

CST 366 - Internet Programming

Spring 2020 - Week 2.2

DOM

What is DOM

Document Object Model

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

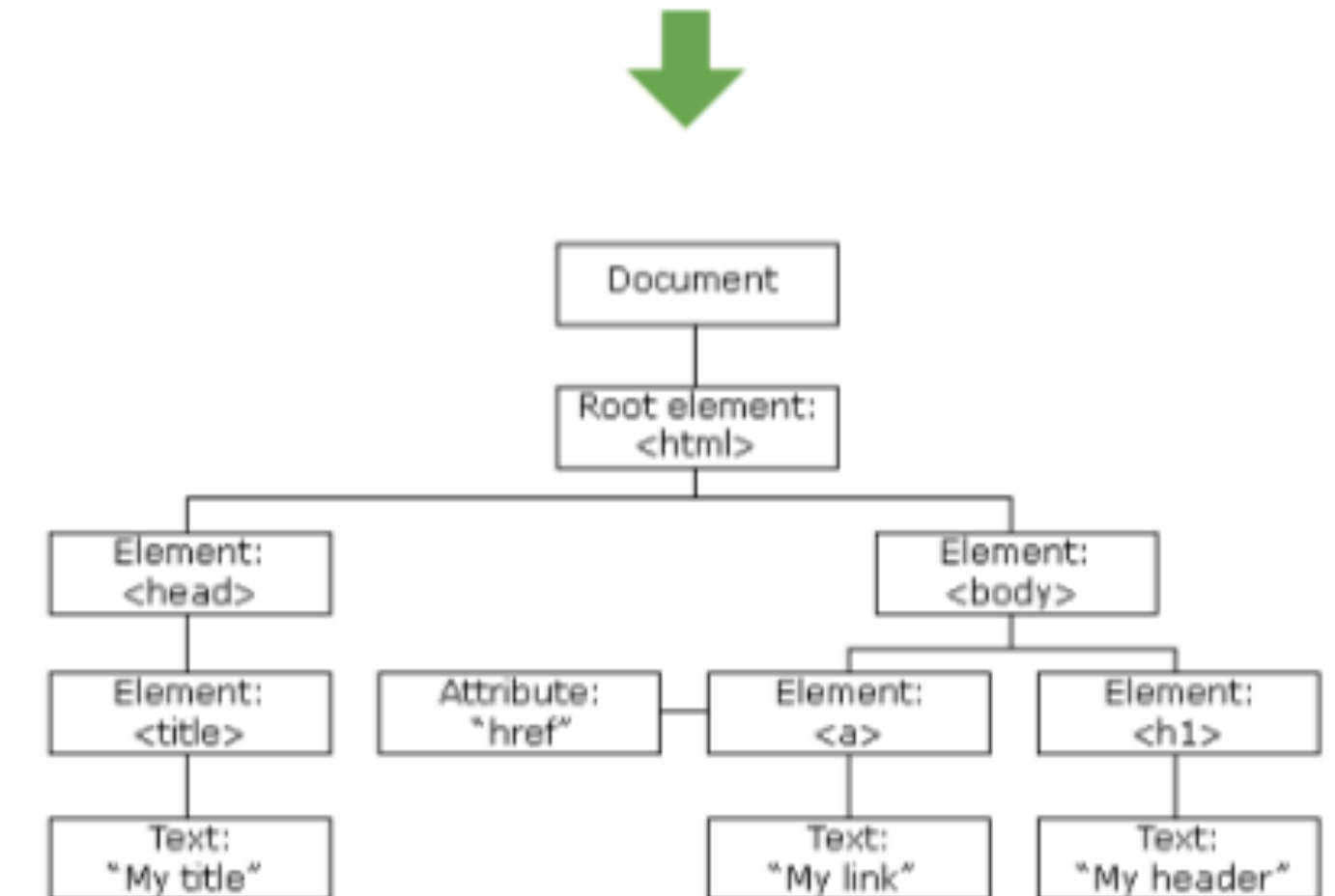
The browser turns every HTML tag into a Javascript object that we can manipulate

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

[My link](#)

