

Simple Fonts with MetaFun

Hans Hagen

Metafonts

Because MetaPost is based on MetaFont, it makes sense to use it for making fonts.

Making a font is an art in itself; something that is actually proven by the many bad looking fonts available, although we have plenty of alternative choices these days. We tend to use free fonts often being made by volunteers, so we can hardly make any demands.

So, instead of complaining (which is not nice anyway) we can try to (at least temporary) come up with a solution ourselves. We're actually talking about missing glyphs here, and MetaPost can be of help.

Also keep in mind that we always had this option or variants of it in ConT_EXt; it's just that we can make the interfaces nicer now. Just don't expect anything spectacular.

What it is not

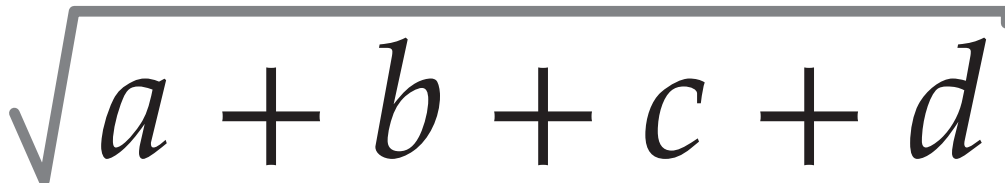
Years ago, mechanisms were added to MkIV to come up with more fancy shapes in, for instance, math. Actually Alan needed it and I wanted a root symbol to look like school times.

```
\useMPLibrary[mat]
\setupmathradical[color=darkgray,alternative=mp]
% \definemathradical [sqrt] [mp=minifun::math:radical:default]
```

So:

```
\scale[height=2cm]{$ \sqrt {a+b+c+d} $}
```

Gives:


$$\sqrt{a + b + c + d}$$

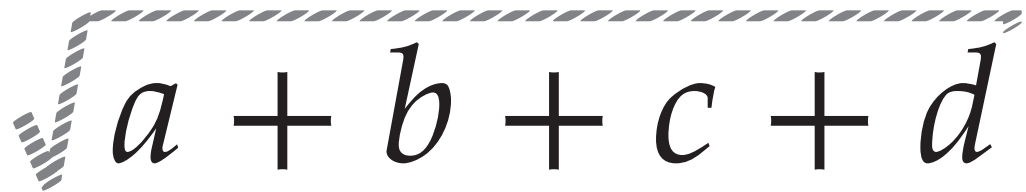
And with:

```

\startuniqueMPgraphic{minifun::math:radical:default}
draw
  math_radical_simple(
    OverlayWidth,OverlayHeight,OverlayDepth,OverlayOffset)
  withpen pencircle
    xscaled (20overlayLineWidth)
    yscaled (10overlayLineWidth/4)
    rotated 30
  dashed evenly
  withcolor OverlayLineColor ;
\stopuniqueMPgraphic

```

We get

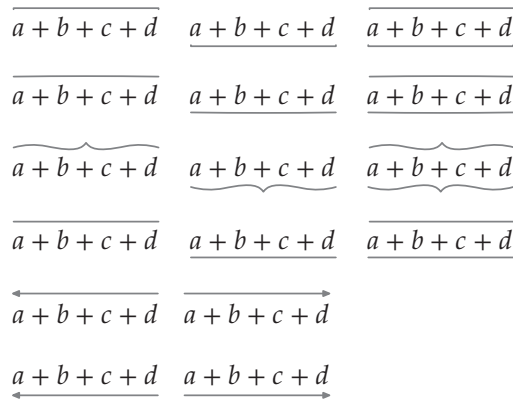


Also think of stackers:

```

\setupmathstackers [both] [color=darkgray,alternative=mp]
\setupmathstackers [top] [color=darkgray,alternative=mp]
\setupmathstackers [bottom] [color=darkgray,alternative=mp]

```



But, these are just overlays, and nothing special. We simply don't use the normal font route, or fancy Lua tricks either. In principle, MkII could do this. I might upgrade it some day even though there is no real demand for it so far (just for fun).

