CSS FOR BACKEND DEVS

Created by Marta Sztybor (http://martasztybor.pl) / @sztyborek (http://twitter.com/sztyborek)

IF YOU'VE EVER FELT LIKE THIS WHILE EDITING CSS...

...THIS TALK IS FOR YOU.

LET'S TALK ABOUT BASICS FIRST

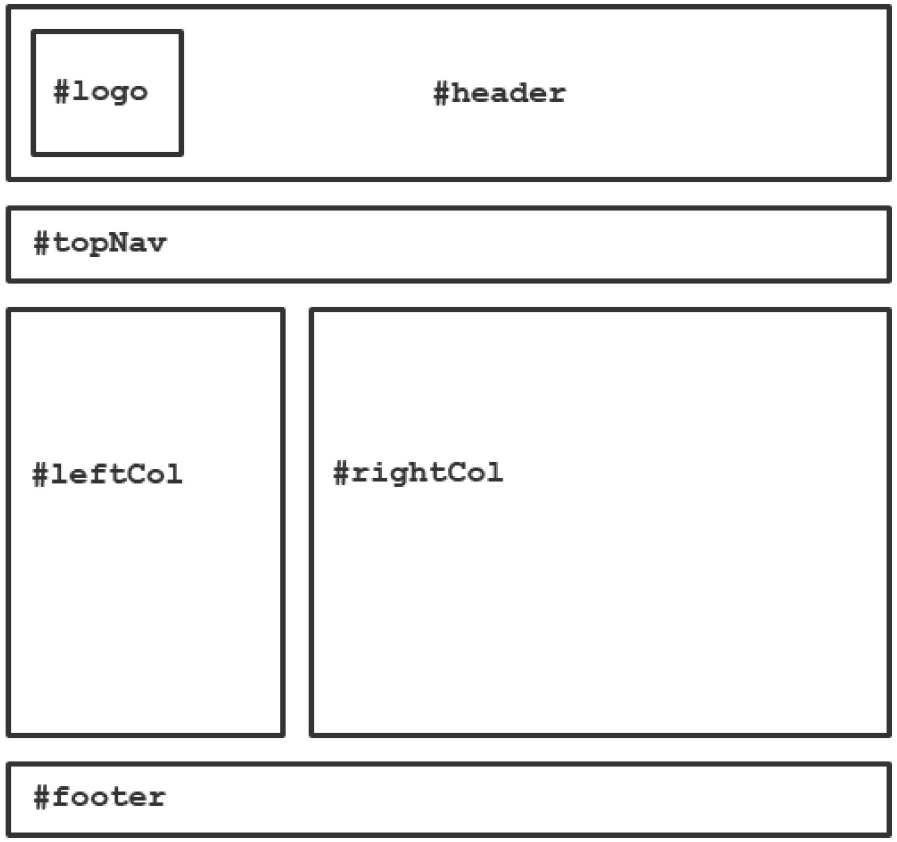
HTML SEMANTICS!

> Semantic HTML is the use of HTML markup to reinforce the semantics, or meaning, of the information in webpages and web applications rather than merely to define its presentation or look.
>
> -- Wiki (https://en.wikipedia.org/wiki/Semantic_HTML)

WHY?

- It's SEO-friendly.
- Solves most problems with accessibility (https://www.marcozehe.de/2015/12/14/the-web-accessibility-basics/) (eg. using **`button`** tag for buttons, not styled **a** tag).
- It adds meaning and increases readability of your code.

