MODULE ONE - JAVASCRIPT FOUNDATIONS

DEBUGGING YOUR JS & WRITING AWESOME CODE

TODAY'S SCHEDULE

1. Cool Things
2. Concept Review
3. What is Debugging? /Debugging Workflow
4. Order of Execution/ Execution Contexts
5. Types of Errors
6. Tips & Tools
7. Handling Errors Gracefully
8. Code Formatting & Best Practice
9. Recap & Looking Ahead

LEARNING OBJECTIVES - WEEK 3

1. construct a variety of programming structures including variables, constants, arrays, objects, functions, conditionals, and constructors;

2. test and debug scripts using validators, DOM inspectors, and error console tools

3. optimize code for increased functionality, performance, readability, and reusability;

COOL THINGS

RESOURCES, LINKS TUTORIALS AND OTHER COOL THINGS…

- https://www.wappalyzer.com/
- https://diana-adrianne.com/purecss-gaze/
- https://www.smashingmagazine.com/2020/04/particle-trail-animation-javascript/
- https://www.knutmelvaer.no/blog/2020/05/how-i-put-the-scroll-percentage-in-the-title-bar/
- https://thezen.zone/

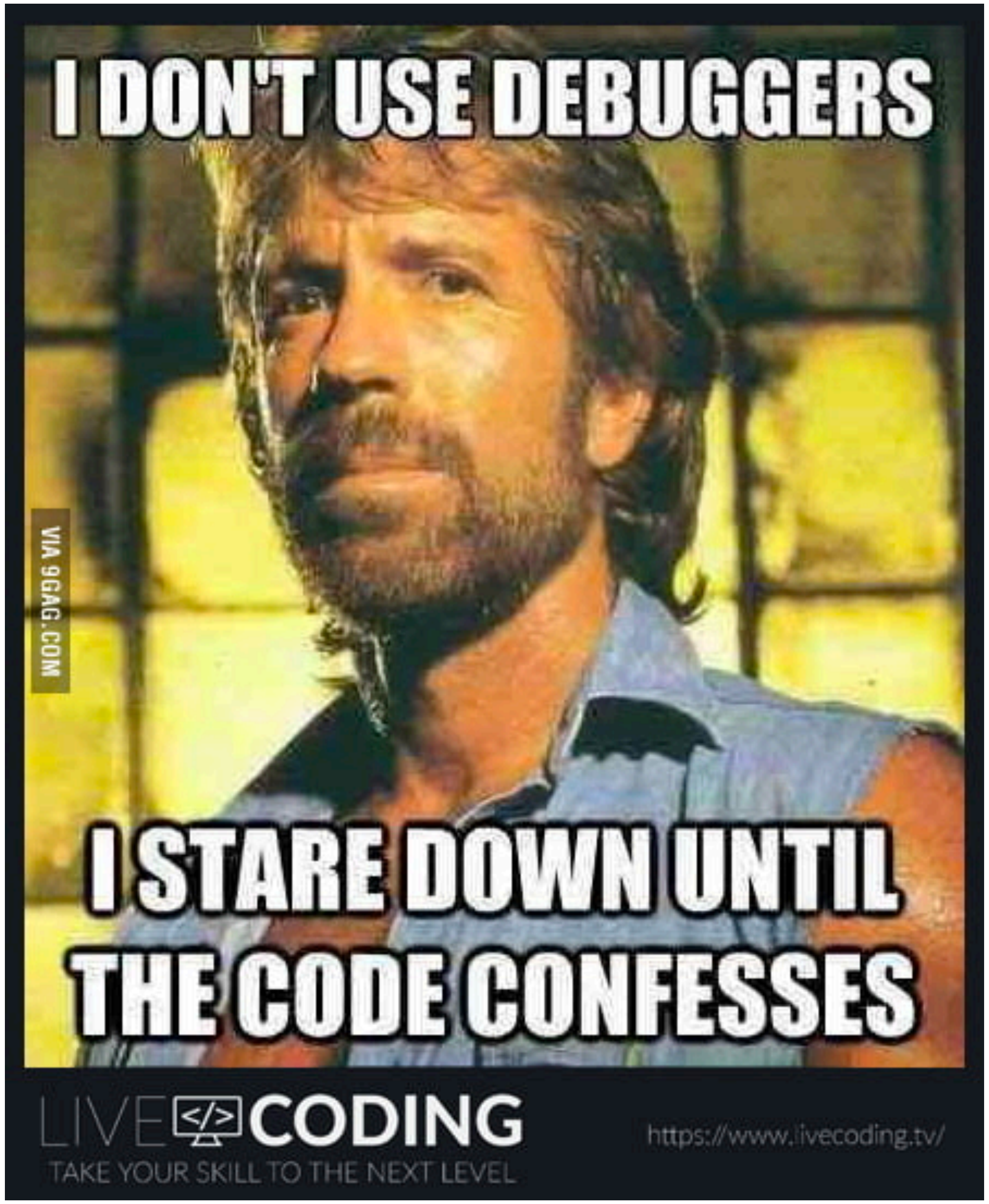
LET'S REVIEW

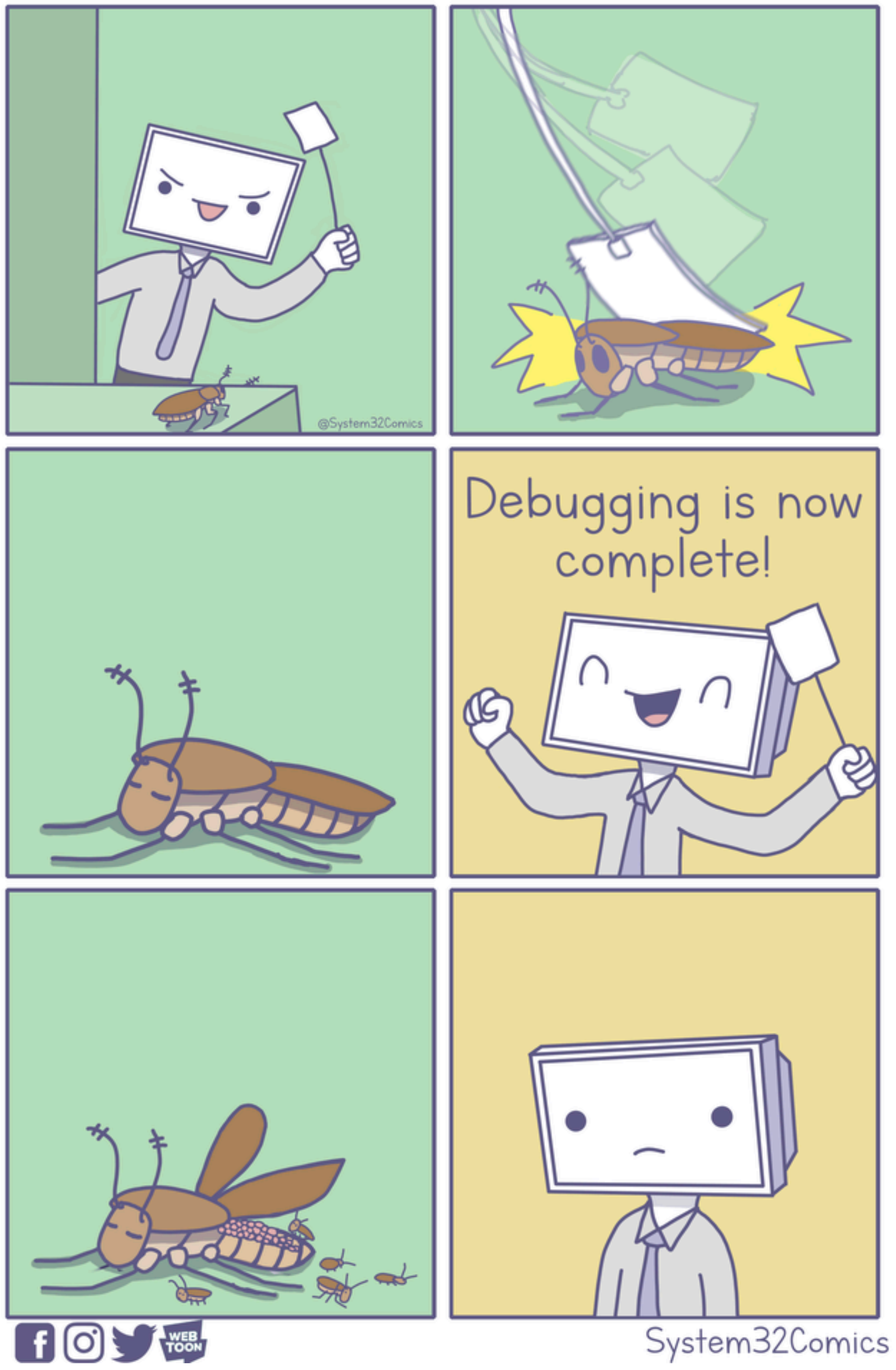
LAST WEEK, WE LEARNED…

- let, var, const
- variable scope
- statements & semicolons
- data types
- functions
- arrays
- loops
- conditional structures/control flow

WHAT IS DEBUGGING?

