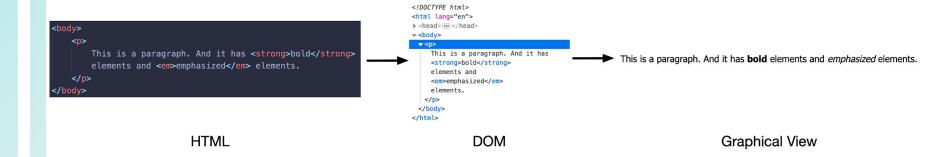
Module 3 - Lecture 8

Document Object Model



The DOM

- An in-memory representation of a web page's structure.
- Generated by the browser from an HTML document.





The DOM

- The DOM may be different than your HTML.
- CSS uses the DOM.

```
Tech Elevator
7100 Euclid Ave.
```



DOM Manipulation



Finding element(s)

getElementById(String);

- Finds an element by its unique identifier. Faster than querySelector.
- Returns an Element. Returns null if not found.
 let element = document.getElementById('someId');

querySelector(String);

- Selects the first descendant element that matches the selector. Selectors are CSS selectors.
- Returns an Element. Returns null if not found.
 let element = document.querySelector('ul > li');

querySelectorAll(String);

- Selects all descendant elements that match the selector.
- Returns a NodeList.
 let nodeList = document.querySelectorAll('ul > li');



Changing Elements

Changing how an element renders in the browser requires manipulating the element. This will update the DOM and force the browser to re-render.

innerText

- The text inside of an element.
- All text (including HTML tags) is replaced.
- All text is treated a string literal. HTML is not interpreted.

- innerHTML

- The HTML inside of an HTML.
- All text (including HTML tags) is replaced.
- HTML is interpreted.
- Do not use with user input!

Getting and Setting input values

Most input elements have a property **value** that contains the current value of the element. Checkboxes have a property **checked**.

- Get the value of a textbox named "toDo"

```
let toDoInput = document.querySelector('input[name=toDo]');
let val = toDoInput.value;
```

Set the value of a textbox

```
toDoInput.value = 'Wash the car';
```

Check a checkbox if it isn't checked.

```
let isFinished = document.querySelector('input[type=checkbox]');
if (isFinished.checked) {
    isFinished.checked = true;
}
```

Manipulating Classes

Elements have a property classList that is a collection of its classes.

 Recall that HTML tags can have multiple classes listed in class attribute. The element below has 2 classes.

```
        Some text here...
```

The **classList** property is a **DOMTokenList**.

You may add a class using the add() method.

```
const mainParagraph = document.querySelector('.main-content');
mainParagraph.classList.add('new-class');
```

- You may remove a class using the **remove()** method.

```
const mainParagraph = document.querySelector('.main-content');
mainParagraph.classList.remove('new-class');
```

Adding to the DOM

- document.createElement(String)
 - Creates a new Element.
 - It is not in the DOM at this point.

```
const newDiv = document.createElement('div');
newDiv.setAttribute('id', 'myNewDiv');
newDiv.classList.add('container');
```

- targetElement.appendChild(Element)
 - Adds *Element* as the last child of *targetElement*.
 - Element is now in the DOM and will be rendered.

```
const sectionElement = document.querySelector('section');
sectionElement.appendChild(newDiv);
```

Adding to the DOM

- targetElement.insertAdjacentElement(position, Element)
 - Adds *Element* at the specified position.
 - position can be one of the following:
 - 'beforebegin' Before the *targetElement*
 - 'afterbegin' Inside of the targetElement before its first child.
 - 'beforeend' Inside of the *targetElement* after its last child.
 - 'afterend' After the *targetElement*

Removing from the DOM

- targetElement.removeChild(Node)
 - Removes a child node from the current element

- targetElement.remove()
 - Removes the element from the tree it belongs to

Traversing the DOM

- targetElement.children returns child elements.
 - Returns an <u>HTMLCollection</u>.
 - Only includes Elements.
- targetElement.childNodes returns all nodes/elements inside.
 - Returns a NodeList.
 - Includes Elements and text content.

```
  This is an <strong>awesome</strong> paragraph.
  <!-- with a comment -->
```

Traversing the DOM

- targetElement.parentNode
 - Returns the parent Node.

- targetElement.nextElementSibling
 targetElement.previousElementSibling
 - Returns the sibling of *targetElement*.
 - Returns null if *targetElement* doesn't have a sibling.

QUESTIONS?

