Javascript Quick Reference

For this quick reference, class notes, and slides, see: ***http://shallowsky.com/javascript***

Comments

Any line starting with **//** is called a comment.
It isn't part of the real program: it's just a place to make notes to yourself.

Testing an HTML page or a Javascript program

To test your HTML or Javascript program, you need to open it in Firefox.
Find the file on disk again, then drag that file into the Firefox window.

Once you've loaded it in Firefox the first time, you can test it after you've made changes by clicking Firefox's **Reload** button.

Alerts

`alert("anything you want");` -- pops up a dialog.

Error Console

Tools -> Error Console gives you a window where you can see any errors you make in your Javascript.
It also gives you a text box where you can try out simple javascript commands.

Strings

Strings are words or sentences in quotes:

"double quotes (2 tics)"
'single quotes (1 tic)'

You can put several strings together with a + sign:

```
alert("Good morning, " + yourname + " Welcome to the class!");
```

Strings are different from numbers:

```
2 + 3       is   5
"2" + "3"   is  "23"
```

Variables

Variables are a place to store things, like strings or numbers.
Use **var** the first time you declare a variable, but not after that.

```
var name = "Hermione Granger";
var age;
name = "Ron Weasley";
age = 17;
```

Writing to the Page

document.write is one way to tell Javascript to write to the web page.

```
document.write("Hello, " + name);
```

Note: You might sometimes see the string "**
**" inside a *document.write()*. That inserts an HTML line break in the output ("br" is short for "break". In Javascript you can write any HTML tags you want, but the line break is one of the most useful.

You might also see "**\n**" (backslash followed by n). That's a way of putting a line break inside the HTMl source itself: it doesn't affect how the page looks, but it makes the page easier to read if anyone looks at it with *View Page Source*. "n" stands for "newline".

Prompt

Use prompt to ask the user something.

```
var name = prompt("What is your name?");
```

Functions

Functions are a way of grouping several operations together:

```
function changePageColor(newbackground) {
    if (! document.getElementsByTagName)
        return false;
    var body = document.getElementsByTagName("body")[0];
    body.style.background = newbackground;
    return false; // don't let the default link action be triggered
}
```

Arrays

An array is a collection of a bunch of similar things.
Access array elements with square brackets, [ ]

```
var colors = [ "blue", "green", "yellow", "red", "black", "white" ];
```

Arrays always start from 0, not 1. So colors[0] is "blue", colors[2] is "yellow", etc.

You can find out how many objects there are in an array with length:

```
var numColors = colors.length;
```

For Loops

A loop is something that repeats over and over.

One type of loop is for. You can use for loops with numbers:

```
for (var i=1; i <= 10; ++i) {
    document.write(i + " ");
}
```

prints: 1 2 3 4 5 6 7 8 9 10

For Loops with Arrays

```
var colors = [ "blue", "green", "yellow", "red", "black", "white" ];

for (var c in colors) {
    document.write(colors[c]);
}
```

"While" Loops

While is another type of loop:

```
var name = "";
while (name == "") {
    prompt("What is your name?");
}
```

Syntax

● Put semicolons at the ends of lines.

● Capitalization is important! `var name;` is different from `var Name;` or `var NAME;`

● Put curly braces **{ }** around the insides of loops, if-else, etc. (It's not required every time, but it helps guard against making errors.)

if, else if, else

Use if and else to test things.

```
if ( 4 <= hour && hour <= 11) {
    document.write("Good morning!\n");
}
else if ( 12 <= hour && hour <= 17) {
    document.write("Nice afternoon, isn't it?\n");
}
else {
    document.write("It must be evening!");
}
```

Equals

Use **=** (one equals sign) to set a variable to something.
Use **==** (two equals signs) to compare a variable against something else.

Comparing numbers

There are lots of ways to compare two numbers:

```
  a == b    a is equal to b (two equals signs)
  a < b     a is less than b
  a > b     a is greater than b
  a <= b    a is less than or equal to b
  a >= b    a is greater than or equal to b
  a != b    a is not equal to b
! means not:       !(a == b) means the same as a != b
&& means AND       || means OR
```

Military Time

Computers use "military time". That means that after noon, hours have 12 added to them.
11 am is 11, noon is 12, 1 pm is 13, 2 pm is 14, ... etc

Confirm

Use **confirm** to ask the user a yes/no question: it will pop up a dialog with OK and Cancel.

```
    var name = prompt("What is your name?");
```

Images: Fixed Position on the Page

To put an image at a specific place on a web page, in your HTML use **position:absolute**:

```
<img id="flower" src="images/flower1.gif"
 style="position:absolute; bottom:20px; right:20px;">
```

The **id=** gives it a name ("flower") that you can use to move it later with Javascript.

Handling Mouse Clicks

```
function handleClick(event) {
  // IE doesn't see the event argument passed in, so get it this way:
  if (window.event) event = window.event;

  var img=document.getElementById("flower");
  img.style.left = event.clientX - 50;
  img.style.top = event.clientY - 50;
}
```

getElementById("flower") refers to the *id="flower"* you used earlier in <*img*>.

Initializing Events

To handle mouse clicks or key events, you have to have an initialization routine that you run when the page is loaded. To do that:

In the HTML:

```
<body onload="init();">
```

In the Javascript:

```
function init() {
  document.onmousedown = handleClick;
}
```

Good Books:

DOM Scripting, Jeremy Keith.
A great introduction to using Javascript in web pages.
The Javascript Anthology, James Edwards and Cameron Adams.
A little more advanced, ideal for programmers and a good collection of tips and techniques.