Crafting Type For The Web

As I begin working on layouts one of the things that started to beg for attention is how to size the text properly for it to be legible online as well as being pleasant to the eye when looking at the page.

> These are notes, observations and learning points from books, websites and the work I've been doing for my layout experiments site.
>
> They also span a long time between writing so they may not read quite the same.

The 62.5% system

When I first started working on the web it was common to see the following css at the very beginning of a stylesheet:

```
body {
  font-size: 62.5%;
  /* Other rules that apply globally */
}
```

This would set the default font size to the document to 10px. Designers would then have to adjust the sizes of most elements based on the 10px default value for example:

```
/* Sets copy size to 16 pixels*/
body {
  font-size: 1.6em;
}

/* Sets h1 to 30 pixels*/
h1 {
  font-size: 3em;
}
```

For core elements this is not too bad, the bigger issue becomes when we use nested elements. Because we set our default font to 10 pixels, we need to set the font size on every single element. Quick example:

```
p, li, td, input, button, label {
    font-size: 1.6em;
}
```

This turns into a nightmare when you have to nest elements. For instance a p inside a li. This p is now 1.6×1.6em. So in order to fix this we have to do something like this:

```
li p, label input, li label, td li {
    font-size: 1em;
}
```

It's doable but it's a royal pain in the ass. There has to be a better way to do this.

Type Scales

In More Meaningful Typography Tim Brown states that:

> We have all heard of the golden mean (also known as the golden ratio or golden section): the self-replicating page with a proportion of 1:1.618 that is said to be found in everything from the design of ancient Greek architecture to the growth patterns of plants.

He further defines a modular scale as:

> … a sequence of numbers that relate to one another in a meaningful way. Using the golden ratio, for example, we can produce values for a modular scale by multiplying by 1.618 to arrive at the next highest number, or dividing by 1.618 to arrive at the next number down.

In these examples we'll use a base of 16px and a golden ratio scale (1 : 1.618)

```
html {
  font-size: 100%;
}

h1 {
  /* 3 steps up on the modular scale */
  font-size: 4.854rem
}

p {
  font-size: 1rem;
}
```

You can create your own Modular Scale using a tool developed by Scott Kellum and Tim Brown and use it to test different scales, not just the golden ratio.

Percentages and ems for the win?

One last possibility is using relative values. In the article The EMs have it: Proportional Media Queries FTW! Lyza Gardner proposes an interesting conundrum when using pixels or points to define both our font sizes and media queries

If you use a fixed size for your media queries and font sizes those will not change when the user zooms in using the browser's built in zooming mechanism? as Lyza writes:

> I'm using the Chrome browser, and I'm viewing our site with a window about 670 pixels wide. With a pixel-based media query, that puts me in the second category of nav experience: all of the top level items are shown horizontally, docked to the top of the content:

Figure 1: Cloud Four Homepage with navigation menu fully displayed

> OK, now I'm going to use the Zoom In command twice to make my text larger.
>
> Figure 2: Cloud Four Homepage with navigation menu zoomed in using a pixel-based media query
>
> Figure 3: Cloud Four Homepage with navigation menu zoomed in using an em-based media query

From reading the article it appears that the media query that the author used did not change when it used pixel values but it did change when we used an em value.

If we're to follow this line of thought we have to change our CSS to something like this:

```
body {
  font-size: 100%;
}

p {
  font-size: 1rem;
}
```

This means that, no matter what size our font ends up at, the base font will remain at 100% of that size... If we also use relative units in our Media Queries they will all

look nice at whatever zoom level or whatever screen size we are at.

How many overrides do we need?

Once we change the default size we can no longer rely on the user agent (AKA: browser) stylesheet to modify the values of those elements we don't explicitly change ourselves. To me that has always begged the question, how many values do we need to change? Are we ok with changing only the size of the elements we use?

We'll explore this in more detail when we convert the values to code.

What values to use

Now let's talk about what values to use.

Measure (characters per line)

> Anything from 45 to 75 characters is widely regarded as a satisfactory length of line for a single-column page set in a serifed text face in a text size. The 66-character line (counting both letters and spaces) is widely regarded as ideal. For multiple column work, a better average is 40 to 50 characters."
>
> — Elements of Typographic Style – Section 2.1.2

The number of characters per line of body text (the CPL or measure) should fall below 100 characters: 80 – 95 characters per line is ideal for single column text. For two column text half the value.

In most cases, text size should be based on this requirement: i.e. fonts should be scaled up or down, or the width of containers adjusted, until you have roughly 80 characters per line. Most sidebar text should be set to approximately 35 characters per line. These settings will need to change as screen sizes narrow, using the principles of responsive design.

If the text is too wide or too narrow we get different kinds of problems:

Too wide – if a line of text is too long the reader's eyes will have a hard time

focusing on the text. Furthermore it can be difficult to continue onto the correct line in large blocks of text.

Too narrow – if a line is too short the eye will have to travel back too often, breaking the reader's rhythm. Too short lines also tend to stress readers, making them begin on the next line before finishing the current one and, potentially, skipping important words.

```
.container {
  width: 20em;
}
```

A lot of other elements influence the width of the text. The first thing to about is the size of the font, remembering that not all fonts have the same width. Another thing to consider is the leading of your content; the separation between lines (discussed in more details later) will affect how the text reads.