MY CODE DOESN'T WORK
I HAVE NO IDEA WHY
MY CODE WORKS
I HAVE NO IDEA WHY


I DON'T USE DEBUGGERS
I STARE DOWN UNTIL THE CODE CONFESSES
VIA 9GAG.COM
LIVE CODING
TAKE YOUR SKILL TO THE NEXT LEVEL
https://www.livecoding.tv/


@System32Comics
Debugging is now complete!
System32Comics

"When you are writing Javascript, do not expect to write it perfectly the first time, Programming is like problem solving: you are given a puzzle and not only do you have to solve it, but you also need to create the instructions that allow the computer to solve it too."

(Duckett, JavaScript & JQuery, 2014)

Before we get too far in, it’s important to understand how to debug our code, the right way to write JavaScript and the best practices we should be following. It’s also a good idea to be aware of the tools that are available to help us write quality code and how to use them.

WHAT IS DEBUGGING?...

- Debugging means fixing issues or problems in your code
- Debugging ideally happens in development rather than production (i.e. when your website or application has been launched)
- All modern browsers and most other environments support “debugging” – a special UI in developer tools that makes finding and fixing errors much easier.

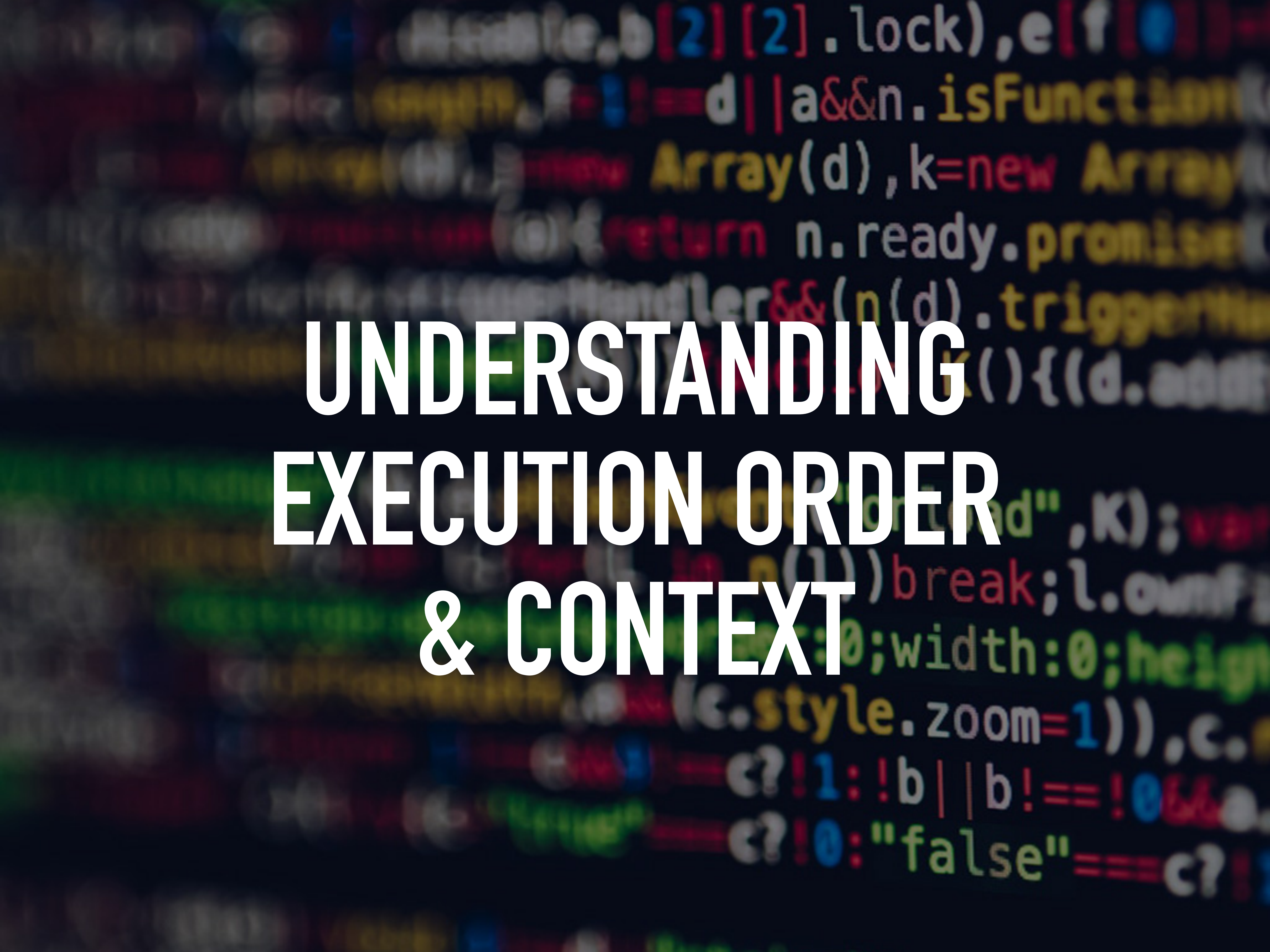
A DEBUGGING WORKFLOW…

1.) Where is the problem?

