# Guideline

case sensitive

define your function before you call them

# Get start

<script type=”text/javascript” src=”js/script.js”></script>

# creating variables

* 1. var year; (variable names could be letters, numbers, \_, $….)

                  var 99problems; not allowed (could not start with numbers)

                   var problems99; allowed

* 1. There is no string, float, int….

# Conditional Code

1. If (d!=100){

//code goes there

}else{}

# For loop

## Access array

For(i=0; i<=cars.length; i++){

Text += cars[i] + “</br>”;

}

# Function in Javascript

## without parameter

//define your function before your call it

function createMessage {

//name, cannot start with numbers

console.log(“We’re in the function”);

//anythings

}

//call function

createMessage();

## with parameters

function myFunction(x,y,z)

//parameters: expected to pass information to myFunction

{ var myVar = x\*y;

console.log(myVar);

//we can return values

return myVar;

}

//call functions

myFunction(754, 346);

myFunction(123, -732);

alert(“Hello world”); //built-in javascript function

//use function to create a result, must have return value

var myresult = myFunction(6, 9);

## Parameter Mismatch

Function calculateLoan(amount, months, interest, name){

//…

}

calculateLoan(10000, 60, 7, “Sam Jones”);//correct

calculateLoan(10000, 60, 7, “Sam Jones”, “Something extra”); //extras are ignored

## Variable Scope

### Situation 1

Function simpleFunction(){

var foo=500;

console.log(foo);

}

simpleFunction(); //output 500

console.log(foo);//output undefined, foo declared inside function, outside the function, foo doesn’t exist

### Situation 2

var foo;

function simpleFunction(){

foo=500;

console.log(foo);

}

simpleFunction();

console.log(foo);//output 500

# Creating Arrays

var multipleValues=[];

multipleValues[0]=50;

multipleValues[1]=60;

multipleValues[2]=”Mouse”;

var multipleValues2=[50,60,”Mouse”];

## initiate array

1. var multipleValues = []
2. var multipleValues = new Array();
3. var txt = new Array(“tim”,”jim”,”kim”);
4. var multipleValues = Array();
5. var multipleValues = Array(5);//create a array with a size of 5

## Array Methods

var multipleValues=[10,20,30,40,50];

var reversedValues=multipleValues.reverse();

.join();

.sort();

console.log(reversedvalue.join());

var myArrayOfLinks=document.getElementsByTagName(“a”);

# Working with strings

## Quotes inside quotes

var phrase = “He said \”that’s fine, \” and left.”;

## String methods

### phrase

var phrase = “This is a simple phrase.”;

var word = phrase.split(“ ”);//split phrase by space

|  |  |
| --- | --- |
| 0 | This |
| 1 | is |
| 2 | a |
| 3 | simple |
| 4 | phrase. |

### IndexOf

Find out if a particular term of word appears anywhere in the string

var phrase = “We want a groovy keyword.”;

var position = phrase.indexOf(“groovy”); //10

//it returns -1 if the term is not found

//there is also .lastIndexOf()

## Slice

var phrase = “Yet another phrase.”;

var segment = phrase.slice(6,11);//6: starting position, 11: ending position

//other

# Working with Dates

var today = new Date();//current date and time

//year, month, day

var y2k = new Date(2000,0,1);

//month starts from 0

var y2k = new Date(2000,0,1,0,0,0)

//hours, minutes, seconds

# Working with Object

## Object Creation

var player = new Object();

player.name = “Fred”;

player.score = 10000;

player.rank = 1;

## Object Creation Shorthand

var player1 = {name: “Fred”, score: 10000, rank: 1};

console.log(player1.name)//Fred

## Situation 1

//create two objects

var player1 = {name: “Fred”, score: 10000, rank: 1};

var player2 = {name: ”Sam”, score: 10000000, rank: 5};

function playerDetails(){

//display information about each player

console.log(this.name + “ has a rank of: “ + this.rank + “ and a score of ” + this.score”);//this: current object, can be player1, can be player2, see which one will be called

}

player1.logDetails = playerDetails;

player2.logDetails = playerDetails;

player1.logDetails(); //output: Fred has a rank of: 1 and a score of 10000

player2.logDetails();

