

Zpravodaj Československého sdružení uživatelů TeXu

Hans Hagen
Emoji Again

Zpravodaj Československého sdružení uživatelů TeXu, Vol. 28 (2018), No. 1-4, 43–58

Persistent URL: <http://dml.cz/dmlcz/150106>

Terms of use:

© Československé sdružení uživatelů TeXu, 2018

Institute of Mathematics of the Czech Academy of Sciences provides access to digitized documents strictly for personal use. Each copy of any part of this document must contain these *Terms of use*.



This document has been digitized, optimized for electronic delivery and stamped with digital signature within the project *DML-CZ: The Czech Digital Mathematics Library* <http://dml.cz>

Since the 10th International CONTEX_T Meeting in 2016, CONTEX_T has supported the OpenType `colr` and `cpal` tables that are used in color fonts and also to produce emoji. The article introduces emoji and uses the MICROSOFT's `seguiemj` font to show how emoji are constructed from glyphs, how emoji can be stacked into sequences, and how the palettes of a color font can be changed in CONTEX_T.

Keywords: LUA, L_UA_TE_X, CONTEX_T, O_PE_NT_YP_E, emoji

Because at the CONTEX_T 2016 meeting color fonts¹ were on the agenda, some time was spent on emoji (these colorful small picture glyphs). When possible I bring kids to the Bacho_TE_X conference so for the 2017 Bacho_TU_G I decided to do something with emoji that, after all, are mostly used by those younger than I am. So, I had to take a look at the current state. Here are some observations.

The UNIC_OD_E standard defines a whole lot of emoji and if mankind manages to survive for a while one can assume that a lot more will be added. After all, icons as well as variants keep evolving. There are several ways to organize these symbols in groups but I will not give grouping a try. Just visit emojipedia.org and you get served well. For this story I only mention that:

- There are quite some shapes and nearly all of them are in color. The yellow ones, smilies and such, are quite prominently present but there are many more.
- A special subset is filled by persons: man, woman, girl, boy and recently a baby.
- The grown ups can be combined in loving couples (either or not kissing) and then can form families, but only upto 2 young kids or gender neutral babies.
- All persons can be flagged with one of five skin tones so that not all persons (or heads) look bright yellow.
- Interesting is that girls and boys are still fond of magenta (pinkish) and cyan (blueish) cloths and ornaments. Also haircuts are rather specific to the gender.

For rendering color emojis we have a few color related O_PE_NT_YP_E font properties available: bitmaps, SVG, and stacked glyphs. Now, if you think of the

¹For that occasion the cowfont, a practical joke concerning Dutch 'koeletters', were turned into a color font and presented at the meeting.

combinations that can be made with skin tones, you realize that fonts can become pretty large if each combination results in a glyph. In the first half of 2017 MICROSOFT released an update for its emoji font and the company took the challenge to provide not only mixed skin tone couples, but also supported skin tones for the kids, including a baby.

This recent addition already adds over 25,000 additional glyphs¹ so imagine what will happen in the future. But, instead of making a picture for each variant, a different solution has been chosen. For coloring this `seguiemj` font uses the (very flexible) stacking technology: a color shape is an overlay of colored symbols. The colors are organized in palettes and it's no big deal to add additional palettes if needed. Instead of adding pre-composed shapes (as is needed with bitmaps and SVG) snippets are used to build alternative glyphs and these can be combined into new shapes by substitution and positioning (for that kerns, mark anchoring and distance compensation is used).

So, a family can be constructed of composed shapes (man, woman, etc) that each are composed of snippets (skull, hair, mouth, eyes). So, effectively a family of four is a bunch of maybe 25 small glyphs overlaid and colored. In Figure 1 we see how a shape is constructed out of separate glyphs. Figure 2 shows how they can be overlaid with colors (we use a dedicated color set).

