

JavaScript and the DOM

Read This Code

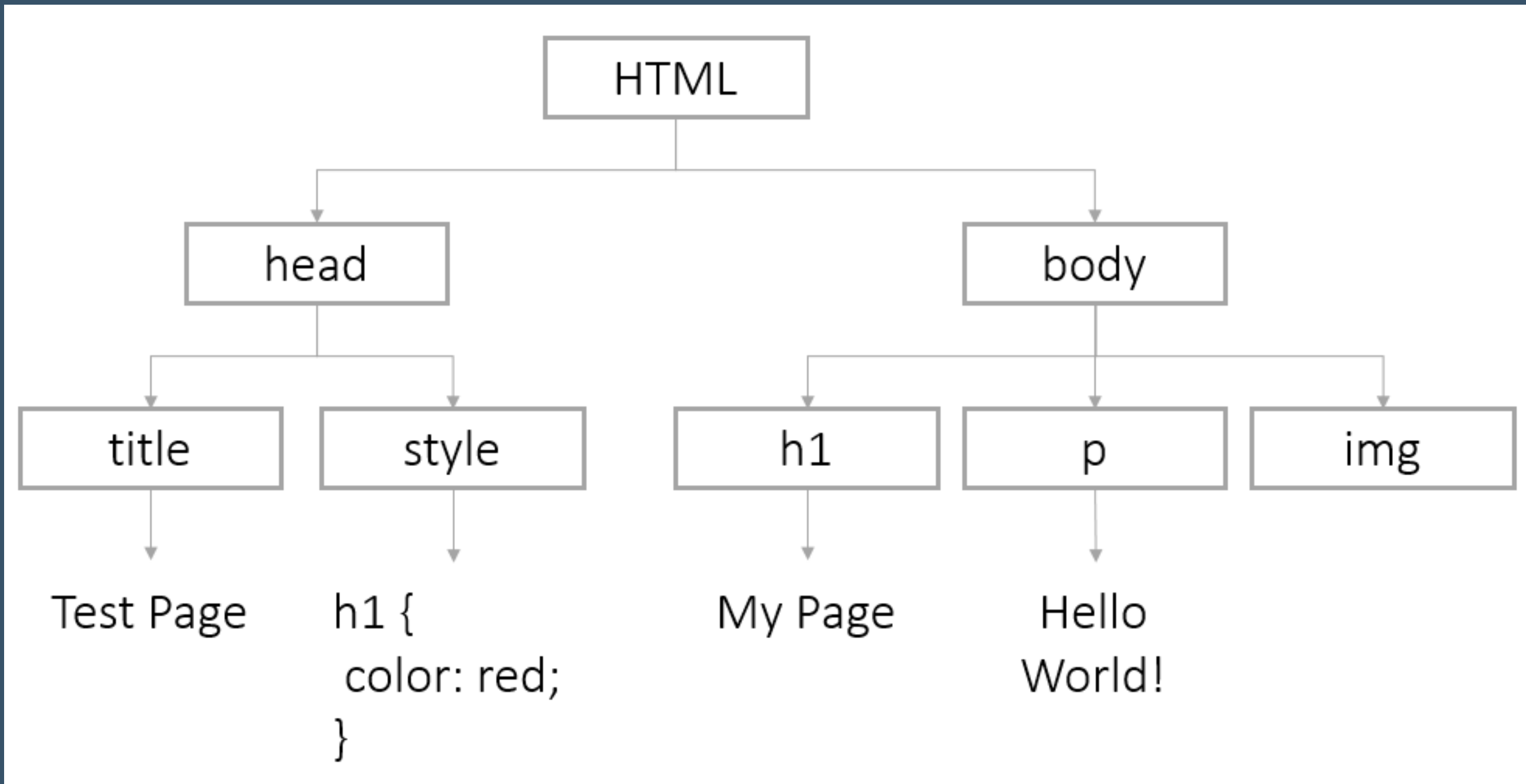
What does it do?

```
for (let headshot of document.querySelectorAll(".staff-member__headshot")) {  
  headshot.addEventListener("mouseover", function(event) {  
    headshot.src = headshot.dataset["gif"];  
  });  
  headshot.addEventListener("mouseout", function(event) {  
    headshot.src = headshot.dataset["src"];  
  });  
}
```

The Document Object Model (DOM) is a programming interface for HTML documents. It represents the page so that programs can change the document structure, style and content. The DOM represents the document as nodes and objects. That way, programming languages can connect to the page.

-- source [MDN, Introduction to the DOM](#)

The DOM tree and DOM nodes



Selecting DOM nodes

You can find and "grab" elements on the page (DOM nodes) with JavaScript.

```

```

There is more than one way to do this.

```
let image = document.getElementById("kittenPic");  
let image = document.querySelector("#kittenPic");
```

querySelector and querySelectorAll

We can use CSS selectors to get elements.

```
// This will return the first element matching the provided selector  
document.querySelector(".profile-photo");
```

```
// This returns a collection of elements matching the selector  
document.querySelectorAll(".profile-photo");
```

Changing things in the DOM

Once we can select DOM elements, we can make changes to them.

We can use JavaScript for...

- Adding or removing element attributes
- Adding or removing classes
- Adding or removing entire elements and/or their content

Let's look at how to do this!

Updating element attributes

```
let mainImg = document.querySelector("#main-image");  
mainImg.src;  
// what will this return?  
mainImg.src = "https://source.unsplash.com/200x300/?ocean";  
// sets the src attribute to this value  
mainImg.style.border = "4px solid purple";  
// sets the style attribute and border property to this value
```


Changing classes

Making changes to classes lets us apply styles to, or remove styles from, elements in the DOM.

```
let el = document.querySelector(".input--name");  
el.classList.add("highlight");
```

Adding things to the DOM

```
let existingElement = document.querySelector(".empty-div");  
// First select where you want to place the new content  
let newElement = document.createElement("div");  
// Create the new element  
let text = document.createTextNode("Hello there!");  
// Create some text  
newElement.appendChild(text);  
// Add text to new element  
existingElement.appendChild(newElement);  
// Add that element to the DOM
```

Removing things from the DOM

```
let elementToRemove = document.querySelectorAll("li")[2];  
elementToRemove.remove();
```

.innerHTML and .innerText

You can directly manipulate the DOM using `.innerHTML`, but you can break it easily.

```
let element = document.getElementById("product-title");  
element.innerHTML = "<h2>Boomerang</h2>";
```

Template literals

If you're using `.innerHTML` often, using *template literals* can help. These allow you to interpolate code in your string.

```
let name = "Amani";  
let content = `

## Hello, ${name}!</h2>`;


```

JavaScript events

User interactions create events on the page. JavaScript can handle these events: detect when something happens and do something in response, including changing the DOM or triggering other events.

Some Types of Events

- when the page loads
- a mouse click or movement
- a tap on a touchscreen
- when a key is pressed
- when something is selected

Event handling

Let's say you want a certain function to run when an image on the page is clicked.

- Select the DOM node for the element that will be clicked (the image).
- Choose the event that will act as the trigger for the function (the mouse click).
- Specify the function that you wish to run when the event occurs.

Event listeners

```
let button = document.querySelector(".cancelButton");  
button.addEventListener("click", function() {  
    alert("Are you sure you want to cancel?");  
});
```

Here we are passing two arguments to the event listener method:

- the event you want to respond to¹
- the *callback function* to run when that event happens

¹There are a lot of events you could listen for. Consult the [MDN Event Reference](#) for their names.