### Attribute Selectors

- It is possible to specify selectors based on the presence or value of an attribute in HTML elements:

<table>
<thead>
<tr>
<th>Selector</th>
<th>Example</th>
<th>Selects</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>[attr=val]</code></td>
<td><code>[href=*].pdf&quot;</code></td>
<td>All elements whose <code>href</code> attribute value ends with the string &quot;*.pdf&quot;</td>
</tr>
<tr>
<td><code>[attr=val]</code></td>
<td><code>[href=&quot;csc&quot;]</code></td>
<td>All elements whose <code>href</code> attribute value contains the substring &quot;csc&quot;</td>
</tr>
<tr>
<td><code>[attr~val]</code></td>
<td><code>[class=&quot;alert&quot;]</code></td>
<td>All elements whose <code>class</code> attribute value contains the word &quot;alert&quot;</td>
</tr>
</tbody>
</table>

- Example:

```html
<p class="red-bold">This is bold text.</p>
```

### Pseudo Classes

- A CSS pseudo class is a keyword added to a selector that indicates a particular state of the selected element.

- Depending on the type of element, its state could include:
  - the navigation history (e.g., whether a link has been visited)
  - the status of its content (e.g., whether a checkbox has been checked)
  - the position of the mouse pointer (e.g., over the element)

- Style directives involving a pseudo class take the form

  ```css
  selector: pseudo-class { property: value; ... }
  ```

- Examples:

```css
a:link { color: blue; }
```

- Link styles:

  ```css
  :hover { color: red; }
  :visited { color: green; }
  a:link { color: blue; }
  ```

- Examples:

```html
<a href="#">Link</a>
```
**Pseudo Elements**

- A CSS pseudo element is a keyword added to a selector that lets you style a specific part of the selected element(s).

- Style directives involving a pseudo element take the form `selector::pseudo-element { property: value; ... }`

- Pseudo elements include:
  - `::first-letter`: Selects the first letter of every `p` element.
  - `::first-line`: Selects the first line of every `p` element.
  - `::marker`: Selects every marker of every `ul` element.
  - `::before`: Creates a pseudo element as first child of every `h1` element.
  - `::after`: Creates a pseudo element as last child of every `h1` element.

- There are a number of additional pseudo classes that relate to the state of form elements.

**External Style Sheets**

- External style sheets place style directives in a separate file or files.

- Multiple HTML documents can link to the same style sheet.

- Allows for a consistent look across multiple HTML documents on a website.

- Makes it possible to change a style in a single place which then propagates automatically to all HTML documents that use that style sheet.

- External style sheets represent the best separation of content and presentation.

**Pseudo Elements**

- A CSS pseudo class is a keyword added to a selector that indicates a particular state of the selected element.

- Pseudo classes include:
  - `h1::before`: Selects the portion of an element that is the first child of its parent.
  - `h1::after`: Selects the portion of an element that is the last child of its parent.
  - `::selection`: Selects the first letter of every `p` element.
  - `::first-line`: Selects the first line of every `p` element.
  - `::marker`: Selects every marker of every `ul` element.
  - `::before`: Creates a pseudo element as first child of every `h1` element.
  - `::after`: Creates a pseudo element as last child of every `h1` element.

- Every Pseudo Elements: Example

- External Style Sheets: Example
Multiple Style Sheets and Adaptive Design

- It is possible to use several external style sheets, and it is quite common to use one that specifies styles for a web browser and another for printing.
- The `media` attribute of the `<link>` element allows to select which external style sheets to use for which medium.

Example:

```html
<link rel="stylesheet" type="text/css" media="screen" href="browser.css" />
<link rel="stylesheet" type="text/css" media="print" href="print.css" />
```

- Different style sheets based on media features, e.g. screen resolution, are often used in adaptive design.
- Adaptive design uses different web pages or different layouts/styles depending on media features.

The Media Attribute

- The value of the `media`-attribute is a string that specifies a boolean combination of criteria based on media types and media features.
- Boolean operators are: `and` (conjunction), `not` (negation), `,` (disjunction).
- Media types are: `print`, `screen`, `speech`, `all`.
- Media features include:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>orientation</td>
<td>portrait or landscape</td>
</tr>
<tr>
<td>resolution</td>
<td>resolution in dpi</td>
</tr>
<tr>
<td>max-resolution</td>
<td></td>
</tr>
<tr>
<td>width</td>
<td>width in px</td>
</tr>
<tr>
<td>min-width</td>
<td></td>
</tr>
<tr>
<td>max-width</td>
<td></td>
</tr>
<tr>
<td>height</td>
<td>height in px</td>
</tr>
<tr>
<td>min-height</td>
<td></td>
</tr>
<tr>
<td>max-height</td>
<td></td>
</tr>
</tbody>
</table>

Example: `screen and (min-width:1920px)`

CSS: Resources


Revision and Further Reading

Read about Selectors and Inheritance in
- Chapter 11: Introducing Cascading Style Sheets
- Chapter 12: Formatting Text