Web Engineering

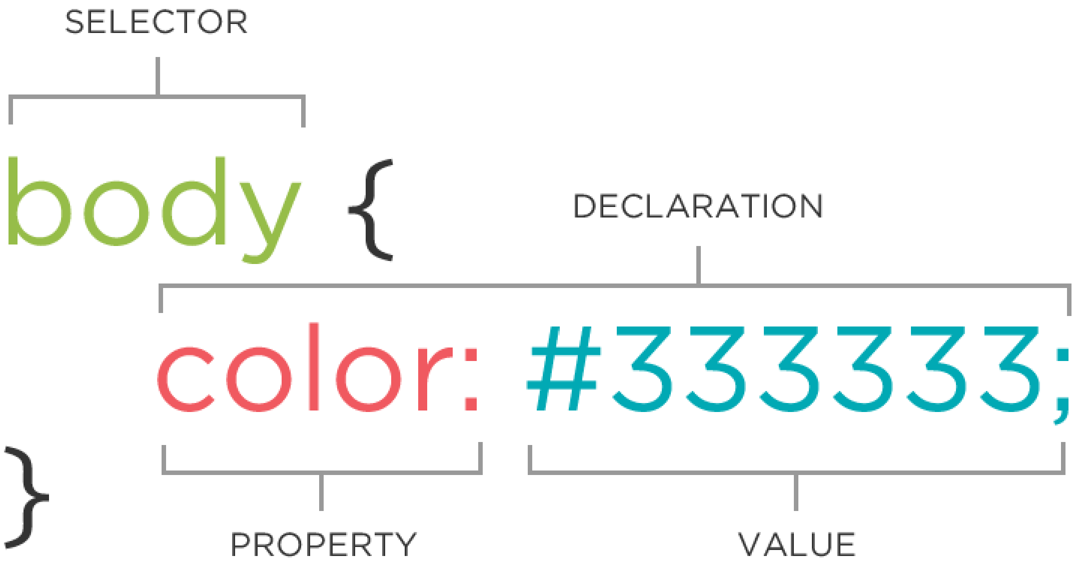
CSS

Prof. Dierk König
Christian Ribeaud

CSS

- **C**ascading **S**tyle **S**heet
- Contains the rules for the *presentation* of **HTML**
- **CSS** + **HTML** = **Web Page**
- One **HTML** Page - Multiple Styles!

CSS Rule



SELECTOR

DECLARATION

```
body {
    color: #333333;
}
```

PROPERTY

VALUE

CSS Selectors

element

CSS

```css
div {
  text-align: center;
  color: red;
}
```

HTML excerpt

```html
<div>Hello</div>
<div>Again</div>
<p>
  <div>Enjoying the day...</div>
</p>
```

CSS Selectors

element

id

CSS

```css
#flower {
  text-align: center;
  color: red;
}
```

HTML excerpt

```html
<div id="flower">rose</div>
<div id="language">JavaScript</div>
```

CSS Selectors

element

id

class

CSS

```css
.center {
  text-align: center;
  color: red;
}
```

HTML excerpt

```html
<div class="center">center</div>
<div class="left">left</div>
<footer class="center">
&copy; Karakun AG
</footer>
```

Special Selectors

attribute

CSS

```css
[title] {
  text-align: center;
  color: red;
}
```

HTML excerpt

```html
<div title="My Div">One</div>
<div title>Two</div>
<div class="left">left</div>
```

Special Selectors

attribute

attribute/value

CSS

```
[title="x"] {
  text-align: center;
  color: red;
}
```

HTML excerpt

```
<div title="x">One</div>
<div title>Two</div>
<div class="left">left</div>
```

Special Selectors

attribute

attribute/value

pseudo-class

CSS

```css
div:hover {
  color: red;
}
```

HTML excerpt

```html
<div title="x">One</div>
<div title>Two</div>
<div>left</div>
```

Combinators

next sibling

CSS

```
p + em {
  color: red;
}
```

HTML excerpt

```
<p><em>Some text</em></p>
<em>Another text</em><br />
<em>Is this red?</em>
```

Combinators

next sibling

all siblings

CSS

```css
p ~ em {
  color: red;
}
```

HTML excerpt

```html
<p><em>Some text</em></p>
<em>Another text</em><br />
<em>Is this red?</em>
```

Combinators

next sibling

all siblings

descendant

CSS

```css
p em {
  color: red;
}
```

HTML excerpt

```html
<p>
  <em>Some text</em><br />
  <span>
    <em>Some text</em>
  </span>
</p>
<em>Another text</em><br />
<em>Is this red?</em>
```

Combinators

next sibling

all siblings

descendant

direct child

CSS

```css
p > em {
  color: red;
}
```

HTML excerpt

```html
<p>
  <em>Some text</em><br />
  <span>
    <em>Some text</em>
  </span>
</p>
<em>Another text</em><br />
<em>Is this red?</em>
```

