Custom Material Design Typography

In a previous post I created a page using Material Design's default typography classes and the fonts they are designed to work with. What I didn't realize is that you can also create custom typographical systems using SASS and the existing typographical infrastructure for material design. I will also explore whether Recursive works well with Material Design

Getting started

Because we're using a variable font with custom axes we need to define the default values in the stylesheet's `:root` element and then use custom properties to handle inheritance problems in variable fonts as documented in Boiling eggs and fixing the variable font inheritance problem

We first define the font using the extended `@font-face` syntax for variable fonts as explained in MDN's Variable Fonts Guide.

> Note that the latest version of the Google Fonts API supports a small set of variable fonts. See Variable fonts & the new Google Fonts API and the initial anouncement in Codepen for more information on hoow to use the new API. Also note that it's not ofoficial yet so don't use it in production.

```
@font-face {
  font-family: 'Recursive'Recursive'   url('../fonts/recursive-2019_11_22-
```

Using the extended `@font-face` syntax we tell the browser that we're loading a variable font using `woff2-variations` as the format.

We also specify the `font-weight` range to be from 300 to 1000. We'll leverage this later when we setup classes for predefined instances.

In the `:root` element we use classes and variables to handle Open Type features. This section is taken from Wakamaifondue stylesheet for Cursive B025 (latest release when the post was written)

The final

```
:root {
  --recursive-aalt: "aalt"off;
  --recursive-case: "case"off;
  --recursive-dlig: "dlig"off;
  --recursive-dnom: "dnom"off;
  --recursive-frac: "frac"off;
  --recursive-numr: "numr"off;
  --recursive-ordn: "ordn"off;
  --recursive-pnum: "pnum"off;
  --recursive-sinf: "sinf"off;
  --recursive-ss01: "ss01"off;
  --recursive-ss02: "ss02"off;
  --recursive-ss03: "ss03"off;
  --recursive-ss04: "ss04"off;
  --recursive-ss05: "ss05"off;
  --recursive-ss06: "ss06"off;
  --recursive-ss07: "ss07"off;
  --recursive-ss08: "ss08"off;
  --recursive-ss09: "ss09"off;
  --recursive-ss10: "ss10"off;
  --recursive-ss11: "ss11"off;
  --recursive-ss20: "ss20"off;
  --recursive-sups: "sups"off;
  --recursive-titl: "titl"off;
  --recursive-zero: "zero"off;
}
```

We then create classes for each opentype feature. If class is applied, update custom property and apply modern font-variant-* when supported.

In this case it is safe to use `@supports` because if the browser supports variable fonts I feel coonfident that it supports feature queries.