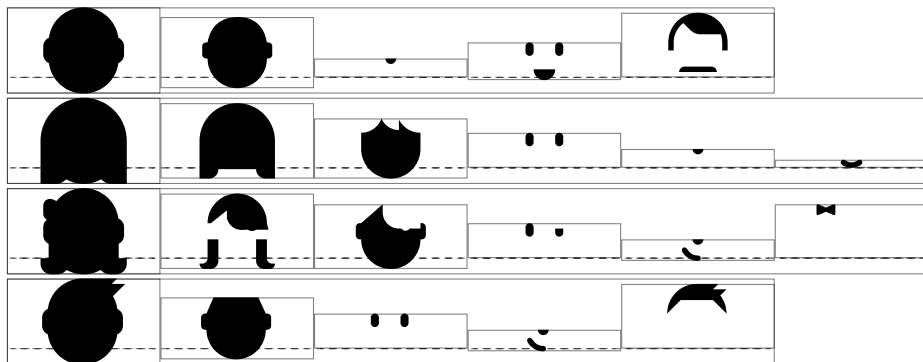


Figure 1 Emoji snippets.

When a font supports it, a sequence of emoji can be turned into a more compact representation. In Figure 3 we see how skin tones are applied in such combinations. Figure 4 shows the small snippets.

¹That is the amount I counted when I added all combinations runtime but the emoji-pedia mentions twice that amount. Currently in `CONTEXT` we resolve such combinations when requested.

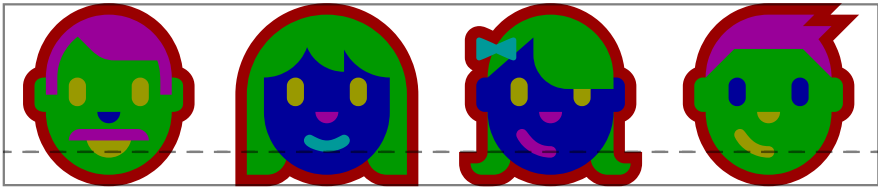


Figure 2 Emoji snippets overlaid.

When we have to choose a font we need to take the following criteria into account:

- What is the quality of the shapes? For sure, outlines are best if you want to scale too.
- How efficiently is a shape constructed? In that respect a bitmap or SVG image is just one entity.
- How well can a (semi)arbitrary combinations of emoji be provided? Here the glyph approach wins.
- Are all skin colors for all human relates shapes supported? Actually it opens the possibility for racist fonts.
- Are all reasonable combinations of persons supported? It looks like (depending on time and version) kissing men or women can be missing, maybe because of social political reasons.
- Are black and white shapes provided alongside color shapes?

Maybe an SVG or bitmap image can have a lot of detail compared to a stacked glyph but, when we're just using pictographic representations, the latter is the best choice.

When I was playing a bit with the skin tone variants and other combinations that should result in some composed shape, I used the UNICODE test files but I got the impression that there were some errors in the test suite, for instance with respect to modifiers. Maybe the fonts are just doing the wrong thing or maybe some implement these sequences a bit inconsistent. This will probably improve over time but the question is if we should intercept issues. I'm not in favour of this because it adds more and more fuzzy code that not only wastes cycles (energy) but is also a conceptual horror. So, when testing, imperfection has to be accepted for now. This is no big deal as until now no one ever asked for emoji support in `CONTEXT`.

When no combined shape is provided, the original sequence shows up. A side effect can be that zero-width-joiners and modifiers become visible. This

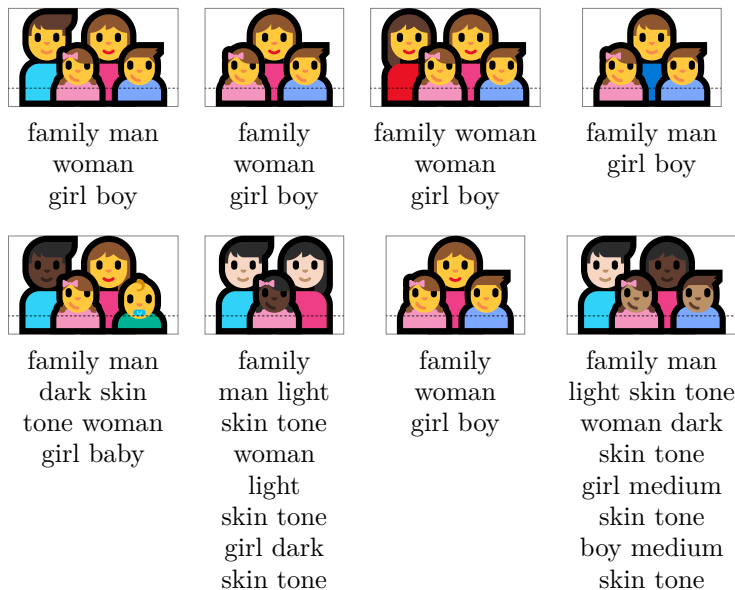


Figure 3 Emoji families and such with skin tones.

