POWERING POTENTIAL

The Document Object Model

→ JavaScript Fundamentals

INTRODUCTION

- What is the DOM?

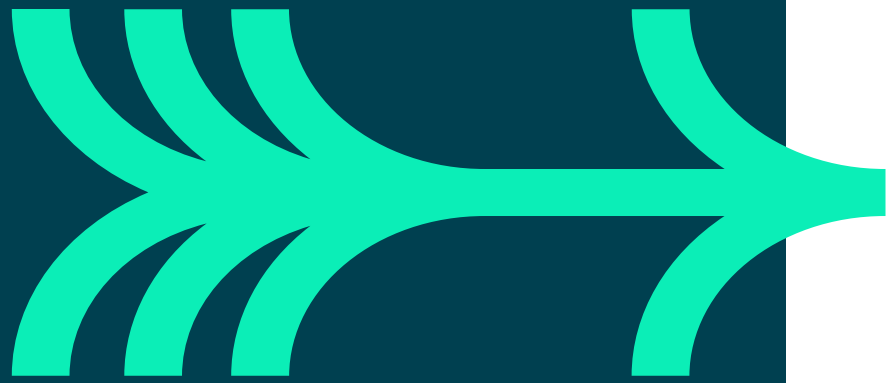
  → The DOM and HTML tree

- Selecting elements

  → Basic Selectors

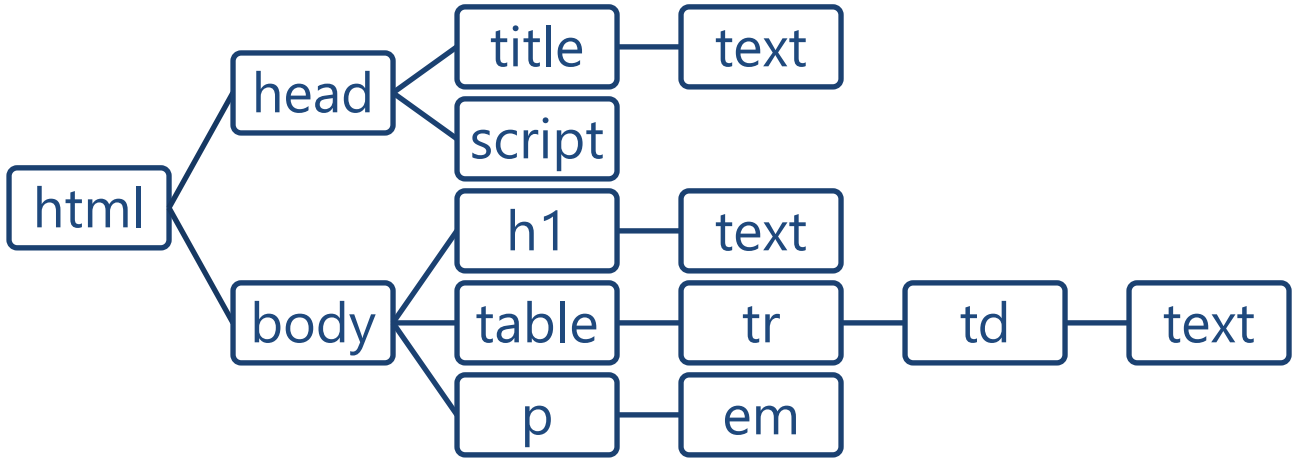
  → CSS Selector patterns

- Arrays of selected objects
- Creating new elements



What is the Document Object Model?

- HTML documents have a hierarchical structure that form the DOM
  - Every element, except <html>, is contained within another
  - Creating a parent/child relationship



- A DOM tree contains two types of elements
  - Nodes
  - Text

HTML markup to DOM object (1)

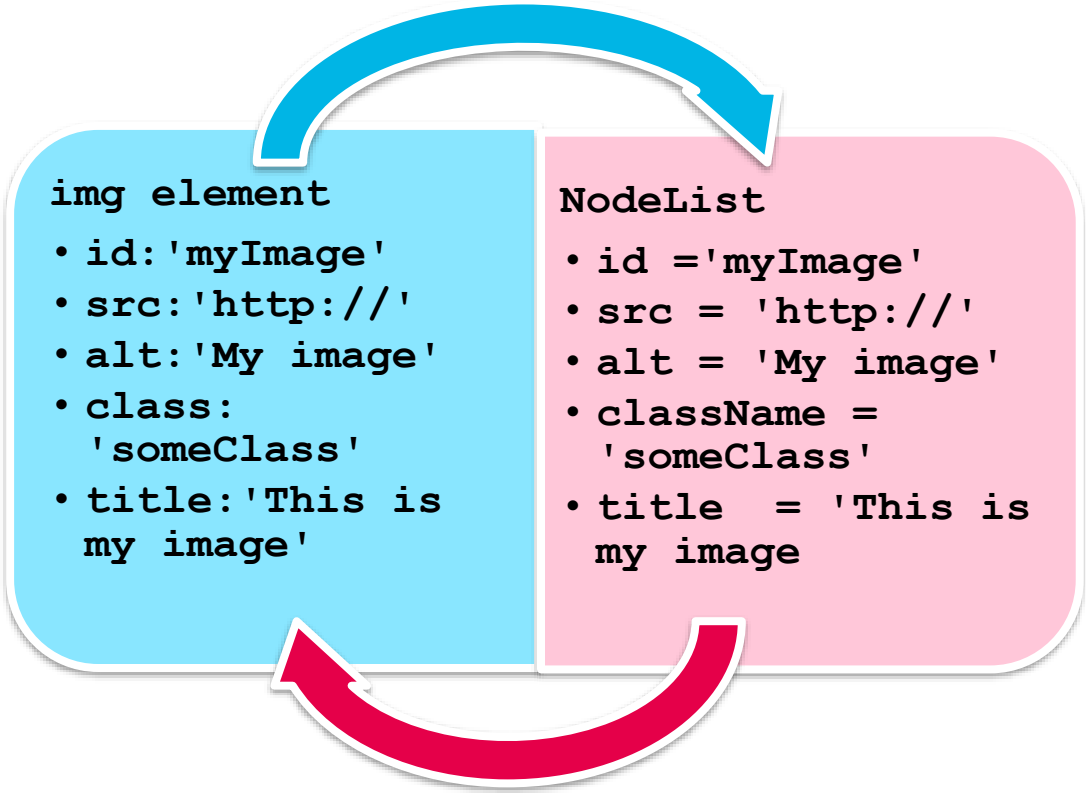
- Consider the following HTML

```
<img id="myImage" src="image.gif" alt="An image" title="This is an image"/>
```