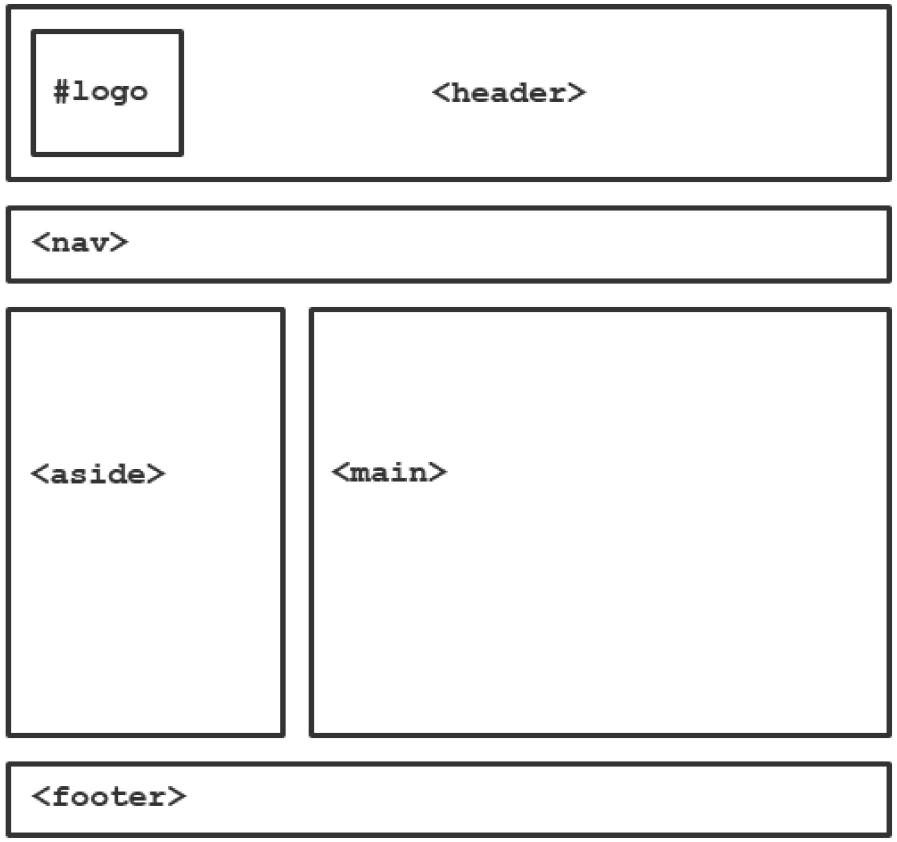
HOW?




```
<div id="header">
        <div id="logo"></div>
</div>
<div id="topNav"></div>
<div id="leftCol"></div>
<div id="rightCol"></div>
<div id="footer"></div>
```

Divitis!

THE SOLUTION




```
<header>
  <div id="logo"></div>
</header>
<nav></nav>

<aside></aside>
<main></main>

<footer></footer>
```

RESOURCES

- MDN HTML element reference (https://developer.mozilla.org/en/docs/Web/HTML/Element)
- HTML document outline on MDN (https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Sections_and_Outlines_of_an_HTML5_documen
- Let's talk about semantics (http://html5doctor.com/lets-talk-about-semantics/
- Accessibility basics (https://www.marcozehe.de/2015/12/14/the-web-accessibility-basics/) - how the HTML semantics matters for accessibility
- A look into proper HTML5 semantics (http://www.hongkiat.com/blog/html-5-semantics/)

BEFORE YOU BEGIN STYLING

- It's important to add some normalize (https://necolas.github.io/normalize.css/) or reset (http://meyerweb.com/eric/tools/css/reset/).
- They'll help you to cope with most of the browser inconsistencies.
- Such stylesheets are usually included in frameworks (usually don't bother if you are using a framework).
- Why? I'll explain in the next few slides.

CSS RULES' WARS

- Cascading
- Inheritance
- Specificity

SPECIFICITY

SELECTORS SPECIFICITY SORTED FROM LOWEST TO HIGHEST

1. Element and pseudo-element

```
div {}
p::after {}
```

2. Class, pseudo-class and attribute

```
.container {}
.list-item:first-child {}
[href^='https://'] {}
```

3. IDs

```
#users-chart {}
#main-nav {}
```

4. Inline styles

```
<ul style="list-style-type: none"></ul>
```

Attributes with **`!important`** overide even inline styles.

CALCULATING SPECIFICITY

1. For every element or pseudo-element add 1
   0 0 0 1

2. For every class, pseudo-class or attribute add 10
   0 0 1 0

3. For every ID add 100
   0 1 0 0

4. For inline style add 1000
   1 0 0 0

WHICH SELECTOR WINS THE BATTLE?

```
UL#MAIN-NAV .SELECTED-ITEM [HREF=^"HTTPS://"]
```

0 1 2 1

```
UL.NAV-WRAP LI:NTH-CHILD(5) A
```

0 0 2 3

MAY THE SPECIFICITY FORCE BE WITH YOU

- Try to keep your selectors specificity as low as possible, as they'll be easier to understand and maintain.
- In general, it's recommended to organize selectors in a stylesheet with an increasing specificity.
- Avoid ID selectors, because they are hard to override.
- Try using naming conventions such as BEM (http://getbem.com/introduction/), BEMIT (http://csswizardry.com/2015/08/bemit-taking-the-bem-naming-convention-a-step-further/) or SUIT (http://suitcss.github.io/).

