Intro to the DOM

Finally, JS meets HTML+CSS

Objectives

- Define what the DOM is
- Understand why DOM Manipulation is awesome
- List a few examples of sites that use JS to manipulate the DOM
- Understand the SELECT, then MANIPULATE workflow

Why Should You Care?

A few examples:

- Games
- Scrolling Effects
- Dropdown menus
- Form Validations
- Interactivity
- Animations
- Every awesome site ever

The DOM

Document Object Model

The Document Object Model is the interface between your Javascript and HTML+CSS

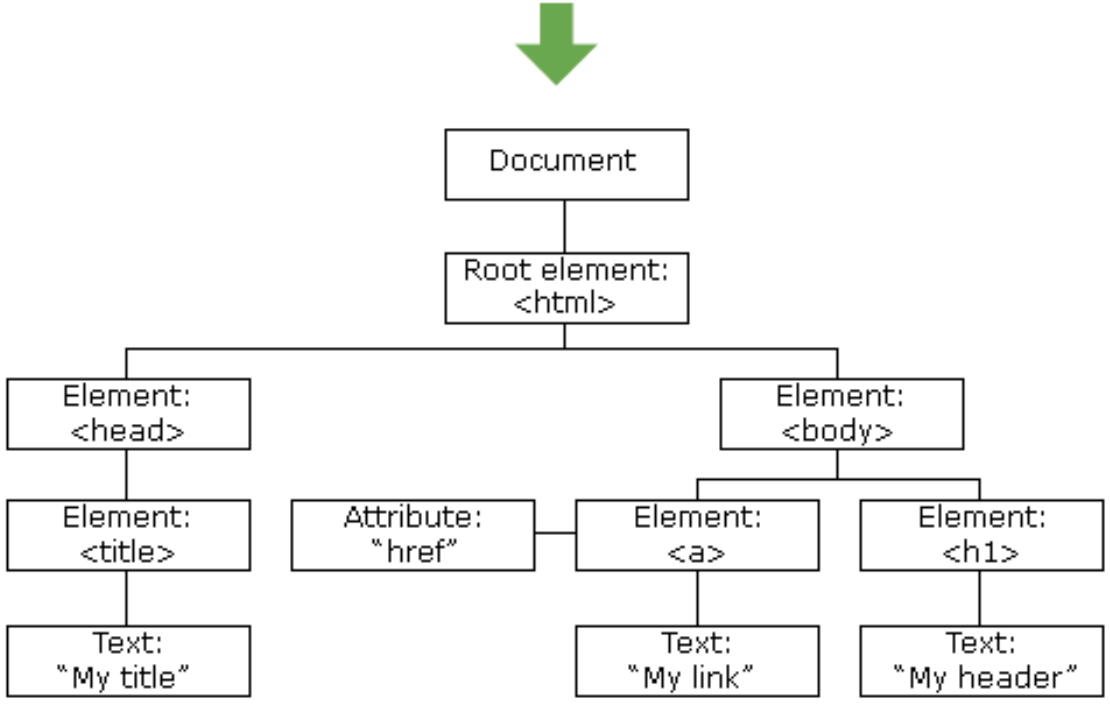
The browser turns every HTML tag into a Javascript object that we can manipulate

```
<!DOCTYPE html>
<html>
<head>
     <title>My title</title>
</head>
<body>
      <a>href="someLink">Mylink</a>
     <h1>My header</h1>
</body>
</html>
```

My link

My header

Everything is stored inside of the *document* object



The Process

SELECT an element and then MANIPULATE

```
<!DOCTYPE html>
<html>
<head>
        <title>My title</title>
</head>
<body>
        <a href="someLink">My link</a
        <h1>My header</h1>
</body>
</html>
```

For our example, we'll change the <h1> color using JS

The Process

SELECT an element and then MANIPULATE

```
var h1 = document.querySelector("h1");
```

SELECT the <h1> and save to a variable

The Process

SELECT an element and then MANIPULATE

```
var h1 = document.querySelector("h1");

h1.style.color = "pink";
```

