



Web Design and Programming

Lecture 7:

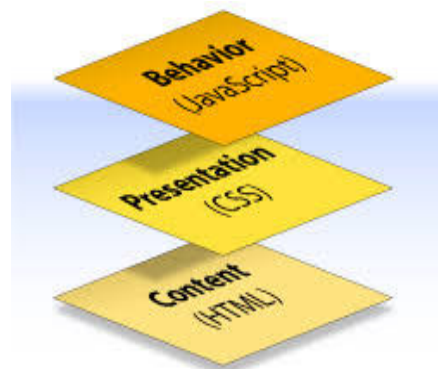
Cascading Style Sheets

Instructor: Ali A. Al-Ani



Web Document

- When we examine the elements of the web document construction, it can consist of up to three layers *content*, *presentation*, and *behavior* as illustrated below.
- **Note:** It's possible to embed all three layers within the same document, but keeping them separate gives us one valuable advantage: we can modify or replace any of the layers without having to change the others.





Cascading Style Sheets

- Certain versions of XHTML also contain presentational element types that specify the appearance of the content. For example, `` and `<i>` can be used to control the presentation of text.
- However, as these types of elements embed presentation layer information within the content layer, they negate any advantage we may have gained by keeping the layers separate.
- *CSS is a separate language with its own syntax. It is the recommended way to control the presentation layer in a web document. The main advantage of CSS over presentational HTML markup is that the styling can be kept entirely separate from the content.*



Cascading Style Sheets

- Cascading style sheets (CSS) are the modern standard for website presentation. When combined with a structural markup language such as HTML, XHTML, or XML (though not limited to these), *cascading style sheets provide Internet browsers with the information that enables them to present all the visual aspects of a web document.*
- Cascading style sheets apply things such as *borders, spacing between paragraphs, headings or images, control of font faces or font colors, background colors and images, textual effects such as underlined or strike-through text, layering, positioning, and a number of other presentational effects*



Cascading Style Sheets: Benefits

- The main benefit of using CSS is we can store all the presentational styles for a 10,000-page web site in a single CSS file. CSS also provides far better control over presentation than do presentational element types in XHTML. By externalizing the presentation layer, CSS offers a number of significant benefits:

1. *All styling is kept in a limited number of style sheets. The positive impact this has on site maintenance can't be overestimated editing one style sheet is obviously more efficient than editing 10,000 HTML files!*



Cascading Style Sheets: Benefits

2. *Since the style sheet is cached after the first request and can be reused for every page on the site, it doesn't have to be downloaded with each web page. Removing all presentational markup from your web pages in favor of using CSS also reduces their size and bandwidth usage by more than 50% in many documented cases.*
3. *The separation of content from presentation makes it easier for site owners to reuse the content for other purposes, such as RSS feeds or text-to-speech conversion.*
4. *Although CSS is designed to be independent of the markup language of the documents to which it is applied, in reality, it's used mainly with HTML and XML (including XHTML).*



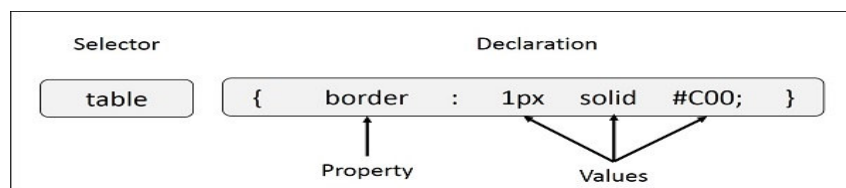
Cascading Style Sheets: Versions

1. **level 1 (CSS1)** was came out of W3C as a recommendation in December 1996. This version describes the CSS language as well as a simple visual formatting model for all the HTML tags.
2. **level 2 (CSS2)** was became a W3C recommendation in May 1998 and builds on CSS1. This version adds support for media-specific style sheets e.g. printers and aural devices, downloadable fonts, element positioning and tables.
3. **level 3 (CSS3)** was became a W3C recommendation in June 1999 and builds on older versions CSS. it has divided into documentations is called as Modules and here each module having new extension features defined in CSS2..



CSS: Style Rule Syntax

- A style sheet is made up of *one or more style instructions (called rules)* that describe how an element or group of elements should be displayed.
- In CSS terminology, the *two main sections* of a rule are the **selector** that identifies the elements to be affected, and the **declaration** that provides the rendering instructions.
- The following example shows of a CSS rule, which as we can see is made up of two parts:





CSS: Style Rule Syntax

1. ***The selector***, which indicates which element or elements the declaration applies to (if it applies to more than one element, you can have a comma-separated list of several elements)
2. ***The declaration***, which sets out how the elements should be styled. The declaration is also split into two parts, separated by a colon:
 1. ***A property***, which is the property of the selected element(s) that we want to affect, in this case the font-family property.
 2. ***A value***, which is a specification for this property; in this case it is the Arial typeface.



