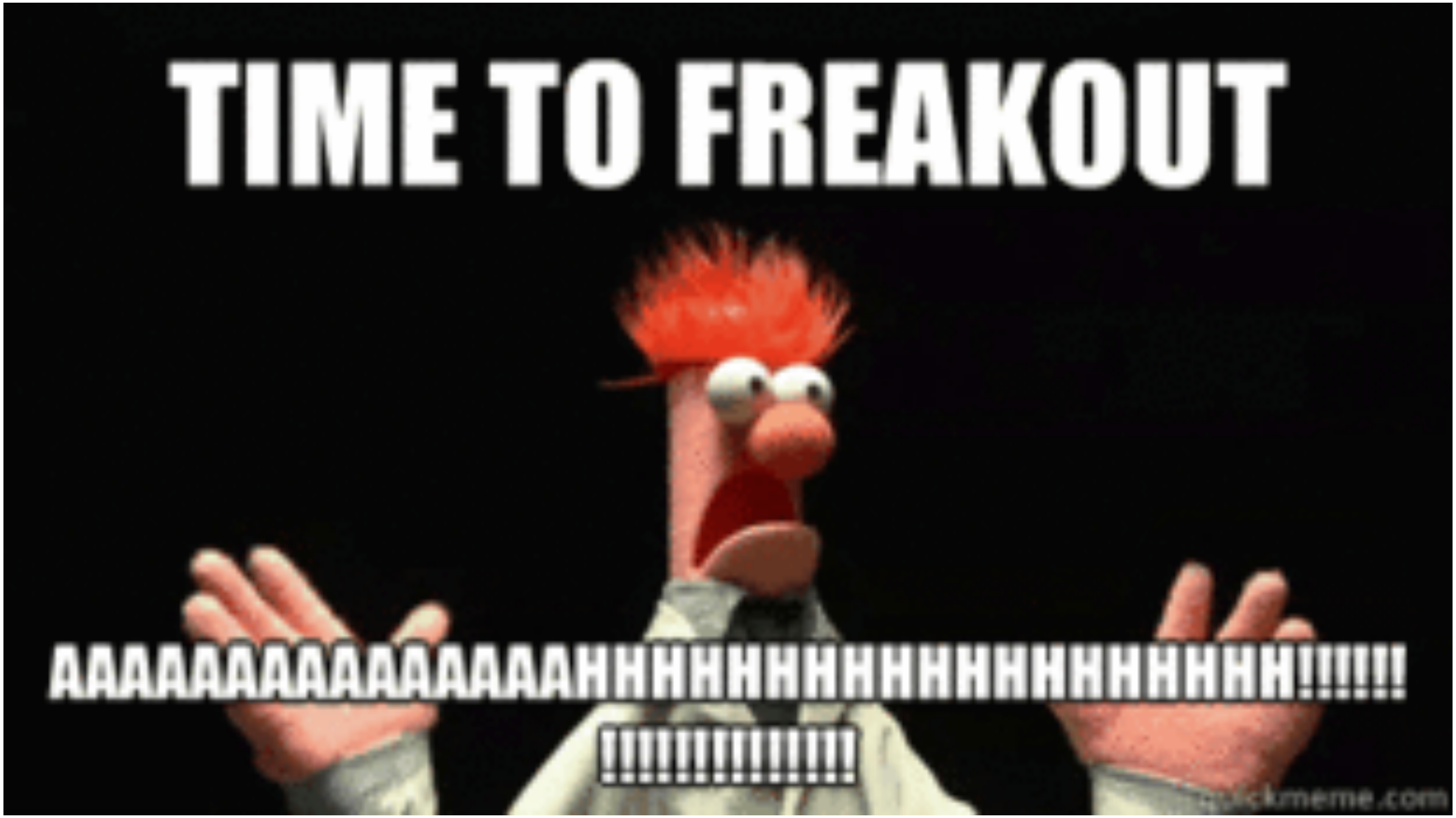
JS and jQuery Jubilee

The Coding Bootcamp

This shouldn't be you…



Remember this!

“You can’t tell whether you’re learning something when you’re learning it—in fact, learning feels a lot more like frustration.”

“What I’ve learned is that during this period of frustration is actually when people improve the most, and their improvements are usually obvious to an outsider. If you feel frustrated while trying to understand new concepts, try to remember that it might not feel like it, **but you’re probably rapidly expanding your knowledge.**”

Jeff Dickey, Author of Write Modern Web Apps with the MEAN Stack: Mongo, Express, AngularJS, and Node.JS

Feedback #1 – Pace is Fast!!!

