**Alerts**

An alert is a box that pops up to give the user a message. For displaying a message "Thanks for your input!", we have to write following code: alert("Thanks for your input!");

Remember **alert** is a keyword i.e a word that has special meaning for JavaScript. Execution of the above code will display the message in alert box with the message that is enclosed in the quotes.

In coding, the quoted text "Thanks for your input!" is called a text string or simply a String. In simple, it is just a string of characters enclosed in quotes.

JavaScript ignores spaces (except in text strings).

Its better to write spaces in your coding for clarity semicolon is used to declare ending of the statement. A semicolon isn't always necessary, but it's easier to end every statement with a semicolon, rather than stop to figure out whether you need one.

window.alert

**Variables for Strings**

a variable is a word that refers to a particular value.

A variable is created when you write var (for variable) followed by the name that you

choose to give it. It takes on a particular value when you assign the value to it. This is a

JavaScript statement that creates the variable name and assigns the value "Mark" to it.

var name = "Mark";

var is the keyword that creates a variable—

the keyword that declares it. Once a variable has been declared, you don't have to declare it

again. You can just assign the new value to it.

You can declare a variable in one statement, leaving it undefined. Then you can assign a

value to it in a later statement, without declaring it again

**Variables for Numbers**

A string isn't the only thing you can assign to a variable. You can also assign a number

Unlike a string, a number is not enclosed in quotes

how does JavaScript know it's not a variable?

Well, because a number, or any combination of characters starting with a number, can't be used

as a variable name. If it's a number, JavaScript rejects it as a variable. So it must be a number.

If you enclose a number in quotation marks, it's a string. JavaScript can't do addition on it.

It can do addition only on numbers not enclosed in quotes.

var originalNum = 23;

2 var newNum = originalNum + 7;

var originalNum = 23;

2 var numToBeAdded = 7;

3 var newNum = originalNum + numToBeAdded;

var originalNum = 23;

2 var numToBeAdded = 7;

3 var newNum = originalNum + numToBeAdded;

**Variable Names Legal and Illegal**

You can't enclose it in

quotation marks. The name can't be a number or start with a number. It can't be any of

JavaScript's keywords—the special words that act as programming instructions, like alert

and var.

Here are the rest of the rules:

A variable name can't contain any spaces.

A variable name can contain only letters, numbers, dollar signs, and underscores.

Though a variable name can't be any of JavaScript's keywords, it can contain keywords.

For example, userAlert and myVar are legal.

Capital letters are fine, but be careful. Variable names are case sensitive. A rose is not a

Rose. If you assign the string "Floribundas" to the variable rose, and then ask JavaScript

for the value assigned to Rose, you'll come up empty.

I teach the camelCase naming convention. Why "camelCase"? Because there is a hump or

two (or three) in the middle if the name is formed by more than one word. A camelCase

name begins in lower case. If there's more than one word in the name, each subsequent

word gets an initial cap, creating a hump. If you form a variable name with only one

word, like response, there's no hump. It's a camel that's out of food. Please adopt the

camelCase convention. It'll make your variable names more readable, and you'll be less

likely to get variable names mixed up.

Make your variable names descriptive, so it's easier to figure out what your code means

when you or someone else comes back to it three weeks or a year from now. Generally,

userName is better than x, and faveBreed is better than favBrd, though the shorter names

are perfectly legal. You do have to balance readability with conciseness, though.

bestSupportingActressInADramaOrComedy is a model of clarity, but may be too much

for most of us to type or read. I'd shorten it.

**Math expressions:**

**Familiar operators**

Wherever you can use a number, you can use a math expression

**+**, **-**, **\***, and **/**, **%** is the modulus operator. It doesn't give you the result of dividing one number by

another. It gives you the remainder when the division is executed.

If one number divides evenly into another, the modulus operation returns 0. In the

following statement, 0 is assigned to the variable.

num++;

This is a short way of writing...

num = num + 1;

It increments the variable by 1.

You decrement using minuses instead of pluses.

num--;

If you place the pluses before the variable, you get a different result.

1 var num = 1;

2 var newNum = ++num;

In the statements above, both num and newNum wind up with a value of 2.

If you put the minuses after the variable, the new variable is assigned the original value,

and the first variable is decremented.

1 var num = 1;

2 var newNum = num--;

num is decremented to 0, and newNum gets the original value of num, 1.

But if you put the minuses before the variable, newNum is assigned the decremented, not

the original, value. Both num and newNum wind up with a value of 0.

1 var num = 1;

2 var newNum = --num;

alert("Thanks, " + userName + "!");

Using the plus operator, the code combines—concatenates—three elements into the

message: the string "Thanks, " plus the string represented by the variable userName plus the

string "!"

Note that the first string includes a space. Without it, the alert would read,

"Thanks,Susan!".

You can concatenate any combination of strings and variables, or all strings or all

variables. For example, I can rewrite the last example this way.

1 var message = "Thanks, ";

2 var banger = "!";

3 alert(message + userName + banger);

Here it is, with three strings.

alert("Thanks, " + "Susan" + "!");

You can assign a concatenation to a variable.

1 var message = "Thanks, ";

2 var userName = "Susan";

3 var banger = "!";

4 var customMess = message + userName + banger;

5 alert(customMess);

If you put numbers in quotes, JavaScript concatenates them as strings rather than adding

them. This code...

alert("2" + "2");

...displays the message "22".

If you mix strings and numbers...

alert("2 plus 2 equals " + 2 + 2);

...JavaScript automatically converts the numbers to strings, and displays the message "2

plus 2 equals 22".

32

**Prompts**

A prompt box asks the user for some information and provides a response field for her

answer.

This code asks the user the question "Your species?" and provides a default answer in the

response field, "human". She can change the response. Whether she leaves the default response

as-is or changes it to something else, her response is assigned to the variable.

var spec = prompt("Your species?", "human");

Prompt code is like alert code, with two differences.

In a prompt, you need a way to capture the user's response. That means you need to start

by declaring a variable, followed by an equal sign.

In a prompt, you can specify a second string. This is the default response that appears in

the field when the prompt displays. If the user leaves the default response as-is and just

clicks **OK**, the default response is assigned to the variable. It's up to you whether you

include a default response.