My notes for Exam 70-480: Programming in HTML5 with JavaScript and CSS3

Note: these notes do not breach any agreement with microsoft. They were made *before* I took (and passed) the test on 2012-10-22. Some notes may be in swedish still, l
me know if you find any. Drop me a line or mention me on twitter (@Mellbourn) or Google+ (klas@mellbourn.net) if you find this guide useful. If you want to improve the
document, comment it and/or contact me for write access.
Many links are included below, but even more are at http://www.delicious.com/mellbourn/70_-_480
I have also made notes for 70-486 Developing ASP.NET MVC 4 Web Applications

This link list is a bit interesting
Free practice exam questions here and here
Maybe buy this test exam?
another suspicious practice exam site

Implement and Manipulate Document Structures and Objects (24%)

- Create the document structure.
  - This objective may include but is not limited to: structure the UI by using semantic markup, including for search engines and screen readers (Section, Article, Nav, Header, Footer, and Aside); create a layout container in HTML
    - The container defines how wide the Web page contents will be, as well as any margins around the outside and padding on the inside
      - ```
        #container {
        width: 870px;
        margin: 0 0 0 20px; /* top right bottom left */
        padding: 0;
        }
        ```
    - **SEO**
      - unique `<title>` for each page
      - `<meta name=”description” content=”Brandon’s Baseball...`
      - urls with words, use a single url for a page (301 to the correct one)
      - easy to navigate (flat hierarchy, with breadcrumb)
      - have a Sitemap file (xml description of navigation)
      - rel=nofollow on links in comments
    - **semantic markup HTML5**
      - <article>
      - <aside>
      - <section>
      - <figure><figcaption>
      - <nav>
      - <fieldset><legend> (groupbox)
        ```

        - <label for="inputfieldid"
    - **semantic markup ARIA**
        - roles: dialog, directory, grid, heading, main, menu, tree
        - states & properties: aria-autocomplete, aria-checked, aria-haspopup
        - landmark roles: role=application, banner, form, main, navigation, search
        - live regions: alert, log, marquee
            - mark regions with `aria-live='polite' // assertive`
            - type of update: *relevant="additions"*
            - *aria-busy=true* during updates
        - alt="" when purely decorative
        - aria-labelledby aira-describedby
- Write code that interacts with UI controls.
    - This objective may include but is not limited to: programmatically add and modify HTML elements; implement media controls; implement HTML5 canvas and SVG graphics
        - add and modify
            - appendChild
            - removeChild
        - media controls

```
<video>
  <source src="video.mp4" type='video/mp4' />
  <source src="video.webm" type='video/webm' />
  <object type="application/x-silverlight-2">
    <param name="source" value="http://url/player.xap">
    <param name="initParams" value="m=http://url/video.mp4">
  </object>
  No native support, download the video <a href="video.mp4">here</a>.
</video>
```

        - video tag attributes: `autoplay, controls, muted, poster, loop`
            - commands `play() pause()`
        - HTML5 canvas
        - `<script type="text/javascript">`
        - `var c=document.getElementById("myCanvas");`
        - `var ctx=c.getContext("2d");`
        - `ctx.fillStyle="#FF0000";`
        - `ctx.fillRect(0,0,150,75);`
        - `ctx.moveTo(0,0);`
        - `ctx.lineTo(300,150);`
        - `ctx.stroke();`
        - `ctx.beginPath();`
        - `ctx.arc(95,50,40,0,2*Math.PI);`
        - `ctx.stroke();`
        - `var grd=ctx.createLinearGradient(0,0,200,0);`

        - `grd.addColorStop(0,"red")`
        - `</script>`
        - SVG
        - examples
            - <svg xmlns="http://www.w3.org/2000/svg" version="1.1">
              <circle cx="100" cy="50" r="40" stroke="black" stroke-width="2" fill="red" />
            -
            - <**g** fill="none">
            - <path stroke="red" d="M5 20 l215 0" />
            -
            - <**defs**>
            - <filter id="f1" x="0" y="0">
            - <feGaussianBlur in="SourceGraphic" stdDeviation="15" />
            - </filter>
            - </defs>
            - <rect width="90" height="90" stroke="green" stroke-width="3" fill="yellow" filter="**url(#f1)**" />
- Apply styling to HTML elements programmatically.
    - This objective may include but is not limited to: change the location of an element; apply a transform; show and hide elements
        - css position
            - `static, absolute, fixed, relative // fixed is relative browser window, absolute relative parent`
        - location
            - `document.getElementById("movetext").style.left = 100px;`
        - apply a transform
        - `-webkit-transform: rotate(30deg);`
            - `translate(), scale(), skew(), matrix()`
        - show and hide
        - `display: block; display: none;` (and take up no space) `display: inline-block; display: list-item;`
        - `visibility: visible; visibility: hidden;` (but still take up space)
- Implement HTML5 APIs.
    - This objective may include but is not limited to: implement storage APIs, AppCache API, and Geolocation API
        - storage
            - `typeof(Storage)!=="undefined"`
            - `localStorage.foo = “bar”` permanent!
            - `sessionStorage` for the session
        - AppCache - cached until manifest changes!
            - `<html manifest=”demo.appcache”>`
            - `CACHE MANIFEST`
            - `# 2012-02-21 v1.0.0`
            - `/theme.css`
            - `/logo.gif`

