Cascading Style Sheet

Basic Concept of CSS

Arie Satia Dharma, S.T., M.Kom
Dosen Informatika

Learning Outcome

- Mahasiswa mampu memahami konsep CSS yang terdiri atas style, selektor, dan properties.
- Mahasiswa mampu menerapkan konsep CSS tersebut untuk menghasilkan tampilan antarmuka halaman web.

Agenda

- Cascading Style Sheet
- CSS Selectors
- CSS Properties

Cascading Style Sheet

What it CSS?

* CSS stands for Cascading Style Sheet.
  * A style sheet is made up of style rules that tell a browser how to present a document. Cascading means written downwards.
* CSS enables the separation of HTML document from document presentation
  * such as layout, colors, fonts, text, background, borders, and floats.
* A standard tool in web design and development.

Why Using CSS? [1]

* **CSS saves time**

  You can write CSS once and then reuse same sheet in multiple HTML pages.

* **Pages load faster**

  Write one CSS rule of a tag and apply it to all the occurrences of that tag, so less code means faster download times.

* **Easy maintenance**

  To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.

* **Superior styles to HTML**

  CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.

Why Using CSS? [2]

* **Multiple Device Compatibility**

  By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.

* **Global web standards**

  Now HTML attributes are being deprecated and it is being recommended to use CSS.

* **Offline Browsing**

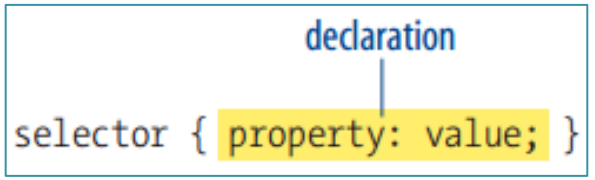
  CSS can store web applications locally with the help of an offline catche.Using of this, we can view offline websites.

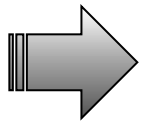
* **Platform Independence**

  The Script offer consistent platform independence and can support latest browsers as well.

Style Rules

* A style rule is made of two parts :
  * **Selector** – that identifies the element or elements to be affected
  * **Declaration** – that provides the rendering instructions. The declaration is made up of a **property** (such as color) and its **value** (green), separated by a colon and a space. One or more declarations are placed inside curly brackets





```
h1{color: green;}
```

```
h1{color: green; text-align: center;}
```

```
h1{
  color: green;
  text-align: center;
}
```

Types of Style Sheet [1]

* Inline Style Sheets
  * An inline style may be used to apply a unique style for a single element.
  * To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.
  * The example below shows how to change the color and the left margin of a <h1> element:
  * Example:

```
<h1 style="color:blue;margin-left:30px;">This is a heading.</h1>
```

Types of Style Sheet [2]

* Embedded/Internal Style Sheets
  * An internal style sheet may be used if one single page has a unique style.
  * Internal styles are defined within the <style> element, inside the <head> section of an HTML page.
  * Example:

```
<head>
<style>
body {
    background-color: linen;
}

h1 {
    color: maroon;
    margin-left: 40px;
}
</style>
</head>
```

Types of Style Sheet [3]

* External Style Sheets
  * With an external style sheet, you can change the look of an entire website by changing just one file!
  * Each page must include a reference to the external style sheet file inside the <link> element. The <link> element goes inside the <head> section.

```
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css">
</head>
```

  * An external style sheet can be written in any text editor. The file should not contain any html tags. The style sheet file must be saved with a .css extension. Here is how the "myStyle.css" looks:

```
h1 {
    color: navy;
    margin-left: 20px;
}
```

Types of Style Sheet [4]

* Imported Style Sheets
  * @import is used to import an external stylesheet in a manner similar to the <link> element.
  * Following is the example showing you how to import a style sheet file into HTML document :

```
<head>
   @import "mystyle.css";
</head>
```

Conflict of Styles [1]

* CSS allows you to apply several style sheets to the same document.
  * The problem is Potent to be conflict
  * To anticipated this problem then **the folks** assign **a hierarchical system** to the various sources of style.
  * The cascade refers to control of the elements on a page: style information is passed down (“cascades” down) until it is overridden.

```
p { color: red; }
p { color: blue; }
p { color: green; }
```

Conflict of Styles [2]