# Document Object Model

## Model

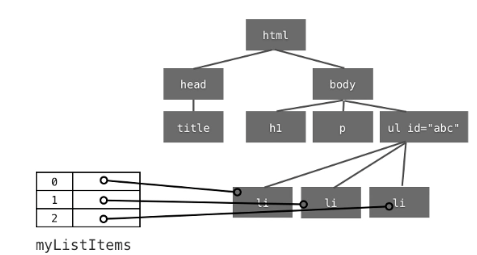
var myLinks=document.getElementsByTagName(“a”);

//go through our document and give us back any of the element nodes that are anchor elements, a tag in your HTML

//array will be empty for below case

var my listItems=documents.getElementsByTagName(“li”);

//find me all the nodes that are list items elements and return them, make array



## Working with Attributes

myElement.getAttribute(“align”);

//or “title”, “class”, “source”…

myElement.setAttribute(“align”, “left”);

//name, value

## Creating Elements

//Creating Elements

//1.

var myNewElement = document.createElement("li");

//2.

myElement.appendChild(myNewElement);// add to DOM

//3. set the text to the list item

myNewElement.innerHTML=”New Item Text”;

//Or (Preferred)

var myText = document.createTextNode(“New list item text”);

myNewElement.appendChild(myText);

## Alternatives to Appendchild

var my NewElement = document.createElement(“li”);

var secondItem = myElement.getElementsByTagName(“li”)[1];

myElement.insertBefore(myNewElement, secondItem);

# JQuery

$(document).ready(function(){});

$(“selector”)

$(variable)

## Retrieve and set value

### .text()

Alert(“Text: ” + $(“div#Test”).text(“A dynamic set text”)); //only the text

### .html()

Alert(“HTML: ” + $(“div#Test”).html”<b><i>A dynamic set html string</i></b>”));

### .val()

Get the values of the form elements such as input, select, textarea

Set value

1. set value from textbox

$(“input:text”).val(“Glenn Quagmire”);

1. from a dropdown select

$(“select.foo option:selected”).val();

1. from a checked checkbox

$(“input:checkbox:checked”).val();

1. from a set of radio buttons

$(“input:radio[name=bar]:checked”).val();

## Css() method

Change the attribute

1. single:

$("p").css("background-color", "yellow");

1. multiple css properties

css({"*propertyname*":"*value*","*propertyname*":"*value*",...});

## Traveral method

1. parent(): direct parent

$(document).ready(function(){  
    $("span").parent().css({"color": "red", "border": "2px solid red"});  
});

1. parents(): all ancestor elements
2. parentsUntil(): all ancestor elements between two given argument

$(“span”).parentUntil(“div”)

//returns all ancestor elements between <span> and <div>

1. children(): direct children
2. find(): returns descendant elements of the selected element all the way down to the last descendant

* $(“div”).find(“span”)

//returns all <span> elements that are descendants of <div>

* $(“div”).find(“.name”)

//dot in front if input is class

1. prev(): searches for the previous sibling
2. next():returns the next sibling element  
       $("h2").next();

### Filtering

1. first(): returns the first element

$("div p").first();

1. last(): return the last element

$("div p").last();

1. eq(): return an element with a specific index of selected elements

$("p").eq(1); //return the second

1. filter(): specify a criteria, return the matched.

$("p").filter(".intro"); //return the <p> with intro class

1. not(): return the unmatched

$("p").not(".intro"); //return the <p> that do not have the class “intro”

## CSS Class

1. add class: addClass() - Adds one or more classes to the selected elements

$("button").click(function(){  
    $("h1, h2, p").addClass("blue");  
    $("div").addClass("important");  
});

## HTML

1. Set Attributes - attr()

$("#w3s").attr("href", "http://www.w3schools.com/jquery");

1. append()-inserts content AT THE END of the selected HTML elements.

$("p").append("Some appended text.");

appendTo()

$('<img>').attr('src', 'images/upsideDownDog.jpg').appendTo('footer');

1. prepend() - Inserts content at the beginning of the selected elements

only prepend the text

$("p").prepend("Some prepended text.");

1. prependTo(): insert HTML elements
2. before() = insertBefore

can create new elements

1. create a new element: using <>

$(‘<img>’)

1. remove() - Removes the selected element (and its child elements)
   1. $(“#div1”).remove();
   2. $('#toc li:contains("Pig and Pepper")').remove();
      1. Isolate particular link by its text: contains()
2. empty() - Removes the child elements from the selected element
   1. $(“#div1”).empty;
3. Replace: replaceWith()
   1. $(“div.second”).replaceWith(“<h2>New Heading”</h2>);
4. Clone() – clone all that type element
   1. $(“p”).clone().appendTo(“body”);

//clone all <p> elements and insert them at the end of the <body> element.

