Introducing XML

Few inventions have revolutionized the way the world thinks and acts faster than the World Wide Web. Two decades ago, it did not exist, whereas today, there are few companies or individuals who have not been affected by it. Most Web pages are written in the Hypertext Markup Language, or HTML, which is an easy-to-use and easy-to-learn language used to describe the structure and formatting of a document. However, HTML does not give authors a way to describe the actual content of their pages. Often, this is not terribly important, but it can pose problems. Search engines, for example, do not know how to differentiate between the uses of common words on Web pages.

A Google search for the word *serenity* will bring up a mixture of results that include pages that discuss the 2005 movie of that name, sites that discuss products that happen to be called *Serenity,* and pages that just happen to use that word.

XML was created to help solve this issue. XML stands for *Extensible Markup Language.* Although XML looks and feels very much like HTML, it has a very different purpose. XML describes the data of a document instead of its visual appearance. Therefore, a Web page about movies written in XML can let search engines or other computers know that its use of *Serenity* is a reference to a film title.

XML Versus HTML

There is a common misconception that XML was invented to replace HTML. In fact, both languages were created from the same parent language, SGML, or *Standard Generalized Markup Language.* Neither is intended to replace the other. HTML, and its latest incarnation, XHTML, are still appropriate for most Web pages because they do not need to describe the data. XML is useful when the data on a site is more important than the appearance. Although many Web sites would benefit from being converted or rewritten as XML, the vast majority are better off as HTML.

It Is Just Text

XML files are plain text. This is one of the great things about XML. No special or expensive software is required to create or maintain XML files. The free text editor included with every operating system will work fine as an XML editor.

It is also important to understand that, as text files, XML cannot “do” anything. A lot of beginning developers expect their XML files to be able to perform advanced programming procedures and are disappointed when they find that they cannot. XML is used to describe the data in the document — and nothing more.

Extensibility

The “extensible” part of XML’s name is perhaps the most important. In HTML, there is a predefined set of tags to use. Unless you want to develop your own browser, you are limited to that set of tags. With XML, there is no predefined tag set. Instead, you, the developer, invent the tags that you want to use as you go along. Whatever tags work for your application are the ones that you can use.

XML documents are often compared to database tables. Although you have to be careful not to take the analogy too far, it is helpful to think of creating a database table when you create an XML file. Just as there is no book or Web site that sets out exactly what you can and cannot call the fields in a database, there is nothing that forces you to use any particular term or set of terms in your XML. This is why XML is sometimes referred to as a *meta language* — a language used to define other languages.

Introducing XSLT

One of the most important aspects of XML documents is that they maintain an absolute separation between the data and the display of that data. HTML is a presentational language, so experienced Web designers are used to thinking of what their page says and what it looks like at the same time; this separation of presentation and content can be difficult to grasp at first.

There are many advantages to separating content and presentation. The biggest of these is that by having the content described in one place and the presentation described elsewhere, developers can very easily repurpose existing documents. Take a situation where you have one set of data that needs to be displayed on the Web and in print. By separating the data from the presentation, one set of presentational rules can be applied for the Web and another for print, without having to update the underlying data at all.

XSLT, or Extensible Style sheet Language Transformations, was developed as a way to give developers a language to use to control the presentation of XML documents.

Advantages of XSLT

XSLT is very powerful. It has the ability to describe many programming concepts with which developers may be experienced, including looping and string matching.

XSLT is flexible. It can be used to generate XML, HTML, or plain-text documents from a single source XML file.

XSLT is XML-based, so it uses the same syntax as the source XML file.

XSLT files are plain text, so they can be created in the same editor being used for XML. Although there are fancy, expensive XSLT editors available, they are not required for development.

Disadvantages of XSLT

Like many powerful languages, XSLT can be difficult to learn at first.

Not all parsers support all aspects of XSLT, so there may be times when your document needs to be modified, possibly even including using nonstandard elements. This is particularly true with legacy systems.

