

Chapter
2

**LANGUAGE
ELEMENTS**

*Get on the
Fast Track!*



TM

**SYS-ED/
COMPUTER
EDUCATION
TECHNIQUES, INC.**

Objectives

You will learn:

- C Handling events.
- C Expressions.
- C Values and variables.
- C Types of variables.
- C Declaring variables.
- C Arithmetic operators.
- C Concatenation.
- C Assignments.
- C Comparisons.

1 Object Oriented Language

JavaScript is an object-oriented language.

JavaScript utilizes:

C	Objects	C	Properties	C	Methods
---	---------	---	------------	---	---------

Objects

An object is some kind of a thing.

With JavaScript, the objects it deals with in Web browsers include windows, forms, and the elements of the form, such as buttons and check boxes.

It is much easier to keep track of the different objects in scripts when they are given names instead of numbers.

Properties

Objects have properties.

With JavaScript, a window has a title, and a form has a check box.

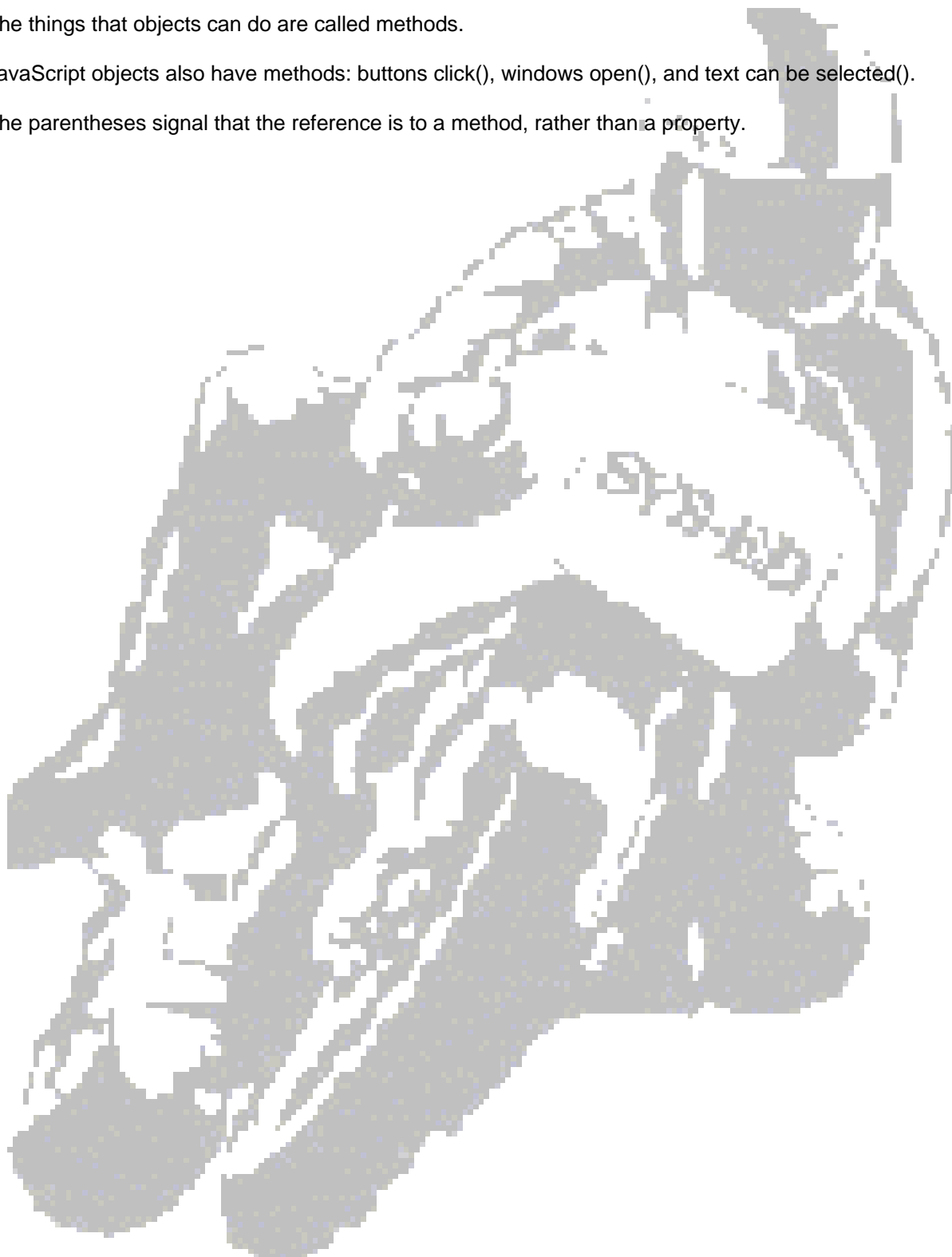
Properties can modify objects, and the same property can apply to completely different objects.

Methods

The things that objects can do are called methods.

JavaScript objects also have methods: buttons `click()`, windows `open()`, and text can be `selected()`.

The parentheses signal that the reference is to a method, rather than a property.



2 Putting the Pieces Together

Objects, properties, and methods can be put together in order to get a better description of an object, or to describe a process.

