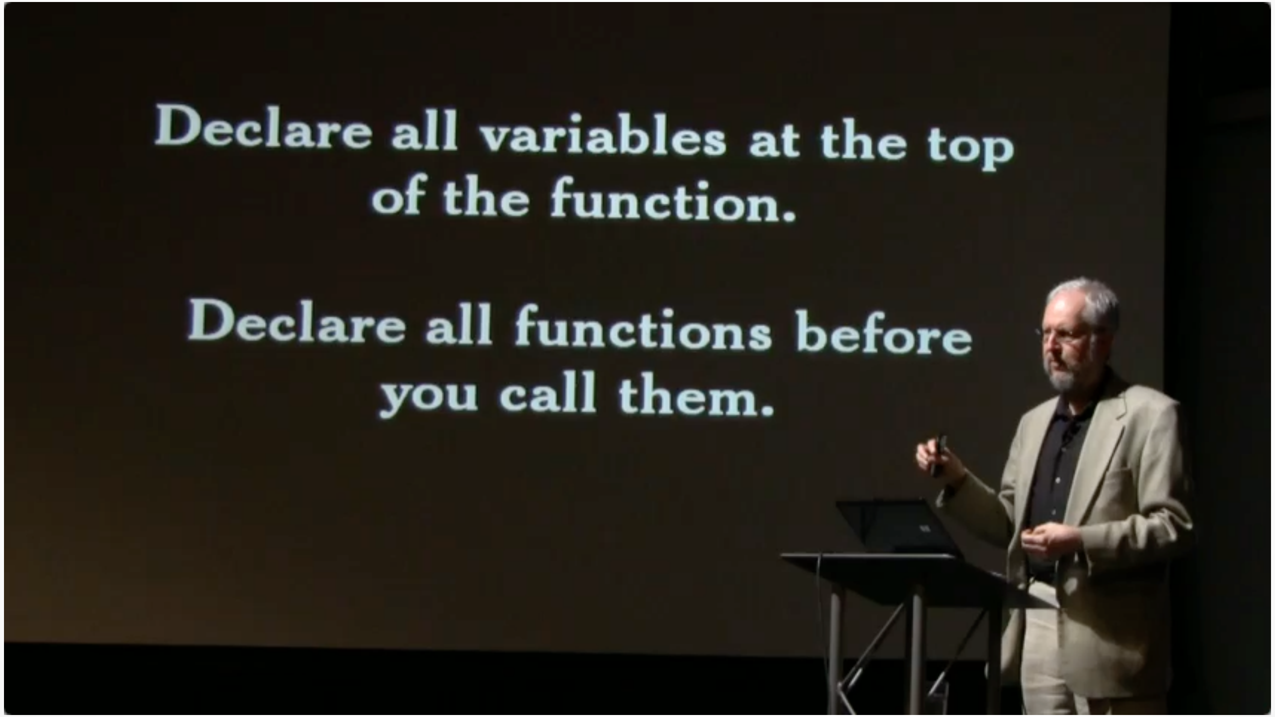
CS 22A JavaScript for Programmers

(Bah-bah Co-fee Way-ou-see-jah-nah)

Today (Day 09)

- Declaring variables and functions correctly to avoid hoisting in ES5
- Function statements vs. function expressions

Avoiding Hoisting in ES5



Declare all variables at the top of the function.

Declare all functions before you call them.

https://developer.mozilla.org/en-US/docs/Glossary/Hoisting

Function statements vs. function expressions

- If the first token of a statement is `function`, then it's a **function statement/function declaration** which <u>will</u> be hoisted.
    - Crockford often uses the term "**function statement**" but Mozilla docs often use the term "**function declaration**"

```
hoisted(); // logs "Surprise!"
function hoisted() { console.log("Surprise!"); }
```

Function statements vs. function expressions

- If the first token of a statement is `function`, then it's a **function statement/function declaration** which will be hoisted.
  - Crockford often uses the term "**function statement**" but Mozilla docs often use the term "**function declaration**"

```
hoisted(); // logs "Surprise!"
function hoisted() { console.log("Surprise!"); }
```

- Otherwise, it's a **function expression** which are not hoisted (thus less confusing and more predictable).

```
notHoisted(); // TypeError: notHoisted is not a function
var notHoisted = function() { console.log("nope"); };
```

Avoiding Hoisting in ES5

- You can't put a function statement/declaration inside of an if statement or your code might run differently across browsers
  - See https://stackoverflow.com/questions/10069204/function-declarations-inside-if-else-statements
- You can avoid (or get warned) about these issues if you use strict mode by putting

```
"use strict";
```

in the **first line** of each function body