IN4MATX 285:
Interactive Technology Studio

Programming: DOM Manipulation

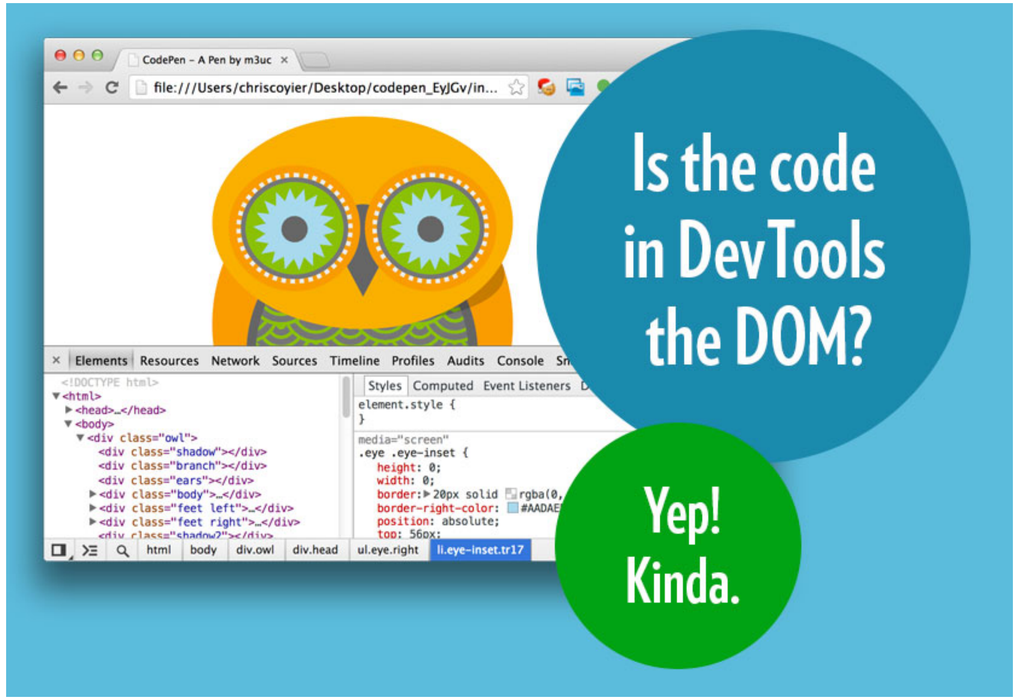
Today’s goals

By the end of today, you should be able to…

- Explain the role of the Document Object Model (DOM)
- Write code which reads the DOM using JavaScript
- Write code which edits the DOM using JavaScript

HTML Document Object Model (DOM)

- A standard for how to get, change, add, or delete HTML elements
- The HTML you write is parsed by the browser and turned into the DOM
- JavaScript can then edit the DOM



https://css-tricks.com/dom/

Live Demo: editing the DOM in your browser

You can also read and edit the DOM with JavaScript

Accessing the DOM in JavaScript

- document object model
- For example, you can write into the DOM with `document.write`
  - Appends to the end of the webpage

```
document.write("<h1>Hello, World!</h1>");

let myCourse = "IN4MATX 133";
document.write("<h1>Hello, " + myCourse + "!");
```

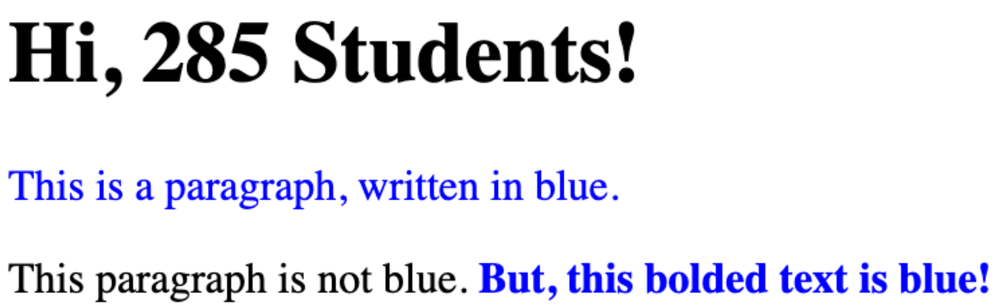
Example webpage

HTML

```
<h1 id="title">Hi, 285 Students!</h1>
<p class="blue">This is a paragraph, written in blue.</p>
<p id="notblue">This paragraph is not blue.
    <strong class="blue">But, this bolded text is blue!</strong>
</p>
```