MANIPULATE using the <*h1*> we selected

One more example

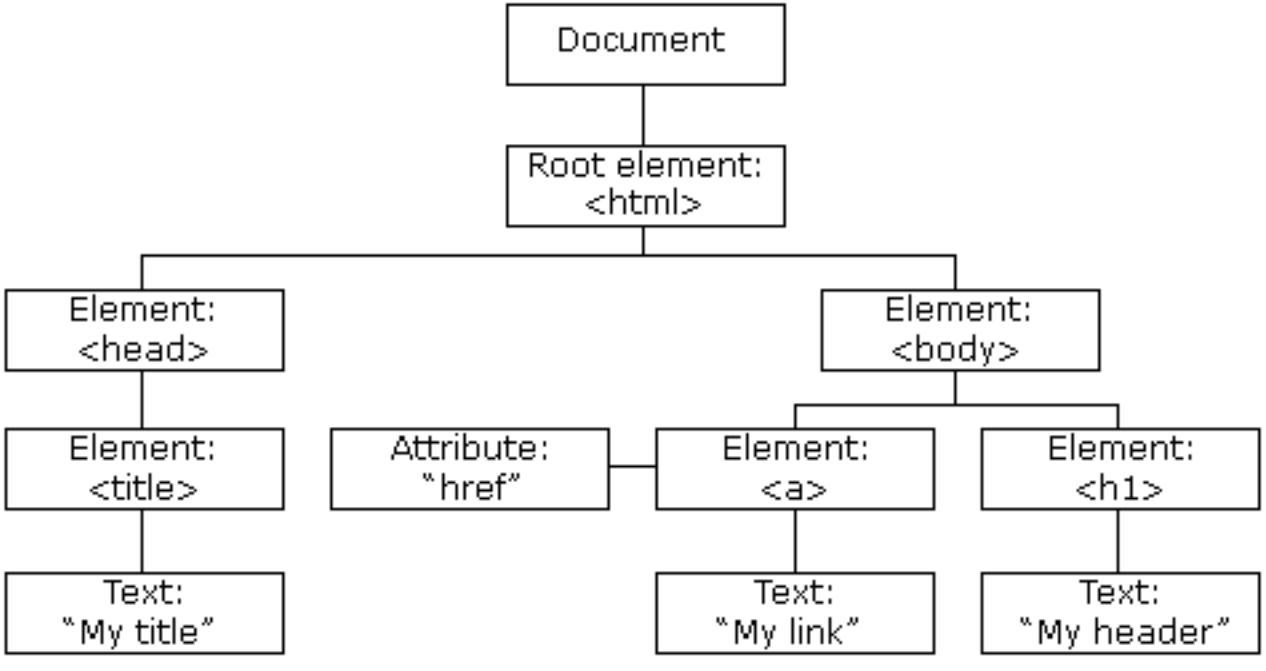
SELECT the <*body*> and change its color every second

```
var body = document.querySelector("body"); //SELE
var isBlue = false;

setInterval(function(){   //MANIPULATE
 if (isBlue) {
   body.style.background = "white";
 } else {
   body.style.background = "#3498db";
 }
 isBlue = !isBlue;
}
, 1000);
```

DOM Selectors

Document

It all starts with the document, the root node



Exercise

Open up the JS console and try these 4 lines:

```
document.URL;
document.head;
document.body;
document.links;
```

Methods

The document comes with a bunch of methods for selecting elements. We're going to learn about the following 5:

- document.getElementById()
- document.getElementsByClassName()
- document.getElementsByTagName()
- document.querySelector()
- document.querySelectorAll()

getElementById

Takes a string argument and returns the one element with a matching ID

```
var tag = document.getElementById("highlight");
```

```
<body>
    <h1>Hello</h1>
    <h1>Goodbye</h1>
    <ul>
        <li id="highlight">List Item 1</li
        <li class="bolded">List Item 2</li
        <li class="bolded">List Item 3</li
    </ul>
</body>
```

getElementsByClassName

Takes a string argument and returns the one element with a matching class

```
r tags = document.getElementsByClassName("bolded");
nsole.log(tags[0]);
```

```
<body>
    <h1>Hello</h1>
    <h1>Goodbye</h1>
    <ul>
        <li id="highlight">List Item 1</li>
        <li class="bolded">List Item 2</li>
        <li class="bolded">List Item 3</li>
    </ul>
</body>
```