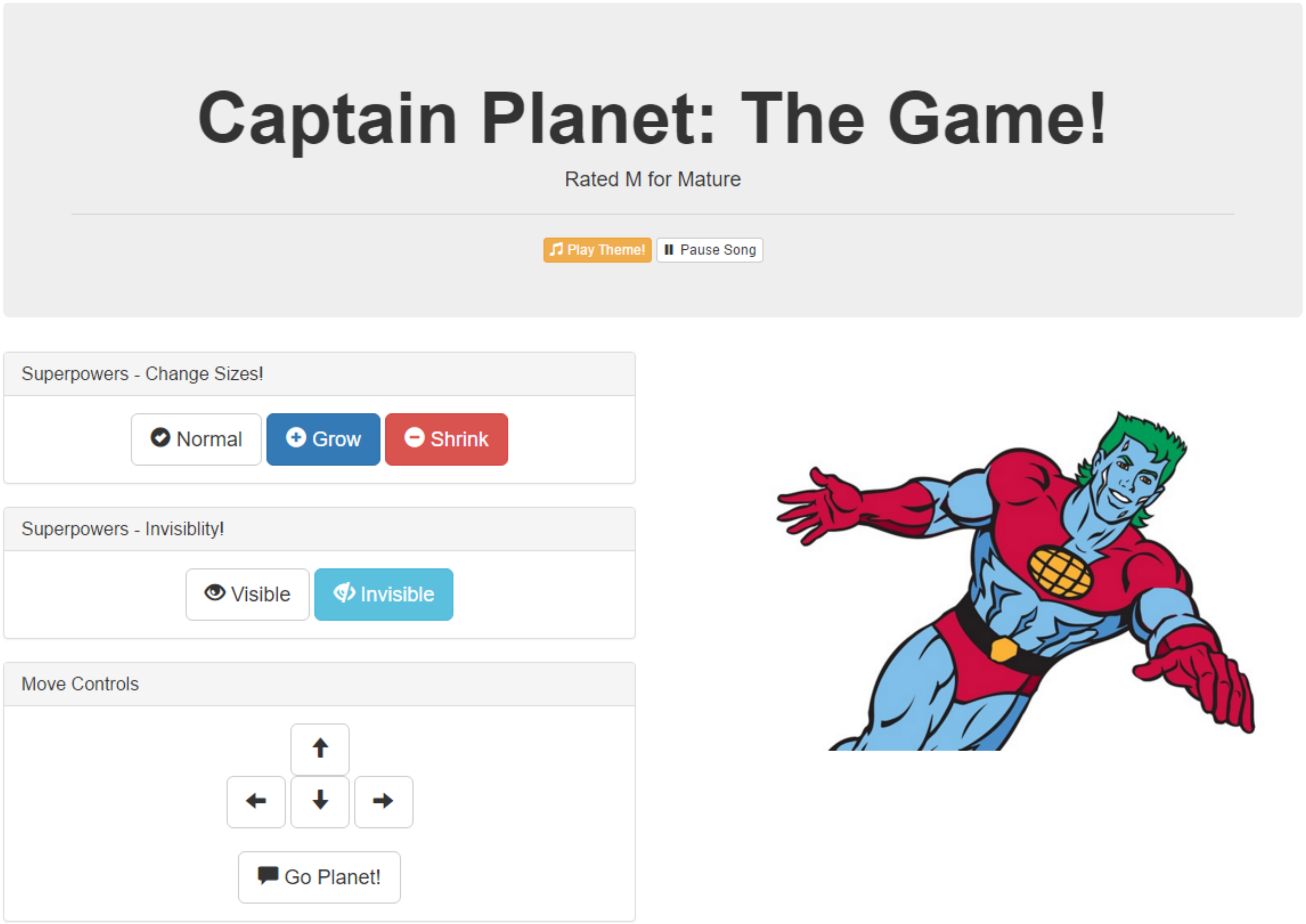
- That said, as instructors / TAs we are here to help.
- As we fall into a class rhythm, feel encouraged to schedule a 1-1 during office hours.
- In addition to using the time to understand concepts… it's a great way for us to identify weaknesses and outline steps to get on the right track.
- These might be before / after class.

Today’s Class

Objectives

1. Play Captain Planet: The GAME!
2. Practice jQuery on Fridge
3. Pretend to learn scoping
4. Understand click events

Captain Planet!


Captain Planet: The Game!
Rated M for Mature
Play Theme!
Pause Song
Superpowers - Change Sizes!
Normal
Grow
Shrink
Superpowers - Invisiblity!
Visible
Invisible
Move Controls
Go Planet!

Demo Time

Instructor: Demo

(CaptainPlanet.html | 1-CaptainPlanet)

> YOUR TURN!!

Activity*:* 1-CaptainPlanet **| Suggested Time:** 7min

Code Dissection:
Examine the code for the Captain Planet Game

Then, in groups, describe how this code works in **5 Steps.**

1.

2.

3.

4.

5.

Pseudocoding – Captain Planet

Solution:

1. An initial HTML Layout was created using Bootstrap.

2. A reference to jQuery was added.

3. Key buttons and images were assigned unique class names

4. jQuery was used to capture when the corresponding buttons were clicked. This was done through the $( ) identifier with the class-name inside.

5. Code was created that changed the css of target classes in response to the click events.

> YOUR TURN!!

Activity*:* 1-CaptainPlanet **| Suggested Time:** 12 min

- Look at the jQuery API Docs and add a button of your own that gives Captain Planet a new power.

  - Examples:
    - Click to… stretch Captain Planet
    - Click to… trigger a maniacal laugh
    - Click to… create clones of Captain Planet
    - Click to… create a shield (hint: border)
    - Click to… create fire or water (hint: images)

- **Slack out a screenshot of the working example**

jQuery Recap

jQuery – In a Nutshell

1. **Find some HTML.**
2. **Attach to an event.**
3. **Do something in response.**

jQuery – In a Nutshell

We use the jQuery $( ) identifier to capture HTML elements.

$(".classname") **$("elementname")**

$("#idname") **$("etc")**

Then we tie the element to a jQuery method of our choosing to capture events and change that element (or a different element)

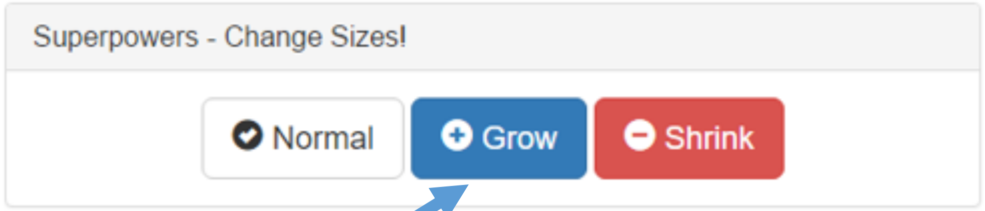
.on("click") **.ready( )**

Finally, we tie the element to a jQuery method of our choosing to capture events and change that element (or a different element)