In JavaScript, these are separated by periods. This is called dot syntax.

Examples of objects and their properties:

C	bicycle.wheels
C	cat.paws.front.left
C	computer.disk.floppy
C	document.images.name
C	window.status

Examples of objects and methods written in dot syntax:

C	cat.purr()
C	document.write()
C	forms.elements.radio.click()

3 Handling Events

Events are mostly caused by user actions.

- C If the user clicks on a button a Click-event occurs.
- C If the mousepointer moves across a link a MouseOver-event occurs.

A JavaScript program reacts to certain events by using event-handlers.

The event-handler for a Click-event is called `onClick`. This tells the computer what to do if this event occurs.

The following code shows an easy example of the event-handler `onClick`. An action by the user on the page triggers an event handler in your script.

In JavaScript, if the user clicks on a button, the `onClick()` event handler will take note of the action and perform whatever duties it was assigned.

When a script is written, it is not necessary to anticipate every possible action that the user might take. The script will only respond to those actions specified by the programmer to occur.

For instance, a page will load just fine without an `onload()` event handler. But the `onload()` command would be used for triggering a script as soon as the page loads.

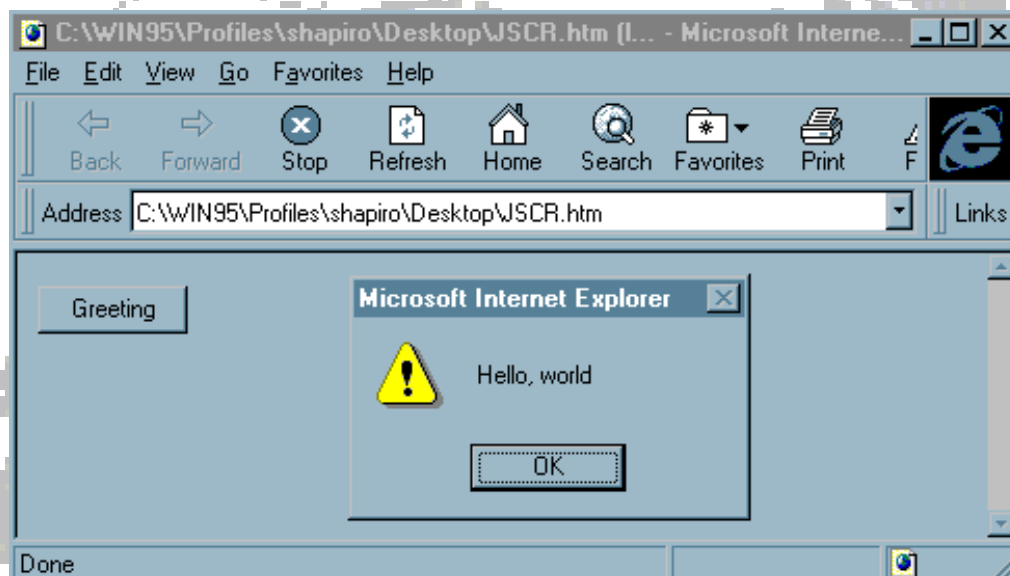
There are twelve JavaScript event handlers:

Event	What it Handles
onAbort	The user aborted loading the page.
OnBlur	The user left the object.
onChange	The user changed the object.
onClick	The user clicked on an object.
onError	The script encountered an error.
onFocus	The user made an object active.
onLoad	The object finished loading.
onMouseOver	The cursor moved over an object.
onMouseOut	The cursor moved off an object
onSelect	The user selected the content of an object.
onSubmit	The user submitted a form.
onUnload	The user left the window.

Example:

```
<form>  
<input type="button" value="Greeting" onClick="alert('Hello, world')">  
</form>
```

- C The form is created with a button.
- C The `onClick="alert('Hello, world')"` inside the `<input>` tag defines what happens when the button is pushed.
- C If a Click-event occurs the computer shall execute `alert('Hello, world')`.
- C `alert()` lets you create popup windows. Inside the brackets you have to specify a string.



A similar popup window can be created with the `prompt()` method. This window accepts an input.

A script can imitate a system message and ask for a certain password.

The text in the popup window shows that the window comes from your web browser and not from your operating system.



4 Expressions

An expression is any valid set of literals, variables, operators, and expressions that evaluates to a single value.

The value can be a number, a string, or a logical value.

There are two types of expressions:

- C those that assign a value to a variable.
- C those that simply have a value.

The expression `x = 7` is an expression that assigns `x` the value seven. This expression itself evaluates to seven. Such expressions use assignment operators.

On the other hand, the expression `3 + 4` simply evaluates to seven; it does not perform an assignment. The operators used in such expressions are referred to as operators.

JavaScript has the following types of expressions:

Arithmetic	Evaluates to a number, for example 3.14159.
String	Evaluates to a character string, for example, "Sys-Ed" or "234".
Logical	Evaluates to true or false.

The special keyword `null` denotes a null value.

In contrast, variables that have not been assigned a value are undefined and cause a runtime error if used as numbers or as numeric variables.

Array elements that have not been assigned a value, however, evaluate to `false`.