- Look at error message
- Check how far script is running, write to the console
- Use breakpoints to show where things go wrong

2.) What is the problem?

- Check variables at the set breakpoints to see where values are as expected
- test smaller pieces/units of code (functions, variables, objects)



UNDERSTANDING EXECUTION ORDER & CONTEXT

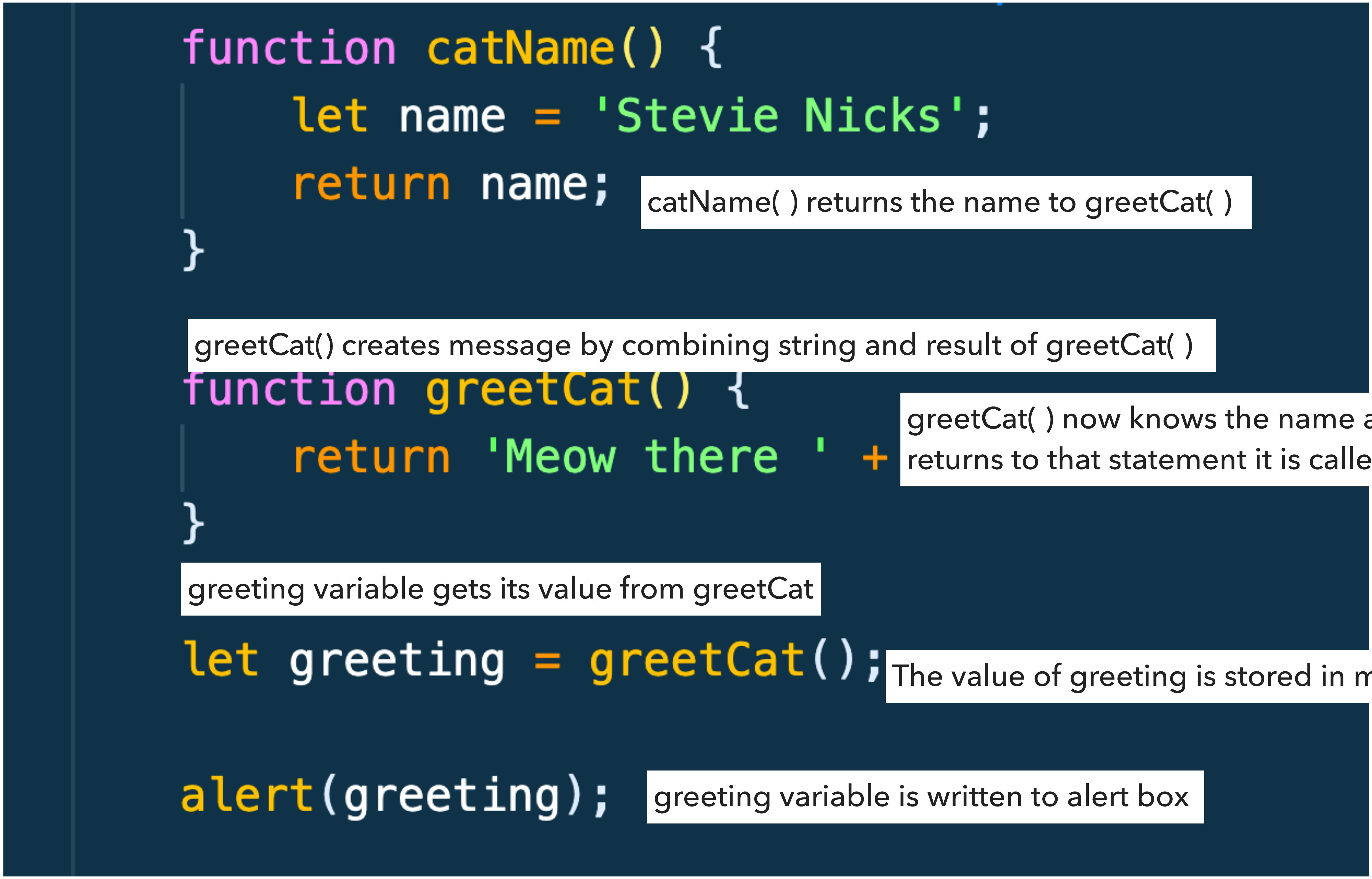
EXECUTION ORDER & CONTEXT

- the order in which statements are executed can be complex, some tasks cannot complete until another statement or function has been run
- understanding the order is important (& helpful when things go wrong!)