.append( ) **.animate( )** **.etc()**

jQuery – Common Example

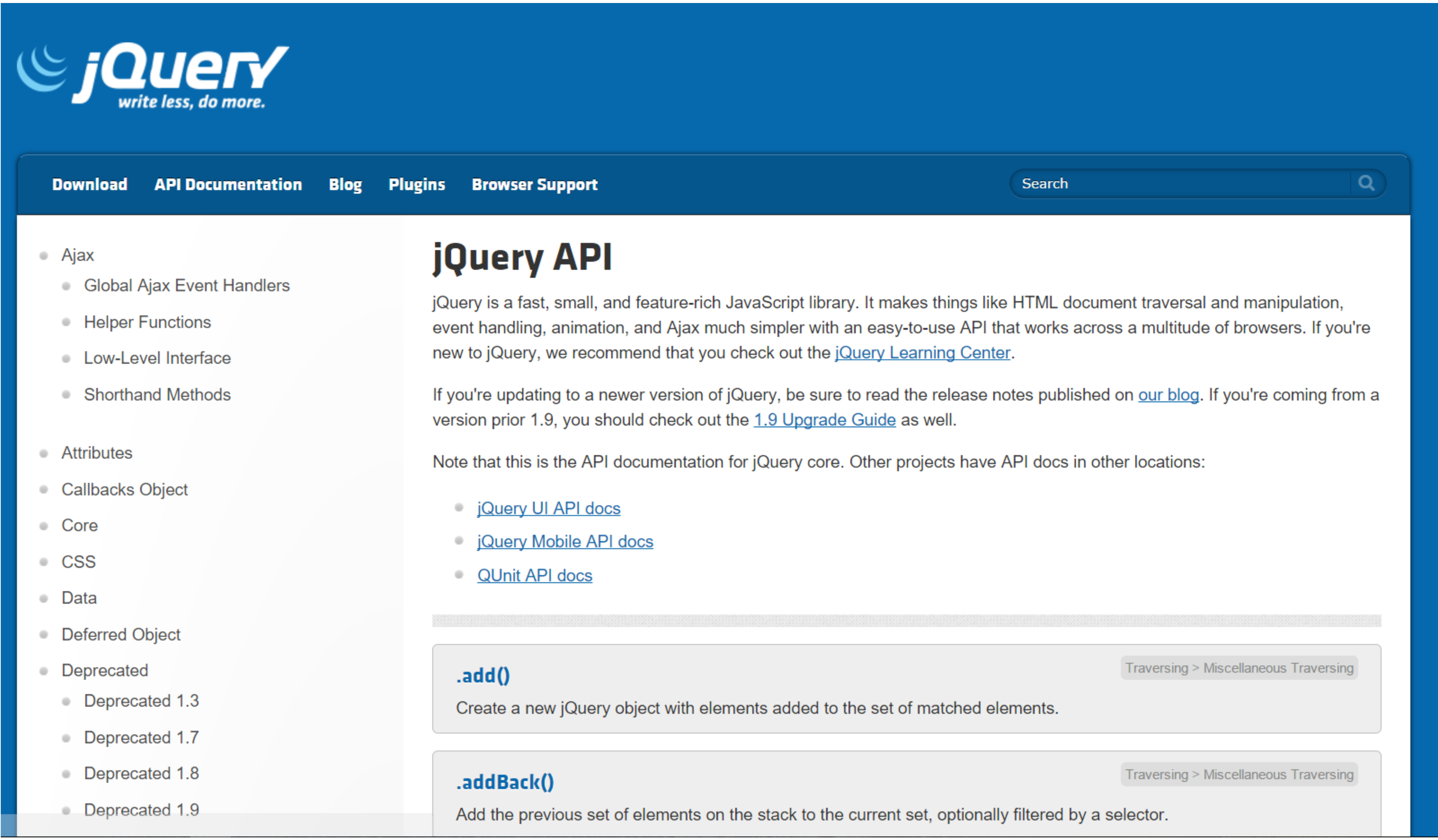
```
$(".growButton").on("click", function() {
  $(".captainplanet").animate({ height: "500px" });
});
```



1. Click the Grow Button

2. Make Captain Planet Grow

Use Documentation When Needed!



- Ajax
  - Global Ajax Event Handlers
  - Helper Functions
  - Low-Level Interface
  - Shorthand Methods
- Attributes
- Callbacks Object
- Core
- CSS
- Data
- Deferred Object
- Deprecated
  - Deprecated 1.3
  - Deprecated 1.7
  - Deprecated 1.8
  - Deprecated 1.9

jQuery API

jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. If you're new to jQuery, we recommend that you check out the jQuery Learning Center.

If you're updating to a newer version of jQuery, be sure to read the release notes published on our blog. If you're coming from a version prior 1.9, you should check out the 1.9 Upgrade Guide as well.

Note that this is the API documentation for jQuery core. Other projects have API docs in other locations:

- jQuery UI API docs
- jQuery Mobile API docs
- QUnit API docs

.add() — Traversing > Miscellaneous Traversing

Create a new jQuery object with elements added to the set of matched elements.

.addBack() — Traversing > Miscellaneous Traversing

Add the previous set of elements on the stack to the current set, optionally filtered by a selector.

Fridge Game!

> YOUR TURN!!

Activity: 2-FridgeGame | **Suggested Time:** 35 min

- Working in groups of 3 complete the code for the fridge activity such that:
  - Javascript dynamically generates buttons for each of the letters on the screen.
  - Clicking any of the buttons leads the SAME letter to be displayed on the screen.
  - Hitting the clear button erases all of the letters from the fridge.
- *Note: This is a <u>challenging</u> exercise. You may want one person to type, while the other two watch over to catch bugs and/or research necessary snippets.*

Crystal Collector!

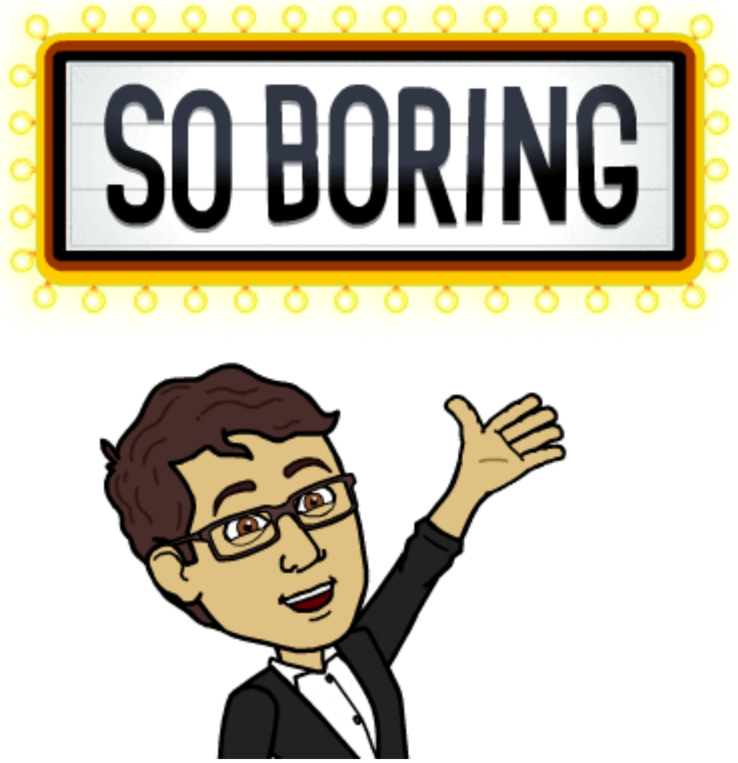
Demo Time

Instructor: Demo

(1-12.html | 3-CrystalExample)

Lexical Scope

Shh… Just Between Us.

