Weaving a Better Web

March 1998 / Cover Story / Weaving a Better Web

The features that made HTML so popular are causing the Web to fall apart. What's next?

Scott Mace, Udo Flohr, Rick Dobson, and Tony Graham

We have a love/hate relationship with HTML. We love its easy learning curve and universality, but we hate its easily broken links and limited formatting. We love its simple and compact syntax, but we hate its rigid formatting and inflexibility. To keep what we love and jettison what we hate, we've scripted it, styled it, tabled it, and framed it. Yet, after more face lifts and tummy tucks than an aging Hollywood star, today's HTML is still just HTML. The broken links and formatting problems are just warts and cellulite that won't go away.

It's time to find some new, fresh talent. Although you probably won't discover them in the corner soda shop, a few new stars are about to break onto the scene with names like Extensible Markup Language (XML), cascading style sheets (CSS), and Dynamic HTML (DHTML). Each works on a slightly different set of HTML 3.2's problems: XML on helping organize and find data, CSS on Web page inheritance and presentation, and DHTML on dynamic presentation of Web content. Aided by the recent HTML 4.0 refresh, these new technologies will beat back HTML's legacy of too many dead links, slow searches, and static pages on today's Internet and intranets.

The bad news: At the time of this writing, browsers are between generations, not yet fully ready to embrace these new technologies and standards. But this lag may be just what hatching standards need, giving developers enough time to rethink the way their Web applications should work before a rewoven Web hits with full force, starting at the end of this year.

Fixes on the Horizon

The fact that HTML has problems is hardly news (see the sidebar "What's Wrong with HTML"). Netscape, Microsoft, Macromedia, and a host of other companies have invested considerable effort in fixing the problems. We've all seen the results: proprietary HTML extensions, ActiveX controls, Java applets, and plug-ins that try to work around HTML's weaknesses. But these fixes all have problems of their own: They're proprietary, or they require users to install an application extension, or they're not completely supported by all browsers.

This year we'll start to see open, standard fixes to many of HTML's problems. XML, DHTML, style sheets, a document object model, and HTML 4.0 will create standard ways to get around most of the big problems we have with HTML today.

XML is probably the most notable. It's already a standard ratified by the World Wide Web Consortium (W3C), and it represents the largest departure for people used to writing standard HTML. XML, which defines document structures rather than how a browser should display a document, will give Web developers a lot more flexibility. It changes the way browsers display, organize, and search information. It could even make broken links a thing of the past. There are rumors that the next version of the Netscape browser (due this spring) will be XML-compliant. Netscape declined to comment. Microsoft has already built an XML-compliant application with the Channel Definition Format. Expect some major changes in the Web starting at the end of this year as sites start using XML.

It is important to note that HTML and XML are not competitors: They complement each other. Browsers will be able to process both, and future HTML standards will likely allow mixing HTML and XML in the same document.

For its part, DHTML aims to provide richer graphics and data with fewer, faster page downloads. In particular, it makes it easy to present information differently depending on user feedback. DHTML is currently undergoing some standards-body fighting as Microsoft and Netscape pitch their different flavors for ratification by the W3C.

Style sheets enable you to create pages that inherit properties from other pages. Currently, CSS goes hand-in-hand with HTML. It appears that XML, too, will have style sheets, specified using the Extensible Style Language (XSL).

W3C's Document Object Model (DOM), now a draft recommendation as part of the DHTML spec, will allow HTML and XML scripts, and other programs, to access structured data under program control. DOM also adds object orientation to page layout and design. For example, HTML elements appear as objects and collections that expose properties and methods. Developers can use DOM and a scripting language, such as JavaScript, JScript, or VBScript, to manipulate the DOM and achieve dynamic styles, content, and positioning. Scripts can manipulate positioning attributes to create animations on an HTML page.

DHTML, and to some extent DOM, have had a rougher birth than XML, with Microsoft and Netscape taking radically different tacks toward serving up dynamic content and defining Web elements as objects. Let's take a closer look at some of these technologies that promise to revolutionize the Web.

