JS and HTML

The DOM, events, selectors, timers and callbacks

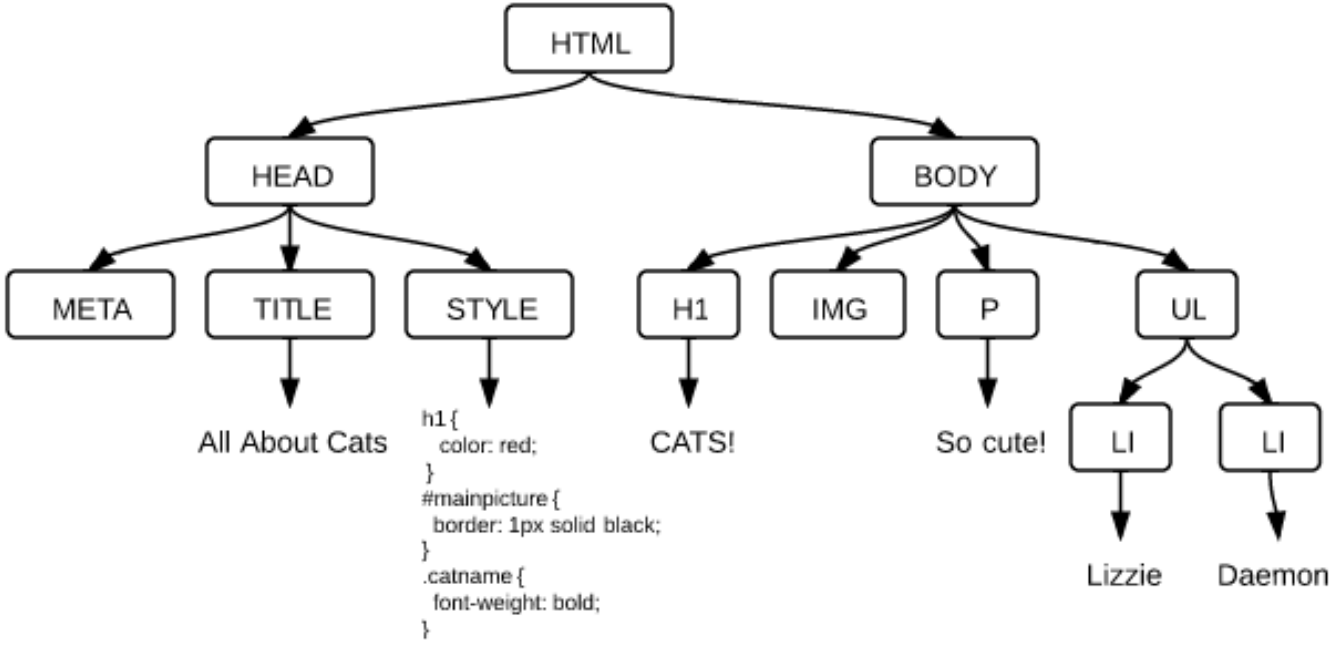
Our Goals

- Review
- Introduce the:
  - DOM
  - Selectors
  - Callbacks
  - Events
  - Animations

What is the DOM?

- It stands for the ***Document Object Model***
- It is the way that Javascript interacts with HTML and CSS
- It's basically a javascript object, but is mostly referred to as a ***tree***
- The browser always has it
- It comes from HTML
- It is your HTML and CSS when it is received and parsed by the browser. It can be different though!

What is the DOM?



Key things about DOM

- Each point of data is called a ***node***
- Each ***node*** can have ***parents***, ***children*** and ***siblings***
- The DOM is accessed through a global variable called:
  - `document`
- We can call methods and access properties - just like an object

Identify away!

```
<!DOCTYPE html>
<html>
<head>
    <title>Some website</title>
</head>
<body>
    <div class="container">
        <h1>Some heading</h1>
        <p>Some text</p>
        <a href="http://www.google.com">Some <span>link</span></a>
    </div>

    <ul>
        <li>A link</li>
        <li>Another link</li>
        <li>Another link</li>
    </ul>
</body>
</html>
```

DOM Access

The document object gives us ways of accessing the DOM, finding elements, changing styles, etc.

The general strategy for DOM manipulation:

- Find the DOM node by using an access method and store it in a variable
- Manipulate the DOM node by changing its attributes, style, inner HTML, or by appending nodes to it

DOM Access by ID

`document.getElementById( id );`

If we had this in our HTML:

```
<img src="http://fillmurray.com/300/300" id="mainImage">

<p id="randomParagraph">
    Gibberish goes here
</p>
```

We could access it like this

```
var img = document.getElementById( "mainImage" );

var p = document.getElementById( "randomParagraph" );
```

DOM Access by Tagname

`document.getElementsByTagName( tagName );`

If we had this in our HTML:

```
<ul>
    <li>A list item</li>
    <li>Another list item</li>
    <li>Another list item</li>
</ul>
```

We could access it like this

```
var listItems = document.getElementsByTagName( "li" );

for (var i = 0; i < listItems.length; i++) {
    var listItem = listItems[i];
}
```

DOM Access by Class name

`document.getElementsByClassName( tagName );`

If we had this in our HTML:

```
<ul>
    <li>A list item</li>
    <li class="item">Another list item</li>
    <li class="item">Another list item</li>
</ul>
```

We could access it like this

```
var listItems = document.getElementsByClassName("item");

for (var i = 0; i < listItems.length; i++) {
    var listItem = listItems[i];
}
```

HTML5 DOM Access

`document.querySelector( cssSelector );`

`document.querySelectorAll( cssSelector );`

```
<ul>
    <li>A list item</li>
    <li class="item">Another list item</li>
    <li class="item">Another list item</li>
</ul>
```

We could access it like this

```
var firstItem = document.querySelector("ul li");
// Only first item

var allItems = document.querySelectorAll("ul li.item");
// All list items with the class item
```

DOM Access - Array vs. Item

```
//////////////////
// Single item //
//////////////////

document.getElementById( "id" );
document.querySelector( "cssSelector" );
// First match

//////////////////////
// Array of items //
//////////////////////

document.getElementsByClassName( "class" );
document.querySelectorAll( "cssSelector" );
// All matches
```

Have a crack at these exercises

DOM Nodes: Attributes

You can access and change attributes!