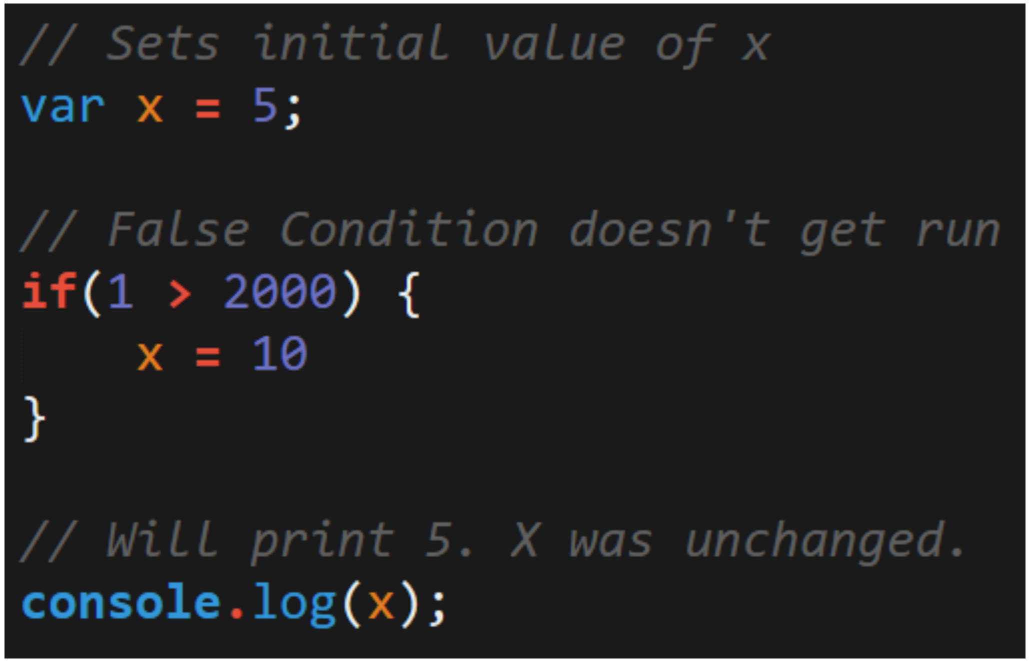


<u>WARNING:</u>
This next section is heavy on theory.

Disclaimer:
It’s not the end of the world if its confusing and/or you’re completely lost.

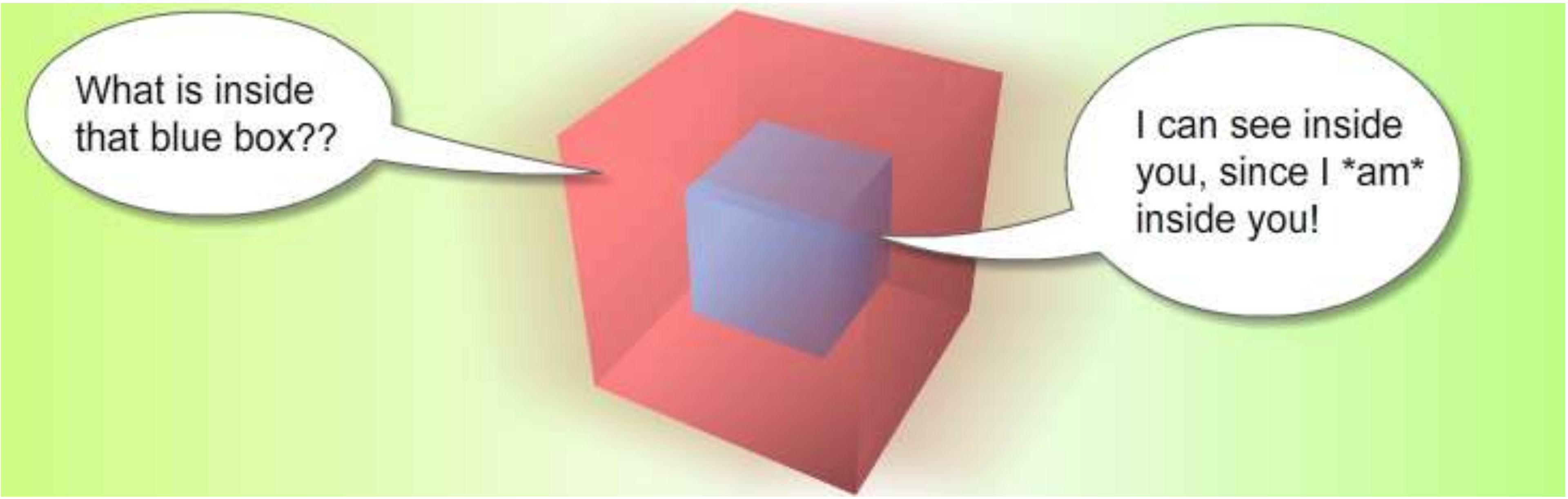
Javascript Scope

- In Javascript curly brackets { } indicate blocks of code.
- In order for the code inside the curly brackets to be executed, it must meet the condition or it must be called (example: functions).
- These blocks of code have the power to affect variables outside the curly brackets if those variables were declared outside – so be careful!



