**Programming notes for web based development**

**Artcat- sample coding**

**tools**

1. ConEmu- operating system Console emulator
2. Atom- development editor
3. Chrome- browser
4. Zoomit- enlarging font
5. Github - repository

**Command prompt/ConEMU commands**

1. cd – change directory
2. dir – list directory
3. mkdir – create directory
4. git
   1. git clone – copy
   2. git status- validation of change status
   3. git add – add changes/validate changes:

*Note: on the command to add the changes, use ‘–m “nature of change”’, to save the set of the generic update map from coming up.*

* 1. git commit- apply changes
  2. git push- push changes to repository :

(copy site name to clipboard for input on push command)

* 1. (future: git pull)

**websites**

1. [github](http://www.github.com/hadera)- repository
2. [elad on github](http://www.github.com/eladrk)- shared repository
3. [w3schools](http://www.w3schools.com/)- tutorial

**html tags**

*Note: it is recommended that all tag names be in lower case***.**

1. Html— core sheet block
2. Body- body of html executable that is visible.
3. Script- execution of named java script
4. P- paragraph
5. Img- picture
6. h2- heading format 2
7. head- heading format
8. title- title format.
9. Main- main program/procedure logic
10. Figure – formats/selects/logic for putting on screen

**javascript**

1. alert- command for creating an alert on the executed page.

**Core members of executable.**

1. Index.html- core shell code for the application
2. App.js- application intelligence
3. License- generic license
4. Images- sample data source
5. Readme.md- generic notes
6. .Gitignore- git created executable?