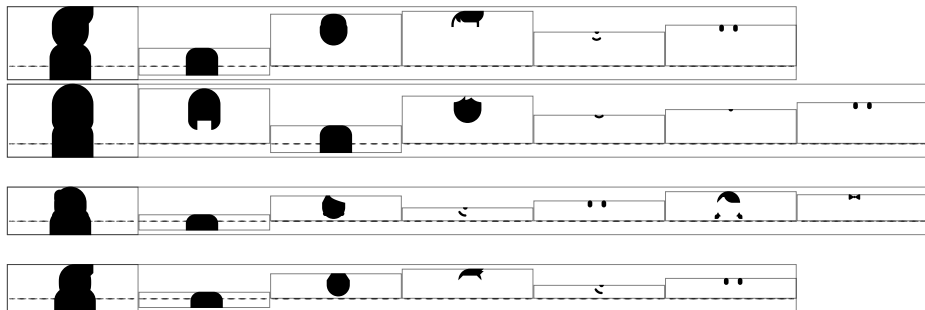


Figure 4 Emoji glyphs.










depends on the fonts. Users probably don't care that much about it. Now how do we suppose that users enter these emoji (sequences) in a document source? One can imagine a pop up in the editor but $\text{T}_{\text{E}}\text{X}$ ies are often using commands for special cases.

We already showed some combined shapes. The reader might appreciate the outcome but getting there from the input takes a bit of work. For instance a two person man light skin tone woman medium skin tone girl medium-light skin tone baby medium-light skin tone involves this:

font 11: seguiemj.ttf @ 10.0pt

features [basic: ccmp=yes, dist=yes, mark=yes, mkmk=yes, script=dflt, tlig=yes, trep=yes] [extra: analyze=yes, autolanguage=position, autoscript=position, checkmarks=yes, colr=yes, curs=yes, devanagari=yes, dummies=yes, extensions=yes, extrafeatures=yes, extraprivates=yes, kern=yes, liga=yes, mathkerns=yes, mathrules=yes, mode=node, spacekern=yes]

step 1

 [+TLT] U+1F468:  U+1F3FB: 
U+200D: ↑ U+1F469:  U+1F3FD:  U+200D: ↑ U+1F467: 
U+1F3FC:  U+200D: ↑ U+1F476:  U+1F3FC: 






feature 'ccmp', type 'gsub_ligature', lookup 's_s_0',
replacing U+1F468 upto U+1F3FB by ligature U+F01C5
case 2

feature 'ccmp', type 'gsub_ligature', lookup 's_s_0',
replacing U+1F469 upto U+1F3FD by ligature U+F01D2
case 2

feature 'ccmp', type 'gsub_ligature', lookup 's_s_0',
replacing U+1F467 upto U+1F3FC by ligature U+F01BC
case 2






feature 'ccmp', type 'gsub_ligature', lookup 's_s_0',
replacing U+1F476 upto U+1F3FC by ligature U+F021A
case 2

step 2

 [+TLT] U+F01C5:  U+200D: ↑ U+F01D2: 
U+200D: ↑ U+F01BC:  U+200D: ↑ U+F021A: 






feature 'ccmp', type 'gsub_contextchain', chain lookup
's_s_2', replacing single U+F01C5 by U+F15C4

step 3

 [+TLT] U+F15C4:  U+200D: ↑ U+F01D2: 
U+200D: ↑ U+F01BC:  U+200D: ↑ U+F021A: 

feature 'ccmp', type 'gsub_contextchain', chain lookup
's_s_3', index 1, replacing character U+200D upto
U+F01D2 by ligature U+F15EC case 4

step 4

 [+TLT] U+F15C4:  U+F15EC:  U+200D: ↑
U+F01BC:  U+200D: ↑ U+F021A: 

feature 'ccmp', type 'gsub_contextchain', chain lookup
's_s_5', index 1, replacing character U+200D upto
U+F01BC by ligature U+F15B9 case 4

step 5  [+TLT] U+F15C4:  U+F15EC:  U+F15B9:  U+200D:   U+F021A: 