For example, the following code executes the function `myFunction` because the array element is not defined:

```
myArray=new Array()  
if (!myArray["notThere"])  
    myFunction()
```

5 Values and Variables

In JavaScript, a piece of information is a value.

There are different kinds of values; the kind you're most familiar with are numbers. A string value is a word or words enclosed in quotes.

Variables contain values.

For example, the variable `myCity` is assigned the string `"New York"`. Another way to write this is `myCity = "New York"`.

The equals sign can be read as "is set to." In other words, the variable `myCity` now contains the value `"New York"`.

Case Sensitivity

JavaScript is case sensitive.

This means that `mycity` is not the same as `myCity`, and either is the same as `MyCity`.

Naming Variables

When naming a variable:

- C a variable name can contain only letters and numbers.
- C spaces or other punctuation marks are not allowed.
- C a variable can't be one of the JavaScript reserved words.

5.1. Types of Variables

JavaScript can have variables of following types:

Type	Description	Example
Number	Any numeric value.	3.1416
String	Character Inside quote marks.	"New York"
Boolean	True or false.	False
Null	Empty and meaningless.	
Object	Any value associated with the object.	
Function	Value returned by a function.	

6 Declaring Variables

Although not required, it is considered good practice to declare variables before using them. This is done by using the `var` statement.

The only time that the `var` statement must be used is when declaring variables that are local to a function.

At all other times, using the `var` statement to declare variables before their use is a recommended practice.

The following code examples are of variable declaration:

```
var mim = "A man, a plan, a canal, Panama!"; // String type.  
var ror = 3; // Numeric type.  
var nen = true; // Boolean type.  
var fif = 2.718281828 // Numeric type.
```

When a variable is to be declared and initialized, but without giving it any particular value, a special value of `null` can be assigned.

```
var zaz = null;  
var notalot = 3 * zaz; // At this point, notalot becomes 0.
```

If a variable is declared without assigning any value to it, it exists but is undefined.

```
var godot;  
var waitingFor = 1 * godot;    // Places the value NaN in waitingFor  
                               // as godot is undefined.
```

A variable can be declared implicitly, without using var, by assigning a value to it.

A variable that has never been declared at all cannot be used. To do so generates an error at runtime.

7 Arithmetic Operators

Operators are the symbols used to work with variables.

Basic Arithmetic Operators

The basic operators supported in JavaScript are:

Operator	What it Does
$x + y$ (Numeric)	Adds x and y together.
$x + y$ (String)	Concatenates text x and text y together Concatenating "mega" and "bytes" produces "megabytes".
$x - y$	Subtracts y from x .
$x * y$	Multiplies x and y together.
x / y	Divides x by y .
$x \% y$	Modulus of x and y . Modulus is the remainder when x is divided by y .

Compound Operators and Unary

Operator	What it Does
x++, ++x	Add one to x. Same as $x = x + 1$.
x--, --x	Subtracts one from x. Same as $x = x - 1$.
-x	Reverses the sign on x.

Although, both `x++` and `++x` add one to `x` they are not identical; the former increments `x` after the assignment is complete and the latter before.

For example, if `x` is 5, `y = x++` results in `y` being set to 5 and `x` set to 6, while `y = ++x` results in both `x` and `y` set to 6. The operator `--` works similarly.

7.1. Concatenation

The `+` operator is used to concatenate two string or a string to a number.

`"New" + " " + "York"` produces `New York`.

If a numeric and string values are mixed together when adding two values together, the result is a string.

For example, `"Room " + 304` results in `"Room 304"`.

7.2. Assignments

When a value is put into a variable, that value is being assigned to the variable. An assignment operator is used for performing this.

For example, the equals operator is used to do an assignment, such as `sSchool = "Sys-Ed"`.

Other than the equals sign, the other assignment operators serve as shortcuts for modifying the value of variables.

Assignment	What it Does
<code>x = y</code>	Sets <code>x</code> to the value of <code>y</code> .
<code>x += y</code>	Same as <code>x = x + y</code> .
<code>x -= y</code>	Same as <code>x = x - y</code> .
<code>x *= y</code>	Same as <code>x = x * y</code> .
<code>x /= y</code>	Same as <code>x = x / y</code> .
<code>x %= y</code>	Same as <code>x = x % y</code> .

7.3. Comparisons

Comparing the value of one variable with another, or the value of a variable against a literal value (ie., a value typed into the expression) is a frequently performed operation.

For example, comparing the value of the day of the week to "Tuesday," can be performed by checking if `todayDate == "Tuesday"`.

Comparison	What it Does
<code>x == y</code>	Returns true if x and y are equal.
<code>x != y</code>	Returns true if x and y are not equal.
<code>x > y</code>	Returns true if x is greater than y.
<code>x >= y</code>	Returns true if x is greater than or equal to y.
<code>x < y</code>	Returns true if x is less than y.
<code>x <= y</code>	Returns true if x is less than or equal to y.
<code>x && y</code>	Returns true if both x and y are true.
<code>x y</code>	Returns true if either x or y are true.
<code>!x</code>	Returns true if x is false.

When comparing strings, be aware that "a" is greater than "A" and that "abracadabra" is less than "be".