

Week 10

Semantic Markup and Page Layout

There's no question, HTML5 is a hot topic for developers.

There's a great story about a university who, when building their campus, didn't create any walking paths. They just planted grass and waited.

A year later, the grass was all worn out where people walked most frequently. So that's where the university paved the actual sidewalks.

It makes perfect sense! The sidewalks were exactly where people actually walked.

The HTML5 new semantic elements were based on that exact same logic (see the [W3C design guidance to "Pave the Cowpaths"](#)).

Semantic elements describe their meaning or purpose clearly to the browser and to the developer. Contrast that with (for example) the <div> tag. The <div> tag defines a division or a section in an HTML document, but it doesn't tell us anything about its content or convey any clear meaning.

<div>

Developers commonly use IDs and/or class names with these `<div>` tags. This conveys more meaning to the developers, but unfortunately, it doesn't help browsers derive the purpose of that markup.

`<div id="header">`

In HTML5, there are new semantically rich elements that can convey the purpose of the element to both developers and browsers.

`<header>`

The W3C mined billions of existing webpages to discover the IDs and class names that developers were already using. Once they threw out `div1`, `div2`, etc., they came up with a list of rich descriptive elements that were already being used, and made those the standards.

Here are a few of the new semantic elements in HTML5:

- `article`
- `aside`
- `figcaption`
- `figure`
- `footer`
- `header`
- `hgroup`
- `mark`
- `nav`
- `section`
- `time`

Because of the semantic richness, you can probably guess what most of these elements do. But just in case, here's a visualisation:



Headers and **footers** are self-explanatory and **nav** creates a navigation or menu bar. You can use **sections** and **articles** to group your content. Finally, the **aside** element can be used for secondary content, for example, as a sidebar of related links.

Here is a simple example of some code that uses these elements.

HTML Page Structure

Below is a visualization of an HTML page structure:

```
<html>
```

```
<head>
```

```
<title>Page title</title>
```

```
</head>
```

```
<body>
```

```
<h1>This is a heading</h1>
```

```
<p>This is a paragraph.</p>
```

```
<p>This is another paragraph.</p>
```

```
</body>
```

```
</html>
```

HTML Events

Keyboard Events

Attribute	Value	Description
<u>onkeydown</u>	<i>script</i>	Fires when a user is pressing a key
<u>onkeypress</u>	<i>script</i>	Fires when a user presses a key
<u>onkeyup</u>	<i>script</i>	Fires when a user releases a key

Mouse Events

Attribute	Value	Description
<u>onclick</u>	<i>script</i>	Fires on a mouse click on the element

<u>ondblclick</u>	<i>script</i>	Fires on a mouse double-click on the element
-------------------	---------------	--

<u>onmousedown</u>	<i>script</i>	Fires when a mouse button is pressed down on an element
--------------------	---------------	---

<u>onmousemove</u>	<i>script</i>	Fires when the mouse pointer is moving while it is over an element
--------------------	---------------	--

<u>onmouseout</u>	<i>script</i>	Fires when the mouse pointer moves out of an element
-------------------	---------------	--

<u>onmouseover</u>	<i>script</i>	Fires when the mouse pointer moves over an element
--------------------	---------------	--

<u>onmouseup</u>	<i>script</i>	Fires when a mouse button is released over an element
------------------	---------------	---