CS 328 HTML5 Coding Standards so far

- last modified: 2016-02-07
- You are required to use "strict-style" HTML5 for CS 328.
  - In general, your documents are expected to validate using the validators noted on bottom of the `html5-template.html` available from the course public web site. (There will be a very few exceptions, which will be noted as the semester continues.)
  - Your documents are also expected to include the links to the W3C experimental HTML5 validator and the CSS3 validator as well as the link to http://validator.nu/ as shown in example page `html5-template.html`.
- Remember that HTML5's intent is to describe a document's content and structure, not how it should look or behave.
- All elements are to be written in all-lowercase.
- All elements are required to be both explicitly opened *and* closed.
  - For elements that can have content, **both** start and end tags are required.
  - For elements that cannot have content, their tag is expected to end in /> (so that it serves as both start and end tag).
- All attribute values are to be written within quotes (either double quotes or single quotes).
- All content within an HTML5 document must be within an appropriate element.
- I am not requiring strict indentation standards for HTML5 yet (I am still trying to figure out a set that I like) -- avoid lines longer than 80 characters, and indent in some consistent, READABLE fashion.
- Use **semantic HTML** -- that is, choose which element to use based on what the content is, NOT how the element appears on the page!
  - (instead, use CSS3 as desired to change how elements appear on the page)
- Within an `a` (anchor) element, use link text content that **describes** the document being linked to.
- Use the `code` element for small, in-line fragments of computer code; use a `code` element nested within a `pre` element for multi-line fragments of code (where you want preservation of indentation and white space).
- HTML5 forms should not use the `table` element for layout purposes -- CSS3 should be used for such layout instead. The `table` element should be only used for truly-tabular data.
- Since the `summary` attribute for a `table` element is now deprecated, you are expected to include a `caption` element within each `table` element describing the table (to help with accessibility).
  - It is also a course style standard to include a `scope` attribute for `th` elements, indicating if that header is the header of a row (`scope="row"`) or of a column (`scope="col"`).
- For a `form` element whose contents are to be submitted, include at least one explicit `input` element of type `submit` (a submit button).

- Use appropriate `label` elements within `form` elements to logically associate text with form controls that should have text associated with them.
- Within a `form` element, choose form control elements that are logically/semantically appropriate for the information desired.
  - (e.g., use logically-grouped radio buttons when a user is selecting exactly one of a small number of options, use check boxes when a user may select 0 or more of a small number of options, etc.)
  - Use an `input` element of type `password` for password/sensitive data entry situations.
- Each (logical) control "element" within a `form` element that is to have a value submitted should have a `name` attribute with a unique name value (unique within that form).