- /main.js
-
- NETWORK:
- login.asp
-
- FALLBACK:
- /html/ /offline.html
- ```
  var appCache = window.applicationCache;

  switch (appCache.status) {
    case appCache.UNCACHED: // UNCACHED == 0
      return 'UNCACHED';
      break;
    case appCache.IDLE: // IDLE == 1
      return 'IDLE';
      break;
    case appCache.CHECKING: // CHECKING == 2
      return 'CHECKING';
      break;
    case appCache.DOWNLOADING: // DOWNLOADING == 3
      return 'DOWNLOADING';
      break;
    case appCache.UPDATEREADY:  // UPDATEREADY == 4
      return 'UPDATEREADY';
  ```
- appCache.swapCache(); // swaps
- appCache.update(): // update cache but does not reload window

■ Geolocation

- navigator.geolocation.getCurrentPosition(showPosition);
- function showPosition(position)
- {
- x.innerHTML="Latitude: " + position.coords.latitude +

■ setInterval(), clearInterval(), setTimeout() and clearTimeout

■ $(window).load(function(){   // executes after images have been loaded too

- Establish the scope of objects and variables.
  - This objective may include but is not limited to: define the lifetime of variables; keep objects out of the global namespace; use the “this” keyword to reference an object that fired an event; scope variables locally and globally
    - delete to undefine variables (but not globals defined without var)
- Create and implement objects and methods.
  - This objective may include but is not limited to: implement native objects; create custom objects and custom properties for native objects using prototypes and functions; inherit from an object; implement native methods and create custom methods
    - http://phrogz.net/JS/classes/OOPinJS.html
    - http://phrogz.net/JS/classes/OOPinJS2.html
    - `Cat.prototype = new Mammal();`
    - SuperCar.prototype = Object.create(Car.prototype);
  ```

        - object literal
            - `personObj={firstname:"John",lastname:"Doe"};`
            - shorthand for
            - `personObj=new Object();`
            - `personObj.firstname="John";`
            - `personObj.lastname="Doe";`
            - using a constructor:
            - `function person(firstname,lastname,age,eyecolor)`
            - `{`
            - `  this.firstname=firstname;`
            - `  this.lastname=lastname;`
            - `  this.changeName=changeName;`
            - `  function changeName(name)`
            - `  {`
            - `    this.lastname=name;`
            - `  }`

Implement Program Flow (25%)

- Implement program flow.
    - This objective may include but is not limited to: iterate across collections and array items; manage program decisions by using switch statements, if/then, and operators; evaluate expressions
        - iterate
            - `for (varName in list)`
            - `for (init; condition; increment)`
            - `do { … } while(exp)`
            - `while(exp) { … }`
        - ```
          switch (exp) {
            case 1:
              somehting;
            break;
            default:
              something;
          }
          ```
        - Falsy: '' (empty string), 0 (zero- number), null, undefined, NaN
        - Truthy: everything else, eg: "non empty string" (string), 1 (number), {}, []
- Raise and handle an event.
    - This objective may include but is not limited to: handle common events exposed by DOM (OnBlur, OnFocus, OnClick); declare and handle bubbled events; handle an event by using an anonymous function
    - Bubbling http://javascript.info/tutorial/bubbling-and-capturing
- Implement exception handling.
    - This objective may include but is not limited to: set and respond to error codes; throw an exception; request for null checks; implement try-catch-finally blocks
        - try {
        - throw “Err1”
        - } catch(err)
        - { if (err ==”Err1”)... }
        - finally { ...
          ```

- Implement a callback.
  - This objective may include but is not limited to: receive messages from the HTML5 WebSocket API; use jQuery to make an AJAX call; wire up an event; implement a callback by using anonymous functions; handle the “this” pointer
    - websockets:
      - (old tech: long polling) use websockets when low latency is important (games, chat, realtime). Note that you need to support high concurrency (many open websockets)
      - `var connection = new WebSocket('ws://h.com', ['soap', 'xmpp']);`
      - connection.onopen // event: now you are allowed to send
      - connection.onerror // event
      - connection.Send(‘your message”) // or binary buffer or blob
      - // server sending to browser:
      - `connection.onmessage = function(e) { console.log(e.data)`
- Create a web worker process.
  - This objective may include but is not limited to: start and stop a web worker; pass data to a web worker; configure timeouts and intervals on the web worker; register an event listener for the web worker; limitations of a web worker
    - link
    - `var worker = new Worker('task.js');`
    - `worker.postMessage(); // Start the worker.`
    - `worker.postMessage(‘data to the worker’);`
    - `worker.onmessage=function(event){ // handle data from worker`
    - `document.getElementById("result").innerHTML=event.data;`
    - `};`
    - `worker.terminate(); // stop the wiorker`
    - inside worker
    - `function timedCount()`
    - `{`
    - `i=i+1;`
    - `postMessage(e.data, i); // data back to page`
    - `setTimeout("timedCount()",500); // like setInterval`
    - `}`
    - `close(); // the worker stops itself`
    - Workers do not have access to: DOM, window, document, parent

Access and Secure Data (26%)

- Validate user input by using HTML5 elements.
  - This objective may include but is not limited to: choose the appropriate controls based on requirements; implement HTML input types and content attributes (for example, required) to collect user input
    - `<input type=”email” title=”this is the error message shown when the entered text does not look like an email”`
    - `<input type=”url”`
    - `<input type=”datetime”`
    - `<input type=”date” min=”2012-01-01” max=”2012-12-31”`
    - `<input type=”time” min=”08:00” max=”18:00” step=”1:00”`
    - `<input type=”number”`

        - `<input type="range" min="0" max="4"/>`
        - `<input type="color"/>`
        - `<input type="text" list="listan"/>`
        - `<datalist id="listan">`
        - `<option value="small">31</option>`
        - `<option value="medium">33</option>`
        - `<option value="large">35</option>`
        -
        - attributes `required disabled autofocus placeholder autocomplete pattern`
- Validate user input by using JavaScript.