Real fonts

For text we need an efficient way to define extra shapes. We don't really want inline graphics every time we use a glyph. We also want to cut-and-paste properly. Basically, the fact that we drop in shapes should be hidden.

We use the same (generic) subsystem that is also used for color fonts, bitmap emojis, svg fonts, etc. The shapes end up as Type3 fonts. Although these have some specific properties and limitations, we can actually make Unicode fonts. The system is not burdened by much overhead, and most of the work happens at embedding time.

```
\definefont[DemoFontA][Serif*default @ 10pt]
\definefont[DemoFontB][Serif*default @ 12pt]
\definefont[DemoFontC][Serif*default @ 14pt]
\definefont[DemoFontD][SerifBold*default @ 14pt]
```

```
\startlines
\DemoFontA first\endash second\emdash third\char"2015\relax fourth
\DemoFontB first\endash second\emdash third\char"2015\relax fourth
\DemoFontC first\endash second\emdash third\char"2015\relax fourth
\DemoFontD first\endash second\emdash third\char"2015\relax fourth
\stoplines
```

first–second—thirdfourth
first–second—thirdfourth
first–second—thirdfourth
first–second—thirdfourth

```
\definefontfeature[exampleone][metapost=symbolsone]

\definefont[DemoFontA][Serif*default,exampleone @ 10pt]
\definefont[DemoFontB][Serif*default,exampleone @ 12pt]
\definefont[DemoFontC][Serif*default,exampleone @ 14pt]
\definefont[DemoFontD][SerifBold*default,exampleone @ 14pt]
```

first–second—third—fourth
first–second—third—fourth
first–second—third—fourth
first–second—third—fourth

```
\startMPcalculation{simplefun}

vardef QuotationDashOne =
  draw image (
    interim linecap := squared ;
    save l ; l := 0.2 ;
    draw (1/2,3) -- (10-1/2,3) withpen pencircle scaled l ;
  )
enddef ;
```

```

lmt_registerglyphs [
  name      = "symbolstone",
  units     = 10,
  usecolor  = true,
  width     = 10,
  height    = 3.1,
  depth     = 0,
] ;

lmt_registerglyph [
  category = "symbolstone",
  unicode  = "0x2015",
  code     = "QuotationDashOne ;"
] ;

\stopMPcalculation

```

```

\definefontfeature[exampletwo][metapost=symbolstwo]

\definefont[DemoFontA][Serif*default,exampletwo @ 10pt]
\definefont[DemoFontB][Serif*default,exampletwo @ 12pt]
\definefont[DemoFontC][Serif*default,exampletwo @ 14pt]
\definefont[DemoFontD][SerifBold*default,exampletwo @ 14pt]

```

first–second–third—fourth
 first–second–third—fourth
 first–second–third—fourth
first–second–third—fourth

```

\startMPcalculation{simplefun}

vardef QuotationDashTwo =
  draw image (
    interim linecap := squared ;
    save l ; l := 0.4 ;
    string weight ; weight := getparameter "mpsfnt" "parentdata" "shared" "rawdata"
"metadata" "weight" ;
    if weight = "semibold" : l := l * 2;
    elseif weight = "bold" : l := l * 3; fi
    draw (1/2,3) -- (10-1/2,3) withpen pencircle scaled l
    withcolor yellow ;
  )
enddef ;

lmt_registerglyphs [
  name      = "symbolstwo",
  units     = 10,
  usecolor  = false,
  width     = 10,
  height    = 3.1,
  depth     = 0,
] ;

% ...

```

contextgroup > context meeting 2020

```
lmt_registerglyph [  
  category = "symbolstwo",  
  unicode = "0x2015",  
  code = "QuotationDashTwo ;"  
] ;
```

`\stopMPcalculation`

More examples

We give some examples (these are also in the modules). Overloading math symbols:

```
meta-imp-kindergarten.mkx1
```

Extending fonts with Don Knuth's dice and tiles (symbols, ligatures, proper Unicode):

```
meta-imp-gamesymbols.mkx1
```

An implementation of Don Knuth's ThirtySix font in various variants (color, random, shapes):

```
meta-imp-threesix.mkx1
```



Photos: Harald König