HTML: Structure, Syntax, and Semantics (Notes)

Part 1. HTML5 and Semantics

What are web semantics?

- They add meaning to web content through specific languages and syntaxes.
- They allow us to control the organization and display of our content.
- Allows search engines to more accurately index and represent your website.
- Allows screen readers and assistive technologies better access to your site.

Semantic HTML5

- HTML has just over 100 elements.
- The goal is to understand a tag's meaning and the intent behind using it.

HTML5 content models

- Originally there were just two content models:
    - Inline Level Elements
        - Appear within the flow of other content
    - Block Level Elements
        - Take up their own line within the flow of the document
- HTML5 has seven main content models:
    - Flow
        - All elements included in the normal flow of a document
        - Contains the majority of the elements in HTML5
        - Has no bearing on how the content is displayed within the user agent
    - Metadata
        - Sets up the presentation or the behavior of the rest of the content
        - Mostly in the <head>
        - Includes base, link, meta, noscript, script, style, and title tags.
    - Embedded
        - Any content that imports other resources into the document
        - Includes audio, canvas, embed, iframe, img, math, object, svg, video, etc.
    - Interactive
        - Any content specifically intended for some sort of user interaction
        - Includes a, audio, button, embed, iframe, img, input, keygen, label, object, select, textarea, video (e.g. if it had controls enabled)
    - Heading
        - Defines the header of a section
        - Includes h1, h2, h3, h4, h5, h6
    - Phrasing
        - Text of the document and any elements used to markup text within paragraph level structures
        - Includes a, abbr, area, audio, b, bdi, bdo, br, button, canvas, cite, code, data, date, datalist, del, dfn, em, embed, i, iframe, img, input, ins, kbd, keygen, label, map, mark, math, meter, noscript, object, output, progress, g, ruby, s, samp, script, select, small, span, strong, sub, sup, svg, textarea, time, u, var, video, wbr
    - Sectioning
        - This is content that defines the scope of the headings and footers
        - Includes article, aside, nav, section
            - All new tags to HTML5

- These help authors create more sophisticated document structures and to write more meaningful code.
- See www.w3.org/TR/html5/dom.html#kinds-of-content

Part 2. HTML5 Document Structure

Structuring HTML5 documents

- Sections and nested sub-sections
    - Example of <h1> through <h6>
- Headings can provide an outline, but htey don't group content.
- HTML5 contains new sectioning elements to help authors add definition to documents.
- HTML5 Sectioning Elements
    - <article> <aside> <nav> <section>
- HTML5 Semantic Grouping Elements
    - <footer> <header> <main>

Defining HTML5 documents

- There is little to no new information here. Instead this is meant for viewers to follow along with to obtain some practice with what has already been talked about...

Supporting legacy browsers

- Script to add in case of old version of IE
- Conditional comment to affect only IE
    - <!-- [if lt IE 9]><script src="X"></script><![end if]-->

Organizing content

- Previously designers focused on visual design and user experience, rather than content.
- Without properly organizing content on a page though you won't be able to structure the page in a meaningful way.
- Break content down into a hierarchy, including primary and secondary content to start.
- This allows you to create wireframes and thumbnail sketches easier.

Creating document sections

- This is an exercise to produce a foundational HTML file...
- <article> vs. <section>
    - Articles are essentially stand-alone content.
    - "The best description I've heard of it is, like an article of clothing, okay? If it can stand on its own, meaning, the content can be removed from the page and it is still understandable, it still represents itself, it could be syndicated, you could include it in an RSS feed, for example, and it would make sense for folks."
    - Sections group thematic content together.

    - “Here's the way I do it. You may be different than me, but this is the way I do it. If the content can stand on its own, if it is something that I would send to an RSS feed for somebody to read without having any of the rest of the page there, then I use an article tag. If, on the other hand, that it's a section within an article, or a thematic grouping, meaning all those elements belong together, but they need to be within their own section in the document outline, that's when I use a section element.”
    - Just use them consistently…
- <aside>
    - An aside element basically says that this content relates to its siblings, meaning, the content on the same level, but that it’s not really part of them. It just related in some way or another.

Identifying the main content

- <main>
    - Identifies main content (and contains articles, sections, etc. rather than replacing them)

Using headings properly

- Headings do most of the work to convey document structure to search engines and other devices.
- Having a strategy for how headings are gonna be used on your page is a critical part of structuring content correctly. It's something that you need to really consider before you begin actually authoring the content.

