On the use of TEX as an authoring language for HTML5

S.K.Venkatesan

TNQ Books and Journals
Chennai

Abstract

The TEX syntax has been fairly successful at marking-up a variety of scientific and technical literature, making it an ideal authoring syntax. The brevity of the TEX syntax makes it difficult to create overlapping structures, which in the case of HTML has made life so difficult for XML purists. We discuss S-expressions, the TEX syntax and how it can help reduce the nightmare that the HTML5 markup is going to create. Apart from this we implement a new syntax for marking-up semantic information (microdata) in TEX.

Introduction

The brevity of TEX syntax has made it fairly successful at marking-up a variety of scientific and technical literature. However, modern markup languages such as (X)HTML and XML have verbose syntax which are not only difficult to author but also produce non-tree like structures such as overlapping structures that needs to be checked for well-formedness. However, TEX and its macros are difficult to parse and validate like XML with DTD or Schema. Many XML versions of TEX have been proposed such as TEXML [1] and XLATEX [1] that is intrinsically close to TEX/LATEX. The main advantage with such a system is that one can introduce validator using DTD or Schema that can check the syntax before passing it to the TeX engine. However, XML syntax is difficult to author and in fact is prone to producing overlapping structures that needs to be avoided in order for it to be well-formed, and as a result these XML versions have not become popular.

In this article, we propose something that is quite the reverse, i.e., TeX as a an authoring syntax for both XML and HTML.

TeX, S-expressions and XML

Let us look the following TEX code:

```
\title[lang=en]{Title of a \textit{plain} article}
```

The same code in LISP like S-expression would be:

```
(title (@lang=”en”) (”Title of a ”) (italic ”plain”) (”article”))
```

The S-expression has the nicety that elements and attributes are treated on par and in fact is an improvement on XML as it allows further nesting within attributes. The corresponding XML code will be:

```
<title lang=”en”>Title of a <italic>plain</italic> article</title>
```

In both TeX and XML syntax further nesting of structures is not possible within attributes, which make TeX ideal for authoring XML or HTML5.

Overlapping markup in HTML

Since HTML is marked-up by humans, there tends to many situations with overlapping elements and other eccentric markups that do not confirm to a well-formed SGML or XML syntax. Consider the HTML markup:

```
<p>A piece of text with <i>unique <b>and</i> strong
  formatting</b> issues</p>
```

An utility like HTMLTidy [1] or TagSoup [2] can convert it into well formed markup such as:

```
<p>A piece of text with <i>unique </i><b><i>and</i>
    <b> strong formatting</b> issues</p>
```

However it is not clear what should be done with such a non-standard markup. HTML5 specification also defines clearly on how such a non-standard markup should be interpreted [4] but the HTML implementations in browsers currently deal with differently from each other.

W3C has been insisting for some time that the next generation of markup should be XML compliant like XHTML+MathML+SVG profiles, with other intricacies such as namespaces. However, more than 99% of HTML pages in the wild have been invalid according to the HTML4 DTD or Schema. This being the case, W3C gave up on the idea of an XML solution and moved on to HTML5 with added elements and features, such as MathML, SVG and video, audio and addtional microdata formats.

With the experience in HTML4 it can be safely predicted that more features one adds to HTML more the scope for non-standard markup with overlaps and entanglement that can create a great deal of difficulty for different browsers and users.

We will consider here, e.g., Microsoft’s own interpretation of MathML in HTML5. Microsoft has been pushing for certain agenda in MathML3 (although much of which has not been accepted by the MathML committee). Based on their own experience with OML, a subset OOXML markup, they would like to add formatting features in

MathML such as bold, italic and paragraph elements inside MathML. Consider the following markup:

```
<math><b><mi>r</mi></b>=<mfenced><mi>x</mi>
  <mi>y</mi></mfenced></math>
```

which could in pure MathML coding would be:

```
<math><mi mathvariant=”bold-italic”>r</mi></b>=<mfenced>
    <mi>x</mi><mi>y</mi></mfenced></math>
```

Mixing elements from different namespaces is one of the side effects one can expect in HTML5. It is not clear if MathML elements could be included within SVG elements and vice versa. One can expect such new non-standard markups to be created that will be quite difficult for browsers to handle.

New elements such as `<section>` have been introduced, so one can expect more confusion:

```
<section>
  <h2>Section title</h2>
  <section>
    <h1>Another section title</h1>
  </section>
</section>
```

It is not clear from the above markup whether `<h1>` is supposed to have `<h2>` some meaning?

In this article we do not want to convey the impression that everything about HTML5 is wild west, rather, it is a rich arena that needs to be authored carefully, because there are many pit falls. However, HTML5 introduces new features like MathML, SVG, Video and audio features that are quite essential for further enrichment of basic content [3].