Style Sheet Hierarchy

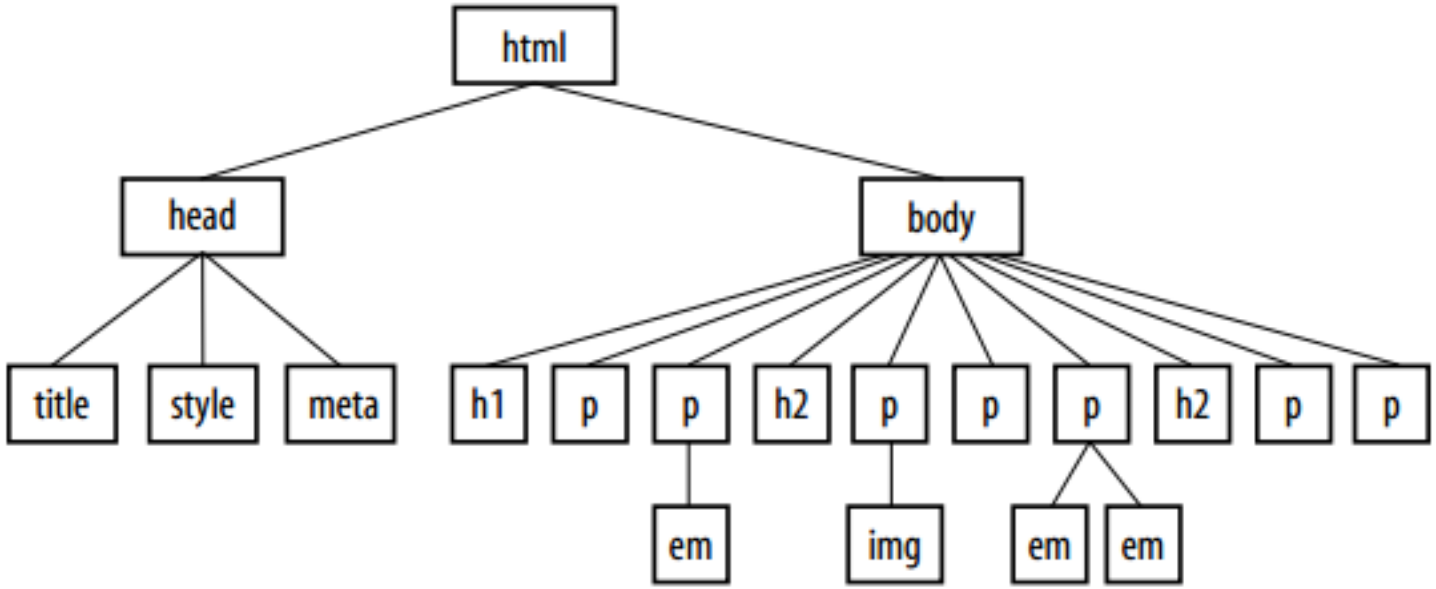
Style information can come from various sources, listed here from general to specific. Items lower in the list will override items above them:

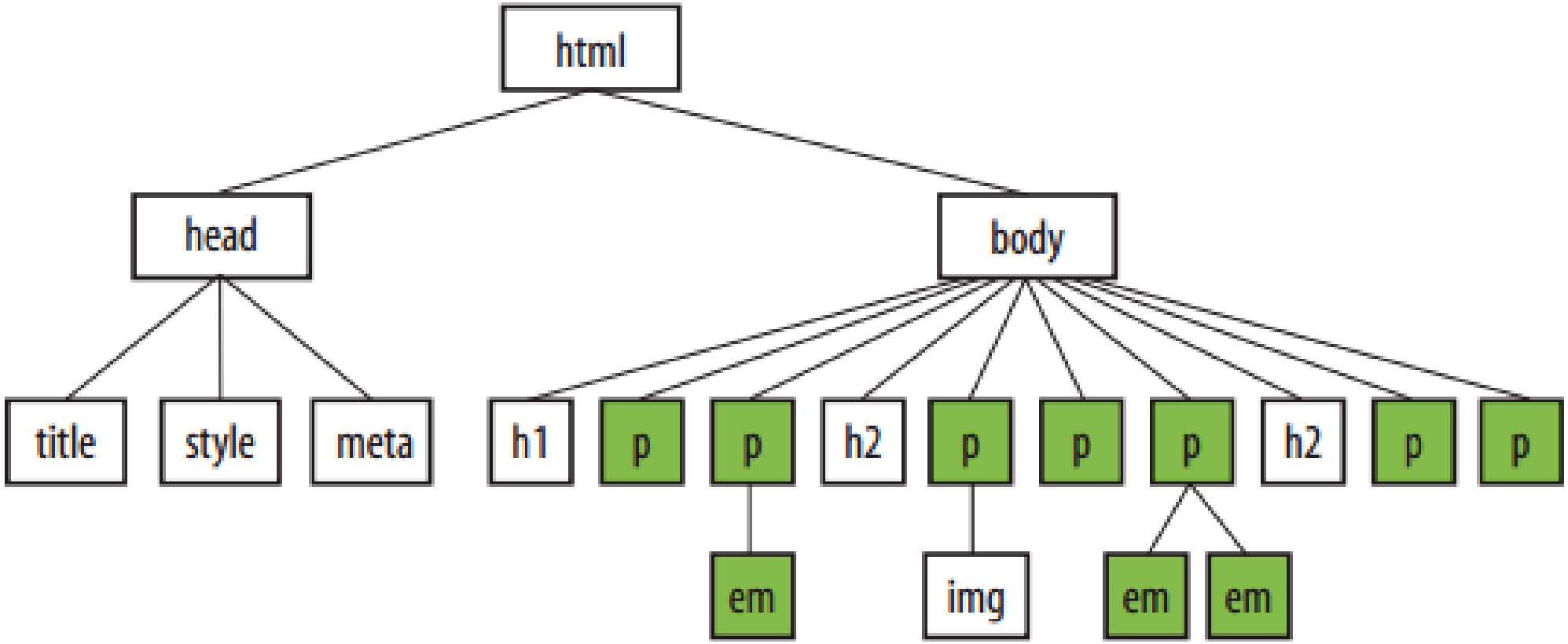
- Browser default settings
- User style settings (set in a browser as a "reader style sheet")
- Linked external style sheet (added with the **`link`** element)
- Imported style sheets (added with the **`@import`** function)
- Embedded style sheets (added with the **`style`** element)
- Inline style information (added with the **`style`** attribute in an opening tag)
- Any style rule marked **`!important`** by the author
- Any style rule marked **`!important`** by the reader (user)

Inheritance

The document tree becomes a family tree when it comes to referring to the relationship between elements.

* Ancestors
* Descendants
* Parent
* Child
* siblings






```
p {font-size: small; font-family: sans-serif;}
```

CSS Selectors

Universal Selector

* The universal selector is an asterisk (*)
  * It's matches all element types in the document.
* The style rule:

```
* { color: grey; }
```

* The universal selector causes problems with form controls in some browsers, so try to avoid the universal selector.

The Element Type Selector

* Also referred to simply as a “type selector,” this selector must match one or more HTML elements of the same name.
* It allows you to apply the same rules to several elements.
* Example :

```
h1, h2, h3 {color: grey;}
```

The Class Selector

* Allows you to match a rule with an element (or elements) carrying a class attribute.

```
<div class="box"></div>
```

* It's declared with a dot preceding a string of one or more characters.

```
.box {width: 240px;Height: 100px}
```

* HTML allows multiple elements on a single page with the same class attribute.
* HTML also allows multiple classes to be added to a single element.

```
<div class="box box-more box-extended"></div>
```

The ID Selector

* The id selector works just like a class selector, but works on the value of id attributes.

```
<div id="container"></div>
```

* The symbol that identifies ID selectors is the octothorpe (#), also known as a hash symbol.

```
#container {
    width: 960px;
    margin: 0 auto;
}
```

* Because the value of an id attribute should be unique within a document, this selector should apply only to the content of one element

The Contextual Selector

Called combinators or a contextual selector because it selects the element based on its context or relation to another element.

* The Child Selector (>)
  * The child selector matches an element that is a direct child of another.

```
p > em {font-weight: bold;}
```

* The Descendant Selector (space)
  * A descendant selector targets elements that are contained within (and therefore are descendants of) another element.

```
li em { color: olive; }
```

The Contextual Selector

Called combinators or a contextual selector because it selects the element based on its context or relation to another element.