CSS: Types of Selectors

- ***A selector***, as its name implies, selects an element in our XHTML document. A few different types of selectors are available, with varying levels of specificity to target a large number of elements or just a few. in other words how many or few elements it selects.
1. ***Universal Selector***: The universal selector is merely an asterisk (*) acting as a “wild card” to select any and all a elements in the document. For example, this rule: `* { color: blue; }` would apply a blue foreground (text) color to all elements.



CSS: Types of Selectors

- 2. Element Selector:** An element selector selects all instances of an element, specified by its tag name. This selector is more specific than the universal selector, but it's still not very specific since it targets every occurrence of an element, no matter how many of them there may be. For example, the rule: ***em { color: red; }*** gives every ***em*** element the same red foreground color, even if there are thousands of them in a document. Element selectors are also known as type selectors.



CSS: Types of Selectors

- 3. Class Selector:** A class selector targets any element that bears the given class name in its class attribute. Because a class attribute can be assigned to practically any element in XHTML, and any number of elements can belong to the same class, this selector is not extremely specific but is still more specific than an element selector. In CSS, class selectors are preceded by a ***dot (.)*** to identify them. For example, this rule will style any elements belonging to the “info” class, whatever those elements happen to be: ***.info { color: purple; }***



CSS: Types of Selectors

4. **ID Selector:** An ID selector will select only the element carrying the specified identifier.

Practically any element can have an id attribute, but that attribute's value may be used only once within a single document. The ID selector targets just one element per page, making it much more specific than a class selector that might target many. ID selectors are preceded by a **hash (#)**. The following rule would give the element with the ID "introduction" a green foreground color: `#introduction { color: green; }.`



CSS: Types of Selectors

5. **Pseudo Class Selector:** A pseudo-class is somewhat akin to a class selector, but it selects an element in a particular state. It's preceded by a **colon (:)**, and only a few pseudo classes are available:

`:link { color: blue; }` //The :link pseudo class selects all elements that are hyperlinks.

`:visited { color: purple; }` //selects hyperlinks whose destination has been previously visited

`:active { color: red; }` //selects links in an active state

`:hover { color: green; }` // selects element that being "hovered" by a user's pointing device

`:focus { color: orange; }` //selects any element in a focused state



CSS: Types of Selectors

6. **Descendant Selector:** One of the most useful and powerful selectors in the CSS arsenal, a descendant selector can be assembled from two or more of the basic selector types (universal, element, class, pseudo class, and ID), separated by spaces, to select elements matching that position in the document tree. These are also called contextual selectors because they target elements based on their context in the document. For example: `#introduction .info p * { color: pink; }`, would select all elements that are descendants of a p element that is a descendant of an element with the class info that is a descendant of the element with the ID introduction.



CSS: Types of Selectors

7. **Grouping Selectors:** We can group several selectors together as part of a single rule so the same set of declarations can apply to numerous elements without redundantly repeating them. A comma separates each selector in the rule: `p, h1, h2 { color: blue; }`.
8. **Combining Selectors:** We can combine two or more selector types, such as an element and an ID or an ID and a class. These combinations can also narrow down the specificity of our selectors, seeking out only the elements we want to style and leaving others alone. For example: `p#introduction a.info:hover { color: silver; }` This rule would apply only to hovered links (a elements) belonging to the info class that are descendants of the paragraph with the ID introduction.



CSS: Types of Selectors

9. **Child Selectors:** We have seen the descendant selectors. There is one more type of selector, which is very similar to descendants but have different functionality. Consider this example: `body > p { color: #000000; }.` This rule will render all the paragraphs in black if they are direct child of `<body>` element. Other paragraphs put inside other elements like `` or `<td>` would not have any effect of this rule.
10. **Attribute Selectors:** we can also apply styles to HTML elements with particular attributes. The style rule below will match all the input elements having a type attribute with a value of text: `input[type = "text"]{ color: #000000; }.` The advantage to this method is that the `<input type = "submit" />` element is unaffected.



CSS - Inclusion : Inline styles

- To style our pages with CSS, we'll also need to connect our style sheets to our documents. When a graphical browser downloads the XHTML document and parses it for rendering, it will automatically seek out CSS rules to instruct it on how the various elements should be presented. There are four ways to associate styles with our HTML document, each with its own benefits and some drawbacks.
1. **Inline styles:** We can include CSS declarations within the optional style attribute of each element in our markup. Inline styles aren't constructed as rules, and there is no selector because the properties and values are attached directly to the element at hand. For example.



CSS - Inclusion : Inline styles

`<h2 style="color: red;">Good eats for hungry geeks</h2>`.