feature 'ccmp', type 'gsub_contextchain', chain lookup
's_s_6', index 1, replacing character U+200D upto
U+F021A by ligature U+F1607 case 4

step 6  [+TLT] U+F15C4:  U+F15EC:  U+F15B9:  U+F1607: 

feature 'ccmp', type 'gsub_contextchain', chain lookup
's_s_7', replacing single U+F15B9 by U+F15AC

step 7  [+TLT] U+F15C4:  U+F15EC:  U+F15AC:  U+F1607: 






feature 'dist', type 'gpos_single', lookup 'p_s_0',
shifting single U+F15C4 by single xy (1.25pt,0pt)
and wh (0pt,0pt)

step 8  [+TLT] U+F15C4:  U+F15EC:  U+F15AC:  U+F1607: 

feature 'dist', type 'gpos_single', lookup 'p_s_1',
shifting single U+F15EC by single xy (0pt,0pt) and
wh (1.25pt,0pt)

step 9  [+TLT] U+F15C4:  U+F15EC:  [kern] U+F15AC:  U+F1607: 

feature 'dist', type 'gpos_contextchain', chain lookup
'p_s_2', shifting single U+F15C4 by single (0pt,0pt)
and correction (1.25pt,0pt)

step 10  [+TLT] [kern] U+F15C4:  U+F15EC:  [kern] U+F15AC:  U+F1607: 

feature 'dist', type 'gpos_contextchain', chain lookup
'p_s_3', shifting single U+F15EC by single
(4.76074pt,0pt) and correction (0pt,0pt)

feature 'dist', type 'gpos_contextchain', chain lookup
'p_s_3', shifting single U+F15EC by single (0pt,0pt)
and correction (8.27148pt,0pt)

step 11  [+TLT] [kern] U+F15C4:  [kern] U+F15EC:  [kern]
 U+F15AC:  U+F1607: 

feature 'dist', type 'gpos_contextchain', chain lookup
 'p_s_5', shifting single U+F15C4 by single (Opt,Opt)
 and correction (-4.76074pt,0pt)

step 12  [+TLT] [kern] U+F15C4:  [kern] [kern] U+F15EC: 
 [kern] U+F15AC:  U+F1607: 

feature 'mark', type 'gpos_mark2base', lookup
 'p_s_27', bound 1, anchoring mark U+F15AC to
 basechar U+F15EC => (6.32812pt,0pt)

step 13  [+TLT] [kern] U+F15C4:  [kern] [kern] U+F15EC: 
 [kern] U+F15AC:  U+F1607: 




feature 'mark', type 'gpos_mark2base', lookup
 'p_s_28', bound 2, anchoring mark U+F1607 to
 basechar U+F15EC => (0.00977pt,0pt)

result  [+TLT] [kern] U+F15C4:  [kern] [kern] U+F15EC: 
 [kern] U+F15AC:  U+F1607: 




A black and white example is the following family woman girl:

font 17: seguiemj.ttf @ 10.0pt




features [basic: ccmp=yes, dist=yes, mark=yes, mknk=yes,
 script=dflt, tlig=yes, trep=yes] [extra: analyze=yes,
 autolanguage=position, autoscript=position,
 checkmarks=yes, curs=yes, devanagari=yes, dummies=yes,
 extensions=yes, extrafeatures=yes, extraprivates=yes,
 kern=yes, liga=yes, mathkerns=yes, mathrules=yes,
 mode=node, spacekern=yes]




step 1  [+TLT] U+1F469:  U+200D: † U+1F467: 





feature 'ccmp', type 'gsub_contextchain', chain lookup
 's_s_4', replacing single U+1F469 by U+F15E2




step 2  [+TLT] U+F15E2:  U+200D: † U+1F467: 

feature 'ccmp', type 'gsub_contextchain', chain lookup
 's_s_5', index 1, replacing character U+200D upto
 U+1F467 by ligature U+F15B0 case 4


































































step 3  [+TLT] U+F15E2: U+F15B0:
 feature 'dist', type 'gpos_single', lookup 'p_s_1',
 shifting single U+F15E2 by single xy (0pt,0pt) and
 wh (1.25pt,0pt)

