JS LIST METHODS

JAVASCRIPT LIST METHODS

(CHEAT SHEET)

.length

Returns the number of items in an array or list

[1, 2, 3, 4].length

<< 4

.join

Returns a string of all the elements concatenated with the specified argument.
Use template literals to avoid the brackets.

["h", "e", "l", "l", "o"].join``
<< "hello"
["h", "e", "l", "l", "o"].join('')
<< "hello"
["h", "e", "l", "l", "o"].join('-')
<< "h-e-l-l-o"
["h", "e", "l", "l", "o"].join`-`
<< "h-e-l-l-o"

.map()

Returns the same array, with every element having been passed through a specified function.

[1, 2, 3, 4].map((element, index) => element * index))
<< [0, 2, 6, 12] //[1x0, 2x1, 3x2, 4x3]

.filter()

Passes every element through a specified function that tests a condition. Returns an array with all the elements that meet the condition.

[1, 2, 3, 4].filter(element => element%2 === 0)
<< [2, 4] //filters out odd numbers

[1, 3, null, "hi", false].filter(Boolean)
<< [1, 3, "hi"]

IMPORTANT NOTE:

All methods that require functions as arguments have three inbuilt parameters: the element of the list, the index of the current element, and the array itself.

For example,
.map((element, index, array) => *magic*)

The parameters must be defined in that order, and you do not need all of them each time. However, you must have the element.

.includes()

Returns a Boolean value, based on whether the list contains the element specified as an argument.

["h", "e", "l", "l", "o"].includes("h")
<< true
["h", "e", "l", "l", "o"].includes("f")
<< false

.pop()

Returns the last element in a list
["my", "array"].pop()
<< "array"

.push()

Adds a specified element to the end of an array and returns the length of the new array

["h", "e", "l", "l"].push("o")
<< 5

.every()

Passes every element through a specified function that tests a condition. Returns a Boolean value depending on whether every element in the array meets the condition.

[1, 2, 3, 4].every(element => element === 1)
<< false

["o", "o", "o", "o"].every(element => element === "o")
<< true

.indexOf()

Returns the index (position in array) of the first instance of a specified element.

["h", "e", "l", "l", "o"].indexOf("l")
<< 2
["h", "e", "l", "l", "o"].indexOf("k")
<< -1

.some()

Passes every element through a specified function that tests a condition. Returns a Boolean value depending on whether AT LEAST ONE element in the array meets the condition

.reduce()

Passes each element of the list through a function. With every pass, the element contributes to building up a value. This value is defined at the start, and is returned once all elements have been passed through.

[1, 2, 3, 4, 5].reduce((sum, element) => sum + element, 0)
<< 15 // 1 + 2 + 3 + 4 + 5

Note: unlike the other methods that use functions, the initial value must be defined as the parameter first (before element), and defined in a comma after the function.

.find()

Passes each element through a specified function, that tests a certain condition. Returns the first element that meets this condition.

[1, 2, 3, 4].find(element => element > 2)
<< 3

.forEach()

Passes every element through a specified function, that performs an (or multiple) action(s).
This works in place of a 'for' loop, for more inline coding.

[1, 2, 3, 4, 5].forEach(element => console.log(element*element))