* The Adjacent Sibling Selector (+)
  * An adjacent sibling selector targets an element that comes directly after another element with the same parent.
* The General Sibling Selector (~)
  * A general sibling selector selects an element that shares a parent with the specified element and occurs after it in the source order. They do not need to follow one another directly.
* Example:

```
h1 + p {font-style: italic;}
```

```
h1 ~ h2 {font-weight: normal;}
```

Descendant combinator E F

```html
<ul>
  <li>List Item 1</li>
  <li>List Item 2
    <ol>
      <li>List Item 2-1</li>
      <li>List Item 2-2</li>
    </ol>
  </li>
  <li>List Item 3</li>
</ul>
```

```css
ul li {background: red;}
```

Child combinator E > F

```html
<ul>
  <li>List Item 1</li>
  <li>List Item 2
    <ol>
      <li>List Item 2-1</li>
      <li>List Item 2-2</li>
    </ol>
  </li>
  <li>List Item 3</li>
</ul>
```

```css
ul>li {background: red;}
```

Adjacent sibling combinator E + F

```html
<h1>Heading</h1>
<p>Paragraph 1</p>
<p>Paragraph 2</p>
<p>Paragraph 3</p>
```

```css
h1+p {font-size: 1.5em;}
```

General sibling combinator E ~ F

```html
<h1>Heading</h1>
<p>Paragraph 1</p>
<p>Paragraph 2</p>
<p>Paragraph 3</p>
```

```css
h1~p {font-size: 1.5em;}
```

The Attribute Selector

* The attribute selector targets elements based on the presence and/or value of HTML attributes, and is declared using square brackets:
* Attribute selectors enable you to use the attributes that an element carries, and their values, in the selector.
* Example:

```
<input type="text">
```

```
input[type="text"] {background-color: #444;width: 200px;}
```

Name	Example	Matches
Existence selector	`p[id]`	Any <p> element carrying an attribute called `id`.
Equality selector	`p[id="summary"]`	Any <p> element carrying an attribute called `id` whose value is `summary`.
Space selector	`p[class~="XHTML"]`	Any <p> element carrying an attribute called `class`, whose value is a list of space-separated words, one of which is exactly the same as `XHTML`.
Hyphen selector	`p[language|="en"]`	Any <p> element carrying an attribute called `language` whose value begins with `en` and is followed with a hyphen (this particular selector is designed for use with language attributes).
Prefix selector (CSS3)	`p[attr^"b"]`	Any <p> element carrying any attribute whose value begins with `b`. (CSS3)
Substring selector (CSS3)	`p[attr*"on"]`	Any <p> element carrying any attribute whose value contains the letters `on`. (CSS3)
Suffix selector (CSS3)	`p[attr$"x"]`	Any <p> element carrying any attribute whose value contains the ends in the letter `x`. (CSS3)

Media Queries

* CSS introduced the media keyword:
  * Allowing you to define a printer-friendly style sheet.
  * You can then define one style sheet for browsers (screen) and a different style sheet for the print version of your web page.
* Example:

```
<link rel="stylesheet" href="Initial.css" />
<link rel="stylesheet" href="Print.css" media="print" />
```

```
@media print
{
    h1, h2, h3
    {
        font-size: 14px;
    }
}
```

Media Attributes

* This media keyword has been enhanced significantly to allow you to query various attributes to determine the appropriate styles.
* Example:

```
@media (max-width:600px)
{
    h1
    {
        font-size: 12px;
    }
}
```

CSS Properties

FONT	BACKGROUND	border-right-color
font	background	border-right-style
font-family	background-attachment	border-right-width
font-size	background-color	border-top
font-size-adjust	background-image	border-top-color
font-stretch	background-position	border-top-style
font-style	background-repeat	border-top-width
font-variant	BORDER	MARGIN
font-weight	border	margin
TEXT	border-color	margin-bottom
color	border-style	margin-left
direction	border-width	margin-right
letter-spacing	border-bottom	margin-top
text-align	border-bottom-color	PADDING
text-decoration	border-bottom-style	padding
text-indent	border-bottom-width	padding-bottom
text-shadow	border-left	padding-left
text-transform	border-left-color	padding-right
unicode-bidi	border-left-style	padding-top
white-space	border-left-width	
word-spacing	border-right	

DIMENSIONS	z-index	list-style-type
height	**OUTLINES**	marker-offset
line-height	outline	**GENERATED CONTENT**
max-height	outline-color	content
max-width	outline-style	counter-increment
min-height	outline-width	counter-reset
min-width	**TABLE**	quotes
width	border-collapse	**CLASSIFICATION**
POSITIONING	border-spacing	clear
bottom	caption-side	cursor
clip	empty-cells	display
left	table-layout	float
overflow	**LIST and MARKER**	position
right	list-style	visibility
top	list-style-image	
vertical-align	list-style-position	

Thank You

Reference

- Robbins J., Niederst. 2012. Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics. O'Reilly Media.
- Jackson C., Jeffrey. 2007. Web Technologies: A Computer Science Perspective. Prentice Hall.
- https://www.w3schools.com/css/default.asp