    - This objective may include but is not limited to: evaluate a regular expression to validate the input format; validate that you are getting the right kind of data type by using built-in functions; prevent code injection
        - `var re5digit=/^\d{5}$/ //regular expression defining a 5 digit number`
          `if (document.myform.myinput.value.search(re5digit)==-1) //if match failed`
        - `var RegularExpression = new RegExp("pattern", ["switch"])`
            - `switches: i (ignore case), g (global replace), m (multiline mode)`
        - `c.replace(/javascript/, "JavaScript"); c.match, c.split, RegExp.test('stringtotest')`
- Consume data.
    - This objective may include but is not limited to: consume JSON and XML data; retrieve data by using web services; load data or get data from other sources by using XMLHTTPRequest
        - `JSON.parse()`
        - `JSON.stringify()`
        - `parser=new DOMParser();`
        - `xmlDoc=parser.parseFromString(txt,"text/xml");`
        - `xmlhttp=new XMLHttpRequest();`
        - `xmlhttp.open("GET","books.xml",false);`
        - `xmlhttp.send();`
        - `xmlDoc=xmlhttp.responseXML;`
- Serialize, deserialize, and transmit data.
    - This objective may include but is not limited to: binary data; text data (JSON, XML); implement the jQuery serialize method; Form.Submit; parse data; send data by using XMLHTTPRequest; sanitize input by using URI/form encoding
        - `binary data - use typed arrays(?)`
            - `var b = new ArrayBuffer(8);`
              `// create a view v1 referring to b, of type Int32`
              `var v1 = new Int32Array(b);`
            - `b.slice(); b.byteLength;`
        -
        - `$('form').submit(function() {`
        - `alert($(this).serialize());`
        - `return false;`
        - `});`
        - `=> a=1&b=2&c=3&d=4&e=5`

        - use these for encoding
            - encodeURIComponent
            - decodeURIComponent
        - These should rarely be used, since they operate on the whole URI (and thus do not encode “/”)
            - encodeURI(uri)
            - decodeURI(uri)

Use CSS3 in Applications (25%)

- Style HTML text properties.
    - This objective may include but is not limited to: apply styles to text appearance (color, bold, italics); apply styles to text font (WOFF and @font-face, size); apply styles to text alignment, spacing, and indentation; apply styles to text hyphenation; apply styles for a text drop shadow
        - `text-align: right, center, left, justify`
        - `text-justify: inter-word;`
        - `letter-spacing:-3px;`
        - `text-decoration: line-through, underline`
        - `text-transform: uppercase, lowercase, capitalize`
        - `text-indent: 50px; // indents the first line of text`
        - `p{font-family:"Times New Roman", Times, serif;} // multiple fallbacks. generic last (serif/sans-serif/monospace)`
        - `font-style: italic, normal`
        - `font-size // px/16=em` , because default size for text in browsers is 16pixels
        - `font-weight: bold`
        - shorthand:
            - `font: oblique 12pt "Helvetica Neue", serif; font-stretch: condensed`
        - ```
          @font-face {
            font-family: Gentium;
            src: url(http://example.com/fonts/Gentium.ttf);
          }
          ```
        - ```
          @font-face {
            font-family: MyGentium;
            src: local(Gentium Bold),   /* prefer to use local if available, full font name */
                 local(Gentium-Bold),   /* local Postscript name */
                 url(GentiumBold.ttf);  /* otherwise, download it */
            font-weight: bold;
          }
          ```
        - hypenation
            - `overflow-wrap/word-wrap: break-word // break long word to prevent overflow`
            - ~~`word-break:hyphenate; // break-all; normal; // wrong!`~~
            - ~~`rare: (word-break: break-word; // break-all; normal)`~~
            - `hyphens: auto; // will *create* hyphens in words (if language known).`
            - `hyphens: manual; // will only hyphenate if a soft hyphen is present  that suggests possible hyphenation`
        - `text-shadow: 2px 10px 5px brown; // x, y, blur radius, color`
- Style HTML box properties.

- This objective may include but is not limited to: apply styles to alter appearance attributes (size, border and rounding border corners, outline, padding, margin); apply styles to alter graphic effects (transparency, opacity, background image, gradients, shadow, clipping); apply styles to establish and change an element's position (static, relative, absolute, fixed)
    - `border-style: dotted, dashed, solid, groove, ridge, inset, outset`
    - `border: 5x solid red; // border-width border-style border-color`
    - `border-radius: 55px 25px // horizontal radius 55, vertical radius 25`
    - `outline: #00FF00 dotted thick; // outline-color outline-style outline-width`
    - `margin:25px 50px; // top and bottom marigns 25, right and left 50`
    - `margin:25px 50px 75px 100px; // top 25, right 50, bottom 75, left 100`
    - `opacity: 0.4; // 1 is opaque, not transparent at all`
    - `clipping`
        - ```
          img
          {
            position:absolute;
            clip:rect(0px,70px,200px,0px);
          }
          ```
    - `background-attachment: fixed; // background does not scroll`
    - `gradient`
        - `linear-gradient( to left top, blue, red);  /* A gradient going from the bottom right to the top left starting blue and finishing red */`
        - `background: linear-gradient( 45deg, blue, red );`
    - `box-shadow: 10px 10px 50x gray`
- Create a flexible content layout.
    - This objective may include but is not limited to: implement a layout using a flexible box model; implement a layout using multi-column; implement a layout using position floating and exclusions; implement a layout using grid alignment; implement a layout using regions, grouping, and nesting
    - flexible box model
        - ```
          footer {
            display: flex;  // flex it!
            flex-flow: row wrap; // row - horizontal, wrap - wrap items below if they do not fit
            align-items: stretch;  // stretch content vertically (alternatives: flex-start, center, flex-end)
          justify-content: space-around; // horizontal spread, alt: flex-start; flex-end; center; space-between;
          }
          ```
        - ```
          #second {
            flex: 2 300px; // prefer width 300px, if there is more take proportion 2 of it
          }
          ```
    - multiple columns
        - column-count: 2
        - column-width: 20px
        - column-gap: 5px
        - column-rule 3px outset red; // how the rule between columns should look
    - layout using CSS regions
        - `article {`
          ```

