Mr Baumgarten's handy Javascript cheat book! (v4)

Because coding can be challenging enough without trying to memorise everything!

General structure

Inside a valid HTML file:

```
<!doctype html>
<html>
   <body>
      <h1>Your HTML website goes here</h1>
   </body>
   <script type="text/javascript">

'use strict';

function myapp() {
   // Your app goes here
}

window.onload = myapp;

   </script>
</html>
```

Creating functions

```
function doesSomething() {
   // function code goes here
   return( result ); // optional
}
```

example:

```
function addTwoNumbers( a, b ) {
   return( a+b );
}
```

There are a number of function creating alternative syntaxes (too many... creates too much confusion, but anyway). An alternative within es5 is:

```
var addTwoNumbers = function( a, b ) {
   return( a+b );
}
```

Using arrow functions (es6) with block body:

```
var addTwoNumbers = (a, b) => { return( a+b ); };
```

Using arrow functions (es6) with concise body (for 1 line functions):

```
var addTwoNumbers = (a, b) => a+b;
```

Because functions can exist as members of objects, they following are also valid...

```
var app = {
   addTwoNumbers : function( a, b) { return (a+b); }
};
var result = app.addTwoNumbers( 4, 8);
```

And in ES6 this would look like...

```
var app = {
   addTwoNumbers( a, b) { return (a+b); }
};
var result = app.addTwoNumbers( 4, 8);
```

Repetition

For loop

```
for ( initialise ; condition ; execute-per-loop ) {
}
```

Example:

```
for (var i=0 ; i<10 ; i=i+1 ) {
   console.log( i );
}
```

While loop
The following is equivalent to the above for-loop:

```
var a = 0;
while ( a < 10 ) {
   a = a + 1;
   console.log( a );
}
```

Selection

If statement

```
if ( condition-is-true ) {
    console("Hey, it was true");
}
```

If - else statement

```
if ( condition-is-true ) {
    // do something
} else {
    // do something else
}
```

If - else if - else statement

```
if ( condition-is-true ) {
    // do something
} else if ( other-condition-is-true ) {
    // do this if the other thing is true
} else {
    // otherwise do this
}
```

The ternary operator

`var result = (condition) ? result_if_true : result_if_false ;`

```
var largerOfTheTwo = (a>b) ? a : b;
```

Simple variables & operations

Note: From ES6 onwards, the `var` keyword should be replaced with `let` or `const`

Numbers

```
var a = 4;
var b = 10;
```

Numeric operations

```
var answer = a + b;     // addition
var answer = a - b;             // subtraction
var answer = a * b;             // Multiplication
var answer = a / b;             // Integer division
var answer = a % b;     // Modulus (remainder)
var answer = Math.pow(a,b);// Exponential
var answer = Math.sqrt(a);      // Square root

var answer = Math.round( 13.4 );
var answer = Math.abs( -13 );
var random = Math.random();     // Between 0 & 1

// Trigonometry
var pi = Math.PI;
var angle = Math.sin( opp / hyp );
var angle = Math.cos( adj / hyp );
var angle = Math.tan( opp / adj );
var ratio = Math.asin( angle );
```

```
var ratio = Math.acos( angle );
var ratio = Math.atan( angle );
```

*** All angles will be in radians not degrees

Strings

```
var s1 = "hello";
var s2 = "What does the fox say?";
```

String operations

```
s1.length;                // 5
s1.charAt(0);             // "h"
s1.charCodeAt(0);         // 104
s2.indexOf("o");          // 6 … first occurance of "o"
s2.lastIndexOf("o");      // 15 … last occurance of "o"
s2.slice(14,17);          // "fox"

s2.replace("fox","goat"); // = "What does the goat say?"
s2.toUpperCase();             // = "WHAT DOES THE FOX SAY?"
s2.toLowerCase();             // = "what does the fox say?"

s2.split(" ");            // =["What","does","the","fox","say?"]

var s3 = "Hi there! "+s2; // ="Hi there! What does the fox say?"
```

Casting

```
var num = Number("10");         // String to num
var s = num.toString();         // Num to string
```