XML: Bigger Than HTML, Smaller Than SGML

HTML is based on Standard Generalized Markup Language (SGML), a much larger metalanguage that predates the Web. SGML specifies grammars for document markup languages, and SGML documents bring their grammar definition with them in the form of the Document Type Definition (DTD). DTD specifies tags used in the document and the meaning of those tags.

HTML is a single SGML application -- a hard-wired set of tags. HTML 3.2, for example, specifies about 70 tags and 50 attributes. Because HTML is a fixed, nonextensible grammar, HTML documents do not need to include the DTD. Its fixed nature makes HTML easy to learn and makes it easy to write HTML viewers. It also means that it can be very difficult to get HTML to do what you want.

The incredibly extensible SGML would fix that particular problem, but SGML is too cumbersome to learn and implement easily. Instead of bringing all of SGML to the Web, the World Wide Web Consortium has proposed a thinner version: XML. You can think of XML as a kind of SGML Lite, intended to bridge the gap between SGML's richness and HTML's ease of use in Web applications. XML is a metalanguage like SGML, but while changes to HTML require an update of the standard, XML is meant to be extended. As soon as an extension is specified within XML, it becomes universally available.

At the SGML/XML conference in Washington, D.C., last December, version 1.0 of the Extensible Markup Language specification was issued as a W3C Proposed Recommendation. Before that, W3C's XML Working Group, chaired by Jon Bosak of Sun Microsystems, had published several working drafts, edited by Tim Bray (Textuality/Netscape), Jean Paoli (Microsoft), and C. M. Sperberg-McQueen (University of Illinois at Chicago). Final ratification of XML 1.0 was due for late January.

XML was designed to be easier to use than SGML. As Richard Light writes in his book *Presenting XML*: XML offers "80% of the benefits of SGML for 20% of its complexity." The XML designers tried to leave out only those parts that are rarely used. That turns out to be quite a lot: The XML specification needs about 30 pages, compared to 500 for SGML. One objective of the XML Working Group is that experienced programmers should be able to develop an XML parser in a week. That said, XML is a verbose format compared to HTML, though compression features in newer versions of the Hypertext Transfer Protocol (HTTP) should ensure that XML documents download efficiently over networks.

What's the catch? XML is not compatible with today's HTML. For one thing, this means you'll need to upgrade your HTML browser to an XML browser. While SGML tools can handle XML (see the sidebar "New Tools for a New Web"), an XML tool will not be able to read all flavors of SGML -- and one of those is HTML. That's because XML uses a slightly different syntax than HTML and enforces syntax rules more rigorously. (See the sidebar "The Power of XML Syntax".) HTML documents will require changes, albeit minor ones, to become XML-compatible.

Why the changes? XML breaks the bounds of HTML's fixed set of tags, letting developers define an unlimited number of tags to describe any data element in a document. These data elements can be nested hierarchies of information, organized just as naturally as papers within file cabinets. A valid XML document is one in which these hierarchies are properly defined and nested.

Declaring these tags and hierarchies at the outset greatly reduces the amount of procedural code a developer has to write to create a structured application. The downside: Developers can't embed any XML tag in any order in documents. Furthermore, for the XML document to be valid, each new tag must be included in a DTD, which can be stored in a separate file.

(As a performance boost, a server can offer up an XML document without its DTD, in which case XML parsers can declare the document "well-formed" without having to refer to the DTD.) If the tags aren't embedded within each other properly, the parser declares the XML document invalid. All this validity checking is more work than HTML, but it yields greater rewards.

The benefits of reworking documents in XML are substantial. Because encoding Web content in XML makes the information's structure more accessible, it helps search engines return more meaningful results. (See the sidebar "XML Namespaces".) XML also introduces concepts that will ease maintenance and make Web applications more stable, including bi-directional and externally stored links. Web clients can be more intelligent and take over tasks that are currently handled by the server.

