Ken's JavaScript, jQuery Notes

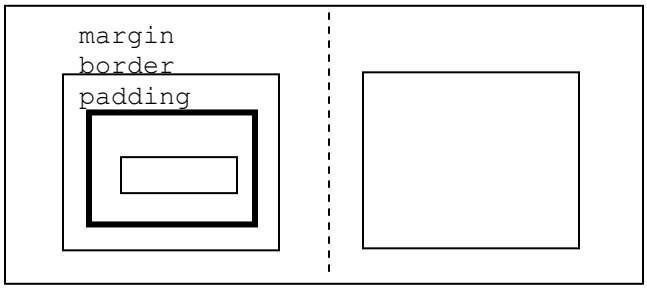
Quick refresher thru code snippets

Contents

Html

```
<!DOCTYPE…  on IE if missing browser uses "quirks" mode
<html>
    <head>
    </head>
    <body>
    </body>
</html>
```

"#" in a form means submit to itself



Setting HTML Form Fields via Javascript

```
<script type='text/javascript'>
  window.onload = initAll;
  function initAll () {
    // set an input field or a list box
    document.getElementById('lsr001').value = 'ken freed';
    // set a yes,no radio button
    document.getElementById('lsr007a').checked = true;
  }
</script>

Name of Requester:
<input name="lsr001" type="text" class="reqd" id="lsr001" size="30"
maxlength="60" />

New lot start?
<input type="radio" name="lsr007" id="lsr007a" value="yes">Yes
<input type="radio" name="lsr007" id="lsr007b" value="no">No
```

Updating the Forms Webpage when Returning from Server

Getting the webpage to update after a form posts to the .php script on the server is tricky

If a webpage has something like:

```
window.onload = initAll;
function initAll () { … }

<?php  include $aFile; ?>
```

A server side return wont trigger client side includes in IE8:

```
<h1 align=center>WSV Form Data Submitted</h1>
<script language="JavaScript" type="text/javascript">
  <!--
  document.write("<p align=center><input type=\"button\"
onClick=\"history.back();\" value=\"RETURN\"></script></script></p>");
  //-->
</script>
<p align=center>Upon return, please refresh the webpage  to display the latest
data</p>
```

This client side scripting won't cause the page to refetch its includes (in IE8) either

The inner html below never shows up unless browser is refreshed

```
function submitshipform() {
  document.getElementById('ship000').innerHTML = 'Form submitted';
  document.shipform.submit();
}
```

Which is called by either a submit or button input:

```
 <input type="button" name="<?php echo $WEBDIR . $_SERVER["PHP_SELF"] ?>"
onClick="submitshipform();" value= "Submit" />
```

What will work is returning to a specific URL from the server

```
  foreach ( $_POST as $key => $value ) {
    if ($value == "Submit") {
      $pieces = explode ("/", $key);
      $returnlink = $pieces[sizeof($pieces)-1];      //last piece
         $returnlink = substr($returnlink,0,strlen($returnlink)-4) . ".php";
  …}

<input type="button" onClick="javascript:location.href ='<?php echo $returnlink
?>';" value="RETURN">
```

CSS Selectors

Defining

File

```
<link type="text/css" rel="stylesheet" href="../master.css">
```

Inline

```
<style type="text/css">
    body {color:red; background-color:black;}
    fieldset {border-style:none; width:30%; float:left;}
    .button {margin-left:30%;}
    p {text-align:center}
</style>
```

Selectors

tag {
#id { *the id is only used once in a page (e.g. the title, or something unique)*
.class {
* { *selects everything*
html { *html is a tag, you can select it (but html is not a decendent, except in IE)*

Combining Selectors

```
TAG ID CLASS (can leave some out, but must maintain order).  Don't use spaces
between them
      p#bio.quote {rule1; rule2; … ruleN; }
Spaces mean decendency (children are one level down, decendents go beyond)
    div a {
        p.quote a {         //<p class="quote">
        div#bio p.quote {    // two levels of decendency,  <div id="bio">
        div#foo p .special:link {   //three levels of decendency
```

Comma mean OR

```
      div#bio, p.quote, a {
```

Using

```
<p class="oneclass twoclass">   theese are two (space delimited) classes
```

Pseudo Class

You don't define these, they're just there

a:link { the colon prefix denotes a pseudo class
a:visited {
a:hover {
a:active {

