

- The **Document Object Model** (DOM) is a programming interface for HTML or XML documents.
- Models document as a tree of nodes.
- Nodes can contain text and other nodes.
- Nodes can have attributes which include style and behavior attributes.
- Possible to get all nodes of a particular type, specific class or id.

Document Object Model

- API to access parsed HTML/XML documents.
- Can be used from any language, but within browsers the only language commonly supported currently is JavaScript.
- Datatypes include document, element, attribute.

- Current document available as document property of global window object.
- Properties include location (URL, giving href, protocol, hostname, port, pathname, search, hash), contentType, body, cookie (cookie defs separated by ;).
- Methods include `getElementsByTagName()`, `getElementsByName()`, `getElementById()`, `getElementsByClassName()`, `querySelector()`, `querySelectorAll()`.
- Allows updating document content dynamically *Dynamic HTML* (DHTML).

- Represents an individual HTML element.
- Properties include `id`, `classList`, `innerHTML` (markup within element), `attributes` (map `NamedNodeMap` of attributes).
- Methods include `getAttribute()`, `getAttributeNames()`, `removeAttribute()`, `setAttribute()`.

- Current best practice is to relegate presentation to stylesheets.
- Can be specified using **external** stylesheets, using `<link>` elements.
- Can also be specified using **internal** stylesheets using `<style>` elements.
- Can also be specified **inline** for an individual element using `style` attribute.
- Precedence (in descending order) inline, internal, external.

Cascading Style Sheets

- *Cascading Style Sheets* (CSS) specifies priority rules (cascade) between different style declarations which may apply to a element.
- A CSS stylesheet consists of a set of rules.
- A rule consists of a selector followed by a brace delimited set of CSS declarations separated by ;.

```
p.highlight {  
  background-color: yellow;  
  color: blue  
}
```

- Will not cover CSS declarations.

Simple CSS Selectors

Universal Selector * selects all elements; usually used in conjunction with other selectors.

HTML Element Names Simply specify name of HTML element. Examples p, a, table.

Class Selectors Name of class preceded by a .. Examples .highlight, .important.

ID Selectors ID of element preceded by #. Examples include #form1, #table1. Note that ID must be unique in document.

[attr] Selects all elements having attribute attr. Examples [href],

Combining Selectors

Constrain Can follow selector by class or id selectors (without spaces). `p.chemical` matches `p` elements having class `chemical`.

Descendent Simply write selectors adjacent to each other separated by a space. Example: `.chemical p` selects all `p` elements which are descendents of a element which has class `chemical`.

Child Write selectors separated by a `>`. Example: `.chemical > p` selects all `p` elements which are direct children of a element which has class `chemical`.

Combining Selectors Continued

Sibling Write selectors separated by a `~`. Example:
`.chemical ~ p` selects all `p` elements which follow (not necessarily immediately) a element which has class `chemical`.

Adjacent Sibling Write selectors separated by a `+`. Example:
`.chemical + p` selects all `p` elements which immediately follow a element which has class `chemical`.

Unobstrusive JavaScript

Different technologies used for different concerns:

Content HTML used for content.

Presentation CSS used for styling.

Behavior JavaScript used to specify behavior.

- Do not mix technologies.
- Best practice is to split out into separate *.html, *.css and *.js files.
- Modern technology blurs lines between concerns; CSS 3 contains support for visual behavior traditionally achieved using JavaScript. Nevertheless it remains a good organizational principle.

In doc.html:

```
<a href="submit.cgi"
  onClick="checkForm(this)"
  style="font-weight: bold">
  Submit
</a>
```

- Uses CSS and JavaScript code within attributes of HTML elements.
- Maintaining file will require content, presentational and programming skills.

- ❶ In `doc.html` maintained by content specialist or a *Content Management System* (CMS): `Submit`.
 - ❷ In `doc.css` maintained by web designer `#submit { font-weight: bold; }`.
 - ❸ In `doc.js` maintained by front-end programmer: `document.getElementById('submit').onclick(checkForm(this))`.
- Separate concerns, separate files, separate specialists.
 - `doc.html` will need to reference `doc.css` stylesheet and `doc.js`.
 - In practice, single `.css` stylesheet, `.js` file shared by multiple `html` documents.

Playing with the DOM

dom-play.html