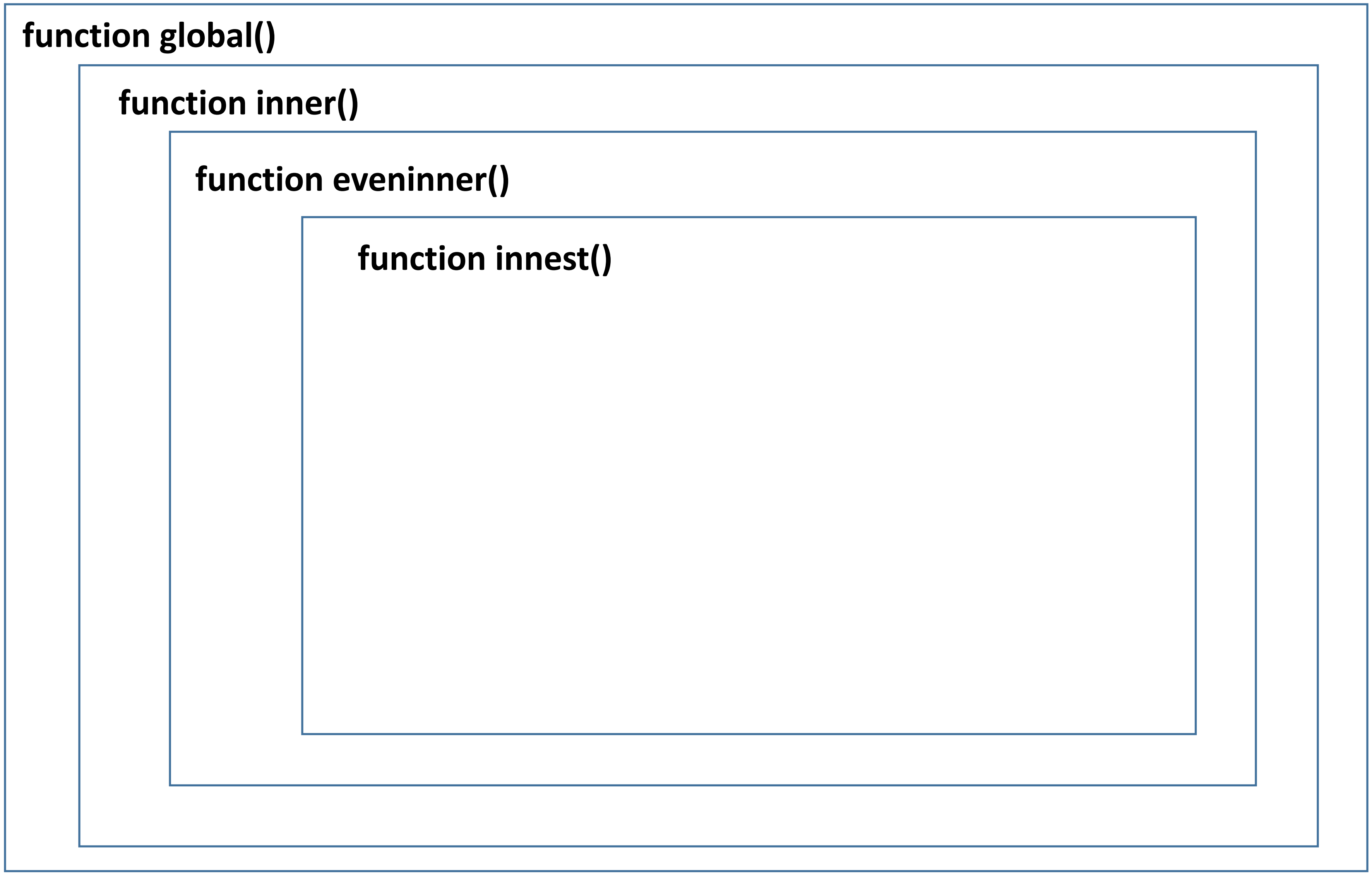
```javascript
// Sets initial value of x
var x = 5;

// False Condition doesn't get run
if(1 > 2000) {
    x = 10
}

// Will print 5. X was unchanged.
console.log(x);
```

Scope = Boxes in Boxes

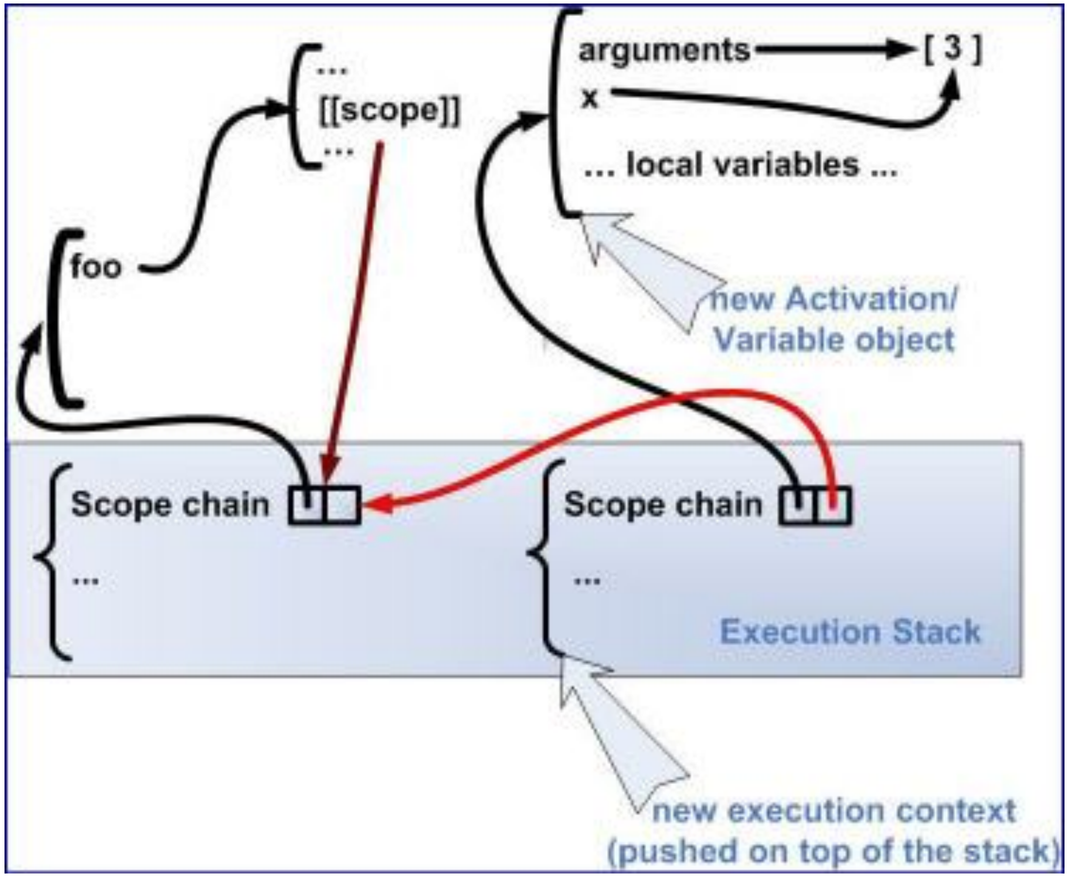

What is inside that blue box??
I can see inside you, since I *am* inside you!

<u>Scope</u> impacts which variables can be accessed by which function.