My header

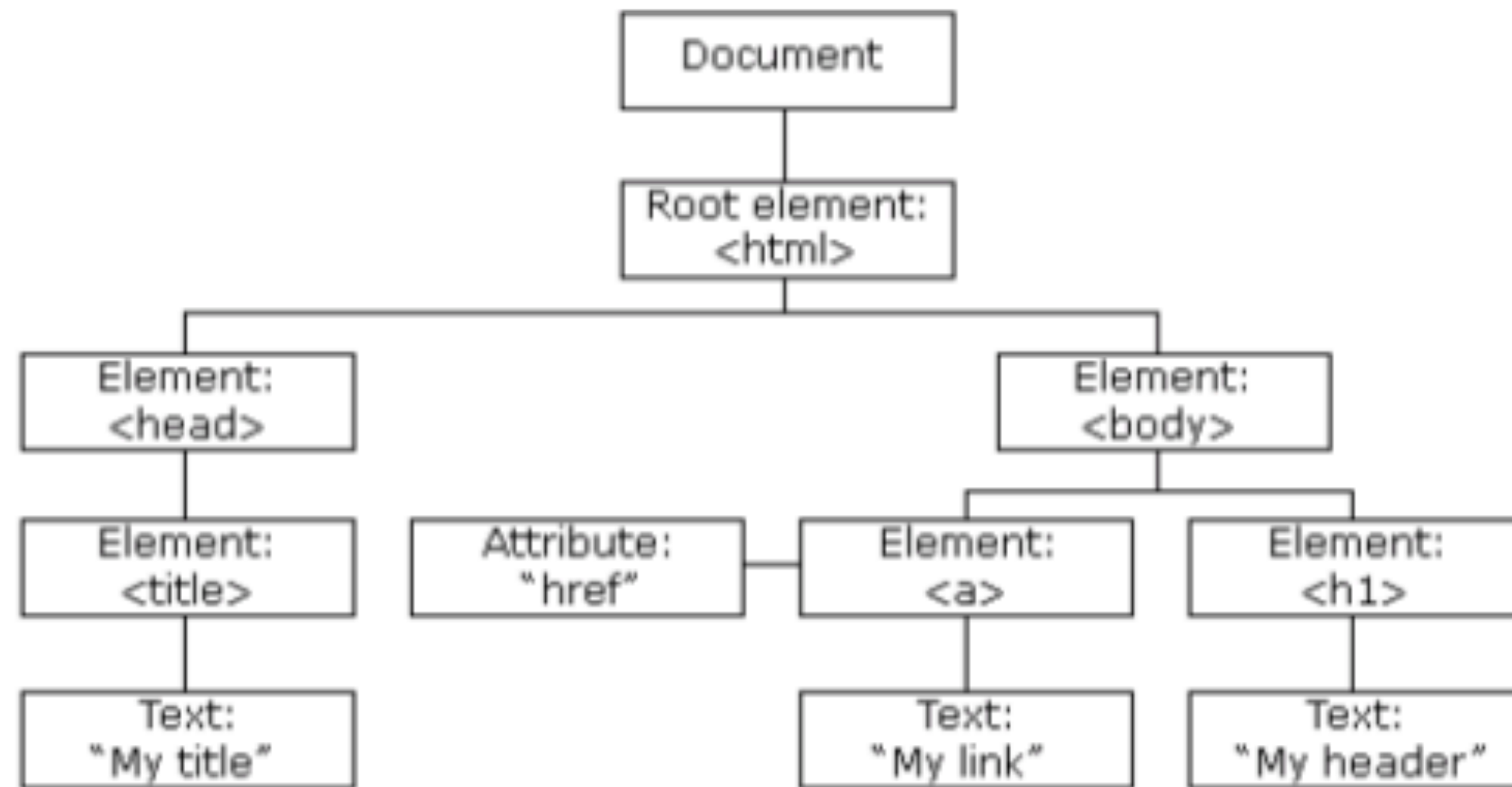
Everything is stored inside of the *document* object



