Cubstart Web Lecture 3

[start recording]

Administrivia

Reminder on attendance policy: lecture & first hour of lab are required

You are allowed up to 4 total unexcused absences and 2 homework drops

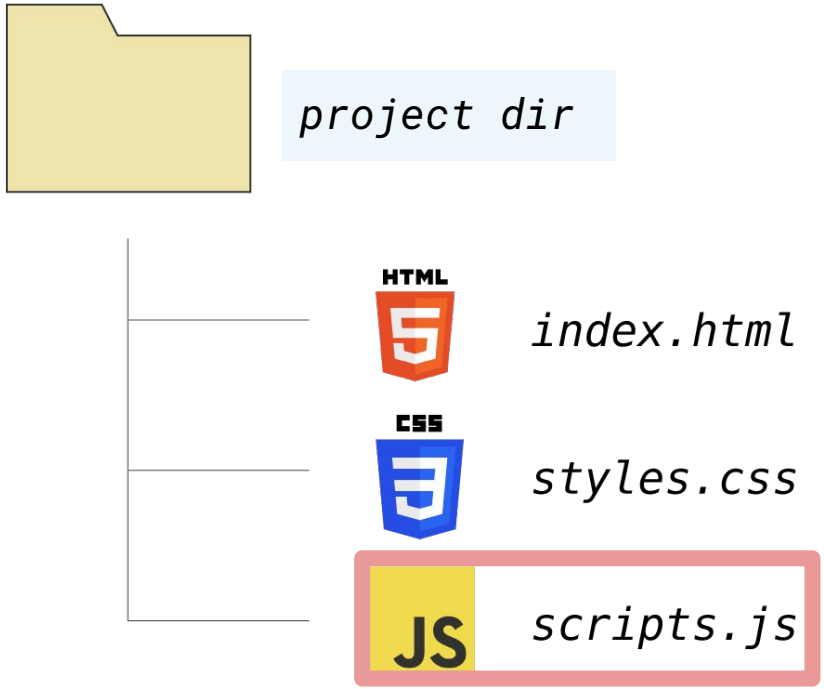
Excused Absences Form can be found on Ed

Administrivia

- HW2 is due this Friday Sept 29th at midnight
- Lab: Friday @ Physics 3 from 4-6PM
  - HW2 walkthrough at the end of lab!
- HW3 will be posted in the next few days
- Today might be a bit slow for those with programming experience, hang tight until the second half!

Recap

Webpage Filesystem



HTML

Search Images **Gmail** Drive Calendar Sites Groups Contacts More »

brandonwang0503@berkeley.edu | Google Account | Settings | Help | Sign out

bConnected powered by Google

cs 170 on piazza | Search Mail | Search the Web | Show search options | Create a filter