Combinators

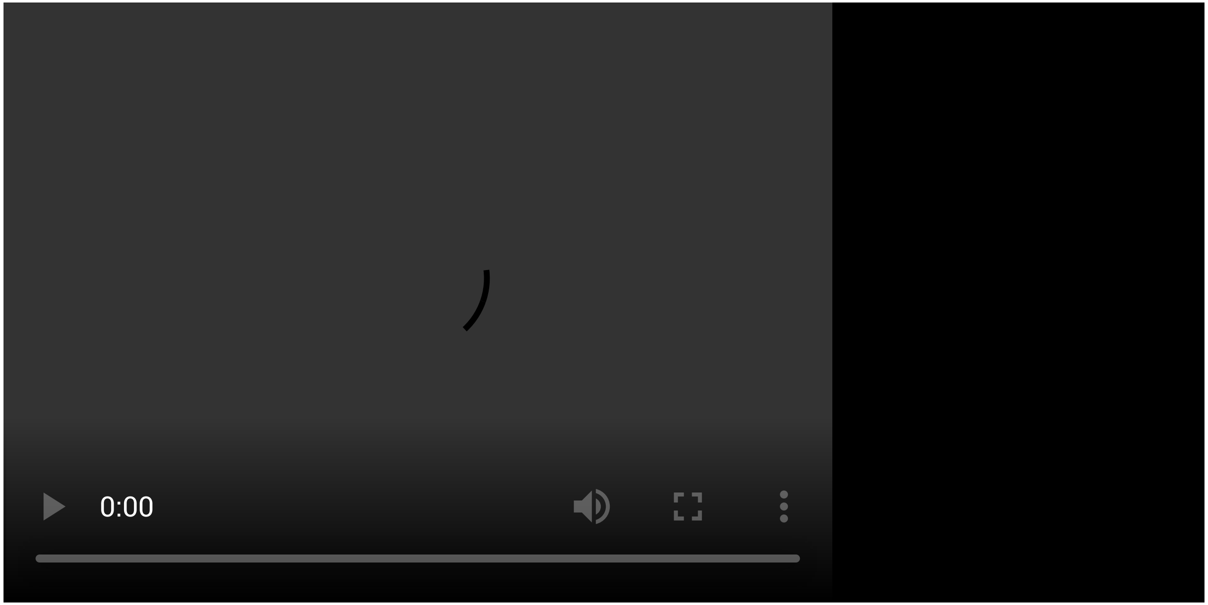
next sibling

all siblings

descendant

direct child

recap



More ...

union

CSS

```
span, h3 {
  color: red;
}
```

HTML excerpt

```
<p>This text is <span>red</span></p>
<h2>That one is NOT red</h2>
<h3>This header is red as well</h3>
```

More ...

union

element with class

CSS

```css
h2.red {
  color: red;
}
```

HTML excerpt

```html
<h2 class="red">This header is red</h2>
<h2>This header is black (default)</h2>
```

More ...

union

element with class

element with attribute

CSS

```css
h2[red] {
  color: red;
}
```

HTML excerpt

```html
<h2 red="true">This header is red</h2>
<h2>This header is black (default)</h2>
```

Adding CSS

inline

- An *inline* **CSS** is used to apply an unique style to a single **HTML** element.
- An *inline* **CSS** uses the style attribute of a **HTML** element.

```
<p style="font-size:20px">Lorem ipsum dolor sit
amet, consectetur adipiscing elit.</p>
```

Adding CSS

inline

internal

- An *internal* **CSS** is used to define a style for a single **HTML** page.
- An *internal* **CSS** is defined in the `<head>` section of an **HTML** page, within a `<style>` element

```
<head>
<style>
body {background-color: powderblue;}
h1   {color: blue;}
p    {color: red;}
</style>
</head>
```

Adding CSS

inline

internal

external

- An *external* style sheet is used to define the style for many **HTML** pages.
- With an *external* style sheet, you can change the look of an entire web site, by changing one file!

```
<head>
<link rel="stylesheet" href="styles.css">
</head>
```

Precendence

Sheet importance

Incrementally:

1. *Inline* overrides **CSS** rules in `<style>` tag and **CSS** file.
2. A *more* specific selector takes precedence over a *less* specific one.
3. Rules that appear later in the code override earlier rules if both have the same specificity.
4. A **CSS** rule with `!important` always takes precedence.

