




Colorful emojis via Unicode and OpenType

Hans Hagen

A recent new (and evolving) addition to OpenType is colored glyphs. One variant (by Microsoft) uses overlays and this method is quite efficient.

```
\definefontfeature[colored][colr=yes]
\definefontsynonym[Emoji]
[file:seguiemj.ttf*default,colored]

\definesymbol[bug][\getglyphdirect{Emoji}
{\char"1F41B}]
\definesymbol[ant][\getglyphdirect{Emoji}
{\char"1F41C}]
\definesymbol[bee][\getglyphdirect{Emoji}
{\char"1F41D}]
```

Here we see a , , and , and they come in color! Since Unicode has started adding such symbols (and more in each release) the distinction between characters and symbols becomes even fuzzier. Of course one can argue that we communicate in pictograms but even then, given that mankind may last a while yet, the Unicode repertoire will explode.






U+1F41B bug U+1F41C ant U+1F41D bee

Figure 1: A few emojis from `seguiemj.ttf`.

Above we have used `seguiemj.ttf`, a font that comes with Windows. Colors are achieved by combining glyphs rendered in different colors. A variant font that uses SVG instead of overlays is `emojionecolor-svginot.ttf`:


























```
\definefontfeature[svg][svg=yes]
\definefontsynonym[Emoji]
[file:emojionecolor-svginot.ttf*default,svg]
```










This time we get , , and , and they look quite different. Both fonts also have ligatures and you can wonder what sense that makes. It makes it impossible to swap fonts and as there is no standard one never knows what to expect.





U+1F41B bug U+1F41C ant U+1F41D bee

Figure 2: The same emojis from `emojionecolor-svginot.ttf`.

How do we know what faces add up to the ligature  and how are we supposed to know that there should be `zwj` between? When we input four faces separated by zero width joiners, we get a four face symbol instead. The reason for having the joiners is probably to avoid unexpected ligatures. The sequence man, woman, boy, boy gives family:  + `zwj`  + `zwj`  + `zwj`  = , but two girls also works:  + `zwj`  + `zwj`  + `zwj`  = , and so does a mixture of kids:  + `zwj`  + `zwj`  + `zwj`  = , although (at least currently):  + `zwj`  + `zwj`  + `zwj`  =     (not stacked). To add to the random fun, the official Unicode family U+1F46A has only three members (in this font): .

In our times for sure many combinations are possible, so:  + `zwj`  + `zwj`  + `zwj`  =  indeed gives a family, but I wonder at what point cultural bias will creep into font design. One can even wonder how clothing and hair styles will demand frequent font updates:    .

In the math alphabets we have a couple of annoying holes because characters were already present in Unicode, so now we forever have to deal with those exceptions. But not so with emojis because here eventually all variants will show up. Although a character A in red or blue uses the same code point, a white telephone (not in this particular font) and black telephone  have their own. And because obsolete scripts are already supported in Unicode and more get added, we can expect old artifacts also showing up at some time. Soon the joystick  will be an unknown item to most, while the Microsoft hololens might get its slot.



U+1F423 hatching chick U+1F424 baby chick U+1F425 front-facing baby chick

Figure 3: Will all animals come in all stages of development?

For sure these mechanisms will evolve and to what extent we support them depends on what users want. At least we have the basics implemented.

◇ Hans Hagen
Pragma ADE
<http://pragma-ade.com>