EXECUTION ORDER & CONTEXT

- the javascript interpreter uses the concept of execution contexts
- one global execution context
- each function creates a new execution context that corresponds to variable scope.

EXECUTION ORDER & CONTEXT



```
function catName() {
    let name = 'Stevie Nicks';
    return name;
}

function greetCat() {
    return 'Meow there ' +
}

let greeting = greetCat();

alert(greeting);
```

catName( ) returns the name to greetCat( )

greetCat() creates message by combining string and result of greetCat( )

greetCat( ) now knows the name and returns to that statement it is called in

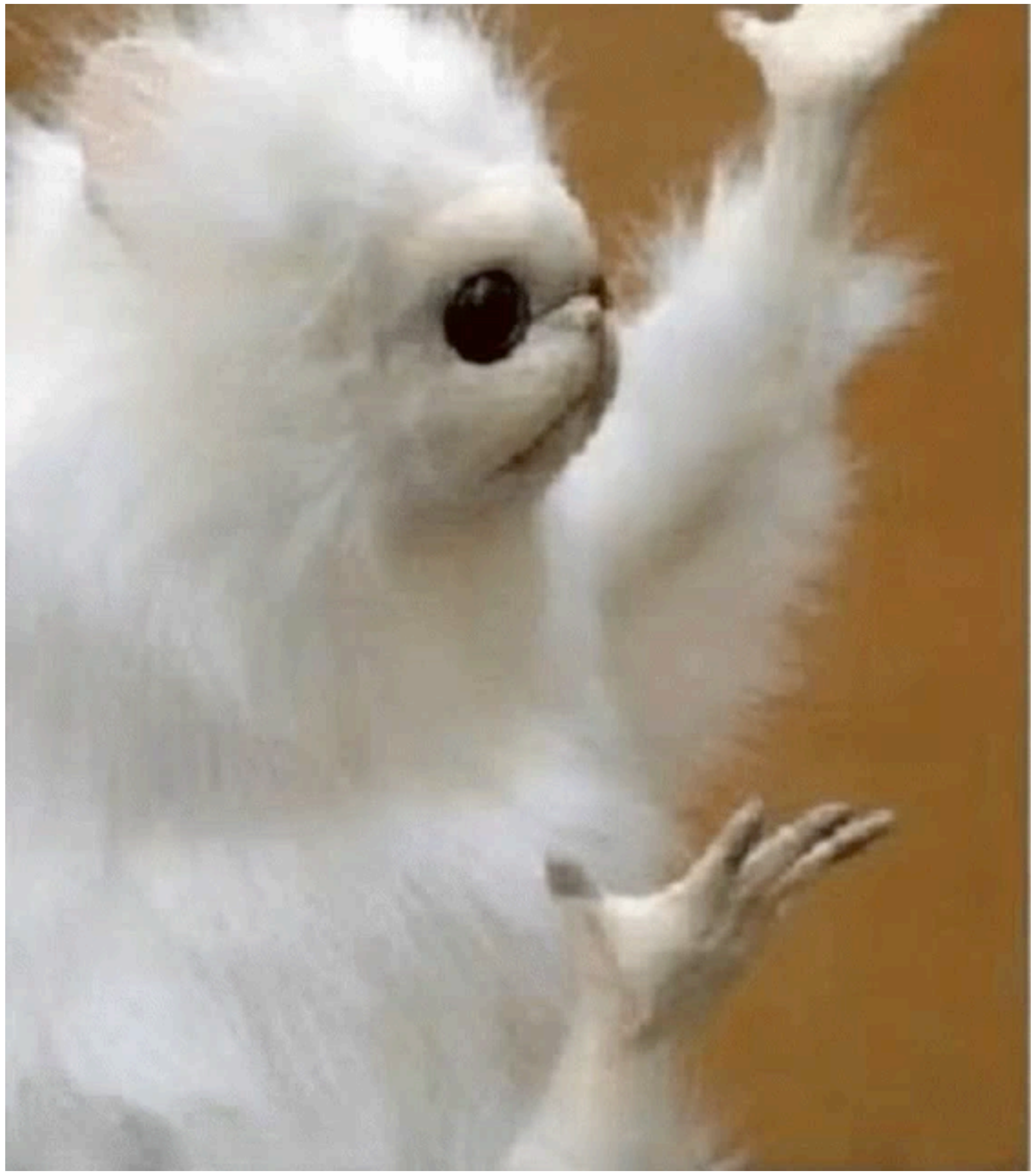
greeting variable gets its value from greetCat