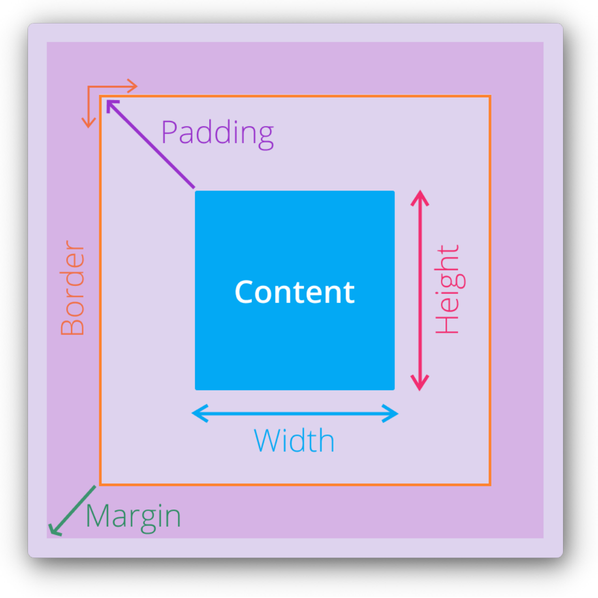
Precendence

Sheet importance

Specificity

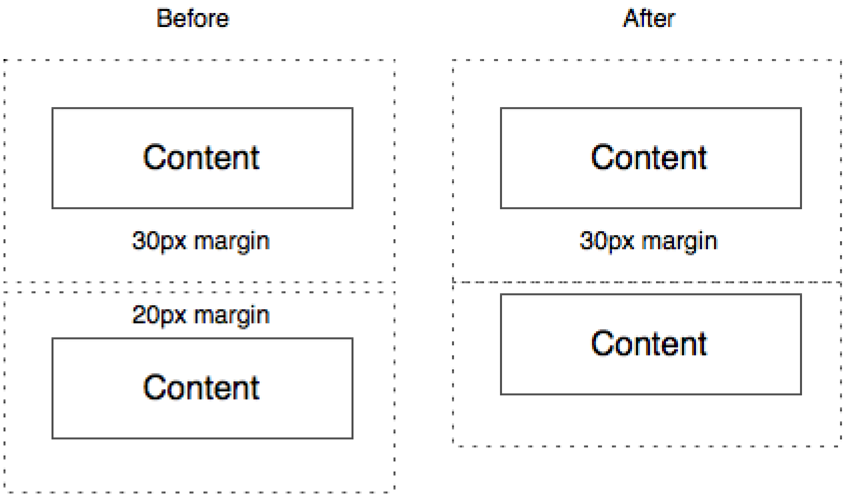
1. *User agent* declarations
2. *User* normal declarations
3. *Author* normal declarations
4. *Author* important declarations
5. *User* important declarations

Box Model



Total width = Margin left + Border left + Padding left + Content Width + Padding right + Border right + Margin right

Margins Collapse



Units

- A whitespace cannot appear between the number and the unit. However, if the value is 0, the unit can be omitted.
- For some **CSS** properties, negative lengths are allowed.

- We have following units:

  - *Absolute* lengths: `cm`, `mm`, `in`, `pt` (*1pt* = 1/72 of *1in*), `px` (*1px* = 1/96th of *1in*)*
  - *Relative* lengths: `em` (size of **M**, local font), `rem` (*root* `em`), `ex` (x-height of current font), `%` (relative to parent), `vw` (relative to 1% of the viewport width), `vh` (relative to 1% of the viewport height)

* `px` are relative to the viewing device. A CSS pixel is different from a device one.

Functions

- Formula: `calc(100% / 12)`
- Attribute value: `attr(name)`
- External reference: `url(“backg.jpg”)`
- Variable: `var(--left-indent)`

Further topics

- Display: `inline`, `block`, `none`, …
- Position: `static`, `relative`, `absolute`, `fixed`, …
- float, clear: `left`, `right`, …
- CSS Preprocessors: Sass, Less, …

Exercises

Assignment 1

1. Change storybook.html document.
2. Include the storybook.txt into a **HTML** table.
3. Put a `<style>` element in the header.
4. Improve some element styles (e.g `h1`, `h2`, `section`, `article`) to make them look nicer.
5. Notice usage of `abbr` and `dfn` elements. What do they stand for? And how to use them?

Exercises

Assignment 1

Assignment 2

1. In the table introduce style classes to make the holidays less prominent.
2. In the table introduce style classes to make the exercise-related dates more prominent.

Exercises

Assignment 1

Assignment 2

Challenge

1. Make sure that your result from *Assignment 2* is visible through the GitHub Pages feature
2. Email a link to the **HTML** page to christian.ribeaud@fhnw.ch.
3. **The best result wins a price!**

For aspiring professionals

Training 1

- If you are not yet confident with using basic **HTML** and **CSS** consider working through http://www.codecademy.com, https://scrimba.com/ or https://codepip.com/.

For aspiring professionals

- Have a look at modern.html page.
- Can you do it with **CSS** flex layout?
- Can you do it with **CSS** grid layout?

Training 1

Training 2

Abilities

- Styling of **HTML** pages with respect to appearance and layout
- Being able to use **CSS** rules creatively
- Avoiding duplication in style information
- Separating *what* from *how*
- Writing maintainable web documents

Knowledge

- Basic selectors: *element*, *class*, *id*
- Selector combinations: collection, descendant, child
- Sourcing: inline, style element, link element
- Basic knowledge of cascading and specificity
- CSS box model
- Basic knowledge of CSS units