

Numbers:	Math Operators:	Array loops	DOM
1, 10.3, 0.2, 1.5e12, infinity	2 + 3 * 10 / 2 10 % 2 // => 0 11 % 2 // => 1	Map takes a function to apply to each element in the array, returns an array of results: arr = [1, 2, 3, 4]; arr.map(double) // => [2, 4, 6, 8] arr.map(x => x * 2) // => [2, 4, 6, 8]	var par = document.getElementById("id"); var elt = document.createElement("div"); par.appendChild(elt); elt.innerText = "Some text!";
Strings:	Boolean Operators:	forEach is map that doesn't return. arr.forEach(function(num) { console.log("The number is", num); })	How to Debug
Double quotes: "hello" Single quotes: 'hi there' Template strings: `there are \${5 + 2} cats`	True when both things to be true: a && b True when at least one is true: a b True when a is false: !a		1. Check for error messages. 2. Think it out; be the computer. 3. Add log statements to investigate. 4. Take a 3 minute break. 5. Google it. 6. Grab a friend. 7. Grab an Expert.
Booleans:	Functions	Classes	How to Build a THING:
true, false	5 ways to make a function: function double(x) { return x * 2; } const anotherDouble = function(x) { return x * 2; } var yetAnotherDouble = x => x * 2; yetAnotherDouble = (x) => x * 2; yetAnotherDouble = (x) => { return x * 2; }; double(2) // 4 Arguments are variables; their value gets set when the function is called. As variables, they're only defined inside the function.	A class is a factory for making objects with useful functions attached. class Student { constructor(name, score) { this.name = name; this.score = score; } grade() { if (this.score >= 90) { return "A"; } else if (this.score >= 80) { return "B"; } else if (this.score >= 70) { return "C"; } else if (this.score >= 60) { return "D"; } else { return "F"; } } }	DON'T START CODING 1. What is the thing? Ask lots of questions. 2. How do we represent the <i>state</i> of the thing? What information is necessary to describe it at one moment in time? 3. How does the state of the thing change in response to user actions?
Kinds of Nothing:			NOW YOU MAY WRITE CODE
undefined, null			4. Build the state representation. Test often. 5. Build the state changes. Test often. 6. Render the state. Make sure it shows the user what they need to know (even if it's ugly right now). 7. Re-render the state when it changes.
Arrays:	for loops		NOW YOU MAY STYLE IT
const arr =[1, 2, 3, "a", "b", "c", ["nested", "array"]] first element of the array: arr[0] the last element: arr[arr.length - 1]	initialize; keep going as long as; after each step, do this for(var i = 0; i < 10; i++) { // runs 10 times. // i starts at 0 and ends at 9. }	const aStudent = new Student("Sam", "10"); aStudent.name // => "Sam" aStudent.grade() // => "F" const bStudent = new Student("Amy", "95"); bStudent.name // => "Sara" aStudent.grade() // => "A"	8. Add lovely, luscious styling.
Objects:	while loops		A haiku by Issa
const obj = { name: "Sam", favoriteNumber: 3, favoriteCheese: "Wensleydale" }; Accessing values: obj["name"] or obj.name	var i = 0; while(i < 10) { // will run 10 times, i starting at 0 i = i + 1; }		cloud becomes a mountain becomes a cloud
Variables			
const [someName] = [somevalue]; Const can only be set once, lasts only as long as the function its in. var [someName] = [somevalue]; Var can be set many times, lasts only as long as the function its in. let [someName] = [somevalue]; Let can be set many times, lasts only as long as the nearest set of {}'s.			