Javascript Variables

New variables in JavaScript are declared using the var keyword:

**var a;**

**var name = "Hassan";**

If you declare a variable without assigning any value to it, its type is undefined. An important difference from other languages like Java is that in JavaScript, blocks do not have scope; only functions have scope. So if a variable is defined using var in a compound statement (for example inside an if control structure), it will be visible to the entire function.

Javascript Operator

JavaScript's numeric operators are +, -, \*, / and % - which is the remainder operator. Values are assigned using =, and there are also compound assignment statements such as += and -=. These extend out to x = x operator y.

**x += 5**

**x = x + 5**

You can use ++ and -- to increment and decrement respectively. These can be used as prefix or postfix operators. The + operator also does string concatenation:

**"hello" + " world"**

**> hello world**

If you add a string to a number (or other value) everything is converted in to a **string first**. This might catch you up:

**"3" + 4 + 5**

**>345**

**3 + 4 + "5"**

**>75**

Adding an empty string to something is a useful way of converting it.

Arrays

Arrays in JavaScript are actually a special type of object. They work very much like regular objects (numerical properties can naturally be accessed only using [] syntax) but they have one magic property called 'length'. This is always one more than the highest index in the array.The old way of creating arrays is as follows:

**var a = new Array();**

**a[0] = "dog";**

**a[1] = "cat";**

 **a[2] = "hen";**

 **a.length**

**>3**

A more convenient notation is to use an array literal:

**var a = ["dog", "cat", "hen"];**

 **a.length**

**>3**

Leaving a trailing comma at the end of an array literal is inconsistent across browsers, so don't do it. Note that array.length isn't necessarily the number of items in the array. Consider the following:

**var a = ["dog", "cat", "hen"];**

**a[100] = "fox";**

**a.length**

**>101**

Remember — the length of the array is one more than the highest index.

Javascript Object

JavaScript objects can be thought of as simple collections of name-value pairs, the fact that this data structure is so widely used is a testament to its versatility. Since everything in JavaScript is an object. The "name" part is a JavaScript string, while the value can be any JavaScript value — including more objects. This allows you to build data structures of arbitrary complexity.

**There are two basic ways to create an empty object:**

**var obj = new Object();**

**var obj = {};**

These are semantically equivalent; the second is called object literal syntax, and is more convenient. This syntax is also the core of JSON format and should be preferred at all times. Once created, an object's properties can again be accessed in one of two ways:

obj["name"] = "Hassan"; Write in the Property

var name = obj["name"]; Read From the Property

**Object literal syntax can be used to initialise an object in its entirety:**

var obj = {

 name: "Carrot",

 "for": "Max",

 details: {

 color: "orange",

 size: 12

 }

**Attribute access can be chained together:**

**obj.details.color**

**> orange**

**obj["details"]["size"]**

**> 12**