getElementsByTagName

Returns a list of all elements of a given tag name, like <li> or <*h1*>

```
var tags = document.getElementsByTagName("li");
console.log(tags[0]);
```

```
<body>
    <h1>Hello</h1>
    <h1>Goodbye</h1>
    <ul>
        <li id="highlight">List Item 1</li>
        <li class="bolded">List Item 2</li>
        <li class="bolded">List Item 3</li>
    </ul>
</body>
```

getElementsByTagName

Returns a list of all elements of a given tag name, like <li> or <*h1*>

```
var tags = document.getElementsByTagName("h1");
console.log(tags[0]);
```

```
<body>
    <h1>Hello</h1>
    <h1>Goodbye</h1>
    <ul>
        <li id="highlight">List Item 1</li>
        <li class="bolded">List Item 2</li>
        <li class="bolded">List Item 3</li>
    </ul>
</body>
```

querySelector

Returns the first element that matches a given CSS-style selector

```
//select by ID
var tag = document.querySelector("#highlight");
```

```
<body>
    <h1>Hello</h1>
    <h1>Goodbye</h1>
    <ul>
        <li id="highlight">List Item 1</li
        <li class="bolded">List Item 2</li
        <li class="bolded">List Item 3</li
    </ul>
</body>
```

querySelector

Returns the first element that matches a given CSS-style selector

```
//select by Class
var tag = document.querySelector(".bolded");
```

```
<body>
    <h1>Hello</h1>
    <h1>Goodbye</h1>
    <ul>
        <li id="highlight">List Item 1</li>
        <li class="bolded">List Item 2</li>
        <li class="bolded">List Item 3</li>
    </ul>
</body>
```

querySelector

Returns **the first element** that matches a given CSS-style selector

```
//select by tag
var tag = document.querySelector("h1");
```

```
<body>
    <h1>Hello</h1>
    <h1>Goodbye</h1>
    <ul>
        <li id="highlight">List Item 1</li>
        <li class="bolded">List Item 2</li>
        <li class="bolded">List Item 3</li>
    </ul>
</body>
```

querySelectorAll

Returns **a list of elements** that matches a given CSS-style selector

```
//select by tag
var tags = document.querySelectorAll("h1");
```

```
<body>
    <h1>Hello</h1>
    <h1>Goodbye</h1>
    <ul>
        <li id="highlight">List Item 1</li>
        <li class="bolded">List Item 2</li>
        <li class="bolded">List Item 3</li>
    </ul>
</body>
```

querySelectorAll

Returns **a list of elements** that matches a given CSS-style selector

```
//select by class
var tags = document.querySelectorAll(".bolded");
```

```
<body>
    <h1>Hello</h1>
    <h1>Goodbye</h1>
    <ul>
        <li id="highlight">List Item 1</li>
        <li class="bolded">List Item 2</li>
        <li class="bolded">List Item 3</li>
    </ul>
</body>
```

Exercise

Come up with 4 different ways to select the first <p> tag

```
<!DOCTYPE html>
<html>
    <head>
        <title>My title</title>
    </head>
    <body>
        <h1>I am an h1!</h1>
        <p id="first" class="special">Hello</
        <p class="special">Goodbye</p>
        <p>Hi Again</p>
        <p id="last">Goodbye Again</p>
    </body>
</html>
```

DOM Manipulation

DOM Manipulation

DOM Manipulation

We're going to cover different ways of:

- changing an element's style
- adding/removing classes
- changing the content of a tag
- changing attributes(src, href, etc.)

