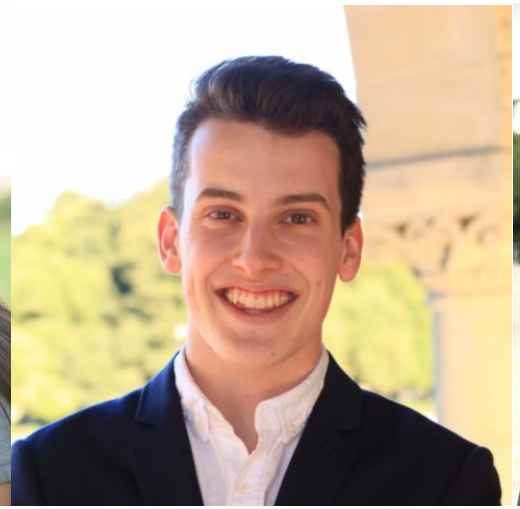
CS+Social Good

CS 106S Winter 2018

hi!!!!11!1!

we are so excited to meet you!

Ashi Agrawal
Stanford ‘19
ashia@stanford.edu

Ryan Cohen
Stanford ‘19
rrcohen@stanford.edu

Kevin Khieu
Stanford ‘18
kkhieu@stanford.edu

Matthew Sun
Stanford ‘20
mattsun@stanford.edu

why section?

well why not, HUH????