Compose Mail

Inbox
Starred
Sent Mail
Drafts
All Mail
Spam (136)
Trash

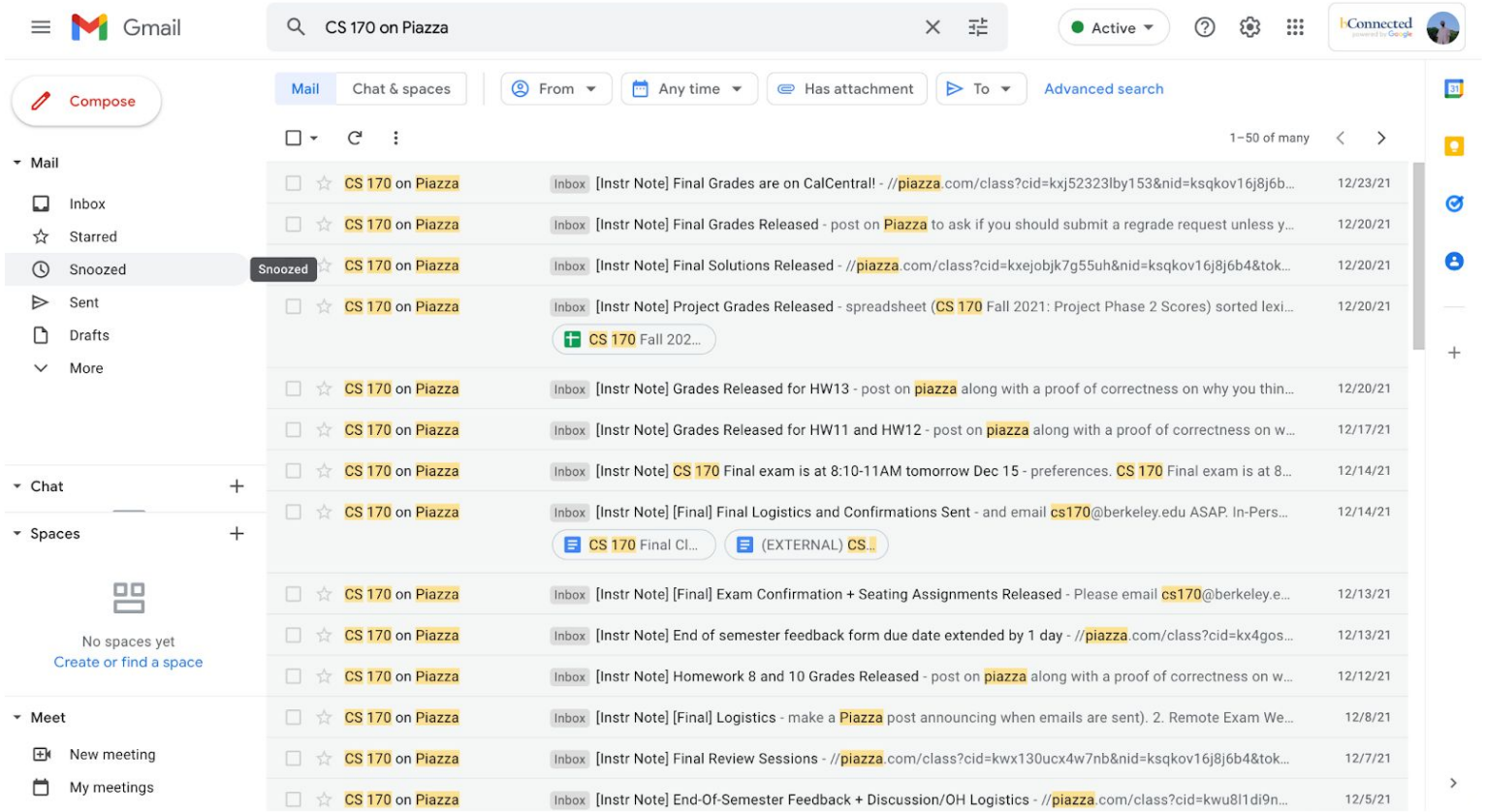
Contacts

Labels
Edit labels

Search results for: **cs 170 on piazza** | Report Spam | Delete | More Actions... | Go | Refresh | **1 - 50** of about **91** **Older ›**