Like XML, XSLT documents are ultimately plain-text files. Although you can build many advanced features into XSLT, you need some separate program to actually interpret the XSLT and do something with it.

Separate Content from Presentation

The concept in XML of separating the content of a document from the presentation is one of the most difficult concepts for modern developers to grasp. The computers we use today provide us with a rich visual environment in which to work. Older developers can recall the days of black screens filled with lines of green text, but few recall those days with anything approaching fondness, and younger developers have no more experience with it than they do driving a Model T.

Given that we are so used to creating documents in word-processing programs that show us exactly what the document will look like when printed, it is not surprising that most of us want our documents to look “pretty” from the very beginning. However, developing XML requires that the developer be more patient and develop the content of the document in a strict code view, while worrying about its visual presentation later.

Advantages of Separation

Keeping the content and presentation separate has many advantages. First, and most important, it allows the document to be easily repurposed. XML documents are designed to hold data. By keeping the presentation separate, you do not tie that data to one specific type of presentation. Therefore, a single XML document can be formatted to display in a word processor, on the Web, in a cell phone, or in a database application, all without needing to change the underlying XML code at all. Second, by keeping any presentational code out of the XML document, that document stays small. Smaller documents are easier to maintain and faster and easier to send electronically, either over the Web or via email.

Introducing XHTML

Even experienced Web designers are often afraid of XHTML, thinking that it is some new incarnation that will require hours of additional learning time. In fact, XHTML is nothing more than a newer version of HTML.

In creating XHTML documents, you use the exact same set of tags that you have always used in HTML 4.01. The only important difference is that XHTML uses a much stronger syntax than its predecessor did.

XHTML Syntax Rules

XHTML uses the XML syntax for writing Web pages. Most HTML 4.01 documents can be converted to XHTML with a few minor modifications.

Use Lowercase for All Tags

In XHTML, all tags must be written in all lowercase letters. This will possibly require the most work to convert a document from HTML 4.01 if that document used uppercase or mixed case for its tags. Care must be taken to ensure that all tags, both opening and closing, are converted to lowercase to create a valid XHTML document.

All Tags Must Be Properly Nested

Although never a specific requirement, it has always been a good idea to properly nest tags in HTML, so this should not be a problem now that it is actually required. Proper nesting simply means that closing tags must be presented in the opposite order from the opening tags, so you can use a “first in, last out” methodology.

All Attributes Must Have a Value, and the Value Must Be Quoted

There are a few attributes in HTML 4.01 that do not have a value but are instead a single word. An example is the `checked` attribute for the `input` tag in forms when creating check boxes and radio buttons. In XHTML, you must have a value, so instead of simply saying `checked` as the attribute, you now use `checked="checked"`. All attribute values must always be surrounded by quotation marks in XHTML, although you can use either single or double quotation marks.

All Tags Must Be Closed

This rule is the one that causes XHTML to look the most different from HTML 4.01 documents. In HTML 4.01, there are some elements, such as `hr`, `br`, `meta`, and `img`, that do not contain content and will thus have no corresponding closing tags. In XHTML, closing tags are required for all elements. Empty elements, such as those listed previously, can be expressly closed with a traditional closing tag, or you can use a shorthand syntax by adding a slash to the end of the tag: `<hr />`, `<br />`, `<meta />`, `<img />`.

You Must Provide a Valid DOCTYPE Declaration

The `DOCTYPE` declaration points to the DTD, or Document Type Definition, that defines the XHMTL tag set. There are three `DOCTYPE`s for XHTML: *transitional,* which enables you to use any tag from the HTML tag set; *frameset,* which enables you to use HTML frames and their assorted tags; and *strict,* which requires that you only use structure tags and attributes, keeping all your presentational code in cascading style sheets.