## Event

### Mouse Event

1. Click()

$(“#target”).click(function(){

Alert(“Handle for .click() called.”);

})

1. .dblclick()

Double click

1. .hover()
2. .mousedown: when the mouse pointer is over the element, and the mouse button is pressed
3. .mouseenter: when the mouse pointer enters the element
4. .mouseleave: when the mouse pointer leaves the element
5. Mouseout: when the mouse pointer leaves the element, trigger the event on descendant element

### Form Event

1. Submit()

$(“#target”).submit(function(event){

//#target: form id

Alert(“Handler for .submit() called.”);

});

1. .blur()

* Sent to an element when it loses focus
* Mostly using on <input> or other form elements (when the move the cursor away from the textbox)

1. Event.preventDefault()

Stop the form submitting and reloading. If no preventDefault(), the page will refresh after clicked submit.

## Effect

* Delay() : set a timer to delay execution
  1. $(“div.first”).slideUp(300).delay(800).fadeIn(400);

//delay time: the timer between these two actions

* FadeIn(): display elements by fading them to opaque
* FadeOut(): hide the element by fading them to opaque;
* FadeTo(duration, opacity [,easing][,complete]): adjust the opacity of the elements
* Duration: string “slow”, or number
* Opacity: number
* complete: a function to call once the animation is complete
* Hide(): hide the matched elements
* Show(): display the matched elements
* Toggle(): can hide again
* SlideDown([duration], [complete]): display with sliding motion
* SlideToggle(): display and hide with sliding motion
* SlideUp(): hide with sliding motion
* .animate(): custom animation of a set of CSS properties
  1. .animate(properties [,duration][,easing][,complete])
  2. $(“#book”).animate({

Opacity: 0.25, left: “+=50”

}, 5000);

$(this).next().slideDown().animate({fontSize:'+=2'},'slow');

* 1. For the displayed element(.slideDown().animate())

# Event Handling

## Handling Events: Method 1

<button onclick=”alert(‘Hello, world’);”>

Run Some JavaScript

</button>

## Handling Events: Method 2

document.onclick = function(){

alert(“You clicked somewhere in the document”);

}

//click the specific element to trigger event handler

var myImage = document.getElementById(“mainImage”);

myImage.onclick = function(){

alert(“You clicked the image”);

}

### Issue 1

If you put <script src="script.js"></script> within <head>…</head>, the browser cannot get the javascript executed. Because the browser is trying to execute this javascript as soon as it gets to it. It’s trying to grab this element called mainImage, but it’s trying to run that around within <head>…</head>, before it passes the rest of the page (mainImage is inside the <body>…</body>). It does not have an element called mainImage so far. So it cannot add itself to onclick.

#### Solution

//It is going to make sure that function does not get called until the document has completely loaded.

//You can only write this windows.onload once per page

function prepareEventHandlers(){

var myImage = document.getElementById(“mainImage”);

myImage.onclick = function(){

alert(“You clicked the image”);

}

}

windows.onload = function(){

//prep anything we need to

prepareEventHandlers();

}

## Handling Events: Method 3

document.addEventListener(‘click’, myFunction, fasle);

document.addEventListener(‘click’, anotherFunc, fasle);

document.removeEventListener(‘click’, anotherFunc, fasle);

//Internet Explore 8 and previous

document.attachEvent(‘onclick’, myFunction);

## Cross-Browser add Event Helper Methods

function addCrossBrowserEventListener(elementName, eventName, functionName){

//dose the addEventListener function exist?

if(elementName.addEventListener){

//yes – use it

elementName.addEventListener(eventName, functionName, fasle);

return true;

}else{

//otherwise use attachEvent for IE8 or previous

elementName.attachEvent(“on” + eventName, functionName);

return true;

}

}

# onBlur and onFocus events

You are filling out a form on a web site, one or more field has a value already pre-filled in, prompting you what to do. You can either click in there or tab into that location and that text will disappear. However, if I tab out of it again, the text comes back to prompt me that I need to put something in there.

Every one of our form elements here gets an onfocus event kicked off when we click into it, when we leave it the element get the onblur event.

