**UDACITY’s Hackreactor JAVASCRIPT:**

**Git:**

\_Git init . first

\_Config global user.name “bryanh210”

\_Then Git add .

**Scope:**

\_Variable inside the curly braces can have access to those on the outer scope and global scope

\_No var === global

EXECUTION CONTEXT vs Lexical cope:

\_In-memory scope structure built as the program executes. It builds all the ones outside first, and then go in

<https://classroom.udacity.com/courses/ud015/lessons/2593668697/concepts/25411890600923>

\_If you call a function many times, each time it will create a new execution context (memory space in the memory). Will not replace anything

\_Function object: Any object that can be called as if it was a function

THIS:

Q: What is “this” bound to?

A: A function that is looked up when it’s being evoked

\_”This” behaves like parameters

No binding of a parameter until the function gets called

\_If there’s no variable.method form (which ‘this’ will refer to the variable), then it will refer to the global scope

\_To bind “this” to a function: Use .call(parameters), so “this” will be bounded to that function, not the globak object

\_red.method.call(yellow,green,blue) => will give me a yellow, green,blue because .call overrides the “.method” rule

\_If no value is passed to the function for setTimeOut, the output is undefined