Math in ConTEXt

1 This talk

This is just a quick overview (Ulrik and Jacko have more detailed talks) of how math is dealt with in ConTEXt MkIV.

2 Some characteristics

I Math is about characters, symbols, accents and visual constructs like radicals that have special meaning.

II We have inline and display math and three font sizes to play with: text, script, scriptscript.

$$a^{b^c} = \frac{d}{e}$$

III Ideally we could enter Unicode but lack of fonts for editing leads to inputting variables (a-z) in ascii.

IV Math is a family business although it stops after a few generations.

3 The way it goes in MKII

You key in some commands:

```
a + \bf b + \bi c = \tt d + \ss e + \cal f
```

In traditional TEX this becomes:

```
a + \fam7 b + \fam8 c = \fam9 d + \fam10 e + \fam11 f
```

This gets typeset as:

$$a + \mathbf{b} + \boldsymbol{c} = \mathtt{d} + \mathsf{e} + \mathcal{f}$$

And represents:

$\mathtt{a}_{\mathtt{/F1}}\ \mathtt{+}_{\mathtt{/F2}}\ \mathtt{b}_{\mathtt{/F3}}\ \mathtt{+}_{\mathtt{/F2}}\ \mathtt{c}_{\mathtt{/F4}}\ \mathtt{=}_{\mathtt{/F5}}\ \mathtt{d}_{\mathtt{/F6}}\ \mathtt{+}_{\mathtt{/F2}}\ \mathtt{e}_{\mathtt{/F7}}\ \mathtt{+}_{\mathtt{/F2}}\ \mathtt{f}_{\mathtt{/F8}}$

So, something happened in between.

4 The status quo

I Code that showed up first dominates potentially better solutions.

II Limitations in fonts (and 7 bit technology) made hacks into standards.

III Small fonts (the 256 boundary) asked for more families than available.

IV Font models are rather resource demanding.

V The plain TEX format steered implementations.

VI The rendering model has proven to be quite adequate in most cases.

VII The rise of Unicode changes the landscape.

Moving on

I We no longer support 8 bit math and use Unicode exclusively.

II We stick to one family per main style so in practice we only have regular and bold.

III Therefore we have (in most cases) ony one math font loaded.

IV Awaiting outcomes of the Gyre Math project we create virtual fonts runtime.

V It is still unclear what Gyre wil provide but we can use the current mechanisms for whatever comes out of it.

VI (This is also a consequence of the fact that MkIV only targets at LuaTEX.)

6 The way it goes in MKIV

You key in some commands:

```
a + \bf b + \bi c = \tt d + \ss e + \cal f
```

In ConTeXt this becomes:

$\texttt{a + b}_{\texttt{bf}}\texttt{ + c}_{\texttt{bi}}\texttt{ = d}_{\texttt{tt}}\texttt{ + e}_{\texttt{ss}}\texttt{ + f}_{\texttt{cal}}$

Which is turned into:

$\texttt{(U+1D44E + U+1D41B + 0x1D484 = U+1D68D + U+1D5BE + U+1D4BB)}_{\texttt{/F1}}$

This gets typeset as:

$$a + \mathbf{b} + \boldsymbol{c} = \mathtt{d} + \mathsf{e} + \mathcal{f}$$

So, something happened after reading in.

7 The consequences

I Traditional math fonts are unified using definitions in the font goodie files.

II Some macros that build symbols are turned into virtual glyphs.

III Ascii math alphabets in the input are remapped onto Unicode.

IV If possible processing is delegated Lua (and more will follow).

V Stylistic sizes are supported as well as scaled fonts.

8 Help from LUA

I Input is normalized to Unicode (relocation). This is also needed for cut and paste.

II Some sequences are collapsed (like negation) again to suit cut and paste.

III For special cases there is optional punctuation control.

IV Some fonts provide alternate math shapes, like for super- and subscripts.

V There is provisional support for auto scaled delimiters.

VI There is experimental support for math in tagged pdf and more will follow when we've redone some math constructs.

A few examples of the implementation

I The math virtual font builder runs on top of the general MkIV virtual loader.

II The characters and symbols are initialized using a database.

III Virtual fonts are defined in goodie files by specifying files and vectors.

IV Patches to fonts and parameter overload can also happen in the goodie file.

V There are tracers that can be handy when developing code or fonts.

Virtual definitions

```
return {
    name = "px-math",
    version = "1.00",
    comment = "Goodies that complement px math.",
    author = "Hans Hagen",
    copyright = "ConTeXt development team",
    mathematics = {
        mapfiles = {
            "mkiv-px.map",
        },
        virtuals = {
            ["px-math"] = {
                { name = "texgyrepagella-regular.otf", features = "virtualmath", main = true },
                { name = "rpxr.tfm", vector = "tex-mr" } ,
                { name = "rpxmi.tfm", vector = "tex-mi", skewchar=0x7F },
                { name = "rpxpplri.tfm", vector = "tex-it", skewchar=0x7F },
                { name = "pxsy.tfm", vector = "tex-sy", skewchar=0x30, parameters = true } ,
                { name = "pxex.tfm", vector = "tex-ex", extension = true } ,
                { name = "pxsya.tfm", vector = "tex-ma" },
                { name = "pxsyb.tfm", vector = "tex-mb" },
                { name = "texgyrepagella-bold.otf", vector = "tex-bf", skewchar=0x7F } ,
                { name = "texgyrepagella-bolditalic.otf", vector = "tex-bi" } ,
                { name = "lmsans10-regular.otf", vector = "tex-ss", optional=true },
                { name = "lmmono10-regular.otf", vector = "tex-tt", optional=true },
            },
        }
    }
}
```