RESOURCES

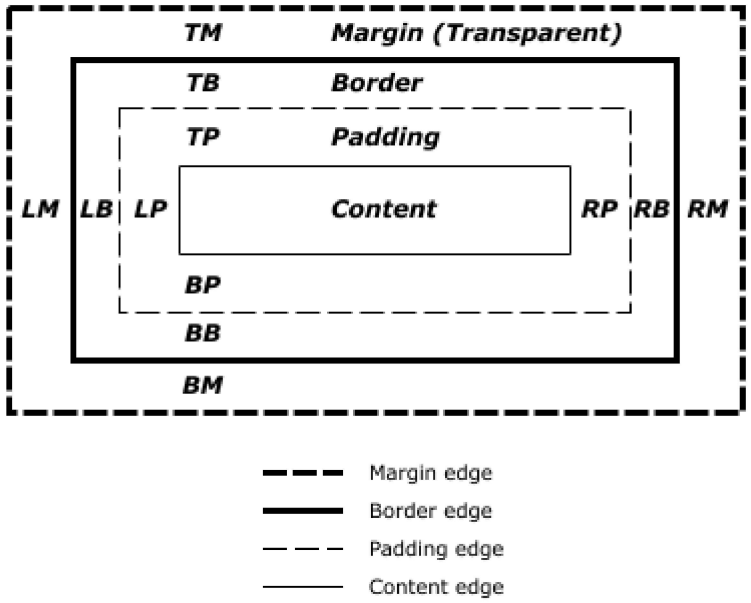
- About CSS specificity (https://developer.mozilla.org/en-US/docs/Web/CSS/Specificity) on MDN
- CSS Specificity is base-infinite (https://css-tricks.com/css-specificity-is-base-infinite/) on CSS-Tricks

BOX MODEL

> Mama always said life was like a box of chocolates. You never know what you're gonna get.
>
> -- Forest Gump

> CSS is like a box of chocolates. You never know what you're gonna get.
>
> -- Developer

WHAT IS THE MAGICAL BOX MODEL?



source: W3C (http://www.w3.org/TR/CSS21/box.html)

THE PROBLEM


```
.box-of-chocolates {
        width: 300px;
        padding: 20px;
        border: 1px solid #333;
}
```

Declared box width: 300px

Rendered box width: 342px

Oh my! It doesn't compute!

THE SOLUTION

```
* {
  box-sizing: content-box;
}
```

Rendered width = content + padding + border

```
* {
  box-sizing: border-box;
}
```

Rendered width = content width

RESOURCES

- CSS Box Sizing by MDN (https://developer.mozilla.org/en/docs/Web/CSS/box-sizing)
- Box sizing explained on CSS-Tricks (https://css-tricks.com/box-sizing/)
- * { Box-sizing: Border-box } FTW (http://www.paulirish.com/2012/box-sizing-border-box-ftw/) by Paul Irish

HOW TO DISPLAY?

CSS DISPLAY PROPERTIES

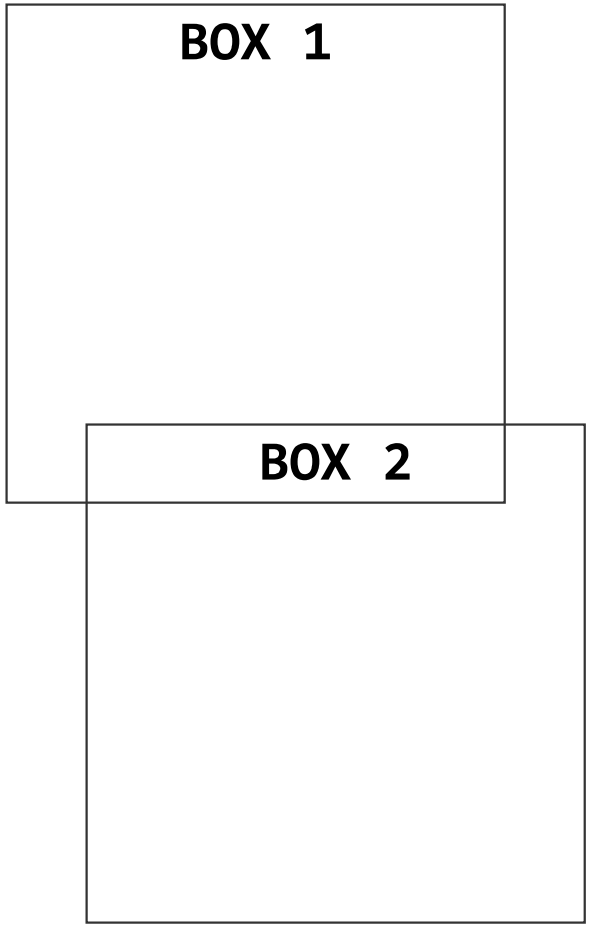
- none
- inline
- block
- inline-block
- list-item
- inline-list-item
- table
- inline-table
- table-cell
- table-column
- table-row
- table-caption
- flex
- inline-flex
- ...

