Interesting things I learnt about layout
(and general CSS...)

Chen Hui Jing / @hj_chen

Do you box model?
Developer Advocate

nexmo®
The Vonage® API Platform

SingaporeCSS

https://singaporecss.github.io

@SingaporeCSS | @hj_chen | @wgao19
Initial value of `display` for all elements is `inline`.

Then how come `<div>`s, paragraphs, lists and the like are `<display: block>`?
Because browser default stylesheets.

- Firefox stylesheet: `resource://gre-resources/html.css`
- Link to Chromium stylesheet
Inline-level element behaviour

- `inline`, `inline-table`, `inline-block`, `inline-flex`, `inline-grid`
- `width` and `height` property does not apply
- Height of content is based on font size
- `vertical-align` property only applies to inline-level and table-cell elements
- Only margins, borders and paddings along the inline axis have any visible effect on an inline box
If an element generates zero boxes, was it really there at all?
Block formatting contexts

The context that block-level boxes participate in

Boxes are laid out one after another, in the block flow direction, from the start of the containing block

Margins along the block flow direction between adjacent block-level boxes in the same block formatting context collapse
What establishes new block formatting contexts?

- Floats
- Absolutely positioned elements
- Block containers that are **not** block boxes
- Block boxes with overflow **other than** visible
- Boxes with display set to **flow-root**
We need a new BFC because...?

1. Prevent collapsing margins

This is a line of text in a p tag.

I'm a box with margins.

I'm another box with margins.

```html
<p>This is a line of text in a p tag.</p>
<div class="block-wrapper">
  <div class="box1">I'm a box with margins.</div>
  <div class="box2">I'm another box with margins.</div>
</div>

.collapse .box2 {
  margin: 0.5em;
  display: inline-block;
}
```
2. Stop text from flowing around the float

I'm a floated box! This is just a bunch of text that is going on and on so it's long enough to wrap around the float, line boxes yo!

```html
<div class="block-wrapper">
  <div class="box1">I'm a floated box!
    <p class="box2">This is just a bunch
    </p>
  </div>
</div>
```

```css
.stop-flow .box1 {
  float: left;
}
.stop-flow .box2 {
  overflow: auto;
}
```
3. Contains floats

Floaty! ^_^ Floaty too! :)
Let’s talk about margin collapsing

1. Between adjacent siblings

```css
.siblings {
  writing-mode: horizontal-tb;
}

.siblings .brother {
  margin-bottom: 1em;
}

.siblings .sister {
  margin-top: 1em;
}
```
2. Between empty boxes

.main { writing-mode: horizontal-tb; }

.main .nothing { margin-top: 1em; margin-bottom: 1em; }

```
.Got stuff
```

```
.Not empty
```

3. Parent and first / last child element

This is the parent element

This is a child element

This is a child element

```
.family .parent {
    writing-mode: horizontal-tp;
}

.family .child {
    margin-bottom: 1em;
}
```
We prevent margin collapse by...

1. Adding something in between the elements

```
.fix-collapse .nothing {
  margin-bottom: 1em;
  margin-top: 1em;
  padding: 0.009px;
}
```
2. Add border to the parent element

```css
.fix-collapse .parent {
    border: 3px solid;
}

.fix-collapse .child {
    margin-bottom: 1em;
}
```
3. Create a new BFC

Refer to section on block formatting contexts 👍
Everything You Need To Know About CSS Margins 🤓

Grid gaps

May cause overflow if you're not careful

```css
.gridgap .grid {
  display: grid;
  grid-template-columns: repeat(4, 25%);
  gap: 1em;
}
```
Overscroll and padding

padding at end side of overflow scroll container not applied

```css
overflow-x: scroll;
padding: 1em;
}

.flexpad .flex__item {
  flex: 1 0 auto;
}

.flexpad .flex::after {
  content: '';  
padding-right: 1em;
}
```
Use the flex shorthand

“Authors are encouraged to control flexibility using the flex shorthand rather than with its longhand properties directly, as the shorthand correctly resets any unspecified components to accommodate common uses.”
About shorthands...

The `border` shorthand

- Sets the same width, colour and style for all four borders of a box
- Unlike `margin` and `padding` shorthands, it **cannot** set different values on the four borders
- Also resets `border-image` to initial value

“It is therefore recommended that authors use the `border` shorthand, rather than other shorthands or the individual properties, to **override any border settings earlier** in the cascade.”
The background shorthand

\(<\text{bg-layer}>\#, \ <\text{final-bg-layer}>\)

where

\(<\text{bg-layer}> = <\text{bg-image}> \ || \ <\text{bg-position}> [ / <\text{bg-size}> ]? \ || \ <\text{repeat-style}> || <\text{attachment}> || <\text{box}> || <\text{box}>\)

\(<\text{final-bg-layer}> = <'\text{background-color}'> \ || \ <\text{bg-image}> \ || \ <\text{bg-position}> [ / <\text{bg-size}> ]? \ || \ <\text{repeat-style}> || <\text{attachment}> || <\text{box}> || <\text{box}>\)

- at least 1 value must occur, the rest is pretty much up to you
- for <position>, can optionally include <bg-size>
- entire set for <bg-layer> can occur multiple times, comma-separated
- only <final-bg-layer> can have <'background-color'>'
“Given a valid declaration, for each layer the shorthand first sets the corresponding layer of each of background-image, background-position, background-size, background-repeat, background-origin, background-clip and background-attachment to that property’s initial value, then assigns any explicit values specified for this layer in the declaration. Finally background-color is set to the specified color, if any, else set to its initial value.”
The animation shorthand

\[\text{<single-animation>}\#\]

where

\[\text{<single-animation>} = \text{<time>} || \text{<easing-function>} || \text{<time>} || \text{<single-animation-iteration-count>} || \text{<single-animation-direction>} || \text{<single-animation-fill-mode>} || \text{<single-animation-play-state>} || [\text{none} | \text{<keyframes-names>} ]\]
Don’t forget about the cascade
Also, reading CSS specifications
CSS property syntax

Loosely based on the Backus-Naur Form (BNF) notation

A sandwich consists of a lower slice of bread, mustard or mayonnaise; optional lettuce, an optional slice of tomato; two to four slices of either bologna, salami, or ham (in any combination); one or more slices of cheese, and a top slice of bread.

```
sandwich ::= lower_slice [ mustard | mayonnaise ] lettuce? tomato? [ bologna | salami | ham ]{2,4} cheese+ top_slice
```

Analogy from How to Read W3C Specs.

Slides | Cheatsheet
Thank you!

https://www.chenhuijing.com

@hj_chen

@hj_chen

@huijing

Header font is Biorhyme by Aoife Mooney
Body font is Cabin by Pablo Impallari