☐	CS 170 on Piazza	Inbox	[Instr Note] Final Grades are on CalCentral! - //**piazza**.com/class?cid=kxj52323lby153&nid=ksqkov16j8j6b4&token=UwiTmvGAnMX to view. Search or link	12/23/21
☐	CS 170 on Piazza	Inbox	[Instr Note] Final Grades Released - post on **Piazza** to ask if you should submit a regrade request unless you have an alternate solution that the rubric	12/20/21
☐	CS 170 on Piazza	Inbox	[Instr Note] Final Solutions Released - //**piazza**.com/class?cid=kxejobjk7g55uh&nid=ksqkov16j8j6b4&token=UwiTmvGAnMX to view. Search or link	12/20/21
☐	CS 170 on Piazza	Inbox	[Instr Note] Project Grades Released - spreadsheet (**CS 170** Fall 2021: Project Phase 2 Scores) sorted lexicographically by team name. Please make	12/20/21
☐	CS 170 on Piazza	Inbox	[Instr Note] Grades Released for HW13 - post on **piazza** along with a proof of correctness on why you think your solution is correct. This will allow for	12/20/21
☐	CS 170 on Piazza	Inbox	[Instr Note] Grades Released for HW11 and HW12 - post on **piazza** along with a proof of correctness on why you think your solution is correct. This will	12/17/21
☐	CS 170 on Piazza	Inbox	[Instr Note] CS 170 Final exam is at 8:10-11AM tomorrow Dec 15 - preferences. **CS 170** Final exam is at 8:10-11AM tomorrow Dec 15 I did not modify the	12/14/21
☐	CS 170 on Piazza	Inbox	[Instr Note] [Final] Final Logistics and Confirmations Sent - and email **cs170**@berkeley.edu ASAP. In-Person: You should have received an email for your	12/14/21
☐	CS 170 on Piazza	Inbox	[Instr Note] [Final] Exam Confirmation + Seating Assignments Released - Please email **cs170**@berkeley.edu if you did not receive this information. Go to	12/13/21
☐	CS 170 on Piazza	Inbox	[Instr Note] End of semester feedback form due date extended by 1 day - //**piazza**.com/class?	12/13/21
☐	CS 170 on Piazza	Inbox	[Instr Note] Homework 8 and 10 Grades Released - post on **piazza** along with a proof of correctness on why you think your solution is correct. This will	12/12/21
☐	CS 170 on Piazza	Inbox	[Instr Note] [Final] Logistics - make a **Piazza** post announcing when emails are sent). 2. Remote Exam We will offer a remote option for taking the exam at	12/8/21
☐	CS 170 on Piazza	Inbox	[Instr Note] Final Review Sessions - //**piazza**.com/class?cid=kwx130ucx4w7nb&nid=ksqkov16j8j6b4&token=UwiTmvGAnMX to view. Search or link	12/7/21
☐	CS 170 on Piazza	Inbox	[Instr Note] End-Of-Semester Feedback + Discussion/OH Logistics - //**piazza**.com/class?cid=kwu8l1di9ngjj&nid=ksqkov16j8j6b4&token=UwiTmvGAnMX	12/5/21
☐	CS 170 on Piazza	Inbox	[Instr Note] Post MT2 Grading Distribution - //**piazza**.com/class?cid=kwosxxkxp2216k&nid=ksqkov16j8j6b4&token=UwiTmvGAnMX to view. Search or link	12/2/21
☐	CS 170 on Piazza	Inbox	[Instr Note] End-Of-Semester Feedback Form + EECS Dept Course Evals (+1 EC) - us improve **CS 170** for future semesters by filling out this anonymous	12/1/21
☐	CS 170 on Piazza	Inbox	[Instr Note] Project Party Friday 2-3 pm in Wheeler 224! - //**piazza**.com/class?cid=kwo6qbwkbqnj1&nid=ksqkov16j8j6b4&token=UwiTmvGAnMX to view.	12/1/21
☐	CS 170 on Piazza	Inbox	[Instr Note] Phase 2 Output Autograder Bug - //**piazza**.com/class?cid=kwk0xr5ydufih&nid=ksqkov16j8j6b4&token=UwiTmvGAnMX to view. Search or link	11/28/21
☐	CS 170 on Piazza	Inbox	[Instr Note] Project Phase 2: Autograder and Leaderboard Released - //**piazza**.com/class?	11/26/21
☐	CS 170 on Piazza	Inbox	[Instr Note] Project Phase 2: Inputs Released - //**piazza**.com/class?cid=kwfg6qkeekx519&nid=ksqkov16j8j6b4&token=UwiTmvGAnMX to view. Search or	11/25/21
☐	CS 170 on Piazza	Inbox	Final Project leaderboard - //**piazza**.com/class?cid=kwfaub5538t2p1&nid=ksqkov16j8j6b4&token=UwiTmvGAnMX to view. Search or link	11/25/21
☐	CS 170 on Piazza	Inbox	Bad Reductions Discussion - //**piazza**.com/class?cid=kwem82pqcfp3w&nid=ksqkov16j8j6b4&token=UwiTmvGAnMX to view. Search or link to	11/24/21
☐	CS 170 on Piazza	Inbox	Activity Digest since 12:34AM for CS 170 on Piazza - happened in **CS 170** on **Piazza**: Homework 13 (due during	11/24/21
☐	CS 170 on Piazza	Inbox	Activity Digest since 10:21PM for CS 170 on Piazza - happened in **CS 170** on **Piazza**: clustering (11/23/	11/23/21
☐	CS 170 on Piazza	Inbox	Activity Digest since 2:44PM for CS 170 on Piazza - happened in **CS 170** on **Piazza**: **CS 170** Office Hours (	11/23/21
☐	CS 170 on Piazza	Inbox	[Instr Note] Lecture Thread 11/23 - //**piazza**.com/class?cid=kwcgtmd0ngy65&nid=ksqkov16j8j6b4&token=UwiTmvGAnMX to view. Search or link to	11/23/21
☐	CS 170 on Piazza	Inbox	Activity Digest since 9:36PM for CS 170 on Piazza - happened in **CS 170** on **Piazza**: Final Exam Distributions	11/23/21
☐	CS 170 on Piazza	Inbox	Are Office Hours available during Thanksgiving break and RRR week? - //**piazza**.com/class?	11/22/21

CSS


Gmail
CS 170 on Piazza
Active
Compose
Mail
Chat & spaces
From
Any time
Has attachment
To
Advanced search
1–50 of many
Mail
Inbox
Starred
Snoozed
Sent
Drafts
More
Snoozed
Chat
Spaces
No spaces yet
Create or find a space
Meet
New meeting
My meetings
CS 170 on Piazza Inbox [Instr Note] Final Grades are on CalCentral! - //piazza.com/class?cid=kxj52323lby153&nid=ksqkov16j8j6b... 12/23/21
CS 170 on Piazza Inbox [Instr Note] Final Grades Released - post on Piazza to ask if you should submit a regrade request unless y... 12/20/21
CS 170 on Piazza Inbox [Instr Note] Final Solutions Released - //piazza.com/class?cid=kxejobjk7g55uh&nid=ksqkov16j8j6b4&tok... 12/20/21
CS 170 on Piazza Inbox [Instr Note] Project Grades Released - spreadsheet (CS 170 Fall 2021: Project Phase 2 Scores) sorted lexi... 12/20/21
CS 170 Fall 202...
CS 170 on Piazza Inbox [Instr Note] Grades Released for HW13 - post on piazza along with a proof of correctness on why you thin... 12/20/21
CS 170 on Piazza Inbox [Instr Note] Grades Released for HW11 and HW12 - post on piazza along with a proof of correctness on w... 12/17/21
CS 170 on Piazza Inbox [Instr Note] CS 170 Final exam is at 8:10-11AM tomorrow Dec 15 - preferences. CS 170 Final exam is at 8... 12/14/21
CS 170 on Piazza Inbox [Instr Note] [Final] Final Logistics and Confirmations Sent - and email cs170@berkeley.edu ASAP. In-Pers... 12/14/21
CS 170 Final Cl...
(EXTERNAL) CS...
CS 170 on Piazza Inbox [Instr Note] [Final] Exam Confirmation + Seating Assignments Released - Please email cs170@berkeley.e... 12/13/21
CS 170 on Piazza Inbox [Instr Note] End of semester feedback form due date extended by 1 day - //piazza.com/class?cid=kx4gos... 12/13/21
CS 170 on Piazza Inbox [Instr Note] Homework 8 and 10 Grades Released - post on piazza along with a proof of correctness on w... 12/12/21
CS 170 on Piazza Inbox [Instr Note] [Final] Logistics - make a Piazza post announcing when emails are sent). 2. Remote Exam We... 12/8/21
CS 170 on Piazza Inbox [Instr Note] Final Review Sessions - //piazza.com/class?cid=kwx130ucx4w7nb&nid=ksqkov16j8j6b4&tok... 12/7/21
CS 170 on Piazza Inbox [Instr Note] End-Of-Semester Feedback + Discussion/OH Logistics - //piazza.com/class?cid=kwu8l1di9n... 12/5/21