Selector Precidence

The more specific the selector, the more precidence

```
1)  p {color:#000}
```

2) p.quote {color:red} *this one will win, because it's more specific*

11) #content p { *an ID is worth 10 points in the rating, since IDs are very specific*

```
If two tags have the same precidence, then the first one defined wins (precidence
goes top to bottom).
```

Color

Color is Text color (background-color is background).
Color is Light color

- Paint color (everything makes black) is the opposite of
- Light color (eveything makes white)

rgb(0,0,0) *black*
rgb(255,255,255) *white*
rgb(#FF,#FF,#FF) *white*
rgb(994422) = rgb(942) *trick!*

JavaScript / jQuery

Script Inclusion

```
Calling a separate Javascript script:
In your html fille put:
      <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
      <html xmlns="http://www.w3.org/1999/xhtml">
      <head>
      <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
      <title>RIAs with JavaScript</title>
      <script type='text/javascript' src='jquery-1.3.2.min.js'></script>
      <script type='text/javascript' src='exercise.js'></script>
      <script type='text/javascript'>
      <!—  inline code goes here -->
      jQuery(document).ready(function() { console.debug(jQuery(".photo a"));
                                                  });
      </script>
                                          <style type="text/css">
body {background-color:silver;}
<!— more css ges here →
                                          </style>
   </head>
   <body>
      <h1>RIAs with JavaScript</h1>
      </body>
      </html>
```

Data Types

Javascript has "dynamic typing"
Javascript has 4 Primitive Data Types

Boolean	var bX = true; // false is the other possibility
Number	var x = 255; var y = 3.1415;
String	var aString = "Hello World; var anotherString = 'foo';

```
aString.length;  // gives the length of thee string
```

Object	var anObject = new Object();
	var anObject = {};

Arrays

```
   var anArray = new Array();
   var anArray = [];  // shorthand for new Array()

      var a = [1, 2, 3];
      var a = ["Ken", 2, "joe", age:35, {} ];  // {} is another object
a.push("another_thing"); var anElement = a.pop();
   a.unshift("in_front");
```

```
var frontElement = array.shift();

   var size = a.length;
a.sort();
a.splice(0,2);

for (var i = 0; i < a.length; i++) {
// Do something with a[i]
    }

var a = [1, 2, 3];
jQuery.each(a, function(index, value) {
// do something with index or value
    });
```

Structures (or sort of structures)

```
var employee = new Object();
employee.name = "Ken Freed";  // NOTE – no new for object fields!
employee.class = new Object();  // employee.class.<new object>
employee.class.name = "Math 101";  // employee.class.name
```

Addressing

```
   var aName = "ken_freed";
var anObject = new Object();
anObject.aName = "ken_freed";
anObject["aName"] = "ken_freed";  // same as previous
```

Variable Substitution (for the Key definition)

```
// w.o. quotes we get the variable's contents
var aName = "ken_freed";
anObject[aName]  is the same as    anObject.ken_freed

   //anObject[aName]  is  "ken_freed";  // NOTE: anObject.ken_freed
   = …
   anObject["instructor"] = new Object();
   anObject["instructor"]["name"] = "Ken Freed";  // same as
   .instructor.name
```

Object Intialization

Note: no quotes for the Key, therefore no variable substitution

```
var person = {
        name: "Ken",                    // NOTE no variable substitution in this syntax
        // "name":"Ken",   // this is the same thing as the previous line
        pets: ["dog","cat","bird"],
        spouse: {
                name:"Vanessa",
```

```
                    age:48
                    }
            };

4 ways to reference the person object's fields:
    person.name
    person["name"] // parenthesis=literal .name
    var aKey = "name"
    person[aKey]  // no parenthesis=deref the variable
```

String

String to Number

```
    var aString = "5";
    var aNumber = Number(aString);
```

Number to String // check on!

```
    var aNumber = 5;
        var aString = parseInt(aNumber);   // check on!

        parseInt ("A", 16);   // returns 10
```

Strict Equality, Inequality

```
        5 = = "5"    // true
        5 = = = "5"  // false, must be same value and TYPE

        5 ! = = "5"  // true, must be different value AND type
```

Collections

```
        var anObj = {};
        anObj.name = "Ken";
        anObj.age = 54;
        anObj.parm3 = "foo";

        for (var aKey on anObj) {
        // aKey = name,age,parm3 at each loop iteration
        }
```

Functions

Anonymous Function

```
aPerson.sort = function (a, b) { // some code // "this" is the
aPerson instance (actually the "window.aPerson" instance)
} );
aPerson.sort(1,2);  // calls the function
```