If we had this in our HTML:

```
<img src="http://fillmurray.com/g/200/300" id="bill">
<a href="https://www.youtube.com/watch?v=jqGnfxa8-pg" id="satantango">
    Satantango
</a>
```

We could access it like this

```
var billImage = document.getElementById("bill");
billImage.src = "http://www.placecage.com/c/200/300";

var satantangoLink = document.getElementById("satantango");
satantangoLink.href = "https://www.youtube.com/watch?v=B7u704uMCmo";
satantango.className = "bestMovie";
```

DOM Nodes: Styles

You can access and change attributes!

```
// Changing Styles

var body = document.getElementsByTagName("body")[0];

body.style.background = "hotpink";
body.style.fontFamily = "Comic Sans";
body.style.paddingTop = "10px";
```

- CSS properties that normally have a hyphen in it, you must camelCase it
- Number properties must have a unit - they won't default to pixels

DOM Nodes: HTML

Each node, has an innerHTML property with the HTML of all of its children

```
// Accessing HTML

var body = document.getElementsByTagName("body")[0];

var bodyContent = body.innerHTML;
console.log( bodyContent );

// Setting HTML

body.innerHTML = "<h1>Bye bye, content!</h1>";
body.innerHTML += "<p>Oh well</p>";
```

Have a crack at these exercises

Creating DOM Nodes

We can make our own HTML elements as well!

```
// Create elements
var newParagraph = document.createElement('p');
var paragraphText = document.createTextNode('New Paragraph!');

// Style an element (before it is on the page)
newParagraph.style.fontFamily = "Comic Sans";
newParagraph.style.color = "hotpink";

// Put the element on the page
newParagraph.appendChild(paragraphText);
document.body.appendChild(newParagraph);
```

Have a crack at these exercises

Some terminology

- **Event:** something that happens
- **Callback:** a function that executes after the event has happened
- **Event listener:** a method that binds an event to a callback

A basic example

```
var myButton = document.querySelector('button#testButton');

myButton.addEventListener('click', function() {
  console.log('button clicked!');
});
```

The basic process:

- Find the element
- Add the event listener and pass in a function to call

That was an anonymous function

```
var myButton = document.querySelector('button#testButton');

myButton.addEventListener('click', function() {
  console.log('button clicked!');
});
```

You can't ever remove that event handler!

Referenced Events

```
var myButton = document.querySelector('button#testButton');

var myCallback = function () {
  console.log('button clicked!');
}

myButton.addEventListener('click', myCallback);

myButton.removeEventListener('click', myCallback);
```

Much better!

Events and *Event*

When you add an event listener, the callback that you specify to use will at some later point be called, and passed an object as its first argument.

That object provides all sorts of details about the event!

```
var myButton = document.querySelector('button#testButton');

var myCallback = function (event) {
  console.log( event );
}

myButton.addEventListener('click', myCallback);

myButton.removeEventListener('click', myCallback);
```

Timers

```
// ONCE OFF //
// window.setTimeout(someCallback, delayInMilliseconds);

window.setTimeout(function () {
    console.log("This function won't run for a second");
}, 1000);

// EVERY SO OFTEN //
// window.setInterval(someCallback, delayInMilliseconds);

window.setInterval(function () {
    console.log("This function will run every second");
}, 1000);
```

Animations in JS - Attributes

```
var bill = document.querySelector("img");

window.setInterval(function () {
  bill.setAttribute("width", bill.width += 10);
}, 100);
```

Animations in JS - Style

```
// Find the element, set the starting point
var bill = document.querySelector("img");
bill.style.opacity = 1.0;

var fadeAway = function () {
    bill.style.opacity = bill.style.opacity - 0.01;
};

window.setInterval(fadeAway, 10);
```

Animations in JS - Style

```
var bill = document.querySelector('img');

bill.style.position = 'absolute';
bill.style.top = '0px';

var watchBillFall = function() {
  var oldTop = parseInt(bill.style.top);
  var newTop = oldTop + 10;
  bill.style.top = newTop + 'px';
};

window.setInterval(watchBillFall, 1000);
```

Starting and Stopping

```
window.clearTimeout( timerID );
window.clearInterval( timerID );
```

```
var bill = document.querySelector("img");
bill.style.opacity = 1.0;

var fadeAway = function () {
    bill.style.opacity = bill.style.opacity - 0.01;
    if (bill.style.opacity < 0.5) {
        window.clearInterval( fadeTimer );
    }
};

var fadeTimer = window.setInterval(fadeAway, 10);
```

Our Goals

- Review
- Introduce the:
  - DOM
  - Selectors
  - Callbacks
  - Events
  - Animations

Your homework

- Previous examples