JavaScript

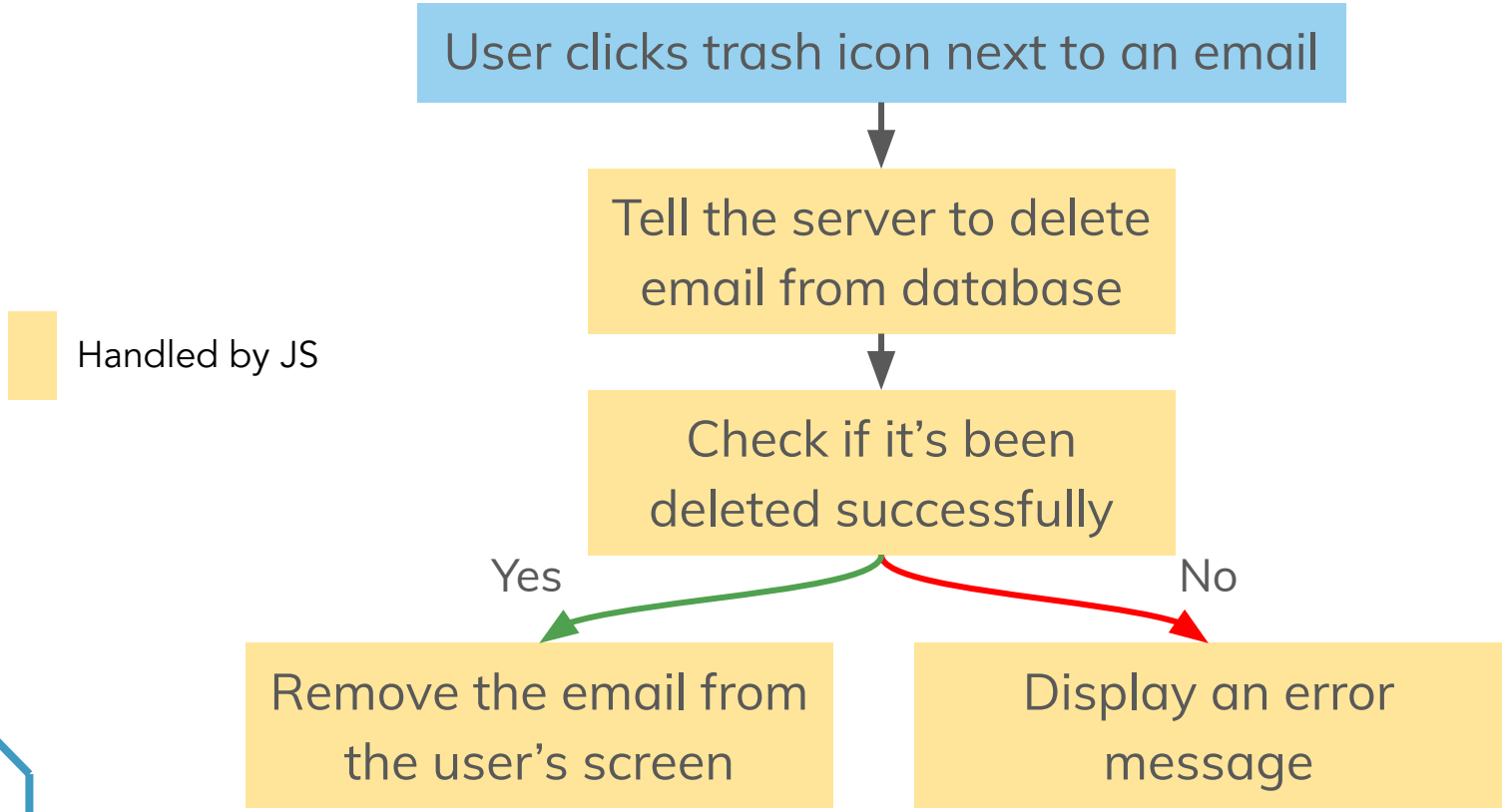
Server:

- How did all the emails load?
- How does Gmail know who I am?
- How am I shown as active when I’m online?