Scope = Boxes in Boxes

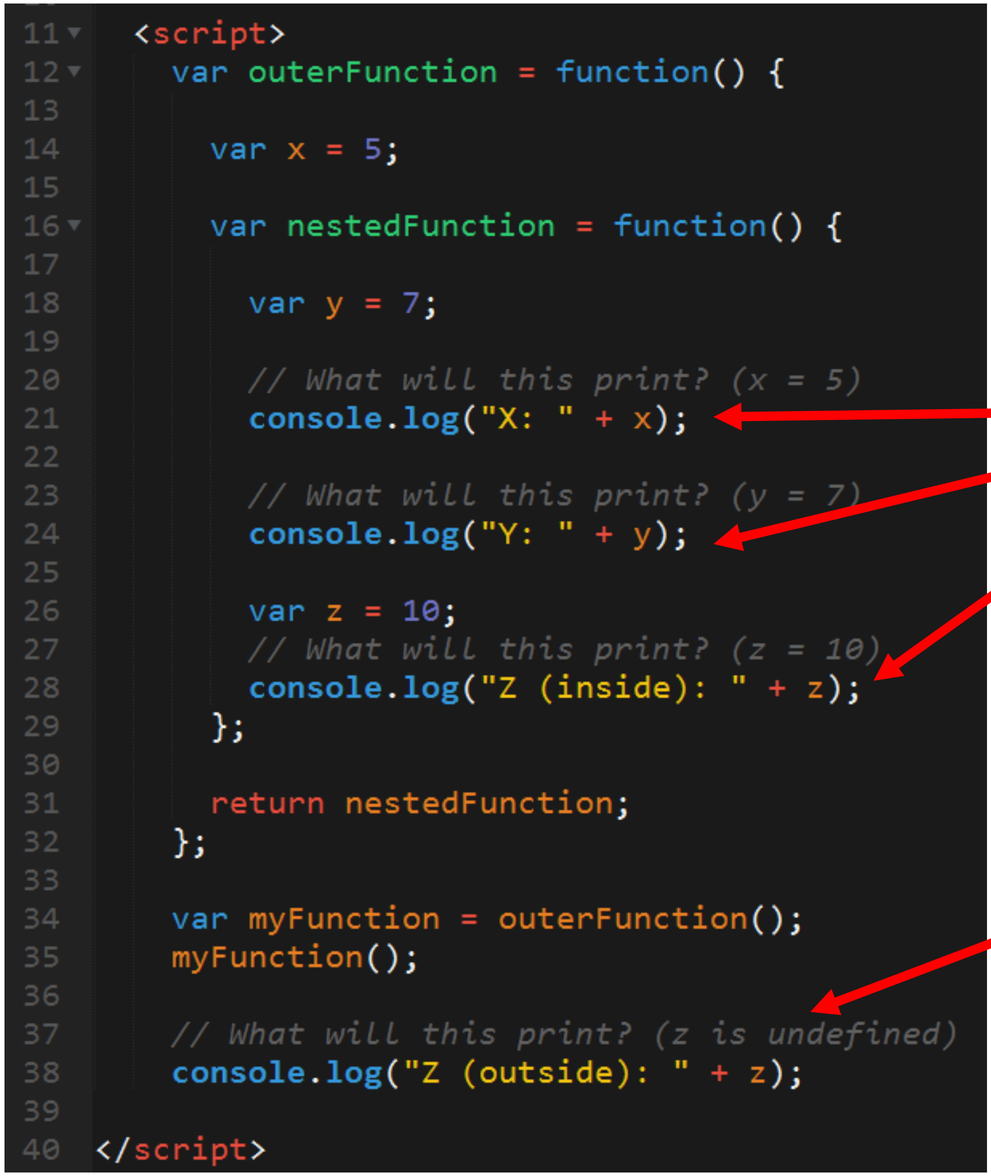


Javascript's Odd Relationship with Scope



For those who have programmed in other languages, Javascript seemingly behaves in unpredictable ways.

Javascript Scope Example (Tricky)



```
<script>
  var outerFunction = function() {

    var x = 5;

    var nestedFunction = function() {

      var y = 7;

      // What will this print? (x = 5)
      console.log("X: " + x);

      // What will this print? (y = 7)
      console.log("Y: " + y);

      var z = 10;
      // What will this print? (z = 10)
      console.log("Z (inside): " + z);
    };

    return nestedFunction;
  };

  var myFunction = outerFunction();
  myFunction();

  // What will this print? (z is undefined)
  console.log("Z (outside): " + z);

</script>
```

Here **nested function** is clearly able to access the variables of their **parent function**.

Whereas **outer function** has no idea what the variable z is because it was declared in a child function.

> YOUR TURN!!

Activity: *4-ScopeOne* | **Suggested Time:** 5 min

- Take a few moments dissecting what I just said.
- Look at the file sent to you and explain to the person next to you what is meant by:
  - The terms parent function and child function
  - The concept that child functions can access parent variables but not vice versa.
- **Be prepared to share!**

> YOUR TURN!!

Activity*: 5-ScopeTwo* **| Suggested Time:** 7 min

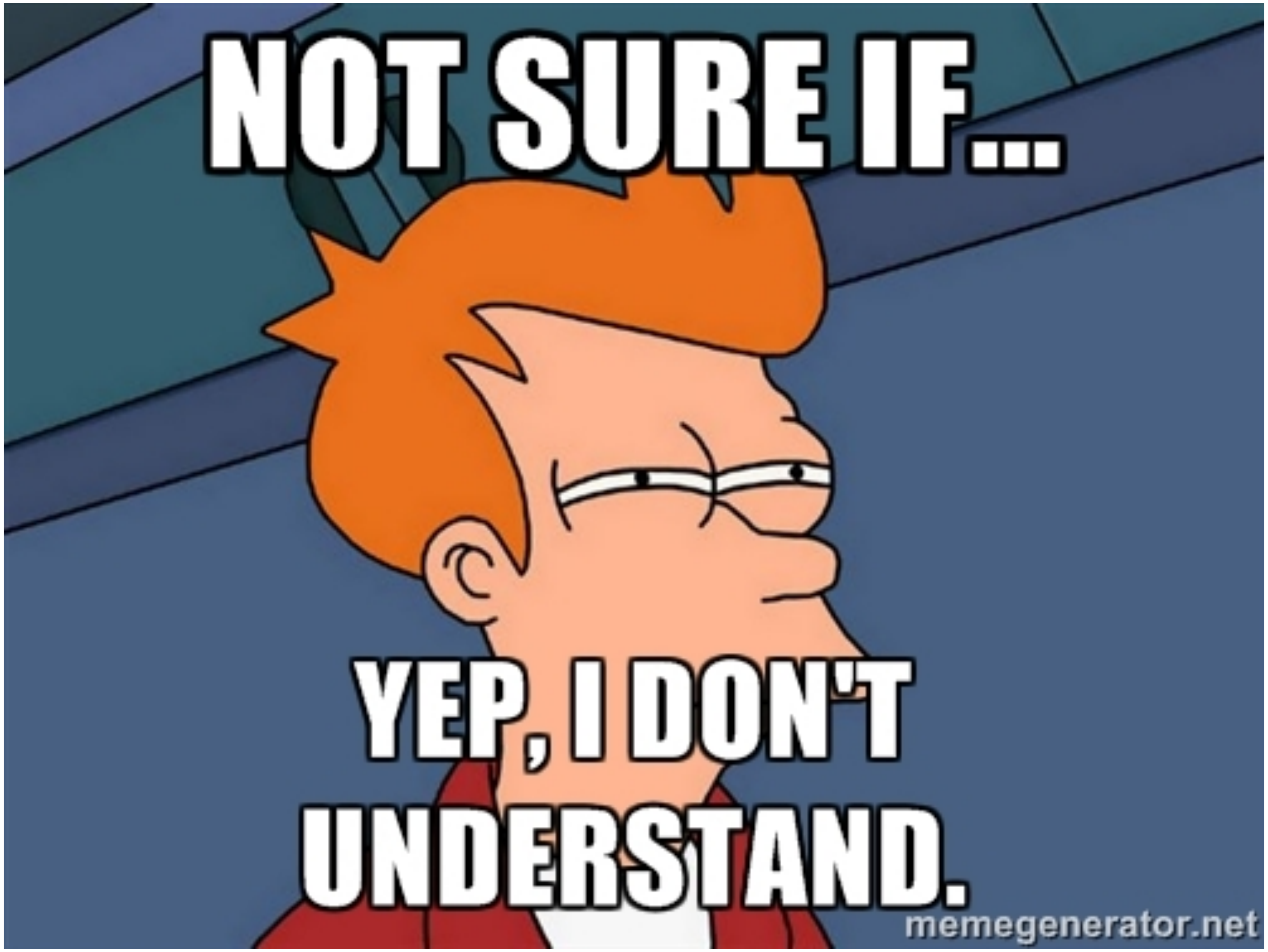
- Take a few moments to dissect the code just sent to you.
- Try to predict what will be printed in each of the examples.
- **Be prepared to share!**
- Note: Pay attention to the unusual use of the keyword: ‘this”

