Handout

Introduction to JavaScript

Lesson 3 - Step 2

M. David Green

Math Operations

Now let's take a deeper dive into the operators that deal with arithmetic.

Addition

I already showed you the addition operator, so you saw how a plus sign can be used for adding numbers or for adding or concatenating strings together. The plus operator when applied to numbers just adds them together. When applied to variables that are strings behaves a little bit differently: it concatenates the strings together, and creates a string that contains all of the different strings as one string.

Subtraction

Subtraction only works with numbers.

```
var smallNumber = 1;
var largeNumber = 1000;
var firstWord = "hello";
var secondWord = "world";

console.log(largeNumber - smallNumber);
// 999

console.log(firstWord - secondWord);
// NaN
```

JavaScript won't report an error if you try to subtract the strings, but the result that comes out might not be what you expect. Similarly, multiplication only works with numbers.

Multiplication

```
var smallNumber = 2;
var largeNumber = 3;
var firstWord = "hello";
var secondWord = "world";

console.log(largeNumber * smallNumber);
// 6

console.log(firstWord * smallNumber);
// NaN

console.log(firstWord * secondWord);
// NaN
```

So note that you can't use multiplication with string variables.

Division

Division only works with numbers as well.

```
var smallNumber = 2;
var largeNumber = 8;
var firstWord = "hello";
var secondWord = "world";

console.log(largeNumber / smallNumber);
// 4

console.log(firstWord / smallNumber);
// NaN

console.log(firstWord / secondWord);
// NaN
```

Modulus (Remainder)

Modulus is an operator that returns the remaining value after a division of numbers. There are many situations in programming where figuring out modulus can be convenient. For example, if you want to find every third item in a list you can use the modulus operator to determine when you've reach the third item so that you can do something special with it.

```
var smallNumber = 4;
var largeNumber = 10;

console.log(largeNumber / smallNumber);
// 2.5

console.log(largeNumber % smallNumber);
// 2
```

Increment and Decrement Variable

Increment and decrement are represented by double plus signs or double minus signs. They're placed right next to the variable that you want to modify and they either add or subtract one from the numerical value of that variable.

```
var smallNumber = 4;

console.log(smallNumber++);
// 4

console.log(smallNumber);
// 5

console.log(smallNumber--);
// 5

console.log(smallNumber);
// 4

var smallNumber = 4;

console.log(++smallNumber);
// 5

console.log(smallNumber);
// 5

console.log(--smallNumber);
// 4

console.log(smallNumber);
```

```
// 4
```

The increment and decrement operators are basically syntactic sugar, that means a little convenience for programmers that lets them quickly add one or subtract one to a variable, because that's something programmers need to do frequently.

Unary Operators

Another operator you probably haven't come across unless you've done some programming before is the unary operator. Unary operators derive a positive or negative numerical value from a number, from a string that represents a number, from a boolean value or even from a `null` variable.

```
var smallNumber = 4;
var stringNumber = "3";
var stringWithNumber = "33 Cherry Tree Lane";
var truth = true;
var falsehood = false;
var notDefined;
var setToNull = null;

console.log(-smallNumber);
// -4

console.log(+stringNumber);
// 3

console.log(+stringWithNumber);
// NaN

console.log(+truth);
// 1

console.log(+falsehood);
// 0

console.log(+notDefined);
// NaN

console.log(+setToNull);
// 0
```

Using the unary operators, you can convert any variable either into a number or into its not a number value, if it's impossible to convert it to a number in JavaScript.

Precedence

Another topic to keep in mind when working with arithmetic operators in JavaScript is precedence. Precedence is the order in which operations occur and operations in an expression in JavaScript are usually performed left to right.

- The top precedence is given operations that happen inside of parentheses within an expression.
- Next, the increment and unary operators are applied.
- Next multiply, divide and modulus are applied.
- And finally, add and subtract.A

```
var smallNumber = 4;
var mediumNumber = 8;
var largeNumber = 1000;

console.log(smallNumber + mediumNumber * largeNumber);
// 8004

console.log((smallNumber + mediumNumber) * largeNumber);
// 12000

console.log(smallNumber++ * mediumNumber);
```

```
// 32 (and smallNumber is now 5)

console.log(largeNumber / --smallNumber);
// 250 (and smallNumber is now 4)

console.log(largeNumber / (smallNumber++ * smallNumber));
// 50 (and smallNumber is now 5)
```

It's worth taking some time to experiment a little bit with the rules of precedence. Go into one of your favorite testing tools and try a few of these out. See if you can predict what the result is going to be consistently, so that you're sure that you know what the rules of precedence are when doing arithmetic with JavaScript numbers.