Against the Balkanization of the Web

Against the Balkanization of the Web

by Jeffery Triggs
Oxford English Dictionary

December, 1995

T he World Wide Web is developing so fast that it is difficult to say anything about it likely to stay ‘‘current’’ for long. I am aiming, therefore, to take at least a medium if not a long view of a particular aspect, the question of whether or not the Hypertext Markup Language (HTML) continues to be the best alternative in creating Web documents, and to remain at least somewhat ‘‘theoretical’’ in my approach.

We are witnessing what I consider a disturbing trend toward bypassing the Web’s native HTML in favor of socalled portable document formats that promise prospective homepagers easier page creation and greater control over the final ‘‘look and feel’’ of documents on the Web. I believe that this is at best shortsighted or at worst a subversive effort on the part of certain commercial enterprises to undermine the essential ‘‘spirit’’ of the Web, and that it threatens us with a Balkanization sadly reminiscent of the early days of electronic text.

Perhaps we need to remind ourselves from time to time what these bad old days were like and what progress has been made since then. In terms of ‘‘mortal years’’, it was not all that long ago that the exchange of even a simple electronic text was not at all a trivial affair. One had to take into account devicedependent operating systems that were not on speaking terms, what one might call reclusive character sets (does anyone still have some EBCDIC lying around?), and of course a whole Babel of mutually exclusive typesetting and word processing codes. If you had a text in Word or Word Perfect, it was by no means a given that even the person down the hall could read it electronically. Of course, in many cases it did not matter, as the electronic text was merely a means to an end - namely a paper copy. Publishers routinely threw away their typesetting tapes once a text had been *printed*.

Many things have happened since then to improve the situation, particularly the concepts embodied in two sets of initials: ASCII and SGML. (If we’re honest, we’ll admit that, as with so many other things, another little initialism, UNIX, pointed the way.) ASCII (the American Standard Code for Information Interchange) provided a character set that was both hardware and software independent. SGML (the Standard

Generalized Markup Language) provided a way to describe the structure and format of an electronic text so that even the most complex documents could be similarly independent - and *exchangeable in electronic form*.

It is worth pausing to make a few points about the nature of SGML. SGML by itself does not *do* anything! It is neither a programming language nor a typesetting language. It simply *describes* a text in ways that make it easy to *process* with other programs. Most people who are impatient with, frustrated by, or ultimately disappointed in SGML fail to understand this. There is nothing magical about SGML in itself, but without it, the exchange and processing of electronic text is made much more difficult. SGML, as I have commented elsewhere, should not be considered a ‘‘used form’’ at all, but it is the ideal ‘‘stored form’’ for electronic text. It offers such text the greatest number of possible *uses*, but all *in potentia* as it were. It is perhaps because of this that SGML was ignored for so long by commercial word processing and desktop publishing software providers. These, of course, are traditionally print and eventdriven, and SGML, because it required an overhead of software development and, perhaps more important, because it was little known to the general public, was not considered profitable. As recently as last Fall, a major computer magazine in an article on desktop publishing made passing mention of SGML and incorrectly identified it as the ‘‘Standard *Graphical* Markup Language’’. The sudden surge of interest in the Web, of course, with its smell of profitability, has changed all this.

Like ASCII and SGML, the Web represents another simple standardizing concept with enormous practical implications. Like them, it was originally devised with the goal of the widespread and free *exchange* of information across platforms and, in this case, across the internet. The original users were scientists and other researchers with a need to share information over wide area networks. The Web’s revolutionary, liberating effect rests on the *separation* of three functions traditionally combined in textmanagement or database applications: document storage, document delivery, and document display. The Hypertext Transfer Protocol (http) is the enabling factor in all this, but it is the standardization around HTML, allowing the development of the many different ‘‘browsers’’ supporting it, that has led to the Web’s explosive acceptance and growth.

There has been considerable confusion about the relation of HTML and SGML. HTML defines itself as a subset of SGML. As such it makes use of a relatively small group of SGML tags, and SGML purists are quick to point out that the subset of meaningful tags in HTML is considerably impoverished. There is an important difference to consider, however. Whereas SGML *describes* a text with a userdefinable tagset but relies upon otherwise unspecified processing for display, the HTML tagset has a standard, defined set of display features, which all Web browsers are expected, in one way or another, to support. It is, in effect, a general, crossplatform screensetting language, and unlike SGML ‘‘in itself’’ *does* something very powerful. Because HTML browsers recognize any SGML tag as a ‘‘tag’’, the two ‘‘markup languages’’ are not at all incompatible. One might almost consider HTML to be a powerful form of applied SGML, driving the latter’s widespread acceptance for the first time.