- ```
    flow-into: article_flow;
  }

  #region1, #region2, #region3, #region4 {
    flow-from: article_flow;
    region-fragment: break; // breaks off when overflowing
  }
  ```

- exclusions
  - ```
    wrap-flow: both; // forces text to flow around
    shape-outside: circle(50%, 50%, 50%); // text outside
    shape-inside: ellipse(...) // text inside
    shape-image-threshold: 0.5; // alpha where ok to text
    shape-margin: 10px;
    ```
- grid alignment
  - display:grid;
  - `grid-column-definition: auto minmax(min-content, 1fr);`
  - `grid-row-definition: auto minmax(min-content, 1fr) auto`
  - `#board    { grid-column-position: 2; grid-row-position: 1; grid-row-span: 2 }`
  - `templates`
    - ```
      grid-template: "title stats"
                     "score stats"
      #title    { grid-area: title }
      ```
  - `names`
    - ```
      grid-definition-columns:
                  "start"        auto
                  "track-start"  0.5fr
                  "thumb-start"  auto
        #lower-label { grid-column: "start" }
          #track       { grid-column: "track-start" "track-end"; align-self: center }
      ```
  - auto means size to content
  - fr means fraction of the redundant space that something gets
  - box alignment
    - `justify-self, align-self` aligns element within parent:
      - `justify-self: auto, start, center, end`
      - `align-self: head, foot, stretch`
    - `justify-content, align-content` content within element
      - `justify-content: auto, start, end, center, space-between, space-around`
      - `align-content: head, center, flex-start, space-between, space-around`
    - `justify-items, align-items` items inside element