step 4  [+TLT] U+F15E2: [kern] U+F15B0:
 feature 'dist', type 'gpos_contextchain', chain lookup
 'p_s_4', shifting single U+F15E2 by single
 (1.25pt,0pt) and correction (0pt,0pt)
 feature 'dist', type 'gpos_contextchain', chain lookup
 'p_s_4', shifting single U+F15E2 by single (0pt,0pt)
 and correction (4.76074pt,0pt)



































































step 5   [+TLT] [kern] U+F15E2: [kern] U+F15B0:
 feature 'mark', type 'gpos_mark2base', lookup
 'p_s_28', bound 1, anchoring mark U+F15B0 to
 basechar U+F15E2 => (0.00977pt,0pt)

















result  [+TLT] [kern] U+F15E2: [kern] U+F15B0:

I will not show all emoji, just the subset that contains the word woman in the description. As you can see the persons in the sequences are separated by a zero-width-joiner. There are some curious ones, for instance a **woman wearing turban** which in terms of UNICODE input is a female combine with a turban wearing man becomes a beardless woman wearing a turban. Woman vampires and zombies are not supported so these are male properties.

 	 	blondhaired woman
  	  	couple with heart woman man
  	 	couple with heart woman woman
  	  	family man woman boy
   	   	family man woman boy boy
  	  	family man woman girl
   	   	family man woman girl boy
   	   	family man woman girl girl
 	 	family woman boy
  	  	family woman boy boy
 	 	family woman girl

		family woman girl boy
		family woman girl girl
		family woman woman boy
		family woman woman boy boy
		family woman woman girl
		family woman woman girl boy
		family woman woman girl girl
		kiss woman man
		kiss woman woman
		man and woman holding hands
		old woman
		pregnant woman
		woman
		woman artist
		woman astronaut
		woman biking
		woman boot
		woman bouncing ball
		woman bowling
		woman cartwheeling
		woman climbing
		woman clothes
		woman construction worker
		woman cook
		woman dancing
		woman detective
		woman elf
		woman facepalming
		woman factory worker
		woman fairy
		woman farmer
		woman firefighter
		woman frowning

		woman genie
		woman gesturing no
		woman gesturing ok
		woman getting haircut
		woman getting massage
		woman golfing
		woman guard
		woman hat
		woman health worker
		woman in lotus position
		woman in steamy room
		woman judge
		woman juggling
		woman lifting weights
		woman mage
		woman mechanic
		woman mountain biking
		woman office worker
		woman pilot
		woman playing handball
		woman playing water polo
		woman police officer
		woman pouting
		woman raising hand
		woman rowing boat
		woman running
		woman sandal
		woman scientist
		woman shrugging
		woman singer
		woman student
		woman surfing
		woman swimming

		woman teacher
		woman technologist
		woman tipping hand
		woman vampire
		woman walking
		woman wearing turban
		woman with headscarf
		woman zombie

So what if you don't like these colors? Because we're dealing with T_EX you can assume that if there is some way around the fixed color sets, then it will be provided. So, when you use C_ON_TE_XT, here is a way to overload them:

```
\definecolor[emoji-red][r=.4]
\definecolor[emoji-green][g=.4]
\definecolor[emoji-blue][b=.4]
\definecolor[emoji-yellow][r=.4,g=.4]
\definecolor[emoji-gray][s=1,t=.5,a=1]
```

```
\definefontcolorpalette[emoji-s]
[black,emoji-gray]
\definefontcolorpalette[emoji-r]
[emoji-red,emoji-gray]
\definefontcolorpalette[emoji-g]
[emoji-green,emoji-gray]
\definefontcolorpalette[emoji-b]
[emoji-blue,emoji-gray]
\definefontcolorpalette[emoji-y]
[emoji-yellow,emoji-gray]
```

```
\definefontfeature[seguiemj-s]
[ccmp=yes,dist=yes,colr=emoji-s]
\definefontfeature[seguiemj-r]
[ccmp=yes,dist=yes,colr=emoji-r]
\definefontfeature[seguiemj-g]
[ccmp=yes,dist=yes,colr=emoji-g]
\definefontfeature[seguiemj-b]
[ccmp=yes,dist=yes,colr=emoji-b]
\definefontfeature[seguiemj-y]
[ccmp=yes,dist=yes,colr=emoji-y]
```

```
\definefont[MyEmojiS]
```

```

[seguiemj*seguiemj-s]
\definefont [MyEmojiR]
[seguiemj*seguiemj-r]
\definefont [MyEmojiG]
[seguiemj*seguiemj-g]
\definefont [MyEmojiB]
[seguiemj*seguiemj-b]
\definefont [MyEmojiY]
[seguiemj*seguiemj-y]

```



Figure 5 Overloading colors by plugging in a sequence of alternate colors.