XML's Structure and Language Elements

Although XML has many parts, you really need to know about three in order to understand how it works: the Document Type Definition (DTD), XML's layout language; the Extensible Style Language (XSL), XML's version of style sheets; and the Extensible Link Language (XLL), a system for handling links beyond HTML's hard-coded, in-line hrefs.

DTD The Document Type Definition specifies the logical structure of a document. It enables you to define the grammar of a document, which, in turn, enables an XML parser to validate a page's use of its tags (see the sidebar "The Power of XML Syntax"). The DTD defines a page's elements and its attributes as well as the relationships among those elements and attributes. For example, the DTD can specify that a list item can occur only within a list.

Ideally, the definitions should be oriented toward describing the data structure associated with the application, rather than how the data should be displayed. In other words, define an element as a headline, and let the style sheets and scripts define how a headline should look. XML DTDs are getting a running start by leveraging the work done on DTDs for a range of applications for SGML. (See the sidebar "Applications Will Drive XML Acceptance").

DTDs aren't mandatory. For simple applications, developers need not build their own DTDs (which is no mean task); they can use predefined, public DTDs, or none at all. Even if a DTD exists for a document, the parser may choose not to check the document's validity against the DTD (as long as the document is well-formed). The server may have already done the check -- time and bandwidth will be saved.

XSL Extensible Style Language is the language used to specify style sheets for XML documents. XSL enables Web browsers to change the presentation of a document -- for example, the order in which data is displayed -- without further interaction with the server. By switching style sheets, the same document can be displayed in large print or Braille, collapsed to show just the outer hierarchical layers, or formatted for print. Imagine a technical manual that adapts to the learning curve of the user: It has styles for beginners and for the more advanced, all generated from the same text base. (Now you see why the DTD shouldn't control how the information is displayed.)

XSL can handle an unlimited number of tags, each in an unlimited number of ways, by virtue of its extensibility. It brings advanced layout features to the Web, such as rotated text, multiple columns, and independent regions. It supports international scripts, all the way to mixing left-to-right, right-to-left, and top-to-bottom scripts on a single page.

Much as XML takes the middle ground between HTML and SGML, the proposed XSL standard takes the middle ground between CSS and SGML's Document Style Semantics and Specification Language (DSSSL). DSSSL defines a full-featured model for formatting objects. Widespread implementation of DSSSL may have been impeded because it uses Scheme syntax, and because it is very complex. In comparison, CSS uses a simpler model (for example, it cannot reorder elements). The XSL proposal supports DSSSL flow objects and CSS objects, uses XML syntax and a declarative language, and provides an escape into ECMAScript for complicated tasks and to allow extensions. Mechanical mapping from CSS to XSL will be possible -- content developers need not learn the full language.

As a technology preview, Microsoft recently released two XSL processors: a command-line utility that produces HTML output from an XML document and an XSL style sheet, and an ActiveX control for displaying XML in a browser. The Microsoft XSL Processor runs on Windows 95 and Windows NT (x86 only) with Internet Explorer 4.0.

XSL is a bit behind the timetable for XML. ArborText, Inso, and Microsoft submitted a proposal for XML to the W3C in August 1997 as a note for discussion. The W3C is creating a separate XSL working group because completing XSL requires a different range of expertise than previous components of XML.

XLL XML's Extensible Link Language will support simple links as they exist on the Web today, but it will go on to implement extended links, including indirect links that can put an end to the dead links and the connector "|" that causes only the relevant part of an element to be retrieved from the server.

In the words of Jon Bosak, who chairs the XML Working Group, "HTML, this so-called 'hypertext markup language,' implements just a tiny amount of the functionality that has historically been associated with the concept of hypertext systems. Only the simplest form of linking is supported -- unidirectional links to hard-coded locations. This is a far cry from the systems that were built and proven during the 1970s and 1980s."