Building navigation

- <nav>
    - The nav element represents a section of navigation on the page.
    - Not just a collection of links.
    - You want to use it for main areas of navigation for the page or site.
- Menus are semantic groupings of links, meaning they’re a grouping of link elements that actually go together and relate to each other.
- Usually they are grouped together via lists, and specifically unordered lists, i.e.

Properly nesting structure

- There is little to no new information here. Instead this is meant for viewers to follow along with to obtain some practice with what has already been talked about...

Structuring headers

- Often, it's semantically desirable to use a header within other distinct page regions. I wanna explore the proper use of headers and the right way to structure their content.

- "The header element represents introductory content "for its nearest ancestor," so whoever it's inside of,"sectioning content or sectioning root," so articles, asides, block quotes, anything like that can have a header inside of it.
- "A header typically contains a group "of introductory or navigational aids." To me, that's really the important part. Is it introductory content? Or is it navigation? That sort of thing. Those are the two things that really fit within a header.
- You can add headers anywhere in the document that you have a group of introductory content, but usually would not add one for a single element (such as just an <h1> element).

Structuring footers

- In HTML5, footers are significantly different than headers in terms of the semantic role they play.
- "The footer element represents a footer for its "nearest ancestor sectioning content or sectioning element." Meaning, it's the footer for the element it's inside of.
- A footer typically contains information about its section, such as who wrote it, links to related documents, copyright data and the like. It's being pretty specific there. It contains information about its section and that information is stuff like who wrote it, disclaimers, copyright data, things like that. They can be longer; You'll notice below that it talks about if the footer contains entire sections, those are typically appendices, index, long colophons, license agreement and stuff like that.
- Footers do not have to be the last element in a section. They normally are, but there's no rule that they have to be.

Checking document structure

- Chrome Extensions: HTML5 Outliner ; HeadingsMap
- JavaScript Library: H50
- Firefox Add-On: HeadingsMap

Sectioning roots

- There's a group of elements that are known as Sectioning roots whose internal structure is considered independent of the regular document outline.
- Examples:
    - <blockquote>
    - <body>
    - <fieldset>
    - <figure>
    - <td>
- What that means is their internal structure is totally ignored when it comes to generating a document outline.

Part 3. Grouping Content with HTML5

Working with figure and figcaption

- <figure>
    - The figure element represents some flow content, optionally with a caption, that is self-contained (like a complete sentence) and is typically referenced as a single unit from the main flow of the document.
    - The element can thus be used to annotate illustrations, diagrams, photos, code listings, etc.
        - So it is really used to define content that illustrates the content that it is related to.
    - A lot of people associate images with figure, but it doesn't have to be that. It could be any content at all. Code examples, graphs, charts, video, whatever you have that's illustrating the content around it in some way, is perfect for the figure element.
- <figcaption>
    - The first ficaption element child of the element, if any, represents the caption of the figure element's contents. If there is no child figcaption element, then there is no caption.

Grouping content with asides

- "The aside "element represents a section of a page "that consists of content that is tangentially "related to the content around the aside element "and which could be considered separate from it.
- This is usually applied to content represented as sidebars in printed typography.
- This does not mean that in HTML it only can be applied to sidebars though.

Using divs in HTML5

-
    - Flow content that can contain anything.
    - "The div element has no special meaning at all, "it represents its children.""It can be used with the class, lang, and title "attributes to mark up semantics common to a group "of consecutive elements."
    - "Authors are strongly encouraged to view "the div element as an element of last resort, "for when no other element is suitable. "Use of more appropriate elements "instead of the div element "leads to better accessibility for readers "and easier maintainability for authors." So what it's trying to tell you is that, "Yes, the div is out there and it still works "the same way it's always worked, "but we would really like you to use "the new elements when they're appropriate.
- When to use a
    - Are you grouping content together that there is no suitable element for? If yes, a div might be right for you.

    - Typically, it's when you're grouping content together for JavaScript controls, like maybe creating some tabs or an accordion widget. And the other would be when you need to group content together simply for styling purposes.

Working with lists in HTML5

- List attributes: reversed, start, and type
- <dl>
    - A definition list element represents an association list consisting of zero or more name-value groups (a description list). A name-value group consists of one or ore names (dt elements) followed by one or more values (dd elements), ignoring any nodes other than dt and dd elements. WIthin a single dl element, there should not be more than one dt element for each name.
    - "Name-value groups may be "terms and definitions, metadata topics and values, "questions and answers, "or any other groups of name-value data." So what that really means is is as long as there's a clear relationship between the name-value pairs,almost anything can be described by a definition list.
- All right, so in certain instances the structure that the definition list, and really all the other lists in HTML5 affords you, allow you to structure your content in a way that better describes how the content relates to each other. That's a really important part of using lists. In our case, the definition list groups all of the links together, it lets people know that they're all related to each other, and we still get the advantage of using the name-value pairs that the definition list gives us. So while the changes that were made in the specification to both the ordered list and the definition list may seem a little minor, what they really do is they allow authors to continue using them, as they have in the past, with a clear conscience about being in conformance with the specification.