CSS

```
.blue {
    color: blue;
}

.arial {
    font-family: Arial;
}
```



GetElementById

- Returns the one element with a matching id

```
let h1Title = document.getElementById('title');
console.log(h1Title);
let notbluePara = document.getElementById('notblue');
console.log(notbluePara);
```

```
<h1 id="title">...</h1>
<p class="blue">...</p>
<p id="notblue">...
<strong class="blue">...</strong>
</p>
```

```
<h1 id="title">Hi, 285 Students!</h1>
```

```
<p id="notblue">…</p>
```

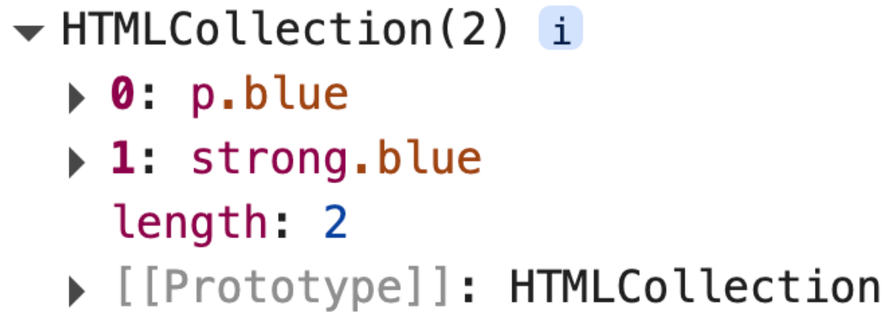
GetElementByClassName

- Returns **all** elements with a CSS class

- Returns an HTMLCollection, which is an array of HTML Elements

```
let blueTags = document.getElementsByClassName('blue');
console.log(blueTags);
```

```
<h1 id="title">...</h1>
<p class="blue">...</p>
<p id="notblue">...
<strong class="blue">...</strong>
</p>
```



GetElementByTagName

- Returns **all** elements with a particular tag

- Returns an HTMLCollection, which is an array of HTML Elements

```
let pTags = document.getElementsByTagName('p');
console.log(pTags);
```

```
<h1 id="title">...</h1>
<p class="blue">...</p>
<p id="notblue">...
<strong class="blue">...</strong>
</p>
```

```
▼ HTMLCollection(2) [p.blue, p#notblue
  ▶ 0: p.blue
  ▶ 1: p#notblue
  ▶ notblue: p#notblue
    length: 2
  ▶ [[Prototype]]: HTMLCollection
```

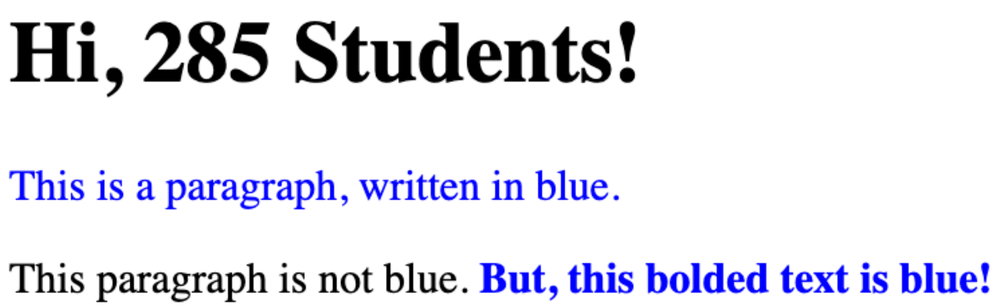
Selecting elements

- We can reference individual HTML elements by calling selector functions

```
/*easiest to select by reusing CSS selectors! */
let cssSelector = 'header p, .title > p';

//selects FIRST element that matches css selector
let elem = document.querySelector(cssSelector);

//matches ALL elements that match css selector
let elems = document.querySelectorAll(cssSelector);
```

Editing elements

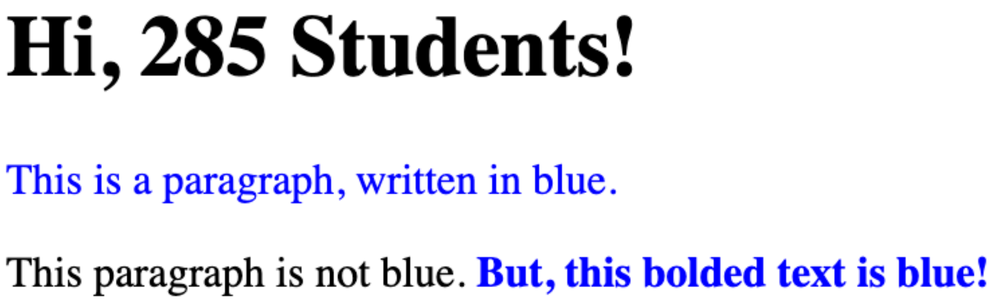
```
let titleTag = document.getElementById('title');
titleTag.textContent = 'This is a new title!';

let notBluePara = document.getElementById('notblue');
notBluePara.innerHTML = 'And <em>this</em> is new HTML.';
```

