**Basic CSS Syntax: Contextual Selectors**

Contextual selectors allow you to define the appearance for HTML tags in parent-child relationships with other HTML tags. In the following example, I have created two instances of the **B** (bold) tag: one by itself, and one where it appears as a child object of a **P** (paragraph) tag.

Example:

<b>Here is some bold text.</b>

<p>Here is a paragraph with a <b>bold</b> word in it.</p>

Now, I could make one CSS declaration which would define all instances of the **B** tag uniformly, as I have done in the past:

b { font-weight:bold; color:#000000; }

But what if I want the **B** tag when it appears as a child of the **P** tag to look different than the **B** tag by itself? The appearance of these **B** tags can be defined separately using a contextual selector in a CSS declaration.

A contextual selector for a declaration states the parent selector first, then a space, then the child selector, followed by a space and the curly-braces containing the properties.

Generic Example:

parenttagname childtagname { property:value; etc... }

Actual Example (defining the **B** tag when it is a child of the **P** tag):

p b { color:#660066; font-weight:bold; }

Using contextual selectors, I can define how an HTML tag will look when it is IN CONTEXT of another specific HTML tag (in other words, when it is a child of another tag). For instance, I can define the **B**tag as a child of the **P** tag separately from the **B** tag as a child of the **TH** tag; I can define the tag appearance based on its CONTEXT, hence the term CONTEXTUAL selector.

I can define properties of the **B** tag generally, and I can ALSO more specifically redefine some or all of those properties when the **B** tag appears as a child of the **P** tag.

Example:

b { font-weight:bold; color:#000000; }

p b { color:#660066; }

In the first CSS declaration above, I have told the browser to make the **B** tag appear bold and black. In the second declaration, I have used a contextual selector to tell the browser to make the **B** tag (when it appears as a child of the **P** tag) appear dark violet. Because I have NOT specified the font-weight property AGAIN within the second CSS declaration above, the **B** tag as child of the **P** tag will INHERIT the font-weight (boldness) property from the general **B** tag declaration.

**Remember**: you can define the main properties for a tag in a general declaration (like the one for the **B** tag above), then override one or more of the properties defined for the tag in different CONTEXTS (situations) using contextual selectors in CSS declarations, as you can also see above.

When creating contextual selectors, you MUST separate the different selectors with spaces. The following example is WRONG:

pb { color:#660066; }

I have NOT used a contextual selector in the above example; Instead, I have made a CSS declaration for some non-existent tag called **"pb"**, which won't do us any good.

When using contextual selectors, I can define contexts for a given tag with as much "granularity" as I like. In other words, I can specify very extensive ancestral relationships using multiple contextual selectors in a CSS declaration, each selector separated by a space.

Example:

b { color:#000000; }

p b { color:#660066; }

table tr td b { color:#006600; }

table tr td p b { color:#000066; }

In the above example, I have defined the **B** tag:

1. by itself as black
2. in context of the P tag as dark violet
3. in context of a TD cell in a TABLE as dark green
4. in context of a P within a TD cell in a TABLE as dark blue

Example:

table tr td { color:#000000; }

table tr th { color:#660066; }

table tr td table tr td { color:#006600; }

table tr th table tr td { color:#000066; }

In the above example, I have defined the color for text in TD and TH cells within a TABLE (black text for TD, dark violet text for TH). I have also defined the color for text in TD cells for a TABLE nested within a TD tag in another TABLE, contrasted with the next declaration, where I have defined the color for text in TD cells for a TABLE nested within a TH tag of another TABLE; so, I am differentiating between TD cells of tables nested within TD tags, versus TD cells of tables nested within TH tags. As you can see, I can become extremely specific about differences in tag appearance in different contexts, using CSS declarations and contextual selectors.

You may or may not ever need to access this level of granularity, but it's important to know how to do so, if you should ever need it.

I would also use this sort of contextual selector when defining list and sub-list appearance:

ul li { list-style-type:disc; }

ul ul li { list-style-type:square; }

In the above example, I have defined the **LI** tag for an unordered list to use the disc-type bullet, while defining the **LI** tag for a nested unordered list (or sub-list) to use the square-type bullet.

Contextual selectors provide a very powerful mechanism for differentiating tag appearance based on the context of the tag. There is also another way to differentiate tags, using something called a CLASS; we'll discuss classes in a later section.