The value of greeting is stored in memory

greeting variable is written to alert box

THE STACK

- JS interpreter processes one line of code at a time
- When a statements needs data from another function, it **stacks** the new function on top of the current task
- Once the new task has been performed, interpreter can go back to the task in hand
- Each time a new task is added to the stack, a new execution context is created



UNDERSTANDING THE CALL STACK:

HTTPS://WWW.YOUTUBE.COM/WATCH?V=W8AEMRVTFLY

EXAMPLE ONE

Module One > Week 3 > Code Examples

TYPES OF ERRORS

TYPES OF ERRORS …

Syntax

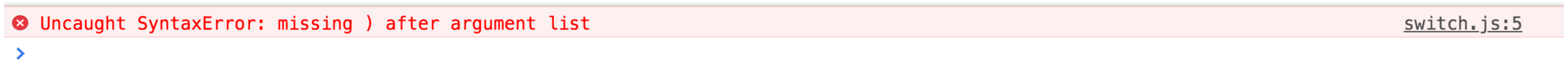
- incorrect use of the language (i.e. a spelling error)

Logic

- the syntax is correct, but the program generates unexpected results

ERROR OBJECTS

- Error - generic error, other errors are based upon this error
- SyntaxError - syntax has not been followed
- ReferenceError - tried to reference a variable that is not declared/within scope
- TypeError - An unexpected data that cannot be coerced
- RangeError - Numbers are not in an acceptable range
- URIError.- encodeURI(), decodeURI() and similar methods used incorrectly
- EvalError - eval() function used incorrectly



```
Uncaught SyntaxError: missing ) after argument list                    switch.js:5
>
```

TIPS & TOOLS

WHAT IF I TOLD YOU
YOUR BROWSER HAS EVERYTHING YOU NEED TO DEBUG JAVASCRIPT
imgflip.com

DEBUGGING TOOLS …

- console.log
- debugger;
- js hint/lint (extensions)
- breakpoints

COMMON ERRORS

- scope issues
- missing semi-colon
- capitalization (case-sensitive)
- mismatching quotes
- spelling mistakes
- missing closing braces
- missing parentheses when testing conditions
- script not loading
- conflicts between scripts

THINGS TO TRY (WHEN THINGS GO WRONG)

- try another browser
- add numbers to see which items are getting logged, shows how far your code runs before the error stops it
- strip it back - remove parts of code and strip it back down to the min you need
- explain the code to someone else
- search - stack overflow
- code playgrounds - js fiddle, js bin
- validation tools - js hint, js lint , JSON lint

NIFTY DEBUGGING TUTORIAL

https://developers.google.com/web/tools/chrome-devtools/javascript

EXAMPLE TWO

Module One > Week 3 > Code Examples

HANDLING ERRORS

HANDLING ERRORS GRACEFULLY…

- ideally, we don't want any errors to happen
- sometimes errors happen beyond our control
- if they do, we want to handle them in a 'graceful' way
- we can do using exception handling