TEX as input format for HTML5

In this section we would like introduce LATEX environment for authoring HTML5. Many of these features have been introduced before, say, e.g., in XLATEX and other concepts.

Main structural elements of the document

HTML5 has introduced new content elements that brings it closer to article class. We propose the following TEX macros.

No.	HTML	LaTeX	Description

1	`<article>#1</article>`	`\begin{article}`
		`{#1}`
		`\end{article}`
			Document

			Section Heading
2	`<h1>#1</h1>`	`\Ha{#1}`	– Level One
3	`<h2>#1</h2>`	`\Hb{#1}`	– Level Two
4	`<h3>#1</h3>`	`\Hc{#1}`	– Level Three
4	`<h4>#1</h4>`	`\Hd{#1}`	– Level Four
5	`<p>#1</p>`	`\p{#1}`	Paragraph
6	`<span>#1</span>`	`\s{#1}`	Text Span

Simple formatting elements

We propose the following TeX macros for HTML formatting elements:

No.	HTML	LaTeX	Description
1	`<b>#1</b>`	`\b{#1}`	bold
2	`<i>#1</i>`	`\i{#1}`	italic
3	`<b><i>#1</i></b>`	`\bi{#1}`	bold-italic
4	`<tt>#1</tt>`	`\m{#1}`	monospace
5	`<sup>#1</sup>`	`\sp{#1}`	superscript
6	`<sub>#1</sub>`	`\sb{#1}`	subscript

MathML elements

We propose the following TeX macros for MathML formatting elements:

No.	MathML	LaTeX	Description
1	`<mrow>#1</mrow>`	`{#1}`	grouping
2	`<mi>#1</mi>`	`{#1}`	variables
3	`<mo>#1</mo>`	`{#1}`	operators
4	`<mn>#1</mn>`	`{#1}`	numbers
5	`<mtext>#1</mtext>`	`\mbox{#1}`	monospace
6	`<mfrac>#1#2</mfrac>`	`\frac{#1}{#2}`	fraction
7	`<msup>#1#2</msup>`	`\{#1}^{#2}`	superscript
8	`<msub>#1#2</msub>`	`\{#1}_{#2}`	subscript
9	`<mover>#1#2</mover>`	`\{#1}^{#2}`	over
10	`<munder>#1#2</munder>`	`\{#1}_{#2}`	under

SVG elements

We propose the following TeX macros for SVG formatting elements:

No.	SVG	LaTeX	Description

1	`<circle cx=”#1” cy=”#2” r=”#3”`
	`style=”stroke:#4;stroke-width:#5;`
	`fill:#6;”/>`
		`\circle[x=#1,y=#2,r=#3`
		`s=#4,w=#5,f=#6]`
			Circle

2	`<ellipse cx=”#1” cy=”#2” rx=”#3”`
	`ry=”#4” style=”stroke:#5;`
	`ry=”#4” stroke-width:#6;fill:#7;”/>`
		`\ellipse[x=#1,y=#2,rx=#3,ry=#4`
		`s=#5,w=#6,f=#7]`
			Ellipse

3	`<rect x=”#1” y=”#2” width=”#3”`
	`height=”#4” style=”stroke:#5;`
	`stroke-width:#6;fill:#7;”/>`
		`\rect[x=#1,y=#2,w=#3,h=#4`
		`s=#5,w=#6,f=#7]`
			Rectangle

These elements can be implemented using LaTeX graphics packages such as `TikZ` [4].

Microdata attributes

Since microdata (semantic) attributes can be added to any of the basic HTML elements we need to be able to add attributes to any of the HTML5 TeX macros.

No.	Microdata	LaTeX	Description
1	itemscope	`\s[is=on]`	Top element that indicates descendants are in scope
2	itemtype	`\s[it=http://data-vocabulary.org/Person]`	Property URL
3	itemid	`\s[iid=p0312]`	Unique ID of the person
4	itemprop	`\s[ip=name]`	Indicates as the name of the person
5	itemref	`\s[ir=http://www.ctan.org/pub/another-article]`	Refeerence URL

References

[1] Dave Raggett, HTML Tidy, http://tidy.sourceforge.net/

[2] John Cowan, TagSoup: A SAX parser in Java for nasty, ugly HTML, http://home.ccil.org/c̃owan/tagsoup/tagsoup.pdf.

[3] Mark Pilgrim, HTML5: Up and Running, Dive into the Future of Web Development, O'Reilly Media, 2010.

[4] Andrew Mertz and William Slough, Graphics with TikZ, The PracTEX Journal, 2007, No. 1