MOST POPULAR `DISPLAY` PROPERTIES

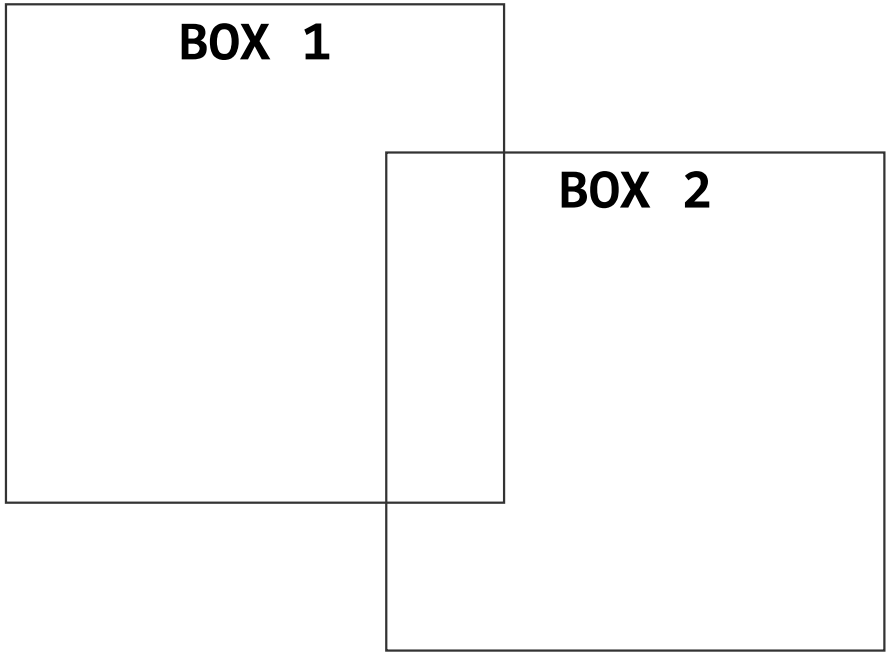
	Left & right margins	Top & bottom margins	Set height or width	Force line break after	**`vertical-align`**
Inline	✓	✗	✗	✗	✓
Block	✓	✓	✓	✓	✗
Inline-block	✓	✓	✓	✗	✓

A SIMPLE GUIDE TO CSS POSITIONING

HAVE YOU EVER WANTED TO DO SOMETHING LIKE...



...but ended up like



POSITON: STATIC

- Default **position** property.
- Element laid out in its current position in the flow.
- **z-index** property doesn't apply.

```
.box {
  width: 200px;
  height: 200px;
  background: #14FFF4;
}
```

POSITON: RELATIVE

- Adjusts element position without changing layout (leaves the space for the element where it should have been when not being positoned).
- **`z-index`** property applies.

```
.box-2 {
  position: relative;
  top: -40px;
  left: 40px;
}
```

POSITON: ABSOLUTE

- Does not leave the space for the element.
- Positions element at a specified position relative to its closest relative positioned ancestor or to the containing block.
- **z-index** property applies.

```
.box-2 {
  position: absolute;
  top: 0;
  left: -100px;
}
```

POSITON: FIXED

- Does not leave the space for the element.
- Positions element at a specified position relative to the viewport.
- **z-index** property applies.

```
.box-2 {
  position: fixed;
  top: 0;
  left: 10px;
}
```