**Appendix 1: Tags and definitions**

|  |  |
| --- | --- |
| **Tag** | **Description** |
| [<!--...-->](http://www.w3schools.com/tags/tag_comment.asp) | Defines a comment |
| [<!DOCTYPE>](http://www.w3schools.com/tags/tag_doctype.asp) | Defines the document type |
| [<a>](http://www.w3schools.com/tags/tag_a.asp) | Defines a hyperlink |
| [<abbr>](http://www.w3schools.com/tags/tag_abbr.asp) | Defines an abbreviation or an acronym |
| [<acronym>](http://www.w3schools.com/tags/tag_acronym.asp) | Not supported in HTML5. Use <abbr> instead. Defines an acronym |
| [<address>](http://www.w3schools.com/tags/tag_address.asp) | Defines contact information for the author/owner of a document |
| [<applet>](http://www.w3schools.com/tags/tag_applet.asp) | Not supported in HTML5. Use <embed> or <object> instead. Defines an embedded applet |
| [<area>](http://www.w3schools.com/tags/tag_area.asp) | Defines an area inside an image-map |
| [<article>](http://www.w3schools.com/tags/tag_article.asp) | Defines an article |
| [<aside>](http://www.w3schools.com/tags/tag_aside.asp) | Defines content aside from the page content |
| [<audio>](http://www.w3schools.com/tags/tag_audio.asp) | Defines sound content |
| [<b>](http://www.w3schools.com/tags/tag_b.asp) | Defines bold text |
| [<base>](http://www.w3schools.com/tags/tag_base.asp) | Specifies the base URL/target for all relative URLs in a document |
| [<basefont>](http://www.w3schools.com/tags/tag_basefont.asp) | Not supported in HTML5. Use CSS instead. Specifies a default color, size, and font for all text in a document |
| [<bdi>](http://www.w3schools.com/tags/tag_bdi.asp) | Isolates a part of text that might be formatted in a different direction from other text outside it |
| [<bdo>](http://www.w3schools.com/tags/tag_bdo.asp) | Overrides the current text direction |
| [<big>](http://www.w3schools.com/tags/tag_big.asp) | Not supported in HTML5. Use CSS instead. Defines big text |
| [<blockquote>](http://www.w3schools.com/tags/tag_blockquote.asp) | Defines a section that is quoted from another source |
| [<body>](http://www.w3schools.com/tags/tag_body.asp) | Defines the document's body |
| [<br>](http://www.w3schools.com/tags/tag_br.asp) | Defines a single line break |
| [<button>](http://www.w3schools.com/tags/tag_button.asp) | Defines a clickable button |
| [<canvas>](http://www.w3schools.com/tags/tag_canvas.asp) | Used to draw graphics, on the fly, via scripting (usually JavaScript) |
| [<caption>](http://www.w3schools.com/tags/tag_caption.asp) | Defines a table caption |
| [<center>](http://www.w3schools.com/tags/tag_center.asp) | Not supported in HTML5. Use CSS instead. Defines centered text |
| [<cite>](http://www.w3schools.com/tags/tag_cite.asp) | Defines the title of a work |
| [<code>](http://www.w3schools.com/tags/tag_code.asp) | Defines a piece of computer code |
| [<col>](http://www.w3schools.com/tags/tag_col.asp) | Specifies column properties for each column within a <colgroup> element |
| [<colgroup>](http://www.w3schools.com/tags/tag_colgroup.asp) | Specifies a group of one or more columns in a table for formatting |
| [<datalist>](http://www.w3schools.com/tags/tag_datalist.asp) | Specifies a list of pre-defined options for input controls |
| [<dd>](http://www.w3schools.com/tags/tag_dd.asp) | Defines a description/value of a term in a description list |
| [<del>](http://www.w3schools.com/tags/tag_del.asp) | Defines text that has been deleted from a document |
| [<details>](http://www.w3schools.com/tags/tag_details.asp) | Defines additional details that the user can view or hide |
| [<dfn>](http://www.w3schools.com/tags/tag_dfn.asp) | Represents the defining instance of a term |
| [<dialog>](http://www.w3schools.com/tags/tag_dialog.asp) | Defines a dialog box or window |
| [<dir>](http://www.w3schools.com/tags/tag_dir.asp) | Not supported in HTML5. Use <ul> instead. Defines a directory list |
| [<div>](http://www.w3schools.com/tags/tag_div.asp) | Defines a section in a document |
| [<dl>](http://www.w3schools.com/tags/tag_dl.asp) | Defines a description list |
| [<dt>](http://www.w3schools.com/tags/tag_dt.asp) | Defines a term/name in a description list |
| [<em>](http://www.w3schools.com/tags/tag_em.asp) | Defines emphasized text |
| [<embed>](http://www.w3schools.com/tags/tag_embed.asp) | Defines a container for an external (non-HTML) application |
| [<fieldset>](http://www.w3schools.com/tags/tag_fieldset.asp) | Groups related elements in a form |
| [<figcaption>](http://www.w3schools.com/tags/tag_figcaption.asp) | Defines a caption for a <figure> element |
| [<figure>](http://www.w3schools.com/tags/tag_figure.asp) | Specifies self-contained content |
| [<font>](http://www.w3schools.com/tags/tag_font.asp) | Not supported in HTML5. Use CSS instead. Defines font, color, and size for text |
| [<footer>](http://www.w3schools.com/tags/tag_footer.asp) | Defines a footer for a document or section |
| [<form>](http://www.w3schools.com/tags/tag_form.asp) | Defines an HTML form for user input |
| [<frame>](http://www.w3schools.com/tags/tag_frame.asp) | Not supported in HTML5. Defines a window (a frame) in a frameset |
| [<frameset>](http://www.w3schools.com/tags/tag_frameset.asp) | Not supported in HTML5. Defines a set of frames |
| [<h1> to <h6>](http://www.w3schools.com/tags/tag_hn.asp) | Defines HTML headings |
| [<head>](http://www.w3schools.com/tags/tag_head.asp) | Defines information about the document |
| [<header>](http://www.w3schools.com/tags/tag_header.asp) | Defines a header for a document or section |
| [<hr>](http://www.w3schools.com/tags/tag_hr.asp) | Defines a thematic change in the content |
| [<html>](http://www.w3schools.com/tags/tag_html.asp) | Defines the root of an HTML document |
| [<i>](http://www.w3schools.com/tags/tag_i.asp) | Defines a part of text in an alternate voice or mood |
| [<iframe>](http://www.w3schools.com/tags/tag_iframe.asp) | Defines an inline frame |
| [<img>](http://www.w3schools.com/tags/tag_img.asp) | Defines an image |
| [<input>](http://www.w3schools.com/tags/tag_input.asp) | Defines an input control |