HANDLING ERRORS GRACEFULLY…

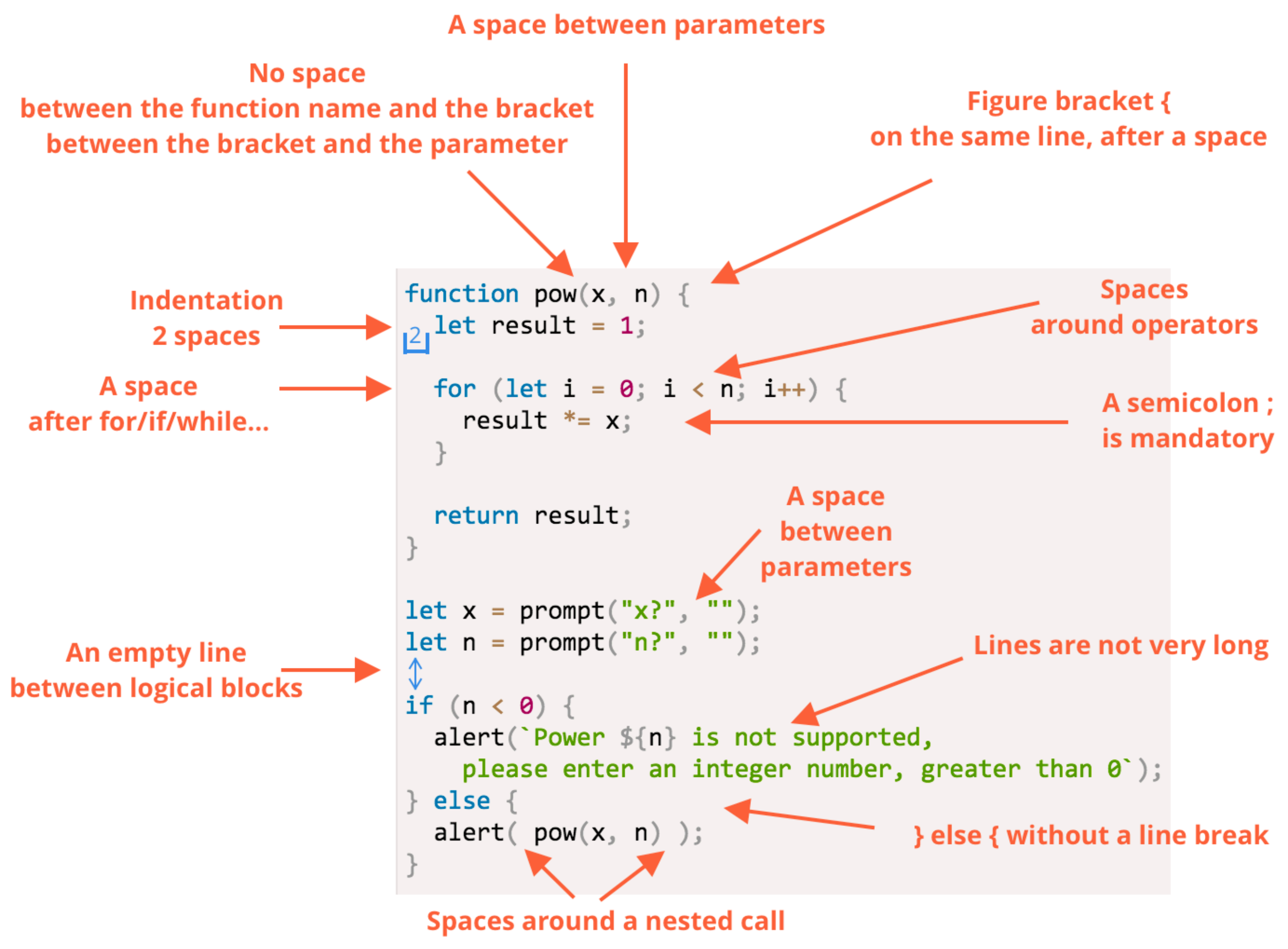
- use try, catch and finally blocks
- Try - try to execute this code
- Catch - If there is an exception, so the
- Finally - this will always be executed (whether try is successful or not)

EXAMPLE THREE

Module One > Week 3 > Code Examples

CODE FORMATTING & QUALITY

WRITING JAVASCRIPT THE RIGHT WAY

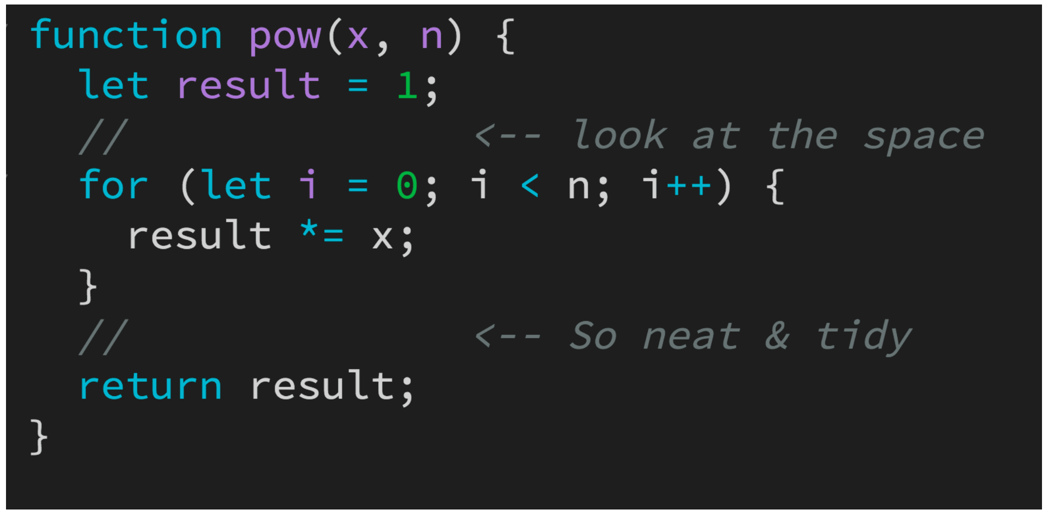


```
function pow(x, n) {
  let result = 1;

  for (let i = 0; i < n; i++) {
    result *= x;
  }

  return result;
}

let x = prompt("x?", "");
let n = prompt("n?", "");

if (n < 0) {
  alert(`Power ${n} is not supported,
    please enter an integer number, greater than 0`);
} else {
  alert( pow(x, n) );
}
```