Client:

- How do I switch between inboxes?
- How do I initiate a search through the search bar?
- How can I delete an email?

Simple Example: Email Deletion



JavaScript (JS) Crash Course

What is JS?

A scripting language

Interpreted (run) by browsers

Allows you to modify the DOM dynamically

Where to put JS Code?

- Just like CSS, we want to put our JS in a separate file (.js)
- Put this tag at the **bottom** of your HTML file

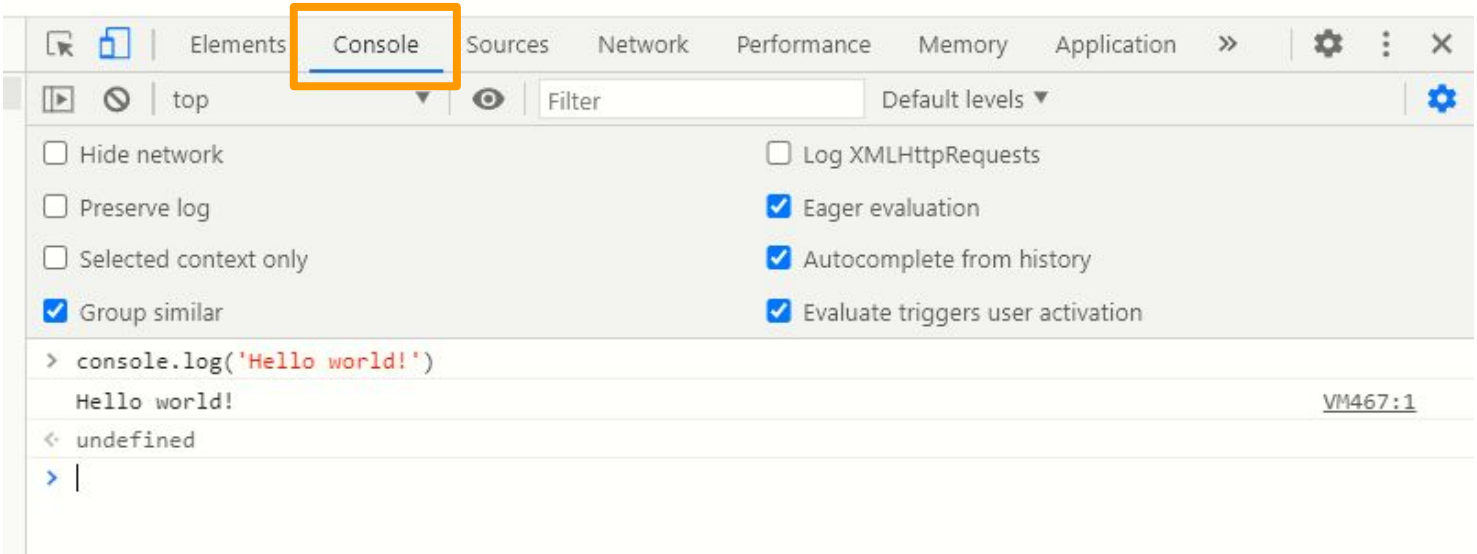
```
<script src="filename.js"></script>
```

- Why at the bottom? Your JS might rely on your HTML elements existing. If JS is at top, browser will run it before the rest of your HTML is rendered, causing errors.

Running JavaScript

All browsers are JavaScript interpreters!

- Inspect Element -> Console



Printing to the console

In your JS code!

`console.log(value)`

```
> console.log('Hello world!')
  Hello world!
< undefined
>
```

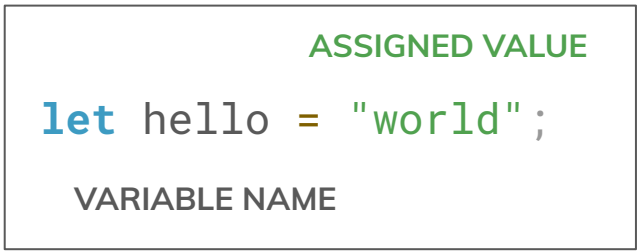
`console.log(value1, value2, ...)`

```
> console.log('Hello', 'world!', 1, 2, 3)
  Hello world! 1 2 3
< undefined
>
```

Declaring Variables

Here are a couple ways to **declare** variables and store values with unique names for use:

- `let hello = "world";`
- `const world = 100;`
- `var random = false; // Don't use this!`



let vs. const vs. var

`let`

- block scoped
- can be reassigned

`const`

- block scoped
- cannot be reassigned

`var`

- globally/function scoped
  - Accessible within the entire function body
- can be reassigned

let:

Block scoped
Reassignable

```
let greeting = "say Hi";
greeting = "say Hello instead";
```

```
let greeting = "say Hi";
let times = 4;

if (times > 3) {
     let hello = "say Hello instead";
     console.log(hello);// "say Hello instead"
 }
console.log(hello) // hello is not defined
```

Block (enclosed by curly brackets)

const:

Block scoped
Not reassignable

```
const greeting = "say Hi";
greeting = "say Hello instead";// error: Assignment to constant variable.
```

Check-in: Declaring variables

What would you fix/change about the following code?

```
const a = 3
a = a + 4
var b = 10
console.log(a + b)
```

ANSWER:

```
let a = 3
a = a + 4
let/const b = 10
console.log(a + b)
```

Primitive types in JS

- **number:** `1, 200, 1.8, 0.5, -1000, 10e5, ...`
- **string:** `"hello", ‘world’, ...`
- **boolean:** `true, false`
- **null** `//empty or blank value`
- **undefined** `//variable has been declared, but not defined`

Everything else is an “Object”, including arrays and functions!

