Introduction

Make it easier to author equations with CSS...

Over the year it has become increasingly easier to represent equations using CSS and HTML while respecting the traditional typographic rules for this type of content. However, there are still cases where the necessary markup and styling makes it extremely difficult to author such content without the support of complex libraries or authoring software.

A few examples of some of the difficulties existing today:

- some inline styles (`vertical-align: 0.374em`) are necessary to represent some constructs such as stacked elements (fractions, matrices). These styles use font-dependent metrics, and therefore make it difficult to support styling that would alter the font family used. They are also fragile with regard to manual editing and content change.
- in other cases, glyph bounding box information is necessary to properly calculate alignment offsets, for example for placement of limits in integrals, and this information needs to be hard coded and is again font specific, fonts with may not be available when used in email or ebook systems.
- some CSS tricks used to properly represent some constructs such as fraction bars and root bars are brittle and can break when a zoom factor is applied to the rendering.

... with improvements that are not only for math

We are faced with an opportunity to simplify specialized authoring tools, make it easier for anyone to use CSS and HTML to author scientific content armed with nothing more than a text editor and to introduce to the web some typographical features that are useful beyond the narrow use case of scientific authoring.

We outline below some of the most common issues faced when laying out equations in the hope to foster a discussion on improvements that could be made to CSS to address them.

- Baseline alignment
- Stretchy constructions/fences
- Enclosures
- Roots
- Accents and decorators
- Center-line/math axis alignment

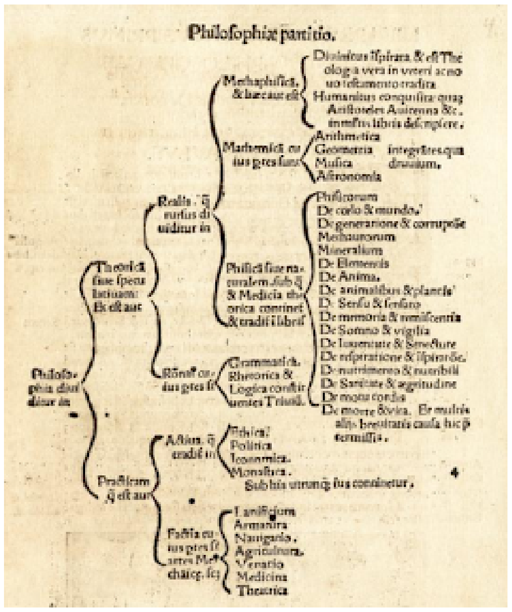
Consideration for Solutions

- **quality of rendering.** The goal of the Community Group is to have fine-grained control using CSS over layout and typography of formulas so that high-quality scientific typography comparable to what can be achieved in print can be implemented using HTML and CSS.
- **minimize the need for inline styles.**
    - reduce the need to track dimension of children for manual alignment/spacing etc
    - reduce the need to provide font metrics for tight bounding boxes, italic correction etc
- **stability of layout.** Reduce the need to recompute layout when child content changes
- **math fonts.** OpenType defines some optional information that can be embedded in a font file and used to render more accurately equations ('MATH' table). This information includes how to place mathematical superscript and subscript, stacked elements and how to composite glyphs for stretched fences. There are some fonts available with these additional tables ('math fonts') including Cambria Math from Microsoft which is pre-installed on Windows, STIX, Neo Euler, Latin Modern and others. To the extent that metadata information for accurate layout can be used from the information embedded in these fonts, they should be.

Stretchy Constructions/Fences

Problem Description

Stretchy constructions (primarily of brackets) are a common typographic feature but currently very difficult to do via CSS.



Use Cases

- https://en.wikipedia.org/wiki/Bracket#Types_and_uses
- https://macrotypography.blogspot.com/2015/11/curly-braces.html
- https://graphicdesign.stackexchange.com/questions/4819/whats-the-best-way-to-make-a-curly-brace-used-for-grouping-items-together
- https://books.google.de/books?id=LFc3DQAAQBAJ&lpg=PA46&ots=vs7rJgfz16&dq=curly%20brace%20typography&pg=PA45#v=onepage&q&f=false

Current Approaches

The typical solution is to track dimensions and manually place boxes with individual glyphs or insert SVG elements with custom dimensions.

Drawbacks

The boxes can easily misalign due to rounding errors, zoom, etc.
Hard coded knowledge of the font being used is required.

