JavaScript Notes

# Examples

## Looping

In the following FOR LOOP example we initialize with i = 0 and iterate while our condition i < 5 is true. We'll increment i by 1 in each loop iteration with i++ as our final-expression:

var ourArray = [];

for (var i = 0; i < 5; i++) {

ourArray.push(i);

}

ourArray will now contain [0,1,2,3,4]

Iterate through the contents of an array with a for loop:

var arr = [10,9,8,7,6];

for (var i=0; i < arr.length; i++) {

console.log(arr[i]);

}

**Note:** Arrays have zero-based numbering, which means the last index of the array is length - 1. Our condition for this loop is i < arr.length, which stops when i is at length - 1.

var ourArr = [ 9, 10, 11, 12];

var ourTotal = 0;

for (var i = 0; i < ourArr.length; i++) {

ourTotal += ourArr[i];

}

WHILE LOOP

runs "while" a specified condition is true and stops once that condition is no longer true.

var ourArray = [];

var i = 0;

while(i < 5) {

ourArray.push(i);

i++;

}

Math.random only returns decimal between 1 and 0. Add Math.floor, and multiplication to get it out of small decimal form, to get a larger whole number:

Math.floor(Math.random() \* 20);

Math.floor(Math.random() \* (max - min + 1)) + min

The above code helps you get a number between a defined max and min. Why do you need the max – min + 1 and then + min? Basically, for our ranges to actually be inclusive on max, it is required for the number used in the function to be 1 greater than our actual max. That’s what this bit of code accomplishes.

## Regular Expressions

Used to find certain expressions in strings.

Ex: /the/gi

/ - Begins regular expression.

The – the pattern we are looking to match.

/ - End of regular expression.

/ - Global, so find all matches in the string and not just the first one.

i – Ignore case while performing search.

\d – Retrieve one digit in a string.

It would be like: /the/\d/g

/\d+/g – Allows the regular expression to match one or more digits.

Note: d seems to select the NUMBER of numbers. Ie, a count.

\s – Finds whitespace in a string.

Whitespace characters include “ “ (space) \r (carriage return) \n (newline) \t (tab) \f (form feed). Counts number of white spaces.Ex:

var expression = /\s+/g;

Regular expression matches can be **inverted** by using an uppercase version of a normal selector.

For example, \s matches any whitespace while \S matches anything that’s not whitespace.

# Definitions

Arrays

.pop() – Removes the last element of an array.

.push() – Adds element to the end of an array.

.shift() – Removes the first element of an array.

.unshift() – Adds an element to the beginning of an array.

Math.random() – Generates a random decimal number between 0 (inclusive) and not quite up to 1 (exclusive).

Math.floor() – Round numbers down to a whole number.

Functions

Function ourFunctionName() {

Console.log(“Hello world”);

}

ourFunctionName()

Function functionWithArgs(thing1, thing2) {

Console.log(thing1 + thing2);

}

functionWithArgs(1, 7)

Loops

For loops are declared with three optional expressions separated by semicolons:

for ([initialization]; [condition]; [final-expression])

* Initialization statement is executed one time only before the loop starts.
* Condition statement is evaluated at the start of the iteration and will continue as long as it evaluates to “True.” When condition is “False” it will stop executing.
* Final expression is executed at the end of each loop iteration, prior to the next condition check and is usually used to increment or decrement the loop counter.

# FreeCodeCamp Sample

## Profile Lookup Waypoint

//Setup

**var** contacts **=** **[**

**{**

"firstName"**:** "Akira"**,**

"lastName"**:** "Laine"**,**

"number"**:** "0543236543"**,**

"likes"**:** **[**"Pizza"**,** "Coding"**,** "Brownie Points"**]**

**},**

**{**

"firstName"**:** "Harry"**,**

"lastName"**:** "Potter"**,**

"number"**:** "0994372684"**,**

"likes"**:** **[**"Hogwarts"**,** "Magic"**,** "Hagrid"**]**

**},**

**{**

"firstName"**:** "Sherlock"**,**

"lastName"**:** "Holmes"**,**

"number"**:** "0487345643"**,**

"likes"**:** **[**"Intriguing Cases"**,** "Violin"**]**

**},**

**{**

"firstName"**:** "Kristian"**,**

"lastName"**:** "Vos"**,**

"number"**:** "unknown"**,**

"likes"**:** **[**"Javascript"**,** "Gaming"**,** "Foxes"**]**

**}**

**];**

**function** lookUpProfile**(**firstName**,** prop**){**

// Only change code below this line

**for** **(var** i**=**0**;** i **<** contacts**.**length**;**i**++){**

**if** **(**contacts**[**i**].**firstName **===** firstName**){**

**if** **(**contacts**[**i**].**hasOwnProperty**(**prop**)){**

**return** contacts**[**i**][**prop**];**

**}** **else** **{**

**return** "No such property"**;**

**}**

**}**

**}**

**return** "No such contact"**;**

// Only change code above this line

**}**

// Change these values to test your function

lookUpProfile**(**"Akira"**,** "likes"**);**

[Explanation here](https://forum.freecodecamp.com/t/freecodecamp-challenge-guide-profile-lookup/18259)