Functions

```
function helloWord ( ) {
// arguments[]  or parms passed in is always defined
// this            is always defined
}

// another way to define a function
var helloWorld = function ( ) {
```

// some code

```
}

function() {
return aNumber;                                  // to return one thing
return {aKey1:aValue1, aKey2:aValue2};  // to return multiple things
}
```

Calling a JavaScript function

```
<input type="button" name="<?php echo $WEBDIR . $_SERVER["PHP_SELF"] ?>"
onClick="submitshipform();" value= "Submit" />

Note:   document.theFormName.submit();
```

Returning

```
<input type="button" onClick="history.back();" value="Back">

<input type="button" onClick="location.href='http://199.19.121.136/somefab.htm'"
value="Return">

<form action="http://www.xyz.com/gohere.htm">
<input type="button" value="Return">
</form>
```

Handling Errors

```
window.onload = initAll;
function initAll () {
      var ans = prompt("Enter a positive number:", "");
      try {
            if (! ans || isNaN(ans) || ans < 0) {
                  throw new Error("invalid entry");
            }
            alert ("Sqrt=" + Math.sqrt(ans));
      } catch (errMsg) {
            alert (errMsg.message);
      }
}
```

ready function

```
(= windows.onload in javascript)
console.debug(jQuery("#news h4"));  // returns [] since it did not find anything to
select
/* this did not return anything because the web page was not yet loaded */
/* the selector is correct
    reads html top to bottom
    executes the exercise.js before it has hit the body tag of the html
    need to wait until the document is finished loading
    need that body to be ready
*/

/*-------------------------------------------------------
      define anonymous functions within these calls,
      since we are not going to reference them anywhere else
-------------------------------------------------------*/
jQuery(document).ready(function() {
```

/* define another callback function here */

```
      /* show all h4 headers, under the html tag <div id="news">...</div> */
      console.debug(jQuery("#news h4"));  // this is called a CALLBACK function,
// since it gets called back after something else occurs
/* this call will now return the text associated with the h4 headers
      i.e.:
```

*** ***

```
    ...
  <h4>New site!</h4>
  <h4>Happy Holidays</h4>
  <h4>New Director</h4>
  </div>

      But note that the h4 verbiage all runs together.
      This will do them separately:
      jQuery("#news h4").each( function() {
      console.debug( jQuery(this).text() ); // we pass the THIS to
      jQuery, not the call as well

});
```

jQuery Object

```
      jQuery("h1")[0]    // 1st object
      jQuery("h1")[1]    // 2nd object
```

Noscript

```
      <noscript>
      <h2>Sorry, this page requires JavaScript</h2>
      </noscript>
```

Will only display back button if javascript is enabled:

```
      <script language="JavaScript" type="text/javascript">
<!--
      document.write("<input type=\"button\"
                  onClick=\"history.back();\" value=\"Back\">");
//-->
    </script>

Html Back Button:
  <a href=\"#\" onclick=\"history.go(-1)\">Go Back</a>
```

User Interaction

```
alert("Welcome to my page");  // alert popup

if ( confirm("Are you sure?")) {  // returns true or false

var ans = prompt("Are you sure?", "");  // 2nd parm is default answer
```

Redirect

```
window.location = "jswelcome.html";
```

Php Html version of redirect

- No relative paths. No further output to the browser

```
header("Location:http://www.kenfreedsoftware.com?msg=access denied");
exit();  // Php - no further brower output
```

Getting the Link Clicked on

```
// this = a link they just clicked on.
// eg: Does it contain kenfreed1 as part of the linkname?

if ( this.toString().indexOf("kenfreed1") < 0 )  {  // if not a local domain
      alert ("WARNING-redriecting to another site");
```

OnClick

To a url

```
<input type="button" value="History" onclick="javascript:location.href =
'http://172.19.121.136/svtc/shipwip/shipwip3_fab2.htm';" />

<input type="button" value="Create forms" onclick="javascript:location.href =
'adv_fab2_oper.php';" />

<input type='button' value='Printable Pdf'
onclick="window.open('http://www.kenfreedsoftware.com/aDoc.pdf',
'mywindow','width=800,height=400')">

<input type='button' value='Printable Pdf'
onclick="window.location.href='http://www.kenfreed1.com/aDoc.pdf">
```

Non Javascript

```
<input type="button" value="Create forms" onclick=window.location.href=
'adv_fab2_oper.php';" />
```

Math

```
 var newVar = Math.floor(1 + Math.random() * 75);
```

Older Browser Trick

```
if ( document.getElementById("anId") ) {  // returns false if not handled
```

