Styling XML with CSS

Author: Mihai Rotaru

Date: 1 Dec 2011

XML can be styled using CSS in a similar way to (X)HTML; in fact, the same CSS file could be used by both XML and XHTML files. In this post, I will demonstrate, using the XML file (Pastebin link) created in the previous post, how to add CSS styling to an XML document. This blog post is based on the short tutorial in section 2.2 of the CSS2.1 W3C Recommendation (http://www.w3.org/TR/CSS2/intro.html#xml-tutorial).

Before telling the XML file about the CSS, obviously the CSS file needs to be created. I wanted to display the movies described in the XML file as a table, with a gray background, a serif font, and green background for the table headers. I ended up creating this short CSS file:

```
body
{
font: 12px Georgia;
}

table
{
border-width: 1px;
border-style: outset;
background-color: lightgray;
}

th
{
background-color: #9acd32;
}
```

Then, I simply added a line to the XML file:

```
<?xml-stylesheet type="text/css" href="style.css"?>
```

Not a very smart approach, as I soon realized. Instead of rendering a lovely table, the browser simply printed all the information about the movies, in a single row. Which makes perfect sense; the CSS states that table headers should have a green background. But how is the browser supposed to know which part of the XML file should be the headers ? Or where is the table ? All it sees is a well-formed XML file.

After massaging the CSS file further - while not touching the XML, in keeping with the 'separate data from style' principle - I managed to get to something resembling a table:

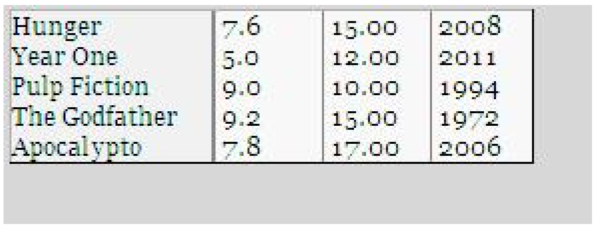
```
movies
{
font: 12px Georgia;
margin: 3px;
width: 260px;
border-width: 1px;
border-style: outset;
background-color: lightgray;
}

title { background-color: lightcyan; }

movie { display: block; background-color: honeydew }

title { display: inline-block; width: 100px; border-right: 2px solid gray; }
rating { display: inline-block; width: 50px; border-right: 1px solid gray; }
price { display: inline-block; width: 50px; border-right: 1px solid gray; }
year { display: inline-block; width: 40px; }
```

How the XML was rendered by Firefox 8.0:



Hunger	7.6	15.00	2008
Year One	5.0	12.00	2011
Pulp Fiction	9.0	10.00	1994
The Godfather	9.2	15.00	1972
Apocalypto	7.8	17.00	2006

Looks like a table, sure, but my dreams of green table header backgrounds were shattered; it cannot be accomplished without modifying the XML file. And I really don't want to do that - because then I would be using the XML file as (X)HTML. Obviously, XHTML is XML by definition; but it's not wrong - in fact, it is required - for XHTML to have tags like 'body', 'table', 'td', 'tr', etc.

XML, on the other hand, I intend to use it for it's intended purpose - storing data in a future-proof way. What if I later decide to write another parser that would use my XML files with data about movies ? Had I decided to change the XML to accommodate CSS styling, my parser would have to look for and parse elements such as 'body', 'h1', etc - and I would much rather have to deal with 'movie' elements, which have the 'title', 'year', etc elements. In other words, the semantics of the XML document would be preserved, without being contaminated with presentational tags - or simply tags that don't make sense in the context of the data which is stored.

I think this highlights very well the problems with styling XML with CSS - while it can work, the desired effects might be difficult to achieve without contaminating the XML document with additional data. This is not desirable; the presentational document should accommodate the data, not the other way around. While attempting this, the CSS file might end up bloated, and using unintuitive work-arounds.

Other drawbacks of CSS stem from it's origin as purely a styling language - it cannot perform any kind of computations. If I wanted my table to list the price with the VAT included, it would not be possible. It is also impossible to sort, or otherwise change the order in which elements appear.

However, styling XML documents with CSS might work very well for simpler document types, which do not require the control that my example did. For example, if the document only contains a few textual elements, then basic presentational properties can be set easily - font, colors, etc.

But probably the most enticing feature of CSS over the alternative (XSL) is the fact that it leverages knowledge of a widespread, almost ubiquitous technology, since CSS has been used extensively for decades. It has a very large user base, and these programmers would be able to style XML documents using CSS with little learning required. XSL, on the other hand, can be quite daunting to learn.

Ultimately, I think a decision should be taken on a per-case basis. In simple cases, CSS works just fine; but when XML documents have a more complex structure, the power of XSL should probably be invoked instead.