In a true hypertext system of the kind envisioned for the XML effort, Bosak explains, all the classic hypertext linking mechanisms will be supported:

- location-independent naming
- bidirectional links
- links that can be specified and managed outside of documents to which they apply
- n-ary hyperlinks (e.g., rings, multiple windows)
- aggregate links (multiple sources)
- transclusion (the link target document appears to be part of the link source document)
- attributes on links (link types)

These will be achieved through XLL, which is currently under development. As XML is based on SGML and XSL on DSSSL, XLL is basically a subset of HyTime

(Hypermedia/Time-based Structuring Language, ISO 10744). It also follows linking concepts specified by the Text Encoding Initiative.

Where Is XML Going?

Since development began in September 1996, XML has acquired an avalanche's momentum. Version 4 of Microsoft's Internet Explorer supports XML, and Netscape may have followed suit by the time you read this. Many other companies, including Adobe, ArborText, Sun, and Xerox, have announced their support. XML will no doubt become the vehicle for publishing SGML-based information on the Web.

Netscape has proposed combining the Meta Content Framework (MCF) with XML. Microsoft based its Channel Definition Format (CDF) on XML.

According to Bosak of the XML Working Group, the applications that will drive the acceptance of XML can be divided into four broad categories:

1. Applications that require the Web client to mediate between two or more heterogeneous databases.
2. Applications that attempt to distribute a significant proportion of the processing load from the Web server to the Web client.
3. Applications that require the Web client to present different views of the same data to different users.
4. Applications in which intelligent Web agents attempt to tailor information discovery to the needs of individual users.

One of the applications that falls into the first category is electronic commerce, particularly if based on Electronic Data Interchange (EDI). In this context, it comes in handy that the structure XML brings to Web data makes it easier to attach digital signatures, as well as to encrypt a document or parts of it. The W3C Digital Signature Initiative is working on XML-based security and authentication. In other applications, where automation and information reuse are required, XML will complement HTML. Whatever the future, the transition will be smooth and users will not have to suffer.

DHTML: HTML Gets Richer

XML, despite any technical advantages, is still new and different from HTML. Many Web developers are going to have problems migrating large sites to XML or training their staffers to work with this more sophisticated language. Wouldn't it be better to just extend HTML's capabilities while maintaining at least some of the familiar syntax? Netscape and Microsoft, with the 4.x releases of their respective browsers, introduced something that each called Dynamic HTML (DHTML). The concept: provide richer graphics and data with fewer, faster page downloads.

Three core benefits of DHTML include dynamic styles, content, and positioning. Dynamic styles enable developers to change the appearance of content without forcing users to download all the content again. Dynamic content lets developers change the text or images that appear on a page so that content can respond interactively to user mouse and keyboard behavior. Dynamic positioning lets page authors move text and images around a page either automatically or in response to user behavior.

Unfortunately for developers, Netscape and Microsoft implemented DHTML differently. The two different DHTML flavors deliver a mixed bag of benefits (see the table "Summary of DHTML Benefits"). Until the W3C releases a DHTML standard, we will likely continue to see few pages that take advantage of this capability.

DHTML's Four Parts

Web developers can combine four things to create dynamic Web pages: cascading style sheets (CSS), HTML 4.0, Document Object Model (DOM), and scripts.

HTML 4.0 In December 1997, the W3C issued a final specification for HTML 4.0. Its many enhancements include incremental display of large tables, scrollable tables with fixed headers, and better support for printing long tables. Enhancements to HTML forms focus on making them more flexible. A new Button tag enables forms to have more than just Submit and Reset buttons. An `accesskey` attribute provides keyboard shortcuts to form fields. An `accept` attribute for the Input tag permits authors to designate valid content. Character sets get a boost: The legitimate HTML 4.0 character set extends beyond the one for Western European languages while still maintaining HTML documents in conformance with SGML.

CSS Controlling the presentation of a document written in a language like XML or HTML, cascading style sheets allow more precise layout and formatting than HTML alone. A new version of CSS is on the horizon: W3C's draft statement for CSS2 at the time of this writing includes a chapter devoted to aural style sheets. Aural rendering of HTML documents will help sight-impaired users gain convenient access to Web content. It can also serve other contexts, such as in-car use, presentation over a home entertainment system, and teaching pronunciation of words.