In Figure 5 we see how this is applied. You can provide as many colors as needed but when you don't provide enough the last one is used. This way we get the overlaid transparent colors in the examples. By using transparency we don't obscure shapes.

The emoji-pedia mentions “Asked about the design, MICROSOFT told emoji-pedia that one of the reasons for the thick stroke was to allow each emoji to be easily read on any background color.” The first glyph in the stack seems to do the trick, so just make sure that it doesn't become white. And, before I read that remark, while preparing a presentation with a colored background, I had

already noticed that using a background was no problem. This font definitely sets the standard.

How do we know what colors are used? The next table shows the first color palette of `seguiemj`. There are quite some colors so defining your own definitely involved some studying.

1	1	2	3	4	5	6	7	8	9	10	11
	12	13	14	15	16	17	18	19	20	21	22
	23	24	25	26	27	28	29	30	31	32	33
	34	35	36	37	38	39	40	41	42	43	44
	45	46	47	48	49	50	51	52	53	54	55
	56	57	58	59	60	61	62	63	64	65	66
	67	68	69	70	71	72	73	74	75	76	77
	78	79	80	81	82	83	84	85	86	87	88
	89	90	91	92	93	94	95	96	97	98	99
	100	101	102	103	104	105	106	107	108	109	110
	111	112	113	114	115	116	117	118	119	120	121
	122	123	124	125	126	127	128	129	130	131	132
	133	134	135	136	137	138	139	140	141	142	143
	144	145	146	147	148	149	150	151	152	153	154
	155	156	157	158	159	160	161	162	163	164	165
	166	167	168	169	170	171	172	173	174	175	

Normally special symbols are accessed in `CONTEXT` with the `symbol` command where symbols are organized in symbol sets. This is a rather old mechanism and dates from the time that fonts were limited in coverage and symbols were collected in special fonts. The emoji are accessed by their own command: `\emoji`. The font used has the font synonym `emoji` so you need to set that one first:

```
\definefontsynonym
  [emoji]
  [seguiemj*seguiemj-cl]
```

Here is an example:

```
\emoji{woman light skin tone}\quad
\emoji{woman scientist}\quad
{\bfd bigger
 \emoji{man health worker}}
```

or typeset:   **bigger** 

The emoji symbol scales with the normal running font. When you ask for a family with skin toned members the lookup can result in another match (or no match) because one never knows to what extend a font supports it.

<code>\expandedemoji</code>	the sequence constructed from the given string
<code>\resolvedemoji</code>	a protected sequence constructed from the given string
<code>\checkedemoji</code>	a typeset sequence with unresolved modifiers and joiners removed
<code>\emoji</code>	a typeset resolved sequence using the <code>emoji</code> font synonym
<code>\robustemoji</code>	a typeset checked sequence using the <code>emoji</code> font synonym

In case you wonder how some of the details above were typeset, there is a module `fonts-emoji` that provides some helpers for introspection.

<code>\ShowEmoji</code>	show all the emoji in the current font
<code>\ShowEmojiSnippets</code>	show the snippets of a given emoji
<code>\ShowEmojiSnippetsOverlay</code>	show the overlaid snippets of a given emoji
<code>\ShowEmojiGlyphs</code>	show the snippets of a typeset emoji
<code>\ShowEmojiPalettes</code>	show the color palettes in the current font

Examples of usage are:

```
\ShowEmojiSnippets
  [family man woman girl boy]
\ShowEmojiGlyphs
  [family man woman baby girl]
\ShowEmoji      [^man]
\ShowEmoji
\ShowEmojiPalettes
\ShowEmojiPalettes[1]
```

A good source of information about emoji is the mentioned emojipedia.org website. There you find not only details about all these symbols but also has some history. It compares updates in fonts too. It mentions for instance that in the creative update of Windows 10, some persons grew beards in the `seguiemj` font and others lost an eye. Now, if you look at the snippets shown before, you can wonder if that eye is really gone. Maybe the color is wrong or the order of stacking is not right. I decided not to waste time looking into that.