Another type: `arrays/lists`

Lists can be used to store multiple values.

For example:

```
let lst = ["go", "bears", ":)"];
```

Types in a list can be different:

```
let vals = [true, “hi”, 3.9];
```

Some useful operations you can do on arrays.

Getting values from lists

```
console.log(lst[1]);

// Output: bears
```

Setting values in lists

```
lst[0] = "hi!"

console.log(lst[0]);

// Output: hi!
```

Arrays/lists

```
let lst = ["go", "bears", ":)"];
```

```
lst.push(1);

console.log(lst);

// Output: [“go”, “bears”, “:)”, 1]
```

```
lst.pop();

console.log(lst);

// Output: [“go”, “bears”, “:)”]
```

Arrays/lists

```
let lst = ["go", "bears", ":)"];
```

```
lst.sort();

console.log(lst);

// Output: [“:)”, “bears”, “go”]
```

```
console.log(lst.includes(“hello”));;

// Output: false
```

Another type: `objects`

Objects are basically dictionaries/maps. They have a set of “keys”, which each correspond to a “value”. Values can be anything: primitives, functions, lists, even other objects. Keys can only be strings or symbols.

For example:

```
let obj = { a: 'go', b: 'bears' };
```

VALUE VALUE

KEY KEY

```
console.log(obj.a); // go

obj.b = 'fish'

console.log(obj.b); // fish
```

Some useful operations you can do on objects.

Check-in: Types

What **types** are the following?

1. `'true'`
2. `false`
3. `5.8`
4. `'6'`
5. `['cub', 8]`
6. `{ name: 'terrance', food: 'taco' }`

ANSWER:

1. String
2. Boolean
3. Number
4. String
5. Array/Object
6. Object

Arguments = data that you pass into a function

```
function functionName(arg1,...) {
    // Do something...
    return returnValue;
}
```

```
function magic(num) {
    return num * 2 + 10;
}

console.log(magic(5)); // 20
console.log(magic(28)); // 66
console.log(magic(14)); // 38
```

Functions

Functions let you reuse a chunk of code without having to write the code again.

They help reduce repetitions in your code.

Basic Math

Some basic math operations

- General: `+, -, *, /`
- Power: `base ** exponent`
- Modulo: `dividend % divisor`
    - Example: `7 % 3 === 1` (Floor div: `7 / 3 === 2`)

Basic Strings

`split`: splits the string into an array of substrings using the delimiter

`includes`: checks if string includes specific character(s)

`toUpperCase & toLowerCase`: converts the string to upper or lowercase

`slice`: returns the specified part of the string (substring)

`replace`: returns the same string, but with the specified replacements

`indexOf`: returns the index of the character(s) within the string and -1 if the character(s) don't exist in the string

`charAt`: returns the character of the string at a given index

For more: check out the MDN page on JS string methods

```
//example console.log()
console.log("here")

//example list
let list1 = ["a", "b", "c"]

//lists can hold different data types
let list2 = ["dog", 18, false]

//get values from list
console.log(list1[1])

//set values in list
list2[2] = true

//add an element to list
list1.push("d")

//remove the last element from list
list2.pop()
```

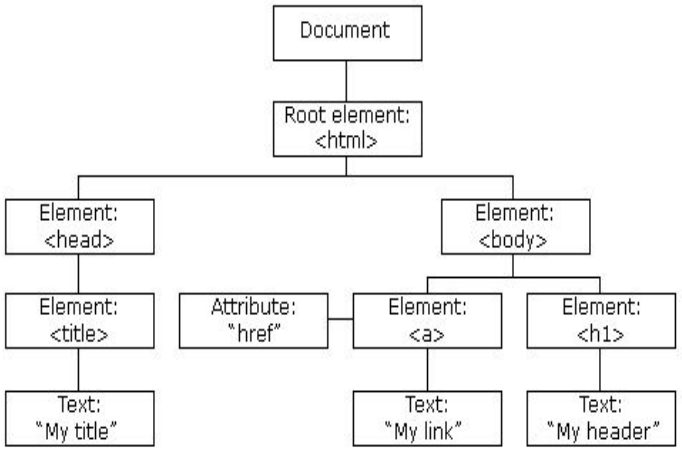
```
//example object
const person = {name: "Jessica", age: 19, student: true}

//get value from object by key
console.log(person.name)

//set value by key
person.age = 20

//example function
function magic(num1, num2) {
    return num1 + num2 - 100
}
console.log(magic(5, 13))

//example string manipulation
let s = "hello"
console.log(s.charAt(4))
console.log(s[4])
```

What is the DOM?