CSS2's specification chapter on the visual rendering model describes relative and absolute positioning issues. These designate rules for the two-dimensional layout of content in an HTML document. A section within the chapter addresses stacking issues that define how to arrange content in a third dimension.

DOM The third major DHTML component that the W3C is creating specifications for is the Document Object Model, which will define a platform-independent programmatic interface to HTML documents. This interface will be able to manipulate the content, structure, and style of the document. With DOM, Web developers can introduce dynamic and interactive content into their Web pages without having to rely on a Web server to provide new content or to change how existing content displays. The W3C will provide DOM bindings for Java and ECMAScript. The DOMFAQ indicates an independent group will submit a COM interface to the DOM that will appear as a W3C Note. There are expectations that various firms will provide bindings for other languages, such as Perl, C++, and VBScript.

Scripts The fourth DHTML component is scripting, and W3C proposes to issue an initial binding of its DOM to ECMAScript. The European Computer Manufacturers Association (ECMA; http://www.ecma.ch/) issued an initial version of ECMAScript-262 in June 1997.

Another version is due in 1998. Microsoft reports that its version of JScript in Internet Explorer 4.0 is compliant with ECMAScript. Extra features in Microsoft's JScript, including COM support, do not violate its conformance with ECMAScript. Netscape's JavaScript 1.2 is not compliant with the current ECMAScript version. The next releases of the Netscape browser and ECMAScript will bring the two into conformance.

Browser Wars

So, those are the parts of DHTML. They seem simple enough, yet DHTML probably won't be standardized until well after you see XML as a standard part of your browser. Why? The browser wars: Netscape wants to do things its way, and Microsoft wants to do things its way.

In order for DHTML's components to interact successfully with one another, they must be compatible with one another. The browser must recognize both the HTML and the CSS syntax. DOM must expose HTML and cascading style sheet elements. The scripting language must recognize the browser as a host, and it must be able to respond to object events while it manipulates object and collection properties and invokes their methods.

Unfortunately, each company has its proprietary extensions to selected components. In general, Microsoft's implementation is more faithful to the current W3C recommendations, possibly because it released its 4.0 browser later than Netscape. Here are some examples.

Microsoft's approach to dynamic styles, positioning, and content exposes all HTML tags as elements and lets page authors directly manipulate their properties as well as allowing dynamic reflowing of the text and images on a page. Netscape's approach exposes fewer elements and relies heavily on layers of HTML content. Authors establish these content layers with layer tags or CSS positioning coordinates. By changing the properties of the layers with JavaScript, page authors can achieve dynamic effects after a page loads. Developers using Netscape's Visual JavaScript Pro can drag and drop HTML, Java, and JavaScript components from a component palette to a Web page. They can also drag third-party JavaBeans and CORBA services from the component palette. Developers can visually build event-based connections or bound property values for two components so they remain synchronized. The package supports Oracle, Informix, Sybase, or ODBC data sources using the included JavaScript components that leverage database connectivity in Netscape Enterprise Server 3.0. A custom property editor enables developers to build interactively a SQL statement for specifying a data extract.

Fonts are another area of contention. Downloadable fonts allow an author to determine the precise font family for text whether or not that font already resides on the browser's workstation. Netscape relies on a font definition file that links to a Web page of installed fonts from any source. Microsoft's approach extends CSS notation to reference font styles. It also incorporates support for Microsoft True Type fonts.

Data binding and multimedia effects are another problem. Only Microsoft's DHTML implementation supports these. Data binding makes it easy and fast for surfers to interact with a data cache in a page because filtering and sorting operations on a local cache do not require a round trip to the server. Also, page authors can apply filters and transitions through style sheets, in-line style attribute settings, and scripts. A set of 14 filters, such as Blur and

FlipH, can add multimedia effects to HTML content. Web developers can also invoke transitions for entering or leaving a URL.

