**JavaScript Reference Sheet**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| If statements | if | if (*condition*) {   //  block of code to be executed if the condition is true;} |
|  | If-else | if (*condition*) {   //  block of code to be executed if the condition is true;} else {    //  block of code to be executed if the condition is false;} |
|  | If-if else-else | if (*condition1*) {   //  block of code to be executed if condition1 is true} else if (*condition2*) {   //  block of code to be executed if the condition1 is false and condition2 is true } else {   //  block of code to be executed if the condition1 is false and condition2 is false} |
|  | switch | switch(expression) {   case x:    *// code block*     break;   case y:    *// code block*     break;   default:     // code block } |
| for Loops | for | For (let i = 0; i < condition.length; i++) {    do something} |
|  | for-in | for (key in object) {   // *code block to be executed* }—loops through keys |
|  | forEach | arrayName.forEach(myFunction);--call a function or  array1.forEach(element => console.log(element)); |
|  | for-of | for (variable of iterable) {   // *code block to be executed* }—loops through values |
|  | while | while (condition) { *// code block to be executed* }  while (i < 10) {   text += "The number is " + i;   i++; } |
|  | do/while | do { *// code block to be executed* } while (condition);  do {   text += "The number is " + i;   i++; } while (i < 10); |
| Functions | Declaration | function myFunction(p1, p2) {   return p1 \* p2;   // The function returns the product of p1 and p2 } |
|  | Arrow | hello = () => {  return "Hello World!"; }  or  hello = () => "Hello World!";  const double2 = value => value \* 2;  double2(10); |
|  | expression | const count = function(array) { // Function expression  return array.length;  } |
|  | shorthand |  |
|  | class | class ClassName {   constructor() { ... } }  class Car {   constructor(name, year) {     this.name = name;     this.year = year;   } } |
|  | Accessing class | const redCar = new Car(Ford, 1971);--example |
| Arrays | literal | const array\_name = [item1, item2, ...]; |
|  |  | const cars = new Array("Saab", "Volvo", "BMW"); |
|  |  | const points = new Array(); const points = []; |
| Object |  | const person = {firstName:"John", lastName:"Doe", age:46}; |
|  |  | const person = {  name: ['Bob', 'Smith'],  age: 32,  gender: 'male',  interests: ['music', 'skiing'],  bio: function() {  alert(this.name[0] + ' ' + this.name[1] + ' is ' + this.age + ' years old. He likes ' + this.interests[0] + ' and ' + this.interests[1] + '.');  },  greeting: function() {  alert('Hi! I\'m ' + this.name[0] + '.');  }  }; |
| Object | Properties | properties: *objectName.propertyName or objectName["propertyName"]* |
| Method |  | person.name = function () {   return this.firstName + " " + this.lastName; };---with “this” keyword |
|  |  |  |
| class |  | class Rectangle {  constructor(height, width) {  this.height = height;  this.width = width;  }  } |
|  | Calling classes | Const square = new Rectangle(10,10); |
|  |  |  |
|  |  | document.getElementById("hikes") |
|  |  |  |
|  |  |  |
| Modules | export | Export const PI = 3.1415926; (from a file called pi.js) |
|  | Import | Import {PI} from ‘.pi.js’: |
|  |  | Import \* as newname from ‘.file\_name.js’; |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| DOM |  |  |
|  |  | start: document.getElementById('start'),  score: document.querySelector('#score strong'), |
|  |  |  |
|  |  |  |
|  |  |  |
| Event Listeners |  | document.getElementById("myBtn").addEventListener("click", displayDate); |
|  |  | element.addEventListener("click", function(){ alert("Hello World!"); }); |
|  |  | someElement.addEventListener('touchstart', process\_touchstart, false); |
|  |  |  |
| Append |  | document.body.appendChild(ul); |

I