| [<ins>](http://www.w3schools.com/tags/tag_ins.asp) | Defines a text that has been inserted into a document |
| [<kbd>](http://www.w3schools.com/tags/tag_kbd.asp) | Defines keyboard input |
| [<keygen>](http://www.w3schools.com/tags/tag_keygen.asp) | Defines a key-pair generator field (for forms) |
| [<label>](http://www.w3schools.com/tags/tag_label.asp) | Defines a label for an <input> element |
| [<legend>](http://www.w3schools.com/tags/tag_legend.asp) | Defines a caption for a <fieldset> element |
| [<li>](http://www.w3schools.com/tags/tag_li.asp) | Defines a list item |
| [<link>](http://www.w3schools.com/tags/tag_link.asp) | Defines the relationship between a document and an external resource (most used to link to style sheets) |
| [<main>](http://www.w3schools.com/tags/tag_main.asp) | Specifies the main content of a document |
| [<map>](http://www.w3schools.com/tags/tag_map.asp) | Defines a client-side image-map |
| [<mark>](http://www.w3schools.com/tags/tag_mark.asp) | Defines marked/highlighted text |
| [<menu>](http://www.w3schools.com/tags/tag_menu.asp) | Defines a list/menu of commands |
| [<menuitem>](http://www.w3schools.com/tags/tag_menuitem.asp) | Defines a command/menu item that the user can invoke from a popup menu |
| [<meta>](http://www.w3schools.com/tags/tag_meta.asp) | Defines metadata about an HTML document |
| [<meter>](http://www.w3schools.com/tags/tag_meter.asp) | Defines a scalar measurement within a known range (a gauge) |
| [<nav>](http://www.w3schools.com/tags/tag_nav.asp) | Defines navigation links |
| [<noframes>](http://www.w3schools.com/tags/tag_noframes.asp) | Not supported in HTML5. Defines an alternate content for users that do not support frames |
| [<noscript>](http://www.w3schools.com/tags/tag_noscript.asp) | Defines an alternate content for users that do not support client-side scripts |
| [<object>](http://www.w3schools.com/tags/tag_object.asp) | Defines an embedded object |
| [<ol>](http://www.w3schools.com/tags/tag_ol.asp) | Defines an ordered list |
| [<optgroup>](http://www.w3schools.com/tags/tag_optgroup.asp) | Defines a group of related options in a drop-down list |
| [<option>](http://www.w3schools.com/tags/tag_option.asp) | Defines an option in a drop-down list |
| [<output>](http://www.w3schools.com/tags/tag_output.asp) | Defines the result of a calculation |
| [<p>](http://www.w3schools.com/tags/tag_p.asp) | Defines a paragraph |
| [<param>](http://www.w3schools.com/tags/tag_param.asp) | Defines a parameter for an object |
| [<pre>](http://www.w3schools.com/tags/tag_pre.asp) | Defines preformatted text |
| [<progress>](http://www.w3schools.com/tags/tag_progress.asp) | Represents the progress of a task |
| [<q>](http://www.w3schools.com/tags/tag_q.asp) | Defines a short quotation |
| [<rp>](http://www.w3schools.com/tags/tag_rp.asp) | Defines what to show in browsers that do not support ruby annotations |
| [<rt>](http://www.w3schools.com/tags/tag_rt.asp) | Defines an explanation/pronunciation of characters (for East Asian typography) |
| [<ruby>](http://www.w3schools.com/tags/tag_ruby.asp) | Defines a ruby annotation (for East Asian typography) |
| [<s>](http://www.w3schools.com/tags/tag_s.asp) | Defines text that is no longer correct |
| [<samp>](http://www.w3schools.com/tags/tag_samp.asp) | Defines sample output from a computer program |
| [<script>](http://www.w3schools.com/tags/tag_script.asp) | Defines a client-side script |
| [<section>](http://www.w3schools.com/tags/tag_section.asp) | Defines a section in a document |
| [<select>](http://www.w3schools.com/tags/tag_select.asp) | Defines a drop-down list |
| [<small>](http://www.w3schools.com/tags/tag_small.asp) | Defines smaller text |
| [<source>](http://www.w3schools.com/tags/tag_source.asp) | Defines multiple media resources for media elements (<video> and <audio>) |
| [<span>](http://www.w3schools.com/tags/tag_span.asp) | Defines a section in a document |
| [<strike>](http://www.w3schools.com/tags/tag_strike.asp) | Not supported in HTML5. Use <del> or <s> instead. Defines strikethrough text |
| [<strong>](http://www.w3schools.com/tags/tag_strong.asp) | Defines important text |
| [<style>](http://www.w3schools.com/tags/tag_style.asp) | Defines style information for a document |
| [<sub>](http://www.w3schools.com/tags/tag_sub.asp) | Defines subscripted text |
| [<summary>](http://www.w3schools.com/tags/tag_summary.asp) | Defines a visible heading for a <details> element |
| [<sup>](http://www.w3schools.com/tags/tag_sup.asp) | Defines superscripted text |
| [<table>](http://www.w3schools.com/tags/tag_table.asp) | Defines a table |
| [<tbody>](http://www.w3schools.com/tags/tag_tbody.asp) | Groups the body content in a table |
| [<td>](http://www.w3schools.com/tags/tag_td.asp) | Defines a cell in a table |
| [<textarea>](http://www.w3schools.com/tags/tag_textarea.asp) | Defines a multiline input control (text area) |
| [<tfoot>](http://www.w3schools.com/tags/tag_tfoot.asp) | Groups the footer content in a table |
| [<th>](http://www.w3schools.com/tags/tag_th.asp) | Defines a header cell in a table |
| [<thead>](http://www.w3schools.com/tags/tag_thead.asp) | Groups the header content in a table |
| [<time>](http://www.w3schools.com/tags/tag_time.asp) | Defines a date/time |
| [<title>](http://www.w3schools.com/tags/tag_title.asp) | Defines a title for the document |
| [<tr>](http://www.w3schools.com/tags/tag_tr.asp) | Defines a row in a table |
| [<track>](http://www.w3schools.com/tags/tag_track.asp) | Defines text tracks for media elements (<video> and <audio>) |
| [<tt>](http://www.w3schools.com/tags/tag_tt.asp) | Not supported in HTML5. Use CSS instead. Defines teletype text |
| [<u>](http://www.w3schools.com/tags/tag_u.asp) | Defines text that should be stylistically different from normal text |
| [<ul>](http://www.w3schools.com/tags/tag_ul.asp) | Defines an unordered list |
| [<var>](http://www.w3schools.com/tags/tag_var.asp) | Defines a variable |
| [<video>](http://www.w3schools.com/tags/tag_video.asp) | Defines a video or movie |
| [<wbr>](http://www.w3schools.com/tags/tag_wbr.asp) | Defines a possible line-break |