Dates

```
var today = new Date(); // todays date
var today = new Date("2016-03-23"); // 23 March 2016
var today = new Date(yr, mnth-1, date, hr, min, sec);
```

Milliseconds since 00:00:00 01.01.1970 UTC (the "epoch")

```
var millis = 1492978013000;       // 2017-04-23 20:06:53
var millis = Date.now();                  // the live time
var today = new Date( millis );           // Converts to date/time obj
var millis = today.getTime();     // Converts back to millis
```

See https://www.epochconverter.com/

Date functions

```
var today = new Date();
today.setFullYear(2016);
today.setMonth(02); // **
today.setDate(23);
today.setHours(08);
today.setMinutes(25);
today.setSeconds(0);
var y = today.getFullYear();
var m = today.getMonth(); // **
var d = today.getDate();
var h = today.getHours();
var n = today.getMinutes();
var s = today.getSeconds();
```

note: The months with Javascript start with a 0 for January, 1 for February through to 11 for December. Everyone always forgets this and gets annoyed at why their program doesn't work as expected. Try to remember it. I never do 😟

```
var today = new Date();
var todayString = today.toLocaleDateString();
```

A "Locale" date is one that is determined by the users language settings. For instance, English (UK) will get "dd/mm/yyyy" where as English (US) will get "mm/dd/yyyy". You can specify the locale by providing a coded String to the function, as the following two demonstrate with the Time equivalent.

```
// US English uses 12-hour time with AM/PM
console.log(today.toLocaleTimeString('en-US'));
// → "7:00:00 PM"

// British English uses 24-hour time without AM/PM
console.log(today.toLocaleTimeString('en-GB'));
// → "03:00:00"
```

Using Unicode

Unicode characters are a quick and easy way to use glyphs, emoji and other symbols in your app without having to create them yourself. Once you know the symbol codes it's just a case of using this code:

```
String.fromCodePoint( 0x1f602 );        // Smiling tears of joy
String.fromCodePoint( 0x1f525 );        // Fire
String.fromCodePoint( 0x1f6c1 );        // Bathtub
String.fromCodePoint( 0x1f354 );        // Cheeseburger
String.fromCodePoint( 0x26bd ); // Soccer ball

String.fromCodePoint( 0x1f1e8 ) +
String.fromCodePoint( 0x1f1ed );        // Swiss flag

String.fromCodePoint( 0x1f1e8 ) +
String.fromCodePoint( 0x1f1ed );        // Swiss flag

String.fromCodePoint( 0x1f1e6 ) +
String.fromCodePoint( 0x1f1fa );        // Australian flag
```

To find the required code, or to browse the available list, visit http://emojipedia.org/ and scroll to the bottom of the page for any emoji to find it's "codepoint".

There are also other "non-emoji" symbols that could come in useful, so search sites such as: https://unicode-table.com/en/#miscellaneous-symbols

See my video on the subject for a demonstration.

Arrays

```
var a = ["dog", "cat"];
```

or

```
var a = [];
a[0] = "dog"; // 1st item has index of 0
a[1] = "cat";
```

or

```
var a = [];
a.push("dog");
a.push("cat");
```

Returns a comma separated list
`a.toString();`

Returns new array with items added
`a.concat(item1[,item2[,... );`

Removes and returns last item
`a.pop();`

Add one or more items to the end
`a.push(item1,... );`

Reverse the array
`a.reverse();`

Remove and return the first item
`a.shift();`

Return a sub-array
`a.slice(start[,end]);`

Sorts the array
`a.sort();`

Iterate over the array asynchronously with a callback

```
a.forEach( function(element, index) {
   // Do something
});
```

Iterate over the array synchronously (warning not guaranteed that it will iterate over the array in sequential order)

```
for (var index in a) {
   var element = a[index];
}
```

Iterate over the array synchronously and in order

```
for (var index of a) {
   var element = a[index];
}
```

Two dimensional arrays

Two dimensional array where values are pre-known

```
var arr = [ [0,0,0,0], [1,1,1,1], [2,2,2,2] ];
```

Two dimensional array where values are not pre-known
(Iterate over the 1st array to create the 2nd)

```
var marks = new Array(10);
for (var i = 0; i < marks.length; i++) {
  marks[i] = new Array(5);
}
```

