Introducing HTML5: Chapter 1: Main Structure

* The HTML doctype:
  + <!doctype html>
  + That’s the whole thing
* The <meta> tag:
  + <meta charset=utf-8 />
  + Notice the lack of quotes, which are optional, and not case-sensitive.
* HTML5 is very relaxed. The <html>, <head>, and <body> tags are optional because browsers will add them in automatically. That said, continue to use these tags.
* Why the new tags?
  + A guy mined a billion web pages and found classes and IDs which were very common (look at the tables)
  + Accessibility advantages, hooks (readers can skip header/nav stuff, etc.)
* Example of the old-school vs new-school tags
  + (look at location 395 for examples)
  + CSS is cleaner as you can apply styles on the tag level instead of wrangling a bunch of classes
  + Note: The <script> element no longer requires you to specify the type of script
* Here they are (the new tags, that is):
  + <header>
    - “...represents a group of introductory or navigational aids”
    - can be used to wrap a section’s table of contents, search form, logos
    - there is also a <hgroup> tag to logically arrange <header> stuff
  + <nav>
    - Used to mark up navigation
    - there is an <aside> tag for asides (look at example at location 576
  + <footer>
    - “a footer for its nearest ancestor sectioning content or sectioning root element”
    - Like <header>, there can be more than one footer on a page
    - <small> has been redefined in html5 to mean “small print” or legalese
    - Check out the fashionable “fat footer” at thecssdiv.co.uk
  + <article>
    - “represents a component of a page that consists of a self-contained composition in a document, page, application, or site and that is intended to be independently distributable or reusable, e.g., in syndication.”
    - Blog posts, tutorials, news stories. Don’t think newspaper article, think article of clothing: a discrete item.

Introducing HTML5: Chapter 2: Text

* More new tags (whiteboard article example)
* <time> element
  + examples: <time datetime=2009-11-13>November 2009</time>
  + <time>20:00</time>
  + The datetime attribute takes a machine-readable date, yyyy-mm-ddThh:mm, see book for details.
* <footer> (not new, but used differently)
  + inside blockquotes, articles, you can even have two or more footers
* Sectioning with sectioning content: <article>, <section>, <nav>, <aside>
  + Groups things in a logical, table-of-contents-like outline
  + example: Syndication and article with <h1>, inside of <h2>
* <hgroup>
  + Used to logically group headers that aren’t really site-wide headings, but rather headers to articles and the like
  + example:  
    <hgroup>  
     <h1>Metafilter</h1>  
     <h2>community weblog</h2>  
    </hgroup>
* Sectioning roots: <blockquote>, <body>, <details>, <fieldset>, <figure>, <td>
  + These elements can have their own outlines but the sections and headings inside do not contribute to the outlines of their ancestors
* <section>
  + a way to logically separate sections of content within, for example, an article. A way to remove ambiguity
  + articles may be inside sections
* WAI-ARIA: Web Accessibility Initiative’s Accessible Rich Internet Applications
  + helps with accessibility
  + Role attribute, e.g., <span role=”heading”>this is a heading</span>
* <aside>
  + represents “a section of a page that consists of content that is tangentially related to the content around the <aside> element, which could be considered separate from that content.”
* <details>
  + provides an expanding/collapsing area
  + no implementations yet, drat the luck
* <figure>
  + A group that can contain an image, a caption (<figcaption>), and any other relevant content to do with the figure, such as a copyright notice, etc.
* <mark>
  + Used to simulate the use of a highlight marker, such as marking search results
* <ruby>, <rp>, <rt>
  + See example at location 1237 of 4600
* <address>
  + for contact details of author, not a generic element for postal addresses
* Removed from HTML5: <big>, <blink>, <center>, <font>, <marquee>, <applet>, and frames (but <iframe> remains)
* global attributes:
  + contenteditable: users can edit the contents of elements with this attribute
  + data-\*: you can pass information to scripts from markup using data-\* (see ch4)
  + draggable: an element can be dragged (see ch9)
  + role, aria-\*: set WAI-ARIA roles, etc.
  + spellcheck: boolean that tells browser to check the element’s spelling
  + tabindex: sets focus order of elements. Can use to programmatically set focus to elements that the browser would otherwise never tab to, good for screen readers.

