# Code Color Guide

Keyword Object Function Number String Comments

# Variable

*An object that stores a single value that can be changed.*

This is used to store data to be manipulated later. It works like a variable in algebra.

Use the ‘let’ keyword to declare a variable. This gets done once in the program. You cannot use a variable that has not been declared.

let value = 10; // Declare value  
console.log(value); // Log the variable’s value to console

# Constant

*A variable whose value cannot be changed during runtime.*

They are useful for tweaking configurations during development.

Use the ‘const’ keyword to declare a constant in the same way as a variable.

const MAXLIMIT = 3; // Declares MAXLIMIT as a constant  
let counter = 4; // Declares a counter variable  
if (counter > MAXLIMIT) {  
 // Checks to see if counter is greater than MAXLIMIT  
 console.log(‘counter is greater than’ + MAXLIMIT);  
}

# Array

*A single variable that stores multiple values.*

Arrays are declared like a variable except all values are enclosed in square brackets, separated by commas. Elements (values) of an array are accessed by their index number. Index numbers start at 0.

let classmates = [‘Josh’, ‘Noah’, ‘Harry’]; // Declare array as a variable  
console.log(classmates[0]); // Prints the first value in the array  
console.log(classmates[2]); //Prints the third value in the array

# Statement

*An executable line of code.*

It is represented by a rectangle in a flow chart.

# Conditional Statement

*A line of code that executes when a given condition is NOT FALSE.*

This is used for “Program Control Flow”. It is represented by a diamond in a flow chart.

# “If” Statement

*A conditional statement that forks the program flow.*

Called by the keyword ‘if’, followed by a condition in parenthesis **()**, followed by a list of statements in curly braces **{}**.

Can also contain an ‘else if’ and ‘else’ branch. ‘else if’ requires a different condition. ‘else’ executes if no prior conditions were met.

Formerly known as an “if/then” statement and is verbalized as “if/then/else if/then/else”.

let name = ‘Isiah’;  
if (name == ‘Noah’) { // If name is Noah, execute statements in these curly braces  
 console.log(‘Hello, Noah! My neighbor.’);  
} else if (name == ‘Harry’) { // If name is not Noah, but IS Harry…  
 // Then say hello to Harry instead  
 console.log(‘Hello, Harry, who sits three rows away’);  
} else { //Else if no other previous conditions were met  
 // Then execute these statements  
 console.log(‘I have no idea who you are’);  
}

# while loop

*A conditional statement that loops through a series of statements if the condition is NOT FALSE.*

It is good for an unknown number of loops.

Called by the keyword ‘while’, followed by a condition in parenthesis **()**, followed by a list of statements in curly braces **{}**.

let looper = true;  
while (looper) {  
 console.log(‘Condition was NOT FALSE, so I did the loop.’);  
 looper = false; // Make looper false, so we won’t repeat loop  
}

# do while loop

*A while statement that runs the loop* ***before*** *checking the condition to run again.*

It is good for an unknown number of loops, where loop must run at least once.

Called by the keyword ‘do’, followed by a list of statements in curly braces **{}**. It is then followed by the ‘while’, followed by the condition.

let looper = false;  
do {  
 console.log(‘This gets logged once, even though the condition is false’);  
}  
while (looper);

# Functions

*A repeatable group of statements.*

Functions can take multiple arguments (parameters). Functions can also return a value as though it were a variable.

Functions are declared with the keyword ‘function’, followed by the function name, and parameter names in parenthesis (). Parameters are separated by commas. A list of statements to be executed go inside of curly braces {}. The return statement is declared with the keyword ‘return’.

function newFunction(argA, argB) {  
 console.log(argA);  
 console.log(argB);  
 let c = argA + argB;  
 return c;  
}  
console.log(newFunction(1, 2));

# Objects

*An object is an identifiable “thing” in Javascript. Everything in Javascript is technically an object.*

Objects have properties and methods. Properties are variables that belong to the object, whereas methods are functions that belong to the object.

Objects are written in JSON format (Javascript Object Notation). Everything inside of curly braces **{}** is part of the object. Objects are made up of smaller objects called attributes (objects, properties and methods), which are defined by an attribute name followed by a colon (**:**), and separated by commas (**,**).

You can access an object’s attributes by putting the object’s name and a period in front of the attribute name. Objects refer to their own scope with the ‘*this*’ keyword.

let Person = {  
 // This is a property attribute  
 name: ‘Bob’,  
 // This is a method attribute  
 sayName: function() {  
 console.log(‘My name is ‘ + this.name);  
 }  
}  
  
// Log the value of the ‘name’ property  
console.log(Person.name)  
// Call the function of the ‘sayName’ attribute  
Person.sayName()