Like UNIX itself, both SGML and HTML suffer unfairly from a perceived ‘‘difficulty’’ among many users whose experience is limited to the coddling of word processors and other commercial software packages. The best HTML editor is almost certainly a powerful text editor like vi or emacs (yes - we need to say these things), yet the past year has seen the appearance of a spate of ‘‘WYSIWYG HTML editors’’ and programs promising easy conversion of various word processor texts to HTML. But these are actually the positive signs of HTML’s general acceptance. For the first time, the big commercial software companies are paying attention to some form of SGML. There are other signs, however, suggesting that some users, not completely converted to the idea of SGML/HTML in the first place, may be ready to circumvent it.

The November 1995 issue of *Publish*, for instance, includes the following as part of a review of what it terms ‘‘Document Exchange Software’’:

> If you really want your Web documents to look like the *print original*, HTML conversion probably isn’t the way to go. You can bypass HTML completely with electronic document exchange software, such as Adobe Acrobat, Common Ground, of Novell’s Envoy. Initially developed for the ‘‘paperless office,’’ these programs retain the page layout document’s design integrity - layout, color, images, and fonts - as they convert them into their own formats for crossplatform distribution. This means users can read, edit, and print files created in any application on any platform *provided they have the document exchange software’s viewer*. (80, italics mine)

The ‘‘downside’’ of this is that the portable document formats lock the user into a single, proprietary display system. If you do not download Adobe Acroread (six megabytes on a UNIX system), there is no way you can see the eight pages daily of New York Times (NYTimesFax). If you do, there is no way you can follow links from them to earlier issues or to the outside, for the document is as hermetically inert as any bitmap of a text. It is not even possible to save and search all or part of the text (one cannot help wondering if this is intentional, a result of ‘‘publisher’s paranoia’’), for the text is really a mess of control characters, in effect a mere picture. In fact, all you *can* do is look at it on screen or print it off as ‘‘hard copy’’. A mere click or two away, any HTML online newspaper such as the Electronic Telegraph, the San Francisco Chronicle, or even the Times’s own stepchild, the wonderful online Boston Globe, quickly shows the difference and the real inferiority of the frozen ‘‘pdf’’ format.

There is perhaps a much more important objection to be made here. The pdf approach repeats and indeed strengthens one of the major fallacies about electronic text: that it should try to duplicate the ‘‘design integrity’’ of some sort of ‘‘print original’’, that its used and stored forms should be as much as possible identical. The SGML/HTML view has always been that a true electronic text should be not only machine displayable, but machine processable. According to this view, the electronic text does not gain its authority from any instance of the text in print, but is valid in itself. The electronic text stands in a sort of Platonic relation to any print or screenbased instantiations. It is this powerfully revolutionary idea that underlies the Web, giving it the freedom, essential to its success, of separating the functions of document delivery and display. Otherwise, the electronic text becomes a mere set of ‘‘pictures’’, and the Web, far from providing open, world wide access to the variety of platforms that such access implies, becomes a Balkanized region of minor potentates ruled by various commercial enterprises. We are thrust backwards into a situation not unlike the days of incompatible, proprietary operating systems. If we don’t revolt at being asked to use Acroread, then why not Microsoft Wordread, or Word Perfectread, or Framemakerread? It doesn’t take much imagination to see the shadows of the powerful proprietary software companies encroaching on the newfound freedom of the Web.

The complaints against HTML, whether from SGML purists or the commercial hawkers of portable document formats, are that it is at present theoretically and representationally confining. Both of these views seem to me to be shortsighted. Everyone who has used it probably has some personal gripe about what it *doesn’t* yet do, but ‘‘yet’’ is the operative word here. If we care to think back just a few years, what HTML *does* now do seems nothing short of amazing. We forget that the Web is very much a toddler, but one under active development. Its independence allows for the kind of development that we see in the progression from the CERN line browser to the latest Netscape beta. Browser development shows every sign of enhancing representation in the future. Side by side with this, the theoretical enhancements of HTML begin to approach the sort of complexity promised by SGML. In short, if we have a little patience, the Web will get better. The SGML people, instead of playing Greeks to HTML’s Romans, in effect looking down their noses at the younger, more vulgar, and more powerful cousin, might be better off accepting the reality of HTML’s continued presence, recognizing HTML as an ally well worth civilizing, and indeed incorporating at least the typesetting features of HTML into their DTDs. In the meantime, the thoughtful use of HTML provides us with the best opportunity to pursue the hitherto elusive goal of online document distribution and publication.

<Untitled>

Table of Contents