~ Map~ Symbols }
659 { "1F70 }{ Alchemical~ Symbols }
660 { "1F78 }{ Geometric~ Shapes~ Extended }
661 { "1F80 }{ Supplemental~ Arrows~C }
662 { "1F90 }{ Supplemental~ Symbols~ and~ Pictographs }
663 { "2000 }{ CJK~ Unified~ Ideographs~ Extension~ B }
664 { "2A70 }{ CJK~ Unified~ Ideographs~ Extension~ C }
665 { "2B74 }{ CJK~ Unified~ Ideographs~ Extension~ D }
666 { "2B82 }{ CJK~ Unified~ Ideographs~ Extension~ E }
667 { "2CEB }{ CJK~ Unified~ Ideographs~ Extension~ F }
668 { "2F80 }{ CJK~ Compatibility~ Ideographs~ Supplement }
669 { "E010 }{ Tags }
670 { "E000 }{ Variation~ Selectors~ Supplement }
671 { "F000 }{ Supplementary~ Private~ Use~ Area~A }
672 % higher up not covered!
673 }

```

If none of the above has matched we are somewhere within a block so we want keep the current name. However, since the case statement was executed within a `\tl_gset:Nx` we have to do this by passing the current block name back.

```

674 { \tl_use:N \g_fmft_block_title_tl }
675 }
676 }

```

*(End definition for `\_fmft_update_block_title:n`.)*

**Key setup (display blocks)** The Unicode blocks may get indicated in different ways: with titles, only through rules, or not at all. Here is the necessary setup.

```

677 \bool_new:N \l_fmft_display_block_bool
678 \tl_new:N \l_fmft_display_block_action_tl
679 \keys_define:nn {__fmft}
680 {
681   ,display-block .choice:
682   ,display-block / titles .code:n =
683     \bool_set_true:N \l_fmft_display_block_bool
684     \tl_set:Nn \l_fmft_display_block_action_tl {\}
685   ,display-block / rules .code:n =
686     \bool_set_false:N \l_fmft_display_block_bool
687     \tl_set:Nn \l_fmft_display_block_action_tl {\ \midrule}
688   ,display-block / none .code:n =
689     \bool_set_false:N \l_fmft_display_block_bool
690     \tl_set:Nn \l_fmft_display_block_action_tl {\}
691   ,display-block .initial:n = titles
692 }

```

That's all of the programming using the L3 layer.

```
693 \ExplSyntaxOff
```

What remains is to require all packages needed ...

```
694 \RequirePackage{longtable,booktabs,caption,fontspec}
```

...and executing all options passed to the package via `\usepackage`.

```
695 \ProcessKeysPackageOptions{__fmft}
696 \</package>

```

**6 The standalone unicodefont.tex file**

```

697 {*standalone}
698 \documentclass{article}
699 \setlength\textwidth{470pt}
700 \setlength\oddsidemargin{0pt}
701 \addtolength\textheight{7\baselineskip}
702 \addtolength\topmargin{-3\baselineskip}
703 \usepackage{unicodefonttable}
704 \def\DEFAULTfontname{Latin Modern Roman}
705 \def\DEFAULTfontfeatures{}
706 \def\DEFAULTtableconfig{}
707 \def\DEFAULTunicodefont{}
708 \begin{document}
709 \typeout{^^J}
710 \ifx\generatetable\undefined
711 \else
712 \typein[\answer]{^^JReuse settings from last time (default yes)?^^J^^J%
713 [ font name = \DEFAULTfontname^^J
714 \space unicode? = \ifx\DEFAULTunicodefont\empty yes^^J
715 \space font features = \DEFAULTfontfeatures
716 \else no\fi^^J
717 \space table config = \DEFAULTtableconfig \space]}
718 \fi
719 \ifx\answer\empty
720 \let\FontNameToTable\DEFAULTfontname
721 \let\IsUnicodeFont\DEFAULTunicodefont
722 \let\FontFeaturesToApply\DEFAULTfontfeatures
723 \let\TableConfigurationToApply\DEFAULTtableconfig
724 \else
725 \typein[\FontNameToTable]%
726 {^^JInput external font name as understood by fontspec, e.g.,^^J%
727 'TeX Gyre Pagella' or 'lmroman10-regular.otf'%
728 \ifx\DEFAULTfontname\empty\else
729 ^^J^^J[default \DEFAULTfontname]\fi:]
730 \ifx\FontNameToTable\empty \let\FontNameToTable\DEFAULTfontname \fi
731 \typein[\IsUnicodeFont]%
732 {^^JIs this a Unicode font?^^J^^J%
733 \ifx\DEFAULTunicodefont\empty [default yes]\else [default no]\fi:]
734 \ifxIsUnicodeFont\empty
735 % \ifx\DEFAULTunicodefont\empty
736 % \else
737 \letIsUnicodeFont\DEFAULTunicodefont
738 % \fi
739 \else
740 \ifx\DEFAULTunicodefont\empty
741 \else
742 \letIsUnicodeFont\empty
743 \fi
744 \fi
745 \ifxIsUnicodeFont\empty
746 \typein[\FontFeaturesToApply]%
747 {^^JInput font feature key/value list to apply%
748 \ifx\DEFAULTfontfeatures\empty\else
749 ^^J^^J[default \DEFAULTfontfeatures]\fi:]
750 \ifx\FontFeaturesToApply\empty \let\FontFeaturesToApply\DEFAULTfontfeatures \fi
751 \else

```