JSON

Creating a JSON object

```
var obj = {};
obj.name = "Mr B";
obj.title = "Computer Science teacher";
```

The following is equivalent to the above

```
var obj = {};
obj["name"] = "Mr B";
obj["title"] = "Computer Science teacher";
```

The following is equivalent to the above

```
var obj = {
   "name" : "Mr B",
   "title": "Computer Science teacher"
};
```

JSON object to JSON string
`var str = JSON.stringify( obj );`

JSON string to JSON object
`var obj = JSON.parse( str );`

AJAX

AJAX: JSON retrieval

```
function ajax( url, callback ) {
    var xhttp = new XMLHttpRequest();
    xhttp.onreadystatechange = function() {
        if (xhttp.readyState === 4 && xhttp.status === 200) {
            callback( JSON.parse(xhttp.responseText) );
        }
    };
    xhttp.open("GET", url, true);
    xhttp.send();
};
```

Example usage

```
ajax("http://api.fixer.io/latest?base=CHF", results);

function results(json) {
    console.log("Received the following data: ", json);
}
```

Canvas

In the HTML add:
`<canvas id="canvas" width="500" height="500"></canvas>`

```
var ctx = document.querySelector("#canvas").getContext("2d");
var w = ctx.canvas.width;
var h = ctx.canvas.height;

// Various properties

ctx.strokeStyle         = "pink";
ctx.fillStyle           = "green";
ctx.lineWidth           = 4;
ctx.font                        = "18pt sans-serif";
ctx.textAlign           = "left"; // center/right
ctx.textBaseline        = "bottom"; //middle/top
```

Draw a line

```
ctx.beginPath();
ctx.moveTo(50,50);
ctx.lineTo(450,50);
ctx.stroke();
```

Draw a rectangle

`fillRect( starting-x, starting-y, change-of-x, change-of-y );`

```
ctx.fillRect( 10, 100, 480, 50 );
```

Draw an ellipse

`ellipse( x-centre, y-centre, x-radius, y-radius, angle-of-rotation, start-angle, end-angle);`

```
ctx.beginPath();
ctx.ellipse( 100, 200, 40, 40, 0, 0, 2*Math.PI);
ctx.stroke();
```

Write text

```
ctx.fillText( message, 250, 20 );
```

Display an image from the DOM

`canvas.drawImage( imageObject, xposition, yposition, width, height);`

Add to your HTML:

```
`<img id="photo" src="http://www.com/path/to/image.jpg" style="display:none">`
```

And in Javascript:

```
var pic = document.getElementById("photo");
canvas.drawImage( pic, 50, 200, 150, 150 );
```

Display an image from url

`canvas.drawImage( imageObject, xposition, yposition, width, height);`

```
var pic = new Image();
pic.src = "http://www.com/path/to/image.jpg";
pic.onload = displayImage;
function displayImage() {
    canvas.drawImage( pic, 50, 200, 150, 150 );
};
```

Events

Mouse Events

Add mouse events to an element (eg button)

`document.querySelector("#go").addEventListener("click",addText);` ... or ...
`document.querySelector("#go").onclick = addText;`

Other useful events: submit, input, change, focus

Add mouse events anywhere in the doc

```
document.addEventListener("mousemove",move);
document.addEventListener("click", go );
document.addEventListener("mousedown",down);
document.addEventListener("mouseup",up);
```

Add mouse events to a canvas

```
ctx.canvas.addEventListener( ...etc );
```

Common properties in the function parameter

```
e.target.value
e.offsetX // releative to the target element
e.offsetY // releative to the target element
```

Keyboard events

Keyboard codes:

Key	Code	Key	Code	Key	Code
Left	37	Up	38	Right	39
Down	40	Delete	8	Tab	9
Enter	13	Shift	16	Ctrl	17
Alt	18	Esc	27	Space	32

Three main types of keyboard events to listen for:

- The KeyDown event is triggered when the user presses a Key.
- The KeyUp event is triggered when the user releases a Key.
- The KeyPress event is triggered when the user presses & releases a Key. (onKeyDown followed by onKeyUp)

