**Number to string conversion**

var n = 17;

binary\_string = n.toString(2); // Evaluates to "10001"

octal\_string = "0" + n.toString(8); // Evaluates to "021"

hex\_string = "0x" + n.toString(16); // Evaluates to "0x11"

**When working with financial or scientific data, you may want to convert numbers to Strings**

var n = 123456.789;

n.toFixed(0); // "123457"

n.toFixed(2); // "123456.79"

n.toFixed(5); // "123456.78900"

n.toExponential(1); // "1.2e+5"

n.toExponential(3); // "1.235e+5"

n.toPrecision(4); // "1.235e+5"

n.toPrecision(7); // "123456.8"

n.toPrecision(10); // "123456.7890"

**Number function with string argument**

parseInt("3 blind mice") // => 3

parseFloat(" 3.14 meters") // => 3.14

parseInt("-12.34") // => -12

parseInt("0xFF") // => 255

parseInt("0xff") // => 255

parseInt("-0XFF") // => -255

parseFloat(".1") // => 0.1

parseInt("0.1") // => 0

parseInt(".1") // => NaN: integers can't start with "."

parseFloat("$72.47"); // => NaN: numbers can't start with "$"

parseInt() accepts an optional second argument specifying the radix (base) of the

number to be parsed. Legal values are between 2 and 36. For example:

parseInt("11", 2); // => 3 (1\*2 + 1)

parseInt("ff", 16); // => 255 (15\*16 + 15)

parseInt("zz", 36); // => 1295 (35\*36 + 35)

parseInt("077", 8); // => 63 (7\*8 + 7)

parseInt("077", 10); // => 77 (7\*10 + 7)

**tostring() method with different object**

({x:1, y:2}).toString() // => "[object Object]"

[1,2,3].toString() // => "1,2,3"

(function(x) { f(x); }).toString() // => "function(x) {\n f(x);\n}"

/\d+/g.toString() // => "/\\d+/g"

new Date(2010,0,1).toString() // => "Fri Jan 01 2010 00:00:00 GMT-0800 (PST)"

**valueOF() with date() function returns number of milliseconds since 1 january 1970**

var d = new Date(2010, 0, 1); // January 1st, 2010, (Pacific time)

d.valueOf() // => 1262332800000

**Date() Object to string or number conversion**

var now = new Date(); // Create a Date object

typeof (now + 1) // => "string": + converts dates to strings

typeof (now - 1) // => "number": - uses object-to-number conversion

**Variable declaration**

var truevar = 1; // A properly declared global variable, nondeletable.

We cannot delete this var with delete operator

fakevar = 2; // Creates a deletable property of the global object.

this.fakevar2 = 3; // This does the same thing.

delete truevar // => false: variable not deleted

delete fakevar // => true: variable deleted

delete this.fakevar2 // => true: variable deleted