```

752 \let\FontFeaturesToApply\DEFAULTfontfeatures
753 \fi
754 \typein[\TableConfigurationToApply]%
755   {\^^JInput table configuration key/value list to apply%
756   \ifx\DEFAULTtableconfig\empty\else
757   ^^J^^J[default
758   \expandafter\detokenize\expandafter{\DEFAULTtableconfig}]\fi:}
759 \ifx\TableConfigurationToApply\empty
760 \let\TableConfigurationToApply\DEFAULTtableconfig
761 \fi
762 \edef\generatetable{\noexpand\displayfonttable
763 \ifx\IsUnicodeFont\empty\else *\fi
764 \ifx\TableConfigurationToApply\empty\else
765   [\expandafter\unexpanded\expandafter{\TableConfigurationToApply}]\fi
766   {\FontNameToTable}%
767 \ifx\FontFeaturesToApply\empty\else[\FontFeaturesToApply]\fi
768 }
769 \fi
770 \makeatletter
771 \protected@write\@auxout{}\{\gdef\string\generatetable
772   {\expandafter\detokenize\expandafter{\generatetable}}\}
773 \protected@write\@auxout{}\{\gdef\string\DEFAULTfontname{\FontNameToTable}\}
774 \protected@write\@auxout{}\{\gdef\string\DEFAULTfontcodefont{\IsUnicodeFont}\}
775 \protected@write\@auxout{}\{\gdef\string\DEFAULTfontfeatures{\FontFeaturesToApply}\}
776 \protected@write\@auxout{}\{\gdef\string\DEFAULTtableconfig
777   {\expandafter\detokenize\expandafter{\TableConfigurationToApply}}\}
778 \makeatother
779 \generatetable
780 \end{document}
781 </standalone>

```

## 7 A samples file

```

782 <*samples>
783 %<<VERBATIMLINE
784 %!TEX program = lualatex
785
786 %VERBATIMLINE
787 \documentclass{article}
788
789 \usepackage{xparse,color}
790
791 \usepackage{fontspec}
792
793 \setmainfont{Linux Biolinum 0}
794 \setmonofont{SourceCodePro}
795
796 \usepackage{unicodefonttable}
797
798 \addtolength\textwidth{30pt}
799
800 \begin{document}
801
802 \listoftables
803
804 \section{Computer Modern --- 8bit font}
805
806 \displayfonttable*[color=none,

```



```

807   range-end = 7F,
808 ]{cmr10}
809
810 %\section{Computer Modern Sans --- 8bit font} \displayfonttable*[] {cmss10}
811
812 \newpage
813
814 \section{TeX Gyre Heros (Helvetica) --- 8bit font}
815
816 \displayfonttable*[color=red,nostatistics=false,
817   hex-digits = head+foot,
818   range-end = FF,
819 ]{ec-qhvr}
820
821 \newpage
822
823 \section{Latin Modern Sans --- OTF font}
824
825 \displayfonttable[
826 % display-block = rules,
827 %   missing-glyph = \tiny\setlength\fbboxsep{0pt}\fbbox{$\times$},
828   hex-digits = block,
829   title-format-cont = \caption[] {\emph{continued}},
830 ]{Latin Modern Sans}
831
832 \newpage
833
834 \section{\TeX{} Gyre Pagella (Palatino) oldstyle figures --- OTF font}
835
836 \displayfonttable{TeX Gyre Pagella}[Numbers=OldStyle]
837
838 \newpage
839
840 \section{Comparing Latin Modern Math with New Computer Modern Math}
841
842 \displayfonttable[compare-with=NewCMMath-Regular.otf, range-end=1FFFF]
843   {latinmodern-math.otf}
844
845 \end{document}
846 </samples>

```

## Index

Numbers written in *italic* refer to the page where the corresponding entry is described or mentioned. Numbers underlined refer to the code line of the definition; numbers in Roman refer to the code lines where the entry is used.

Symbols	B
$\backslash$ , ..... 135	<code>\baselineskip</code> ..... 701, 702
$\backslash$ , ..... <u>904</u>	<code>\begin</code> ..... 42, 708, 800
$\backslash-$ ..... <u>906</u>	<code>\bfseries</code> ..... 372
$\backslash\backslash$ ..... 69, 71, 74, 76, 79, 142, 144, 146, 288, 355, 356, 365, 368, 375, 377, 379, 684, 687, 690	bool commands: <code>\bool_gset_false:N</code> ..... 364, 383, 391, 392, 404, 405 <code>\bool_gset_true:N</code> 209, 260, 396, 403 <code>\bool_if:NTF</code> ..... 42, 47, 68, 70, 73, 75, 78, 143, 145, 350, 354, 360, 362, 374, 382
A	
<code>\addtolength</code> ..... 701, 702, 798	
<code>\answer</code> ..... 712, 719	