Other approaches

- Vertical stretchy construction with only 2-3 parts (with webfonts). https://codepen.io/pkra/pen/aLjGxZ
- Traditional vertical construction (switching between fonts/glyphs and stretchy construction) using element/container queries, CSS (with webfonts). https://codepen.io/pkra/pen/gvvagd
- CSS borders (no webfonts)
  - Horizontal (single-element) stretchy brace using CSS radial gradients. https://codepen.io/lrenhrda/pen/hkLIe
- unexplored ideas

- Leveraging variable fonts
- Houdini custom paint

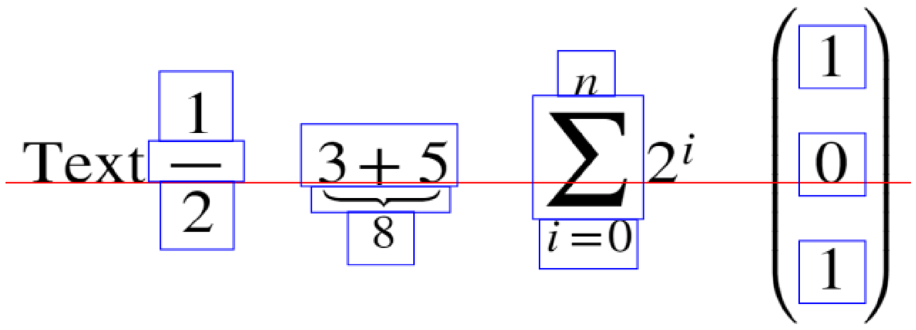
Related Problems

A notable related difficulty: in traditional (print) layout, capable fonts will ship several glyphs of different sizes and only switch to a general stretchy construction when the dimension of the child container is too great for these glyphs.
Automating this kind of switch is not possible with CSS right now (without element queries). In particular, with such an approach the height and width of the stretchy construction will depend on the dimensions of the children.

Baseline Alignment

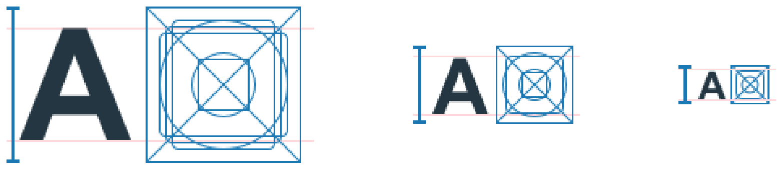
Problem Description

Aligning complex inline content with the baseline of the text is currently difficult.



Non-equation use cases

- SVG icons with text
  https://blog.prototypr.io/align-svg-icons-to-text-and-say-goodbye-to-font-icons-d44b3d7b26b4



- aligning an inline table/grid with the baseline at a particular row of the grid.

Current approaches

The dominant solution is to track dimensions and set `style="vertical-align:..."`

Other approaches

- Under/over, sub+superscripts and fractions using nested flexbox. https://codepen.io/daniwiris/pen/JOqvpB
- Under/over using nested inline-table https://codepen.io/pkra/pen/BYwBXB
- CSSWG GitHub issue: align-at-child https://github.com/w3c/csswg-drafts/issues/1339

CSS Inline Layout Module Level 3's “alignment-baseline: alphabetic” might be another path towards this feature, https://www.w3.org/TR/css-inline-3/#propdef-alignment-baseline (in the draft of 2018-08-08).

Center-line/math axis alignment

Problem statement

Center-line alignment is in some ways a more advanced case of baseline alignment. Traditional (print) equation layout engines have a concept of a "math axis" or similar which provides important information for equation layout.
The typical example is to ensure that a minus sign aligns with the fraction line or with the middle of an equality sign. Another example is the alignment of the minus sign or equal sign with the tip of a stretchy curly bracket (used in piecewise functions).

Non-equation use cases

As the name "math axis" suggests, this is fairly specific to equation layout.

Current approaches

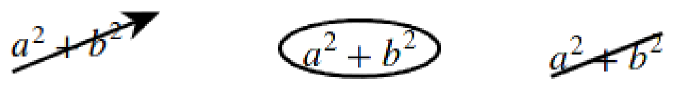
The dominant solution is to track dimensions and set style="vertical-align:..."; this is combined with webfonts that ensure glpyh and font metrics work out.

Potential simplifications

CSS Inline Layout Module Level 3 has alignment-baseline: mathematical https://www.w3.org/TR/css-inline-3/#propdef-alignment-baseline (in the draft of 2018-08-08).

Enclosures

Equation layout uses enclosures to style content.



Examples:

- (US) long division notation
- actuarial sign (wiki)
- phase-or-angle
- radical/root

- strike through lines (updiagonal, downdiagonal, vertical, horizontal)
- strike through arrows (various kinds)
- madruwb (Arabic factorial)

TeX/LaTeX packages provide an infinite variation of such designs and MathML provides the menclose element alongside an (open-ended) list of attribute names (including the above as well as various borders).
Some of these are relatively easily realized using CSS, however aligning borders across (pseudo) elements causes issues, making these solutions brittle.
A particular problem are diagonal strike through arrows and generally centering strike throughs.

Roots

While formally a special case of enclosures, roots are special in that they can also come with an additional index (cube root etc). This poses additional complications.

$$\sqrt{a^2 + b^2}$$

Accents and Decorators

For example, an arrow over a variable.

Inherits the problems of stretchy constructions with the addition that the symbol is close to another element.