Strings

```
var aChar = aString.charAt(0);
var aPos = aString.indexOf("@", 1);   // start a 1, 0 is 1st char
var aNumber = parseInt(aString);
```

Trim, Length

```
  var anItl = document.getElementById('wsv001').value;
  var anEmail = document.getElementById('wsv003').value;
  var aLot = document.getElementById('wsv005').value;

  //trim
  anItl = anItl.replace(/^\s+|\s+$/g, '') ;
  anEmail = anEmail.replace(/^\s+|\s+$/g, '') ;
  aLot = aLot.replace(/^\s+|\s+$/g, '') ;

  if ((anItl.length == 0) || (anEmail.length == 0) || (aLot.length == 0)) {
…
```

Popups

var imgWindow = window.open(imgName, "imgWin",
"width=320, height=240, scrollbars=no");

Confirmation

<input type=submit name='clear order'
onclick="return confirm('Are you sure?');" value='clear order'>

Event Handlers

Returning false from a javascript event handler keeps the link from executing

```
<input type="submit" name="aSubmittal" id="wsv_submit"
  onClick="return validate_wsv();" value="Store for Later" />
```

Common Example:

```
window.onload = initFunction;
function initFunction () {
      // setup event handlers, e.g.:
for (var i = 0; i < document.forms.length; i++) {
      document.forms[i].onsubmit = function () {
            is_form_valid();
      }

      document.getElementById("sunroof").onclick = doorSet;
}
```

Mouse events	onclick, oncontextmenu, ondblclick, onmousedown, onmouseup
Form field events	onblur, onchange, onclick, onfocus, onreset, onselect, onsubmit
Window events	onabort, onblur, onerror, onfocus, onload, onmove, onresize, onscroll, onunload
Keyboard events	onkeydown, onkeypress, onkeyup

Keyboard events

```
document.onkeydown = keyHit;
function keyHit(evt) {

      var ltArrow = 37;
      var rtArrow = 39;
      var thisKey = 0;

      if (evt) {    // trick! - browser might not pass event
        thisKey = evt.which;
      } else {
        thisKey = window.event.keycode;
      }

      if (thisKey = = ltArrow) {
```

Checking Form Info with Javascript

```
<script type="text/javascript">
  <!-- hide from older browsers
  window.onload = initForms;
  function initForms() {

    document.contactForm.onsubmit = function () { return validateForm(); }
  }

  function validateForm() {
    if (document.getElementById("nameFld").value.length <= 0) {
      alert('Please enter a Name');
      return false;
    } else {
      return true;
    }
  }
</script>


…
  <input type="submit" name="submit" id="submit" value="Send it in!" />
</form>
```

Alternative (without global onsubmit handler)

```
  <form id="emailform" name="emailform" method="post"
        action="email_form.php"
        onsubmit="return validateForm()">
…
    <input type="submit" name="submitname" id="email_submit"
       value="Email Dock to Ship this Lot" />
  </form>
```

DOM

Common DOM calls for form validation

document.

getElementsByTagName("p").	innerHTML = 'No way!'
getElementById("anId").	innerHTML = 'No way!' .value

```
function alertSummary()
{
alert(document.getElementById("table1").summary);
}
</script>
</head>
<body>

<table id="table1" border="1" summary="Example table of employees">
```

Table object properties

Methods	Description
align	Gets or sets the alignment of the table.
bgColor	Gets or sets the background color of the table. Color name or hexadecimal value.
border	Gets or sets the thickness of the inherent, 3D border of the table (different from the generic "border" property of CSS). Numeric string value. **Example:** document.getElementById("atable").border="3"
borderColor	**IE only** property that gets or sets the border color of the "border" property above.
borderColorDark	**IE only** property that gets or sets the "dark" aspect of the "border" property above.
borderColorLight	**IE only** property that gets or sets the "light" aspect of the "border" property above.
caption	References the caption element of a table.
cellPadding	Gets or sets the cellPadding attribute of the table. Value should be numeric string (ie: "5") or percentage (ie: "10%").
cells[]	Returns an array containing all the td elements inside the table, in source order. **Example(s):**