        - css grouping:
          `h1,h2,p { color:green;`
        - css nesting
          `.marked p { ...`
- Create an animated and adaptive UI.
  - This objective may include but is not limited to: animate objects by applying CSS transitions; apply 3-D and 2-D transformations; adjust UI based on media queries (device adaptations for output formats, displays, and representations); hide or disable controls
    - animate objects by applying CSS3 Transitions
      - transition-property: all; // opacity left top
      - transition-duration 300ms;
      - transition-timing-function: ease-out;
      - transition: all 300ms ease-out; // slower at the end. ease-in, ease,
      - :hover { transform: scale(1) // must find a transform to tchange
    - 3-D transformations
      - `rotateX, rotate3d, translate3d, scaleZ, matrix3d, perspective`
    - `alternative animation:`
      - `animation-name:myfirst;`
      - `animation-duration:5s;`
      - `animation-timing-function:linear;`
      - `animation-delay:2s;`
      - `animation-iteration-count:infinite;`
      - `animation-direction:alternate;`
      - `animation-play-state:running;`
      - `@keyframes myfirst`
      - `{`
      - `0%   {background:red; left:0px; top:0px;}`
      - `25%  {background:yellow; left:200px; top:0px;}`
      - `50%  {background:blue; left:200px; top:200px;}`
      - `75%  {background:green; left:0px; top:200px;}`
      - `100% {background:red; left:0px; top:0px;}`
      - `}`
  - `detect` browser features and capabilities
  - in javascript
    - navigator.userAgent.indexOf("MSIE")>0
    - better to detect features and capabilities
      - `if(window.addEventListener) {  //supports`
      - `if(typeof window.addEventListener !== “undefined”)`
  - **if**(Modernizr.fontface)**{**
  - If a feature is lacking you can use **shims** (proprietarty emulator) or **polyfills** (exact HTML5 api emulator)
  - vendor specific extensions to CSS (-o-opacity = opacity for opera)
    - -moz-
    - -webkit-

        - -ms-
        - use all versions and then without prefix to make it work everywhere
    -
- CSS media queries
    - `@media only screen and (max-width: 850px) {`
    - `('screen' as opposed to 'print' or 'projection')`
    - `@media (orientation: landscape) {`
- set viewport in layout
    - `<meta name="viewport" content="width=device-width">`

- Find elements by using CSS selectors and jQuery.
    - This objective may include but is not limited to: choose the correct selector to reference an element; define element, style, and attribute selectors; find elements by using pseudo-elements and pseudo-classes (for example, :before, :first-line, :first-letter, :target, :lang, :checked, :first-child)
        - selectors
            - `:target` if the element id matches the #-tag in the url
            - `E[foo~="bar"]` matches <E foo="baz bar qux"
            - `E[foo*="bar]` matches anything with substring bar
            - `E:nth-last-of-type(n)` matches the E element that is the n:ht from the end of its type
            - `E:checked`
            - `E:first-line`
            - `E::before` insert content before
            - `E + F` an F *immediately* preceded by an E
            - `E ~ F` an F preceded by a E at any point
- Structure a CSS file by using CSS selectors.
    - This objective may include but is not limited to: reference elements correctly; implement inheritance; override inheritance by using !important; style an element based on pseudo-elements and pseudo-classes (for example, :before, :first-line, :first-letter, :target, :lang, :checked, :first-child)
        - you can force inheritance of a style from the parent `{ border:inherit; }`
        - jQuery specific pseudo-classes: `:has() :eq() [name!="value"] :animated :header: :first :gt() :header :hidden [type="image"] :last :lt() :odd :parent :selected :visible`

other interesting stuff:
`list-style`

`background-clip: border-box; // padding-box; content-box`

concat()
parseFloat to parse floats
toFixed(2) to represent a number with exactly two decimals
array
slice()
isNaN() to detect if user has entered letters after the number