## Html:

<p>

<label for="email">Email:</label>

<input type="text" value="your email" name="email" id="email" tabindex="20" />

</p>

## Javascript

var emailField = document.getElementById("email");

emailField.onfocus = function() {

if ( emailField.value == "your email") {

emailField.value = "";

}

};

emailField.onblur = function() {

if (emailField.value == "") {

emailField.value = "your email";

}

};

# Hiding and showing form sections

## Html

<div id=”tourSelection”>

…

## JavaScript

function preparePage(){

document.getElementById(“brochures”).onclick=function(){

if(document.getElementById(“brochures”).checked){

}

}

}

windows.onload=function(){

preparePage();

}

# UI Enhancement

## Setting Inline Styles

myElement.style.color = “red”;

myElement.style.color = “#ff0000”;

myElement.style.left = “40px”;

myElement.style.backgroundRepeat = “repeat-x”;

//背景图片是否重复或拉伸 (no-repeat, repeat, repeat-x, repeat-y)

## Style Property Naming

### CSS:

#example {

width: 230px;

color: #fff;

font-weight: bold;

background-color: #193742;

}

### JavaScript:

//hyphens->camelCase

myElement.style.width = “230px”;

myElement.style.color = “#fff”;

myElement.style.fontWeight = “bold”;

myElement.style.backgroundColor = “#193742”;

## Setting the Class

myElement.className = “someCSSclass”;

//whatever the name of the class is in your CSS file

//clear the class name

myElement.className = “”;

## Ex1:

//when click, mainContent will apply another CSS style (class)

function preparePage() {

document.getElementById(“mainContent”).onclick = function(){

if(document.getElementById(“mainContent”).className == “example”){

document.getElementById(“mainContent”).className ==””;

} else {

document.getElementById(“mainContent”).className = “example”;

}

};

}

windows.onload = function(){

preparePage();

}

# Minifying your code

## Minification tools

JSMin

YUI Compressor

Google Closure Compiler

# Code Checker

## Jslint

### Options

Assume a browser

### Issue

Expected ' "use strict"; ' before 'document'.:

“use strict”;

document.getElementById("join").style.position = "absolute";

# JavaScript Libraries

## JQuery

### link jquery to html

<script src=”jquery-1.6.1.min.js”></script>

<script src=”script.js”></script>

##### using http and https pages

<script src=”//ajax.googleapis.com/ajax/libs/jquery/1.6.1/jquery.min.js”></script>

### Regular javascript vs jquery

#### javascript:

document.getElementById(“myDiv”).className=”highlight”;

//need a id=myDiv in html

//work on individual element

#### jquery:

jQuery(“#myDiv”).addClass(“highlight”);

//Or

$(“#myDiv”).addClass(“highlight”);

/\*we can give jquery an id to find by using the hash or $, but we can also find other part of the page that we can’t get with getElementById \*/

//work on a group of elements

//get all elements with a particular class

JQuery(“.someClass”)

//get all elements with a particular tag

JQuery(“p”) //all paragraph

JQuery(“p:contains(‘packages’)”).addClass(“highlight”);

JQuery(“a”)//all anchors

JQuery(“li”).addClass(“highlight”);

JQuery(“li:last”).addClass(“highlight”); //add the class to the very last one

//give all the paragraph that have a class of description already on them

jQuery(“p.description”)

//take 4000ms to hide all paragraphs

$(“p”).hide(4000);

//work with function

$(“h2”).click(function(){

// $(this).text(“You clicked me!”);

})

# Strict Mode

1. declare variable before call it
2. cannot define a function with multiple parameters with the same name

## Turn on strict mode for entire file

//add

“use strict”;

//to top of the javascript file

# Regular Expression

## Creating Patterns

var myRe = /^hello/;

// hello appears at the start of the string

var myRe = /hello$/;

// at the end

var myRe = /hel+o/;

//previous character appears once or more: helo, hello, hello

var myRe = /hel\*o/;

//previous character appears zero or more: heo, hello, helllllo

var myRe = /hel?o/;

//zero or one, “heo”, “helo” match

//”hello”, “helllllo” would not match

/hello|goodbye/

//either, or

/he..o/;

//. any character

/\wello/;

// \w alphanumeric or \_

/\bhello/;

// \b word boundary, like space, new line

/[crnld]ope/;

// […] range of chars to match on

# AJAX

Example: Google, and started typing, and found it able to start to fill out the information that it thinks you are looking for and actually update the page as you are typing

## Create the request

var myRequest;

//feature check!

if (window.XMLHttpRequest) {

//Firefox, Safari

myRequest = new XMLHttpRequest();

} else if (windows.ActiveXObject) {

//IE

myRequest = new ActiveXObject(“Microsoft.XMLHTTP”);

}