L3: CSS Basics

Web Engineering

188.951 2VU SS20

Jürgen Cito

L3: CSS Basics

- Introducing style and stylesheets
- CSS3 Selectors
- Box-model
- Units
- Custom Properties (“CSS Variables”)

Learning Goals

- Understand separation of markup and style
- Attach style information to elements and understand specificity
- Apply margins, borders, and paddings to elements (the CSS box-model)
- Understand absolute and relative units of positioning and sizing

CSS Overview

Cascading Style Sheets: describes style and layout of a document

Recommended by the W3C to separate content and design

- Initial problem when style and content were mixed
- Layout got removed gradually with every new standard
- Levels: CSS1 ⊆ CSS2 / CSS 2.1 ⊆ CSS3

Integration into HTML

- Inline – using the style attribute in elements
- Internal – using the <style> element in <head>
- External – linking to an external CSS file in <head>

External way preferred to separate structure/content and style/layout

CSS3

Fully backwards compatible to CSS2

Modules

- Selectors
- Box Model
- Background and Borders
- Image Values
- Text Effects
- 2D/3D Transformations
- Animations
- Multiple Column Layout
- User Interface
- …

"There will never be a CSS4!"
- Tab Atkins Jr, member of CSS Working Group

CSS Syntax and Selectors

Syntax

```
selector { property:value; }
```

declaration

Inline Syntax

```
<tagname style="property:value;">content</tagname>
```

Selectors

Type Selector

Select a group of elements via their name

```
<h1>...</h1>
h1 { font-size: 12pt; }
```

ID Selector

Select a single unique element via id (‘#’)

```
<p id="first">…</p>
#first { color: red; }
```

Class Selector

Select a group of elements via class (‘.’)

```
<p class="small">…</p>
<h1 class="small">…<h1/>
.small { font-size: 5pt; }
```


CSS Selector Reference: http://www.w3schools.com/cssref/css_selectors.asp

CSS Additional Selectors

- Additional Selectors
  - Descendants: Separate using white-space
  - Children: Separate using '>'
  - Siblings: Separate using '~'
  - Adjacent Siblings: Separate using '+'
  - Attribute: Specify attribute via '[att=val]'
- New CSS3 Selectors
  - :nth-child, :first-of-type, [*attribute**=*value*], …

```
body p { ... }
body > p { ... }
span ~ em { ... }
p + div { ... }
h1[title] { ... }
h1[title="a"] { ... }
```

```
p:nth-child(2) { ... }
p:only-child { ... }
```

Selectors can be grouped by separating them via comma ','

Specificity determines style when multiple selectors match

Specificity Calculator: https://specificity.keegan.st/

CSS Additional Selectors - More Examples

Selector	Meaning		Example
***Descendant* selector**	Matches all descendants of an element	p a { }	Select <a> elements inside <p> elements
***Child* selector**	Matches a direct child of an element	h1>a { }	Select <a> elements that are directly contained by <h1> elements.
***First child* selector**	Matches the first child of an element	h1:first-child { }	Select the the elements that are the first child of a <h1> element.
***Adjacent* selector**	Matches selector	h1+p { }	Selects the first <p> element after any <h1> element
***Negation* selector**	Selects all elements that are not selected.	body *:not(p)	Select all elements in the body that are not <p> elements.
***Attribute* selector**	Selects all elements that define a specific attribute.	input[invalid]	Select all <input> elements that have the invalid attribute.
Equality attribute selector	Select all elements with a specific attribute value	p[class="invisible"]	Select all <p> elements that have the invisible class.

Source: Thomas Latoza, SWE 432 GMU, Fall 2017, "HTML & CSS" Lecture

CSS Selector Specificity

Specificity: Which CSS rule applies to my element?