Listening for keyboard events

- `document.addEventListener( "keydown", keyDown );` or `document.onkeydown = keyDown;`
- `document.addEventListener( "keyup", keyUp );` or `document.onkeyup = keyUp;`
- `document.addEventListener( "keypress", keyPress );` or `document.onkeydown = keyPress;`

The handler would look at the keyCode property as follows:

```
function keyDown(e) {
   lastKey = e.keyCode;
}
```

Timer events

Time delay is in milliseconds

Set a timer

`setInterval( timer, 40 );` ...or...
`var timerID = setInterval( timer, 40 );`

Clear timer

`clearInterval(timerID);`

One off timer

`setTimeOut( functionToExecute, 40 );`

Sounds

```
function soundChomp() {
    var music = document.createElement('audio');
    music.src = "https://pbaumgarten.cs.isl.ch//dist/pacman/pacman_chomp.mp3";
    music.loop = false;
    music.play();
}
```

DOM

```
<img id="pic" src="picture1.jpg">
<input type="text" id="msg">
<input type="button" id="go" value="Go">
<div id="output">The innerHTML goes here</div>
```

Get/set element values
`document.getElementById("output").innerHTML = "...";`

Get/set attributes
`document.getElementById("pic").getAttribute("src");`
`document.getElementById("pic").setAttribute("src", "picture2.jpg");`

With form fields such as the `<input>` tags, rather than using getAttribute, you can use a shortcut of .value such as:
`var msgBox = document.getElementById("msg").value;`

CSS styling
`document.getElementById("msg").style.display = "none";`

Browser dialogs
`window.alert( messageString );`
`window.prompt( messageString ); // returns what was typed`
`window.confirm( messageString ); // returns true or false`

Console
`console.log( messageString );`

Detecting screen / device

Return true/false if we are on a mobile device (all 1 line).
`var isMobile = navigator.userAgent.toLowerCase().indexOf('mobile') > 0 ? true : false;`

Get the browser window dimensions.
`var w = window.innerWidth;`
`var h = window.innerHeight;`

Detect if the window has resized (such as a phone/tablet being rotated) and adjust our width/height settings accordingly.

```
function adjustSize() {
   w = window.innerWidth;
   h = window.innerHeight;
}
window.onresize = adjustSize;
```

Web storage

Stored in browser. Can be seen/checked in Dev Tools.

Set
`localStorage.color = '#a4509b';` // OR
`localStorage['color'] = '#a4509b';` // OR
`localStorage.setItem('color', '#a4509b');`

Get
`var color = localStorage.color;` // OR
`var color = localStorage.getItem('color');`

The main catch to be aware of with localStorage is that it only stores key/value pairs. In other words you can only store Strings not complex JSON Objects or Arrays, unless you Stringify them first! Example set/get for a JSON object or array:

Set
`localStorage.somethingGreat = JSON.stringify(obj);`

Get
`var obj = JSON.parse(localStorage.somethingGreat);`

Geolocation

Test availability

```
if ("geolocation" in navigator) {
  /* geolocation is available */
}
```

Get current position (once)

```
navigator.geolocation.getCurrentPosition( whereAmI );

function whereAmI( position ) {
    var lat = position.coords.latitude;
    var lon = position.coords.longitude;
    console.log("You are at "+lat+" , "+lon);
}
```

Monitor current position (keep re-executing if I move)

```
var id = navigator.geolocation.watchPosition( whereAmI );

function whereAmI( position ) {
    var lat = position.coords.latitude;
    var lon = position.coords.longitude;
    console.log("You are at "+lat+" , "+lon);

    navigator.geolocation.clearWatch(id); // cancel monitor
}
```

Firebase

Setup

1. Goto https://console.firebase.google.com
2. Create an account
3. Create new project
4. Click “Add firebase to your web app”
5. Copy and paste the contents of the popup into your HTML file.

Google login

1. Requires the previous section, “Firebase: Setup”.
2. Log in to your firebase console (https://firebase.console.google.com)
3. Goto: Authentication

4. Goto: Sign in method
5. In the “Sign in providers” list, enable the one for Google.
   a. In the popup box, move the slider to enabled. You can leave the textboxes empty. Click Save.
6. In the “OAuth redirect domains” list, click “Add Domain”
   a. This is a list of websites that are approved to use your login system. In the popup, enter the address of your website. eg: myusername.cs.isl.ch
7. Now let’s setup your code.