SIZING UNITS

CSS SIZING UNITS

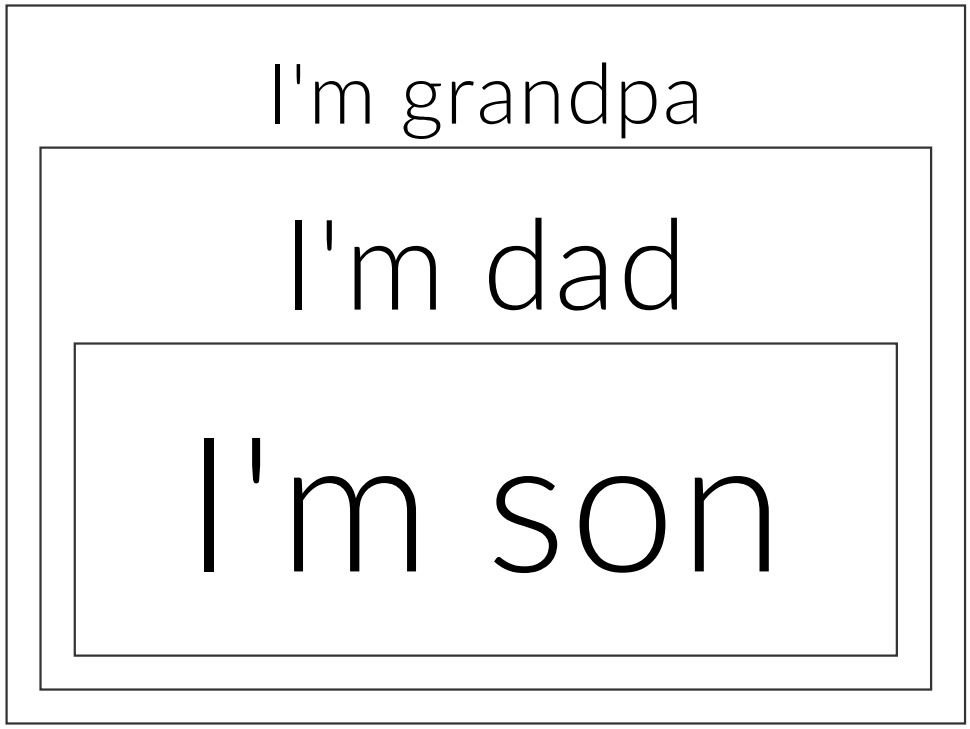
- px
- em
- %
- ex
- cm
- mm
- in
- pt
- pc

- ch
- rem
- vh
- vw
- vmin
- vmax

EMS AND COMPOUNDING

em is a relative unit based on parent **font-size**.

If **1em = 16px** then **1.5em = 24px**




```
.grandpa {
  font-size: 1.5em;
}

.dad {
  font-size: 1.5em;
}

.son {
  font-size: 1.5em;
}
```

REMS

> In the case of rem units, however, the font-size is dependent on the value of the root element (or the html element).
>
> -- CSS-Tricks Almanac (https://css-tricks.com/almanac/properties/f/font-size/)

SIZING FONTS USING REMS

```
<div class="outer">
        <h1>Heading</h1>
        <p>I'm an outer section content.</p>

        <div class="inner">
                <h1>Inner heading</h1>
                <p>I'm an inner section content.</p>
        </div>
</div>
```

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

.outer {
        font-size: 1.5rem; /* 1.5 * 16px  = 24px
*/
}

.inner {
        font-size: 2rem; /* 2 * 16px = 32px */
}
```

RESOURCES

- Font size with rem (http://snook.ca/archives/html_and_css/font-size-with-rem) by Jonathan Snook
- Font size on CSS-Tricks Almanac (https://css-tricks.com/almanac/properties/f/font-size/)
- Confused about rem and em? (https://j.eremy.net/confused-about-rem-and-em/) by Jeremy Church
- Just use pixels (https://benfrain.com/just-use-pixels/) by Ben Frain
- There's more to the CSS rem unit than font sizing (https://css-tricks.com/theres-more-to-the-css-rem-unit-than-font-sizing/) by Roman Rudenko

RESPONSIVE WEB DESIGN

MAKE YOUR WEBSITE LOOK GREAT ON ALL DEVICES

source: johnpolacek.github.io
(http://johnpolacek.github.io/scrolldeck.js/decks/responsive/)

VIEWPORT META TAG

A typical mobile-optimized site contains something like the following:

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

The **`width`** property controls the size of the viewport. It can be set to a specific number of pixels like **`width=600`** or to the special value **`device-width`** value which is the width of the screen in CSS pixels at a scale of 100%. (...)

The **`initial-scale`** property controls the zoom level when the page is first loaded.

-- MDN (https://developer.mozilla.org/en/docs/Mozilla/Mobile/Viewport_meta_tag)

MOBILE FIRST

WHY?

- Mobile Web usage is increasing.
- Overloading mobile devices with too much information (it's a pain for users wi bandwidth).
- Progressive enhancement (https://www.w3.org/wiki/Graceful_degradation_versus_progressive_enhance

HOW?

```
.mobile-first-component {
        display: none;
}

@media screen and (min-width: 992px) {
        .mobile-first-component {
                width: 50%;
        }
}

@media screen and (min-width: 1200px) {
        .mobile-first-component {
                width: 100%;
        }
}
```

QUESTIONS?