Style

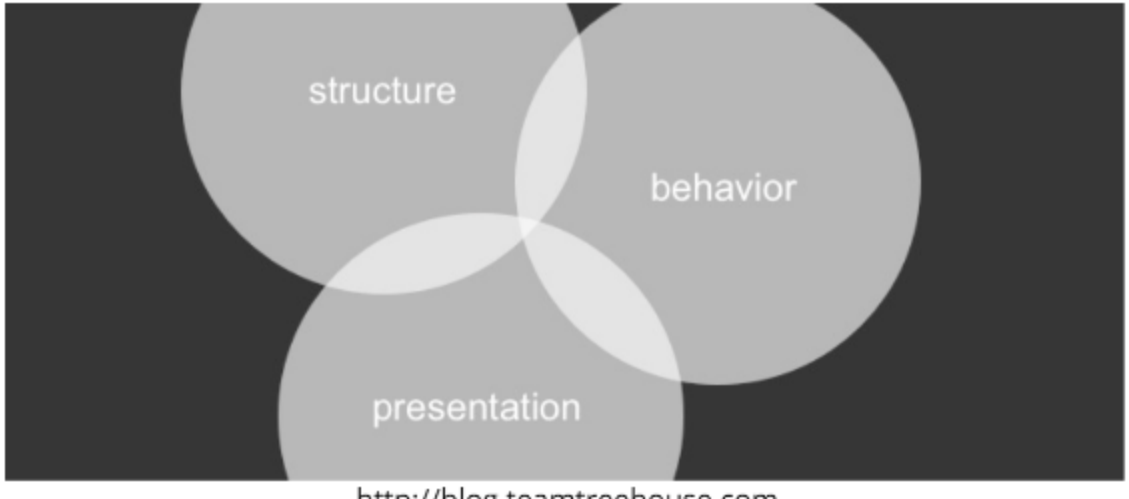
The style property is one way to manipulate an element's style

```
//SELECT

var tag = document.getElementById("highlight");


//MANIPULATE

tag.style.color = "blue";

tag.style.border = "10px solid red";

tag.style.fontSize = "70px";

tag.style.background = "yellow";

tag.style.marginTop = "200px";
```

Is This a Bad Idea?

It is recommended for styles to be defined in a separate file or files. The style property allows for quick styling, for example for testing purposes. - MDN

Separation of Concerns



http://blog.teamtreehouse.com

An Alterative

Rather than directly manipulating style with JS, we can define a CSS class and then toggle it on or off with JS

```
//INSTEAD OF THIS:
var tag = document.getElementById("highlight");
tag.style.color = "blue";
tag.style.border = "10px solid red";

        /*DEFINE A CLASS IN CSS*/
        .some-class {
          color: blue;
          border: 10px solid red;
        }

var tag = document.getElementById("highlight");
//ADD THE NEW CLASS TO THE SELECTED ELEMENT
tag.classList.add("some-class");
```

classList

A read-only list that contains the classes for a given element. It is **not an array**.

```
/*DEFINE A CLASS IN CSS*/
.another-class {
  color: purple;
  fontSize: 76px;
}
```

```
var tag = document.querySelector("h1");

//ADD A CLASS TO THE SELECTED ELEMENT
tag.classList.add("another-class");

//REMOVE A CLASS
tag.classList.remove("another-class");

//TOGGLE A CLASS
tag.classList.toggle("another-class");
```

textContent

Returns a string of all the text contained in a given element

```
<p>
  This is an <strong>awesome</strong> paragrap
</p>
```

```
/Select the <p> tag:
var tag = document.querySelector("p");

//Retrieve the textContent:
tag.textContent //"This is an awesome paragrap

//alter the textContent:
tag.textContent = "blah blah blah";
```

innerHTML

Similar to textContent, except it returns a string of all the HTML contained in a given element

```
<p>
  This is an <strong>awesome</strong> paragraph
</p>
```

```
//Select the <p> tag:
var tag = document.querySelector("p");

tag.innerHTML
//"This is an <strong>awesome</strong> paragraph
```

Attributes

Use *getAttribute()* and *setAttribute()* to read and write attributes like *src* or *href*

```
<a href="www.google.com">I am a link</a>
<img src="logo.png">

var link = document.querySelector("a");
link.getAttribute("href");  //"www.google.com
//CHANGE HREF ATTRIBUTE
link.setAttribute("href","www.dogs.com");
///<a href="www.dogs.com">I am a link</a>

//TO CHANGE THE IMAGE SRC
var img = document.querySelector("img");
img.setAttribute("src", "corgi.png");
//<img src="corgi.png">
```

DOM Events

Making things interactive

Events are everywhere

- Clicking on a button
- Hovering over a link
- Dragging and Dropping
- Pressing the Enter key

The Process

We **select** an element and then add an **event listener**

"Listen for a click on this <button>"

"Listen for a hover event on the <h1>"

"Listen for a keypress event on text input"

The Syntax

To add a listener, we use a method called *addEventListener*

```
element.addEventListener(type, functionToCall);
```

```
var button = document.querySelector("button");
button.addEventListener("click", function() {
    console.log("SOMEONE CLICKED THE BUTTON!");
})
```