<table>
<tr><td></td><td>var mytable=document.getElementById("atable")
var tdref=new Array() //array to hold references to table cells
for (var i=0; i<mytable.cells.length; i++)
tdref[tdref.length]=mytable.cells[i] //store references to table cells

alert(tdref.length) //alerts number of cells in the table</td></tr>
<tr><td>cellSpacing</td><td>Gets or sets the cellSpacing attribute of the table. Value should be numeric string (ie: "5") or percentage (ie: "10%").</td></tr>
<tr><td>frame</td><td>Specifies which sides of a table outer border to show. The possible string values are:
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>above</td><td>Show top border only.</td></tr>
<tr><td>below</td><td>Show bottom border only.</td></tr>
<tr><td>border</td><td>Show all sides of border.</td></tr>
<tr><td>box</td><td>Show all sides of border.</td></tr>
<tr><td>hsides</td><td>Show top and bottom borders only.</td></tr>
<tr><td>lhs</td><td>Show left border only.</td></tr>
<tr><td>rhs</td><td>Show right border only.</td></tr>
<tr><td>void</td><td>Hide all borders (default).</td></tr>
<tr><td>vsides</td><td>Show left and right borders only.</td></tr>
</table>
The "border" property must also be set in order for the effect to be visible (ie: border="3").</td></tr>
<tr><td>height</td><td>IE only property that specifies the height of the table.</td></tr>
<tr><td>rows[]</td><td>Returns an array containing all the rows (tr) elements inside the table, in source order. This includes all rows found in THEAD, TFOOT and TBODY elements.
The rows[] array of a table section contains only the rows for that section. The table sections are: THEAD, TFOOT and TBODY.</td></tr>
<tr><td>rules</td><td>Specifies which sides of cells' interior border to show.
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>all</td><td>Show border around each cell.</td></tr>
<tr><td>cols</td><td>Show border around columns only.</td></tr>
<tr><td>groups</td><td>Show border between cell groups only (thead, tfoot, tbody, colgroup, or col elements).</td></tr>
<tr><td>none</td><td>Don't show any border between cells.</td></tr>
<tr><td>rows</td><td>Show border between rows only.</td></tr>
</table></td></tr>
</table>

	The "border" property must also be set in order for the effect to be visible (ie: border="3").	
summary	Gets or sets the "summary" attribute of the table, a description of the table that can be accessed via scripting.	
tBodies[]	Returns an array containing all the tbody elements inside the table, in source order. Note that this includes implicit tbody elements- every table has at least one tbody element.	
tFoot	References the tFoot elment of the table.	
tHead	References the tHead elment of the table.	
width	Gets or sets the width of the table.	

Table tr object properties

The tr element of a table supports several properties of its own that are accessible via scripting. The most important one is the below:

Methods	Description
cells[]	Returns an array containing all the td or th elements within a table row.

(todo - need to cleanup and organize this, these are just reminders for now).

```
var allTags = document.getElementsByTagName("*");
var pTags = document.getElementsByTagName("p");
var allForms = document.getElementsByTagName("form");

anElement = document.getElementById("anId");

document.getElementById("anId").innerHTML = "something new";

if (document.images[i].parentNode.tagName = = "A") {

var theSrc = document.image[i].src;  //  eg: images/myPic.jpg

.parentNode
.childImage
.childImg.src
.overImage
.overImage.src
.outImage
.outImage.src
.clickImage
.clickImage.src

.item
.removeChild(lastP);
```

```
.insertBefore(newP, oldP);
.replaceChild(newP, oldP);


document.links.length
document.links[i]

.style.backgroundColor = "#00F";

var allTags = document.getElementsByTagName("*");
for (var i = 0; i < allTags.length; i++ ) {
        if (allTags[i].className.indexOf("req_class") > -1) {

var userName = document.cookie.split("=")[1];  // 0 is 1st, 1 is 2nd

var comments = document.getElementsByTagName("TextArea")[0].value;
var newTxt = document.createTextNode(inText);
var newP = document.createElement("p");
newP.appendChild(newP);
var docBody = document.getElementsByTagName("body")[0];
docBody.appendChild(newP);

could also:
document.getElementById("someId").innerHTML = "<p>" + inText + "</p>";
```

Date/Time

```
var dayName = new Array("Sunday", …
var monthName = new Array("dummy", "Jan", …

var now = new Date();
var thisMonth = now.getMonth();
var dtString = "Today is " + dayname[now.getDay()]

now.getFullYear()
now.getDate()
now.getHours();
now.getMinutes();
now.getSeconds();

setTimeout("aFunction()", 1000);  // calls aFunction after 1000 mS
// use Tcl-like recursion to call repeatedly, eg:
function showTime() {
…
  setTimeout("showTime()", 1000);
}
```