Introducing CSS

The Web was invented as a way for scientists to share information. Although scientific papers are of course very important, their writers focus primarily on the content of the page and do not generally worry so much about the visual appearance of the document. The original versions of HTML therefore contained little or no presentational markup. However, as soon as the Web became more generally popular, companies started wanting to present their content in a much more visually rich way. Because HTML did not provide for the ability to set things like colors and background images, early developers simply added their own HTML tags to achieve these looks. Over time, the language became a horrible mishmash of logical and presentational markup, much of it specific to one browser or another. There was a period of time in the mid-1990s where many Web designers would build two different versions of their page, so one would work correctly in Internet Explorer and the other in Netscape.

Cascading style sheets (CSS) were developed to solve these problems. By removing all the presentational code from HTML and replacing it with a language designed specifically for presentation, Web designers are now able to create much smaller and simpler Web pages that can work across many different browsers and operating systems. Their Web pages also load faster and can easily be adapted to work in other media, such as on cell phones and PDAs. They are also much more accessible to people with disabilities.

CSS Syntax

CSS uses a different syntax from HTML or XML. It is not tag-based. Instead, it consists of rules, which are made up of a selector and a declaration. The selector is the tag to which the rule should be applied. The declaration is made up of property/value pairs, with each property and value separated by a colon, and each pair separated by a semicolon. The entire declaration is enclosed within curly braces — for example:

```
p {color:#FF0000; font-size:95%; font-family:Arial,
Helvetica, sans-serif;}
```

CSS Versions

Currently, CSS exists in two widely supported versions. Version 1 provides for font properties such as font face, size, and emphasis; colors of text and backgrounds; text attributes, including word and line spacing; alignment of elements; margins, padding, and borders; and ID and class selectors. Version 2 added support for positioning of elements in a variety of ways; several new, but not widely supported, font properties such as shadows; and the support for different media types such as printers and projectors. A third version of CSS that will include many more powerful ways to specify selectors and several new features, such as transparency, has been in the works for nearly a decade.

Browser Support

Support for CSS in the browsers has long been a thorn in the side of developers. The good news is that most of the modern browsers come very close to fully supporting CSS1, and almost all of them support the most common features of CSS2. Mozilla's Firefox browser is perhaps the most standards-compliant browser currently available and supports almost all of CSS1 and CSS2 — and even a few scattered CSS3 properties. In late 2006, Microsoft released Internet Explorer 7, which has the best implementation of CSS of any browser ever released by that company. Although IE7 still has some odd quirks and bugs in its CSS support, it is far, far better than its predecessor. Apple's Safari, the third most commonly used browser, is also very good at supporting CSS1 and CSS2, although it too has some odd bugs here and there.

View XML in a Browser

Most modern Web browsers have the ability to parse XML, but only at a basic level. They will read the XML document and ensure that it is well-formed. If the document is not well-formed, the browsers will return an error, which specifies exactly what is wrong in the document and on which line the error occurred, although the line and actual error may not always be precisely accurate. If there are no errors, the browsers will display the document.

Because browsers are designed primarily for XHTML display, they will examine the XML document for a reference to a style sheet. If one is present, they will use it. Currently, modern browsers support CSS versions 1 and 2 and XSLT version 1. Older browsers, such as Internet Explorer 5 and 6, may be buggy in their display of the page when using CSS. There are even a few examples in which the browser will correctly display certain CSS properties when applied to XHTML but not when the same properties are applied to XML.

If the document is not attached to a style sheet, the browser will use its own internal XSLT style sheet to display the XML's document tree, a view that closely resembles the plain code view from the original editor. Both Mozilla Firefox and Microsoft Internet Explorer add extra functionality via DHTML to the view of XML, enabling you to expand and collapse tags to hide child tags, thus making browsing through documents much easier.

View XML in a Browser

IN INTERNET EXPLORER

1. Click File ➔ Open.

   The Open dialog box appears.

2. Click Browse.

3. Navigate to and click the XML file that you want to open.

   - You need to set the Files of Type drop-down list to All Files.

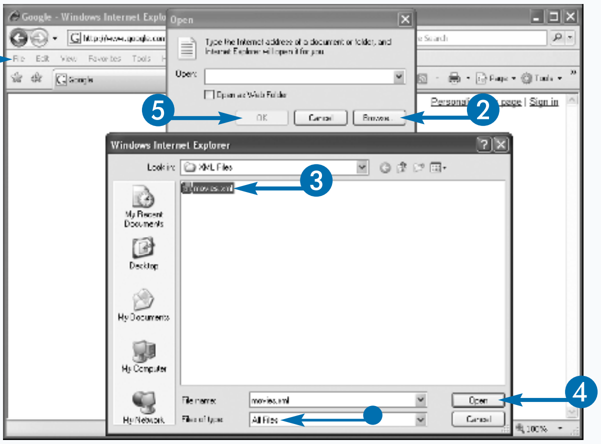
4. Click Open.

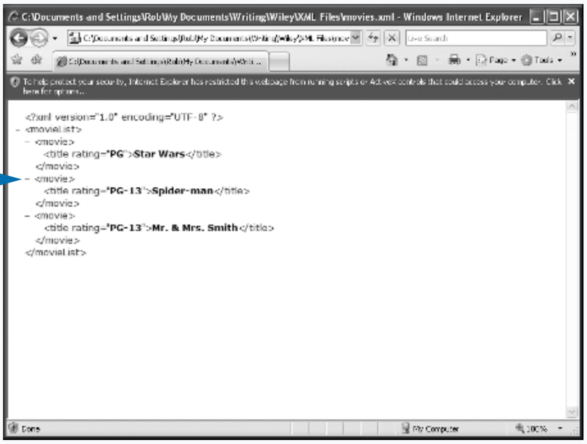
   All files in the directory are displayed.

5. Click OK.

   The file opens in the browser, displaying the document tree.

6. Click the minus sign next to a tag to collapse it or the plus sign to expand it.




6
C:\Documents and Settings\Rob\My Documents\Writing\Wiley\XML Files\movies.xml - Windows Internet Explorer
Live Search
Page
Tools
Done
My Computer
100%

To help protect your security, Internet Explorer has restricted this webpage from running scripts or ActiveX controls that could access your computer. Click here for options...

```
  <?xml version="1.0" encoding="UTF-8" ?>
- <movieList>
  - <movie>
      <title rating="PG">Star Wars</title>
    </movie>
  - <movie>
      <title rating="PG-13">Spider-man</title>
    </movie>
  - <movie>
      <title rating="PG-13">Mr. & Mrs. Smith</title>
    </movie>
  </movieList>
```

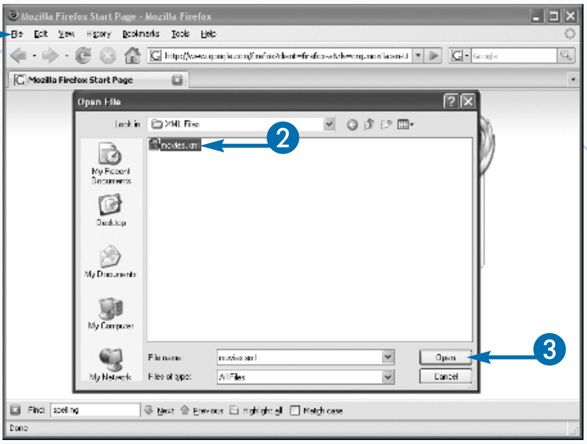
IN FIREFOX

1. Click File ➔ Open File.

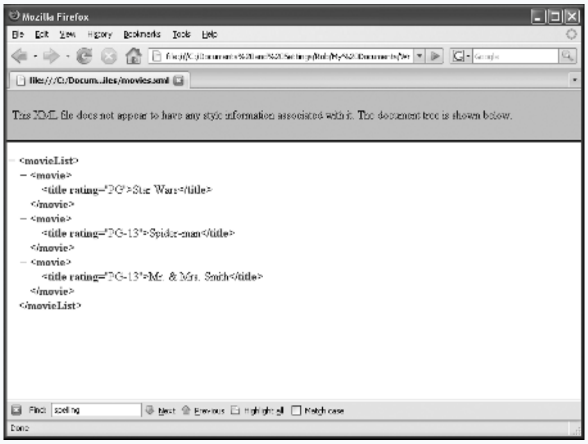
   The Open File dialog box appears.

2. Navigate to and click the XML file that you want to open.

3. Click Open.


1
2
3
Mozilla Firefox Start Page - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Mozilla Firefox Start Page
Open File
Look in: XML Files
movies.xml
My Recent Documents
Desktop
My Documents
My Computer
My Network
File name: movies.xml
Files of type: All Files
Open
Cancel
Find: spelling Next Previous Highlight all Match case
Done

The file opens in the browser, displaying the document tree.


Mozilla Firefox
File Edit View History Bookmarks Tools Help
file:///C:/Docum...iles/movies.xml
Find: spelling Next Previous Highlight all Match case
Done

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<movieList>
– <movie>
    <title rating="PG">Star Wars</title>
  </movie>
– <movie>
    <title rating="PG-13">Spider-man</title>
  </movie>
– <movie>
    <title rating="PG-13">Mr. & Mrs. Smith</title>
  </movie>
</movieList>
```

Extra

If a style sheet is provided, the browsers will render the XML document according to the style sheet. If you used CSS to style the XML, you will get a page that looks very much like a traditional Web page. If you used XSLT to transform the document to XHTML, you will see the resulting XHTML; if you transformed to XML, you will get the newly created XML document tree.

Neither Internet Explorer nor Firefox will validate the XML by itself. It is possible to download an extension to Internet Explorer, called the Internet Explorer Tools for Validating XML and Viewing XSLT Output. This free download, available from Microsoft's Web site, adds the ability to right-click an XML document and have Internet Explorer validate it against a schema or DTD. As of this writing, no similar extension exists for Firefox. There are several XML validators available for Firefox, but none are as simple to use as Microsoft's is for Internet Explorer. The same holds true for Safari and other browsers for the Macintosh.

Introducing the Anatomy of an XML Document

All XML documents follow the same basic structure. All well-formed XML documents are made up of an XML prolog, a series of optional parsing instructions, and the document itself. The document is made up of a series of nested tags.

Because all documents have this identical structure, any parser can correctly interpret the XML, even if the actual document contents vary greatly. This is a big part of what makes XML such a versatile and powerful language.

The XML Prolog

The XML prolog is a declaration that lets the parser know that this is an XML document, which version of XML is being used, and which character encoding set should be applied to the document. The XML prolog is not a tag, but rather a special instruction to the parser. Therefore, it has a slightly different syntax from the tags:

```
<?xml version="1.0" encoding="UTF-8" ?>
```

The question marks at the beginning and end of the prolog are what designates it as a parsing instruction instead of a tag.

Although there is currently only one version of XML in widespread use, by identifying it in all documents, you ensure that your XML 1 document will continue to be correctly parsed in the future, even if later versions of the language change radically. The encoding is the character set to be used on the document. UTF-8, the most common encoding set, has become a de facto standard for XML, HTML, email, and many other uses, as it allows your document to correctly display a wide variety of characters and symbols. XML supports other encoding standards as well.

Parsing Instructions

Following the XML prolog, you can optionally provide one or more parsing instructions. The most common parsing instruction is a link to a style sheet that uses either CSS or XSLT:

```
<?xml-stylesheet type="text/xsl" href="moviestyles.xsl" ?>
```

Like the prolog, this is not an XML tag, so we use the question marks at the beginning and end to designate it as a parsing instruction. The `type` attribute provides the MIME type encoding, which tells the parser what kind of document it should expect to see. In the case of both CSS and XSLT, you are using text documents, with the appropriate subtype. The `href` attribute provides the path to the style sheet. This can be either relative or absolute, and if absolute, can be given either as a full file path on a local machine or shared server or as a URL for a remote resource.

The Document

The document itself is the XML data that you plan to add. This can be just a few lines or many hundreds or even thousands of lines. The document will consist of nested XML tags. You must provide one root element whose tag wraps around every other element, so its closing tag will be the very last line of the document. Within the root, you can have as many parent/child relationships as you need for your document. Remember, with XML you get to make up the information as you go. There is no limit as to how many elements you can use or how complex the structure of your document should be.

Choose a Good Text Editor

XML documents are plain text, so they can be created in almost any program. However, there are many applications on the market that are better for editing XML than others.

Ultimately, the decision as to which editor you use will be based on personal choice; important factors include price and operating system availability.

Altova XMLSpy

XMLSpy from Altova is a full-featured commercial application. It is extremely powerful, especially when combined with the other products in the Altova XML suite, which include a program to visually create XSL documents and a program to help visually create schemas. XMLSpy can import XML data from a wide variety of external sources, can correctly read and interpret XSL and schema documents, and provides many other advanced features. It also includes a built-in browser, based on the Internet Explorer engine, for viewing XML documents.

XMLSpy's nicest feature, however, is perhaps its code editor. XMLSpy will automatically create closing tags, and if you attach the XML document to a schema, it will even provide code hinting as to the appropriate tags and attributes to use in your document. It provides for code coloring to help you know when you have made a mistake and line numbers to help find errors.

The biggest downside to XMLSpy is its cost. It is perhaps the most expensive XML editor on the market and can be almost cost-prohibitive in its Enterprise version. Altova does provide for a free 30-day trial of the product.

Microsoft XML Notepad 2007

Microsoft released this free application early in 2007 to provide developers with a simple, friendly method of creating XML. Whereas its small size makes for a quick download and it is certainly easy to use, it lacks a code view editor, instead forcing developers to create their XML in a visual environment.

Eclipse

Eclipse is an open-source Java editor. Its open architecture has allowed it to be reconfigured to serve other purposes, including an XML plug-in that gives it many advanced XML features. The main program and the XML plug-in are available for free and work on practically any platform. In fact, the program does not even need to be installed; it can be run directly off a flash drive. Although the plug-in lacks many features, it is under constant development, with new features and editions being released regularly.

Exchanger

The Exchanger XML editor is a cross-platform, feature-rich editor. There is a free version available for noncommercial use and a professional edition available for purchase. Some of its key features include support for XML schemas, DTDs, and RELAX NG editing, the ability to search on XPath and regular expression strings, a visual data-grid view, and a powerful XSLT debugger.

Adobe Dreamweaver CS3

Adobe Dreamweaver CS3, available for both Windows and Macintosh systems, is not, strictly speaking, an XML editor, but it has a fantastic code editor that includes many of the same features as XMLSpy, such as code coloring, tag completion, and line numbers. In its Design view, Dreamweaver also enables developers to visually create styles, using either CSS or XSL. However, because its focus is on Web design, you can only create XSL documents that transform to XHTML. Also, Dreamweaver has no support for schemas at all.

Oxygen

Oxygen is an XML and XSLT editor available for both Windows and Macintosh systems and is very affordable. Its editor supports XML, XSLT, XML schemas, and RELAX NG. It includes an XSLT debugger and also contains a Subversion client to assist in version control and tracking when working in a group setting.

Notepad

Although not technically an XML editor, Notepad, the free text editor available on all copies of Microsoft Windows, will work for creating XML because it does not attempt to add any of its own markup. Notepad does not provide any additional tools, so you are completely on your own to type everything. It also cannot parse the document, so you will need to use a browser or other parser to check your documents for well-formedness.