Introducing HTML5: Chapter 3: Forms

* Update and Delete are added to form actions Get and Post
* Most form elements no longer required to be inside a form, can instead use attribute form, e.g., <input form=foo></input>
* New input types:
  + No requirements on error reporting, different browsers will behave differently
  + <input type=email>, validates email addresses. Empty is allowed unless required attribute is present
  + <input type=url>: Doesn’t have to be a web URL, could be tel: or ftp:, etc.
  + <input type=date>: Opera renders a calendar widget, Blackberry has a similar rendering. The browser should take care of the specifics for you, to ensure the date is passed back as machine-friendly
  + <input type=time>: allows input of time in 24-hour format, validates it
  + <input type=datetime>: validates a precise date and time
  + <input type=month>: nice because of localization, e.g., Janiver for French, etc.
  + <input type=week>: Opera renders a calendar and highlights a week based on the day you pick
  + <input type=number>: works with min, max and step attributes.
  + <input type=range>: will have a slider of some sort
  + <input type=search>: doesn’t really do anything, the difference between type text and type search is stylistic
  + <input type=tel>: expects a telephone number. No special validation going on, though, allows non-numeric characters. Provides a hook for some future functionality
  + <input type=color>: Brings up a color picker. Implemented on Blackberry only
* New attributes for <input>:
  + autofocus: can set focus to input on page load using this boolean attribute
  + placeholder: acts as a textbox hint or watermark that disappears when you set focus on the textbox
  + required: as you would expect, makes a form field a required field
  + multiple: allows multiple entries, for example, multiple files or email addresses
  + pattern: allows validation against a regular expression
  + autocomplete: lets you override the browser’s autofill settings in case you have a field used for sensitive data, etc.
* Modernizr:
  + Javascript library to test for existence of HTML5 features
  + [www.modernizr.com](http://www.modernizr.com/)
  + example: if(!Modernizr.inputtypes[email]) {// fall back on old validation, etc}
* Overriding browser default error messages:
  + see code example on 1783 of 4600
* Skip validation:
  + Use novalidate or formnovalidate

Introducing HTML5: Chapter 4: Video and Audio

* new <video> and <audio> tags
  + can replace <object> and <embed>
  + note: <embed> is now part of the HTML5 spec, will validate
* <video> can have styles applied, is open and can be downloaded like <img>
* Example with attributes:
  + <video src=myvideo.ogv, autoplay controls poster=videostill.jpg width=x height=y loop preload=”metadata” />
* Codecs: Ogg Theora/Ogg Vorbis, some Apple-specific codecs, new Google open-source codec VP8, when combined with vorbis it is webM.
* Can support multiple sources/codecs using <source> element like so:
  + <video>  
     <source src=video.ogv type=’video/ogg; codecs=”theora, vorbis”’>  
     <source …>  
     <p>Video not supported</p><!-- only displays if you don’t have any codecs-->  
    </video>
* Can specify lo-res for phones using media attribute, like so:
  + <video>  
     <source src=”hi-res.ogv” media=”min-device-width: 800px;”>  
     <source src=”lo-res.ogv”>  
    </video>
* New Javascript API to work with media elements

Chapter 5: Canvas

* new <canvas> element, 2d drawing context.
* uses include graphing tools, games, whatever has to do with drawing
* Too much to cover in a short time. Canvas could be its own study course. Fun but maybe not the highest priority subject.

Introducing HTML5: Chapter 6: Data Storage

* Cookies are bad
* The new thing the cool kids are using:
  + Web Storage (in all the latest databases)
  + Web SQL Databases (yes, that’s how you say it) still being considered by IE, FF
* First, Web Storage. Two kinds of storage:
  + sessionStorage: Available only to the window until window is closed.
  + localStorage: Available to all windows that are open on the domain, and is persistent after windows are closed
* More about Web Storage:
  + Factoid: Web Storage: Limit per domain exceeds 5MB vs cookies’ 4KB limit
  + NOTE: If cookies are disabled in Firefox, Web Storage throws security error
  + API:
    - readonly attribute unsigned long length;
    - getter DOMString key(in unsigned long index);
    - getter DOMString getItem(in DOMString key); // Note the type coercion
    - setter creator void setItem(in DOMString key, in any data);
    - deleter void removeItem(in DOMString key);
    - void clear();
  + Example uses:
    - sessionStorage.setItem(‘twitter’, ‘@rem’); alert(sessionStorage.getItem(‘twitter’)); // shows @rem
  + Alternate Expando usage:
    - sessionStorage.twitter = ‘@rem’;  
      alert(sessionStorage.twitter); // shows @rem
  + NOTE: Using a key with the name of ‘key’ is a bad idea
  + Store JSON-like object without JSON support using stringify method (see <http://www.json.org/json2.js)>
  + Firebug lets you look at your storage stuff in the DOM tab: window > Storage
* Secondly, Web SQL Databases!
  + Uses SQLLite syntax
  + Support is in progress for FF and IE
  + Open/create a database as follows:
    - var db = openDatabase(‘myDatabase’, ‘1.0’, ‘database description’, 2 \* 1024 \* 1024);
    - This creates a database called myDatabase, using version 1.0, that is 2 MB in size
    - The version number needs to be correct or an error is thrown
  + Check if openDatabase is supported: if (window.openDatabase) {...}
  + In order to run SQL operations on our newly-created database ‘db’, such as creating a table, you must start a “transaction” like so:
    - db.transaction(function (tx){  
         tx.executeSql(‘CREATE TABLE IF NOT EXISTS tweets  
        -> (id unique, screen\_name, date, tweet)’, [], function () { // this function is the               //‘success’ callback  
         // load the table with tweets. Note that the [] is SQL arguments, see next example  
        );  
      });
  + INSERT example:
    - tx.executeSql(‘INSERT INTO tweets (id, screen\_name, date, text)  
      VALUES (?, ?, ?, ?)’, [tweet.id, tweet.from\_user, time / 1000, tweet.text]);
    - The question marks map to the sql arguments in the brackets
  + Transactions:
    - if something fails inside a transaction, the whole transaction is rolled back