Semantic Markup in HTML5

This workshop will introduce attendees to core HTML5 code and concepts, with an emphasis on getting hands-on experience with the new semantic elements.

**Learning Objectives**

In this workshop, attendees will:

- gain an understanding the recent history of HTML5 and WHATWG's approach to writing the spec.
- practice marking up content using new semantic elements in a text editor.
- be introduced to the semantic nuances of the new elements, including re-defined HTML4 elements.
- receive resources for styling new elements and learning more about HTML5.


**HTML(5), a history**

"The idea that HTML's evolution should be reopened was tested at a W3C workshop in 2004, where some of the principles that underlie the HTML5 work... as well as the aforementioned early draft proposal covering just forms-related features, were presented to the W3C jointly by Mozilla and Opera. The proposal was rejected on the grounds that the proposal conflicted with the previously chosen direction for the Web's evolution; the W3C staff and membership voted to continue developing XML-based replacements instead.

Shortly thereafter, Apple, Mozilla, and Opera jointly announced their intent to continue working on the effort under the umbrella of a new venue called the WHATWG. A public mailing list was created, and the draft was moved to the WHATWG site. The copyright was subsequently amended to be jointly owned by all three vendors, and to allow reuse of the specification.

The latter requirement in particular required that the scope of the HTML5 specification include what had previously been specified in three separate documents: HTML4, XHTML1, and DOM2 HTML. It also meant including significantly more detail than had previously been considered the norm.

In 2006, the W3C indicated an interest to participate in the development of HTML5 after all, and in 2007 formed a working group chartered to work with the WHATWG on the development of the HTML5 specification. Apple, Mozilla, and Opera allowed the W3C to publish the specification under the W3C copyright, while keeping a version with the less restrictive license on the WHATWG site.
Since then, both groups have been working together."
--

As of January 2011: "HTML is the main focus of the WHATWG community. HTML5 is a snapshot of HTML, which is being worked on by the WHATWG community and also the W3C HTML Working Group. Going forward, the WHATWG is just working on "HTML", without worrying about version numbers. When people talk about HTML5 in the context of the WHATWG, they usually mean just "the latest work on HTML", not necessarily a specific version."

# HTML Design Principles

"The WHATWG was based on several core principles, in particular that technologies need to be backwards compatible, that specifications and implementations need to match even if this means changing the specification rather than the implementations, and that specifications need to be detailed enough that implementations can achieve complete interoperability without reverse-engineering each other." -whatwg.org

Read more about the design principles at: http://www.w3.org/TR/html-design-principles

- Support existing content. Don't break the web. This means inoperability and consistent error-handling by all browsers.
- Degrade web content in older user agents gracefully.
- Don't reinvent the wheel; evolution not revolution.
- "Pave the Cowpaths" - adopt widespread usages rather than forbidding them
- Solve real problems web content faces today with pragmatic solutions.
- Priority of Constituencies: users > authors > implementors > specifiers > theoretical purity
- Address security concerns directly in the spec.
- Accessibility!

# Starting an HTML5 page in Dreamweaver

2. Page Type: HTML, Layout: <none> (these should be set by default)
3. Change DocType to HTML 5. Click "Create".
Getting Started: Your page structure

Important differences in syntax:
HTML5 is syntax agnostic; you can write HTML or XHTML style
- quoting attributes isn't required like in XHTML
- nor do you have to self-close the tag, like `<meta charset=utf-8 />`
- capitalizing letters is okay

(X)HTML5 Validator tool: http://html5.validator.nu

Doctype and Character Encoding

The Doctype:
```html
<!doctype html>
```
"Upgrading' to HTML5 can be as simple as changing your doctype. Upgrading to the HTML5 doctype won’t break your existing markup, because all the tags defined in HTML 4 are still supported in HTML5. But it will allow you to use — and validate — new semantic elements like `<article>`, `<section>`, `<header>`, and `<footer>`." - http://diveintohtml5.info/introduction.html

Character Encoding:
```html
<meta charset="UTF-8">
```
"You should always specify a character encoding on every HTML document, or bad things will happen. You can do it with the HTTP Content-Type header, the `<meta http-equiv>` declaration, or the shorter `<meta charset>` declaration, but please do it." - http://diveintohtml5.info/semantics.html
HTML5 gives us new elements that unambiguously denote landmarks in a page, evolving the current practice from re-using the semantically neutral <div> element over and over with made-up IDs and classes (which will still work just fine in HTML5).


```html
<body>
<header>
  <hgroup>
    <h1>Learning HTML(5)</h1>
    <h2>A page for practicing new elements</h2>
  </hgroup>
</header>
</body>
```

The `<header>` element represents a group of introductory or navigational aids. A header element is intended to usually contain the section's heading (an h1–h6 element or an hgroup element), but this is not required. The header element can also be used to wrap a section's table of contents, a search form, or any relevant logos. The spec also allows `<nav>` to be used in the `<header>`. `<header>` can be used multiple times on a page as necessary within other structural elements.

