**Comments**

//in-line comment

/\* Block comment \*/

**Data Types & Variables**

*undefined, null, Boolean, string, symbol, number, object*

*var myName = “Kev”;*

*myName = 8;*

*let ourName = “Kevin”;*

*const pi = 3.14;*

**Assignment**

*var a;*

*var b = 2;*

*a = 7;*

*b = a; console.log(a)*

**Initializing Variable**

*var a = 9;*

**Numbers**

*var sum = 10 + 0;*

*var difference = 45 - 33;*

*var product = 8 \* 0;*

*var quotient = 66 / 0;*

*myVar = 0;*

*myVar = myVar + 1; myVar++;*

*myVar = myVar – 1; myVar--;*

*var myDecimal = 5.7;*

*var product = 2.0 \* 0.0;*

*var quotient = 4.4/2.0;*

*var remainder = 11%3;*

**String**

*firstName = “Kevin”;*

*lastName = “Tran”;*

*var myStr = “I am a \“double quoted\” string inside \“double quotes\””;*

*myStr = '<a href="http://example.com" target="\_blank">Link</a>';*

*firstName.length; 🡪 5*

*firstName[0] 🡪 K*

*firstName[0] = “T”; //Cannot do b/c strings are immuatable*

*firstName = “Tevin”; //This is allowed though*

**Escape Sequences in Strings**

*\’ single quote*

*\” double quote*

*\\ backslash*

*\n newline*

*\r carriage return*

*\t tab*

*\b backspace*

*\f form feed*

**Concatenation**

*var outstr = “I come first ” + “I come second”;*

**Functions Javascript Review**

*function wordBlanks(myNoun, myAdjective, myVerb, myAdverb){*

*var result = “”;*

*result += "The " + myAdjective + " " + myNoun + " " + myVerb + " to the store " + myAdverb + "."}*

*console.log(worldBlanks(“dog”, “big”, “ran”, “quickly”));*

*🡪The big dog ran to the store quickly.*

**Arrays**

*var ourArray = [“Kevin”, 23];*

//Nested array

*var myArray = [[“The universe”, 42], [“everything”, 2]];*

*var array2 = [1, 2, 3]; array2[0] 🡪 1*

*array2[0] = 30; 🡪 [30, 2, 3]*

//Array of arrays

*myArray = [[1,2,3], [4,5,6], [7,8,9], [[10, 11, 12], 13, 14]]];*

*var myData = myArray[0,0]; 🡪 1*

//Push/Pop

*var ourArray = [“K”, “J”, “cat”];*

*ourArray.push([“happy”, “joy”]);*

*var removedElement = ourArray.pop(); //Removes last element*

*ourArray.shift(); //Removes first element*

*overArray.unshift(“Happy”); //Adds to beginning of the array*

**If Statements**

*if(isTrue){*

*console.log(“Yes it’s true”);*

*}*

*if(val == 12){*

*//true*

*}*

*function compareEquality(a, b){*

*if(a == b){*

*return “Equal”;*

*}*

*return “Not equal”;*

*}*

*console.log(compareEquality(10, “10”));*

*a == b will convert the string to an int*

*a === b will not convert the types*

*|| &&*

**Else Statement**

*if(val > 10){*

*return “Greater than 10”;*

*} else if (val < 5){*

*return “Less than 5”;*

*} else{*

*return “Between 5 and 10”;*

*}*

**While Loop**

*var i = 0;*

*while(i < 4){ do{*

*i++; i++*

*} }while(i<5)*

**For Loop**

for(var i = 0; i < 4; i++){

}

**Switch Statement**

val = 1;

switch(val){

case 1:

answer = “alpha”

break;

case 2:

answer = “beta”;

break;

default:

answer = “N/A”

break;

}

**Objects**

*var ourDog = {*

*“name”: “Camper”,*

*“legs”: 4,*

*“tails”: 1,*

*“friends”: [“everything!”],*

*};*

*var testObj = {*

*“hat”:”ballcap”,*

*“shirt”:”jersey”,*

*“shoes”:”cleats”*

*};*

*var hatValue = testObj.hat;*

*var shirtValue = testObj.shirt;*

*var shoesValue = testObj[“shoes”];*

*var myDog = {*

*“name”: “Camper”,*

*“legs”: 4,*

*“tails”: 1,*

*“friends”: []*

*}*

myDog.name = “Happy”;

//Adding new properties

myDog[‘bark’] = “woof!”

