Figures and Counters

Playing with figures and counters can produce some very interesting bits of presentation for our content.

The post will talk about the `figure` and `figcaption` elements, what they do and how to use them.

We will then discuss CSS counters, what they do and how they work when inserted into HTML elements.

Finally we'll look at what to do with figures and counters together. I hope you can find your own creative uses.

Figure and figcaption

The HTML `figure` represents self-contained content with an optional caption represented by the `figcaption` child element. The figure, its caption, and its contents are referenced as a single unit.

I've always thought of figures as images with captions but the HTML spec says something different about the element:

> The figure element represents some flow content, optionally with a caption, that is self-contained (like a complete sentence) and is typically referenced as a single unit from the main flow of the document.
>
> "Self-contained" in this context does not necessarily mean independent. For example, each sentence in a paragraph is self-contained; an image that is part of a sentence would be inappropriate for figure, but an entire sentence made of images would be fitting.

> The element can thus be used to annotate illustrations, diagrams, photos, code listings, etc.
>
> From: https://html.spec.whatwg.org/#the-figure-element

One thing that the specification cautions authors is how to reference figures (any kind) from within the document:

> When a figure is referred to from the main content of the document by identifying it by its caption (e.g., by figure number), it enables such content to be easily moved away from that primary content, e.g., to the side of the page, to dedicated pages, or to an appendix, without affecting the flow of the document.

> If a figure element is referenced by its relative position, e.g., “in the photograph above” or “as the next figure shows”, then moving the figure would disrupt the page’s meaning. Authors are encouraged to consider using labels to refer to figures, rather than using such relative references, so that the page can easily be restyled without affecting the page’s meaning.
>
> From https://html.spec.whatwg.org/#the-figure-element

So, according to the HTML specification we can do something like this:

```html
<figure>
  <ul>
    <li>Item 1</li>
    <li>Item 2</li>
    <li>Item 3</li>
  </ul>
  <figcaption>List 1: Example list</figcaption>
</figure>
```

This example has a few drawbacks. The main one is that it hardcodes the number of the item in question, in this case the list number. THis means that if we inser list prior to the one we labeled number one, we’ll have to manually change the numbers. This is error prone and brittle if we have too omany items of one kind in the document.

We can leave it as Example List but that’s confusing, particularly if you have to many similar items. It may also make it difficult to reference individual items as suggested in the specification.

CSS Counters

CSS Counters address the need for a means to automate the numbering of figure items. The idea is that we use counters and insert them, along with acompanying text, in the place we specify.

The process is as follows:

1. We initialize the counters by using `counter-reset` with the name of the counters we want to reset
   1. We can repeat this for multiple counters
   2. The `counter-reset` function can be placed anywhere in the stylesheet. In this example we placed the initialization in the root element selector
2. Wherever we want to increase the value of the counter we use `counter-increment`
   1. We can repeat this for multiple counters
   2. The `counter-increment` function can be placed anywhere in the stylesheet. In this example we placed the increment in the `section h2::before` selector
3. Insert the content you want to display using a combination of text and the `counter(counter-name)` for the counter that you want to use inside a `content` property

The CSS code example below illustrates how to use counters to automatically set up a counter for chapter numbers and attach these counters to the first h2 element inside a section.

```css
body {
  counter-reset: chapter;
}

section h2::before {
  counter-increment: chapter;
  content: "Chapter " counter(chapter) ": ";
}
"Chapter "
```

The HTML shows one possible way to write HTML that uses the CSS in the previous block to auto number the sections we use as chapter separators.

```html
<h2>Book Title</h2>

<section>
  <h2>Example Chapter</h2>

  <p>Some text for the chapter.</p>
</section>
```

You can see a working example in this code pen

See Automatic Numbering With Counters in the CSS Lists Module Level 3 specification for more information and additional counter functionality.

Experimenting

The following are some examples of what we can do with counters and how to best leverage counters and figures to make our content more appealing and easier to read.

Multiple counters

The first experiment is how to handle multiple counters in the same document. The example CSS code below works with four different counters: chapters, figures, tables and lists with each of these counters using different elements too trigger counter increment and displaying the content.

I picked these particular types of counters to illustrate different ways to use them.

The initialization works the same as the single counter example. `counter-reset` can take more than one value separated by spaces.

```css
:root {
  /* initializing counters */
  counter-reset: chapter figure tables lists;
}
```

The `chapter` counter works the same as our previous example.

```
section {
  counter-increment: chapter;
}
section h2::before {
  content: "Chapter " counter(chapter) ": ";
}
"Chapter "
```

The `figure` counter works with our default figure setup without adding any additional information.

```
figure {
  counter-increment: figure;
}
figure figcaption::before {
  content: "Figure " counter(figure) ": ";
}
"Figure "
```

The `table` counter works with the HTML table elements and uses the `caption` child to display the counter in a similar way to what the default figure counter does.

```
table {
  counter-increment: tables;
}
table caption {
  padding: 10px;
  caption-side: bottom;
}
table caption::before {
  content: "Table " counter(tables) ": ";
}
"Table "
```

The `lists` counter uses a modified version of the `figure` counter to account for it being a special type of figure.

```
figure.lists {
  counter-increment: lists;
}
figure.lists figcaption::before {
  content: "List " counter(lists) ": ";
}
"List "
```

Counters reset at chapter level

The previous example initialized/rest the counters in the root element of the stylesheet. There may be times when we want to reset items further dow the style tree.

This example will reset the `figure` counter on every section, restarting the figure numbering on every section, regardless of how many we figures we have on each chapter.

```
:root {
  counter-reset: chapter;
}

section {
  counter-reset: figure;
  counter-increment: chapter;
}
section h2::before {
  content: "Chapter " counter(chapter) ": ";
}
```

```
"Chapter "figure {
  counter-increment: figure;
}
figure figcaption::beforent: "Figure " counter(figure) ": ";
}
"Figure "
```

Using more than one counter

So far all our examples have used a single counter to show the positioning of a item on the page.

There are times where we want to show more specific information about the image, for example: this is the first image of chapter two or the third table on chapter 5.

To do this we have to use more than one counter when writing the counter content.

We'll take the same code as the previous example and change the content inside `figcaption::before` to add the chapter counter and a string to separate the chapter counter from the figure counter. The rest of the code is the same.

```
:root {
  counter-reset: chapter;
}

section {
  counter-reset: figure;
  counter-increment: chapter;
}
section h2::before {
  content: "Chapter " counter(chapter) ": ";
}

"Chapter "figure {
  counter-increment: figure;
}
figure figcaption::beforent: "Figure " counter(chapter) '-' counter(figur
}
"Figure "
```

Using counters is not exclusive to figures. It’s just a matter oof hoow creative you are and what you’re trying to accomplish.