Patches

```
local patches = fonts.handlers.otf.enhancers.patches

local function patch(data,filename,threshold)
    local m = data.metadata.math
    if m then
        local d = m.DisplayOperatorMinHeight or 0
        if d < threshold then
            patches.report("DisplayOperatorMinHeight(%s -> %s)",d,threshold)
            m.DisplayOperatorMinHeight = threshold
        end
    end
end

patches.register("after","check math parameters","asana",function(data,filename) patch(data,filename,1350) end)

local function less(value,target,original) return 0.25 * value end

return {
    name = "asana-math",
    version = "1.00",
    comment = "Goodies that complement asana.",
    author = "Hans Hagen",
    copyright = "ConTeXt development team",
    mathematics = {
        parameters = {
            StackBottomDisplayStyleShiftDown = less,
            StackBottomShiftDown             = less,
            StackDisplayStyleGapMin          = less,
            StackGapMin                      = less,
            StackTopDisplayStyleShiftUp      = less,
            StackTopShiftUp                  = less,
            StretchStackBottomShiftDown      = less,
            StretchStackGapAboveMin          = less,
            StretchStackGapBelowMin          = less,
            StretchStackTopShiftUp           = less,
        }
    }
}
```

Definitions

```
[0x007C] = {
    adobename="verticalbar",
    category="sm",
    cjkwd="na",
    contextname="textbar",
    description="VERTICAL LINE",
    direction="on",
    linebreak="ba",
    mathspec={
        {
            class="nothing",
            name="arrowvert",
        },
        {
            class="delimiter",
            name="vert",
        },
        {
            class="open",
            name="lvert",
        },
        {
            class="close",
            name="rvert",
        },
        {
            class="relation",
            name="mid",
        },
    },

    unicodeslot=0x007C,
}
```

```
[0x2111]={
    adobename="Ifraktur",
    category="lu",


    description="BLACK-LETTER CAPITAL I",
    direction="l",
    linebreak="al",
    mathclass="default",
    mathname="Im",

    specials={ "font", 0x0049 },
    unicodeslot=0x2111,
}
```

```
[0x1D69A] = {
    category="ll",



    description="MATHEMATICAL MONOSPACE SMALL Q",
    direction="l",
    linebreak="al",

    specials={ "font", 0x0071 },
    unicodeslot=0x1D69A,
}
```

Typefaces

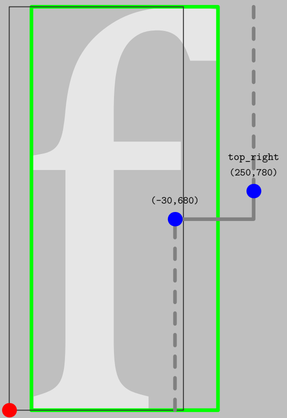
```
\starttypescript [math] [latin-modern] [size]
    .......
    \definebodyfont [10pt] [mm]
        [mr=LMMathRoman10-Regular sa 1,
         mb=LMMathRoman10-Bold    sa 1]
    .......
\stoptypescript

\starttypescript [math] [latin-modern]
    .......
    \definefontsynonym[LMMathRoman10-Regular][LMMath10-Regular@lmroman10-math]
    .......
    \definefontsynonym[LMMathRoman10-Bold]   [LMMath10-Bold@lmroman10-boldmath]
    .......
    \loadfontgoodies[lm-math]
\stoptypescript

\starttypescript [modern,default]
    \definetypeface [modern] [rm] [serif] [modern] [latin-modern]
    \definetypeface [modern] [ss] [sans]  [modern] [latin-modern]
    \definetypeface [modern] [tt] [mono]  [modern] [latin-modern]
    \definetypeface [modern] [mm] [math]  [modern] [latin-modern]
\stoptypescript
```

Virtual glyphs

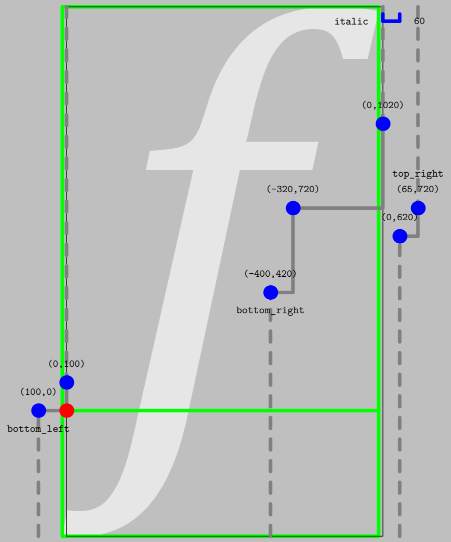
```
local function jointwo(main,characters,id,size,unicode,u1,d12,u2)
    local c1, c2 = characters[u1], characters[u2]
    if c1 and c2 then
        local w1, w2 = c1.width, c2.width
        local mu = size/18
        characters[unicode] = {
            width    = w1 + w2 - d12*mu,
            height   = max(c1.height or 0, c2.height or 0),
            depth    = max(c1.depth or 0, c2.depth or 0),
            commands = {
                { "slot", id, u1 },
                { "right", -d12*mu } ,
                { "slot", id, u2 },
            }
        }
    end
end

jointwo(main,characters,id,size,0x21A6,0xFE321,0,0x02192)             -- \mapstochar\rightar
jointwo(main,characters,id,size,0x21A9,0x02190,joinrelfactor,0xFE323) -- \leftarrow\joinrel\
jointwo(main,characters,id,size,0x21AA,0xFE322,joinrelfactor,0x02192) -- \lhook\joinrel\righ
```

Tracing

There is quite some tracing built into MkIV and there are also some extra modules, like `s-fnt-23`.



U+00066



U+1D453