//Delete properties

delete myDog.bark;

//Check property

var myObj = {

gift: “pony”,

pet: “kitten”,

bed: “sleigh”

};

function checkObj(checkProp){

return myObj.hasOwnProperty(checkProp);

}

**Random**

*var randomNumBet0and19 = Math.floor(Math.random()\*20);*

*//Random number between myMin & myMax, inclusive*

*function ourRandomRange(myMin, myMax){*

*return Math.floor(Math.random()\*(myMax – myMin + 1)) + myMin;*

*}*

ourRandomRange(1, 9);

**Conversion**

*parseInt(“42”);*

*function convertToInteger(str){*

*return praseInt(str, 2);*

*}*

*convertToInteger(“10011”); //binary number*

**Ternary Operator**

*//condition ? statement-if-true : statement-if-false;*

*function checkEqual(a, b){*

*return a===b ? true:false;*

*//return a===b; 🡨 same result*

*}*

*checkEqual(1,2);*

*function checkSign(num){*

*return num > 0 ? “positive”: num < 0 ? “negative” : “zero”;*

*}*

*console.log(checkSign(0));*

**Difference between Let & Var, Const**

*//Starting with ES6 you can use let and const*

*let only allows you to declare once*

*scope of let is limited to block statement or expression declared in*

*function checkScope(){*

*“use strict”;*

*var i = “function scope”;*

*if(true){*

*i = “block scope”;*

*console.log(i);*

*}*

*console.log(i);*

*return i;*

*}*

* *block scope block scope*

*function checkScope(){*

*“use strict”;*

*let i = “function scope”;*

*if(true){*

*let i = “block scope”;*

*console.log(i);*

*}*

*console.log(i);*

*return i;*

*}*

* *block scope function scope*

*//Const cannot be reassigned, arrays can be reassigned by index*

*const WORD = “Cool”;*

**Arrow Functions**

*var myConcat = (arr1, arr2) => {*

*return arr1.concat(arr2);*

*};*

*const myConcat = (arr1, arr2) => arr1.concat(arr2);*

*console.log(myConcat([1,2],[3,4,5]));*

* *[1,2,3,4,5]*

**Rest Operator**

*const sum = (function(){*

*return function sum(…args){*

*//const args = [x,y,z];*

*return args.reduce(a,b)=>a+b,0);*

*};*

*})();*

*console.log(sum(1,2,3));*

**Destructuring**

*const [z, x, , y] = [1, 2, 3, 4, 5, 6, 7, 8, 9];*

*console.log(z, x, y); 🡪 1, 2, 4*

*let a = 8, b = 6;*

*(() => {*

*“use strict”;*

*[a, b] = [b, a];*

*})();*

*console.log(a, b); 🡪 6, 8*

*const source = [1,2,3,4,5,6,7,8,9,10];*

*function removeFirstTwo(list){*

*const [a, b, …arr] = list;*

*return arr;*

*}*

*const arr = removeFirstTwo(source);*

*console.log(arr); 🡪 [1,2,3,4,5,6,7,8,9,10]*

*console.log(source); 🡪 [3,4,5,6,7,8,9,10]*

**Template Literals**

*const person = {*

*name: “Zodiac”,*

*age: 56*

*};*

*const greeting = `Hello, my name is ${person.name}! I am ${person.age} years old.`;*

*console.log(greeting);*

**Object Function**

*const bicycle = {*

*gear: 2,*

*setGear(newGear){*

*“use strict”;*

*this.gear = newGear;*

*}*

*};*

*bicycle.setGear(3);*

**Class Syntax**

*var SpaceShuttle = function(targetPlanet){*

*this.targetPlanet = targetPlanet;*

*}*

*var zeus = new SpaceShuttle(‘Jupiter’);*

*console.log(zeus.targetPlanet);*

*class(targetPlanet){*

*constructor(targetPlanet){ =*

*this.targetPlanet;*

*}*

*}*

*var zeus = new SpaceShuttle(‘Jupiter’);*

*function makeClass(){*

*class Vegetable{*

*constructor(name){*

*this.name = name;*

*}*

*}*

*return Vegetable;*

*}*

*const Vegetable = makeClass();*

*const carrot = new Vegetable(‘carrot’);*

*console.log(carrot.name);*

**Private Variables in Class**

*class Book{*

*constructor(author){*

*this.\_author = author;*

*}*

*//getter*

*get writer(){*

*return this.\_author;*

*}*

*//setter*

*set writer(updatedAuthor){*

*this.\_author = updatedAuthor;*

*}*

*}*

*function makeClass(){*

*class Thermostat{*

*constructor(temp){*

*this.\_temp = 5/9\*(temp-32);//Convert to C*

*}*

*get temperature(){*

*return this.\_temp;*

*}*

*set temperature(updatedTemp){*

*this.\_temp = updatedTemp;*

*}*

*}*

*return Thermostat;*

*}*

*const Thermostat = makeClass();*

*const thermos = new Thermostat(76);*

*let temp = thermos.temperature; //Property doesn’t have ()*

*thermos.temperature = 26;*

*temp = thermos.temperature;*

**Import & Export**

*export const capitalizeString = str => str.toUpperCase();*

*import { capitalizeString } from “./string\_function.js”;*

*const cap = capitalizeString(“Hello!”);*

*//Import everything*

*import \* as capitalizeString from “./string\_funciton”;*