> YOUR TURN!!

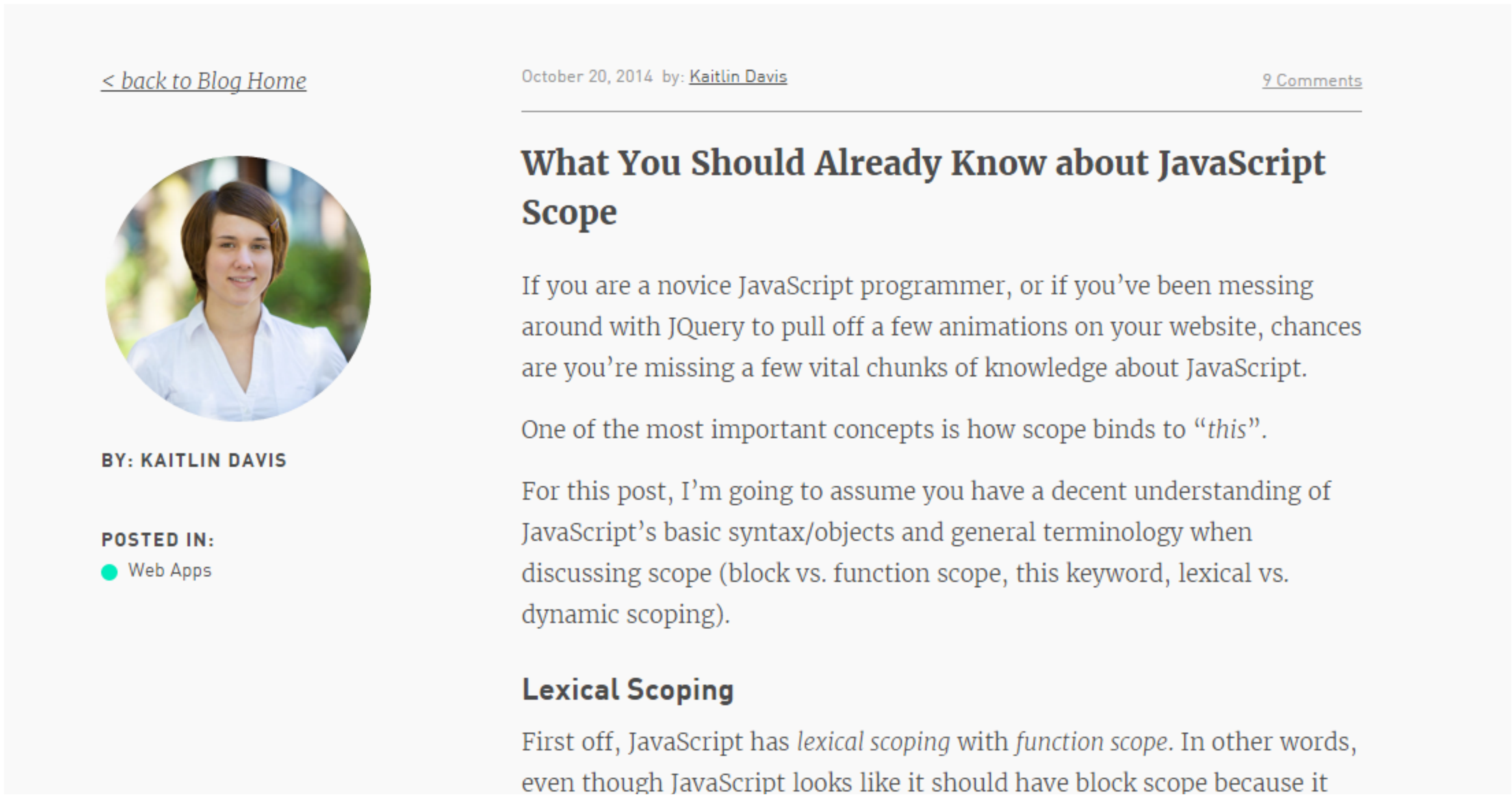
Activity*: 6-ScopeThree* **| Suggested Time:** 7 min

- Take a few moments to dissect the code just sent to you.
- Try to predict what will be printed in each of the examples.
- **Be prepared to share!**
- Note: Pay attention to the unusual use of the keyword: ‘this”

You Probably…



Helpful Article (If you'd like to learn more…)



< back to Blog Home

October 20, 2014 by: Kaitlin Davis

9 Comments

What You Should Already Know about JavaScript Scope

If you are a novice JavaScript programmer, or if you've been messing around with JQuery to pull off a few animations on your website, chances are you're missing a few vital chunks of knowledge about JavaScript.