The `<hgroup>` element represents the heading of a section. The element is used to group a set of

Semantic Markup in HTML5 - 4
h1–h6 elements when the heading has multiple levels, such as subheadings, alternative titles, or taglines.... The point of using hgroup...[in this example] is to mask the h2 element (which acts as a secondary title) from the outline algorithm."

Using <hgroup>:
• grouping a title with related metadata
• grouping a title with subtitles

Wait, we can have more than one <h1> per page? Yes, but some browsers, Google and ATs have to catch up.
"In HTML 4, the only way to create a document outline was with the <h1>–<h6> elements. If you only wanted one root node in your outline, you had to limit yourself to one <h1> in your markup. But the HTML5 specification defines an algorithm for generating a document outline that incorporates the new semantic elements in HTML5. The HTML5 algorithm says that an <article> element creates a new section, that is, a new node in the document outline. And in HTML5, each section can have its own <h1> element." - http://diveintohtml5.info/semantics.html

Learn more about outlining content in HTML5: http://html5doctor.com/outlines
If you want to test your outline to be sure it's understood correctly, put your code in the HTML5 Outliner: http://gsnedders.html5.org/outliner

**Navigation**

```html
<nav>
  <h2>Menu</h2>
  <ul>
    <li><a href="#">About Me</a></li>
    <li><a href="#">Archives</a></li>
  </ul>
</nav>
```

The nav element represents a **section of a page that links to other pages or to parts within the page**: a section with navigation links. Not all groups of links on a page need to be in a nav element — only sections that consist of major navigation blocks are appropriate for the nav element. In particular, it is common for footers to have a short list of links to common pages of a site, such as the terms of service, the home page, and a copyright page. The footer element alone is sufficient for such cases, without a nav element. However, the use of <nav> is allowed in <header>.

More information about other semantic uses for <nav> are discussed at http://html5doctor.com/nav-element
Structure: Sectioning Content

<article>
The article element 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. This could be a forum post, a magazine or newspaper article, a Web log entry, a user-submitted comment, an interactive widget or gadget, or any other independent item of content.

<section>
The section element represents a generic document or application section. A section, in this context, is a **thematic grouping of content, typically with a heading**. Examples of sections would be chapters, the tabbed pages in a tabbed dialog box, or the numbered sections of a thesis. A Web site's home page could be split into sections for an introduction, news items, contact information.

When determining if you should put content in a <section> element, ask yourself "Is all of this content related?" If yes, use <section>, if not, use <div>.

What about <div>? "The section element is not a generic container element. When an element is needed only for styling purposes or as a convenience for scripting, authors are encouraged to use the div element instead." Some examples: A container for all page content, styling the introductory paragraph of an article.

Why is there no <content> element? "The first piece of content that's not in a <header>, <nav>, <aside> or <footer> is the beginning of the main content, regardless of whether it's contained in an <article>, a <div>, or whether it is a direct descendent of the <body> element." - pg, 9, *Introducing HTML5*

When to not use <section>:  
- Don't use it just as hook for styling or scripting; that's a div  
- Don't use it if article, aside or nav is more appropriate  
- Don't use it unless there is naturally a heading at the start of the section  
- Use the article element instead of the section element when it would make sense to syndicate the contents of the element.

http://html5doctor.com/the-section-element

<footer>
The footer element represents a **footer for its nearest ancestor sectioning content** or sectioning root element. A footer typically contains information about its section such as who wrote it, links to related documents, copyright data, and the like. Footers don’t necessarily have to appear at the end of a section, though they usually do. When the footer element contains entire sections, they represent appendices, indexes, long colophons, verbose license agreements, and other such
The time element represents either a time on a 24 hour clock, or a precise date in the proleptic Gregorian calendar, optionally with a time and a time-zone offset. If you want to include a time too, add the letter T after the date (or a space), then the time in 24-hour format, then a timezone offset.