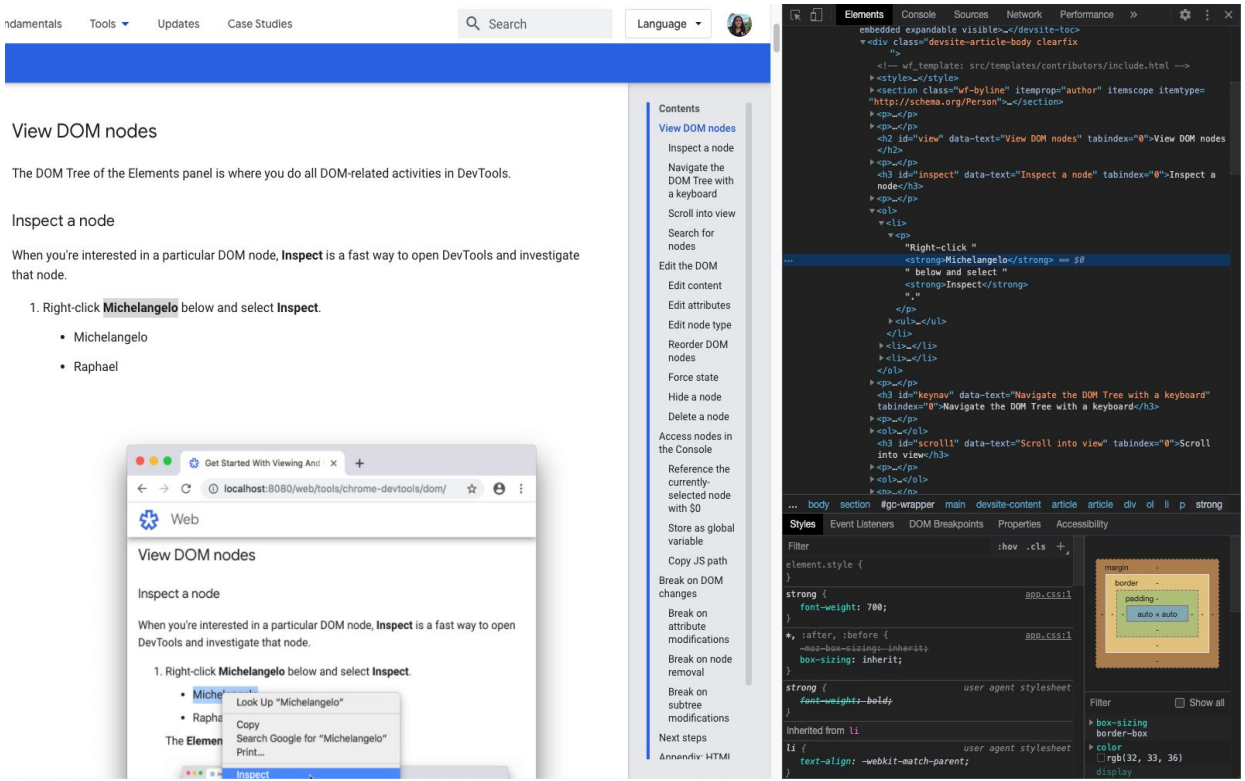
Representation of the webpage elements
Generated from HTML
Can be manipulated by JavaScript dynamically!

Document Object Model (DOM)

- How the browser “thinks” about elements on the webpage
- Browser generates DOM tree from HTML
- Hierarchy between parent and children (hence a tree)



Inspecting the DOM



Inspect Element!

Interacting with the DOM

Modifying DOM Elements

Adding, removing and replacing elements in the DOM

Here's the full MDN reference of the DOM API

Creating Elements

Create elements in the DOM using... `createElement`

```
document.createElement(tag);
```

For example, here's how to create a `div` element:

```
document.createElement('div');
```

Note: this returns an Element object in JS, but doesn't attach it to the DOM yet...

Accessing Elements

Access elements in the DOM by using `getElementById`

```
document.getElementById(id);
```

Similarly:

- `getElementsByTagName`
- `getElementsByClassName`

Adding Elements

Add one DOM element as a child of another using `appendChild`

Syntax

```
parentElement.appendChild(childEl);
```

Now we can create and add our own elements to the page like this:

```
let button = document.createElement('button');

let box = document.getElementById('boxId');

box.appendChild(button);
```

Removing Elements

Remove an element from a parent element using `removeChild`

Syntax

```
parent.removeChild(child);
```

Replacing Elements

Replace an element inside a parent element using `replaceChild`

Syntax

```
parent.replaceChild(newChild, oldChild);
```

DOM Element Properties

Here are some useful properties of DOM elements:

- **`.className`** returns the class property of the element
  - `<span class="someClass"></span>`
- **`.id`** returns the id of the element
  - `<span id="someId"></span>`
- **`.textContent`** returns the text contained within the element
  - `<span>"Element text"</span>`
- **`.style`** returns the style property of the element
  - `<span style="color: red"></span>`
- **`.value`** returns the value of input elements
  - `<input value={someValue}>`