**Appendix 2: JavaScript Common Mistakes**

[« Previous](http://www.w3schools.com/js/js_best_practices.asp)

[Next Chapter »](http://www.w3schools.com/js/js_performance.asp)

This chapter points out some common JavaScript mistakes.

Accidentally Using the Assignment Operator

JavaScript programs may generate unexpected results if a programmer accidentally uses an assignment operator (=), instead of a comparison operator (==) in an if statement.

This **if** statement returns**false** (as expected) because x is not equal to 10:

var x = 0;  
if (x == 10)

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_equal_1)

This **if** statement returns**true**(maybe not as expected), because 10 is true:

var x = 0;  
if (x = 10)

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_equal_2)

This **if** statement returns**false** (maybe not as expected), because 0 is false:

var x = 0;  
if (x = 0)

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_equal_3)

|  |  |
| --- | --- |
| **Note** | An assignment always returns the value of the assignment. |

Expecting Loose Comparison

In regular comparison, data type does not matter. This if statement returns true:

var x = 10;  
var y = "10";  
if (x == y)

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_loose_1)

In strict comparison, data type does matter. This if statement returns false:

var x = 10;  
var y = "10";  
if (x === y)

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_loose_2)

It is a common mistake to forget that switch statements use strict comparison:

This case switch will display an alert:

var x = 10;  
switch(x) {  
    case 10: alert("Hello");  
}

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_loose_3)

This case switch will not display an alert:

var x = 10;  
switch(x) {  
    case "10": alert("Hello");  
}

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_loose_4)

Confusing Addition & Concatenation

**Addition** is about adding **numbers**.

**Concatenation** is about adding **strings**.

In JavaScript both operations use the same + operator.