An Example

Let's display a message when a button is clicked

```
<button>Click Me</button>
<p>No One Has Clicked Me Yet</p>

var button = document.querySelector("button");
var paragraph = document.querySelector("p");

//SETUP CLICK LISTENER
button.addEventListener("click", function() {
    paragraph.textContent = "Someone Clicked the Button!
});
```

An Example

We could also rewrite it using a named function

```
var button = document.querySelector("button");
var paragraph = document.querySelector("p");

button.addEventListener("click", changeText);

function changeText() {
  paragraph.textContent = "Someone Clicked the Buttor
}
```

So Many Events!

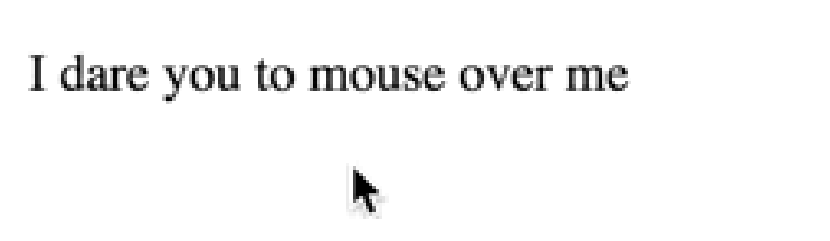
MDN lists over 300 different events! Here are some of the more common ones:

- click
- mouseover
- dblclick
- keypress
- drag

Another Example

Let's try a quick example using mouseOver

```
<p>I dare you to mouse over me</p>

var paragraph = document.querySelector("p");

//SETUP MOUSE OVER LISTENER
paragraph.addEventListener("mouseover", function() {
    paragraph.textContent = "Stop hovering over me!"
});
```



Adding mouseout

Let's use *mouseout* so that our message changes back when the user is done hovering

```
var paragraph = document.querySelector("p");

//SETUP MOUSE OVER LISTENER
paragraph.addEventListener("mouseover", function() {
    paragraph.textContent = "Stop hovering over me!";
});

//SETUP MOUSE OUT LISTENER
paragraph.addEventListener("mouseout", function() {
    paragraph.textContent = "Phew, thank you for leaving me alo
});
```

A Minor Change

We can DRY up our code with one small change:

```
var paragraph = document.querySelector("p");

//SETUP MOUSE OVER LISTENER
paragraph.addEventListener("mouseover", function() {
    this.textContent = "Stop hovering over me!";
});

//SETUP MOUSE OUT LISTENER
paragraph.addEventListener("mouseout", function() {
    this.textContent = "Phew, thank you for leaving me alone"
});
```

Phew, thank you for leaving me alone

DOM Problem Set

Selectors + Events

Color Changer

Toggle the body's background color between purple and white, when a button is clicked

```
<button onclick="myFunction()">CLICK ME</button>
```

```
var body=document.querySelector("body");
var ispurple=false;
function myFunction(){
    if(ispurple)
    {
        body.style.background="white";
    }
    else
    {
        body.style.background="purple";
    }
    ispurple=!ispurple;
}
```

Rainbow List

Change the background color of 7 different 's when a button is clicked, each to a different color of the rainbow. Make your code DRY!

```
<button onclick="myFunction()">CLICK ME</button>
<ul>
  <li id="List1">List Item 1</li>
  <li id="List2">List Item 2</li>
  <li id="List3">List Item 3</li>
  <li id="List4">List Item 4</li>
  <li id="List5">List Item 5</li>
  <li id="List6">List Item 6</li>
  <li id="List7">List Item 7</li>
</ul>
```


```
function myFunction(){
    var item1=document.getElementById("List1");
    item1.style.backgroundColor="violet";
    var item2=document.getElementById("List2");
    item2.style.backgroundColor="indigo";
    var item3=document.getElementById("List3");
    item3.style.backgroundColor="blue";
    var item4=document.getElementById("List4");
    item4.style.backgroundColor="green";
    var item5=document.getElementById("List5");
    item5.style.backgroundColor="yellow";
    var item6=document.getElementById("List6");
    item6.style.backgroundColor="orange";
    var item7=document.getElementById("List7");
    item7.style.backgroundColor="red";
}
```