A space between parameters
No space between the function name and the bracket between the bracket and the parameter
Figure bracket { on the same line, after a space
Indentation 2 spaces
2
Spaces around operators
A space after for/if/while...
A semicolon ; is mandatory
A space between parameters
Lines are not very long
An empty line between logical blocks
} else { without a line break
Spaces around a nested call

https://javascript.info/coding-style

WRITING JAVASCRIPT THE RIGHT WAY

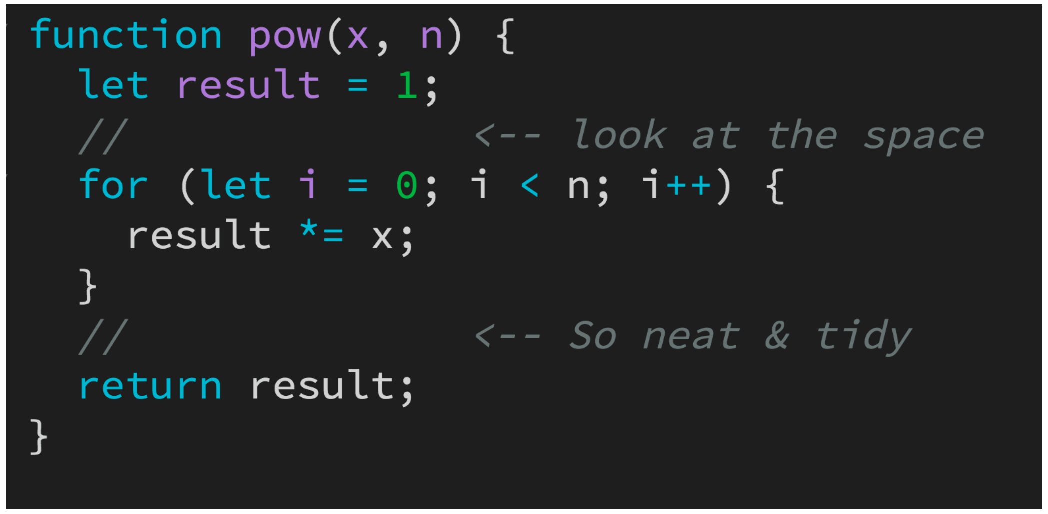


```
function pow(x, n) {
  let result = 1;
  //               <-- look at the space
  for (let i = 0; i < n; i++) {
    result *= x;
  }
  //               <-- So neat & tidy
  return result;
}
```

- A semicolon should be present after each statement, even if it could possibly be skipped.
- Try to avoid nesting code too many levels deep.
- Use a style guide & ensure that your team sticks to it

HELPFUL TOOLS

- https://beautifier.io/
- https://prettier.io/
- Code Editor/IDE extensions



```
function pow(x, n) {
  let result = 1;
  //                <-- look at the space
  for (let i = 0; i < n; i++) {
    result *= x;
  }
  //                <-- So neat & tidy
  return result;
}
```

STYLE GUIDES EXAMPLES

- https://google.github.io/styleguide/javascriptguide.xml
- https://github.com/airbnb/javascript
- https://github.com/rwaldron/idiomatic.js
- https://standardjs.com/

RECAP TIME

RECAP

- sometimes things go wrong in our code
- when there are errors in our code, we need to find them using problem-solving, deduction and various tools available
- we also need to handle errors beyond our control (in a graceful way)
- writing quality code is awesome & there are tools & best practices to follow that can help us do this

WEEKLY TASKS: QUIZ THREE, PRACTICE QUESTIONS

NEXT WEEK : INTRO TO OBJECTS

SOURCES

- https://developer.mozilla.org/en-US/docs/Learn/JavaScript
- https://javascript.info/arrow-functions-basics
- https://javascript.info/coding-style
- Jon Duckett. 2014. JavaScript and JQuery: Interactive Front-End Web Development (1st. ed.). Wiley Publishing.