Another quote: “Support for color emoji presentation on MS WINDOWS is limited. Many applications on MS WINDOWS display emojis with a black and white text presentation instead of their color version.” Well, we can do better with `TEX`, but as usual not that many people really cares about that. But it’s fun anyway.

We end with a warning. When you use ‘ligatures’ like this, you really need to check the outcome. For instance, when MICROSOFT updated the font in 2017, same gender couples got different hair style for the individuals so that one can still distinguish them. However, kissing couples and couples in love (indicated by a heart) seem to be removed. Who knows how and when politics creep into fonts:

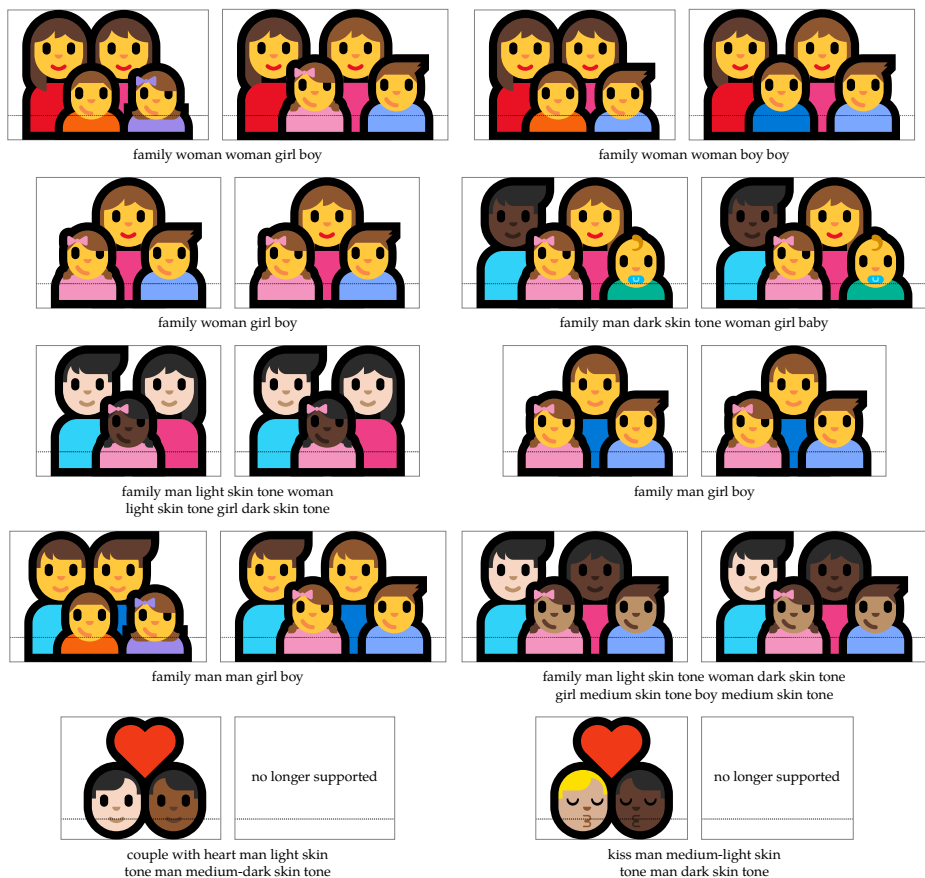


Figure 6 Incompatible updates.

is public mixed couple kissing permitted, do we support families with any mix of gender, is associating pink with girls okay or not, how do we distinguish male and female anyway? In Figure 6 we see the same combination twice, the early 2017 rendering versus the late 2017 rendering. Can you notice the differences?

Zase emoji

Od desátého Mezinárodního setkání uživatelů CONTEXTu v roce 2016 podporuje CONTEXT opentypové tabulky `colr` a `cpal`, které se používají v barevných písmech a také pro přípravu emoji. Článek představuje emoji a na příkladu písma

seguiemj od firmy MICROSOFT ukazuje, jak jsou znaky emoji sestaveny, jak je lze spojovat do posloupností a jak lze v CONTEXtu upravit barevnou paletu písma.

Klíčová slova: LUA, L^AT_EX, CONTEX_T, OPENTYPE, emoji

Hans Hagen, pragma@wxs.nl