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Cubstart Lecture 2</title>
</head>
<body>
    <h1 id="title">Welcome to Cubstart Lecture 2: JavaScript!</h1>
    <div class="box">box1</div>
    <div class="box">box2</div>
    <script src="scripts.js"></script>
</body>
</html>
```

scripts.js

```
let titleEl = document.getElementsByTagName("h1")

titleEl.textContent = 'We have updated this!'

let newDivEl = document.createElement("div");
let boxEl = document.getElementsByClassName("box")

newDivEl.textContent = "hello world"
boxEl.appendChild(newDivEl);

let boxEls = document.getElementsByClassName("box")

```

Event Handlers

Event handlers are used to trigger **actions** (execute functions) when an **event** occurs on a **target** (element), e.g. when a user clicks a button.

Syntax

```
target.addEventListener('eventName', doAction);
```

```
let button = document.getElementById('btn');

function doAction() {

    console.log('Button clicked!');

}

button.addEventListener('click', doAction);
```

Check-in: Modifying the DOM

How do I add a paragraph element to the document body?

A: `document.body.appendElement(document.createChild('p'))`

B: `document.body.createElement(document.appendChild('p'))`

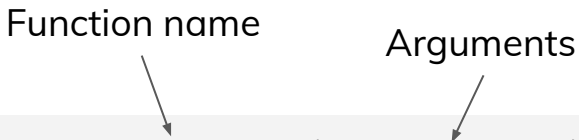
C: `document.body.appendChild(document.createElement('p'))`

D: `document.body.createChild(document.appendElement('p'))`

[IF THERE IS TIME]

Arrow Function Expression =>

```
const magic = (num) => {
    return (num * 2) + 10;
}

console.log(magic(5)); // 20
console.log(magic(28)); // 66
```




```
const magic = (num1, num2) => {
    return (num1 * num2) + 10;
}

console.log(magic(5, 5)); // 35
console.log(magic(28, 2)); // 66
```

[IF THERE IS TIME]

Arrow Functions in Use

```
let lst = [1, 5, 10, 15, 20];
```

```
const helper = (element) => {

    return element > 5;

};

console.log(lst.find(helper));

// Output: 10
```

```
console.log(lst.find((element) => { return elt > 5 }));

// Output: 10
```

Control Flow

Conditionals, loops, and more functions

```
if (condition) {
    // Do something...
} else {
    // Do something else...
}
```

```
if (condition1) {
    /* logic if condition1 is true */
} else if (condition2) {
    /* logic if condition1 is false
and condition2 is true */
} else {
    /* logic if condition1 and
    condition2 are false */
}
```

If Statements

`if, else if, else` statements

If statements allow you to execute code based on certain **conditions**

`if` runs the block if the condition is **truthy.**
Falsey values:

- `false`
- `0`
- `null`
- `undefined`
- `‘’`
- `NaN`

Everything else is truthy!

Conditionals: loose equality operators

== and != are **loose equality** operators (they don't compare the value types, only the values).

Some examples:

```
true == 1 // true

[] == 0 // true

6 == '6' // true

5 == 6 // false
```

=== and !== are **strict equality** operators (they compare the type of the object and it's value).

Some examples:

```
true === 1 // false

[] === 0 // false

6 === '6' // false

6 === 6 // true
```

Conditionals: && and | |

```
condition1 && condition2
```

&&: *and* operator evaluates first falsey value, or the last value if there are none

Example:

```
true && 1 // 1
```

```
0 && false // 0
```

```
5 < 6 && 4 + 5 == 1 // false
```

```
condition1 || condition2
```

| |: *or* operator evaluates to first truthy value, or the last value if there are none

Example:

```
true || false // true
```

```
5 == 6 || 1 + 1 == 2 // true
```

```
5 == 6 || 4 == 6 // false
```

Check-in: Conditionals

What would the following print?

```
let lst = [0, 1]
if (lst) {
    console.log('nice')
} else {
    console.log('oops')
}

console.log(lst[0] && lst[1])

console.log(lst[1] == '1')
```

While Loops

```
while (condition) {
    console.log("Condition true");
}
console.log("Condition false");
// Output
// Condition true
// Condition true
// ...
// Condition false
```

- While loops are used to iterate as long as a certain **condition is true**
- When the condition becomes false we exit the loop stop executing the code inside

For Loops

```
for (let i = 0; i < n; i++) {
    console.log("Num: " + i);
}
// Output
// Num: 0
// Num: 1
// ...
// Num: n
```

Really just a fancy while loop.

- Mostly used when you are iterating a certain **number of times**
- Examples...
  - Printing numbers from 0 to n
  - Iterating through every value in a list

Check-in: Loops

What would the following print?

```
let a = 1
while (a < 10) {
    a = a + 1
}
console.log(a)

for (let b = 0; b < 10; b++) {
    a += b
}
console.log(a)
```

[stop recording]

Attendance: Lecture 3

https://forms.gle/LAAZ28LAEzEcpfP59

cubstartattendance.web.app

Secret word: