# Semantic Markup

As you learn about HTML and the Web you may find that you encounter one specific word repeatedly that is often left undefined. That word is *semantic*. You may read statements such as "we went looking for a semantic element" or "We try and be as semantic as we can", yet never get a clear picture of what the word *semantic* means. In this article, we'll explore the world of semantic markup, come up with a working definition of the term, and apply the concept to the way we write HTML markup.

**Contents** [hide]

1 What is Semantic Markup?
- 1.1 Two Practices that Enable Semantic Markup
  - 1.1.1 Using HTML Elements Correctly
  - 1.1.2 Seperating Content and Presentation
- 1.2 Defining Semantic Markup

2 Why is Semantic Markup Important?

3 How Do We Write Semantic Markup?
- 3.1 Document Structure
- 3.2 Textual Meaning
- 3.3 Media Type
- 3.4 Correlation Tags

4 Closing Thoughts

5 Related Elements

## What is Semantic Markup?

According to Dictionary.com, *semantics* refers to the correct interpretation of the meaning of a word or sentence. To use a word semantically is to use it in a way that is properly aligned with the meaning of the word. When we misuse a word we are not using it semantically. Many HTML tags have semantic meaning. That is, the element itself conveys some information about the type of content contained between the opening and closing tags. For example, when a browser encounters an `h1` heading it interprets that tag to mean that the contents of the `h1` element constitute the most important heading of the section that contains the element. The semantic meaning of an `h1` tag is that it is used to identify the most important header of a specific web page or section.

## Two Practices that Enable Semantic Markup

There are two different practices that must be put into place if we are going to write semantic markup.

1. Semantic markup requires that HTML elements be used according to their intended purpose.
2. Semantic markup requires the separation of content and presentation.

### Using HTML Elements Correctly

When writing semantic markup, we use HTML tags to tell browsers something about the contents of the element. In semantic markup, tags are no longer just a way to get content to show up on a web page in a human-readable format. The tags themselves become a way to tell a machine (whether a browser, a computer, a smartphone, or another smart device) something about the meaning of the content. To write semantic markup, we must use HTML tags correctly so that our markup is both human-readable and machine-readable.

### Seperating Content and Presentation

In the past, it was common to use markup to define styles and to control web page layout. Heading levels were selected not based on hierarchy but based on the styles applied by the web browser, tables were used for web page layout rather than to organize tabular data, some HTML tags (such as `frameset`) were created for the express purpose of defining web page layout, and so forth. When we write semantic markup we can no longer select HTML elements based on visual presentation. Instead, we select HTML elements based on their semantic meaning, and then use CSS to define the visual presentation of our content. When writing semantic markup, the presentation of web page elements is kept completely separate and distinct from the markup of the content itself.

### Defining Semantic Markup

With those two practices in mind, we can define semantic markup in this way: *Semantic markup* is the use of a markup language such as HTML to convey information about the meaning of each element in a document through proper selection of markup elements, and to maintain complete separation between the markup and the visual presentation of the elements contained in the document.

## Why is Semantic Markup Important?

Good CSS can make bad markup invisible to the average website visitor. However, no amount of styling will make bad markup more meaningful to a computerized visitor such as a search engine web crawler, browser translation tools, or assistive technologies such as screen readers. According to Bruce Lawson, the semantic use of HTML elements "enhances accessibility, searchability, internationalization, and interoperability." In other words, writing semantic markup is mandatory if you want your website to be accessible to all visitors, to achieve a high search engine ranking, to be available to visitors from around the world, and to interface effectively with other web services. Writing semantic markup is about creating web content that is both human and computer readable. When the web can be read equally well by both humans and computers, it becomes more accessible since computers are better able to analyze its contents, index it, deliver it, and developers are better able to tie different sources of information together into new web services.

## How Do We Write Semantic Markup?

We write semantic markup by selecting and using HTML tags properly, and by selecting tags that convey something about the information marked by the tags. There are elements in HTML that are semantic and elements that are non-semantic. Examples of non-semantic elements are `div` and `span`. These tags don't tell the computer anything about the meaning of the contents of the element. While useful, and fine to use in some cases, if a semantic tag is available and appropriate for a specific use, use it before resorting to a non-semantic tag. Many semantic tags come from the analysis of web page markup completed by companies like Google and Opera. What these companies have found is that many websites use `id` and `class` attributes to hint at the meaning of the contents of non-semantic elements. For example, they found lots of divs that looked like this: `<div id="nav">`, `<div id="header">`, and `<div id="footer">`. Findings like these helped the W3C identify and target new semantic tags to include in HTML5 such as: `nav`, `header`, `footer`, `article`, and `aside`. We can group the most common and important semantic elements into four categories:

- Document structure tags
- Textual meaning tags
- Media type tags
- Correlation tags

## Document Structure

In the past, the `div` element was the main way sections of a website were identified and grouped. However, with the release of HTML5, we have several new tags to work with that provide semantic meaning in addition to the grouping attributes offered by the `div` tag:

- `header` : A container to be used for a web page header which typically contains the site logo, heading elements, and site navigation.
- `footer` : A container to be used for a web page footer which typically contains authorship, contact, and copyright information in addition to navigational links and a link back to the top of the web page.
- `main` : A high-level element used to contain all of the content that is unique to a single web page and not repeated across multiple web pages.
- `nav` : An element to contain blocks of site navigation links. This element is typically placed in the page `header` and `footer` , and may also be used in an `aside` (sidebar) element as well.
- `section` : The `section` element is used to mark off sections of a document, such as chapters or major sections of a long form post.
- `aside` : Use to identify content that is related to the main content on the page but not part of the primary flow of the document. For example, the `aside` element may contain a glossary definition of a term that appears in a blog post or it may contain advertisements related to the contents of the page.
- `article` : The article element is used to identify a block of content suitable for reuse and syndication in other settings, such as a blog post or technical article.

Review our Document Tutorial to learn more about using these semantic tags that add structure to a web page.

## Textual Meaning

In the early days of the web it was common to see markup like this:

```
<style> .italics {     font-style: italic; } </s
tyle> <p>Some paragraph content including one <s
pan class="italics">italicized</span> word.</p>
```

Today we (hopefully) wouldn't dream of doing something like that since the `span` element tells the browser and other computerized visitors absolutely nothing about the meaning or purpose of the text nested in the between the opening and closing tags. Rather than use the non-semantic `span` tag, we'd add `em` tags around the words that should appear in italics. By using `em` tags, visitors using screen readers or other computers accessing the content would understand that the tags were applied to add emphasis to the tagged content. The `em` element is just one example of how HTML tags add semantic meaning to textual content. Other examples include:

- `h1`, `h2`, `h3`, `h4`, `h5`, and `h6`: Heading element tags are used to identify text that should appear as a heading. The highest level, or most important, heading is `h1` which is followed by heading levels `h2` through `h6` in order of descending importance.
- `strong`: Text that is marked with `strong` tags is given added importance and is usually displayed in a **bold** typeface.
- `mark`: The `mark` tag is used to highlight text of specific importance in a specific context. For example, it can be used to highlight every occurrence of a search term in a search results page.
- `cite`: The `cite` element is used to identify the original work from which a bit of content originates.
- `blockquote` and `q`: The `blockquote` and `q` (quote) elements are used to identify text that is a direct quotation

from another source.

- `time` : The `time` element can be used to tell browsers, web crawlers, and other smart devices that a specific bit of content represents time on a 24-hour clock or a specific calendar date.

Our Fonts and Web Typography Tutorial provides a great deal more detail surrounding the proper use of these tags to assign semantic meaning to textual content.

## Media Type

HTML5 also includes three tags that identify the type of media served up between the tags. These tags serve a dual purpose. First, they signal to the browser the need to queue up a specific technical resource such as a video playback engine. Second, they assign semantic meaning to the content.

- `audio` : Used to add one or more sources of audio content to a document and to allow the browser to pick the best option based on the visitor's device and browser.
- `video` : Similar to the `audio` element but used to add video content to a markup document.
- `picture` : The picture element is used to allow a web browser to pick the best image from the available options based on the results of a media query.

You can learn more about embedding `audio` and `video` elements in our HTML5 Media Tutorial. In addition, our article on the use of images on the web provides additional information on when to use the `picture` element and when to stick with the `img` element.

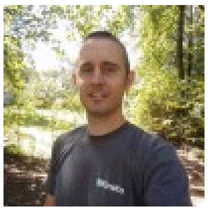
## Correlation Tags

Several HTML elements are used to signal a correlation between multiple elements. For example, the use of an ordered list ( `ol` ) tells the browser that the items on the list are related to eachother and need to appear in a specific order. Other elements that are used to signal correlation between multiple elements include:

- `ul` : Unordered lists are used to signal a relationship between the items on the list and to indicate that they do not need to be understood in a specific order. Read our Lists Tutorial to learn more about how to use both ordered and unordered lists.
- `figure` : The `figure` element is used to group together a piece of content, such as an image, chart, graph, or text, and a caption marked off by `figcaption` tags. By nesting the caption and the content between `figure` tags a relationship between the nested elements is identified. Our images page contains more information about implementing this helpful tag.
- `address` : This attribute is used to associate contact information with the parent element that contains the `address` element. For example, when added to an `article` , the `address` element provides contact information for the article author, and when added to a web page `footer` the `address` identifies contact information for the web page owner.