Emphasizing text correctly

- <b> <i>
    - Bold, italics, respectively.
    - Simply visual. No change of meaning or semantics.
-
    - Move from presentational tags to logical tags.
        - E.g. Screen readers change inflection of sentences based on these tags.
    - Have semantic value.
        - Screen reader example again.

Citing works semantically

- <q>
    - Adds proper quotation marks around quotes.
- <cite>
    - In the HTML5 specification, the cite element is used, really, to reference any creative work or an author of a creative work.

Using the address element

- <address>
    - The address element represents the contact information for its nearest article or body element ancestor. If that is the body element, then the contact information applies to the document as a whole.
    - My take on that would be when, semantically,you want a person, an individual, or a robot to be able to search through and find the proper contact information for an author of a particular document or article, then it's really the proper use for it.
    - You don't wanna wrap every bit of contact information in an address element, because that's really not what it's for. It's really for the contact information for the generator of a specific article or document.
    - The contact info just needs to be inside of it. So if you have some brief descriptive text, it's fine.
    - Within an address element, you can't have a lot of additional structures. So you can't use any sectioning elements,you can't use footers or headers, you're really looking at that being just flow content.

Using the small element

- <small>
    - The small element represents side comments such as small print, and specifically legal text or any other text that would normally be considered fine print.
    - Small print typically features disclaimers, caveats, legal restrictions or copyrights. Small print is also sometimes used for attribution or for satisfying licensing requirements.
    - That the small element does not de-emphasize or lower the importance of text emphasized by the em element or marked as important by the strong element.
    - The small element should not be used for extended spans of text.

Using the mark element

- <mark>
    - The mark element represents a run of text in one document marked or highlighted for reference purposes, due to its relevance in another context.
    - When used in a quotation or other block of text referred to from the prose, it indicates a highlight that was not originally present but which has been added to bring the reader’s attention to a part of the text that might not have been considered important by the original author when the block was originally written, but which is now under previously unexpected scrutiny.
        - So if you're quoting somebody and you wanna take part of that quote and emphasize it for the reader and it wasn't originally emphasized then the mark element is perfect for that.

    - When used in the main prose of a document, it indicates a part of the document that has been highlighted due to its likely relevance to the user's current activity.

Working with date and time

- <time> and <time datetime="X">
    - See here https://w3c.github.io/html/textlevel-semantics.html#the-time-element
    - Makes dates and times machine readable (e.g. for search engines)
    - You will never memorize this. Refer back whenever you need to.

Defining link relationships

- For definitions, see here: https://w3c.github.io/html/links.html#links-created-by-a-and-area-elements
- Can add semantic value and information between two linked resources.
    - https://w3c.github.io/html/links.html#allowed-keywords-and-their-meanings
- Examples
    - rel="stylesheet
    - rel="author"
    - rel="license"
    - rel="nofollow external"
- Rel Attributes and Search Engines
    - Now, the rel attribute can do a lot of things for us in terms of working with search engines, as well. A lot of the values that it is picked up in the HTML5 specification really are sort of a response to existing practices within the world of blogging. If I go up, for example, into the aside that holds our ad copy. This might be rotational ad copy. You might not really know who is going to be buying ads. They may be relevant to your site, they may be somebody you support.
    - But they might also be somebody that you really don't support or that isn't extremely relevant. A lot of times, you don't want search engines to relate your page with the link that might go out to ad content at all. So you can use the rel attribute to do that. If I go into the link here, our to melbikerentals.com, I can add a rel attribute to that. I can use the rel attribute of nofollow. Now, nofollow is basically telling search engines,"Don't follow this link and index it "in terms of how it relates to our page." Essentially, search engines really won't relate this content at all.
- You can have multiple rel values separated only by whitespace.

Part 4. Extending Meaning

Writing comments

- TBD

Working with meta tags

- TBD

Using class and ID attributes

- TBD

Using ARIA landmark roles

- TBD

Introducing microformats, microdata, and RDFa

- TBD

Using microdata

- TBD

Nesting rich data

- TBD

Checking page semantics

- TBD

Conclusion

- TBD