There are three parts to a `<time>` element:
1. A machine-readable date (and time). If a time is specified, a timezone offset is required.
2. Human-readable text content
3. An optional pubdate flag

What does the pubdate attribute mean? It means one of two things. If the `<time>` element is in an `<article>` element, it means that this timestamp is the publication date of the article. If the `<time>` element is not in an `<article>` element, it means that this timestamp is the publication date of the entire document.

Source: http://diveintohtml5.info/semantics.html

Examples and more information about durations in `<time>` at: http://www.brucelawson.co.uk/2012/best-of-time

The aside element represents a section of a page that consists of content that is tangentially related to the content around it, and which could be considered separate from that content. Such sections are often represented as sidebars in printed typography. The element can be used for...
typographical effects like pull quotes or sidebars, for advertising, for groups of nav elements, and for other content that is considered separate from the main content of the page.

**The semantics of `<aside>` in and out of `<article>`:**
"With the new definition of aside, it's crucial to remain aware of its context. When used within an article element, the contents should be specifically related to that article (e.g., a glossary). When used outside of an article element, the contents should be related to the site (e.g., a blogroll, groups of additional navigation, and even advertising if that content is related to the page)." - http://html5doctor.com/aside-revisited/

```html
<mark>
`Mike once said:`
<p>Google won't last, <mark>they will fail</mark> at search and advertising as nothing will topple Yahoo.</p>
<p>Of course, we now know he was wrong. Google has not failed, they excelled in search and online advertising, making them a very profitable company.</p>
</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. 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.


Said another way, "The mark element doesn't attach any importance to the content within it, other than to show that it's currently of interest." - *Jeremy Keith, HTML5 for Web Designers, pg 60.*

Image and more discussion at http://html5doctor.com/draw-attention-with-mark
Text-Level Semantics: Redefining HTML4 Elements

<i> — was italic, now for text in an “alternate voice,” such as foreign words, technical terms and typographically italicized text

<b> — was bold, now for “stylistically offset” text, such as keywords and typographically emboldened text

<em> — was emphasis, now for stress emphasis, i.e., something you’d pronounce differently

<strong> — was for stronger emphasis, now for strong importance, basically the same thing (stronger emphasis or importance is now indicated by nesting)

<hr> — was horizontal rule, now used for a paragraph-level thematic break in text, "such as separating different topics within a section of prose, or between scenes in a novel. However you can use it anywhere you can use a <p>. While not widely used these days (given the dull default browser renderings), it can be replaced via CSS with an image."

For more examples, visit:
http://html5doctor.com/i-b-em-strong-element
http://html5doctor.com/small-hr-element

<small>

Using <small> around a Creative Commons license link with rel="license"

<small>a rel="license"
href="http://creativecommons.org/licenses/by-sa/3.0/">Creative Commons Attribution Share-alike license</a></small>

<small> — was for smaller text, now used for side comments and small print (legal disclaimers, caveats, copyright info, licensing, attribution).
• `<small>` is the inline equivalent of `<aside>`, content that is not the main focus of the page.
• `<small>` text does not need to be smaller than surrounding text — if all you want is smaller text use CSS instead.
• Use `<small>` only on inline content.
• Finally, `<small>` doesn’t affect the semantics of `<em>` or `<strong>`.

- http://html5doctor.com/small-hr-element

```
<address>

<footer>
  <address>
    For more details, contact
    <a href="mailto:js@example.com">John Smith</a>.
  </address>
  <p><small>© copyright 2038 Example Corp.</small></p>
</footer>
```

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. Typically, the address element would be included along with other information in a footer element.

The address element must not be used to represent arbitrary addresses (e.g. postal addresses), unless those addresses are in fact the relevant contact information. (The `<p>` element is the appropriate element for marking up postal addresses in general.) The address element must not contain information other than contact information.

- http://www.whatwg.org/specs/web-apps/current-work/multipage/sections.html#the-address-element

```
<figure> and <figcaption>

```html
<figure>
  <pre><code>&lt;hgroup&gt;
    &lt;h1&gt;The title of my arti
    &lt;h2&gt;A witty subtitle wit
  &lt;/hgroup&gt;&lt;/code&gt;&lt;/pre>
  <figcaption>Figure 1: An examp</figcaption>
</figure>
```

The figure element represents some flow content, optionally with a caption, that is self-contained 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, that are referred to from the main content of the document, but that could, without affecting the flow of the document, be moved away from that primary content, e.g. to the side of the page, to dedicated pages, or to an appendix.

The first figcaption 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.

