Lecture date: 6 March 2017

**Cascading Style Sheet**

* Specifies the presentation aspects of structurally marked up documents
* Developed by Hakon Wium Lie (CHSS) and Bert Bos (SSP)
* Versions:
  + CSS 1 (1996) - monolitic
  + CSS 2.1 (June 2011) – monolithic
  + CSS 3 – modules
    - To allow browser vendors implement what they want to implement and features can become recommendations when they are already mature enough
    - No centralized repository to view available features
    - No “level 3” means it didn’t existed in v2.1
* CSS Preprocessors and Frameworks
  + Use advance syntax, the convert it to flat hierarchy when compiled
  + Syntactically Awesome Style Sheet (SASS), LESS, 960 Grid System, Bootstrap, Foundation, Materialize, etc.
  + **@mixin** 
    - mixing a block of properties in another block of properties
    - Used with **@include**

**Syntax:**

@mixin pm ($v) { margin: $v; }

p{ @include pm(10px) <!— other lines of codes here --> }

@function twice($v) { @return 2\* $v; }

p{ padding: twice(10px) <!—other lines of codes here --> }

@for $i from 1 through 6 {

h# { $i } {

interpolation

font-size: 22px-2\*$i;

}

}

* **sass-lang.com**

CSS Frameworks

Bootstrap – **getbootstrap.com**

**HTML/XHTML Stylesheets**

* **Author styles** – CSS created by the author

1. External stylesheets (recommended)
   * Easier management

* Faster (caching)
* Multiple html
* Minimize bandwidth

**Attributes:**

* media (screen, print, speech, tv, projection)
* rel (alternate stylesheet, stylesheet)
* type
* href
* title

1. embedded styles – uses style element, for quick prototyping
2. inline styles – uses style attribute

* **User styles** – used by the user
* **User agent styles** – default stylesheet of browsers (ex. Default CSS 2.1 stylesheet for HTML 4)

**CSS Statements**

* **At rules**
* **@charset** – specify character set
* **@import** – import stylesheet
* **@media** – media query
* **@font-face** – key frame animation
* **@page**
* **@document** – stylesheet for specific page/domain
* **@namespace** – svg
* **CSS Rule sets (a.k.a. CSS rules, style rules)**
* Consist of a selector (body, : )

**CSS Selector**

* **Selector** 
  + select elements in HTML
  + Elements – subjects of the selector
* Targets element for formatting
* **Selector syntax**
* Sequences of simple selectors separated by combinators, with one pseudo element possibly appended to the last sequence

Ex. div #abc > p.xyz [title] + span:first-child::after

* type selector, id selector, combinators, class attribute, pseudo class, pseudo element
* group of selectors – comma-separated selectors
* **simple selectors**
* **type** – name of html element
* **universal** - \* (asterisk - everything)
* **attribute** – [attr = / v= / l= / ^= / $= / \*= value]
* single, language attribute, CSS3
* **class** – period (.)
* **id** - #
* **pseudo-class**
  + dynamic pseudo-class
    - link pseudo-class
      * :link
        + :visited
* User action pseudo-class
  + hover
  + active
  + focus
* target pseudo-class
* :target (CSS3)
* language pseudo-class
* :lang()
* Ex. lang(en){ <p lang=”en-US”> }

Lecture date: 9 March 2017

* UI element state pseudo-classes (interactive elements)
  + - * + :enabled (CSS3)
        + :disabled (CSS3)
        + :checked (CSS3)
        + :indeterminate (CSS3)
* Structural pseudo-classes
  + :root – root element
  + :first-child – not CSS3
  + :last-child
  + :only-child
  + :nth-child – 2n+1
  + :nth-last-child()
  + :first-of-type
  + :last-of-type
  + :only-of:type
  + :nth-of-type()
  + :nth-last-of-type()
  + :empty – no child
* negation pseudo-classes
* :not() (CSS3)
* **Combinators**
* descendant combinators (whitespace)
* child combinators ( > )
* sibling combinators
  + - * adjacent sibling combinatory ( + )
        + immediately followed (ex. h1+p in <h1><p><p>)
* general sibling combinatory ( ~) – CSS3
* anything that follows (ex. h1+p in <h1><p><p>)
* **Pseudo-elements**
  + - ::first-letter (CSS3), :first-letter
      * Example: <h1> Hello </h1>
    - ::first-line (CSS3), :first-line
    - ::before (CSS3), :before
    - ::after (CSS3), :after

**CSS Rule Precedence**

Ways to refer and format an HTML element:

<div><p id=x class=y> … <em> … </p></div>

* + div p { color: red; }
  + div > p
  + #x
  + :y
  + \*
  + P
  + p#x
  + p.y
  + [id=x]
  + [class=y]
  + div p#x
* **by origin and importance**
  + user agent important declarations
  + user important declarations
  + author important declarations
  + author normal declarations
  + user normal declarations
  + user agent normal declarations
* **by specificity**
  + inline style
  + number of ID selectors
  + number of class selectors, attribute selectors, and pseudo-classes
  + number of type selectors and pseudo-elements
* **by order**
* last declaration wins

Lecture date: 13 March 2017

\*Cascade effect – more than one style rule applied on one element

Id selector = #

Class selector = .

Attribute selector = []

Pseudo-class = :

**CSS Declarations**

* **properties**
  + **shorthand properties** – single property representing many other
  + **vendor-specific extensions** (a.k.a. vendor prefixes)
  + **initial** – explicitly set the initial value of a property
  + **inherit** – explicitly setting property value inherited
  + **unset**
  + **revert**
* **values** (values and units)
  + **keywords**
  + **numbers**
  + **dimensions**
    - length, angle, duration, frequency, resolution
    - length units
      * **font-relative**: em, ex, ch, rem
      * **viewport-percentage**: vw, vh, vmin, vmax
      * **absolute lengths**: cm, mm, q, in, pt, pc, px
* angle units: deg, grad, rad, turn
* durations units: s, ms
* frequency units: hz, khz
* resolution units: dbi, dpcm, dppx
  + **percentages**
  + **URLs and URIs**
  + **Colors**
  + **Strings**
  + **Functions:** calc(), attr(), counter(), counters(), linear-gradient(), radial-gradient(), translate(), scale(), rotate(), etc.s