The final block of this section takes all the values of the opoen type variables and sets them appropriately using font-variation-settings

```
.recursive-aalt {
  --recursive-aalt: "aalt"on;
}

"aalt".recursive-case recursive-case: "case"on;
}

.recur"case".recursive-dlig ive-dlig: "dlig"on;
}

@supports (f"dlig"@supports (font-variant-ligatures: discretionary-ligatu
  .recursive-dlig dlig: "____";
    font-variant-ligatures: discretionary-ligatures;
  }
}

.recursive-dnom {
"____".recursive-dnom {
  --recursive-dnom: "dnom"on;
}

.recursive-frac {
  --recursive-frac: "frac"on;
}

@supports (font-variant-numeric: diagonal-fractions) {
  .recursive-frac "____";
    font-variant-numeric: diagonal-fractions;
  }
}

.recursive-numr {
  --re"____".recursive-numr {
  --recursive-numr: "numr"on;
}

.recursive-ordn {
  --recursive-ordn: "ordn"on;
```

```
}

@supports (font-variant-numeric: ordinal) {
  .recursive-ordn {
    --recursive-ordn: "____";
    font-variant-numeric: ordinal;
  }
}

.recursive-pnum {
  --recursive-pnum: "pnum"on;
}

@supports (font-variant-numeric: proportional-nums) {
  .recursive-pnum {
    --recursive-pnum: "____";
    font-variant-numeric: proportional-nums;
  }
}

.recursive-sinf {
  --recursive-sinf: "sinf"on;
}

.recursive-ss01 {
  --recursive-ss01: "ss01"on;
}

.recursive-ss02 {
  --recursive-ss02: "ss02"on;
}

.recursive-ss03 {
  --recursive-ss03: "ss03"on;
}

.recursive-ss04 {
  --recursive-ss04: "ss04"on;
```

```
}

.recursive-ss05 {
  --recursive-ss05: "ss05"on;
}

.recursive-ss06 {
  --recursive-ss06: "ss06"on;
}

.recursive-ss07 {
  --recursive-ss07: "ss07"on;
}

.recursive-ss08 {
  --recursive-ss08: "ss08"on;
}

.recursive-ss09 {
  --recursive-ss09: "ss09"on;
}

.recursive-ss10 {
  --recursive-ss10: "ss10"on;
}

.recursive-ss11 {
  --recursive-ss11: "ss11"on;
}

.recursive-ss20 {
  --recursive-ss20: "ss20"on;
}

.recursive-sups {
  --recursive-sups: "sups"on;
}
```

```
@supports (font-variant-position: super) {
  .recursive-sups {
    --recursive-sups: "____";
    font-variant-position: super;
  }
}

.recursive-titl {
  --recursive-titl: "titl"on;
}

@supports (font-variant-caps: titling-caps) {
  .recursive-titl {
    --recursive-titl: "____";
    font-variant-caps: titling-caps;
  }
}

.recursive-zero {
  --recursive-zero: "zero"on;
}

@supports (font-variant-numeric: slashed-zero) {
  .recursive-zero {
    --recursive-zero: "____";
    font-variant-numeric: slashed-zero;
  }
}

.recursive-aalt,
.recursive-case,
.recursive-dlig,
.recursive-dnom,
.recursive-frac,
.recursive-numr,
.recursive-ordn,
.recursive-pnum,
.recursive-sinf,
```

```
.recursive-ss01,
.recursive-ss02,
.recursive-ss03,
.recursive-ss04,
.recursive-ss05,
.recursive-ss06,
.recursive-ss07,
.recursive-ss08,
.recursive-ss09,
.recursive-ss10,
.recursive-ss11,
.recursive-ss20,
.recursive-sups,
.recursive-titl,
.recursive-zero {
  font-feature-settings: var(--recursive-aalt),
    var(--recursive-case), var(--recursive-dlig), var(--recursive-dnom), v
}
```

The next block uses pre-defined instances of the Recursive font. Because the values are exclusive to the instance I didn't think it necessary to deefine them using CSS variables. It's possible but too time consuming for me.

`font-weight` is a predefined axis for variable fonts and it's supported well enough that we can take it out of `font-variation-settings` and use the CSS property on its own.

Inside `font-variation-settings`, the uppercased axes (MONO and CASL) are custom axes to the font we're using so they will always need to go here as there are no equivalent CSS properties.

The lowercased axes (slnt and ital) are predefined axes but need to go inside `font-variation-settings` to avoid confusion.

These are examples of the instance classes produced by Wakamaifondue. I didn't want to list all 64 instances here :-)

```
// Variable instances.
```

```
.recursive-mono-linear {
  font-weight:  400;
  font-variation-settings: "MONO"1, "CASL"0, "slnt"0, "ital"0.5;
}

"MONO".recursive-mono-linear-italic nt-weight:  400;
  font-variation-settings: "MONO"1, "CASL"0, "slnt"-15, "ital"1;
}

.recur"MONO".recursive-mono-casual ght:  400;
  font-variation-settings: "MONO"1, "CASL"1, "slnt"0, "ital"0.5;
}

.recursive-m"MONO".recursive-mono-casual-italic {
  font-weight:  400;
  font-variation-settings: "MONO"1, "CASL"1, "slnt"-15, "ital"1;
}
```

After all the work setting Open Type features and variable font instances we can work with Material Design using our custom font.

We import the SCSS files for typography and grid to include in the final result.

The last thing we do is to override the base font family too use Recursive. Because the variable font can be either sans serif or mono spaced you will have to override the font family everywhere the font is mono spaced.

```
@import "@mater"@material/typography/mdc-typography"@import "@material/lay

$mdc-typography-font-family: unquote("Recursive, Arial, Helvetica");
```

You can see an example in action here: Towards the Splendid City and the code is here