- Here we can see that the properties are added as the value of the style attribute. There is no **need for a selector** (because the style is automatically applied to the element that carries the style attribute), and there are no curly braces. We still need to separate each property from its value with a colon and each of the property-value pairs from each other with a semicolon.
- *However, we should avoid using inline styles. They mix presentation with our structural markup, thus negating one of the primary advantages of using CSS.*



CSS - Inclusion : Embedded Style

2. **Embedded Style Sheets:** We can embed style rules within the head element of our document, and those rules will be honored only for the document in which they reside. An embedded style sheet (sometimes called an internal style sheet) is contained within the style element.

```
<html >
<head><title></title>
<style type="text/css">
h2 { color: red; }
p { color: gray; }
</style> </head>
<body> <h2> embed style </h2> <p> embed style </p>
</body> </html>
```



CSS - Inclusion : External Style

3. **External style sheets:** The third and best option is to place all our CSS rules in a separate, external style sheet, directly connected to our documents. An external style sheet is a plain-text file that we can edit using the same text editing software we use to create our XHTML documents, saved with the file extension **.css**. This approach completely separates presentation from content and structure—they're not even stored in the same file. A single external style sheet can be linked from and associated with any number of XHTML documents, allowing your entire website's visual design to be controlled from one central file. An XHTML document links to an external style sheet via a **link element** in the document's head.



CSS - Inclusion : External Style

- The **<link /> element** can be used to create a link to CSS style sheets. The **<link /> element** is always an empty element that describes the relationship between two documents.
- When used with style sheets, the **<link /> element** must carry three attributes: **type**, **rel**, and **href**. Here is an example of the **<link /> element** finding a CSS file

```
<link rel="stylesheet" type="text/css" href="../stylesheets/interface.css" />
```

- In addition to the core attributes, the **<link /> element** can also take the following attributes:

Charset dir href hreflang media rel rev style target type



CSS - Inclusion : External Style

- There are several advantages to using **external CSS style** sheets rather than **internal style sheets** or **inline style rules**, including the following:
 1. *The same style sheet can be reused by all of the web pages in our site.*
 2. *We can change the appearance of several pages by altering just the style sheet rather than each individual page; this is particularly helpful if we want to change our company's colors, or the font used for a certain type of element wherever that element appears across the whole site.*
 3. *The style sheet can act as a style template to help different authors achieve the same style of document without learning all of the individual style settings.*



CSS - Inclusion : External Style

4. **@import Rule:** @import is used to import an external style sheet in a manner similar to the <link> element. Here is the generic syntax of @import rule.

<head>

<@import " mystyle.css ";

</head>

- We can use another syntax as well:

<head>

<@import url("mystyle.css");

</head>.



The Cascade Order

- We have discussed four ways to include style sheet rules in a an HTML document. Here is the rule to override any Style Sheet Rule.
 1. *Any inline style sheet takes highest priority. So, it will override any rule defined in `<style>...</style>` tags or rules defined in any external style sheet file.*
 2. *Any rule defined in `<style>...</style>` tags will override rules defined in any external style sheet file.*
 3. *Any rule defined in external style sheet file takes lowest priority, and rules defined in this file will be applied only when above two rules are not applicable.*



CSS: Examples

1. *Set the Text Color: `<p style="color:red;"> This text will be written in red. </p>`*
2. *Set the Text Direction: `<p style="direction:rtl;"> The text will be rendered right to left </p>`*
3. *Set the Space between Characters: `<p style="letter-spacing:5px;"> This text is having space between letters.</p>`*
4. *Set the Space between Words: `<p style="word-spacing:5px;"> This text is having space between words. </p>`*
5. *Set the Text Indent: `<p style="text-indent:1cm;"> This text will have first line indented by 1cm and this line will remain at its actual position this is done by CSS text-indent property. </p>`*



CSS: Examples

6. Set the Text Alignment:

1. `<p style="text-align:right;"> This will be right aligned.</p>`
2. `<p style="text-align:center;">This will be center aligned.</p>`
3. `<p style="text-align:left;">This will be left aligned. </p>`

7. Decorating the Text:

1. `<p style="text-decoration:underline;"> This will be underlined </p>`
2. `<p style="text-decoration:line-through;"> This will be striked through.</p>`
3. `<p style="text-decoration:overline;">This will have a over line.</p>`



CSS: Examples

8. Set the Font Family: `<p style="font-family:georgia,garamond,serif;"> This text is rendered in either georgia, garamond, or the default serif font depending on which font you have at your system. </p>`

9. Set the Font Size:

1. `<p style="font-size:20px;">This font size is 20 pixels</p>`
2. `<p style="font-size:small;">This font size is small</p>`
3. `<p style="font-size:large;">This font size is large</p>`

10. Shorthand Property: `<p style="font:italic small-caps bold 15px georgia;"> Applying all the properties on the text at once. </p>`



UNIVERSITY OF DIYALA

The End

Department of Computer Science
College of Science