- Inline styles added to an element overwrite any external CSS
  (Do not use other than for experimentation, then remove/move to external stylesheet)
- Informally: Most specific rules wins
  Enables writing generic rules applying to many elements that are overriden by specific rules
- CSS infers a specificity score
  - Selector with most #id selectors wins
  - If count(#id) is the same, the selector with the highest number of the following wins:
    - .classes, :pseudo-classes, [attributes]
  - If these are tied, selector with highest number of elements (tags) wins
  - If still tied, source order defines score

Specificity Calculator: https://specificity.keegan.st/

CSS Properties
Standard Properties

Formatting Text/Fonts

- Font family, style, size, and weight
  - Use font fallback
- Color
- Line Height
- Text Alignment

AaBb

```
font-family: Arial, sans-serif;
font-style: italic;
font-size: 1.2em;
font-weight: bold;
color: #00ff00;
line-height: 120%;
text-alignment: center;
```

Background

- Color
- Image, Repeat, Attachment, Position

```
background-color: rgb(250,20,16);
background-image: url("bg.jpg");
background-repeat: repeat-x;
background-position: right top;
```

Lists

- Item marker or Image

```
list-style-type: circle;
list-style-image: url('logo.gif');
```

Borders

- Rounded corners
- Shadow

```
border: 2px solid #A1A1A1;
border-radius: 25px;
```

```
border: 1px solid black;
box-shadow: 3px 3px 3px #FF9900;
```

CSS Properties
Sizes and Proportions

Absolute values

- Anchored in phyiscal unit or pixel unit
- For fixed sized rendering (printed pages, images)
- Inches (in), Centimeters (cm), Millimeters (mm), Points (pt), Picas (pc)
- Pixel (px): Relative to screen resolution, but absolute for output device

Relative values

- Anchored in parent size, font size or viewport size
- For screen rendering and easy accessible content
- Parent Size: % (relative to parent)
- Font Size: em (relative to font square), ex (relative to 'x'-height), ch (relative to '0' glyph), rem (relative to root element font-size)
- Viewport Size: vw (relative to width of initial containing block), vh (relative to height of initial containing block), vmin (min of vh and vw), vmax (max of vw and vh)

Read more on em and em here:
https://j.eremy.net/confused-about-rem-and-em/

Calculated values

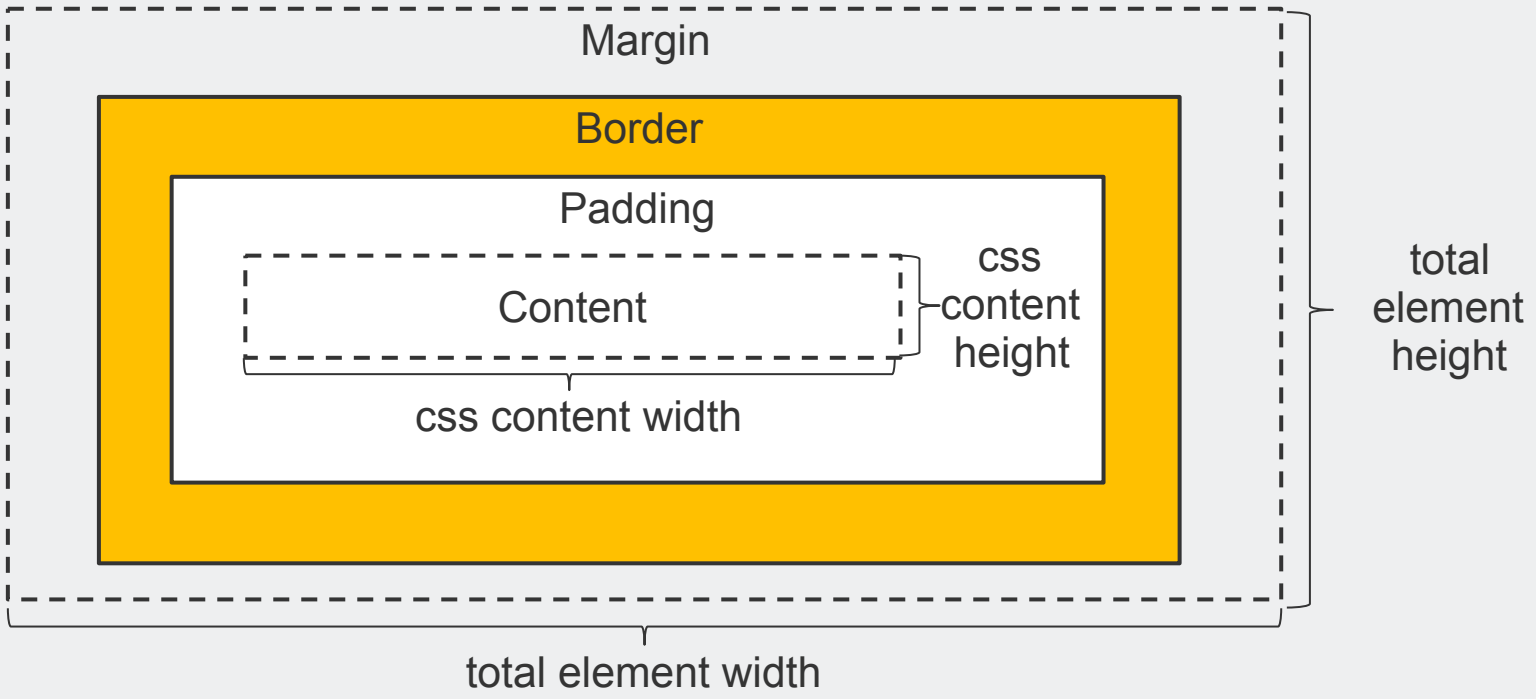
- calc: Calculate attribute value by adding or subtracting sizes. Space required!

```
width: calc(2.5em - 3px)
```


