CSCI E-33a

CS50's Web Programming
with Python and JavaScript

Spring 2020

Barbara Karakyriakou · Teaching Fellow

- Email: karakyriakou@fas.harvard.edu
- Phone: 617 820 6930

- Section Meetings: Wednesdays 8:30pm-10:00pm EST
- Office Hours: Saturdays 1:00pm - 2:30pm EST

Section 4: JavaScript

Agenda

- JavaScript
  - Declarations
  - Comments
  - Functions
  - DOM Manipulation
  - Event Handling
- Problem Set 3 Instructions & Hints

JavaScript

JavaScript is a lightweight, interpreted, object-oriented language, and it is best known as the scripting language for Web pages.

Recommended for JavaScript practice: https://jsfiddle.net/

Declarations

JavaScript has three kinds of variable declarations.

- var: Declares a function scope variable
- let: Declares a block-scoped, local variable
- const: Declares a block-scoped, read-only named constant

Comparison Table

	var	let	const
reassigned	O	O	X
Scope	Function Scope	Block Scope	Block Scope
Reference before declaration	O	X	X

Declarations

The use of let is recommended over var, because redeclaring a variable using the var keyword can impose problems.

For example, redeclaring a variable inside a block will also redeclare the variable outside the block:

```
var x = 5;
{
  var x = 10;
}
// x is 10
```

```
let x = 5;
{
  let x = 10;
}
// x is 5
```

Comments

Single line comments start with //.

```
// a single line comment
```

Multi-line comments start with /* and end with */.

```
/* a multi-line comment */
```

Functions

A JavaScript function is defined with the function keyword, followed by a name, followed by parentheses ().

```
function myFunction(parameter1, parameter2) {
    return parameter1 * parameter2
}
```

Functions

A JavaScript function can be anonymous:

```
const square = function(p1) {
  return p1*p1
}
```

Functions

A JavaScript function can be used as a variable.

```
let text = "The temperature is " + Celsius(68) + " Celsius”;
```

HTML

```
<p id="demo"></p>
```

CSS

JavaScript + No-Library (pure JS) Tidy

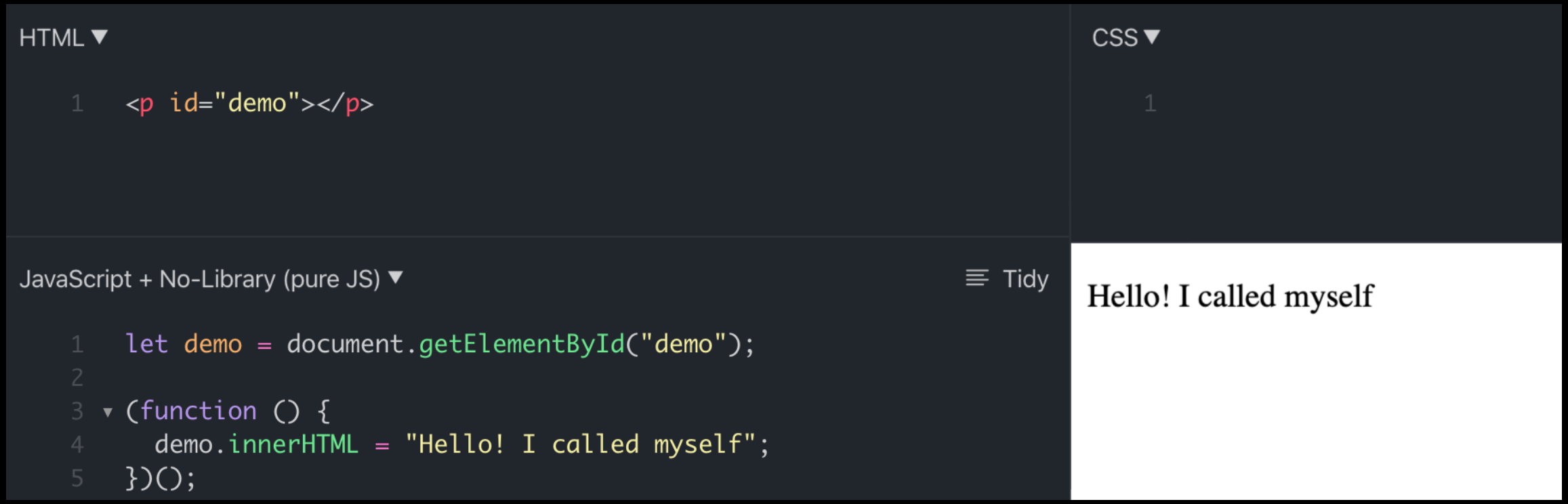
```
let demo = document.getElementById("demo");

demo.innerHTML = "The temperature is " + Celsius(68) + " Celsius";

function Celsius(fahrenheit) {
  return (5/9) * (fahrenheit-32);
}
```

The temperature is 20 Celsius

Functions

A JavaScript function can be self invoked. You have to add parentheses around the function to indicate that it is a function expression.

```
(function () {
  demo.innerHTML = "Hello! I called myself";
})();
```



DOM (Document Object Model) Manipulation

Method:

The most common way to access an HTML element is to use the id of that element.

In my previous examples the getElementById method used id="demo" to find the appropriate HTML element.

Property:

To get the content of an element use the innerHTML property.

The innerHTML property can be used to get or change/replace any HTML element.

DOM (Document Object Model) Manipulation

Other methods to get HTML elements:

- `document.getElementsByTagName("p");`
- `document.getElementsByClassName(”myClass");`
- `document.querySelectorAll(”p.myClass");`

Other properties:

- `.Attribute:` the atribute value of an HTML element
- `.style.Property:` the style property of an HTML element

Event Handling

Events

A script (JavaScript) can be executed when an HTML event occurs.

HTML event examples:

- When a user clicks the mouse
- When the mouse hovers over an element
- When an input field is changed
- When an HTML form is submitted
- A web page was loaded

Event Handling

Event Listener Method

The `addEventListener()` method attaches an event handler to the specified element. It takes two arguments: the event and the function.

Examples:

- `document.getElementById("myButton").addEventListener("click", myFunction);`
- `element.addEventListener("mouseover", mySecondFunction);`
- `element.addEventListener("mouseout", myThirdFunction);`

Project 3

Mail

- Download the project zip folder and unzip it
- Move the project to a location that you have easy access through your terminal
- Once you open the folder you will find two subfolders, mail, project3, and a manage.py file
- Run `python(3) manage.py makemigrations mail` to make migrations for the mail app.
- Run `python(3) manage.py migrate` to apply migrations to your database.

Project 3

Next steps:

1. Check your templates

- inbox.html: need to add a division for viewing emails.
  - From, To, Subject, Timestamp, Body
  - Buttons: Reply, Archive, Unarchive

Project 3

Next steps:

2. Check Static

- styles.css: it is recommended that you to add some style
- inbox.js: your JavaScript code goes here

Project 3

inbox.js - what is needed:

- Anonymous function:
  - When form is submitted, send a new email
  - Allow user to mark an email as archived or unarchived
  - Handle when user replies to an email
- load_mailbox function:
  - You need to add a query for the latest emails
  - You will find the fetch() method useful here. Remember the syntax:

```
fetch(`…`).then(response => response.json()).then(emails => {…});
```

Project 3

inbox.js - what is needed:

- New Functions
  - Add email(s) to the mailbox
  - Archive an email
  - Reply to an email
  - Send an email

Q&A