- The tag has a type of <img> and four attributes
  - **`id`**
  - **`src`**
  - **`alt`**
  - **`title`**
- The element is read and interpreted by the browser into a DOM
  - Each element becomes a NodeList object
  - Assigned a property based on the html attribute

HTML markup to DOM object (2)

- HTML is translated into DOM elements, including the attributes of the tag and the properties created from them.



Selecting elements

- HTML DOM elements can be selected via JavaScript
  - Single elements can be selected in the following ways:

```
let x = document.getElementById('id');
```

```
let y = document.querySelector('#id');
```

  - Multiple elements can be selected using the following approaches:

```
let allP = document.getElementsByTagName('p');
```

```
let allA = document.querySelectorAll('div > a');
```

Basic Selectors

- CSS Selectors allow us to obtain almost any DOM element

Selector	Definition
'a'	This selector matches all link (<a>) elements.
#specialID	This selector matches elements that have an id of specialID.
'.specialClass'	This selector matches elements that have the class of specialClass.
'a#specialID.specialClass'	This selector matches links with an id of specialID and a class of specialClass.
'p a.specialClass'	This selector matches links with a class of specialClass declared within <p> elements.

Child, container, and attribute selectors (1)

- These selectors are part of the CSS specification
- Only exceptionally old browsers won't support them (pre-IE8)

Selector	Description
*	Matches any element
E	Matches all elements with tag name E
E F	Matches all elements with tag name F that are descendants of E
E>F	Matches all elements with tag name F that are direct children of E
E+F	Matches all elements F immediately preceded by sibling E
E~F	Matches all elements F preceded by any sibling E

Attribute selectors example

- Use attribute selectors with care as they can be expensive
  - A complex search pattern
- ^= operator finds attributes starting with a value

```
document.querySelectorAll('a[href^="http"]');
```

- $= operator finds attributes ending with a value

```
document.querySelectorAll('a[href$=".doc"]');
```

- *= operator finds attributes containing the value

```
document.querySelectorAll('a[href*="name"]');
```

Selecting by position

- Elements can be selected by position in relation to other elements

Selector	Description
:first-of-type	The first match of an element on a page. li a:first-of-type returns the first link also under a list item.
:last-of-type	The last match of the page. li a:last-of-type returns the last link also under a list item.
:first-child	The first child element. li:first-child returns the first item of each list.
:last-child	The last child element. li:last-child returns the last item of each list.
:only-child	Returns all elements that have no siblings.
:nth-child(n)	The nth child element. li:nth-child(2) returns the second list item of each list.
:nth-child(even|odd)	Even or odd children. li:nth-child(even) returns the even children of each list.

Creating new content – DOM programming

- The DOM can have new objects added to it using JavaScript

```
let el = document.createElement('p');
```

- This creates the **<p>** part of the html but not its text
  - The text node is part of DOM as well as the markup

```
let text = document.createTextNode ('stuff');
```

- Then you must append the text node to the element
  - Then the element to the DOM tree

```
el.appendChild(text);
document.querySelector('#id').appendChild(el);
```

Creating new content – innerHTML and textContent

- In the olden days, IE broke the DOM programming standard
  - All browsers now support **`innerHTML`** and **`innerText`** (prefer **`textContent`** over **`innerText`**)
- These functions allow us to add to the DOM in a quick and dirty way
    - The entire JavaScript string is parsed into a HTML element
    - Beware that older browsers can face injection attacks!

```
let el = document.querySelector('#id');
el.innerHTML = "<em>cool</em>";
```

QuickLab 20

→ Creating new content using the DOM

REVIEW

- What is the DOM?
- The DOM and HTML tree
- Selecting elements
- Basic Selectors
- CSS Selector patterns
- Arrays of selected objects
- Creating new elements