Sample boiler plate for a simple Firebase login system:

```
<!DOCTYPE html>
<html>
<body>
    <div>Your webpage content goes here</div>
</body>
<script src="https://www.gstatic.com/firebasejs/3.6.4/firebase.js">
</script>
<script type="text/javascript">
    "use strict";

    var config = {
        apiKey: "---your-data-here---",
        authDomain: "---your-data-here---",
        databaseURL: "---your-data-here---",
        storageBucket: "---your-data-here---",
        messagingSenderId: "---your-data-here---"
    };

    firebase.initializeApp(config);

    function login() {
        function newLoginState( user ) {
            if (user) {
                // User is signed in.
                console.log("[user sign in]",user);
                app(user);
            } else {
                // No user is signed in.
                var provider = new firebase.auth.GoogleAuthProvider();
                firebase.auth().signInWithRedirect(provider);
            }
        }
        firebase.auth().onAuthStateChanged(newLoginState);
    }

    function app(user) {
        // Our function app will run once we have logged in

    }

    window.onload = login;

</script>
</html>
```

Save to & load from data store

1. Requires the previous section, “Firebase: Setup”.
2. You should also complete the “Firebase: Google login” section otherwise you will have zero security on your database - anyone can read/write your database without having to use your program.

3. Log in to your firebase console (https://firebase.console.google.com)
4. Database
5. Now let's setup your code.

The following is the code from my sample "what's your status" app in my video. The key lines showing how firebase have comments preceding them.

```
function app(user) {

    function receiveUpdate(received) {
        // Converts a firebase object into a regular JSON object for us to read from
        var data = received.val();

        document.getElementById("messages").innerHTML = "";
        console.log("[setData] Recieved = ",data);
        for (var key in data) {
            var person = data[key];
            var p = "<p>"+person.displayName;
            p += " ("+person.email+") said: "+person.status+"</p>";
            document.getElementById("messages").innerHTML += p;
        }
    }

    function setMyStatus(e) {
        var myUpdate = {};
        myUpdate.email = user.email;
        myUpdate.displayName = user.displayName;
        myUpdate.status = document.getElementById("myStatus").value

        // Saves the JSON object, upUpdate, to a firebase branch that is given the name of the
        fb.child( user.uid ).set( myUpdate );
    }

    // Creates the firebase object, fb.
    var fb = firebase.database().ref('samples/status');

    // Creates event handler. On change of value in FB, execute the function "receiveUpdate"
    fb.on('value', receiveUpdate);

    document.getElementById("myStatus").addEventListener("input", setMyStatus);
}
```

Coding with Chrome (Drawing API)

NOTE: This is not regular Javascript. If you are not using "Coding with Chrome" this page is not for you. You are probably looking for the canvas.

```
draw.rectangle( x, y, width, height, color, borderColor, borderSize );
draw.rectangle( 250, 200, 350, 350, '#00ff00', '#000000', 1 );

draw.circle( x, y, radius, color, borderColor, borderSize );
draw.circle( 150, 150, 100, '#ff0000', '#000000', 1 );

draw.line( startX, startY, endX, endY, color, borderSize );
draw.line( 50, 50, 300, 250, '#0000ff', 5 );

draw.point( x, y, color, borderSize );
draw.point( 25, 25, '#ff00ff', 10 );

draw.ellipse( x, y, width, height, color, borderColor, borderSize );
```

```
draw.ellipse( x, y, width, height, color, borderColor, borderSize );
draw.ellipse( 150, 150, 200, 100, '#ff0000', '#000000', 1 );

draw.triangle( x1, y1, x2, y2, x3, y3, color, borderColor, borderSize );
draw.triangle( 50, 50, 150, 150, 300, 100, '#ffff00', '#ff0000', 1 );

draw.text( string, x, y, color );
draw.text( 'hello world', 10, 30, '#aaaaaa' );

command.write( string );
command.write( 'Batman' );
```