## Closing Thoughts

If you're new to HTML take the time to learn how to use all of these different HTML tags semantically. If you aren't sure that you're using the right tag, take a few minutes and do some research. As we've seen, using the right tag is important. If you've been working with HTML for a while now, take the time to learn about the new HTML5 elements and how to properly use them. HTML has grown increasingly complex over the last several years, and it can be tempting to keep using `div` elements with `class` and `id` attributes, but the accessibility and interoperability promise of semantic HTML5 tags is reason enough to embrace these new semantic elements. As Internet access becomes more widespread, smart devices proliferate, and the web further integrates into the fabric of modern society, the need for markup to be semantically accurate becomes increasingly evident. No longer is web page content isolated to desktop computers and accessed with just a few web browsers. Today, the semantic web is growing up all around us. By ensuring that every bit of markup you touch is semantic, you play a part in enabling the ongoing growth of the increasingly interconnected web.



Jon Penland

Jon is a freelance writer, travel enthusiast, husband and father. He writes about web technologies such as WordPress, HTML, and CSS.

## Related Elements

| Element Name | Attributes | Notes |
|---|---|---|
| section | | The <section> element is a structural HTML element used to group together |

| | | |
|---|---|---|
| | | related elements. Each <section> typically includes one or more heading elements and additional elements presenting related content. |
| progress | | The <progress> element is used to create a progress bar to serve as a visual demonstration of progress towards the completion of task or goal. The max and value attributes are used to define how much progress (value) has been made |

| | | |
|---|---|---|
| | | towards task completion (max). |
| output | | The <output> element is used to display the result of a calculation. The <output> element is typically used in conjunction with a parent <form> and sibling <input> elements to perform a calculation. The actual calculation is typically completed using JavaScript. |
| menuitem | | The <menuitem> element is used to add menu items and commands to |

| | | |
|---|---|---|
| | | contextual pop-up menus (the menus that appear when you right-click in a web browser). |
| main | | The <main> element is used to denote the content of a webpage that relates to the central topic of that page or application. It should include content that is unique to that page and should not include content that is duplicated across multiple webpages, such as headers, footers, and primary |

| | | |
|---|---|---|
| | | navigation elements. |
| aside | | The <aside> element is used to identify content that is related to the primary content of the webpage, but does not constitute the primary content of the page. Author information, related links, related content, and advertisements are exampes of content that may be found in an aside element. |
| article | | The <article> element identifies a self-contained piece of |

| | | |
|---|---|---|
| | | content which could theoretically be distributed to other websites and platforms as a stand-alone unit. The <article> element is a good choice to contain entire blog posts, news articles, and similar content. |
| acronym | | The <acronym> element and title attribute was used to associate a full-text explanation with an acronym. The <acronym> element has been deprecated in HTML5 and <abbr> |

| | | |
|---|---|---|
| | | should be used instead. |
| abbr | title | The <abbr> element is used along with a title attribute to associate a full-text explanation with an abbreviation or acronym. Website visitors do not see the text in the title attribute, but browsers, search engines, and assistive technologies do use this information. |
| menu | | The <menu> element defines an instance of a menu. This experimental HTML feature has very |

| | | |
|---|---|---|
| | | limited browser support, but may soon be an effective way to add menu items to context menus and to create interactive web application menus. |
| <strong> HTML Tag | | The <strong> element is used to identify text that is of greater importance than the surrounding text. By default, all browsers render <strong> text in a bold typeface. |
| address | | The <address> element |

| | | |
|---|---|---|
| | | identifies contact information relevant to the current site, page, document, section, or article. It should not be used to identify addresses in any other context. |
| dfn | | The &lt;dfn&gt; element is used to identify the defining instance of a term in an HTML document. When a term is wrapped in &lt;dfn&gt; tags, browsers and web crawlers will understand that nearby text contains |

| | | |
|---|---|---|
| | | a definition of the term. |
| headlines | align | The <h1>, <h2>, <h3>, <h4>, <h5>, and <h6> elements are used to create headings in descending order of importance where <h1> is the most important and <h6> the least. |
| ins | | The <ins> element is used to identify text that has been inserted into a document. It is often paired with a <del> element which identifies deleted text replaced by the text |

| | | |
|---|---|---|
| | | contained in the <ins> element. |