DOM Selectors

Document

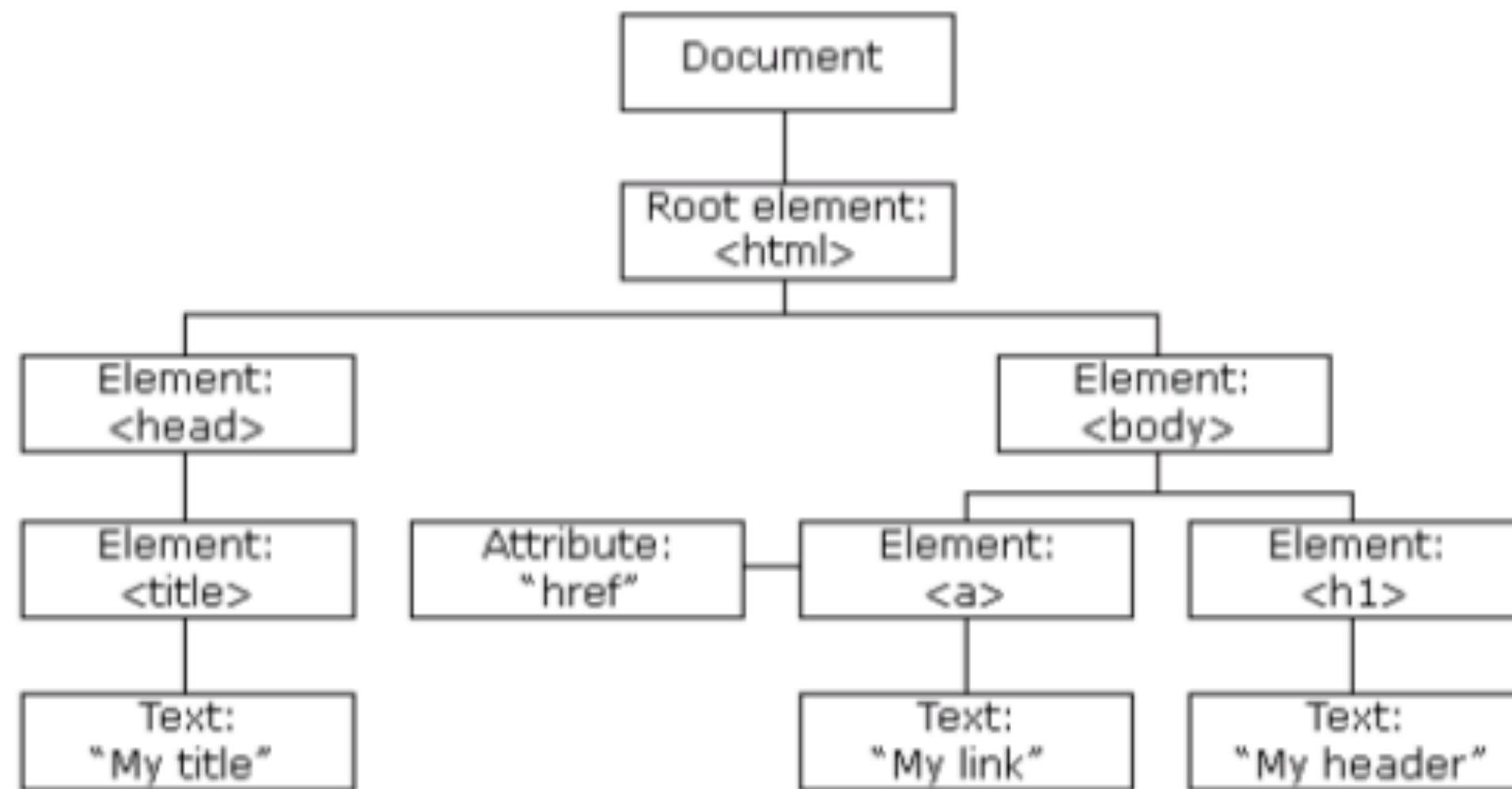
It all starts with the document, the root node