Because of this, adding a number as a number will produce a different result from adding a number as a string:

var x = 10 + 5;          // the result in x is 15  
var x = 10 + "5";        // the result in x is "105"

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_add_1)

When adding two variables, it can be difficult to anticipate the result:

var x = 10;  
var y = 5;  
var z = x + y;           // the result in z is 15  
  
var x = 10;  
var y = "5";  
var z = x + y;           // the result in z is "105"

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_add_2)

Misunderstanding Floats

All numbers in JavaScript are stored as 64-bits **Floating point numbers** (Floats).

All programming languages, including JavaScript, have difficulties with precise floating point values:

var x = 0.1;  
var y = 0.2;  
var z = x + y            // the result in z will not be 0.3  
if (z == 0.3)            // this if test will fail

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_floats)

To solve the problem above, it helps to multiply and divide:

Example

var z = (x \* 10 + y \* 10) / 10;       // z will be 0.3

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_floats_ok)

Breaking a JavaScript String

JavaScript will allow you to break a statement into two lines:

Example 1

var x =  
"Hello World!";

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_string_1)

But, breaking a statement in the middle of a string will not work:

Example 2

var x = "Hello  
World!";

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_string_2)

You must use a "backslash" if you must break a statement in a string:

Example 3

var x = "Hello \  
World!";

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_string_3)

Misplacing Semicolon

Because of a misplaced semicolon, this code block will execute regardless of the value of x:

if (x == 19);  
{  
    // code block    
}

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_semicolon)

Breaking a Return Statement

It is a default JavaScript behavior to close a statement automatically at the end of a line.

Because of this, these two examples will return the same result:

Example 1

function myFunction(a) {  
    var power = 10    
    return a \* power  
}

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_return_1)

Example 2

function myFunction(a) {  
    var power = 10;  
    return a \* power;  
}

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_return_2)

JavaScript will also allow you to break a statement into two lines.

Because of this, example 3 will also return the same result:

Example 3

function myFunction(a) {  
    var  
    power = 10;    
    return a \* power;  
}

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_return_3)

But, what will happen if you break the return statement in two lines like this:

Example 4

function myFunction(a) {  
    var  
    power = 10;    
    return  
    a \* power;  
}

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_return_4)

The function will return undefined!

Why? Because JavaScript thinks you meant:

Example 5

function myFunction(a) {  
    var  
    power = 10;    
    return;  
    a \* power;  
}

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_mistakes_return_5)

Explanation

If a statement is incomplete like:

var

JavaScript will try to complete the statement by reading the next line:

power = 10;

But since this statement is complete:

return

JavaScript will automatically close it like this:

return;

This happens because closing (ending) statements with semicolon is optional in JavaScript.

JavaScript will close the return statement at the end of the line, because it is a complete statement.

|  |  |
| --- | --- |
| **Note** | Never break a return statement. |

Accessing Arrays with Named Indexes

Many programming languages support arrays with named indexes.

Arrays with named indexes are called associative arrays (or hashes).

JavaScript does **not** support arrays with named indexes.

In JavaScript, **arrays** use **numbered indexes**:

Example:

var person = [];  
person[0] = "John";  
person[1] = "Doe";  
person[2] = 46;  
var x = person.length;         // person.length will return 3  
var y = person[0];             // person[0] will return "John"

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_array_associative_1)

In JavaScript, **objects** use **named indexes**.

If you use a named index, when accessing an array, JavaScript will redefine the array to a standard object.

After the automatic redefinition, array methods and properties will produce undefined or incorrect results:

 Example:

var person = [];  
person["firstName"] = "John";  
person["lastName"] = "Doe";  
person["age"] = 46;  
var x = person.length;         // person.length will return 0  
var y = person[0];             // person[0] will return undefined

[Try it Yourself »](http://www.w3schools.com/js/tryit.asp?filename=tryjs_array_associative_2)

Ending an Array Definition with a Comma

Incorrect:

points = [40, 100, 1, 5, 25, 10,];

Some JSON and JavaScript engines will fail, or behave unexpectedly.