logistics

TIME
Thursdays, 4:30 – 6:20 PM

LOCATION
Littlefield 107

DURATION
10 weeks

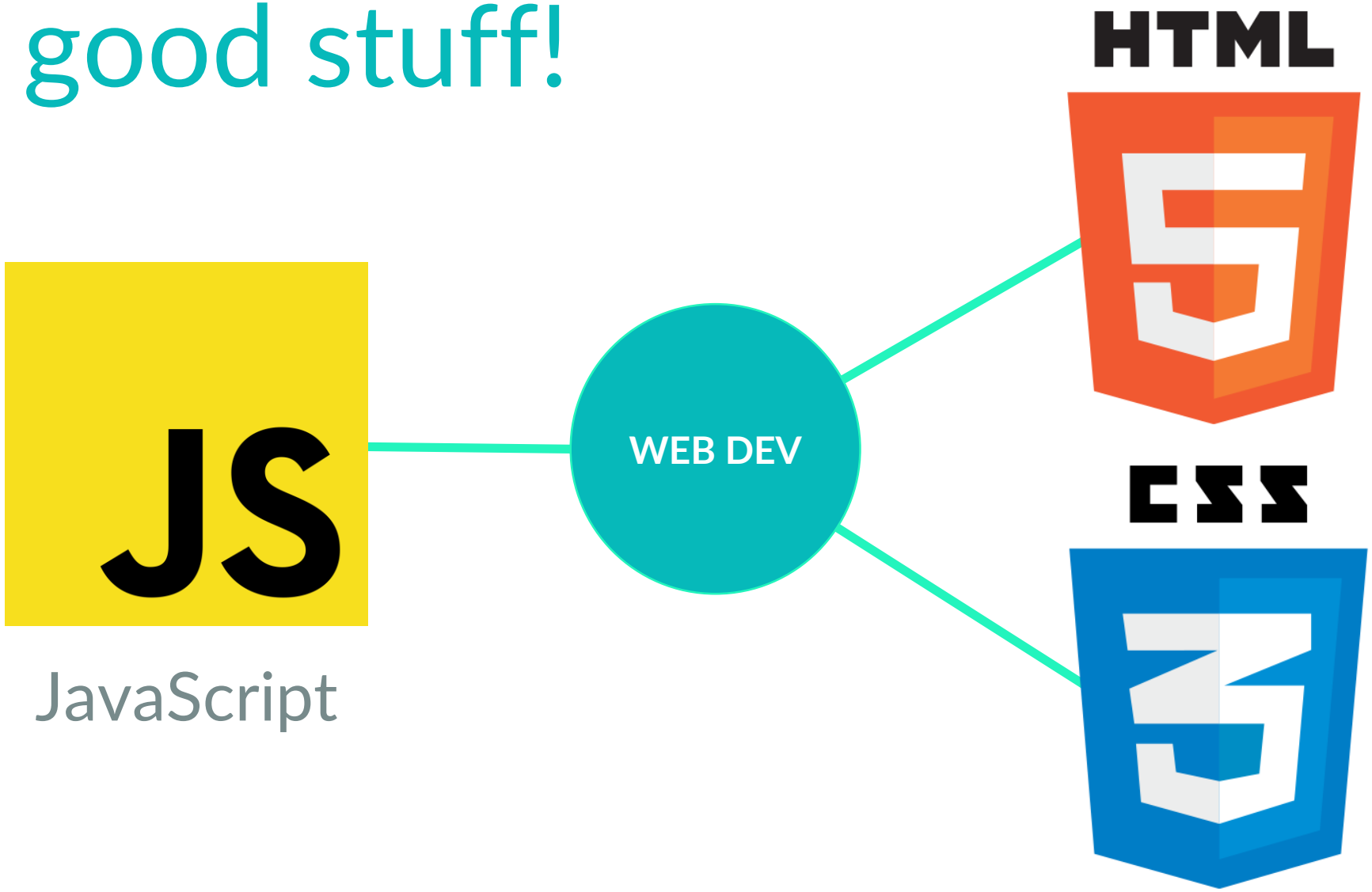
CREDIT/GRADING
1 unit, Pass/Fail

GRADING
Attendance (≥9/10)
Checkoff forms
Final reflections

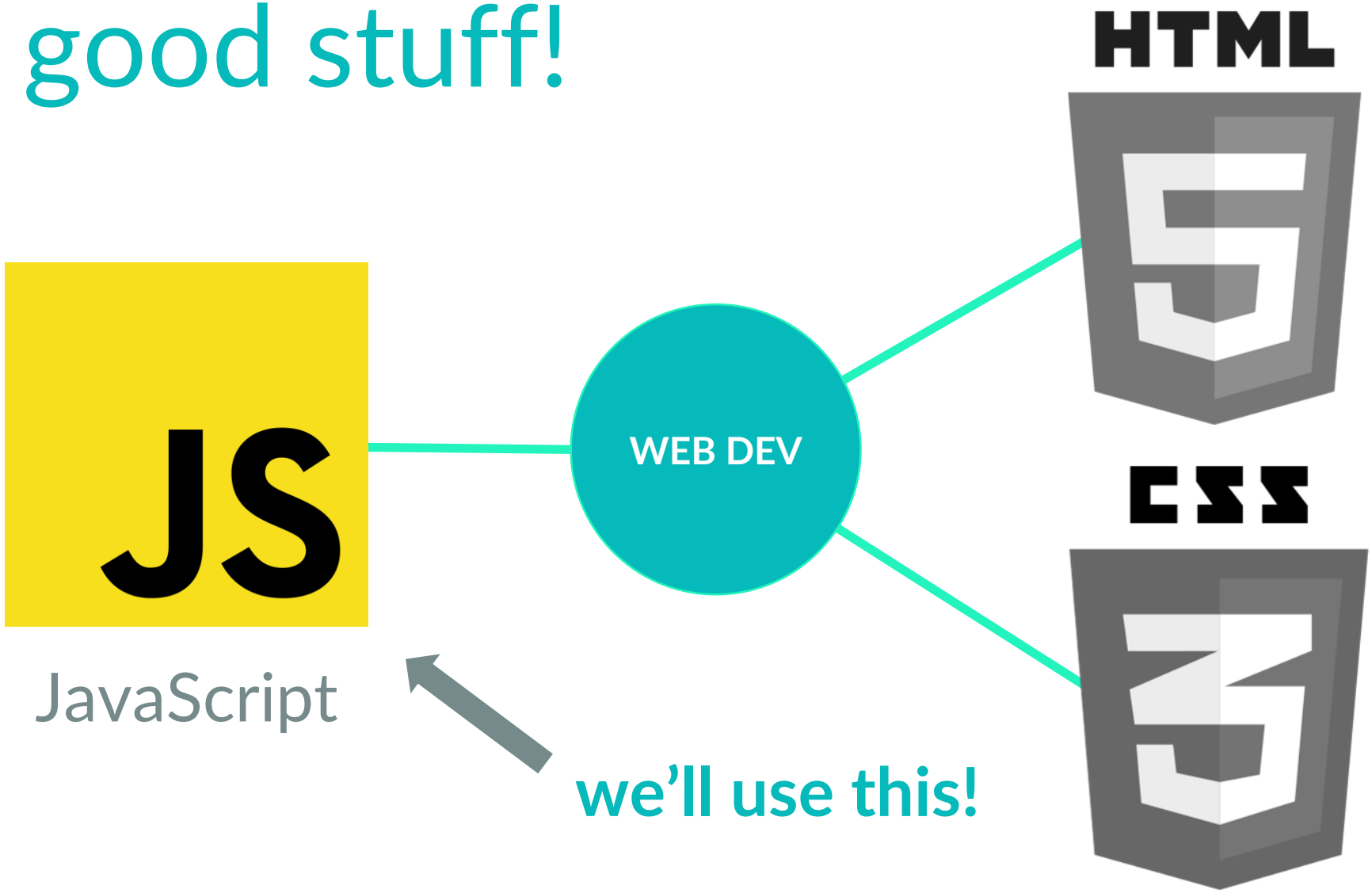
TEAM
Ashi Agrawal, Ryan Cohen
Kevin Khieu, Matthew Sun