DOM Selectors

Document

It all starts with the document, the root node



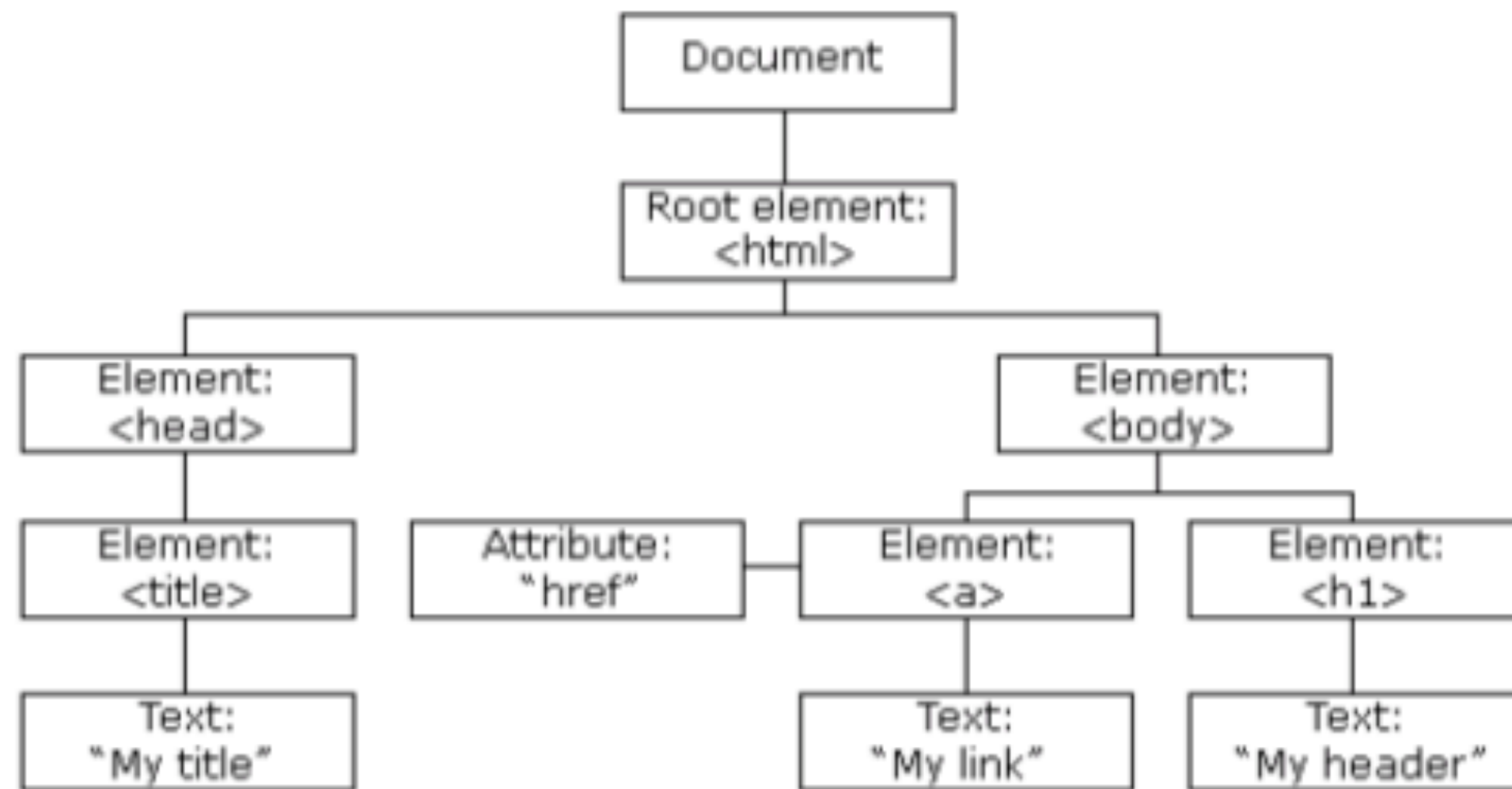
Open up the JS console and try these 4 lines:

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

DOM Selectors

Document

It all starts with the document, the root node



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()`

DOM Selectors

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>
```


DOM Selectors

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()

getElementsByClassName

Takes a string argument and returns a list of elements that have a matching class

```
var tags = document.getElementsByClassName("bolded");  
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>
```

DOM Selectors

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()

getElementsByTagName

Returns a list of all elements of a given tag name, like `` 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>
```


DOM Selectors

querySelector

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

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

```
<body>  
  <h1h1>  
  <h1h1>  
  <ul>  
    <li id="highlight">List Item 1</li>  
    <li class="bolded">List Item 2</li>  
    <li class="bolded">List Item 3</li>  
  </ul>  
</body>
```

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()

DOM Selectors

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>
```

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()

DOM Selectors

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>
```

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()

DOM Manipulation

```
<!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

[My link](#)

My header

[My link](#)

My header

DOM Manipulation

```
<!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

[My link](#)

My header

[My link](#)

My header

[My link](#)

My header

[My link](#)

My header

SELECT an element and then MANIPULATE

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

SELECT the <h1> and save to a variable



DOM Manipulation

```
<!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

[My link](#)

My header

[My link](#)

My header

SELECT an element and then MANIPULATE

```
var h1 = document.querySelector("h1");
h1.style.color = "pink";
```

MANIPULATE using the `<h1>` we selected

[My link](#)

My header



[My link](#)

My header

DOM Manipulation

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

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

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

[My link](#)

My header



[My link](#)

My header

DOM Manipulation

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";
```

DOM Manipulation

Style

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



```
//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";
```


```
//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");
```


DOM Manipulation

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";
```

```
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");
```


DOM Manipulation

textContent

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

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

```
/Select the <p> tag:  
var tag = document.querySelector("p");  
  
//Retrieve the textContent:  
tag.textContent //"This is an awesome paragraph"  
  
//alter the textContent:  
tag.textContent = "blah blah blah";
```

DOM Manipulation

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"
```

DOM Manipulation

Attributes

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

```
<a href="www.google.com">I am a link</a>  

```

```
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");  
//
```

DOM Events

Events are everywhere

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

To register a DOM event: we **select** an element and then add an **event listener**

DOM Events

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!");  
});
```


DOM Events

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!";  
});
```



No One Has Clicked Me Yet

DOM Events

So Many Events!

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

- click
- mouseover
- dblclick
- keypress
- drag

DOM Events

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!";
});
```

I dare you to mouse over me

DOM Events

So Many Events!

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

- click
- mouseover
- dblclick
- keypress
- drag

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 alone";
});
```

I dare you to mouse over me