Correct:

points = [40, 100, 1, 5, 25, 10];

Ending an Object Definition with a Comma

Incorrect:

person = {firstName:"John", lastName:"Doe", age:46,}

Some JSON and JavaScript engines will fail, or behave unexpectedly.

Correct:

person = {firstName:"John", lastName:"Doe", age:46}

Undefined is Not Null

With JavaScript, **null** is for objects, **undefined** is for variables, properties, and methods.

To be null, an object has to be defined, otherwise it will be undefined.

If you want to test if an object exists, this will throw an error if the object is undefined:

Incorrect:

if (myObj !== null && typeof myObj !== "undefined")

Because of this, you must test typeof() first:

Correct:

if (typeof myObj !== "undefined" && myObj !== null)

Expecting Block Level Scope

JavaScript **does not** create a new scope for each code block.

It is true in many programming languages, but **not true** in JavaScript.

It is a common mistake, among new JavaScript developers, to believe that this code returns undefined:

Example:

for (var i = 0; i < 10; i++) {  
    // some code  
}  
return i;

**Appendix 3: JavaScript Reserved Words**

In JavaScript you cannot use these reserved words as variables, labels, or function names:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| abstract | arguments | boolean | break | Byte |
| case | catch | char | class\* | Const |
| continue | debugger | default | delete | Do |
| double | else | enum\* | eval | export\* |
| extends\* | false | final | finally | Float |
| for | function | goto | if | implements |
| import\* | In | instanceof | int | interface |
| let | long | native | new | Null |
| package | private | protected | public | Return |
| short | static | super\* | switch | synchronized |
| this | throw | throws | transient | True |
| try | typeof | var | void | volatile |
| while | with | yield |  |  |

Words marked with\* are new in ECMAScript5

JavaScript Objects, Properties, and Methods

You should also avoid using the name of JavaScript built-in objects, properties, and methods:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Array | Date | Eval | function | hasOwnProperty |
| Infinity | isFinite | isNaN | isPrototypeOf | length |
| Math | NaN | Name | Number | Object |
| prototype | String | toString | undefined | valueOf |

Java Reserved Words

JavaScript is often used together with Java. You should avoid using some Java objects and properties as JavaScript identifiers:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| getClass | Java | JavaArray | javaClass | JavaObject | JavaPackage |

Windows Reserved Words

JavaScript can be used outside HTML. It can be used as the programming language in many other applications.

In HTML you must (for portability you should) avoid using the name of HTML and Windows objects and properties:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| alert | All | anchor | Anchors | Area |
| assign | blur | button | Checkbox | clearInterval |
| clearTimeout | clientInformation | close | closed | confirm |
| constructor | crypto | decodeURI | decodeURIComponent | defaultStatus |
| document | element | elements | embed | embeds |
| encodeURI | encodeURIComponent | escape | event | fileUpload |
| focus | form | forms | frame | innerHeight |
| innerWidth | layer | layers | link | location |
| mimeTypes | navigate | navigator | frames | frameRate |
| hidden | history | image | images | offscreenBuffering |
| open | opener | option | outerHeight | outerWidth |
| packages | pageXOffset | pageYOffset | parent | parseFloat |
| parseInt | password | pkcs11 | plugin | prompt |
| propertyIsEnum | radio | reset | screenX | screenY |
| scroll | secure | select | self | setInterval |
| setTimeout | status | submit | taint | Text |
| textarea | top | unescape | untaint | window |

HTML Event Handlers

In addition you should avoid using the name of all HTML event handlers.

Examples:

|  |  |  |  |
| --- | --- | --- | --- |
| onblur | onclick | onerror | Onfocus |
| onkeydown | onkeypress | onkeyup | onmouseover |
| onload | onmouseup | onmousedown | Onsubmit |

Nonstandard JavaScript

In addition to reserved words, there are also some nonstandard keywords used in some JavaScript implementations.

One example is the **const** keyword used to define variables. Some JavaScript engines will treat const as a synonym to var. Other engines will treat const as a definition for read-only variables.

Const is an extension to JavaScript. It is supported by the JavaScript engine used in Firefox and Chrome. But it is not a part of the JavaScript standards ES3 or ES5. **Do not use it**.