Bring your laptops to every class!

the good stuff!



the good stuff!



the good stuff!

JavaScript

FROM WIKIPEDIA:

high-level, **dynamic**, **untyped**, and **interpreted** programming language

prototype-based with **first-class functions**

supports **object-oriented programming**

let's program some stuff!

…but first, open
repl.it/languages/javascript

JavaScript variables

```
var x; // dynamic typing
typeof x; // should print out ‘undefined’
x = 42; // typeof x == ‘number’
x = “CS106S rocks” // typeof x == ‘string’
x = true; // typeof x == ‘boolean’
```

Variable Scoping

```
// Two scopes: global and function local
var globalVar;
function() {
    var localVar;
    if(globalVar > 0) {
        var localVar2 = 2;
    }
}
```

Variable Scoping (cont.)

```
// All var statements hoisted to top of scope
```

```
function foo() {
    var x;
    x = 2;
}
```

```
function foo() {
    x = 2;
    var x;
}
```

number

- no distinction between int and double
- stored in floating point
  - Watch out! `(0.1 + 0.2) == 0.3 is false`
- interesting: `NaN, Infinity` are numbers
  - `1/0 == Infinity`
  - `Math.sqrt(-1) == NaN`

string

```
var valleyQuote = 'Our startup is like Uber for blockchain and it’s going to revolutionize the way we deliver pizza. #investpls'
valleyQuote.length // 108
```

- variable length
- + is string concat operator `// 'CS' + 'SG' == 'CSSG'`
- useful methods: `indexOf()`, `charAt()`, `search()`, `replace()`, `toUpperCase()`, `substr()`, etc.

boolean

```
var cs = true;
var socialGood = true;
var cssg = cs && socialGood; // true
```

- Either `true` or `false`
- Values are classified as **truthy** or **falsy:**
  - Used when values are converted to a boolean
- Falsy: false, 0, “”, null, undefined, NaN
- Truthy: anything that isn’t falsy

object

```
var berkeley = {intelligence: 0, tears: Infinity};
var stanford = {intelligence: 100, smiles: Infinity};
```

- Unordered collection of name-value pairs called **properties**
- Name can be any string: `var x = { "": "empty", "---": "dashes"}`
- Referenced either like a structure or like a hash table with string keys:
  - `stanford.smiles` or `stanford["smiles"]`

object

- Properties can be added or deleted

```
var queen = {};
queen.name = "Beyonce"; // queen.name returns "Beyonce"
```

- To remove use delete:

```
delete queen.name; // queen is now an empty object
```

- To enumerate use `Object.keys()`:

```
Object.keys({name: "Alice", age: 23}) = ["name", "age"]
```

Arrays

```
var youIs = ['smart', 'kind', 'important',];
```

- Special objects: `typeof youIs == 'object'`
- Zero-indexed
- Can be **sparse** and **polymorphic**
  - `youIs[5] = 100; // ['smart', 'kind', 'important',,, 100]`
- Like strings, have many methods: `youIs.length == 3`
  - `push, pop, shift, unshift, sort, reverse, splice`