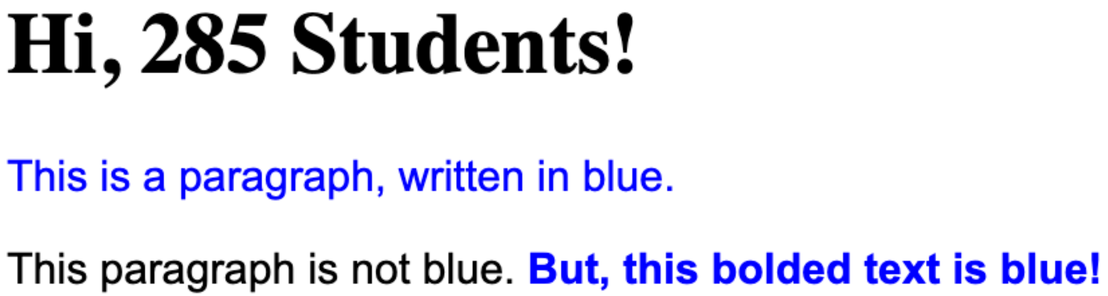


Editing elements

- Can add/remove classes

```
let paraTags = document.getElementsByTagName('p');
for(let para of paraTags) {
    para.classList.add('arial');
}
```





Adding listeners

- In HTML, you can specify a function to get called in response to a click action

```
<button onclick="myFunction()">Click me</button>
```

- Will look for a function with the name `myFunction` in your JavaScript

https://www.w3schools.com/jsref/event_onclick.asp

Adding listeners

- Listeners can also be added in code
- Can indicate a method which should be called when that element causes an event

```
//respond to "click" events
elem.addEventListener('click', callback);
```

Reflecting on DOM Manipulation

Reflecting on DOM Manipulation

- There's **so** much that you can do via DOM Manipulation
  - Create new HTML elements, remove existing elements
  - Give elements new ids or other attributes
  - Edit or add to CSS Stylesheets
- Basically every interactive webpage leverages DOM Manipulation
  - Check your favorite webpage to see how it changes as you interact with it

Today’s goals

By the end of today, you should be able to…

- Explain the role of the Document Object Model (DOM)
- Write code which reads the DOM using JavaScript
- Write code which edits the DOM using JavaScript

IN4MATX 285:
Interactive Technology Studio

Programming: DOM Manipulation

Additional Slides

Document ready: JavaScript

- The DOM won't be visible until the page has finished loading
  - Otherwise elements you're trying to access might not exist
  - `onload` function will run after the page has loaded

```
<head>
  <script>
    function pageLoaded() {
      alert("Page Loaded!");
    }
  </script>
</head>
<body onload="pageLoaded()">
</body>
```

Event types

```
'click'      //mouse or button clicked

'dblclick'   //double-clicked

'hover'      //mouse hover

'focus'      //element gains focus (important!)

'mouseenter' //mouse is moved over an element

'mouseleave'  //mouse leaves the element

'mousedown' //mouse button is pressed

'keydown'    //key is pressed

//... and more!
```

https://developer.mozilla.org/en-US/docs/Web/Events

Listeners

- Listener will be passed an **event** parameter
  - Sometimes useful, but can often be ignored

```
elem.addEventListener('click', function(event) {
    console.log('You clicked me!');

    //get what was clicked;
    Let clickedElem = event.target;
});
```

Remember, parameters are optional

The “target” (source) of the event

Adding new elements

- Use `createElement`, pass in the tag to create

```
let accordionItem = document.createElement('div');
accordionItem.className = 'accordion-item';
//Create a header element for the course
let accordionHeader = document.createElement('h2');
accordionItem.appendChild(accordionHeader);
accordionHeader.className = 'accordion-header';
```

- Needs to be added somewhere to your page, relative to another element
  - Typically added as a child (nested) with `appendChild`

https://developer.mozilla.org/en-US/docs/Web/API/Document/createElement
https://developer.mozilla.org/en-US/docs/Web/API/Node/appendChild