- from the spec:
http://www.whatwg.org/specs/web-apps/current-work/multipage/grouping-content.html#the-figure-element

You should choose between <aside> or <figure> by asking yourself if the content is essential to understanding the section:

• If the content is simply related and not essential, use <aside>.
• If the content is essential but its position in the flow of content isn’t important, use <figure>.
• Having said that, if its position relates to previous and subsequent content, use a more appropriate element — e.g., a <div>, a plain old <img>, a <blockquote>, or possibly even <canvas>, depending on its content.

http://html5doctor.com/the-figure-figcaption-elements

**Styling in CSS**

```html
<!--[if lt IE 9]>
<script src="dist/html5shiv.js"></script>
<![endif]-->
```

CSS assumes that elements are display: inline, so we need to specify new block elements as such.

```css
article, aside, details, figcaption, figure, footer, header, hgroup, menu, nav, section
{display:block;}
```

Even when all browsers update their default stylesheets to display these elements as block by default, it’s still a good rule to leave in for older browsers (like pre-IE9 versions).

IE CSS Script for displaying and printing HTML5 block elements:
http://code.google.com/p/html5shiv

**Contact Information**

St. Edward's University Instructional Technology Training
http://think.stedwards.edu/computerhelp/training | training@stedwards.edu

• Watch our screencasts online: http://www.youtube.com/StEdwardsITtraining
• Web design resources: http://www.diigo.com/list/meganura/web-essentials
Register for more free workshops: http://frc.stedwards.edu/workshop
Need time to work on your website with assistance from a Trainer? Come by the training room during Innovation Creation Lounge hours.

Resources

Curated resources for web design and HTML5 success:
http://www.diigo.com/list/meganura/web-essentials

WHATWG recommends the following sites for learning what designers and developers can implement today. All of these sites are written in HTML5:
• The HTML5 spec for developers: http://developers.whatwg.org
• Dive into HTML5: http://diveintohtml5.info
• Is this HTML/CSS feature supported by browsers? http://caniuse.com
• HTML5 Doctor: http://html5doctor.com
• Mozilla: https://developer.mozilla.org/en/HTML/HTML5
• Opera: http://dev.opera.com/web

Books:
HTML5 for Web Designers: http://www.abookapart.com/products/html5-for-web-designers
Introducing HTML 5: http://introducinghtml5.com

Official Sites:
WHATWG: www.whatwg.org
WHATWG details how to keep up with spec changes and editions:
http://wiki.whatwg.org/wiki/FAQ#HTML5
HTML5 Editor's Draft Spec: http://dev.w3.org/html5/spec/Overview.html

Further learning:
Video for Everybody: http://camendesign.com/code/video_for_everybody
   and the VFE generator: http://sandbox.thewikies.com/vfe-generator
HTML5 Element Index (with definitions): http://html5doctor.com/element-index
HTML5 differences from HTML4: http://dev.w3.org/html5/html4-differences
HTML5 Boilerplate: http://html5boilerplate.com
Bruce Lawson's Blog: http://www.brucelawson.co.uk/category/html5
A block of flow content (not inline phrasing content)

Is it a major navigation block?
- Yes: `<nav>`
  - Site or in-page navigation (anything you’d use a “skip to nav” link for)
  - → html5doctor.com/nav
- No: Does it make sense on its own? e.g. in a feed reader
  - Yes: `<article>`
    - News article, weblog or forum post, comment on an article, sidebar widget etc, with a heading...
    - → html5doctor.com/article
  - No: Is it required to understand the current content?
    - Yes: `<figure>`
      - One or more images, graphics, code samples etc, plus optional `<figcaption>`...
      - → html5doctor.com/figure
    - No: Could you move it to an appendix?
      - Yes: `<section>`
        - A section of the page, or chapter of an `<article>`, with a heading...
        - → html5doctor.com/section
      - No: Is it logical to add a heading?
        - Yes: Appropriate element
        - No: Does it have any semantics?
          - Yes: Possibly `<p>`, but possibly `<address>`, `<blockquote>`, `<pre>`...
          - No: `<aside>`

Sidebar, comments section, pullquote, glossary, advertising, footnote etc that’s tangentially related to the page or content...
- → html5doctor.com/aside

Flow content with no additional semantics, e.g. for CSS hooks...
- → html5doctor.com/div

*Sectioning content element*
These four elements (and their headings) are used by HTML5’s outlining algorithm to make the document’s outline
- → html5doctor.com/outline