CSS Values and Units: https://www.w3.org/TR/css3-values/
Font Size tips: https://www.w3.org/QA/Tips/font-size

CSS Properties
Box Model

- Content width and height
- Margin, Padding, and Border can be set for left, right, top, bottom
- Total element width calculated as css content width + padding + border + margin



CSS Properties
Positioning

Positioning of Elements

```
position: absolute;
left: 10px;
top: 10px;
```

- Standard page flow is static
  - horizontal, one element after another (inline vs block)
- Coordinates can position elements differently (top, bottom, left, right)
  - Fixed: Element removed from flow
  - Relative: Position relative to position in flow (original space still taken)
  - Absolute: Position relative to first non-static parent or html
  - z-index defines which elements should be placed in front

Floating of Elements

```
float: left;
float: right;
clear: left;
clear: both;
```

- Push element left or right
- Following elements float around
- Use clear to turn floating off

CSS Properties
Pseudo Classes

Pseudo-classes / Pseudo-elements

- Use information present outside the document tree
- Pseudo-classes (excerpt)
  - Based on user input
    - :hover, :focus – Element the user hovers the mouse over / has selected via tabbing
    - :visited, :link – All visited/unvisited links
  - Based on form status
    - :enabled, :disabled – Whether user may input something in an element or not
    - :required, :optional – Whether inputs are required or not
    - :valid, :invalid – Whether a form or an input has erroneous input or not
  - Based on DOM position
    - :nth-child(n), :nth-last-child – If this element is the n-th child of it's parent
    - :nth-of-type(n) – If this element is the n-th child of the same type of it's parent
- Pseudo-elements
  - ::first-line, ::first-letter - First letter or line
  - ::selection – Current selection
  - ::before, ::after – Content inserted before or after the specified element

```
a:hover { color: #ff0099; }
```

```
p::first-letter {
  font-size: 20px;
}
```

```
p::before {
 content: 'Nav';
 display: block;
}
```

CSS Custom Properties “Variables”

Introduces “variables” named custom properties

- Enables reuse of values by introducing a common name
- Custom Property Syntax `--name: value`
  - n*ame* is case-sensitive
  - value can be any valid CSS value
- Has to defined in a certain scope (selector)
  - Selector that property is defined in determines scope of usage
  - :root Pseudo class common best practice to introduce global properties
- Access variable with `var(--name)`
  - Fallback values (if - -name does not exist) with comma `var(--name, black)`
- Encapsulation: Other people can use and style your components without knowing your internal CSS structure

```
:root {
  --primary-color: #ff0099;
  --alarm-border: 1px dashed red;
}

p.start {
  color: var(–primary-color)
  border: var(--alarm-border)
}
```

```
h2 {
  –alarm-color: darkred;
}

:root {
  --alarm-color: crimson;
}

h2.alarm, div.alarm {
  color: var(–alarm-color, red)
}
```