Javascript Basics

This will be a beginners javascript cookbook. I will try to organize definitions and commands in a functional way.

Main Definitions and Examples

single line comment /* */:: multi line comment

Variables

var
: Can be local or global.

const
: These variables cannot be redefined

NOTE	whether you define a variable locally or globally is very important with how you can call it

Strings

Appending Variables to strings

strVarName += **secStrVarName**
: strings over multiple lines can be connected out of string litterals. They can be appended using +=

Concatenating strings

strName += **"text to be concatenated"**
: This works the same as appending string variable names.

String Length

variableName.string()
: Find length of string

Bracket Notation: Find characters of strings

Use bracket notation to specify and select a specific char. Just like how you would for strings in c

varName[0]
: number in bracket will be the letter at that position in the char count.

escape characters

\'
: Single quote

\"
: \Double Quote

\\
: Backslash

\n
: New Line

\r
: Carriage return

\t
: Tab

\b
: Word boundary

\f
: Form feed

String Immutability

Individual chars of strings cannot be changed, you can only change the full variable

Math

These operations are used with strings as well.

Compound assignment

Assigns the value of the variable operated on by the other value to the variable.

+=
: a = a + b; === a+=b;

-=
: a = a - b; === a-=b;

***=**
: a = a * b; === a*=b;

/=

  a = a / b; === a/=b;

Arrays

Store several pieces of information in one place

Simple arrays

var arrayName = **["strings in quotes", 3,];**

  Strings in quotes, numbers are normal

Nested or Multidimensional Arrays

[["Text", 3], ["more Text", 5]]

  Arrays can be nested like this.

Array Indexes

arrayName[0];

  This will select the first item in the array.

- Arrays are mutable. By selecting the array index, you can change its value

Access multidimensional arrays with Indexes

arrayName[0][1];

  This selects the 2nd item of the first array.

Array Manipulation

push()

Push a parameter onto the **end** of the array

arrayName.push(1)

  This will add the number 1 to the end of the array

pop()

take a value off the end of an array. This value can be stored and assigned to a variable. This removes the last item * create an variable and assign it to equal the value from the other aray

var newVar = **arrayName.pop();**

  This will make newVar equal the last value of the array, that array will no longer contain that value.

shift()

Just like pop but takes the first item instead of the last.

var newVar = **arrayName.shift();**

  Will make newVar equal the first value of the array *unshift()*

Add elements to the front of the array. Just like push but for the beginning of the array.

arrayName.unshift("Inserted information");

This would make the string "Inserted information the content of arrayName[0]"

Functions

Declare a function to use again

function functionName() {function Content}

This would create a function called functionName().

Returning a value from a function

Return a statement to send a value out of the function. Note that a function does not have to return a value. If return is not set to anything then the function can still be called but the return value is undefined

return "value to be returned, don’t use quotes for numbers";

When called, the function will return whatever value it’s told to.

Assignment with a returned Value

1. set a function return a value
2. assign that value to a variable as follows:

   var variableName = functionName(3){}

   This code is predicated on the function taking a numeric value as an argument. It will process the number 3 and return whatever value it is told to give you relating to the input information.

Arguments to pass values into functions

Make a function take values as arguments

function functionName(arg1){}

arg1 would represent the input information, this can be requested from the user, be a value returned by another function…

Scope

Variables defined as var vs const and inside our outside fo functions will be able to be called.

Global Scope

A variable defined globally can be seen everywhere

.Local Scope

These variables are only visible within its function

NOTE	the same name can be used for different global and local variables. It will prioritize the local variable.