The purpose of standards is to create a reference for the components so browser manufacturers can have a minimum set of requirements to meet. Then, at least for some core set of functions, DHTML code will behave consistently across browsers that are in conformance with the standards. As browser manufacturers strive to deliver the best value to their clients, we can expect a continuing stream of extensions to the standards that result in differences outside the core functions.

The two standards organizations, W3C and ECMA, are busy issuing DHTML component specifications. When these organizations finish their work, browser manufacturers will have an open set of specifications to which they can conform for cross-browser compatibility.

Call to Arms

HTML isn't dead, but it is suffering from its own success -- and every time you get a "404 URL not found" error message, you're suffering, too. In order to keep the Web growing and push its power into more applications, we need to start replacing simple HTML with more powerful alternatives. Perhaps the most powerful alternative within reach is XML 1.0, the recently ratified standard. Its power is that it forces developers to describe content rather than presentation.

Couple XML with style sheets (which *do* control presentation), scripting, and a document object model (which enables developers to change content without revisiting a server) and you have a solution to problems as diverse as too many AltaVista hits and poor performance. Although it will require developers and users to retool, the migration to XML must begin. The future of the Web depends on it.

Other XML Tools and Applications

Copernican Solutions XML Developer's Toolkit
Checks, validates, loads, and accesses XML documents.
Internet: http://www.copsol.com/products/xdk/XDK
Norbert's XML Parser
Used during development of XML; downloadable from NXP Web site.
Internet: http://www.edu.uniklu.ac.at/~nmikula/NXP/
Jade
Will be one of the first packages to support XSL; downloadable from Jade Web si
Internet: http://www.jclark.com
Silknet eService 98
Enterprise customer-service application will support XML during 1998
to integrate data from Vantive, Scopus, and Remedy customer-service
applications.
Internet: http://www.silknet.com/

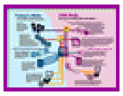
Where to Find

Adobe
San Jose, CA
Phone: 408-536-6000
Internet: http://www.adobe.com
ArborText
Ann Arbor, MI
Phone: 313-997-0200
Internet: http://www.arbortext.com
Grif
St. Quentin en Yvelines Cedex, France
Phone: +33 (0)1 30 12 14 30
Internet: http://www.grif.fr
Macromedia
San Francisco, CA
Phone: 415-252-2000
Internet: http://www.macromedia.com
Microsoft
Redmond, WA
Phone: 425-882-8080
Internet: http://www.microsoft.com/standards/xml
Netscape
Mountain View, CA
Phone: 650-937-2555
Internet: http://www.netscape.com
Information on products in the web category
HotBYTEs - information on products covered or advertised in BYTE

Summary of DHTML Benefits

Summary of DHTML Benefits			
Benefit	**Microsoft DHTML**	**Netscape DHTML**	**Description**
Dynamic styles	*	*	Change the appearance of styles on a Web page.
Dynamic content	*	*	Change the content on a Web page.
Dynamic positioning	*	*	Move the position of content on a Web page.
Font embedding	*	*	Download fonts with a Web page so content always displays in a specified font.
Data binding	*		HTML extensions that facilitate tight client-side integration with data sources. Current Microsoft approach relies heavily on ActiveX controls.
Filters and transitions	*		Filters and transitions to achieve low-level multimedia effects, such as fades, glows, drop shadows, and checkerboard transitions.
Key: * = yes			

Today's Web

illustration_link (81 Kbytes) 

Scott Mace is a senior editor at BYTE. He can be reached at scott.mace@byte.com. Udo Flohr is a BYTE contributing editor based in Hannover, Germany. You can reach him by sending e-mail to flohr@dfn.de. Rick Dobson, Ph.D., is president of CAB, Inc., a database and Internet development consultancy. You can reach him at his firm's Web site, http://www.cabinc.win.net. Tony Graham is an SGML consultant with Mulberry Technologies, Inc. (Rockville, MD) and the maintainer of the DSSSL users' mailing list. You can reach him at tgraham@mulberrytech.com.
