**Intro to JavaScript: Modifying and creating HTML DOM elements with JavaScript**

**Browser Inspector**
Tool to inspect underlying HTML/CSS of an HTML page as well as view the console. Works for Chrome and Firefox.

**Windows/Linux**
Ctrl + Shift + I

**Mac**
Command + Shift + I

**Including JS**
To include a JavaScript file on an HTML page, include with HTML script tags referencing the source location. Include after body content but within HTML tags.

```html
<script type="text/javascript" src="/PATH/TO/SCRIPT.js"></script>
```

**Declaring Functions**
Functions can be declared with a name and accept parameters. Anonymous functions can also be declared as callbacks.

```javascript
const add = (x, y) => {
  return (x + y);
}

functionWithCallback(() => {
  return "anonymous";
});
```

An anonymous function declared as a callback for another function.

**JavaScript Objects**
Otherwise known as a JSON object, these objects have attributes with values that can be assigned.

```javascript
let user = {
  name: "Aaron Sipser"
};
```

```javascript
console.log(user.name);
```

Attributes of a JSON object can be accessed by dot notation.

```javascript
console.log(user["name"]);```

Attributes of a JSON object can be accessed with brackets.

**Console Log**
Print things to console to help debug. Access console on web page by opening inspector in a browser.

```javascript
console.log("hello world");
```

**Declaring Variables**
Initialize a new variable. Use let for variables you may reassign. Constants should never be reassigned.

```javascript
let x;
```

Declares the variable x without assigning.

```javascript
let y = "hello";
```

Declares the variable y and assigns to a string.

```javascript
const z = "world";
```

Declares the constant z and assigns to a string.

**Selecting DOM Elements**
Select existing DOM elements on the HTML file the script is included in by element id.

```javascript
let DOM = document.getElementById("elementId");
```
Select/Modify DOM Element Attributes
Select DOM element’s attributes such as inner text and classes. Some of the following examples refer to a variable DOM which refers to a DOM element.

```javascript
DOM.innerText = "welcome home, boss";
```
This changes the text within the element's tags. Only accepts strings.

```javascript
DOM.innerHTML = "<p>welcome home, boss</p>";
```
This changes the HTML between the element's tags. Will accept and parse HTML tags.

```javascript
DOM.className = "class-one class-two";
```
Sets and overwrites the class of the element to classes: "class-one" and "class-two".

```javascript
DOM.id = "element-id";
```
Sets the id of the element to "element-id".

```javascript
DOM.setAttribute("id", "element-id");
```
Sets the id of the element to "element-id".

Tips
Different DOM elements have different attributes. It’s highly recommended that you look up the attributes of the element you are trying to access.

For example, input tags have value attributes and not innerText.

Create DOM Element
Create a new DOM element with JavaScript. Don’t forget to add the element to the page.

```javascript
let DOM = document.createElement("div");
```
Creates a new <div></div> element.

Add DOM Element to Page
Created DOM elements must be explicitly added to a page.

```javascript
let parent = document.getElementById("parent");
let DOM = document.createElement("div");
parent.appendChild(DOM);
```
Selects the element with the id "parent". Creates a new <div></div> element and adds it to the parent element.

Event Listeners
Add event listeners to HTML DOM elements.

```javascript
DOM.addEventListener("click", function() {
    console.log("clicked!");
});
```
Here DOM is a variable that refers to a DOM element and an anonymous function is triggered when the element is clicked.