As with many of these things you'll have to experiment with the values for the fonts and the design you're working with.

Leading (Line Height)

> "Vertical space is metered in a different way [to horizontal space]. You must choose not only the overall measure – the depth of the column or page – but also a basic rhythmical unit. This unit is the leading, which is the distance from one baseline to the next."
>
> — Elements of Typographic Style – Section 2.1.2

The one thing I always forget is that **line-height spacing is added equally above and below the text** so we need to be careful when setting the `line-height` attribute to remember that the distance between elements is twice that which we set in the attribute so be mindful that you don't pick too large a number.

```
body {
  font-size: 16px;
}
```

```
p {
  font-size: 1em;
  /* This sets a space of 2.5 units between paragraphs */
  line-height: 1.25;
}
```

It is also important to note that `line-height` is the only attribute in CSS that can and should be used without a unit.

Font size

Generally speaking, readers want text to be larger than most designers are comfortable with. As a designer, you are looking at the content of a page every day; after a period of time, you will stop "seeing" it, and assume that everyone reads the text as instinctively as you do. It's generally a good rule to size text slightly larger than you think it needs to be.

I keep the default font size that is about 14 points and 16 pixels. It works well for most of the projects I develop. As with all the things in this essay, your mileage may vary.

Translating the values to code

When starting a new project I work with the following values derived from a modular scale using 16px as my base and the Golden Ratio as my scale. The one deviation from a strict modular scale is that I sometimes uses fractions like in the headings below, mostly because I don't want my larger headings to be too larger and still want to differentiate the different headings.

```
:root {
  font-size: 100%;
}
body {
  font-family: Roboto, "Open Sans", Arial, serif;
  line"Open Sans"25;
}
```

```
h1 {
  /* 4 steps up in the modular scale */
  font-size: 6.472rem;
}

h2 {
  /* 3.5 steps up in the modular scale */
  font-size: 5.663rem;
}

h3 {
  /* 3 steps  up in the modular scale */
  font-size: 4.854rem;
}

h4 {
  /* 2.5 steps up in the modular scale */
  font-size: 4.045rem;
}

h5 {
  /* 2.0 steps up in the modular scale */
  font-size: 3.236rem;
}

h6 {
  /* 1.5 steps up in the modular scale */
  font-size: 2.427rem;
}

p {
  font-size: 1rem;
}
```

This is a good starting point but it’s not perfect. We have to make sure that the measure, leading and font size all work together to make a good reading experience.

You may also have to change the leading in the headings to accommodate the larger font size and, as you can imagine, you'll have to do the same thing for other elements in your page. I normally make my blockquotes slightly larger in size, for example. Your particular design may need similar modifications.

This scale will need modifiifications based on the font and the type of reading block that you're building.

Media Queries

How many breakpoints should we use for our media queries should we use?

To start I'll take the SASS Mixins from The 100% correct way to do CSS breakpoints and use them as the starting point for my own work and researcgh

```
@mixin for-phone-only {
  @media (max-width: 599px) { @content; }
}
@mixin for-tablet-portrait-up {
  @media (min-width: 600px) { @content; }
}
@mixin for-tablet-landscape-up {
  @media (min-width: 900px) { @content; }
}
@mixin for-desktop-up {
  @media (min-width: 1200px) { @content; }
}
@mixin for-big-desktop-up {
  @media (min-width: 1800px) { @content; }
}
```

The idea is that starting from the top each query will match all elements above it so that the last one that matches (if more than one does) will apply to the document.

You don't have to apply all queries to all elements to your page.

You are not limited to the default set of media queries. There may be situations where we need special behavior for certain form factors. For example,

hover behaviors dob't work properly in mobile devices so we may want to remove any hover styles and let Javascript and Pointer Events handle hover in a way that works across mouse and touch devices.

Use the font, Luke

Rather than provide a single solution that will (not) work on all fonts it would work better if you create a block of text (based on your site/app content) and match it with fonts that you want to use for headings and any other specialized text blocks.

If you want a more detailed reference, look at Chapters 2, 3, and 4 of Flexible Typesetting by Tim Brown to see the process we should follow.

Leverage OpenType Features

OpenType fonts have features built into the fonts that allow for a variety of tricks and modifications to the font for specific purposes. Explore what your chosen font can do and how well do the extra features match your needs (or don't).

Variable fonts

Variable fonts introduce interesting possibilities to typesetting for the web. Imagine if using one (variable) font would cover all the needs for your page.

Using tools like Wakamaifondue you can inspect your variable fonts and get the different predefined instances but also the Open Type features available on the font itself.

It will also generate a CSS stylesheet that you can drop into your projects using that font to use all the Variable font instances and OpenType features by adding classes to your content.

It's ok to leave and come back :)

I've been in situations where I've looked at the page for so long that I'm seeing the result I want rather than what's on the page and am surprised when people who are seeing the content fresh point to the width of the text or the leading in the copy as not being correct.

Test, test again, leave it for a few days and then do some more testing... It never ends.