One of the most important concepts is how scope binds to "*this*".

For this post, I'm going to assume you have a decent understanding of JavaScript's basic syntax/objects and general terminology when discussing scope (block vs. function scope, this keyword, lexical vs. dynamic scoping).

Lexical Scoping

First off, JavaScript has *lexical scoping* with *function scope*. In other words, even though JavaScript looks like it should have block scope because it

BY: KAITLIN DAVIS

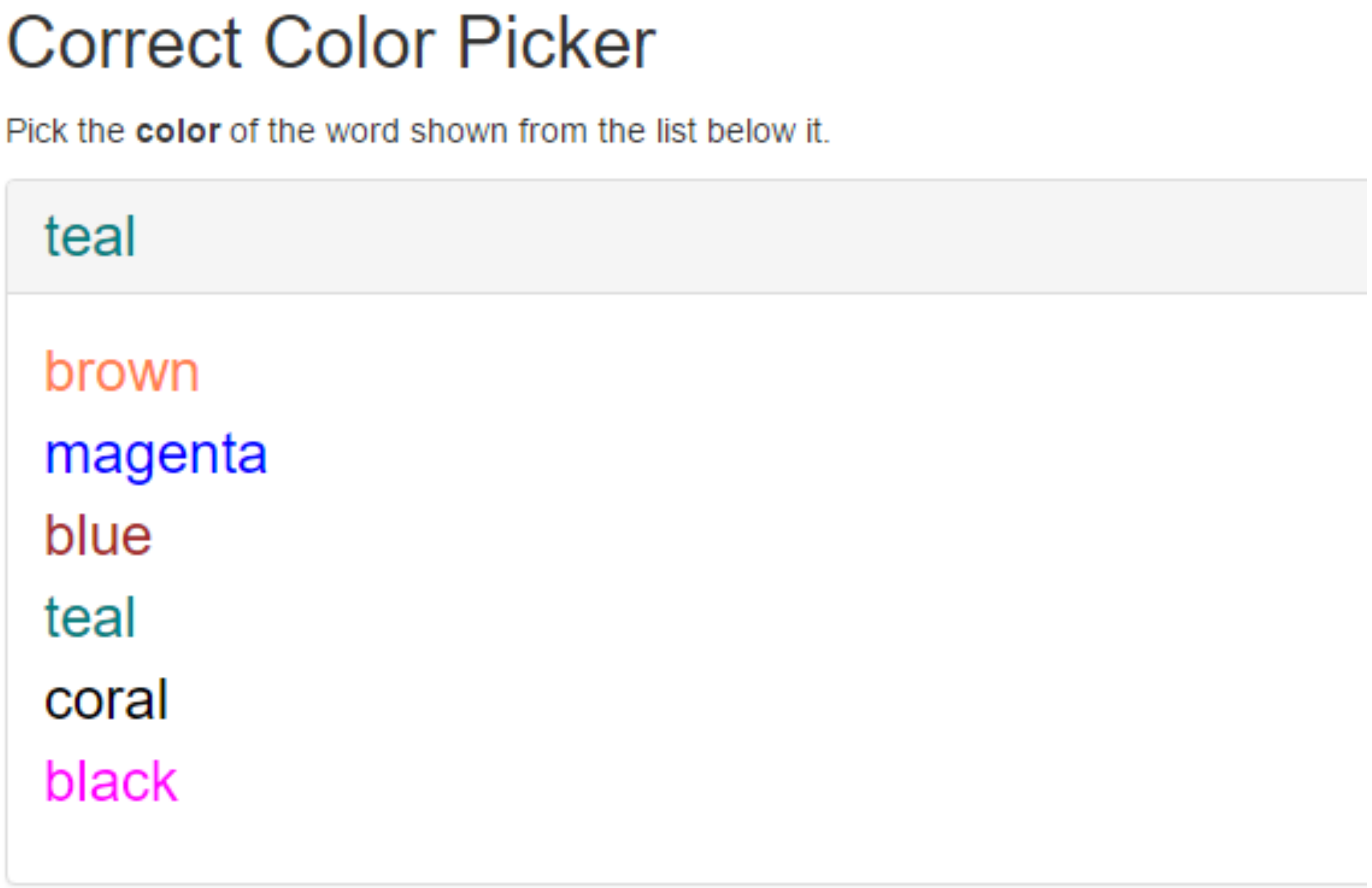
POSTED IN:

● Web Apps

Build a Brain Teaser

(Time Permitting)

Color Picker – Brain Teaser



Correct Color Picker

Pick the **color** of the word shown from the list below it.

teal

brown
magenta
blue
teal
coral
black

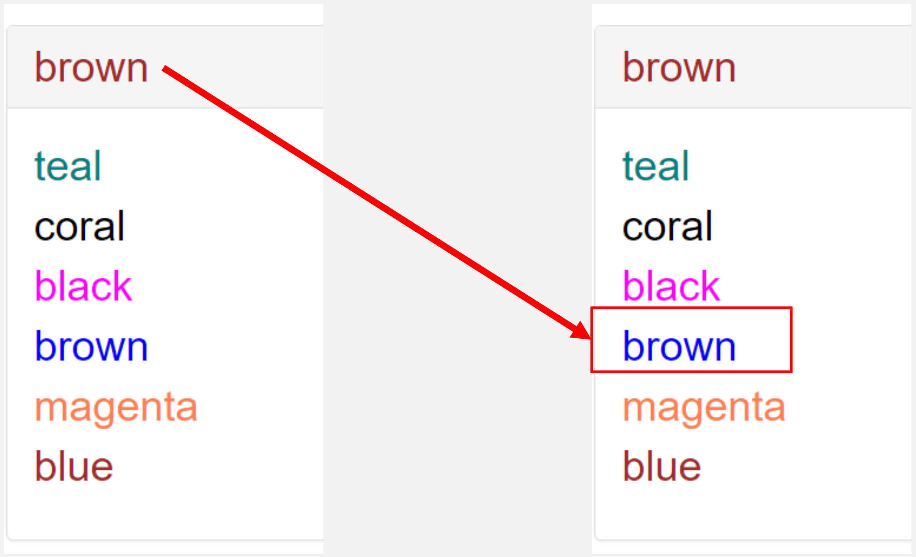
> YOUR TURN!!

Activity: *7-ColorCorrector* **| Suggested Time:** 20 min

- Using the files sent to you as a starting point, add the missing code such that the Color Corrector game works correctly.

- **To win, you should be picking the "word" that matches the color of the text at the top.**

- Ex:



Questions