**Semantic code: What? Why? How?**

Semantic code returns to this original concept and encourages web designers to write code that describes the content rather than how that content should look.

* Semantic code is more straightforward for people to understand too, so if a new web designer picks up the code, they can learn it much faster.
* semantic code aids accessibility
* Search engines need to understand what your content is about to rank you properly on search engines. Therefore, semantic code tends to improve your placement on search engines, as it is easier for them to understand.
* the design is held separately from your content, semantic code allows anybody to add or edit pages without having to have an astute eye for design. You describe the content, and the cascading style sheet defines what that content looks like.
* Because semantic code does not contain design elements, it is possible to change the look and feel of your site without recoding all of the HTML.

A <div> is a block-level element that is commonly used to identify large groupings of content, and which helps to build a web page’s layout and design. A <span>, on the other hand, is an inline-level element commonly used to identify smaller groupings of text within a block-level element.

We’ll commonly see <div>s and <span>s with class or id attributes for styling purposes. Choosing a class or id attribute value, or name, requires a bit of care. We want to choose a value that refers to the content of an element, not necessarily the appearance of an element.

#### Deciding Between <article>, <section>, or <div> Elements

At times it becomes fairly difficult to decide which element—<article>, <section>, or <div>—is the best element for the job based on its semantic meaning. The trick here, as with every semantic decision, is to look at the content.

Both the <article> and <section> elements contribute to a document’s structure and help to outline a document. If the content is being grouped solely for styling purposes and doesn’t provide value to the outline of a document, use the <div> element.

If the content adds to the document outline and it can be independently redistributed or syndicated, use the <article> element.

If the content adds to the document outline and represents a thematic group of content, use the <section> element.

## div or section or article?

So which should you use and when?

If the content within the element is not **semantically related**, then use a **<div>**. If the semantically related content is also able to be **self-contained**, then use an **<article>**. Otherwise, use a **<section>**.

Actually, you are quite right when it comes to header/footer. Here is some basic information on how each of the major HTML5 tags can/should be used (I suggest reading the full source linked at the bottom):

**section** – Used for grouping together thematically-related content. Sounds like a div element, but it’s not. The div has no semantic meaning. Before replacing all your div’s with section elements, always ask yourself: “Is all of the content related?”

**aside** – Used for tangentially related content. Just because some content appears to the left or right of the main content isn’t enough reason to use the aside element. Ask yourself if the content within the aside can be removed without reducing the meaning of the main content. Pullquotes are an example of tangentially related content.

**header** – There is a crucial difference between the header element and the general accepted usage of header (or masthead). There’s usually only one header or ‘masthead’ in a page. In HTML5 you can have as many as you want. The spec defines it as “a group of introductory or navigational aids”. You can use a header in any section on your site. In fact, you probably should use a header within most of your sections. The spec describes the section element as “a thematic grouping of content, typically with a heading.”

**nav** – Intended for major navigation information. A group of links grouped together isn’t enough reason to use the nav element. Site-wide navigation, on the other hand belongs in a nav element.

**footer** – Sounds like its a description of the position, but its not. Footer elements contain informations about its containing element: who wrote it, copyright, links to related content, etc. Whereas we usually have one footer for an entire document, HTML5 allows us to also have footer within sections.

**Source**: <https://clzd.me/html5-section-aside-header-nav-footer-elements-not-as-obvious-as-they-sound/>

Additionally, here's a description on article, not found in the source above:

**article** – Used for element that specifies independent, self-contained content. An article should make sense on its own. Before replacing all your div’s with article elements, always ask yourself: “Is it possible to read it independently from the rest of the web site?”