

STUFF I LEARNED FROM

---

**FORWARD JS**

# FORWARD JS

- ▶ Dedicated to JavaScript
- ▶ Been around about 2.5 years
- ▶ Held in San Francisco
- ▶ **AMAZING:** amount of content, depth of content, type of content, quality of speakers (and San Francisco is pretty cool)
- ▶ Type of content:
  - ▶ workshops - intensive, one day per topic, in the code
  - ▶ conference sessions - an hour or two, not in the code

## WHAT I'M GOING TO PRESENT

- ▶ Workshop: Advanced JavaScript Fundamentals
- ▶ Workshop: 4 Semesters of Computer Science in 6 Hours

## ADVANCED JAVASCRIPT FUNDAMENTALS

- ▶ Instructor: Kyle Simpson, JavaScript Developer/Instructor/Author
- ▶ “You Don’t Know JS” - a free version of his book here: <https://github.com/getify/You-Dont-Know-JS>

# ADVANCED JAVASCRIPT FUNDAMENTALS

- ▶ Scope and Closures
- ▶ this, prototypes, and inheritance

## ADVANCED JAVASCRIPT FUNDAMENTALS

- ▶ Scope and Closures
  - ▶ Scope: where to look for things
    - ▶ “the set of rules that govern how the Engine can look up a variable by its identifier name and find it, either in the current Scope, or in any of the Nested Scopes it's contained within”

Global Scope

```
<script>
  var stuff = "this is a variable";
  var mainFunction = function() { Scope of mainFunction
    var moreStuff = "more stuff";
    var x = 1;
    var y = 2;
    var add = function() { Scope of add
      var base = 10;
      var moreStuff = "Adding more stuff";
      console.log(base + x + y);
    }
    add();
  }
  mainFunction();
</script>
```

# ADVANCED JAVASCRIPT FUNDAMENTALS

- ▶ Scope and Closures
  - ▶ Note: JS uses Lexical Scope
    - ▶ Most programming languages do
  - ▶ What determines scope in JavaScript?
    - ▶ Functions
    - ▶ let (in ES6)
  - ▶ Function declaration vs Function expression



# ADVANCED JAVASCRIPT FUNDAMENTALS

- ▶ Compiler Theory
  - ▶ Not interpreted, instead compiled
    - ▶ [Example](#)
  - ▶ 2 Phases
    - ▶ Compile
    - ▶ Execute

# ADVANCED JAVASCRIPT FUNDAMENTALS

- ▶ Scope and Closures
  - ▶ How scope gets built
    - ▶ "Hoisting"
      - ▶ function declarations (first)
      - ▶ var declarations (second)

## ADVANCED JAVASCRIPT FUNDAMENTALS

- ▶ Scope and Closures

- ▶ Closure: A function which uses a variable from its context.

- ▶ Great resource: <http://doctrina.org/JavaScript:Why-Understanding-Scope-And-Closures-Matter.html>

```
<script>
  var stuff = "this is a variable";

  var mainFunction = function() {

    var moreStuff = "more stuff";

    var x = 1;
    var y = 2;

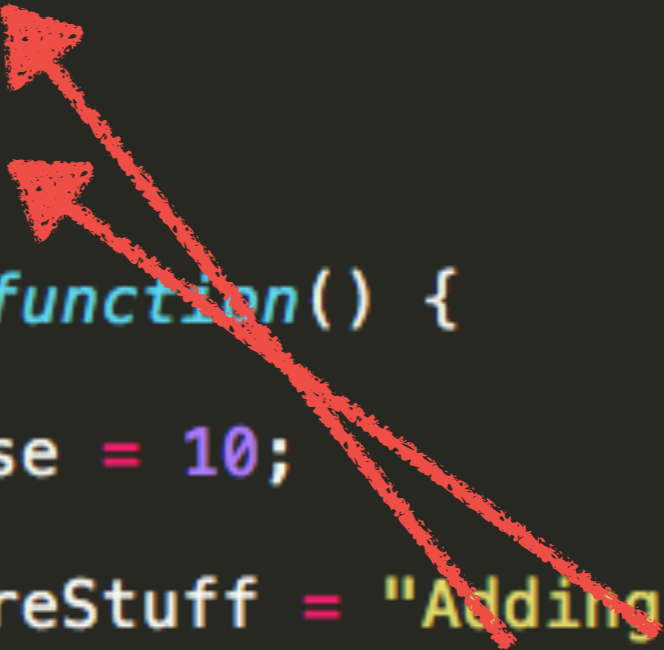
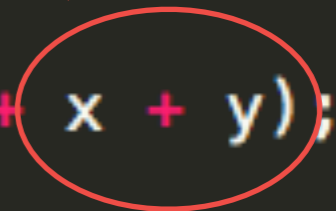
    var add = function() {
      var base = 10;
      var moreStuff = "Adding more stuff";
      console.log(base + x + y);
    }

    add();

  }

  mainFunction();
</script>
```

Closure



# ADVANCED JAVASCRIPT FUNDAMENTALS

- ▶ this
  - ▶ Misconceptions
    - ▶ this refers to the function itself
    - ▶ this refers to this function's scope
  - ▶ How do I know what this is bound to?
    - ▶ Example
    - ▶ Ask these 4 questions in this order.
      - ▶ Called with new? Use the newly constructed object.
      - ▶ Called with call or apply (or bind)? Use the specified object.
      - ▶ Called with a context object owning the call? Use that context object.
      - ▶ Default: undefined in strict mode, global object otherwise.

## ADVANCED JAVASCRIPT FUNDAMENTALS

- ▶ this

- ▶ Are there exceptions to these 4 rules?

- ▶ Yep, see here: <https://github.com/getify/You-Dont-Know-JS/blob/master/this%20%26%20object%20prototypes/ch2.md>

## 4 SEMESTERS OF COMPUTER SCIENCE IN 6 HOURS

- ▶ Instructor: **Brian Holt**, Senior UI Engineer at Netflix
- ▶ Link to pretty much his entire presentation/workshop - <http://btholt.github.io/four-semesters-of-cs/>

## 4 SEMESTERS OF COMPUTER SCIENCE IN 6 HOURS

- ▶ Term: algorithm
  - ▶ A step-by-step procedure for solving a problem.



# 4 SEMESTERS OF COMPUTER SCIENCE IN 6 HOURS

- ▶ Topics
  - ▶ BigO
  - ▶ Recursion
  - ▶ Sorting
  - ▶ Data Structures
  - ▶ A lunch conversation

## 4 SEMESTERS OF COMPUTER SCIENCE IN 6 HOURS

- ▶ Sorting
  - ▶ Bubble Sort
  - ▶ Insertion Sort
  - ▶ Merge Sort
  - ▶ Quick Sort

## 4 SEMESTERS OF COMPUTER SCIENCE IN 6 HOURS

- ▶ A lunch conversation
  - ▶ What is developing?
  - ▶ Interviewing - the results at Google

## 4 SEMESTERS OF COMPUTER SCIENCE IN 6 HOURS

### ▶ Takeaways

- ▶ As a UI Engineer, I haven't been missing much.
- ▶ "Programming is really just the ability to google the right terms."
- ▶ "Computer Science is the science of tradeoffs."

# LAST ITEMS

- ▶ Some video-recorded presentations from ForwardJS:  
<https://forwardcourses.com/lectures>

**FORWARDJS.COM**

**HELD TWICE A YEAR**