Checking Equality

Who would win: == or ===

== (loose equality)

- Compares two values for equality, *after* converting both values to a common type
  - `3 == '3' // true`

=== (strict equality)

- Neither value is implicitly converted before comparison
- Different types → unequal
  - `3 === '3' // false`

undefined and null

- `undefined` – does not have a value assigned

```
var x; // x has a value of undefined
x = undefined; // can be explicitly stored
typeof x == 'undefined'
```

- `null` – a value that represents whatever the user wants it to (sentinel)

```
typeof null = 'object'
```

- Both are **falsy** but not equal (`null == undefined`, `null !== undefined`)

Conditionals & Loops (you know the drill...)

```
if (condition) {
    // do something
}
for (var i = 0; i < 10; i++) {
    // do something 10 times
}
```

```
while (condition) {
    // do something
    // while condition
    // is truthy
}
```

function type

```
// untyped parameters
function isGoodPerson(isIn106S) {
      if(isIn106S) {
            return 'Definitely. No question.';
      } else {
            return 'Maybe? Like, it’s possible.';
      }
}
// all functions return a value; default is undefined
```

First-class function example

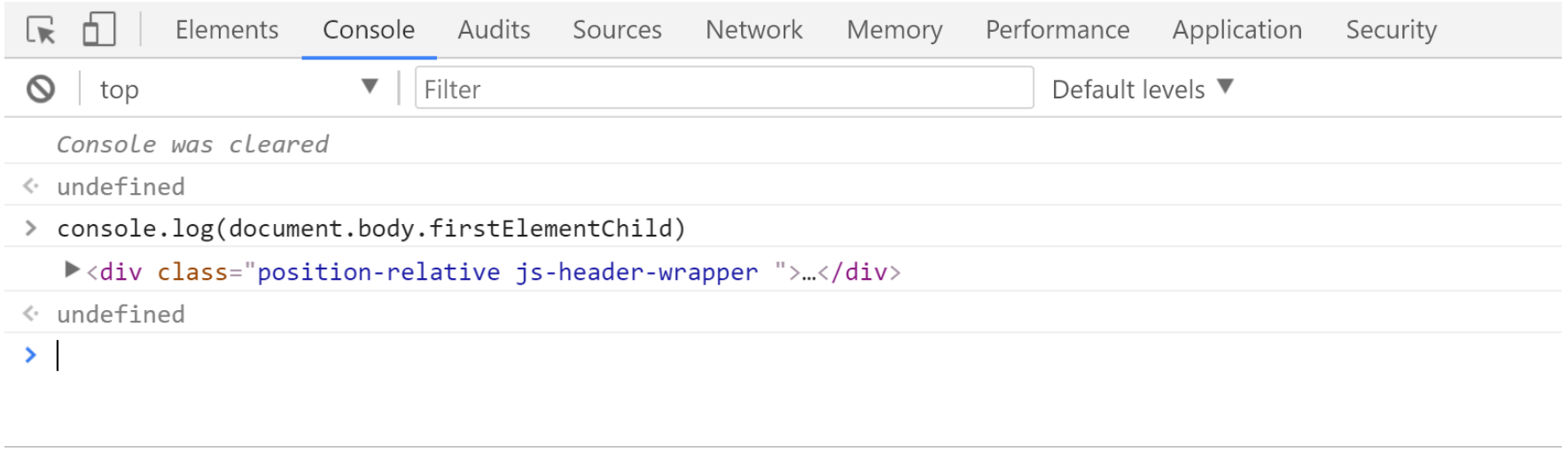
```
function myFunc(routine) {
      console.log('Called with', routine.toString());
      var retVal = routine(10);
      console.log('retVal', retVal);
}
myFunc(function (x) {
            console.log('Called with', x);
            return x+1;
      }
);
```

Console

Chrome:

Right click → Inspect Element *or*

command + alt + i (**Mac**) or ctrl + shift + i (**Windows**)



funzo assignment time

Let's put your knowledge to the test by making a program that helps your friends feel better!

Copy the code at `bit.ly/cssgnicebot` into a repl.it and fill in the code for `saySomethingNice`, `upliftingQuote`, `inspiringQuote`, and `soothingQuote`.

See you next time, friends!

Before next time:

Install Python 3

Install Node.js